Bamshad Mobasher
Sarabjot Singh Anand (Eds.)

Intelligent Techniques for Web Personalization

LNAI 3169

IJCAI 2003 Workshop, ITWP 2003
Acapulco, Mexico, August 2003
Revised Selected Papers

Springer

Lecture Notes in Artificial Intelligence 3169

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Bamshad Mobasher   Sarabjot Singh Anand (Eds.)

Intelligent Techniques for Web Personalization

IJCAI 2003 Workshop, ITWP 2003
Acapulco, Mexico, August 11, 2003
Revised Selected Papers

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Bamshad Mobasher
DePaul University, Center for Web Intelligence
School of Computer Science, Telecommunication and Information Systems
Chicago, Illinois, USA
E-mail: mobasher@cs.depaul.edu

Sarabjot Singh Anand
University of Warwick, Department of Computer Science
Coventry CV4 7AL, UK
E-mail: s.s.anand@warwick.ac.uk

Library of Congress Control Number: 2005935451

CR Subject Classification (1998): I.2.11, K.4.1, K.4.4, C.2, H.3.4-5, H.5.3, I.2

ISSN 0302-9743
ISBN-10 3-540-29846-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-29846-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11577935 06/3142 5 4 3 2 1 0

Preface

Web personalization can be defined as any set of actions that can tailor the Web experience to a particular user or set of users. The experience can be something as casual as browsing a Web site or as (economically) significant as trading stock or purchasing a car. The actions can range from simply making the presentation more pleasing to anticipating the needs of a user and providing customized and relevant information. To achieve effective personalization, organizations must rely on all available data, including the usage and click-stream data (reflecting user behavior), the site content, the site structure, domain knowledge, user demographics and profiles. In addition, efficient and intelligent techniques are needed to mine these data for actionable knowledge, and to effectively use the discovered knowledge to enhance the users' Web experience. These techniques must address important challenges emanating from the size and the heterogeneity of the data, and the dynamic nature of user interactions with the Web.

E-commerce and Web information systems are rich sources of difficult problems and challenges for AI researchers. These challenges include the scalability of the personalization solutions, data integration, and successful integration of techniques from machine learning, information retrieval and filtering, databases, agent architectures, knowledge representation, data mining, text mining, statistics, user modelling and human–computer interaction. Throughout the history of the Web, AI has continued to play an essential role in the development of Web-based information systems, and now it is believed that personalization will prove to be the "killer-app" for AI.

The collection of papers in this volume include extended versions of some of the papers presented at the ITWP 2003 workshop as well as a number of invited chapters by leading researchers in the field of intelligent techniques for web personalization. The first chapter in the book provides a broad overview of the topic and a comprehensive bibliography of research into Web personalization that has been carried out in the past decade. The rest of the chapters are arranged in five parts each addressing a different aspect of the topic. Part I consists of three chapters focussed on user modelling. In the first of these chapters, Craig Miller describes the current state of our understanding of how users navigate the Web and the challenges in modelling this behavior. Further, the necessary capabilities of a working cognitive model of Web navigation by a user, an implementation of such a model and its evaluation are described. Next, Naren Ramakrishnan describes his view of personalization based on capturing the interactional aspects underlying a user's interaction with the Web in an attempt to model what it means for a website to be personable. The final chapter in this part of the book, by Bettina Berendt and Max Teltzrow, rather than modelling the user per se, discusses results from a user study aimed at understanding the privacy concerns of users and the effect of these concerns on current personalization strategies. They argue for improved communication of privacy practice and benefits to the users resulting from data disclosure and a better understanding of the effect of various types of data on the performance of the resulting personalization.

The second part of the book consists of three chapters on recommender systems. In the first of these chapters Fabiana Lorenzi and Francesco Ricci provide a survey of case-based approaches to recommendation generation and propose a unifying framework to model case-based recommender systems. In the following chapter Lorraine McGinty and Barry Smyth describe a novel approach to item selection, known as adaptive selection, that balances similarity and diversity during a user interaction with a reactive recommender system. They show how adaptive selection can dramatically improve recommendation efficiency when compared with standard forms of critiquing. Finally, Robin Burke surveys the landscape of possible hybrid systems for personalization, describing several ways in which base recommenders can be combined to form hybrid systems.

The third part of the book consists of three chapters on enabling technologies. The first of these, by Chuck Lam, introduces the use of associative neural networks for user-based as well as item-based collaborative filtering. It also discusses the use of principal component analysis for dimensionality reduction. In the next chapter Tiffany Tang et al. propose the use of heuristics to limit the size of the candidate item set, hence improving the performance of traditional user-based collaborative filtering. Finally, Birgit Hay et al. propose a new algorithm for mining interesting Web navigational patterns that can be used for personalizing future interactions.

The fourth part of the book consists of three chapters on personalized information access. The first of these chapters, by Kevin Keenoy and Mark Levene, surveys the current state of the art in personalized Web search. Apostolos Kritikopoulos and Martha Sideri follow this with a chapter describing an approach to personalizing search engine results using Web communities. Finally Tingshao Shu et al. present an approach to predicting a user's current information needs using the content of pages visited and actions performed.

The final part of the book consists of four chapters on systems and applications. The first chapter in this part, by Barry Smyth et al., describes the application of personalized navigation to mobile portals to improve usability. Next, Magdalini Eirinaki et al. present their system for personalization based on content structures and user behavior. Arif Tumer et al. then present a privacy framework for user agents to negotiate the level of disclosure of personal information on behalf of the user with Web services. Finally, Samir Aknine et al. present a multi-agent system for protecting Web surfers from racist content.

August 2005

Bamshad Mobasher
Sarabjot Singh Anand

Table of Contents

Intelligent Techniques for Web Personalization
Sarabjot Singh Anand, Bamshad Mobasher 1

User Modelling

Modeling Web Navigation: Methods and Challenges
Craig S. Miller 37

The Traits of the Personable
Naren Ramakrishnan 53

Addressing Users' Privacy Concerns for Improving Personalization Quality: Towards an Integration of User Studies and Algorithm Evaluation
Bettina Berendt, Maximilian Teltzrow 69

Recommender Systems

Case-Based Recommender Systems: A Unifying View
Fabiana Lorenzi, Francesco Ricci 89

Improving the Performance of Recommender Systems That Use Critiquing
Lorraine McGinty, Barry Smyth 114

Hybrid Systems for Personalized Recommendations
Robin Burke 133

Enabling Technologies

Collaborative Filtering Using Associative Neural Memory
Chuck P. Lam 153

Scaling Down Candidate Sets Based on the Temporal Feature of Items for Improved Hybrid Recommendations
Tiffany Ya Tang, Pinata Winoto, Keith C.C. Chan 169

Discovering Interesting Navigations on a Web Site Using SAM^{I}
Birgit Hay, Geert Wets, Koen Vanhoof 187

Personalized Information Access

Personalisation of Web Search
Kevin Keenoy, Mark Levene 201

The Compass Filter: Search Engine Result Personalization Using Web Communities
Apostolos Kritikopoulos, Martha Sideri 229

Predicting Web Information Content
Tingshao Zhu, Russ Greiner, Gerald Häubl, Bob Price 241

Systems and Applications

Mobile Portal Personalization: Tools and Techniques
Barry Smyth, Kevin McCarthy, James Reilly 255

IKUM: An Integrated Web Personalization Platform Based on Content Structures and User Behavior
Magdalini Eirinaki, Joannis Vlachakis, Sarabjot Singh Anand 272

A Semantic-Based User Privacy Protection Framework for Web Services
Arif Tumer, Asuman Dogac, I. Hakki Toroslu 289

Web Personalisation for Users Protection: A Multi-agent Method
Samir Aknine, Aurélien Slodzian, Ghislain Quenum 306

Author Index 325

Intelligent Techniques for Web Personalization

Sarabjot Singh Anand[1] and Bamshad Mobasher[2]

[1] Department of Computer Science, University of Warwick, Coventry CV4 7AL, UK
s.s.anand@warwick.ac.uk

[2] Center for Web Intelligence, School of Computer Science, Telecommunications and Information Systems, DePaul University, Chicago, Illinois, USA
mobasher@cs.depaul.edu

Abstract. In this chapter we provide a comprehensive overview of the topic of Intelligent Techniques for Web Personalization. Web Personalization is viewed as an application of data mining and machine learning techniques to build models of user behaviour that can be applied to the task of predicting user needs and adapting future interactions with the ultimate goal of improved user satisfaction. This chapter survey's the state-of-the-art in Web personalization. We start by providing a description of the personalization process and a classification of the current approaches to Web personalization. We discuss the various sources of data available to personalization systems, the modelling approaches employed and the current approaches to evaluating these systems. A number of challenges faced by researchers developing these systems are described as are solutions to these challenges proposed in literature. The chapter concludes with a discussion on the open challenges that must be addressed by the research community if this technology is to make a positive impact on user satisfaction with the Web.

1 Introduction

The term *information overload* is almost synonymous with the Internet, referring to the sheer volume of information that exists in electronic format on the Internet and the inability of humans to consume it. The freedom to express oneself through publishing content to the Web has a number of advantages, however, the task of the consumer of this content is made more difficult not only due to the need to assess the relevance of the information to the task at hand but also due to the need to assess the reliability and trustworthiness of the information available.

Information retrieval technologies have matured in the last decade and search engines do a good job of indexing content available on the Internet and making it available to users, if the user knows exactly what he is looking for but often, search engines themselves can return more information than the user could possibly process. Also, most widely used search engines use only the content of Web documents and their link structures to assess the relevance of the document to the user's query. Hence, no matter who the user of the search engine is, if the same query is provided as input to the search engine, the results returned will be exactly the same.

The need to provide users with information tailored to their needs led to the development of various information filtering techniques that built profiles of users and attempted to filter large data streams, presenting the user with only those items that it believes to be of interest to the user.

The goal of personalization is to provide users with what they want or need without requiring them to ask for it explicitly [1]. This does not in any way imply a fully-automated process, instead it encompasses scenarios where the user is not able to fully express exactly what the are looking for but in interacting with an intelligent system can lead them to items of interest.

Intelligent Techniques for Web Personalization is about leveraging all available information about users of the Web to deliver a personal experience. The "intelligence" of these techniques is at various levels ranging from the generation of useful, actionable knowledge through to the inferences made using this knowledge and available domain knowledge at the time of generating the personalized experience for the user. As such, this process of personalization can be viewed as an application of data mining and hence requiring support for all the phases of a typical data mining cycle [2] including data collection, pre-processing, pattern discovery and evaluation, in an off-line mode, and finally the deployment of the knowledge in real-time to mediate between the user and the Web.

In this chapter we provide an overview of the topic of Intelligent Techniques for Web Personalization. In Section 2 we describe the process of personalization in terms of an application of a data mining to the Web. Section 3 provides a classification of approaches to Web personalization while in Section 4 we describe the data available for mining in the Web domain, specifically for the generation of user models. Section 5 describes the various techniques used in generating a personalized Web experience for users highlighting the advantages and disadvantages associated with each approach. Issues associated with current approaches to Web personalization are discussed in Section 6. The important issue of evaluating Web personalization is discussed in Section 7. Finally the chapter concludes in Section 8 with a discussion on the current state and future direction of research in Web personalization.

2 The Personalization Process

Personalization aims to provide users with what they need without requiring them to ask for it explicitly. This means that a personalization system must somehow infer what the user requires based on either previous or current interactions with the user. This in itself assumes that the system somehow obtains information on the user and infers what his needs are based on this information.

In the context of this book, we focus on personalization of the Web or more generally, any repository of objects (items) browseable either through navigation of links between the objects or through search. Hence, the domain we address includes Intranets and the Internet as well as product/service catalogues. More formally, we assume that we are given a universe of n items, $I = \{i_j : 1 \leq j \leq n\}$, and a set of m users, $U = \{u_k : 1 \leq k \leq m\}$, that have shown an interest, in the past, in a subset of the universe of items. Additionally, each user, u_k, may be described as a t-dimensional vector $(a_1^k, a_2^k, ..., a_t^k)$ and each item, i_j, by an s-dimensional vector $(b_1^j, b_2^j, ..., b_s^j)$. Further domain knowledge about the items, for example, in the form of an ontology, may also be available. We will assume the existence of a function $r_{u_k} : I \rightarrow [0,1] \cup \perp$ where $i_j = \perp$ signifies that the item i_j has not been rated by the user, u_k [1] that assigns a rating to each item in I. Let $I_k^{(u)}$ be the set of items currently unrated by the user u_k, i.e. $I_k^{(u)} = \{i_j : i_j \in I \wedge r_{u_k}(i_j) = \perp\}$. Similarly let $I_k^{(r)}$ be the set of items rated by the user u_k, i.e. $I_k^{(r)} = I - I_k^{(u)}$.

The goal of personalization is to recommend items, i_j, to a user u_a, referred to as the *active user*, where $i_j \in I_a^{(u)}$ that would be of interest to the user.

Central to any system capable of achieving this would be a user-centric data model. This data may be collected implicitly or explicitly but in either case must be attributable to a specific user. While this seems obvious, on the Web it is not always straightforward to associate, especially implicitly collected data with a user. For example, server logs provide a rich albeit noisy source of data from which implicit measures of user interest may be derived. Due to the stateless nature of the Web, a number of heuristics must be used along with technologies such as cookies to identify return visitors and attribute a sequence of behaviours to a single user visit/transaction [3].

Once the data has been cleansed and stored within a user-centric model, analysis of the data can be carried out with the aim of building a user model that can be used for predicting future interests of the user. The exact representation of this user model differs based on the approach taken to achieve personalization and the granularity of the information available. The task of learning the model would therefore differ in complexity based on the expressiveness of the user profile representation chosen and the data available. For example, the profile may be represented as vector of 2-tuples $u_k^{(n)}(< i_1, r_{u_k}(i_1) >, < i_2, r_{u_k}(i_2) >, < i_3, r_{u_k}(i_3) > .... < i_n, r_{u_k}(i_n) >)$ where i_j's $\in I$ and r_{u_k} is the rating function for user u_k. In the presence of a domain ontology, the user profile may actually reflect the structure of the domain [4], [5], [6]. Recently, there has been a lot of research interest in generating aggregate usage profiles rather than individual user profiles [7], that represent group behaviour as opposed to the behaviour of a single user. The distinction between individual and aggregate profiles for personalization is akin to the distinction between lazy and eager learning in machine learning.

The next stage of the process is the evaluation of the profiles/knowledge generated. The aim of this stage is to evaluate how effective the discovered knowledge is in predicting user interest. Common metrics used during this phase are coverage, mean absolute error and ROC sensitivity. See Section 7 for a more detailed discussion on evaluation metrics.

The deployment stage follows evaluation, where the knowledge generated and evaluated within the previous two stages of the process is deployed to generate recommendations in real-time as the users navigate the Web site. The key challenge at this stage is scalability with respect to the number of concurrent users using the system.

An essential, though often overlooked, part of the personalization process is the monitoring of the personalization. Anand et al. suggest that the success of the person-

[1] Note that a while we assume a continuous scale for rating, a number of recommender systems use a discrete scale. However, our formalisation incorporates this case as a simple linear transformation can be performed on the scale to the [0,1] interval.

alization should be based on lift in business process based metrics [8]. Other than just monitoring the effectiveness of the knowledge currently deployed, an essential aspect of monitoring the effect of personalization is profile maintenance. User interests are dynamic and their evolution must be detected and adapted to for effective personalization to take place. Additionally, personalization itself can influence user behaviour. Techniques for identifying this change and adapting the personalization system to it are not well understood, requiring further research.

In terms of the learning task, personalization can be viewed as a

- Prediction Task: A model must be built to predict ratings for items not currently rated by the user. Depending on whether the user ratings are numeric or discrete, the learning task can be viewed as a being one of regression or classification.
- Selection Task: A model must be built that selects the N most relevant items for a user that the user has not already rated. While this task can be viewed as one of post processing the list of predictions for items generated by a prediction model, the method of evaluating a selection based personalization strategy would be very different from that of a prediction based strategy (see Section 7).

3 Classifications of Approaches to Personalization

In this section we discuss various dimensions along which personalization systems can be classified based on the data they utilize, the learning paradigm used, the location of the personalization and the process that the interaction takes with the user.

3.1 Individual Vs Collaborative

The term personalization impresses upon the individuality of users and the need for systems to adapt their interfaces to the needs of the user. This requires data collected on interactions of users with the system to be modelled in a user-centric fashion. Typically, data is collected by the business with which the user is interacting and hence the business has access to data associated with all its customers.

A personalization system may choose to build an individual model of user likes and dislikes and use this profile to predict/tailor future interactions with that user. This approach commonly requires content descriptions of items to be available and are often referred to as *content-based filtering systems*. NewsWeeder [9] is an example of such a system that automatically learns user profiles for netnews filtering. In the case of NewsWeeder the user provides active feedback by rating articles on a scale of 1 to 5. The process of building a profile for a user requires the transformation of each article into a bag or words representation, with each token being assigned a weight using some learning method such as *tfidf* [10] or minimum description length [11]. The profile is then used to recommend articles to the user.

An alternative approach to recommendation is to not only use the profile for the active user but also other users with similar preferences, referred to as the active user's neighbourhood, when recommending items. This approach is referred to as *social or collaborative filtering*. An example of such a system is GroupLens, also aimed at recommending netnews articles [12]. GroupLens defines a user profile as an n-dimensional vector, where n is the number of netnews articles. If an articles has been rated by the user, its corresponding element in the vector contains the rating. Note that as opposed to content-based filtering, the actual content descriptions of the articles is not part of the profile. Articles not currently rated by the active user but rated highly by users in the neighbourhood of the active user are candidates for recommendation to the active user. While GroupLens only uses rating data, collaborative approaches that utilise both content and user rating data have also been proposed [13], [14].

A major disadvantages of approaches based on an individual profile include the lack of serendipity as recommendations are very focused on the users previous interests. Also, the system depends on the availability of content descriptions of the items being recommended. On the other hand the advantage of this approach is that it can be implemented on the client side, resulting in reduced worries for the user regarding privacy and improved (multi-site) data collection for implicit user preference elicitation.

The collaborative approach also suffers from a number of disadvantages, not least the reliance on the availability of ratings for any item prior to it being recommendable, often referred to as the new item rating problem. Also, a new user needs to rate a number of items before he can start to obtain useful recommendations from the system, referred to as the new user problem. These issues along with others such as sparseness are discussed in greater detail in Section 6.

3.2 Reactive Vs Proactive

Reactive approaches view personalization as a conversational process that requires explicit interactions with the user either in the form of queries or feedback that is incorporated into the recommendation process, refining the search for the item of interest to the user. Most reactive systems for personalization have their origins in case-based reasoning research [15], [16], [17]. Reactive systems can be further classified based on the types of feedback they expect from the user. Common feedback mechanisms used by these systems include value elicitation, critiquing/tweaking [17], rating and preference feedback [18]. Value elicitation and tweaking/critiquing are feature based approaches to feedback. While in value elicitation the user must provide a rating for each feature of each recommendation object presented to the user, based on its suitability to the users needs, in tweaking/critiquing the user only provides directional feedback (for example, "too high", "too low") on feature values for the recommended object. Rating and preference are feedback approaches at the object level. In rating based feedback, the user must rate all the recommendations presented to him, based on their 'fit' with his requirements. In preference feedback the user is provided with a list of recommendations and is required to choose one of the recommendations that best suits his requirement. The system then uses this feedback to present the user with other, similar objects. The iterations continue until the user finds an object of interest or abandons the search. Examples of such recommender systems include Entree [19], DIETORECS [20] and ExpertClerk [21]. For a more detailed discussion on these feedback mechanisms see [16], [17].

Proactive approaches on the other hand learn user preferences and provide recommendations based on the learned information, not necessarily requiring the user to provide explicit feedback to the system to drive the current recommendation process. Proactive systems provide users with recommendations, which the user may choose to select or ignore. The users feedback is not central to the recommendation process as is the case in reactive systems. Examples of proactive systems include the recommendation engine at Amazon.com [22] and CDNOW, Web mining based systems such as [23], [24], [25], GroupLens [26], MovieLens [27] and Ringo [28].

3.3 User Vs Item Information

Personalization systems vary in the information they use to generate recommendations. Typically, the information utilized by these systems include:

- Item Related Information: This includes content descriptions of the items being recommended and a product/ domain ontology
- User Related Information: This includes past preference ratings and behaviour of the user, and user demographics

Systems that use item related information generally deal with unstructured data related to the items [29], [9]. Once this data has been processed, into relational form such as a bag-of-words representation commonly used for textual data, a user profile is generated. The profile itself may be individual as in the case of NewsWeeder [9] or based on group behaviour [13].

Most systems that use user related information, tend to be based on past user behaviour such as the items they have bought or rated (implicitly or explicitly) in the past. Fewer systems use demographic data within the recommendation process. This is due to the fact that such data is more difficult to collect on the Web and, when collected, tends to be of poor quality. Also, recommendations purely based on demographic data have been shown to be less accurate than those based on the item content and user behaviour [30]. In his study of recommender systems, Pazzani collected demographic data from the home pages of the users rather than adding the additional burden on the user to provide data specifically for the system. Such data collection outside of a controlled environment would be fraught with difficulties. In Lifestyle Finder [31], externally procured demographic data (Claritas's PRIZM) was used to enhance demographic attributes obtained from the user, through an iterative process where the system only requests information pertinent to classifying the user into one of 62 demographic clusters defined within the PRIZM classification. Once classified, objects most relevant to that demographic cluster are recommended to the user.

In addition to systems that depend solely on item related or user related information, a number of hybrid systems have been developed that use both types of information. Section 5.4 discusses these systems in greater detail. An example of such a system is the bibliographic system proposed by Haase et al. [5]. In addition to data on user behaviour, two domain ontologies are also available to the system describing the content of the items in a more structured form than that used by NewsWeeder. Hasse et al. define a user model based on user expertise, recent queries, recent relevant results (implicitly obtained by user actions on previous recommendations), a vector of weights for content features and a similarity threshold.

3.4 Memory Based Vs Model Based

As described in Section 2, the process of personalization consists of an offline and online stage. The key tasks during the offline stage are the collection and processing of data pertaining to user interests and the learning of a user profile from the data collected. Learning from data can be classified into memory based (also known as lazy) learning and model based (or eager) learning based on whether it generalizes beyond the training data when presented with a query instance (online) or prior to that (offline).

Traditional Collaborative filtering (see Section 5.2) and content based filtering based systems (see Section 5.1) that use lazy learning algorithms [32], [33] are examples of the memory-based approach to personalization, while item-based and other collaborative filtering approaches that learn models prior to deployment (see Section 5.3) are examples of model-based personalization systems.

As memory based systems simply memorise all the data and generalize from it at the point of generating recommendations, they are more susceptible to scalability issues. Section 6.3 discusses some of the solutions proposed in literature to address the scalability of memory based personalization systems. As the computationally expensive learning occurs offline for model-based systems, they generally tend to scale better than memory based systems during the online deployment stage. On the other hand, as more data is collected, memory based systems are generally better at adapting to changes in user interests compared to model based techniques that must either be incremental or be rebuilt to account for the new data.

Memory based systems generally represent a user profile using a vector representation though more expressive representations such as associative networks [34] and ontological profiles [35] have also been proposed.

3.5 Client Side Vs Server Side

Approaches to personalization can be classified based on whether these approaches have been developed to run on the client side or on the server-side. The key distinction between these personalization approaches is the breadth of data that are available to the personalization system. On the client side, data is only available about the individual user and hence the only approach possible on the client side is *Individual.*

On the server side, the business has the ability to collect data on all its visitors and hence both Individual and Collaborative approaches can be applied. On the other hand, server side approaches generally only have access to interactions of users with content on their Web site while client side approaches can access data on the individuals interactions with multiple Web sites.

Given these characteristics, most client side applications are aimed at personalized search applicable across multiple repositories [36], [37]. The lack of common domain ontologies across Web sites, unstructured nature of the Web and the sparseness of available behavioral data currently reduce the possibilities for personalization of navigational as opposed to search based interactions with the Web.

4 Data

Explicit data collection has typically been modelled as ratings of items, personal demographics and preference (including utility) data. Preference data refers to information that the user provides that can help the system discern which items would be useful to the user. When declared explicitly it can take the form of keywords/product categories (e.g. genres in movie/music databases) or values for certain attributes that describe the objects (e.g. cotton as the preferred material in an apparel store). Utility data refers to information regarding how the user would measure the fit of the objects recommended with his requirements. For example, if two suppliers for the same product exist, with supplier A providing the product at a premium rate over supplier B but with the advantage of free insurance for a predefined period, different users will have different thresholds for the extra cost of purchasing the product from supplier A [38], [39]. We refer to data that defines these preferences as utility data. Rating data may take the form of a discrete numeric value or an unstructured textual form such as reviews of products. While using numeric values is computationally easier to leverage, they are also less reliable as users associate these discrete values subjectively, for example, three stars according to one user may be equivalent to two stars for another user.

Implicit data collection refers to any data that can be collected on the user unobtrusively by "watching" their interaction with the system. Once again the objective is to obtain ratings from various discernable actions of the user. The actions and the associated inferences are dependent on the type of system being personalized. For example, in the Web domain in general, the linger time [2] is taken to be an implicit indicator of interest in the object [26]. Additionally, in an e-commerce context, actions such as adding an item to the basket, purchasing an item, deleting an item from the basket can all imply differing levels of interest in the item [40] as could bookmarking of pages [41], visit frequency, following/passing over a link and saving a page on a news/content site [42]. Claypool et al. [43] evaluated a number of possible implicit interest indicators and concluded that linger time and amount of scrolling can be useful indicators of interest. They also provided a useful categorization of interest indicators.

One issue with implicit data collection is that most observations are positive in nature and it is up to the system to use some heuristics to decide on what defines a negative observation. For example, the use of the back button after the user spends only a short time on a page can be inferred as being a negative observation or the choosing of a document from a list may render the other items in the list as being classified as not interesting [44], [45]. Even when certain negative actions are observed such as the deletion of an item from a shopping trolley, heuristics must be used to decide on how the initial interest in an item, i.e. inserting of the product in the shopping basket, must be amended when the item is deleted from the basket. Schwab et al. [46] propose a system that only employs positive feedback data to avoid the use of such heuristics. Hotle and Yan [47] showed that implicit negative feedback data can greatly improve the effectiveness of a conversational recommendation system, however, care must be taken in deciding what feedback can be attributed as being negative.

It is worth noting at this point that some of the implicit interest indicators used in these evaluations required data to be collected on the client side, while other data can be collected on the Web server, albeit with some inaccuracy, servicing the user request.

Explicit data input has a cost associated with it as it requires users to detract from their principle reason for interacting with the system and provide data, the benefits of which are intangible to the user. A number of studies carried out by the IBM User Interface Institute in the early 1980's confirm that, in general, users are motivated to get

[2] The time spent viewing an item and its associated content.

started with using a system and do not care about spending time up front on setting up the system, as is required by personalization systems that are dependent on explicit data being provided by the user. Carroll and Rosson [48] refer to this phenomenon as the "paradox of the active user" as users would save time in the long term by taking some initial time to optimize the system but that's not how people behave in the real world. While the studies were not aimed at personalization systems per se, the conclusion of the studies that engineers must not build products for an idealized rational user, rather they must design for the way users actually behave is just as valid for personalization systems. Studies in personalization show that without tangible benefits for the user, the user tends to read a lot more documents than they bother ranking [49]. By generating data that indicates a users interest in an object without the user needing to provide this information would result in more data and a reduction in sparsity, that exists especially in large information resources, typical of the Web. Additionally, privacy concerns also imply that users on the Internet tend to only provide accurate information that is deemed essential. Berendt and Teltzrow [50] suggest that users on the Internet exhibit varying degrees of privacy concerns and a large percentage of users would be happy to impart with various degrees of private information based on the perceived benefit to them in doing so. An interesting implication for designing personalization systems.

5 Personalization Techniques

In this section we describe the various approaches used for generating a personalized Web experience for a user.

5.1 Content-Based Filtering

Content based filtering systems have their roots in information retrieval. The approach to recommendation generation is based around the analysis of items previously rated by a user and generating a profile for a user based on the content descriptions of these items. The profile is then used to predict a rating for previously unseen items and those deemed as being potentially interesting are presented to the user. A number of the early recommender systems were based on content-based filtering including Personal WebWatcher [45], InfoFinder [51], NewsWeeder [9], Letizia [44] and Syskill and Webert [52]. Mladenic [53] provides a survey of the commonly used text-learning techniques in the context of content filtering, with particular focus on representation, feature selection and learning algorithms.

Syskill and Webert learns a profile from previously ranked Web pages on a particular topic to distinguish between interesting and non-interesting Web pages. To learn the profile, it uses the 128 most informative words, defined using expected information gain, from a page and trains a naïve Bayes classifier to predict future, unseen pages as potentially interesting or not for the user. The user may provide an initial profile for a topic, which in the case of Syskill and Webert, requires the definition of conditional probabilities for each word, given a page that is (not) interesting to the user. As pages get rated, these initial probabilities are updated, using conjugate priors [54], to reflect the rating of the pages by the user.

Rather than requiring the user of explicitly rate documents, Letizia uses implicit indicators of interest coupled with *tfidf* to compute content similarity between previosuly browsed interesting pages and candidate pages in the proximity of the users current browsing activity. To maximise the added value of the system, as opposed to the depth-first search carried out by most Web users, Letizia carries out a breadth first search of the hyperlinked documents, maintaining a list of documents that it believes to be relevant to the user.

Schwab et al. [46] propose the use of a naïve Bayes and nearest neighbor approach to content based filtering to build a user profile from implicit observations. In their approach they specifically abstain from using any heuristics for assigning certain observations as negative feedback, instead modifying the use of nearest neighbor and naïve Bayes to deal with only positive observations through the use of distance and probability thresholds. They also proposed a novel approach to feature selection based on the deviation of feature values for a specific user from the norm.

The main drawback of content-based filtering systems is their tendency to overspecialize the item selection as recommendations are solely based on the users previous rating of items, resulting in recommended items being very similar to previous items seen by the user. User studies have shown that users find online recommenders most useful when they recommend unexpected items [55], alluding to the fact that the overspecialization by content-based filtering systems is indeed a serious drawback. One approach to dealing with this problem is to inject some form of diversity within the recommendation set (see Section 6.5).

5.2 Traditional Collaborative Filtering

Goldberg et al. [56] first introduced collaborative filtering as an alternative to content based filtering of a stream of electronic documents. The basic idea as presented by Goldberg et al. was that people collaborate to help each other perform filtering by recording their reactions to e-mails in the form of annotations.

The application of this technology for recommending products has gained popularity and commercial success [57]. In a recommendation context, collaborative filtering works as described below.

Users provide feedback on the items that they consume, in the form of ratings. To recommend items to the active user, u_a, previous feedback is used to find other likeminded users (referred to as the user's neighbourhood). These are users that have provided similar feedback to a large number of the items that have been consumed by u_a. Items that have been consumed by likeminded users but not by the current user are candidates for recommendation. The assumption made by these systems is that users that have had common interests in the past, defined by feedback on items consumed, will have similar tastes in the future.

The rating data that is input to a collaborative filtering system is often referred to as a ratings matrix where each column is associated with an item in I, and each row contains the ratings of the items by an individual user.

To achieve its goal of providing useful recommendations, a collaborative filtering system must provide algorithms for achieving the following:

– a metric for measuring similarity between users, for neighbourhood formation
– a method for selecting a subset of the neighbourhood for prediction
– a method for predicting a rating for items not currently rated by the active user

A number of metrics have been proposed for measuring the similarity between users including Pearson and Spearman Correlation [12], the cosine angle distance [58], Entropy, Mean-squared difference and constrained Pearson correlation [28]. The most commonly used metric is the cosine angle which has been shown to produce the best performance. It is calculated as the normalized dot product of user vectors:

$$sim(u_a, u_b) = \frac{u_a \cdot u_b}{\|u_a\|^2 . \|u_b\|^2}$$

Once the similarity of the active user with all other users has been computed, a method is required to calculate the ratings for each item $i_j \in I_a^{(u)}$. The most commonly used approach is to use the weighted sum of rank

$$r_{u_a}(i_j) = \overline{r}_{u_a} + \frac{\sum_{u_k \in U_j} sim(u_a, u_k) \times (r_{u_k}(i_j) - \overline{r}_{u_k})}{\sum_{u_k \in U_j} sim(u_a, u_k)}$$

where $U_j = \{u_k \mid u_k \in U \wedge u_k(i_j) \neq \perp\}$ and $\overline{r}_{u_a}$ and $\overline{r}_{u_k}$ are the average ratings for users u_a and u_k respectively.

As the number of users and items increases, this approach becomes infeasible. Other than performance considerations, there is also a case to be made for reducing the size of the neighborhood with respect to the accuracy of the recommendations [59] as with a majority of neighbors not similar to the current user, the noise generated by their ratings can reduce the accuracy of the recommendations. Hence a method is required to select a subset of users, defining the neighborhood of the current user. Only users in the active users neighbourhood are then used to predict item ratings. Two approaches have been used in literature to select the neighborhood. One is based on a threshold on the similarity value [28] and the other uses a threshold on the number of neighbors, irrespective of the similarity value, which is traditionally used by the k-nearest neighbor approach to lazy learning. One of the problems with using a threshold on similarity is that as the number of items increases, the sparsity of the active user's neighbourhood increases, reducing the coverage of the recommender system. On the other hand, when using a fixed number of neighbours, the accuracy of the predictions will be low for users that have more unique preferences.

A number of variants have been proposed to the basic collaborative filtering process described above. First, Herlocker [60] proposed the use of a significance weighting that incorporated a measure of how dependable the measure of similarity between two users is. The idea behind this weight was the fact that, in traditional collaborative filtering, two users would be considered equally similar whether they had two items rated in common or whether it was fifty. Intuitively, this would seem strange as in the first case we are basing the similarity measurement on a very small amount of data. Empirical evaluation carried out by Herlocker et al. suggested that neighbors based on these small samples were bad predictors of the interests of the active user. As a result, they proposed a significance measure that associated a weight in the unit interval to each user, based on how many items were involved in the similarity calculation. Both, the similarity metric and the significance weight were used when generating the active user's neighbourhood.

Secondly, traditional collaborative filtering gives an equal importance to all items within the similarity calculation. Noting that not all items are equally informative, Herlocker et al. [60] proposed the introduction of a variance weighting to take into account the variability of values within a single column of the ratings matrix. A low variance would suggest that most users have a similar rating for the item and as such the item is less effective in discriminating between users and should therefore have little effect of the similarity calculation between two users. Breese et al. proposed the use of inverse user frequency where items less frequently rated were given a lower weight [59]. They also proposed case amplification that heightened the weight associated with those users that had a similarity, to the active user, close to 1.

Finally, to deal with the fact that ratings are inherently subjective and users tend to have different distributions underlying their item ratings, normalization of ratings provided by each user was proposed by Resnick et al. [12]. Rankings were scaled based on their deviations from the mean rating for the user. An alternative method for performing the scaling of ratings is to compute z-scores to also take into account the differences in spread of the ratings [60].

While collaborative filtering is commercially the most successful approach to recommendation generation, it suffers from a number of well known problems including the cold start/latency problem (see Section 6.1) and sparseness within the rating matrix (see Section 6.2). Traditional collaborative filtering also suffers from scalability issues (see Section 6.3). More recently, malicious attacks on recommender systems [61] (see Section 6.9) have been shown to affect traditional user-based collaborative filtering to a greater extend than model based approaches such as item-based collaborative filtering.

5.3 Model Based Techniques

Model based collaborative filtering techniques use a two stage process for recommendation generation. The first stage is carried out offline, where user behavioral data collected during previous interactions is mined and an explicit model generated for use in future online interactions. The second stage, is carried out in real-time as a new visitor begins an interaction with the Web site. Data from the current user session is scored using the models generated offline, and recommendations generated based on this scoring. The application of these models are generally computationally inexpensive compared to memory-based approaches such as traditional collaborative filtering, aiding scalability of the real-time component of the recommender system.

Model generation can be applied to explicitly and implicitly obtained user behavioural data. While the most commonly used implicit data is Web usage data, data pertaining to the structure and content are also often used.

A number of data mining algorithms have been used for offline model building including Clustering, Classification, Association Rule Discovery, Sequence Rule Discovery and Markov Models. In this section we briefly describe these approaches.

Item-Based Collaborative Filtering. In item-based collaborative filtering the offline, model building, process builds an *item similarity matrix*. The item similarity matrix, IS, is an $n \times n$ matrix where $IS[j,t]$ is the similarity of items i_j and i_t. Rather than basing item similarity on content descriptions of the items, similarity between items is based on user ratings of these items, hence each item is represented by an m dimensional vector, and the similarity computed using metrics such as (adjusted-) cosine similarity and correlation-based similarity [62]. The recommendation process predicts the rating for items not previously rated by the user by computing a weighted sum of the ratings of items in the item neighbourhood of the target item, consisting of only those items that have been previously rated by the user.

The model itself can be rather large, being in $O(n^2)$. An alternative is to store only the similarity values for the k most similar items. k is referred to as the model size. Clearly as k becomes small, the coverage as well as accuracy of the model will reduce.

Evaluation of the item-based collaborative filtering approach [62] showed that item-based collaborative filtering approaches provide better quality recommendations than the user based approach for rating prediction.

Clustering Based Approaches. Two main approaches to clustering for collaborative filtering have been proposed. These are item-based and user-based clustering. In user-based clustering, users are clustered based on the similarity of their ratings of items. In item based clustering, items are clustered based on the similarity of ratings by all users in U. In the case of user-based clustering, each cluster centre $C_k^{(U)}$ is represented by an n-dimensional vector, $C_k^{(U)} = (ar_1, ar_2, ..., ar_n)$, where each ar_j is the average item rating for (or average weight associated with) item i_j by users in cluster k. In the case of item-based clustering the cluster centre is represented by an m-dimensional vector $C_k^{(I)} = (q_1, q_2, ..., q_m)$, where each q_i is the average ratings by user, u_i of items within the cluster.

In the case of Web usage or transaction data a number of other factors can also be considered in determining the item weights within each profile, and in determining the recommendation scores. These additional factors may include the link distance of pages to the current user location within the site or the rank of the profile in terms of its significance.

The recommendation engine can compute the similarity of an active user's profile with each of the discovered user models represented by cluster centroids. The top matching centroid is used to produce a recommendation set in a manner similar to that used in user-based collaborative filtering.

Various clustering algorithms have been used, including partitioning algorithms such as, K-means for item and user-based clustering [63], ROCK [64] for item-based clustering, agglomerative hierarchical clustering [64] for item-based clustering, divisive hierarchical clustering for user-based and item-based clustering [65], mixture resolving algorithms such as EM [66] to cluster users based on their item ratings [59] and Gibbs Sampling [59].

Motivated by reducing the sparseness of the rating matrix, O'Connor and Herlocker proposed the use of item clustering as a means for reducing the dimensionality of the rating matrix [64]. Column vectors from the ratings matrix were clustered based on their similarity, measured using Person's correlation coefficient, in user ratings. The clustering resulted in the partitioning of the universe of items and each partition was treated as a separate, smaller ratings matrix. Predictions were then made by using traditional collaborative filtering algorithms independently on each of the ratings matrices.

Kohr and Merialdo proposed the use of top-down hierarchical clustering to cluster users and items. Clustering results in two cluster hierarchies, one based on the item ratings by users and the other based on the user ratings of items [65]. For the active user, the predicted rating for an item is generated using a weighted average of cluster centre coordinates for all clusters from the root cluster to appropriate leaf node of each of the two hierarchies. The weights are based on the intra-cluster similarity of each of the clusters.

Association and Sequence Rule Based Approaches. Association and Sequence rule discovery [67], [68] techniques were initially developed as techniques for mining supermarket basket data but have since been used in various domains including Web mining [69]. The key difference between these algorithms is that while association rule discovery algorithms do not take into account the order in which items have been accessed, sequential pattern discovery algorithms do consider the order when discovering frequently occurring itemsets. Hence, given a user transaction $\{i_1, i_2, i_3\}$, the transaction supports the association rules $i_1 \Rightarrow i_2$ and $i_2 \Rightarrow i_1$ but not the sequential pattern $i_2 \Rightarrow i_1$.

The discovery of association rules from transaction data consists of two main parts: the discovery of frequent itemsets [3] and the discovery of association rules from these frequent itemsets which satisfy a minimum *confidence* threshold.

Given a set of transactions T and a set $I = \{I_1, I_2, \ldots, I_k\}$ of itemsets over T. The *support* of an itemset $I_i \in I$ is defined as

$$\sigma(I_i) = \frac{|\{t \in T : I_i \subseteq t\}|}{|T|}$$

An association rule, r, is an expression of the form $X \Rightarrow Y\ (\sigma_r, \alpha_r)$, where X and Y are itemsets, $\sigma_r = \sigma(X \cup Y)$ is the support of $X \cup Y$ representing the probability that X and Y occur together in a transaction. The confidence for the rule r, α_r, is given by $\sigma(X \cup Y)/\sigma(X)$ and represents the conditional probability that Y occurs in a transaction given that X has occurred in that transaction.

Additional metrics have been proposed in literature that aim to quantify the interestingness of a rule [70], [71], [72] however we limit our discussion here to support and confidence as these are the most commonly used metrics when using association and sequence based approaches to recommendation generation.

The discovery of association rules in Web transaction data has many advantages. For example, a high-confidence rule such as {special-offers/, /products/software/} $\Rightarrow$ {shopping-cart/} might provide some indication that a promotional campaign on software products is positively affecting online sales. Such rules can also be used to optimize the structure of the site. For example, if a site does not provide direct linkage between two pages A and B, the discovery of a rule {A} $\Rightarrow$ {B} would indicate that providing a direct hyperlink might aid users in finding the intended information.

The result of association rule mining can be used in order to produce a model for recommendation or personalization systems [73,74,75,76]. The *top-N* recommender

[3] Itemsets which satisfy a minimum *support* threshold.

systems proposed in [76] uses association rules for making recommendations. First all association rules are discovered from purchase data. Customer's historical purchase information is then matched against the left-hand-side of the rule in order to find all rules supported by a customer. All right-hand side items from the supported rules are sorted by confidence and the first N highest ranked items are selected as the recommendation set.

One problem for association rule recommendation systems is that a system cannot give any recommendations when the dataset is sparse. In [73] two potential solutions to this problem were proposed. The first solution is to rank all discovered rules calculated by the degree of intersection between the left-hand-side of rule and a user's active session and then to generate the top k recommendations. The second solution is to utilize collaborative filtering: the system finds "close neighbors" who have similar interest to a target user and makes recommendations based on the close neighbor's history. In [74] a collaborative recommendation system was presented using association rules. The proposed mining algorithm finds an appropriate number of rules for each target user by automatically selecting the minimum support. The recommendation engine generates association rules for each user, among both users and items. If a user minimum support is greater than a threshold, the system generates recommendations based on user association, else it uses item association.

In [75] a scalable framework for recommender systems using association rule mining was proposed. The proposed recommendation algorithm uses an efficient data structure for storing frequent itemsets, and produces recommendations in real-time, without the need to generate all association rules from frequent itemsets. In this framework, the recommendation engine based on association rules matches the current user session window with frequent itemsets to find candidate pageviews for giving recommendations. Given an active session window w and a group of frequent itemsets, we only consider all the frequent itemsets of size $|w| + 1$ containing the current session window. The recommendation value of each candidate pageview is based on the confidence of the corresponding association rule whose consequent is the singleton containing the pageview to be recommended. In order to facilitate the search for itemsets (of size $|w| + 1$) containing the current session window w, the frequent itemsets are stored in a directed acyclic graph, called a *Frequent Itemset Graph*. The Frequent Itemset Graph is an extension of the lexicographic tree used in the "tree projection algorithm" [77]. The graph is organized into levels from 0 to k, where k is the maximum size among all frequent itemsets. Given an active user session window w, sorted in lexicographic order, a depth-first search of the Frequent Itemset Graph is performed to level $|w|$. If a match is found, then the children of the matching node n containing w are used to generate candidate recommendations.

When discovering sequential patterns from Web logs, two types of sequences are identified: Contiguous or Closed Sequences and Open Sequences [69]. Contiguous sequences require that items appearing in a sequence rule appear contiguously in transactions that support the sequence. Hence the contiguous sequence pattern $i_1, i_2 \Rightarrow i_3$ is satisfied by the transaction $\{i_1, i_2, i_3\}$ but not by the transaction $\{i_1, i_2, i_4, i_3\}$, as i_4 appears in the transaction between the items appearing in the sequence pattern. On the other hand, both transactions support the rule if it were an open sequence rule.

Given a transaction set T and a set $S = \{S_1, S_2, \ldots, S_n\}$ of frequent (contiguous) sequential patterns over T, the support of each S_i is defined as follows:

$$\sigma(S_i) = \frac{|\{t \in T : S_i \text{ is (contiguous) subsequence of } t\}|}{|T|}$$

The confidence of the rule $X \Rightarrow Y$, where X and Y are (contiguous) sequential patterns, is defined as

$$\alpha(X \Rightarrow Y) = \frac{\sigma(X \circ Y)}{\sigma(X)},$$

where $\circ$ denotes the concatenation operator. The Apriori algorithm used in association rule mining can also be adopted to discover sequential and contiguous sequential patterns. This is normally accomplished by changing the definition of support to be based on the frequency of occurrences of subsequences of items rather than subsets of items [78].

To aid performance of the recommendation process, sequential patterns are typically stored in the form of a single trie structure with each node representing an item and the root representing the empty sequence. Recommendation generation can be achieved in $O(s)$ by traversing the tree, where s is the length of the current user transaction deemed to be useful in recommending the next set of items. Mobasher et al. [79] use a fixed size sliding window, of size m, over the current transaction for recommendation generation. Hence the maximum depth of the tree required to be generated is m+1. The size of the trees generated during the offline mining can be controlled by setting different minimum support and confidence thresholds.

An empirical evaluation of association and sequential pattern based recommendation showed that site characteristics such as site topology and degree of connectivity can have a significant impact on the usefulness of sequential patterns over non-sequential (association) patterns [80]. Additionally, it has also been shown that contiguous sequential patterns are particularly restrictive and hence are more valuable in page prefetching applications rather than in recommendation generation [79].

A technique related to the use of sequential rules is that of modeling Web interactions as Markov Chain models. A Markov model is represented by the 3-tuple $\langle A, S, T \rangle$ where A is a set of possible actions, S is the set of all possible states for which the model is built and T is the Transition Probability Matrix that stores the probability of performing an action $a \in A$ when the process is in a state $s \in S$. In the context of recommendation systems, A is the set of items and S is the visitor's navigation history, defined as a k-tuple of items visited, where k is referred to as the order of the Markov model. As the order of the Markov model increases, so does the size of the state space, S. On the other hand the coverage of that space, based on previous history, reduces, leading to an inaccurate transition probability matrix. To counter the reduction in coverage, various Markov models of differing order can be trained and used to make predictions. The resulting model is referred to as the All-Kth-Order Markov model [81]. The downside of using the All-Kth-Order Markov model is the large number of states. Also, the issue regarding the accuracy of transition probabilities especially for the higher order Markov models is not addressed. Selective Markov models that only store some of the states within the model have been proposed as a solution to this problem [82]. A post pruning approach is used to prune out states that cannot be expected to be accurate predictors. Three pruning approaches based on the support, confidence and estimated error were proposed.

Rather than pruning states as a post process, sequence rule discovery and association rule discovery algorithms actively prune the state space during the discovery process using support. A further post pruning, based on confidence of the discovered rules, is also carried out. Hence the Selective Markov model is analogous to sequence rule discovery algorithms. Note however that the actual pruning process based on confidence proposed by Deshpande and Karypis [82] is not the same as that carried out during sequence rule discovery. Evaluation of Selective Markov models showed that up to 90% of states can be pruned without a reduction in accuracy. In fact some improvements in model accuracy resulted from pruning.

Graph Theoretic Approaches. Aggarwal et al. proposed a graph theoretic approach to collaborative filtering in which ratings data is transformed into a directed graph, nodes representing users and edges representing the predictability of a user based on the ratings of another user [83]. A directed edge exists from user u_i to u_j if user u_j predicts user u_i. To predict if a particular item, i_k, will be of interest to user u_i, assuming i_k has not been rated by the user, the shortest path from u_i is calculated to any user, say u_r, who has rated i_k and a predicted rating for i_k by u_i is generated as a function of the path from u_i to u_r.

Mirza et al. provide a framework for studying recommendation algorithms by graph analysis [84]. In their framework, ratings data is represented as a bipartite graph $G = \langle U \cup I, E \rangle$ with nodes representing either users or items, while edges represent ratings of items by users. A social network is constructed using the concept of a jump which is defined as a mapping from the ratings data to a subset of $U \times U$. Mirza et al. define a number of different types of jump, the simplest being a skip, results in an edge between two users if there exists at least one item that both of them have rated. In general, different social networks emerge based on the definition of the jump used. Mirza describes a number of ways in which jumps can be defined [85]. One such jump that mirrors traditional collaborative approaches to recommendation is the hammock jump, which requires a user defined parameter, w, known as the *hammock width*. For an edge to exist between two users u_k and u_l within the resulting social network, the hammock width must be less than or equal to $\mid I_k^{(r)} \cap I_l^{(r)} \mid$. The skip is, therefore, a special case of the hammock jump with hammock width 1. A third graph, called a recommender graph is then defined as a bipartite directed graph $G_R = \langle U \cup I, E_R \rangle$, with nodes $i_k \in I$ restricted to having only incoming edges. The shortest path from a user, u_i to an item in the graph can then be used to provide the basis for recommendations.

5.4 Hybrid Techniques

Other than the approaches discussed above, a number of hybrid approaches to personalization have also been proposed. These hybrid recommenders have been motivated by the observation that each of the recommendation technologies developed in the past have certain deficiencies that are difficult to overcome within the confines of a single recommendation approach. For example, the inability of collaborative filtering approaches to recommend new items items can be solved by coupling it with a content based recommendation approach. Not surprisingly, the most common form of hybrid recommender combines content based and collaborative filtering. An example of such a system is Fab [14], a recommendation system for Web content. Fab consists of a number of collection and selection agents. Collection agents are responsible for gathering pages pertaining to a small set of topics of interest of users. As the topics are based on user interests these may evolve with time to reflect the changing interests of the system's users. The selection agents select a set of pages for specific users out of the overall set of pages collected by the collection agents. The user rates each page presented to him by the selection agent. Each user has its own selection agent that contains a profile based on keywords contained in pages that have been previously rated by the user. Ratings for individual pages are also passed back to the original collection agents that can refine their own collection profile. Note that the collection agents profile is based on ratings from various users as opposed to just one user as is the case for the selection agent. The collaborative component of the system is based on the definition of a neighbourhood for each user within which pages rated highly are shared.

Another form of hybrid recommender that has recently been gaining a lot of attention is that which combines item ratings with domain ontologies (see Section 6.7).

More generically, Pazzani showed that combining various recommendations generated using different information sources such as user demographics, item content and user ratings (collaboratively) increases the precision of the recommendations [30].

Based on their study on the impact of site characteristics on the usefulness of sequential patterns over non-sequential (association) patterns [80], Nakagawa and Mobasher [86] proposed a hybrid recommendation system that switched between different recommendation systems based on the degree of connectivity of the site and the current location of the user within the site. Evaluation of this approach revealed that the hybrid model outperformed the base recommendation models in both precision and coverage.

Burke provides a comprehensive analysis of approaches to generating hybrid recommendation engines [87].

6 Issues

The study of recommendation systems over the last decade have brought to light a number of issues that must be addressed if these systems are to find acceptance within the wider context of personalized information access. In this section we discuss these issues. Along with a description of the issue we also discuss solutions that have been proposed to date to resolve them.

6.1 The Cold Start and Latency Problem

Personalization systems expect to have some information available on the individual users so that they can leverage this information to present items of interest to the user in future interactions. Hence, a new user with no interaction history poses a problem to the system as it is unable to personalize its interactions with the user. This is often referred to as the *new user problem*. The lack of useful interactions may put the user off the system before the system is able to gather the data it requires to start personalizing its interactions with the user.

A similar issue is posed by the introduction of a new item. When a new item becomes available, the lack of rating data means that systems that depend on item ratings solely (for example, collaborative filtering based approaches) cannot recommend the new item before a considerable history of has been collected. This is often referred to as the *New Item or Latency* problem. A collaborative filtering system provides no value to the first user in a neighborhood to rate an item. This need for altruistic behavior can further delay the introduction of a new item into the recommendation process [49].

The new user problem is even more acute at the point of time when a collaborative system is initially installed as not only is rating data not available for a single user but there is no rating data for any users of the system which is referred to a the *Cold Start* problem.

An approach often used to alleviate the new user/ item problem has been to use hybrid recommendation techniques, typically those that combine collaborative techniques with content based filtering techniques [88], or those based on demographic profiling [31].

Massa and Avesani propose the incorporation of a Web of trust within the recommendation process and show that using this additional information can be very effective in addressing the new user problem [89]. However, this does assume the existence of a Web of trust which in itself may not be available.

Middleton et al. [4] propose the use of an external ontology as seed knowledge for a recommender system as a solution to the cold start problem. The Quickstep recommender system developed by Middleton et al. aims to provide academics with recommendations of papers of interest. Feedback from the academics is incorporated into an ontology based user profile. To avoid the cold start problem, Quickstep uses information from the research publication and personnel database of the academic institution to populate an initial profile for the user. This approach obviously assumes the availability of an external ontology that may not always be available.

Haase et al. [5] approach the cold start problem by reusing the properties of a peer-to-peer network using profiles of similar peers in the semantic neighborhood to initialize the profile of a new peer.

6.2 Data Sparseness

Sparsity refers to the fact that as the number of items increases, even the most prolific users of the system will only explicitly or implicitly rate a very small percentage of all items. As a result, there will be many pairs of customers that have no item ratings in common and even those that do will not have a large number of common ratings. The nearest neighbor computation resulting from this fact will not be accurate and hence a low rating for an item would not imply that similar items will not be recommended [90]. To counter the effect of an increasing number of items, for collaborative filtering to provide accurate predictions, the number of users required to rate a sizeable number of items will be much higher than that required when the number of items is small.

Sarwar et al. [49] evaluated the benefit of using simple information filtering bots on Usenet news to generate ratings for new items published. The bots generated ratings for items based on the correctness of spellings, length of article and length of the included message. The value of these bots was evaluated in various Usenet news groups. The filter bots are treated in similar manner to ordinary users and hence their ratings are only used when these filterbots are in the neighbourhood of a current user. Good et al. [27] extended this research by using a number of information filtering agents in the domain of movie recommendation that used genre, cast and keywords for generating ratings. Some of these bots included a learning component for example, a bot that used inductive logic programming [91] to learn a model for predicting ratings based on genre and keywords. Good et al. also suggested a number of ways in which ratings from individual bots could be combined with user ratings to generate rating predictions.

Motivated by the observation that as the number and diversity of items increases, it is less likely that a user's rating of an item will be affected by all other item ratings, for recommending Usenet news articles, Resnick et al. [12] showed that creating separate item partitions for each discussion group can improve performance of the recommender system. However, such a process is by its very nature not transferable to other domains, requiring a domain specific partitioning scheme to be devised for every new domain that the technique is applied to. O'Connor and Herlocker [64] investigated the use of item clustering to discover groups of items that show similar ratings behavior from users. Pearson correlation coefficient was used to compute the similarity between items. That is, two items were deemed as being similar if there was a strong correlation between the ratings of these items by users in general. Evaluation of this approach using MovieLens data however showed that while partitioning based on item clustering provides more accurate recommendations than random partitioning, genre based partitioning outperformed all of the item ratings based clustering approaches.

Goldberg et al. [92] proposed the use of a *gauge set* of items. This is a set of items that all users of the system must rate to seed the system. The gauge set provides the basis for a more accurate measurement of similarity between users as it would consist of a dense rating submatrix.

6.3 Scalability

Memory based approaches such as traditional collaborative filtering suffer from scalability issues as the number of users increases as well as an increasing number of candidate items.

A number of solutions have been proposed to deal with an increasing user base. The most widely used approach is to use a model-based approach to collaborative filtering rather than one that is memory based. An alternative is to limit the number of users that must be compared when making predictions for the active user. This can be achieved by either limiting the number of profiles stored (instance selection) or by indexing the user base and searching only a part of the whole user base for an active user (instance indexing).

Yu et al. [93] proposes an metric for use in instance selection, based on the information theoretic measure of mutual information, called relevance. The rationale behind instance selection is that, for a given target item, the rating of other items by a user should provide enough information to support the rating by the user of the target item. If this is not the case, then the user would probably not provide a useful basis for predicting the rank of the target item for the target user. Hence relevance of an instance (user) for predicting the rank, r, of a particular item, i, is calculated as $r(u,i) = \sum_{j \in J} I(V_i; V_j)$, where J is the set of all items other than j, rated by the user, u and $I(.,.)$ is the mutual information measure. Using the relevance metric, only the top N user instances are used for predicting the rating for the target item.

Chee et al. [94] propose the use of k-means to iteratively partition the rating matrix based on the rating similairty of users. The leaf nodes of the resulting binary tree consist of "cliques" of users with similar tastes. During prediction, the tree is navigated and similarity evaluated only within the clique of the active user.

Dealing with the issue of scalability with respect to the number of items is akin to feature subset selection and dimensionality reduction in machine learning. The most commonly used approach to dimensionality reduction applied to recommender systems have been singular value decomposition [95], [58] and principal component analysis [92]. Not only has singular value decomposition been shown to effectively reduce the dimensionality of the ratings matrix, but it has also been shown to improve accuracy of the recommendations when applied to less sparse ratings matrices through reduction in noise within the rating matrix. Approaches to incrementally build models based on singular value decomposition have also been investigated so as to avoid the expensive rebuilding of the model as new data becomes available [96], [97].

Tang et al. propose a the use of heuristics to limit the number of items considered [98]. For the movie recommendation domain they suggest using the temporal feature of items (year of release of a movie) to limit the set of candidate movies for recommendation.

6.4 Privacy

Currently U.S. laws impose little restrictions on private parties communicating information about people, leaving it up to the parties involved to define the extent of any such communication through a contract [99]. In particular, an online business may provide their customers with a privacy policy that would outline under what conditions, if at all, the business would share the information they hold about the customer. Breech of such a contract entitles the customer to bring a law suit on the business but not on any third party that has gained access to data as a consequence of the breech. In particular it is common place for a business suffering bankruptcy to sell the data they hold on their customers. Such a sale is currently supported by the law in the U.S [100].

Even if a business does not explicitly sell customer data, services such as collaborative filtering based recommender systems can be exploited to gain insights into individual customers preferences [101]. This is particularly true of users who rate products across different domains, referred to as straddlers. While such users are particularly desirable to enable collaborative systems to generate serendipitous recommendations, it also means that a user, who is obviously aware of their own preferences, or indeed an individual masquerading as a user with a certain set of preferences, could potentially gain insights into straddlers. Using a graph-theoretic representation for recommender systems, Ramakrishnan et al. [101] provide an analysis of the effect of two recommender system parameters, the hammock width and hammock path length, on the risk to straddlers. Their study concluded that a hammock width just below the value that splits the graph into a set of disconnected components carries the greatest risks for straddlers.

6.5 Recommendation List Diversity

While most research into recommending items has concentrated on the accuracy of predicted ratings, other factors have been identified as being important to users. One such factor is the diversity of items in the recommendation list. In a user survey aimed at evaluating the effect of diversification on user satisfaction, applied to item-based and user-based collaborative filtering, found that it had a positive effect on overall satisfaction even though accuracy of the recommendations was affected adversely [102]. The study further concluded that introducing diversity affects user satisfaction to a greater extent when item-based collaborative filtering is used, while it has no measurable affect on user-based collaborative filtering.

Smyth and McClave [103] proposed three approaches to introducing diversity into recommendation sets. The basic approach is to balance similarity of an item to the target with the diversity of the current items within the recommendation set. Diversity was measured as the average distance between the candidate recommendation and all items currently with the recommendation set. Ziegler proposed an approach to diversity maintenance [102] similar to the bounded greedy selection approach proposed by Smyth and McClave.

Sheth [104] modelled the information filtering task as a population of profiles, per user, that evolve using genetic operators of crossover and mutation. The profiles are generated using standard text mining functions such as *tfidf* on documents presented to the user by the profile and the relevance feedback received. While the crossover operator exploits the fitness of the current population of profiles, mutation is used to introduce some diversity into the population. Unlike other content based filtering approaches, as the profiles evolve through the use of the genetic operators, it is more likely that a level of serendipity can be maintained within the recommendation set.

6.6 Adapting to User Context

Personalization aims to "hide" the rigidity of the Internet by providing useful, contextually relevant information and services to the user. However, context as a concept has rarely been incorporated into personalization research. One of the reasons for this is that it is hard to arrive at a consensus of what defines context let alone modeling the concept. Lieberman and Selker provide a useful starting point for defining context, defining it as "everything that affects the computation except the explicit input and output" [105]. Unfortunately, this definition in itself does not make the modeling of context possible as we cannot consider all previous user interactions with a system as context for the current interaction and nor can we explicitly measure context, hence we must use current behavior to discover the user context and then use this context to predict the current behavior of the user so as to better service his requirements. If we assume that user behavior is predictable based on past interactions, we now must select only those previous interactions that were undertaken within the same context and use them to predict the needs of the user.

Contextual retrieval is also viewed as an important challenge in the information retrieval community [106]. Parent et al. [36] proposed a client-side Web agent that allows the user to interact with a concept classification hierarchy to define the context of the query terms provided. The agent uses portions of the hierarchy to expand the initial search query, effectively adding ‘user intent’ to the query. Sieg et al. [6] define context by the portions of the concept hierarchy (such as the Yahoo Directory) that match the user query. Each node of the concept hierarchy has a vector representation based on the documents contained in the node and all its subcategories. Previously accessed documents are clustered (an offline process) and the cluster centres form the user’s profile. When a query is issued, all clusters from the user profile that have a similarity with the query above a pre-defined threshold are selected. The query is matched against the concept hierarchy and a subset of nodes are chosen from the concept hierarchy that have a certain amount of similarity to the query. The selected clusters are then used to further refine the selection. In [35], user context is captured via nodes in a concept lattice induced from the original ontology and is updated incrementally based on the user’s interactions with the concepts of the ontology. Updates are initiated through the user selecting or deselecting concepts within the lattice that were considered to be of interest by the system based on the user’s long-term and short term memories. The context is represented as a pair of term vectors, one for the selected concepts and the other representing the deselected concepts.

6.7 Using Domain Knowledge

Dai and Mobasher [107] provide a framework for integrating domain knowledge with Web usage mining for user based collaborative filtering. They highlight that semantics can be integrated at different stages of the knowledge discovery process.

Mobasher et al. proposed the use of semantic knowledge about items to enhance item-based collaborative filtering [90]. Their approach is to represent the semantic knowledge about an item as a feature vector and calculate the similarity based on this information to other items. This item-similarity is then combined with rating similarity to get an overall measure of item similarity which is used to predict the rating by a user of a currently unrated item.

Cho and Kim [108] apply a product taxonomy with Web usage mining to reduce the dimensionality of the rating database when searching for nearest neighbours while Niu, Yan et al. [109] build customer profiles based on product hierarchy in order to learn customer preferences.

Middleton et al. use an ontological profile for a user within their research paper recommendation system, QuickStep [4]. The profile is based on a topic hierarchy alone. They also attempt to use externally available ontologies based on personnel records and user publications to address the cold-start problem for their recommendations system. The existence of such additional knowledge, while applicable in their specific application domain, cannot however be assumed in a general e-tailer scenario.

Haase et al. create semantic user profiles from usage and content information to provide personalized access to bibliographic information on a Peer-to-Peer bibliographic network [5]. The semantic user profile consists of the expertise, recent queries, recent relevant instances and a set of weights for the similarity function.

Ghani and Fano [29] proposed a recommender system based on a custom-built knowledge base of product semantics. The focus within the paper is on generating ”soft” attributes from online marketing text, describing the products browsed, and using them to generate cross category recommendations.

6.8 Managing the Dynamics in User Interests

Most personalization systems tend to use a static profile of the user. However user interests are not static, changing with time and context. Few systems have attempted to handle the dynamics within the user profile.

In NewDude [110] the user model consists of a short term interests and a long term interests model. The short term interests model is based on the n most recently rated stories. Each item (story) is represented as a term vector using *tfidf*. The similarity of the target item to items within the short term interest profile is computed using cosine similarity. If the similarity of the target item to another story in the short term interest profile is greater than a threshold value, it is deemed as being a known story and is therefore discarded. Alternatively, stories from the short term interest profile that have a similarity value greater than a threshold value are deemed to be in the neighbourhood of the target item and are used to predict a rating for the target item. If the target is deemed to be of interest to the user, it is recommended, alternatively it is discarded. If the neighbourhood of the target item within the short term interest profile is empty, the long term interest profile is used to classify the target item. The long term memory is based on the 150 most informative words appearing in the items and the model is based on the multinomial formulation of the naïve Bayes [111].

Rather than use a fixed number of most recent user interactions, Koychev and Schwab suggest the use of a continuous weighting function that associates a higher weight to more recent interactions with a user [112]. Tests using a linear weighting function showed some improvements in predictive accuracy.

An alternative approach is based on the evolution of a population of profiles per user [113], [104]. As interests of users change, profiles that better reflect their current interests become more prominent within the population. Moukas and Zacharia separate out the two roles of information filtering and discovery and describe a market-based control scheme to control the fitness of information and discovery agents.

6.9 Robustness

The dependence of personalization systems on item ratings provided by users and their use of these ratings in generating social recommendations also opens them to abuse. For example, an interested party may decide to influence item recommendations by inserting false ratings for a subset of items that they have an interest in. Attacks of this nature are referred to as shilling [4] [115] or profile injection [116].

Recent research has begun to examine the vulnerabilities and robustness of different recommendation techniques, such as collaborative filtering, in the face of shilling attacks [116,117,115,118]. O' Mahony [119] identify two key types of attacks

– Push: This is an attack aimed at promoting a particular item by increasing its ratings for a larger subset of users

[4] A shill is an associate of a person selling a good or service, who pretends no association and assumes the air of an enthusiastic customer [114].

– Nuke: This is an attack aimed at reducing the predicted ratings of an item so that it is recommended to a smaller subset of users

These attacks take the form of the insertion of a number of new users with a set of rating that either provide high or low ratings to particular items.

A number of different attack models have been identified in literature. The *sampling attack* [118] is primarily of theoretical interest as it requires the attacker to have access to the ratings database itself. The *random attack* [115] forms profiles by associating a positive rating for the target item with random values for the other items. The *average attack* [115] assumes that the attacker knows the average rating for each item in the database and assigns values randomly distributed around this average, except for the target item. These attacks have been found to be effective against user-based collaborative recommendation algorithms, but less so against item-based recommendation.

A *segmented attack* [120] associates the pushed item with a small number of popular items of similar type. It pushes an item to a targeted group of users with known or easily predicted preferences. Profiles are inserted that maximize the similarity between the pushed item and items preferred by the group. This attack model ensures that the pushed item will be recommended to those users that are its target segment. It is particularly effective against item-based recommendation algorithms to a degree that broader attacks are not. This attack also requires very limited knowledge about the system and the users. An attacker needs to know only a group of items well liked by the target segment and needs to build profiles containing only those items.

The study of attack models and their impact on recommendation algorithms can lead to the design of more robust and trustworthy personalization systems. The notion of trust, which is essential to the practical success of recommender systems, is further discussed below. Another important goal is the development of metrics to help quantify the effect of those attacks (see Section 7).

6.10 Trust

A user study conducted by Sinha et al. found that, in general, two types of recommendations need to be generated by a recommender system. These are *trust-generating recommendations* and *useful recommendations* [55]. They define trust-generating recommendations as items that the user has previously experienced and suggest that while these recommendations are not "useful" to the user, they build trust between the user and the system. They also found that users preferred using trusted sources for recommendations.

However, most collaborative filtering systems base the generation of recommendations simply on the similarity of the target users previous ratings with that of other users, not explicitly dealing with the issue of trust. This opens such systems to attacks such as shilling as described in Section 6.9.

Recently researchers have begun looking at how trust can be incorporated into the recommendations process [89], [121], [122]. Massa and Avesani propose the use of a "Web of trust", a social network with users as nodes and directed weighted edges signifying a level of trust from one user to another. In their implementation of a trust-aware recommendation system, a user was allowed to rate not just items but also users based on the usefulness of their reviews/ ratings. Only users trusted by the target user were then employed within the recommendation generation process. Through the propagation of trust within the network, users not specifically rated by the target user may also participate in the recommendation process. Some of the additional advantages resulting from the use of such a trust based social network include alleviation of the new user problem commonly faced by traditional collaborative filtering systems as well as attack-resilience [121].

As opposed to depending on the user providing a Web of trust to the recommender system, O'Donovan and Smyth [122] investigated the possible learning of trust metrics from the ratings data available within the system itself. They defined two metrics for trust, one at the profile level and the other at the item level, based on the correctness of previous recommendations. The trust metric was combined with the similarity metric during neighborhood formulation.

Herlocker et al. investigated the ability of a recommender system to generate explanations for how individual recommendations were generated [123] . Three key points within the collaborative approach to recommendation generation that could provide useful information to be communicated to the user were identified as the user profile generation, neighbourhood formulation and neighbour rating combination for prediction.

They further identified the two key goals of generating explanations. The first was aimed at building trust with the user through provision of logical explanations for the recommendations generated. The second was aimed at providing the user with the ability to identify whether a recommendation is based on weak data. Additional benefits include improved data collection as a result of involving the user in the recommendation process and greater acceptance of the recommender as a decision aide. They further identified over twenty explanation interfaces and evaluated them using volunteer users of the MovieLens recommender. Of these the most valued feedback were histograms of ratings by neighbours, past performance of the recommender for the user and similarity to other items within the user's profile. Another useful finding of the study was that explanation interfaces must be simple to be successful and care must be taken not to overload to the user with information.

While explanations can be viewed positively by users, providing explanations such as ratings may further influence the user's own ratings as even simple feedback in terms of the predicted rating has been shown to influence the user's own rating [124].

7 Evaluation of Personalization Systems

Evaluation of personalization systems remains a challenge due to the lack of understanding of what factors affect user satisfaction with a personalization system. It seems obvious that a system that accurately predicts user needs and fulfils these needs without the user needing to expend the same resources in achieving the task as he would have, in the absence of the system, would be considered successful. Hence personalization systems have most commonly been evaluated is terms of the accuracy of the algorithms they employ.

Recent user studies have found that a number of issues can affect the perceived usefulness of personalization systems including, trust in the system, transparency of the recommendation logic, ability for a user to refine the system generated profile and diversity of recommendations [125], [126], [102].

For a business deploying a personalization system, accuracy of the system will be little solace if it does not translate into an increase in quantitative business metrics such as profits or qualitative metrics such as customer loyalty.

Hence the evaluation of personalization systems needs to be carried out along a number of different dimensions, some of which are better understood that others and have well established metrics available. The key dimensions along which personalization systems are evaluated include

– User Satisfaction
– Accuracy
– Coverage
– Utility
– Explainability
– Robustness
– Performance and Scalability

Attempts to measure user satisfaction range from using business metrics for customer loyalty such as RFM and life-time value through to more simplistic measures such as recommendation uptake. For example, the físchlár video recommendation system [127] implicitly obtains a measure of user satisfaction by checking is the recommended items were played or recorded.

As stated in Section 2, personalization can be viewed as a data mining task. The accuracy of models learned for this purpose can be evaluated using a number of metrics that have been used in machine learning and data mining literature such as mean absolute error (MAE) and area under the ROC curve, depending on the formulation of the learning task (see Section 2).

In the prediction task, MAE has been commonly used in collaborative filtering literature [28], [60], [93]. Other accuracy metrics used for the prediction task with numeric ratings include root mean squared error and mean squared error, that implicitly assign a greater weight to predictions with larger errors, and normalized mean squared error [92] that aims to normalize MAE across datasets with varying rating scales. Massa and Avesani suggest another variant of MAE called the mean absolute user error that calculates the mean absolute error for each user and then averages over all users [89]. This was based on their observation that recommender systems tend to have lower errors when predicting ratings by prolific raters rather than less frequent ones. This metric is particularly useful when the number of items in the test set per user varies, for example, if it is based on a percentage of items rated by a user.

Precision and Recall are standard metrics used in information retrieval. While precision measures the probability that a selected item is relevant, recall measures the probability that a relevant item is selected. Precision and recall are commonly used in evaluating the selection task [128], [58], [129]. The F1 measure that combines precision and recall, has also been used for this purpose task [130], [131].

Coverage measures the percentage of the universe of items that the recommendation system is capable of recommending. For the prediction task it is calculated as the percentage of unrated items, a rating for which can be predicted by the system. An alternative is to calculate coverage as a percentage of items of interest to a user rather than considering the complete universe of items [132].

Breese et al. suggested a metric based on the expected utility of the recommendation list [59]. The utility of each item is calculated by the difference in vote for the item and a "neutral" weight. The metric is then calculated as the weighted sum of the utility of each item in the list where the weight signifies the probability that an item in the ranked list will be viewed. This probability was based on an exponential decay. In the context of navigating a hyperlinked repository, other metrics have also been proposed that measure utility based on the distance of the recommended item from the current page referred to as navigation distance [133]. Another factor affecting utility of a recommendation is the novelty of the recommendation in the context of the overall recommendation list.

A number of metrics have been proposed in literature for evaluating the robustness of a recommender system. Each of these metrics attempt to provide a quantitative measure of the extent to which an attack can affect a recommender system. Stability of prediction [119] measures the percentage of unrated (user,items) pairs that have a prediction shift less that a predefined constant. Power of an attack [119] on the other hand measures the average change in the gap between the predicted and target rating for the target item. The target item is the item that the attack is attempting to push or nuke. The power of attack metric assumes that the goal of the attack is to force item ratings to a target rating value. Noting that the effect of an attack on an items current rating is not necessarily going to affect its ability to be recommended, Lam and Herlocker [61] proposed an alternative metric called the Change in Expected change in top-N occupancy. It is calculated as the average expected occurrence of the target items in the top-N recommendation list of users.

The performance and scalability dimension aims to measure the response time of a given recommendation algorithm and how easily it can scale to handle a large number of concurrent requests for recommendations. Typically, these systems need to be able to handle large volumes of recommendation requests without significantly adding to the response time of the Web site that they have been deployed on.

8 Conclusions and Future Directions

In this chapter, we have provided a comprehensive review of intelligent techniques for Web personalization. We have taken the view that Web personalization is an application of data mining and must therefore be supported during the various phases of a typical data mining cycle. We have described the various explicit and implicit data sources available along with the typical approaches used to transform this data into useful user profiles/models that can be used to generate recommendations. We have also described various approaches to generating recommendations from a set of user profiles/ models. Research into this topic has raised a number of interesting issues related to the personalization process. These are issues that need to be addressed by any personalization system that aims to provide robust, accurate and useful personalized content to its users. We also provide a description of the current understanding of how these systems should be evaluated, describing some of the most commonly used metrics within personalization literature.

While a lot has been achieved in the last decade of research into personalization, a number of challenges and open research questions still face researchers.

A key part of the personalization process is the generation of user models. Commonly used user models are still rather simplistic, representing the user as a vector of ratings or using a set of keywords. Even where more multi- dimensional information has been available, such as when collecting implicit measures of interest, the data has traditionally been mapped onto a single dimension, in the form of ratings. More expressive models need to be explored.

In particular profiles commonly used today lack in their ability to model user context and dynamics. Users rate different items for different reasons and under different contexts. The modelling of context and its use within recommendation generation needs to be explored further. Also, user interests and needs change with time. Identifying these changes and adapting to them is a key goal of personalization. However, very little research effort has been expended on this topic to date. This is partly due to the fact that at the deployment stage, the models used are static due to a trade-off between expressiveness of the profiles and scalability with respect to the number of concurrent personalization requests. Recently research has begun to explore user models that are based on ontological information. These richer profiles have shown promise in comparison to systems that limit user models to a vector representation. However this research is very much in its infancy and warrants further research.

Memory based approaches traditionally used for personalization suffer from scalability issues with respect to the size of the user base as well as the size of the universe of items. The applicability of research into instance selection for memory based learning [134] to collaborative filtering needs to be investigated. Also a number of indexing mechanisms based on similarity [135] have been proposed. The applicability of these to sparse data sets typically found in recommender systems needs to be investigated.

With regard to the robustness of recommenders, our understanding of attack models is still in its infancy as is our understanding of the extent to which these attacks affect the different approaches to developing recommender systems. Most studies have tended to evaluate the effect of these attacks on user-based and item-based collaborative filtering. More research needs to be carried out into how robust other model based and hybrid approaches to recommendation generation are to these attacks. Little work has been carried out into quantifying how difficult it would be to identify and prevent attacks from taking place. Data Mining has been applied successfully to network intrusion detection. Can similar techniques be applied to identifying attacks on recommender systems?

The ultimate goal of personalization is a lift in user satisfaction. However, most research into personalization has focussed evaluation on the accuracy of predicted ratings and little agreement has emerged as to what factors, other than prediction accuracy affect user satisfaction. Even less agreement exists with regard to how the effect of personalization on these factors should be measured. A lot more user studies need to be carried out to gain a better understanding of these issues. The development of more personalization exemplars with the necessary infrastructure to conduct large scale user testing is required. In addition to user satisfaction, more business oriented metrics need to be developed to measure the true economic benefit to businesses that deploy such systems.

User studies have shown explanability of recommendations as an important factor in user satisfaction, however, most systems for generating recommendations are hard to explain other than at the generic conceptual level. Explanation facilities developed in the context of knowledge based systems may provide some useful insights into how similar facilities can be developed for recommendation systems.

The use of trust within the computation of neighbourhoods has been shown to alleviate some of the issues associated with pure collaborative filtering such as the new user problem and robustness. However they require additional input from users in the form of trust networks. Some early work into using introspective learning for measuring trustworthiness of users within collaborative filtering has shown potential and warrants further investigation.

References

1. Mulvenna, M., Anand, S.S., Buchner, A.G.: Personalization on the net using web mining. Communication of ACM **43** (2000)
2. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: Crisp-dm 1.0: Step-by-step data mining guide. http://www.crisp-dm.org (2000)
3. Cooley, R., Mobasher, B., Srivastava, J.: Data preparation for mining world wide web browsing patterns. Knowledge and Information Systems **1** (1999)
4. Middleton, S.E., Shadbolt, N.R., Roure, D.C.D.: Ontological user profiling in recommender systems. ACM Transactions on Information Systems **22** (2004) 54–88
5. Haase, P., Ehrig, M., Hotho, A., Schnizler, B.: Personalized information access in a bibliographic peer-to-peer system. In: Proceedings of the AAAI Workshop on Semantic Web Personalization, AAAI Workshop Technical Report (2004) 1–12
6. Sieg, A., Mobasher, B., Burke, R.: Inferring user's information context: Integrating user profiles and concept hierarchies. In: Proceedings of the 2004 Meeting of the International Federation of Classification Societies. (2004)
7. Mobasher, B., Dai, H., Luo, T., Sung, Y., Nakagawa, M., Wiltshire, J.: Discovery of aggregate usage profiles for web personalization. In: Proceedings of the Web Mining for E-Commerce Workshop (WebKDD'2000), held in conjunction with the ACM-SIGKDD Conference on Knowledge Discovery in Databases (KDD'2000). (2000)
8. Anand, S.S., Mulvenna, M., Chevalier, K.: On the deployment of web usage mining. In Berendt, B., Hotho, A., Mladenic, D., van Someren, M., Spiliopolou, M., Stumme, G., eds.: Web Mining: From Web to Semantic Web. LNAI 3209. Springer-Verlag (2004) 23–42
9. Lang, K.: Newsweeder: Learning to filter netnews. In: Proceedings of the 12th International Conference on Machine Learning. (1995)
10. Salton, G.: Developments in automatic text retrieval. Science **253** (1991) 974–980
11. Rissanen, J.: Modelling by shortest data description. Automatica **14** (1978) 465–471
12. Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., , Riedl, J.: Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 Computer Supported Collaborative Work Conference. (1994)
13. Mobasher, B., Dai, H., Luo, T., Sung, Y., Zhu, J.: Integrating web usage and content mining for more effective personalization. In: Proceedings of the International Conference on E-Commerce and Web Technologies. (2000)

14. Balabanovic, M., Shohan, Y.: Fab: Content-based, collaborative recommendation. Communications of the ACM **40** (1997) 66–72
15. Burke, R.: Knowledge-based recommender systems. Encyclopedia of Library and Information Systems **69** (2000)
16. McGinty, L., Smyth, B.: Improving the performance of recommender systems that use critiquing. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
17. Lorenzi, F., Ricci, F.: Case-based recommender systems: a unifying view. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
18. McGinty, L., Smyth, B.: Comparison-based recommendation. In: Proceedings of the 6th European Conference on Case-based Reasoning. (2002)
19. Burke, R., Hammond, J., Kulyukin, V.A., Lytinen, S.L., Tomuro, N., Schoenberg, S.: Question answering from frequently asked question files. AI Magazine **18** (1997) 57–66
20. Fesenmaier, D.R., Ricci, F., Schaumlechner, E., Wober, K., Zanella, C.: Dietorecs: Travel advisory for multiple decision styles. In Frew, A.J., Hitz, M., O'Connor, P., eds.: Information and Communication Technologies in Tourism. Springer-Verlag (2003) 232–241
21. Shimazu, H.: Expertclerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops. Artificial Intelligence Review **18** (2002) 223–244
22. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing (2003) 76–80
23. Mobasher, B., Cooley, R., Srivastava, J.: Automatic personalization based on web usage mining. Communication of ACM **43** (2000)
24. Mobasher, B., Dai, H., Luo, T., , Nakagawa, M.: Effective personalization based on association rule discovery from web usage data. In: Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), held in conjunction with the International Conference on Information and Knowledge Management. (2001)
25. Eirinaki, M., Vlachakis, J., Anand, S.S.: Ikum: An integrated web personalization platform based on content structures and usage behaviour. In: Intelligent Techniques in Web Personalisation. LNCS. Springer-Verlag (2005)
26. Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J.: Applying collaborative filtering to usenet news. Communications of the ACM **40** (1997) 77–87
27. Good, N., Schafer, J.B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., Riedl, J.: Combining collaborative filtering with personal agents for better recommendations. In: Proceedings of the 1999 Conference of the American Association of Artificial Intelligence (AAAI-99). (1999) 439–446
28. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating word of mouth. In: Proceedings of CHI. (1995) 210–217
29. Ghani, R., Fano, A.: Building recommender systems using a knowledge base of product semantics. Accenture Technology Labs (2002)
30. Pazzani, M.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review **13** (1999) 393–408
31. Krulwich, B.: Lifestyle finder: Intelligent user profiling using large-scale demographic data. AI Magazine **18** (1997) 37–45
32. Maes, P.: Agents that reduce work and information overload. Communications of the ACM **37** (1994) 30–40
33. Yang, Y.: Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In: Proceedings of the 7th International ACM-SIGIR Conference on Research and Development in Information Retrieval. (1994) 13–22
34. O'Riordan, A., Sorensen, H.: An intelligent agent for high-precision text filtering. In: Proceedings of the 4th International Conference on Information and Knowledge Management. (1995) 205–211

35. Sieg, A., Mobasher, B., Burke, R., Prabhu, R.G., Lytinen, S.: Representing user information context with ontologies. In: Proceedings of HCI International Conference. (2005) 210–217
36. Parent, S., Mobasher, B., Lytinen, S.: An adaptive agent for web exploration based on concept hierarchies. In: Proceedings of the 9th International Conference on Human Computer Interaction. (2001)
37. Cassel, L., Wolz, U.: Client side personalization. In: Proceedings of the Second DELOS Network of Excellence Workshop on Personalization and Recommender Systems in Digital Libraries. (2001)
38. Guttman, R., Maes, P.: Agent-mediated integrative negotiation for retail electronic commerce. In: Proceedings of the Workshop on Agent Mediated Electronic Trading. (1998)
39. Murthi, B., Sarkar, S.: The role of the management sciences in research on personalization. Review of Marketing Science Working Papers **2** (2002)
40. Nichols, D.M.: Implicit rating and filtering. In: Proceedings of the fifth DELOS Workshop on Filtering and Collaborative Filtering. (1998) 31–36
41. Rucker, J., Polanco, M.: Siteseer: Personalized navigational for the web. Communications of the ACM **40** (1997) 73–76
42. Lieberman, H.: Autonomous interface agents. In: Proceedings of the ACM Conference on Human Factors in Computing Systems. (1997) 67–74
43. Claypool, M., Le, P., Waseda, M., Brown, D.: Implicit interest indicators. In: Proceedings of the 6th International Conference on Intelligent User Interfaces. (2001) 33–40
44. Lieberman, H.: Letizia: An agent that assists web browsing. In: Proceedings of the 14th International Joint Conference in Artificial Intelligence. (1995) 924–929
45. Mladenic, D.: Personal web watcher: Implementation and design. Technical Report IJS-DP-7472, Department of Intelligent Systems, J. Stefan Institute, Slovenia (1996)
46. Schwab, I., Kobsa, A., Koychev, I.: Learning about users from observation. In: Adaptive User Interfaces: Papers from the 2000 AAAI Spring Symposium. (2000)
47. Holte, R.C., Yan: Inferring what a user is not interested in. In: Lecture Notes In Computer Science (Proceedings of the 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence). Springer-Verlag (1996) 159–171
48. Carroll, J., Rosson, M.B.: The paradox of the active user. In Carrol, J.M., ed.: Interfacing Thought: Cognitive Aspects of Human-Computer Interaction. MIT Press (2005)
49. Sarwar, B.M., Konstan, J.A., Borchers, A., Herlocker, J., Miller, B., Riedl, J.: Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In: Computer Supported Cooperative Work. (1998) 345–354
50. Berendt, B., Teltzrow, M.: Addressing users' privacy concerns for improving personalization quality: Towards an integration of user studies and algorithm evaluation. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
51. Krulwich, B., Burkey, C.: Learning user information interests through extraction of semantically significant phrases. In: Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access. (1996)
52. Pazzani, M., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning **27** (1997) 313–331
53. Mladenic, D.: Text-learning and related intelligent agents: A survey. IEEE Intelligent Agents (1999) 44–54
54. Heckerman, D.: A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Corporation (1995)
55. Sinha, R., Swearingen, K.: Comparing recommendaions made by online systems and friends. In: Proceedings of Delos-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries. (2001)

56. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Communications of the ACM **35** (1992)
57. Schafer, J., Konstan, J., , Riedl, J.: Recommender systems in e-commerce. In: Proceedings of the ACM Conference on Electronic Commerce. (1999)
58. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Application of dimensionality reduction in recommender system - a case study. In: ACM WebKDD 2000 Web Mining for E-Commerce Workshop. (2000)
59. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence. (1998) 43–52
60. Herlocker, J., Konstan, J., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 1999 Conference on Research and Development in Information Retrieval. (1999)
61. Lam, S., Riedl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th international conference on World Wide Web. (2004) 393–402
62. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International World Wide Web Conference. (2001)
63. Ungar, L., Foster, D.P.: Clustering methods for collaborative filtering. In: Proceedings of the Workshop on Recommendation Systems. (1998)
64. O'Connor, M., Herlocker, J.: Clustering items for collaborative filtering. In: Proceedings of ACM SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation. (1999)
65. Kohrs, A., Merialdo, B.: Clustering for collaborative filtering applications. In: Computational Intelligence for Modelling, Control & Automation. IOS Press (1999)
66. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society **39** (1977) 1–38
67. Agrawal, R., Imielinski, T., Swami, A.: Mining associations between sets of items in massive databases. In: Proceedings of the ACM-SIGMOD 1993 International Conference on Management of Data. (1993) 207–216
68. R. Agrawal, R.S.: Mining sequential patterns. In: Proceedings of the International Conference on Data Engineering (ICDE). (1995)
69. Baumgarten, M., Buchner, A.G., Anand, S.S., Mulvenna, M.D., Hughes, J.: User-driven navigation pattern discovery from internet data. web usage analysis and user profiling. In Masand, B., Spiliopoulou, M., eds.: Web Usage Analysis and User Profiling: Proceedings of the WEBKDD'99 Workshop. Lecture Notes in Computer Science 1836. Springer-Verlag (2000) 74–91
70. Padmanabhan, B., Tuzhilin, A.: Unexpectedness as a measure of interestingness in knowledge discovery. Decision Support Systems **27** (1999) 303–318
71. Silberschatz, A., Tuzhilin, A.: What makes patterns interesting in knowledge discovery systems. IEEE Transactions on Knowledge and Data Engineering **8** (1996) 970–974
72. Tan, P., Kumar, V., Srivastava, J.: Selecting the right objective measure for association analysis. Information Systems **29** (2004) 293–313
73. Fu, X., Budzik, J., Hammond, K.J.: Mining navigation history for recommendation. In: Proceedings of the 2000 International Conference on Intelligent User Interfaces, New Orleans, LA, ACM Press (2000)
74. Lin, W., Alvarez, S.A., Ruiz, C.: Efficient adaptive-support association rule mining for recommender systems. Data Mining and Knowledge Discovery **6** (2002) 83–105
75. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Effective personalization based on association rule discovery from web usage data. In: Proceedings of the 3rd ACM Workshop on Web Information and Data Management (WIDM01), Atlanta, Georgia (2001)

76. Sarwar, B.M., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommender algorithms for e-commerce. In: Proceedings of the 2nd ACM E-Commerce Conference (EC'00), Minneapolis, MN (2000)
77. Agarwal, R., Aggarwal, C., Prasad, V.: A tree projection algorithm for generation of frequent itemsets. In: Proceedings of the High Performance Data Mining Workshop, Puerto Rico (1999)
78. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the International Conference on Data Engineering (ICDE'95), Taipei, Taiwan (1995)
79. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Using sequential and non-sequential patterns for predictive web usage mining tasks. In: Proceedings of the IEEE International Conference on Data Mining. (2002)
80. Nakagawa, M., Mobahser, B.: Impact of site characteristics on recommendation models based on association rules and sequential patterns. In: Proceedings of the IJCAI'03 Workshop on Intelligent Techniques for Web Personalization. (2003)
81. Pitkow, J., Pirolli, P.: Mining longest repeating subsequences to predict world wide web surfing. In: Proceedings of Second USENIX Symposium on Internet Technologies and Systems. (1999)
82. M. Deshpande, G.K.: Selective markov models for predicting web-page accesses. In: Proceedings SIAM International Conference on Data Mining. (2000)
83. Aggarwal, C.C., Wolf, J.L., Wu, K., Yu, P.: Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In: Proceedings of the ACM KDD Conference. (1999) 201–212
84. Mirza, B.J., Keller, B.J., Ramakrishnan, N.: Studying recommendation algorithms by graph analysis. Journal of Intelligent Information Systems **20** (2003) 131–160
85. Mirza, B.J.: Jumping connections: A graph theoretic model for recommender systems. MSc Thesis, Virginia Tech (2001)
86. Nakagawa, M., Mobahser, B.: A hybrid web personalization model based on site connectivity. In: Proceedings of the WebKDD Workshop at the ACM SIGKKDD International Conference on Knowledge Discovery and Data Mining. (2003)
87. Burke, R.: Hybrid systems for personalized recommendations. In: Intelligent Techniques in Web Personalisation. LNAI. Springer-Verlag (2005)
88. Smyth, B., Cotter, P.: A personlized television listing service. Communication of the ACM **43** (2000) 107–111
89. Massa, P., Avesani, P.: Trust-aware collaborative filtering for recommender systems. In: Proceedings of International Conference on Cooperative Information Systems. (2004)
90. Mobasher, B., Jin, X., Zhou, Y.: Semantically enhanced collaborative filtering on the web. In Berendt, B., Hotho, A., Mladenic, D., van Someren, M., Spiliopolou, M., Stumme, G., eds.: Web Mining: From Web to Semantic Web. LNAI 3209. Springer-Verlag (2004)
91. Cohen, W.: Fast effective rule induction. In: Proceedings of the 12th International Conference on Machine Learning. (1995)
92. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval **4** (2001) 133–151
93. Yu, K., Xu, X., Ester, M., Kriegel, H.P.: Feature weighting and instance selection for collaborative filtering: An information-theoretic approach. Knowledge and Information Systems **5** (2003)
94. Chee, S.H.S., Han, J., Wang, K.: Rectree: An efficient collaborative filtering method. In Carrol, J.M., ed.: Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery. LNCS 2114. Springer-Verlag (2001) 141–151
95. Pryor, M.H.: The effect of singular value decomposition on collaborative filtering. Dartmouth College, Computer Science Technical Report PCS-TR98-338 (1998)

96. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Incremental svd-based algorithms for highly scaleable recommender systems. In: Proceedings of the Fifth International Conference on Computer and Information Technology. (2002) 345–354
97. Brand, M.: Fast online svd revisions for lightweight recommender systems. In: Proceedings of the 3rd SIAM International Conference on Data Mining. (2003)
98. Tang, T., Winoto, P., Chan, K.C.C.: Scaling down candidate sets based on the temporal feature of items for improved hybrid recommendations. In: Intelligent Techniques in Web Personalisation. LNCS. Springer-Verlag (2005)
99. Volokh, E.: Personalization and privacy. Communication of the ACM **43** (2000) 84–88
100. Canny, J.: Collaborative filtering with privacy. In: Proceedings of the IEEE Security and Privacy Conference. (2002) 45–57
101. Ramakrishnan, N., Keller, B.J., Mirza, B.J., Grama, A.Y., Karypis, G.: Privacy risks in recommender systems. IEEE Internet Computing (2001) 54–62
102. Ziegler, C., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Proceedings of the 14th international conference on World Wide Web. (2005) 22–32
103. B.Smyth, McClave, P.: Similarity vs diversity. In: Proceedings of the 4th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development. (2001) 347 – 361
104. Sheth, B.: A learning approach to personlized information filtering. Masters Thesis, Massachusetts Institute of Technology (1994)
105. Lieberman, H., Selker, T.: Out of context: Computer systems that adapt to, and learn from, context. IBM Systems Journal **39** (2000) 617–632
106. Allan, J., Aslam, J., Belkin, N., Buckley, C., Callan, J., Croft, B., Dumais, S., Fuhr, N., Harman, D., Harper, D.J., Hiemstra, D., Hofmann, T., Hovy, E., Kraaij, W., Lafferty, J., Lavrenko, V., Lewis, D., Liddy, L., Manmatha, R., McCallum, A., Ponte, J., Prager, J., Radev, D., Resnik, P., Robertson, S., Rosenfeld, R., Roukos, S., Sanderson, M., Shwartz, R., Singhal, A., Smeaton, A., Turtle, H., Voorhees, E., Weischedel, R., Xu, J., C.Zhai: Challenges in information retrieval and language modelling: Report of a workshop held in the centre for intelligent information retrieval. ACM SIGIR Forum **37** (2002) 31–47
107. Dai, H., Mobasher, B.: A road map to more effective web personalization: Integrating domain knowledge with web usage mining. In: Proceedings of the International Conference on Internet Computing. (2003) 58–64
108. Cho, Y., Kim, J.: Application of web usage mining and product taxonomy to collaborative recommendations in e-commerce. Expert Systems with Applications **26** (2004) 233–246
109. Niu, L., Yan, X., , Zhang, C., , Zhang, S.: Product hierarchy-based customer profiles for electronic commerce recommendation. In: Proceedings of the 1st International Conference on Machine Learning and Cybernetics. (2002) 1075–1080
110. Billsus, D., Pazzani, M.J.: User modeling for adaptive news access. User Modelling and User-Adapted Interaction **10** (2000) 147–180
111. McCallum, A., Nigam, K.: A comparison of event models for naïve bayes text classification. In: AAAI/ICML-98 Workshop on Learning for Text Categorization, Technical Report WS-98-05, AAAI Press (1998)
112. Koychev, I., Schwab, I.: Adapting to drifting user's interests. In: Proceedings of ECML2000/MLNet workshop on Machine Learning in the New Information Age. (2000)
113. Moukas, A., Zacharia, G.: Evolving a multi-agent information filtering solution in amalthaea. In: Proceedings of the First International Conference on Autonomous Agents. (1997) 394–403
114. : Wikipedia: The free encyclopedia. http:\\en.wikipedia.org\wiki\Main_Page (2000)
115. Lam, S., Reidl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th International WWW Conference, New York (2004)

116. Burke, R., Mobasher, B., Zabicki, R., Bhaumik, R.: Identifying attack models for secure recommendation. In: Beyond Personalization: A Workshop on the Next Generation of Recommender Systems, San Diego, California (2005)
117. Burke, R., Mobasher, B., Bhaumik, R.: Limited knowledge shilling attacks in collaborative filtering systems. In: Proceedings of the 3rd IJCAI Workshop in Intelligent Techniques for Personalization, Edinburgh, Scotland (2005)
118. O'Mahony, M., Hurley, N., Kushmerick, N., Silvestre, G.: Collaborative recommendation: A robustness analysis. ACM Transactions on Internet Technology **4** (2004) 344–377
119. Mahony, M.O., Hurley, N., Kushmerick, N., Silverstre, G.: Collaborative recommendations: A robustness analysis. ACM Transactions on Internet Technologies **4** (2004) 344–377
120. Mobasher, B., Burke, R., Bhaumik, R., Williams, C.: Effective attack models for shilling item-based collaborative filtering systems. In: Proceedings of the WebKDD 2005 Workshop, in conjunction with the ACM SIGKKDD 2005, Chicago (2005)
121. Massa, P., Bhattacharjee, B.: Using trust in recommender systems: an experimental analysis. In: Proceedings of the 2nd International Conference on Trust Management. (2004)
122. Donovan, J.O., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th international conference on Intelligent user interfaces. (2005) 167–174
123. Herlocker, J., Konstan, J., Riedl, J.: Explaining collaborative filtering recommendations. In: Proceedings of ACM 2000 Conference on Computer Supported Cooperative Work. (2000) 241–250
124. Cosley, D., Lam, S.K., Albert, I., Konstan, J.A., Riedl, J.: Is seeing believing? how recommender interfaces affect users' opinions. In: Proceedings of the SIGCHI conference on Human factors in computing systems. (2003) 585–592
125. Swearingen, K., Sinha, R.: Beyond algorithms: An hci perspective on recommender systems. In: Proceedings of the ACM SIGIR Workshop on Recommender Systems. (2001)
126. Sinha, R., Swearingen, K.: The role of transaprency in recommender systems. In: CHI '02 extended abstracts on Human factors in computing systems. (2002) 830–831
127. Smeaton, A., Murphy, N., O'Connor, N.E., Marlow, S., Lee, H., McDonald, K., Browne, P., Ye, J.: The físchlár digital video system: a digital library of broadcast tv programmes. In: Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries. (2001) 312–313
128. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: Proceedings of the tenth International conference on Information and knowledge management. (2001) 247–254
129. D.Billsus, Pazzani, M.J.: Learning collaborative information filters. In: Proceedings of ICML. (1998) 46–53
130. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Discovery and evaluation of aggregate usage profiles for web personalization. Data Mining and Knowledge Discovery **6** (2002) 61–82
131. Basu, C., Hirsh, H., Cohen, W.: Recommendation as classification: Using social and content-based information in recommendation. In: Proceedings of the Recommender System Workshop. (1998) 11–15
132. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems **22** (2004) 5–53
133. Anderson, C., Domingos, P., Weld, D.: Adaptive web navigation for wireless devices. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. (2001) 879–884
134. Wilson, D., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning **38** (1997) 257–286
135. Daelemans, W., van den Bosch, A., Weijters, T.: Igtree: Using trees for compression and classification in lazy learning algorithms. Artificial Intelligence Review **11** (1997) 407–423

Modeling Web Navigation: Methods and Challenges

Craig S. Miller

DePaul University
cmiller@cs.depaul.edu

Abstract. A computational cognitive model of Web navigation is a working computer system that simulates human users searching for items in a Web site. A fully working model must automate aspects of human perception, decision making and physical control. To successfully predict human behavior, these automated processes must be consistent with the cognitive and physical limitations of human users. Predicted behavior might include which links users select, when they select them and when they backtrack to previous pages. In this chapter, the necessary capabilities of a working model are described in detail. These include processes that simulate users scanning a page of links, assessing each link, selecting a link and deciding when to backtrack. Accurately modeling link assessment for a variety of users is critical for successful predictions and is perhaps the greatest challenge in creating a useful model. Several approaches to link assessment are presented. The implementation details of one model are described, which are then evaluated by correlating the model's timing predictions to results from user studies.

1 Introduction

Cognitive models account for human behavior from an information-processing perspective. As a functioning computer program, these models simulate aspects of human perception, cognition and decision making as they accomplish some task. In this way, it offers an account of how humans perform the task. In this case, the task is Web navigation and the goal is to simulate human users navigating a Web site in order to find the items they are seeking.

Web navigation is perhaps the most common strategy for finding an item in a Web site (Katz & Byrne, 2003). Sometimes called "browsing," it involves identifying relevant links on a Web site and selecting those that will likely lead to the sought-after item. Usually several iterations of page scanning and link selection are required before the targeted item is found. Often some backtracking is needed for cases when misleading links are selected.

A contrasting strategy is the use of a site's keyword search facility. Sometimes simply called "search," this method requires the user to specify some query terms that hopefully identify the user's content goal. The site's search facility then returns a page of links relevant to the specified terms. Keyword search requires the user to *recall* relevant terms whereas the principal cognitive skill for Web navigation requires the user to *recognize* relevant terms. While both strategies have their uses, some empirical studies suggest that users more frequently employ Web navigation and, when they do, are more likely to find the targeted item (Katz & Byrne, 2003; Campagnoni & Erlich, 1989).

Both Web navigation and keyword search have an assumption that the user has some kind of content goal that may be ultimately fulfilled on a content page. This goal may be well defined in the form of a specific object or it may refer to a general category of items. This assumption may not be true for some styles of Web interaction. For example, users may choose to interact with a Web site to merely review its contents.

While Web navigation does not account for all the ways in which users may interact with a Web site, it is perhaps the predominate activity. Web designers develop Web sites with the aim of supporting effective Web navigation. With this in mind, a model of Web navigation provides a substantial account for how users interact with Web pages and is of considerable interest to those who design them.

A cognitive model of Web navigation is useful in a variety of ways. Assuming the model reasonably approximates human usage, it predicts human behavior. Its predictions indicate which links users will select and when users backtrack to previous pages. By assigning time costs to its actions, the model can predict the time required to find a targeted item. With little cost, the model can provide predictions for a range of parameters on a variety of structures. The working model can run in place of user studies. Unlike user studies, the model also offers an explanation of human behavior. This insight allows a designer to understand the impact a change in design might have or how a result might generalize to other structures.

In this chapter, I review the Web navigation task and discuss approaches to model it. I start by describing the cognitive activities that support Web navigation and discuss several approaches to modeling each. Then I review one particular model called MESA, including what it models, some evaluation results, and how it has been useful. I present some challenges that still need to be addressed. Finally, I discuss some of MESA's implications on intelligent systems that try to infer the intent of users.

2 Performing Web Navigation

A complete cognitive model of Web navigation must account for the following activities that support the task:

– Visually scanning links on a page
– Assessing links with respect to the user's navigation goal.
– Selecting links.
– Assessing when to return to a previous page to attempt an alternate path.

For now, it will be convenient to analyze these activities separately, but later we will consider examples where these activities strongly interact and mutually determine what strategies are used.

2.1 Visually Scanning a Web Page

When users encounter a navigation page, they typically identify the links on the page and sequentially attend to them. In the simplest case, the page provides a serial list of links that imply a logical order in which they should be evaluated. For example, the links could be arranged vertically from top to bottom and thus imply the order in which many users would scan them.

While many Web pages use a simple serial layout, perhaps most pages present an arrangement of links whose spatial placement by itself does not imply a scanning order. Visual attributes such as motion (e.g. blinking elements), large fonts or bright colors generally attract a user's attention. Faraday (2000) has incorporated these visual attributes into a working model that indicates a plausible order in which users would scan a page. The model also follows a left-to-right, top-to-bottom scan after the starting point has been determined.

Often a Web page groups links that are related to each other and labels the grouping with a higher level category. In these cases, users usually choose to scan the group labels before they consider the link selections within each group (Hornof & Halverson, 2003).

Users' experiences with Web sites may also influence the order in which they scan a page. For example, many users have learned that the content in banner advertisements do not contain content that interest them. Consequently many users skip their links (Benway, 1998) even if they cannot entirely ignore them (Burke, Gorman, Nilsen, & Hornof, 2004).

Also, many Web sites have adopted a consistent scheme for placing links on pages. Top level categories may be placed horizontally at the top of the page and secondary categories may be displayed vertically at the left side of the page. Users who learn this scheme may only scan the links pertinent to their navigation goals.

2.2 Link Assessment

As users attend to each link while scanning a page, they assess the link label with respect to their navigation goal. They gauge how likely the link will lead to the target. Various terms have been used for this subjective measure. They include residue (Furnas, 1997), relevance (Young, 1998) and information scent (Pirolli & Card, 1999).

Sometimes the assessment is trivial. A user may see the exact text or image that precisely matches the navigation goal. In these cases, the assessment can be made based on the superficial properties of the label. For example, if the user is looking for bicycles, the string "Bicycles" or an image of a prototypical bicycle immediately indicates that selecting this link will lead to these items.

Other times the user needs to assess the link label as a category and evaluate the extent to which the user's navigation goal belongs to the category. For example a search for bicycles may involve assessing a link labeled with "Sporting Goods." A useful proxy for category membership is semantic similarity. Analysis tools such as Latent Semantic Analysis (LSA) produce a similarity metric for a pair of phrases (Landauer & Dumais, 1997). These pairs may correspond to the navigation goal (e.g. bicycles) and the label (e.g. "Sporting Goods") and the resulting similarity metric can indicate the label's relevance. LSA has been used to evaluate label quality in Web applications (Blackmon, Kitajima, & Polson, 2003). Later in this chapter, I will further discuss various approaches for estimating link relevance.

2.3 Link Selection Strategies

Assessing relevance of a link label may not in itself determine whether the link will be selected. A link selection strategy may depend on a threshold. If the relevance is above an established threshold, the link is selected. Otherwise, scanning proceeds to the next link for assessment. The threshold may be lowered for a secondary pass if the first pass failed to identify successful links.

An alternate selection strategy is link comparison. A user may first assess several links and then select the link with the highest relevance. Cognitively, the comparison strategy requires more resources than the threshold strategy since the user must remember the highest link value and where the link is located (Miller & Remington, 2004). Determining which strategy is more efficient depends on the quality and distribution of the links labels. For a well designed page that has one relevant link and no misleading links, the threshold strategy is more efficient since the relevant link is selected as soon as its label is assessed.

So far, we have assumed that these three abilities are modular activities. However, they may interact in practice. For example, if the user already knows the visual features for identifying the label, the user may use a preattentive search strategy. For some visual features such as color, the desired target draws the user's attention without requiring a serial scan (Triesman & Souther, 1985). In one study, users were able to identify and select the color-coded link without scanning and assessing the other links on the page (Ehret, 2002). In another study, there is evidence that users can evaluate multiple targets at once if they know the actual text of their target (Hornof & Halverson, 2003).

2.4 Backtracking Strategies

Sometimes users select links that do not lead to the target. With this possibility in mind, they must continually reassess if they are pursuing a path that will lead them to their goal. For one strategy, they may decide to return to the previous page when they no longer see links whose relevance exceeds their selection criterion. Alternatively, they may choose to lower their criterion momentarily to remove any doubt that they have made the wrong selection. In previous work, Roger Remington and I provide accounts of this second strategy (Miller & Remington, 2002). We call this the *opportunistic strategy* since users can explore less likely options on the current page while the opportunity presents itself. If none of these options succeed, they can confidently rule out this path and return to the previous page.

3 Modeling Approaches

There are no comprehensive cognitive models of Web navigation. Some models have focused on how users scan menus (Byrne, 2001; Hornof & Halverson, 2003) and Web pages (Faraday, 2000). Other models account for how users navigate through a sequence of pages (Lynch, Palmiter, & Tilt, 1999; Chi, Rosien, Supattanasiri, Williams, Royer, Chow, Robles, Dalal, Chen, & Cousins, 2003). While some of the models operate on actual Web sites (Lynch et al., 1999; Chi et al., 2003), none consider the visual attributes in a page for constraining the order in which links are evaluated and selected.

Usually the construction of a cognitive model starts with idealized assumptions. For example, the Max Model (Lynch, Palmiter, & Tilt, 1999) assumes that the user always selects the links that lead to the target. It predicts navigation times by summing typical costs of the user actions needed to reach the target. Card, Moran and Newell (1983) show that this approach can be effective for predicting task completion times of practiced users, but it is not clear how well the predictions of the Max Model correspond to actual users navigating a Web site (Pirolli & Card, 2000).

Other approaches use bounded rationality as a guiding principle for constructing a working model. Here the assumption is that people generally act in ways that will efficiently achieve their goals, at least within the bounds of their knowledge, cognitive resources and physical abilities. Bounded rationality and its variants have a long history for their successful application in predictive models (Simon, 1981Anderson, 1990). Rational analysis has led to predictive models of information foraging (Pirolli & Card, 1999). For Web navigation, the SNIF-ACT model is constructed using ACT-R, a cognitive architecture motivated by rational analysis (Pirolli & Fu, 2003). MESA is a cognitive model of Web navigation I developed with Roger Remington (Miller & Remington, 2004). In the next section, I describe our approach to using principles of bounded rationality and abstraction in developing the model. A more detailed description with extensive traces appears in Miller and Remington (2004).

4 The MESA Model

In developing MESA, we followed three principles:

- The limited capacity principle
- The simplicity principle
- The rationality principle

Combining the limited capacity principle and the rationality principle is our approach to bounded rationality. We construct the model using navigation strategies that minimize cognitive resources. For example, as we have noted, the threshold strategy for selecting links minimizes memory requirements (limited capacity principle) while also being effective for most scenarios (rationality principle).

We are also interested in a model that is reasonably easy to understand, implement and replicate. With the simplicity principle, we seek a model that approximates human behavior to produce useful predictions while avoiding complexities that provide only marginal benefits. For example, MESA assumes a fixed time cost for assessing the relevance of each link. While actual time costs certainly vary as a function of label length, label relevance and the user's reading ability, it is not clear if an account of these variations would produce substantially better predictions since the variations may average out in actual usage.

4.1 Modeling the Web Site and Web Browser

At this time, MESA does not interact with actual Web sites. Instead, its simulations run on abstract representations. While these representations do not specify the visual attributes of the site's pages, they indicate site's abstract structure, often called the information architecture.

Figure 1 shows a simple example of an abstract site structure. The rectangle at the top represents a starting page with four links. Each of those links leads to additional pages, each with two links. The shaded rectangle represents the page that has the user's target or navigation goal. Each link is represented with a number from 0 to 1. Based on the label for the link, this number represents the user's subjective assessment of how likely selecting the link will lead to the target. This assessment is made independent of the other links on the page. In this way, a page may have any number of links with relevance assessments close to one.

The numbers correspond to one user's assessment. A different user may evaluate the link labels in a different way. Thus, this representation models a site and a user's interpretation of its link labels with respect to the user's navigation

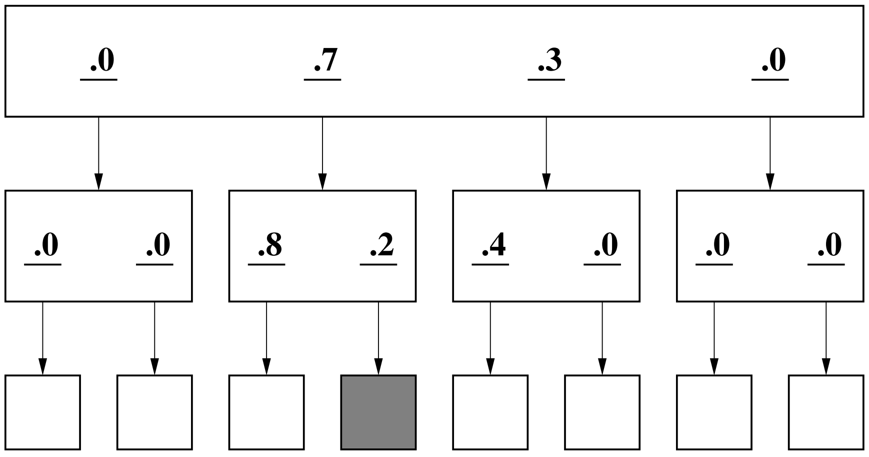


Fig. 1. Abstract representation of a Web site

goal. In the next section, I will review some methods for obtaining relevance assessments based on real link labels.

These structures can represent different cases. For a well designed Web site, the most relevant links lead to the user's goal. This is the case for the top level page for the site in Figure 1. However, the second page (with links marked .8 and .2) has a misleading link. That is, a highly relevant link (.8) does not lead to the target. For this representation, the user must select the less relevant link (.2) in order to find the target. Figure 1 thus represents well designed links at the top level and a misleading link at the second level. By manipulating the values that correspond to relevance assessments, this representation scheme can model well designed sites, where the most relevant links lead to the navigation goal, and poorly designed sites, which consist of many misleading links.

We simulate the most common operations of a Web browser:

- Selecting a link
- Pressing the back button
- Highlighting which links have already been selected

In one study of Web usage, selecting a link and pressing the back button comprise of more than 80% of Web navigation actions for viewing a page.

4.2 MESA Strategies

Applying our principles of Web design, we have implemented the threshold strategy for selecting links and the opportunistic strategy for temporarily delaying when to retreat to a previous page. For this strategy, the threshold may be lowered for a secondary pass in order to explore marginally relevant links before it returns to the previous level.

At this time, MESA makes no commitment on the order in which links are scanned. Eventually we would like to incorporate visual rules that determine this order. For now, we either work with simple designs that imply an order (e.g. a list running from top to bottom) or we randomize the order in our simulations.

The flowchart in Figure 2 summarizes the threshold and opportunistic strategies. Starting with a new page, it scans links in a serial order. If it assess a link whose relevance exceeds the threshold, that link is selected. Otherwise, it continues scanning and assessing links. When it reaches the end of the page, it rescans the page a second time with a lower threshold unless one of the following is true:

- The threshold has already been lowered.
- The model can determine that it did not encounter any marginally relevant links on the first scan.

To recall the presence of a marginally relevant link, the model keeps one memory bit that indicates whether such a link was encountered. To model the short-term memory of a human (limited capacity principle), it loses this memory when it selects a link to scan a new page. In these cases, the model may perform

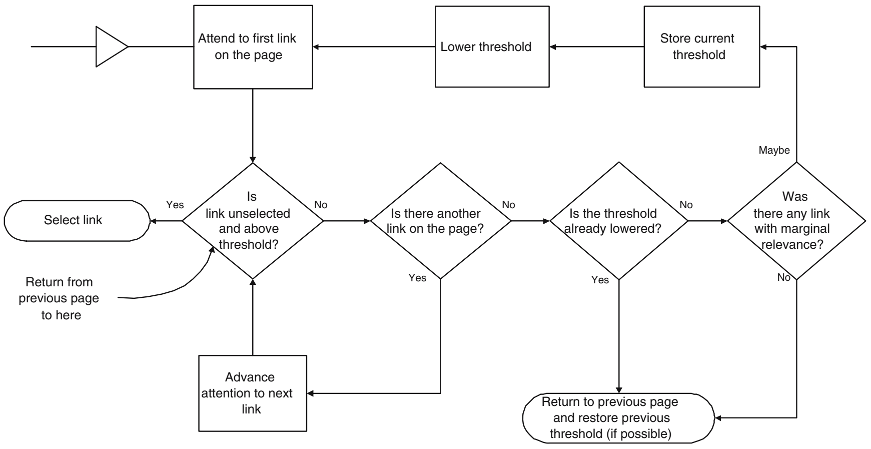

Attend to first link on the page
Lower threshold
Store current threshold
Maybe
Is link unselected and above threshold?
Yes
No
Select link
Is there another link on the page?
No
Yes
Is the threshold already lowered?
No
Yes
Was there any link with marginal relevance?
No
Return from previous page to here
Advance attention to next link
Return to previous page and restore previous threshold (if possible)

Fig. 2. Flowchart summarizing MESA's strategies

a second scan when it returns even if it had not encountered any marginally relevant links on this page.

The model also maintains a limited memory for the threshold at a previous level. If the memory capacity has not been exceeded, the model can restore its threshold to the pre-existing value. We have explored the effects of various memory limits elsewhere (Miller & Remington, 2004).

To illustrate how MESA navigates an example structure, let us consider Figure 1. For this example, we will use .5 as the initial selection threshold. Starting with the first link at the top level (.0), its value is not close to the threshold. Link scanning proceeds to the second link (.7). This value is above the threshold and the model selects it. The next page appears. The first link on this page (.8) is above the threshold. The model selects it but finds that this link does not lead to the target. Returning to the second level, the model scans the next link (.2) but does not select it because it is below the threshold. However, before the model returns to the top level, the opportunistic strategy temporarily reduces the threshold to .1. Rescanning the page, it then selects the second link and finds the target. If the model had not found the target, it would have returned to the top level and continue scanning using the original threshold. At this level, it may reduce the threshold after the initial scan if it still does not find the target.

As the model simulates Web navigation, it employs three basic operations:

– Assess the relevance of a link label
– Select a link
– Return to the previous page (i.e. press the "Back" button)

By assigning plausible time constants to these operations, the model can predict the amount of time needed to find targets in a Web structure. In one of our studies, we obtained good absolute fits with 500 milliseconds for assessing a link's relevance, 2.5 seconds for selecting a link and 1.5 seconds for returning to the previous page.

Not every user employs the threshold and opportunistic strategies like our model. For example, Howes, Payne and Richardson (2002) provide evidence that some users consider the relevance of competing links at the top level for deciding whether to pursue less relevant links at a secondary level. However, it is less clear if a substantial number of users employ this strategy under most circumstances. In the case of the Howes *et al.* study, menu structures consisted of only 2 links per page. Moreover, it appears that many participants might have used strategies consistent with MESA. With this mind, our approach is to continue with the simpler modeling strategies as long as they produce reasonably accurate timing predictions.

Despite the simplicity of MESA's strategies, the model accounts for a broad range of user behavior. If each visited page contains one link with high relevance and the remaining links with low relevance, the model quickly navigates deep into the Web site's structure. Alternatively, if the top level page has several links with high relevance each leading to pages with low relevance links, the model exhibits shallow exploration. Intermediate exploration behaviors are created by modeling various distributions of link relevance across the pages. Thus, the simple threshold and opportunistic strategies yield a range of behaviors as determined by link relevance. Our working hypothesis is that these strategies provide a useful approximation for understanding most users as they navigate Web sites. We may ultimately consider more sophisticated strategies but only if they achieve substantially better correlations to human behavior.

4.3 Assessing Link Relevance

For its simulations, MESA requires relevance assessments for the links in the structure. One method for setting relevance values involves starting with a Web site whose links are ideally labeled. That is, the links that lead to the target are valued at 1.0 and the links that do not lead to the target are valued at 0.0. This structure represents a perfectly designed Web site for a user and a navigation goal. We can then create less ideal sites by adding random noise to the link values. Adding more noise increases the probability that the model selects misleading links, that is, links that do not lead to the target. We have used this approach to show how the model replicates results in a user study that compared the effectiveness of three different structures (Larson & Czerwinski, 1998). Here effectiveness is defined as the average time required to find targets. With sufficient noise, the model's simulated times ranked the effectiveness of the structures in the same order as that of the user study.

A second approach is to employ human raters. For the results in Miller and Remington (2004), we and a third judge rated text labels with respect to a target. Each label-target pair received one of three discrete ratings: probably lead to the target, possibly but unlikely lead to the target, and highly unlikely to lead to the target. By assigning values to these categories (respectively 1.0, 0.5 and 0.0), we obtain numerical averages and variations. More recently, I have collected ratings from larger samples. I will review these results in the next section.

Assigning random noise to ideal relevance values is a low-cost method, but it assumes a general level of quality throughout the site. The use of human raters provides a better assessment but requires costly procedures for collecting the ratings. We are considering automatic methods for assessing label relevance, which I further explore later in this chapter.

5 Evaluation

We have assessed MESA's validity by comparing its predictions to results from human user studies (Miller & Remington, 2004). At the abstract structural level, we have reported how MESA produces results that are consistent with those in several menu selection studies (Miller, 1981; Snowberry, Parkinson, & Sisson, 1999) as well as the Web study by Larson and Czerwinski (1998). We also report our own user study where we asked human participants to find 8 department store items in structures consisting of nearly 500 items. We used our judged ratings (3 raters) to create relevance values for the MESA simulations. To account for variation among the site's human users, we ran the simulations on structures that had relevance values spread along a normal distribution as defined by the average and standard deviation of the judged assessments. In this way, the simulations always used the same relevance values for when the judges agreed on the value but used a range of relevance values for when the judges disagreed. The simulated times were then averaged across 1500 runs (100 for each human participant) producing a total of 24 predicted times (8 targets on three different

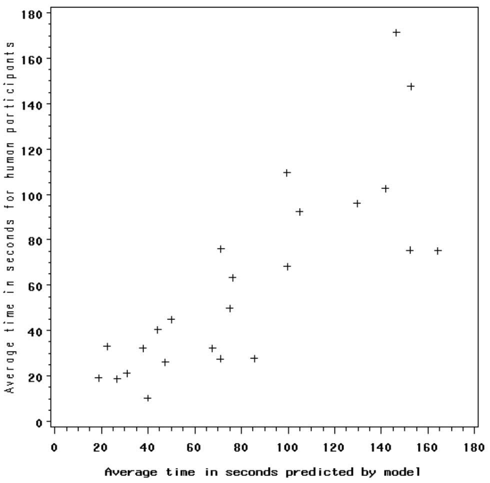


Fig. 3. MESA's predictions compared to actual navigation times

structures). These simulations produced timing predictions that had a Pearson correlation of 0.79 to the navigation times.

While there is some precedent for using expert judgments to evaluate user interfaces (e.g. see Nielsen and Mack, 1994, for a collection of methods involving expert assessments), it is not clear how well these judgments match the behavior of real users. With the goal of obtaining more accurate relevance assessments, I asked 17 human participants to rate the labels for each of the 8 targets. These participants were recruited using the same method as the user study. Even though there was some disagreement between these ratings and the judged ratings (the Pearson correlation is .74), the simulated predictions from these 17 participants only produced a marginally greater correlation (.81) to the navigation times from the user study. Figure 3 shows how the predicted times of the model match those produced by human users. The x-axis shows the times predicted by MESA and the y-axis shows the average time for the participants. To produce the simulated times, the time costs for evaluation, selection and return were respectively 500 milliseconds, 2.5 seconds and 1.5 seconds.

At this time, it is not clear if better predictions will come with more accurate assessments of label relevance or by revising the model so that its strategies better match those of human users. A future goal is to obtain better relevance assessments by having the same human users provide the assessments and the navigation times.

6 Future Directions for Assessing Relevance

Collecting assessments of label relevance from a large number of users is reasonable and necessary for validating the model. However, the use of a large number of human raters is too costly as a routine method for evaluating the quality of a Web structure. Asking human users to rate labels requires about as much time as asking them to perform the actual navigation tasks. Using a small number of expert judges is more feasible but runs a greater risk of being less reliable. Even with a reduced number of human raters, the expense of an exhaustive assessment is considerable. For example, a site with 1000 targets and 100 category labels requires 100,000 assessments.

While it may be possible to effectively use a more manageable representative sample of human-rated assessments, a simulation performing a comprehensive evaluation would need to use some kind of automatic method. At this time, there does not appear to be any method that is currently able to replace human assessments of label relevance. Here I review some approaches, their current shortcomings and possible directions for improvement.

6.1 Co-occurrence of Label and Target

If one has access to documents that contain the site's labels and targets, one can use the frequency of how often the label and target co-occurs as the basis of a similarity measure. A useful measure results by normalizing the frequency with respect to how often the label and the target appear independently. Pirolli and Card (1999) have successfully used co-occurrence to predict user behavior in an information foraging task. More recently, Pirolli and Fu (2003) used the Web for their model of Web navigation.

Successfully using co-occurrence requires that the label and the target frequently appear in the body of documents. Web search engines provide access to a large number of Web pages increasing the likelihood that full label and target names appear with a sufficient frequency. However, some specialty items may still not appear with enough frequency. For example, the site for our user study has the item "Tripod grill" and its relevance to the category "BBQ Tools and Gadgets" needs to be assessed. Unfortunately a Web search engine did not identify many pages with the full names: 94 pages for "Tripod grill", 3 pages for "BBQ Tools and Gadgets" and 1 page for them combined. These numbers are too small to reliably assess relevance. Of course, stemming the words and breaking up the phrases would increase the number of matches, but this would then lose contextual information.

6.2 Latent Semantic Analysis

Latent Semantic Analysis (LSA) provides a similarity measure that could be used for relevance. Like counting co-occurrences, it depends on a body of documents. However, its measure is not fully determined by how often the label and target appear together. Also word order is not considered. Instead it treats the component words as vectors and vector similarity depends not just on co-occurrence but also how often related words appear together.

Without word order as a consideration, LSA requires extended text to to provide the context. Blackmon, Kitajima and Polson (2003) used LSA for diagnosing usability problems in Web sites by describing the navigation goal with a lengthy segment of text (100–200 words). However, the shorter names of labels and targets in the domain of our user study do not appear to be sufficiently rich for successfully applying LSA. When examining the text labels that led to the navigation goals of our user study, the correlation between the relevance ratings produced by LSA and the judged ratings was only .28 and not significant ($p = .18$). These LSA similarity ratings were calculated using the LSA Web site (http://lsa.colorado.edu, accessed February 3, 2003). It is probable that better correlations would come from an analysis based on a larger body of documents containing text more relevant to our domain.

6.3 Wordnet

One shortcoming of using a similarity metric is that it does not distinguish between category membership and other similarity relationships. For example, most users would not select a link labeled "Tricycles" in order to find bicycles. Yet, a simple similarity measure would indicate a high level of similarity between this label and this navigation goal.

Wordnet is an online database of words and phrases that provides relational connections between the entries (Miller, 1995). Among the connections is the hyponymy connection denoting category membership. This relation could be useful in distinguishing between general similarity and category membership. While there exist Wordnet-based measures that could serve as relevance measures (see Pedersen, Patwardhan, and Michelizzi, 2004, for a summary), measures of category membership that could extend to assessing relevance have yet to be developed. Moreover, additional work is needed to extend the database to labels that are not explicitly noted in the database.

6.4 Modeling Variation Among Users

For any of these approaches, it is not enough for them to return measures of relevance that would be useful for information retrieval. If our goal is to model users, a valid method must be able to model a range of users. With human raters, we can simulate user variation by creating multiple sets of assessments whose levels vary as a function of the variation in the human ratings. For automated similarity measures, we need approaches for simulating this variation. For example, we may find that the least common terms in corpora produce the greatest variation of assessments among humans. If so, frequency of occurrence may serve as a good source for simulating assessment variation.

7 Implications for Intelligence Systems That Infer User Intent

Our results indicate that MESA's opportunistic strategy is a useful describing human navigation patterns. This has implications for an intelligent system that tries to infer a user's intent based on the user's link selections. This includes systems that analyze Web server logs with the goal of determining what the user was looking for. It also includes recommendation systems.

Figure 4 shows an example structure where a user's link selections may mislead a system that is attempting to infer the user's intent. Scanning from left to right, the user would select the second link in Page A (valued at .6). Page C is then scanned. On the first pass, no links are selected, but MESA predicts that the user will perform a second pass at a lower threshold that would cause all of these links to be selected. The user returns to Page A only after learning that all of these links do not lead to the selected target. If the user's memory allows it, the threshold is restored to its original value (e.g. .5) when returning to Page A. Otherwise, the third link on Page A (valued at .2) might also be selected as well as two links on Page D. It is only after these choices are exhausted would the user select the highly relevant links that lead to the target (assuming that the user has not given up).

This navigation of the site in Figure 4 illustrates how users may select many links whose labels are only marginally relevant to their navigation goals. In this way, most of the selected links are not good indicators of user intent. If an intelligent system were to consider the site structure and the sequence of link selections, it might do well to disregard (or at least minimize the influence of)

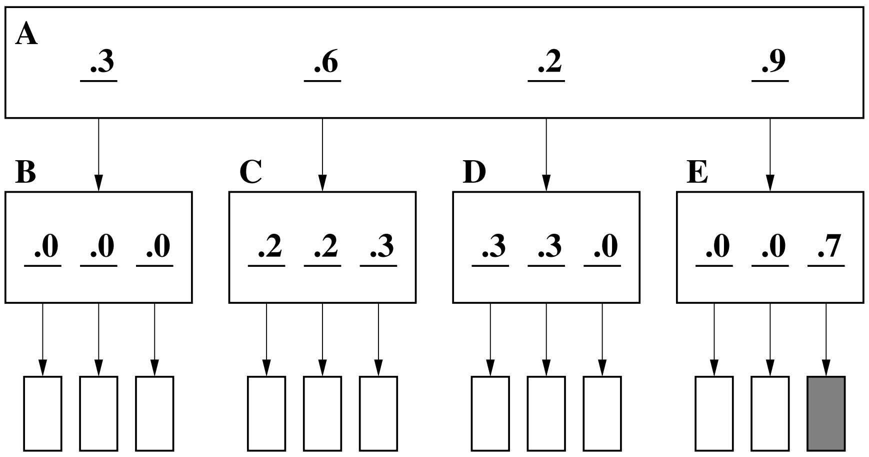


Fig. 4. Example structure that may mislead intelligent systems

link selections made on lower pages if the user eventually returns to a higher level to try other links.

8 Closing Comments

In the long term, we would like to see systems like MESA simulate users on actual Web sites with the goal of providing useful feedback on the accessibility of the site's content. In the place of costly user studies, the model would simulate users and indicate where real users would encounter difficulties. Of course, the construction of this kind of model would need to overcome some challenging obstacles, some of which I have reviewed here.

Still, even in its current form, models like MESA have been useful. Already we have applied MESA to resolve issues addressing the optimal number of links per page. We also envision using MESA early in the design process to compare the effectiveness of different structures. At this point in the development process, the abstract representations are appropriate and relevance values can be determined using a combination of methods. Finally, models like MESA give us an understanding of human behavior. This insight has a range of uses including a better understanding of what may be inferred when a user selects a link.

Acknowledgements

The author thanks Sven Fuchs and Sarabjot Anand for their helpful comments on earlier versions of this chapter. Funding for this work was provided by the Human-Automation Integration Research portion of the Airspace Operation Systems Project of NASA's Airspace Systems Program.

References

Anderson, J. R. (1990). *The adaptive character of thought.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Benway, J. P. (1998). Banner blindness: The irony of attention grabbing on the world wide web. *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting* (pp. 463–467).

Blackmon, M. H., Kitajima, M., & Polson, P. G. (2003). Repairing usability problems identified by the cognitive walkthrough for the web. *Proceedings of the conference on Human factors in computing systems* (pp. 497–504). ACM Press.

Burke, M., Gorman, N., Nilsen, E., & Hornof, A. (2004). Banner ads hinder visual search and are forgotten. *Extended Abstracts of ACM CHI 2004: Conference on Human Factors in Computing Systems.* ACM Press.

Byrne, M. D. (2001). Act-r/pm and menu selection: Applying a cognitive architecture to hci. *International Journal of Human-Computer Studies*, *55*, 41–84.

Byrne, M. D., John, B. E., Wehrle, N. S., & Crow, D. C. (1999). The tangled web we wove: A taskonomy of www use. *Proceedings of CHI'99 Human Factors in Computing Systems* (pp. 544–551). New York: ACM Press.

Campagnoni, F. R., & Erlich, K. (1989). Information retrieval using a hypertext-based help system. *Proceedings of the 12th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 212–220). ACM Press.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Chi, E. H., Rosien, A., Supattanasiri, G., Williams, A., Royer, C., Chow, C., Robles, E., Dalal, B., Chen, J., & Cousins, S. (2003). The bloodhound project: automating discovery of web usability issues using the infoscentp simulator. *Proceedings of the conference on Human factors in computing systems* (pp. 505–512). ACM Press.

Ehret, B. D. (2002). Learning where to look: Location learning in graphical user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 211–218). ACM Press.

Faraday, P. (2000). Visually critiquing web pages. *The 6th Conference on Human Factors & the Web.* Retrieved from http://www.tri.sbc.com/hfweb/faraday/FARADAY.HTM on April 19, 2004.

Furnas, G. W. (1997). Effective view navigation. *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 367–374). ACM Press.

Hornof, A. J., & Halverson, T. (2003). Cognitive strategies and eye movements for searching hierarchical computer displays. *Proceedings of the conference on Human factors in computing systems* (pp. 249–256). ACM Press.

Howes, A., Payne, S. J., & Richardson, J. (2002). An instance-based model of the effect of previous choices on the control of interactive search. *Proceedings of the 24th Annual Meeting of the Cognitive Science Society* (pp. 476–481). Hillsdale, NJ: Lawrence Erlbaum Associates.

Katz, M. A., & Byrne, M. D. (2003). Effects of scent and breadth on use of site-specific search on e-commerce web sites. *ACM Trans. Comput.-Hum. Interact.*, *10*(3), 198–220.

Landauer, T. K., & Dumais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, *104*, 211–240.

Larson, K., & Czerwinski, M. (1998). Web page design: Implications of memory, structure and scent for information retrieval. *CHI'98: Human Factors in Computing Systems* (pp. 25–32). New York: ACM Press.

Lynch, G., Palmiter, S., & Tilt, C. (1999). The Max model: A standard web site user model. *Proceedings of the 5th Annual Human Factors and the Web Conference.* Retrieved from http://zing.ncsl.nist.gov/hfweb/proceedings/lynch/index.html on August 26, 2002.

Miller, C. S., & Remington, R. W. (2002). Effects of structure and label ambiguity on information navigation. *Conference Extended Abstracts on Human Factors in Computer Systems* (pp. 630 – 631). New York: ACM Press.

Miller, C. S., & Remington, R. W. (2004). Modeling information navigation: Implications for information architecture. *Human-Computer Interaction*, *19*(3), 225 – 271.

Miller, D. P. (1981). The depth/breadth tradeoff in hierarchical computer menus. *Proceedings of the Human Factors Society*, Vol. 25 (pp. 296–300).

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, *38*(11), 39–41.

Nielsen, J., & Mack, R. L. (Eds.). (1994). *Usability inspection methods.* Indianapolis: Wiley.

Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). Wordnet::similarity - measuring the relatedness of concepts. *Proceedings of Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics.*

Pirolli, P., & Card, S. (1999). Information foraging. *Psychological Review*, *106*, 643 – 675.

Pirolli, P., & Card, S. (2000). A web site user model should at least predict something about users. *internetworking*, *3*.

Pirolli, P., & Fu, W.-T. (2003). Snif-act: A model of information foraging on the world wide web. *Proceedings of the Ninth International Conference on User Modeling.*

Simon, H. A. (1981). *The sciences of the artificial, second edition.* Cambridge, MA: The MIT Press.

Snowberry, K., Parkinson, S. R., & Sisson, N. (1999). Computer display menus. *Ergonomics*, *26*, 699–712.

Triesman, A., & Souther, J. (1985). Search asymmetry: A diagnostic for preattentive processing of separable features. *Journal of Experimental Psychology*, *114*(3), 285 – 310.

Young, R. M. (1998). Rational analysis of exploratory choice. In M. Oaksford, & N. Chater (Eds.), *Rational models of cognition* (pp. 469–500). Oxford University Press.

The Traits of the Personable

Naren Ramakrishnan

Department of Computer Science,
Virginia Tech, VA 24061, USA
naren@cs.vt.edu,
http://people.cs.vt.edu/~ramakris

Abstract. Information personalization is fertile ground for application of AI techniques. In this article I relate personalization to the ability to capture partial information in an information-seeking interaction. The specific focus is on personalizing interactions at web sites. Using ideas from partial evaluation and explanation-based generalization, I present a modeling methodology for reasoning about personalization. This approach helps identify seven tiers of 'personable traits' in web sites.

1 Introduction

Web personalization has become so pervasive that, as an enabling technology, it transcends a constantly growing set of applications in electronic commerce, knowledge management, information access, social schemes for decision making, and user interfaces. In some application contexts, personalization has come to occupy such a central role that it is now difficult to imagine a user experience without it. For instance, Riedl [1] estimates that there are at least 23 different types of personalization at Amazon's e-commerce site!

The word 'personalization' lends itself to many individual interpretations, all of which indisputably provide some form of customization. There are broadly two schools of thought. The first adopts the viewpoint that to qualify as personalization research, an approach must employ some form of *user model*, obtained implicitly or explicitly. The notion of user model is itself a rich one, and can range from simple aggregations of usage patterns by analyzing weblogs [2,3] or transaction records to richer representations of capabilities, interests, and preferences, e.g., see research in adaptive hypermedia [4]. The second school of thought de-emphasizes user models in favor of a flexibility of information access, typically via multiple interaction pathways or dialogs through a site. Here the idea is that by placing fewer constraints on interaction, the user experience can be more personalized, although there is no 'understanding' of the user *per se* by the system. Examples here are faceted browsing interfaces [5] and conversational systems [6].

This chapter grew out of an attempt at trying to answer the question: *What does it mean for a web site to be personable?* Rather than stop at the cliched observation that there are many forms of personalization [7], we are interested in deriving some long lasting attributes of personalization solutions, especially with an eye toward accommodating both schools of thought mentioned above. Of course, this is a difficult goal and we will necessarily make some simplifying assumptions. Nevertheless, the ideas described here are not too abstract as they capture a wide variety of practical personalization situations, referred to here as *traits*.

2 Personalizing Interaction

Let us start with the working assumption that a website is personable if it allows a user's information seeking goals to be met *effectively*. A user's interaction with a web site can be thought of as a dialog between the user and the underlying information system, using the communication facilities afforded by the web site. Thus, when the user clicks on a hyperlink or submits data in a form, information is implicitly communicated from the user to the system. In response, the system presents information back to the user (including opportunities for further user input). Many such dialogs happen in a browsing context.

Consider a hierarchical US Congressional website, where the user progressively makes choices of politician attributes—*state* at the first level, *branch* at the second level, followed by levels for *party*, and *district/seat*—by browsing (see Fig. 1). Imagine how a user would pursue the following tasks:

1. Find the webpage of the Democratic Representative from District 17 of Florida.
2. Find the webpage of each Democratic Senator.

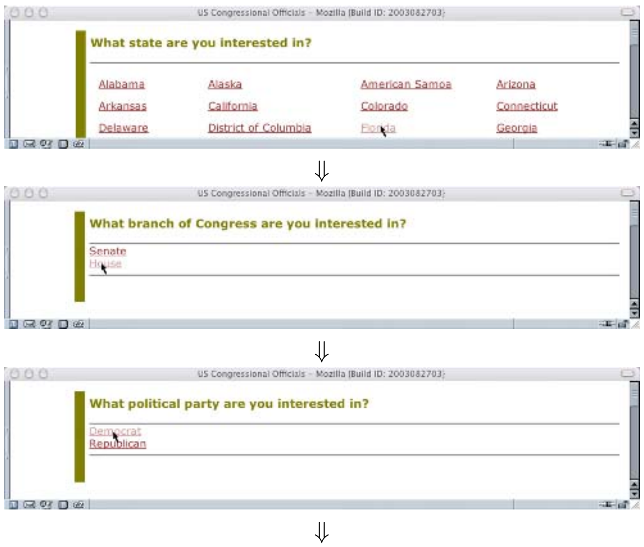
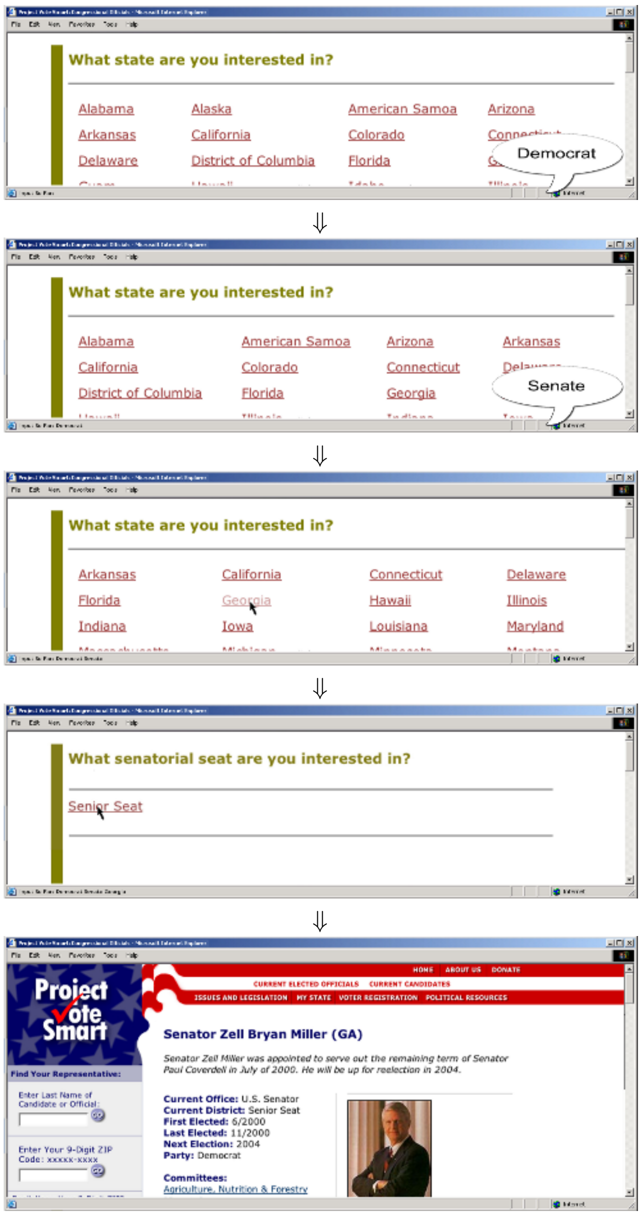

US Congressional Officials - Mozilla (Build ID: 2003082703)
What state are you interested in?
Alabama
Alaska
American Samoa
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
District of Columbia
Florida
Georgia
What branch of Congress are you interested in?
Senate
House
What political party are you interested in?
Democrat
Republican

Fig. 1. In-turn interaction with a website



What state are you interested in?

Alabama	Alaska	American Samoa	Arizona
Arkansas	California	Colorado	Connecticut
Delaware	District of Columbia	Florida	Georgia

What state are you interested in?

Alabama	American Samoa	Arizona	Arkansas
California	Colorado	Connecticut	Delaware
District of Columbia	Florida	Georgia	

What state are you interested in?

Arkansas	California	Connecticut	Delaware
Florida	Georgia	Hawaii	Illinois
Indiana	Iowa	Louisiana	Maryland

What senatorial seat are you interested in?

Senior Seat

Project Vote Smart

HOME ABOUT US DONATE

CURRENT ELECTED OFFICIALS CURRENT CANDIDATES

ISSUES AND LEGISLATION MY STATE VOTER REGISTRATION POLITICAL RESOURCES

Find Your Representative:

Enter Last Name of Candidate or Official:

Enter Your 9-Digit ZIP Code: xxxxx-xxxx

Senator Zell Bryan Miller (GA)

Senator Zell Miller was appointed to serve out the remaining term of Senator Paul Coverdell in July of 2000. He will be up for reelection in 2004.

Current Office: U.S. Senator
Current District: Senior Seat
First Elected: 6/2000
Last Elected: 11/2000
Next Election: 2004
Party: Democrat

Committees:
Agriculture, Nutrition & Forestry

Fig. 2. A web session illustrating the use of out-of-turn interaction in a US congressional site. This progression of interactions shows how the (Democrat, Senate, Georgia, Senior) interaction sequence, which is indescribable by browsing, may be realized. In steps 1 and 2, 'Democrat' and 'Senate' are spoken out-of-turn (resp.) when the systems solicits for state. In step 3, the user clicks 'Georgia' as the state (an in-turn input). The screen at step 4 shows that only the Senior Senator from Georgia is a Democrat, and leads the user to his homepage.

The first task can be satisfied by typical drill-down browsing because it involves supplying responsive information at each level (click ‘Florida’ first, ‘House’ next, and so on). Such an interaction where the user merely clicks on presented hyperlinks is called an *in-turn* interaction (see Fig. 1). The word ‘in-turn’ is drawn from conversational nomenclature and refers to a turn-taking scenario where the website queries for a certain aspect of politician at each turn, and the user makes choices for these aspects in the order in which they are requested. Notice that each hyperlink click, or in-turn input, communicates *partial information* about the desired politician. Achieving the second task by communicating only in-turn information would require a painful series of drill-downs and roll-ups, in order to identify the states that have at least one Democratic Senator, and to aggregate the results. While the user has partial information about the desired politicians, s/he is unable to communicate it by in-turn means. For instance, at the outset the user would like to specify that she is interested in Democratic Senators whereas the website is requesting a specification of state instead.

Out-of-turn interaction is our solution to support flexible communication of partial information not currently requested by the system. One manifestation is to allow the speaking of utterances into the browser. Fig. 2 describes how we can use it to achieve Task 2 above. At the top level of the site, the user is unable to make a choice of state, because s/he is looking for states that have Democratic Senators. S/he thus speaks ‘Democrat’ out-of-turn, causing some states to be pruned out (e.g., Alaska). At the second step, the site again solicits state information because this aspect has not yet been communicated by the user. The user speaks ‘Senate’ out-of-turn, causing further pruning (e.g., of American Samoa), and retaining only regions that have Democratic Senators. At this point, the goal has been achieved (the user notices 31 states satisfying the criteria), and s/he proceeds to browse through the remaining hyperlinks. Notice that these are contextually relevant to the partial information supplied thus far, so that when ‘Georgia’ is clicked, there is only one choice of seat (Senior) implying that the other Senatorial seat is not occupied by a Democrat.

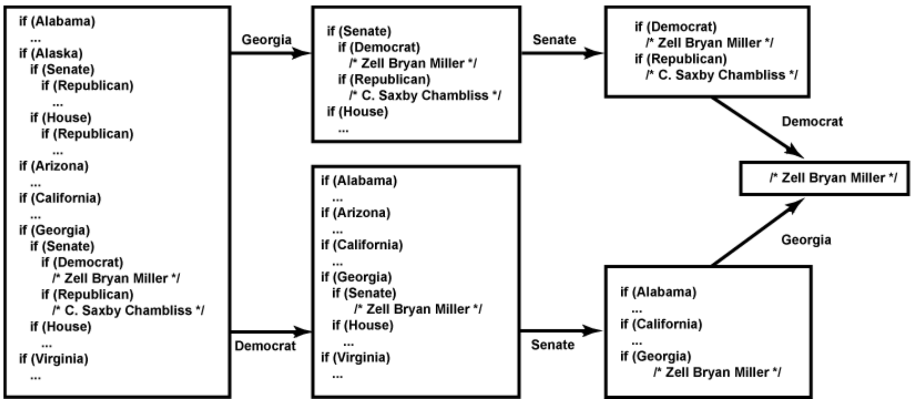
Out-of-turn interaction should be contrasted with the typical solution adopted in today’s websites, namely *faceted browsing* that enumerates all possible dialog options in the site design, i.e., in-turn. Directly supporting all permutations of facets in the browsing structure in this manner results in a cumbersome site design, with a mushrooming of choices at each step. Out-of-turn interaction must also be viewed distinctly from search engines, which are characterized by specification of *complete* information. In this case, the interaction is terminated by returning a flat list of results, which curbs the user-site dialog. OOT interaction *continues* the dialog and situates future dialog choices (e.g., hyperlink options) in the context of previously supplied partial information.

Since out-of-turn interaction is unintrusive, optional, and preserves the closed nature of navigation through the site, it can be interleaved with hyperlink clicks as many times as desired (the stateful implementation of these interactions described below also allows the user to utilize the back-button for backtracking purposes). Such an interaction, with both in-turn and out-of-turn elements, is called a *mixed-initiative interaction* [8,9]. An interaction with only in-turn inputs, in contrat, can be referred to as a *site-initiated* interaction.

2.1 Representational Approach

Interestingly, both site-initiated and mixed-initiative interactions can be supported in the same dialog programming model! To see how, it is helpful to think of modeling a website as the program of Fig. 3 (left) where the nesting of conditionals reflects the hierarchical hyperlink structure and each program variable denotes a hyperlink label. For an in-turn sequence, the top series of transformations in Fig. 3 depicts what we want to happen. For the interaction of Fig. 2, the bottom series of transformations depicts what we want to happen (For ease of presentation, we are considering only the party, state, and branch of Congress aspects). Notice that both sequences start and end with the same representation, but take different paths.

The first sequence of transformations corresponds to *interpreting* the program in the order in which it is written, i.e., when the user clicks on 'Georgia,' that variable is set to one and all other state variables (e.g., 'Alabama') are set to zero, and the program is interpreted. This leads to a simplified program that now solicits for branch of congress. The second sequence of transformations involves 'jumping ahead' to nested program segments and simplifying them even before outer portions are evaluated. Such a non-sequential evaluation is well known in the programming languages literature to be *partial evaluation* ([10]; see Fig. 4), a technique for specializing programs given some (but not all) of their input. Thus, when the user says 'Democrat' out-of-turn, the program is partially eval-



```
if (Alabama)
  ...
if (Alaska)
  if (Senate)
    if (Republican)
      ...
  if (House)
    if (Republican)
      ...
if (Arizona)
  ...
if (California)
  ...
if (Georgia)
  if (Senate)
    if (Democrat)
      /* Zell Bryan Miller */
    if (Republican)
      /* C. Saxby Chambliss */
  if (House)
    ...
if (Virginia)
  ...
```

```
if (Senate)
  if (Democrat)
    /* Zell Bryan Miller */
  if (Republican)
    /* C. Saxby Chambliss */
if (House)
  ...
```

```
if (Democrat)
  /* Zell Bryan Miller */
if (Republican)
  /* C. Saxby Chambliss */
```

```
/* Zell Bryan Miller */
```

```
if (Alabama)
  ...
if (Arizona)
  ...
if (California)
  ...
if (Georgia)
  if (Senate)
      /* Zell Bryan Miller */
  if (House)
    ...
if (Virginia)
  ...
```

```
if (Alabama)
  ...
if (California)
  ...
if (Georgia)
      /* Zell Bryan Miller */
```

Fig. 3. Staging dialogs using program transformations. The top series of transformations mimic an in-turn dialog with the user specifying (Georgia: Senate: Democrat), in that order. The bottom series of transformationscorrespond to a mixed-initiative dialog where the user specifies (Democrat: Senator: Georgia), in that order.

```c
int pow(int base, int exponent) {
  int prod = 1;
  for (int i = 0; i < exponent; i++)
    prod = prod * base;
  return prod;
}
```

```c
int pow2(int base) {
  return (base * base);
}
```

Fig. 4. Illustration of the partial evaluation technique. A general purpose `power` function written in C (left) and its specialized version (with `exponent` statically set to 2) to handle squares (right). Automatic partial evaluators (e.g., C-Mix) use techniques such as loop unrolling and copy propagation to specialize given programs.

uated with this variable set to one (and 'Republican' set to zero). The simplified program continues to solicit for state at the top level, but some states are now removed since the corresponding program segments involve dead-ends. Notice that since PE can be used for interpretation, it can support the first interaction sequence as well.

This simple example shows that what is important is a representation of interaction and an expressive operator (PE) for supporting personalization. We say that a representation is *personable* for a user's information-seeking activity if there is a sequence of partial evaluations of the representation that can support the activity.

A realistic dialog model for interacting with websites requires a complete suite of representation and transformation options, for details see [11]. In addition, there are often interesting dependencies underlying attributes that should be harnessed in the personalization system. For instance, if the user says 'Senior seat,' he is referring to a Senator, not a Representative. Saying 'North Dakota' and 'Representative' in the current political landscape defines a unique member of Congress (no party information is needed), and so on. This is very similar to query expansion strategies utilized in information retrieval systems or association rules applied to web site restructuring [2]. For instance, the association rule 'Senior Seat $\Rightarrow$ Senator' holds with confidence 100% in the site structure, immediately suggesting a possible expansion of the input. In [11] we generalize these ideas and present a theory of 'staging transformations' that helps reason about what partial input has been specified thus far, whether it is legal, whether such input can be expanded, and perhaps even remove the need for further interaction. Essentially, we can think of staging transformations as a combination of site transformations and pruning operators, based on partial input. The cited reference further describes robust and scalable XML-based technology for large websites as well as user studies with this approach.

3 More Choices of Representations

Partial evaluation is one way to exploit partial information via a representation. Explanation-based generalization (EBG) is another. Even though they are computationally equivalent [12], we will begin by making a distinction and later show the implications of their equivalence for personalization.

With PE, a user experiences personalization because the site allows him to provide partial information. With EBG, a user experiences personalization because the site knows some partial information about him. EBG is thus best understood here as a technique that incorporates partial information *prior* to a user interaction, whereas PE incorporates partial information *during* a user's interaction.

We introduce EBG by considering a very different form of personalization. Consider a book-reader (Linus) revisiting the amazon.com website; a greeting prompts 'Welcome back Linus.' After Linus selects a book for purchase, the website skips the questions for credit card and shipping address when processing the order. This is presumably because the answers to these parts of the interaction are being reused from a previous session. Admittedly, this is a useful form of personalization.

```
Book Selection:
if (Mystery)
 if (Harry Potter)
    .........
else if (Science)
 if (John Nash)
    .........
Payment:
if (MasterCard)
   .........
else if (Visa)
   .........
Shipping Options:
if (Fedex)
   .........
```

```
Book Selection:
if (Mystery)
 if (Harry Potter)
    .........
else if (Science)
 if (John Nash)
    .........
```

Fig. 5. (left) Default interaction representation experienced by Amazon users. (right) Interaction representation experienced by Linus. Lines such as '**Payment:**' are comments intended to show program structure.

Fig. 5 shows two representations, the default representation seen by Amazon users and the representation experienced by Linus. It is as if the site has performed some 'free' partial evaluations just for Linus! According to our original definition, both representations are personable for Linus's activity but Linus has to provide two extra pieces of information with the representation of Fig. 5 (left). Per EBG terminology, we say that there is a difference between them in terms of *operationality*. Operationality deals with the issue of whether the site should remember Linus's credit card and payment information or whether it should require Linus to supply it during every interaction. This dilemma is actually at the heart of EBG.

3.1 Using EBG

Before we study EBG in more detail, we will make some preliminary observations. The above dilemma is actually a dilemma for the designer of the personalization system and reduces to the problem of identifying templates of interaction for users. A template — such as the returning customer template — defines a starting point for a user interaction and identifies the program variables that can be involved in the interaction. The tradeoff in designing templates is between the partial evaluations performed by the site (in the template) before the interaction begins and the partial evaluations conducted by the user during the interaction.

We can appreciate the difference by considering more users than just Linus. If the design is set up so that the site performs most of the partial evaluations, then a lot of templates will be needed to support all possible users. Each template provides a considerable amount of personalization but every user has to determine the right template for his interactions. A mushrooming of template choices can cause frustrations for users. Conversely, we can attempt to reduce the number of templates but then some users might find that there is no template that *directly* addresses their information-seeking goals. They might then proceed to use a default vanilla template such as Fig. 5 (left) (assuming that it is supported). Such users may be able to satisfy their goals but will experience longer interaction sequences and a not-so-personalized interaction. The trick is to compress many intended scenarios of interaction into a few template structures.

EBG is a systematic way to cluster the space of users and to determine dense regions of repetitive interactions that could be supported. In Amazon, one important distinction is that made between returning customers and new customers. The top-level prompt at the site makes this distinction (this is automated with cookies) and transfers are made to different interaction sequences.

How and why did Amazon decide on these two templates? Why not a distinction such as ‘reading for pleasure versus reading for business or education?’ Or, ‘students versus professionals?’ Two issues are important here. First, given a customer, can the right template be determined *easily*? Determining if a customer is a new or returning customer is admittedly easier to automate than determining if the person reads for pleasure! Second, the distinctions used for templating interactions should translate into significantly different interaction sequences. Else, the distinction is useless in practice. In the case of the returning customer, for instance, Amazon can provide more personalized recommendations and exhibit a greater understanding of the customer’s preferences and habits. Balancing these considerations is a long-studied problem in EBG; it is interesting that it surfaces in such a natural way in the personalization context.

At this point it should be clear that PE and EBG support different types of personalization. While PE addresses the expressiveness with which a user can supply partial information to the system, EBG addresses the expressiveness by which the system exploits partial information about the user. While with PE we assume that the user provides the partial information in the current visit, EBG requires past navigation experiences to create ‘templates’ which are then

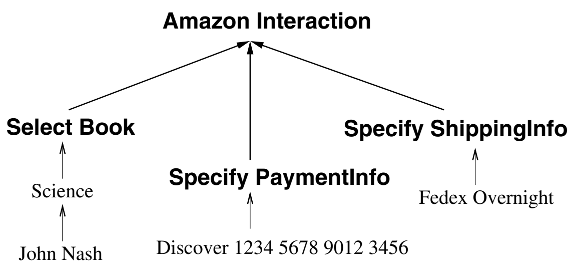


Fig. 6. Explaining a user's interaction as completing an information-seeking task

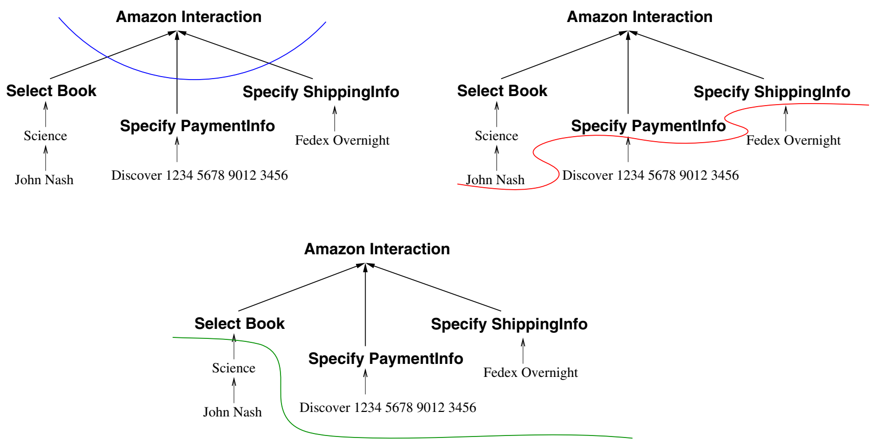


Fig. 7. Different choices of operationality boundaries lead to different templates of personalized interaction

operationalised. Hence EBG is more aligned toward the web mining approach to personlisation [3], involving an offline model building and then an online application of the model.

3.2 Operationality Considerations

EBG is an approach to reason from specific scenarios of interaction to general templates of interaction that should be supported. A user's unpersonalized interaction with a web site is observed and a general template is derived from it. The first step is to use a domain theory to explain the user's interaction. For our purposes, a domain theory captures the site layout, task models, browsing semantics, and their role in information-seeking interactions. Explaining a user's successful interaction at a site with respect to the domain theory will help identify the parts of the interaction that contribute to achieving the personalization objectives. DeJong [13] shows that an explanation can be viewed as a tree where each leaf is a property of the example being explained, each internal node models an inference procedure applied to its children, and the root is the final conclusion supported by the explanation (namely, that the scenario was an example of successful interaction). The explanation tree is used to define a space of personable representations. Searching within this space is the second step in EBG and is called operationalization.

Consider that Linus first used the Amazon site to select a book about John Nash (which he found by browsing through the Science section of the site), paid with his Discover card, and chose Fedex to ship the book. Explaining this interaction of Linus would lead to the proof tree shown in Fig. 6. The tree shows how Linus satisfied the requirements of an Amazon interaction; in this case, by satisfying the requirements for selecting a book, specifying a payment information, and specifying his shipping details. Each of these sub-requirements were in turn satisfied by particular interaction sequences. Operationalization can then be thought of as drawing a cutting plane through the explanation tree. Every node below the plane is too specific to be assumed to be part of all scenarios. The structure above the plane is considered the persistent feature of all usage scenarios and is expressed in the personalization system design. The user is then expected to supply the details of the structure below the plane so that the proof can be completed. Recall that since the proof below the plane is provided by the user's clicks and selections, it can be performed in a mixed-initiative manner.

Fig. 7 shows three ways of drawing a plane through the tree of Fig. 6. The top left really draws the plane at the level of an Amazon interaction, implying that the site will capture no personalization aspects. Every detail is meant to be supplied by the user in his interaction. It is not even assumed, for instance, that the user will buy a book. This gives us the vanilla template that caters to all users. The top right of Fig. 7 draws the cutting plane to include the selection of the book as subsumed by the system, leaving the payment and shipping address to be supplied by the user. This is obviously a very strange notion of operationality! The template resulting from this option would be appropriate only if the same John Nash book is to be purchased over and over again with different credit card and shipment options! The bottom slice of Fig. 7 is probably the reasonable one where the payment and shipping options are subsumed by the system, leaving the user to select the book. It recognizes the fact that in a future interaction, the user is likely to purchase a different book.

Deriving a generalized template of interaction also depends on the class of users it is intended to support. Is the template obtained from Linus supposed to apply only to *his* future interactions or can it be applied to other users as well? Once again, there is a tradeoff. For instance, if we have multiple users in mind then Fig. 7 (top right) no longer looks silly. Implementing this template amounts to creating a 'If you would like to buy the John Nash book, click here to give payment options' link. Contrarily, Fig. 7 (bottom) would be strange here since payment information and shipping details are not transportable across users.

After a template is derived, we have the option of explaining another user's interaction and deriving a new template, if this user's interaction is not well captured by the existing template. As mentioned earlier, we need to be careful about an explosion in the number of templates if this process is repeated. Typically, the default vanilla representation is always retained as one of the templates since there will be many users about whom the site has no prior information.

3.3 Domain Theories for Information-Seeking Interactions

Operationality is thus a matter of utility and an example corresponds to a scenario of interaction. We can evaluate operationality choices by conducting usability studies and determining the coverage of templates; example scenarios of interaction can be obtained by observation and think-aloud protocols. But where do domain theories come from?

While there is significant understanding of information-seeking interactions, there are no large, pertinent, domain theories available for the studies considered here. In [14], we handcrafted a domain theory for reasoning about interactions at the 'Pigments through the Ages' website (http://webexhibits.org/pigments) and used it with EBG to design a personalization system. Pigments is a website that uses pigment analysis catalogs to identify and reveal the palettes of painters in different eras and genres. The domain theory involved an explicit crawl of the site and a 'Background' webpage at the site that outlined a schema for how the website should be used. A group of 10 participants were identified and, after a period of acquaintance, were asked to identify one specific query (or analysis) and use the facilities at the site to answer their query. The exact interaction sequences (including clicked hyperlinks, manual information integration) was recorded for all the participants and then explained using the domain theory. This process revealed that starting from either artists, paintings, or eras, the users systematically browsed through subcategories or compared palettes to arrive at the relevant pigments (used by that artist, in the painting, or in that era, respectively). Furthermore, all pigments share common modes of information seeking, such as browsing through their history of use, procedures for preparation, and technical details of their chemical composition. We hence operationalized the explanation structure(s) as two function invocations in sequence, the first to determine an appropriate pigment category, and the second to browse through the entries in that category by various means. We thus arrived at a single structure in support of all the 10 scenarios. This structure was then evaluated with a set of 15 (different) users who provied 35 scenarios, all except two of which passed our test. The two unrealizable scenarios involved ambiguity of the query that required more contextual information than was modeled in our study.

At the end of this process, there is some optimism that domain theories can be prototyped for certain recurring themes of information-seeking interactions. Besides supporting the construction of explanations, domain theories can help in organizing software codebases for information system design. In other application domains e.g., voice interface design and directory access protocols, this form of codebase organization is already taking place. For instance, commercial speech recognition APIs provide support for task-oriented dialogs (e.g., confirmations, purchase order processing) that make it easy to prototype applications. Such an organization will greatly benefit the study of information personalization.

4 Personable Traits

I have presented two ways to think about personalization; both represent an information-seeking interaction and exploit partial information to deliver a customized experience. Together, they can help capture a variety of personalization scenarios. The EBG viewpoint is more prevalent than the PE viewpoint because the way EBG harnesses partial information lends better to implementation technologies. These observations point us to identifying the expressiveness in which partial information can be utilized by and communicated to an information system.

In Fig. 8, I identify seven tiers of personable traits along such an axis, from most simple to most sophisticated. Alongside each tier is also listed the primary way in which partial information is modeled and harnessed (PE or EBG or both). In reading the following paragraphs, the reader should keep in mind that the presence of EBG is a situation where the site knows something about the user whereas PE captures a situation where the user conveys something to the site. It should also be remarked that many of the personalization solutions surveyed here do not have explicit EBG or PE leanings; it is only our modeling of interaction that permits thinking of them in this manner.

Improving the Addressability of Information	**PE+EBG**
Dialog Structuring and Management	**PE+EBG**
Context Creation and Use	**PE+EBG**
Abstract Interaction	**PE**
User Profiling	**EBG**
Flexible Interaction	**PE**
Remembrance	**EBG**

Fig. 8. Seven tiers of personalization, from simplest (bottom) to most sophisticated (top)

Remembrance

This is an EBG mode of exploiting partial information and refers to the case where simple attributes of a user are remembered, such as credit cards and shipment options. Amazon is a prime example; Citibank Inc. used to provide a toolbar that provided the same functionality. The partial information is thus being exploited in a per-user manner. Web sites that capture and summarize simple form of interaction history (e.g., top 10 visited pages) also fall into this category. Here, explanations from multiple user sessions are operationalized at the leaf level into a single template. This enables a type of personalization that is not specific to any user. For an EBG technique that can support this form of specialization, see [15].

Flexible Interaction

This is a PE mode of personalization and supports simple forms of mixed-initiative interaction. The partial information is expected to be supplied by the user and personalization enhances the way in which it can be supplied. A good example is websites that allow the provision of expected, but out-of-turn information, such as in the US Congress application described earlier. Voice-activated systems are more advanced than websites in their support for this type of personalization [16].

User Profiling

Our third tier is another example of EBG and is considerably more involved than remembrance. Here, what the site knows about a user is not restricted to simple attribute-value information but is actually a sophisticated model of prior interactions. For instance, Amazon suggests 'Since you liked Sense and Sensibility, you will also like Pride and Prejudice.' A user's prior interaction is captured and explained. The explanation is operationalized at the level of an internal representation, to be used in a future interaction. This form of personalization has become very popular and many machine learning techniques have been used to induce the internal representation (e.g., to learn a profile of the user). Some of these techniques are now very sophisticated and try to work with many implicit indicators.

Abstract Interaction

Just as user profiling extends remembrance in an EBG mode, abstract interaction extends flexible interaction in a PE mode. Here the partial information that a user can supply is not restricted to values for program variables but can be some abstract property of her interaction. For instance, the user could be interested in movies that featured the lead actor in *Titanic*, but may be unable to frame her partial information as 'movies where Leonardo Di Caprio acted.' I am not aware of any websites that provide such a functionality in any general way. Transformation techniques for supporting such abstract interpretation are also scarce (but see [17,18]).

Context Creation and Use

This tier of personalization involves both EBG and PE. An example is the shopping basket at Amazon that allows a user to begin an interaction (PE) and save the state of the interaction to be resumed later (EBG). When the user returns to the site, the shopping basket can be checked out by providing the payment and delivery information. The ultimate goal of this tier is to use context creation capabilities to help stage interactions. In many cases such staging naturally breaks down into a context creation phase and a context usage phase.

Dialog Structuring and Management

I have said that EBG and PE utilize partial information in different ways. However, if the operationality boundary is moved down, then information meant to be supplied by the user becomes prior knowledge already known to the site. This shows that 'designing a personalization system' versus 'using a personalizaton system' is quite an artificial distinction. The former just corresponds to choosing a level of operationality (a partial evaluation, of the domain theory), and the latter corresponds to capturing user requests (again, via further partial evaluations, in this case of the template). This argument leads to the equivalence between EBG and PE established in [12]. This tier of personalization removes the distinction between EBG and PE and the interaction resembles more a dialog, with all the associated benefits of a conversational mode. There are not many websites that support such a tier of personalization but this problem has been studied in other delivery mechanisms such as speech technologies [8].

Improving the Addressability of Information

The holy grail of personalization is to provide constructs that improve the addressability of information. Consider how a person can communicate the homepage of, say, the *AI Magazine* to another. One possibility is to specify the URL; in case the reader is unaware, the URL is quite lengthy. Another is to just say "Goto google.com, type *AI Magazine*, and click the 'I'm feeling Lucky' button." The advantage of the latter form of description is that it enhances the addressability of the magazine's webpage, by using terms already familiar to the visitor. This tier of personalization thus involves determining and reasoning about the addressability of information as a fundamental goal, before attempting to deliver personalization. All the previous tiers have made implicit assumptions about addressability. Solutions in this tier take into account various criteria from the user (or learn it automatically from interactions) and use them to define and track addressability constraints. Such information is then used to support personalization. This helps exhibit a deeper understanding of how the user's assumptions of interaction dovetail with his information-seeking goals. The first steps toward understanding addressability have been taken [19]. However, the modeling of interaction here assumes a *complete information* view, rather than partial information.

5 Discussion

My view of personalization is admittedly a very simple one. It only aims to capture the *interaction* aspects underlying a personalized experience and not many others such as quality, speed, and utility. For instance, Amazon's recommender might produce better recommendations than some other bookseller's but if they have the same interaction sequences, then the modeling methodology presented here cannot distinguish between them. The contribution of the methodology is that by focusing solely on modeling interaction, it provides a vocabulary for reasoning about information-seeking. One direction of future work is to prototype software tools to support the types of analyses discussed above (in a manner akin to [20]).

While I have resisted the temptation to unify all meanings of the word 'personalization,' I will hasten to add that EBG and PE are only two ways of harnessing partial information. Any other technique that addresses the capture, modeling, or processing of partial information in the context of interactions will readily find use as the basis for a personalization system. The operative keyword here is, thus, *partial.* A long-term goal is to develop a theory of reasoning about representations of information systems, especially as pertaining to information-seeking [21]. The ideas presented here provide a glimpse into what such a theory might look like.

Acknowledgements

This article is a gift to my mother, who suggested the title. My view of personalization has benefited from interactions with Robert Capra, Saverio Perugini, Manuel Perez, John Riedl, Joseph Konstan, and Ed Fox.

References

1. J. Riedl. Personalization and Privacy. *IEEE Internet Computing*, Vol. 5(6):pages 29–31, Nov-Dec 2001.
2. B. Mobasher, R. Cooley, and J. Srivastava. Automatic Personalization Based on Web Usage Mining. *Communications of the ACM*, Vol. 43(8): pages 142–151, 2000.
3. M.D. Mulvenna, S.S. Anand, and A.G. Buchner. Personalization on the Net using Web Mining. *Communications of the ACM*, Vol. 43(8): pages 122–125, 2000.
4. P. Brusilovsky. Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, Vol. 11(1-2):pages 87–110, 2001.
5. M. Hearst, A. Elliott, J. English, R.R. Sinha, K. Swearingen, K.-P. Yee. Finding the Flow in Web Site Search. *Communications of the ACM*, Vol. 45(9):pages 42–49, 2002.
6. C.A. Thompson, M.H. Goker, and P. Langley. A Personalized System for Conversational Recommendations. *Journal of Artificial Intelligence Research*, Vol. 21:pages 393–428, 2004.
7. D. Riecken. Personalized Views of Personalization. *Communications of the ACM*, Vol. 43(8):pages 26–28, 2000.
8. J.F. Allen, D.K. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. Towards Conversational Human-Computer Interaction. *AI Magazine*, Vol. 22(4):pages 27–37, Winter 2001.
9. S. Perugini and N. Ramakrishnan. Personalizing Websites with Mixed-Initiative Interaction. *IEEE IT Professional*, Vol. 5(2):pages 9–15, Mar-Apr 2003.

10. N.D. Jones. An Introduction to Partial Evaluation. *ACM Computing Surveys*, Vol. 28(3):pages 480–503, September 1996.
11. M. Narayan, C. Williams, S. Perugini, and N. Ramakrishnan. Staging Transformations for Multimodal Web Interaction Management. In *Proceedings of the Thirteenth International World Wide Web Conference (WWW'2004)*, New York, NY, pages 212-223, May 2004.
12. F. van Harmelen and A. Bundy. Explanation-Based Generalisation = Partial Evaluation. *Artificial Intelligence*, Vol. 36(3):pages 401–412, 1988.
13. G. DeJong. Explanation-Based Learning. In A.B. Tucker, editor, *The Computer Science and Engineering Handbook*, pages 499–520. CRC Press, 1997.
14. N. Ramakrishnan, M.B. Rosson, and J.M. Carroll. Explaining Scenarios for Information Personalization. Technical Report cs.HC/0111007, Computing Research Repository (CoRR), http://xxx.lanl.gov/abs/cs.HC/0111007.
15. N.S. Flann and T.G. Dietterich. A Survey of Explanation-Based Methods for Inductive Learning. *Machine Learning*, Vol. 4:pages 187–266, 1989.
16. N. Ramakrishnan, R. Capra, and M.A. Pérez-Quiñones. Mixed-Initiative Interaction = Mixed Computation. In P. Thiemann, editor, *Proceedings of the ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation (PEPM)*, pages 119–130. ACM Press, January 2002.
17. C. Consel and S.-C. Khoo. Parameterized Partial Evaluation. *ACM Transactions on Programming Languages and Systems*, Vol. 15(3):pages 463–493, July 1993.
18. N.D. Jones, C.K. Gomard, and P. Sestoft. *Partial Evaluation and Automatic Program Generation.* Prentice Hall International, June 1993.
19. W. Jones, H. Bruce, and S. Dumais. Keeping Found Things Found on the Web. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, pages 119–126. ACM Press, November 2001.
20. A. Wexelblat and P. Maes. Footprints: History-Rich Tools for Information Foraging. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'99)*, pages 270–277. Pittsburgh, PA, 1999.
21. G. Marchionini. *Information Seeking in Electronic Environments.* Cambridge University Press, 1997.

Addressing Users' Privacy Concerns for Improving Personalization Quality: Towards an Integration of User Studies and Algorithm Evaluation[*]

Bettina Berendt and Maximilian Teltzrow

Institute of Information Systems, Humboldt-Universität zu Berlin
Spandauer Str. 1, D-10178 Berlin, Germany
http://www.wiwi.hu-berlin.de/{~berendt|~teltzrow}

Abstract. Numerous studies have demonstrated the effectiveness of personalization using quality criteria both from machine learning / data mining and from user studies. However, a site requires more than a high-performance personalization algorithm: it needs to convince its users to input the data needed by the algorithm. Today's Web users are becoming increasingly privacy-conscious and less willing to disclose personal data. How can the advantages of personalization (and hence, of disclosure) be communicated effectively, and how can the success of such strategies be measured in terms of improved personalization quality? In this paper, we argue for a tighter integration of the HCI and computational issues involved in these questions. We first outline the problems for personalization that arise from the combination of users' privacy concerns and sites' current policies of dealing with privacy issues. We then describe the results of an experiment that investigated the effects of changes to a site's interface on users' willingness to disclose data for personalization. This is followed by an overview of studies of the sensitivity of mining algorithms to changes in the availability of these types of data. Based on this, we outline a research agenda for future evaluation studies and user agent design.

Various personalization systems have been developed in recent years and their benefits described [26, 27]. Personalized systems require data about individuals to successfully adapt to the user. However, users are getting more and more concerned about their privacy. A meta-study of 30 surveys has shown that Internet users strongly dislike the collection and use of personal data [45]. These privacy concerns represent a major impediment for a more wide-spread use of personalization [29] and user-adaptive e-commerce [14]. Yet, current Web privacy statements are typically written in a way that seems as if site operators do not want users to read them: whereas 76% of respondents indicated that they find privacy policies very important [16], it has been found that users hardly pay any attention to them.[1]

[*] This research was supported by the Deutsche Forschungsgemeinschaft, Berlin-Brandenburg Graduate School in Distributed Information Systems (DFG grant no. GRK 316/2).

[1] For example, on the day after the company Excite@home was featured in a *60 Minutes* segment on Internet privacy, only 100 out of 20 million unique visitors accessed that company's privacy pages. Many site managers claim that fewer than 1% of all users read privacy policies [27].

This situation has left site operators and researchers wondering how to better communicate their data collection policies and the advantages arising from them. In this paper, we address this question from an evaluative and instrumental perspective. In Section 1, concerns from a selection of consumer privacy surveys are highlighted. We then outline factors influencing users' data disclosure behavior, which in return may impact the quality of personalization results. Section 2 describes the results of an experiment that suggests an influence of a site's *communication design* on users' willingness to share data. Section 3 then takes a more computational viewpoint and discusses quantitative methods of measuring the influence of data availability on personalization quality. Data availability will be operationalized in terms of *levels of identity disclosure*. The results also show that the availability of data interacts with the personalization algorithm chosen and with site characteristics.

While personalization algorithm and site characteristics are the site operator's decision parameters, the availability of data is the user's decision parameter. A more user-oriented evaluation methodology will represent a shift in emphasis and require changes in methodology. Thus, in Section 4, we conclude by outlining requirements for the design of evaluation studies and site-user interfaces. In particular, we propose that an increased level of transparency of how a user's data provision affects recommendation quality will prove beneficial for both users and sites.

This work has implications for privacy research and practice, especially for managers of personalization sites. Moreover, it highlights links between HCI and computational aspects of personalization and suggests further work in the development of privacy-preserving personalization systems.

1 Problems for Personalization That Arise from Users' Privacy Concerns and Sites' Current Policies of Dealing with Privacy Issues

Privacy concerns are a severe drawback to personalization. In this section, we describe data categories relevant for personalization and a selection of findings from consumer studies to give an insight into current user concerns.

1.1 A Categorization of Data Used for Personalization

Personalization requires two types of knowledge: *individual-user information*, i.e., knowledge about the user to whom a recommendation is to be made, and *background knowledge* about what to recommend based on the individual-user information. The first type of knowledge consists of the (potentially personal) data that the individual user discloses; the second type consists of (i) information about the product catalog and business rules (e.g.: if a user is interested in action movies, recommend Terminator to him) and (ii) patterns derived from historical data (e.g., ratings given by previous users; site navigation patterns). This distinction is reflected in Kobsa, Koenemann, and Pohl's [27] classification into user data, usage data, environment data (all concerning the individual user), and usage regularities. The P3P classification can be regarded as a further refinement of this idea. The Platform for Privacy Preferences (P3P) [12] provides Web site managers with a standardized way to disclose how their site collects, uses, and shares personal information about users. It provides several pre-defined *types of data*. It specifies a "data schema" describing sets of "data elements", which are specific items of data a service might collect online. For example, it differentiates data categories such as "physical contact information", "unique identifiers", "purchase information", "computer information", "navigation and click-stream data", or "demographic and socioeconomic data".

Since background knowledge relies on individual-user information, in the following we will concentrate on the subclasses of individual-user information. Table 1 shows them, adding the types of data disclosed when typical shopping dialogue questions are asked (used in, e.g., [42, 45]).

A closer look at user perceptions and concerns reveals that a further criterion needs to be taken into account for classifying data. Investigating the concerns of the "pragmatic majority" [2] of users who exhibit a medium degree of privacy concerns, Spiekermann, Grossklags, and Berendt [42] found that one group were particularly concerned about disclosing aspects of their identity, while others were particularly opposed to revealing a personal profile. This distinction groups data across the previous classifications, as shown in Table 1. The resulting complexity indicates that an analysis of privacy concerns and their effects on personalization should start by focusing on specific subclasses of these concerns.

Table 1. Subclasses of individual-user information

Data	Identity	Profile
1. User data [27]		
1.1. Demographic and socioeconomic data [12]		X
1.2. Physical contact data [12]	X	
1.3. Product- or usage-related preferences; ratings (e.g., [42, 45]; collaborative filtering approaches)		X
1.4. Personal preferences (e.g., [42, 45])		X
2. Usage data [27]		
2.1. Navigation and click-stream data [12]		X
2.2. Unique identifiers, e.g., cookies [12]	X	
3. Environment data [27]		
3.1. Computer information [12]	X (cf. Section 3.1)	X [42]

1.2 Privacy Concerns and Perceptions of the Privacy-Personalization Tradeoff

Using the categorization introduced in the previous section, Table 2 shows the results of a recent meta-study [45].

Table 2. Consumer privacy concerns and affected data categories

Consumer Concerns	Data categories affected	
Internet Users who are concerned about the security of personal information: 83% [15], 70% [19], 84% [18]	Demographic and socioeconomic data, physical contact data	User data
People who have refused to give (personal) information to a Web site: 82% [14]	Demographic and socioeconomic data, physical contact data	
Internet users who supplied false or fictitious information to a Web site when asked to register: 34% [14], 24% [18]	Demographic and socioeconomic data, physical contact data	
People wanting businesses to seek permission before using their personal information for marketing: 90% [40]	Demographic and socioeconomic data, physical contact data	
People who are concerned about tracking on the Internet: 60% [15], 54% [18]	Navigation and click-stream data, computer information	Usage Data
Internet users who say they set their computer to reject cookies: 25% [14], 3% [15] (31% in warning modus), 10% [18]	Unique identifiers	
Users uncomfortable with tracking across multiple Web sites: 91% [22]	Navigation and click-stream data, computer information	
Internet users who delete cookies periodically: 52% [37]	Unique identifiers	
Users uncomfortable with schemes that merged tracking of browsing habits with an individual's identity: 82% [22]	Physical contact data, Navigation and click-stream data, computer information	User data, Usage Data
People who are concerned if a business shares their information for a different than the original purpose: 91% [38], 90% [40]	All	All

Although consumers are concerned about data collection, several surveys indicate that users would provide personal information more willingly in return for personalized services:

- Users would provide, in return for personalized content, information on their name (88%), education (88%), age (86%), hobbies (83%), salary (59%), or credit card number (13%) [15].
- 27% of Internet users think tracking allows the site to provide information tailored to specific users [18].
- 73% of online users find it useful if a site remembers basic information such as name and address [37].
- People are willing to give information to receive a personalized online experience: 51% [37], 40% [40].

Further user studies have shown that consumers are in general more willing to share personal information in return for *benefits* (e.g., [21, 48]).

1.3 Variables That Influence Users' Privacy Concerns

The cited studies demonstrate that users are highly concerned about their privacy when interacting online but would disclose personal information in return for personalized content. In order to address this trade-off, one has to consider several key factors that influence consumers' willingness to share personal data, in particular: general privacy attitudes about data collection of specific data types (cf. [2, 42]), site reputation (cf. [17, 47]), types of data collected (cf. [2]), purpose of data use, recipient of data [12] as well as the design and presentation of personalization benefits and privacy policies (cf. [25, 36]).

In the following, we will consider privacy attitudes and site reputation as variables that cannot be modified by a site operator in the short-term.

The types of data necessary to personalize a Web site are to a large extent specified by the personalization algorithms; in the following, we will treat them as constant. (However, sites should and do consider findings on the relative willingness to disclose these data in their choice of algorithms.)

The purpose of data use is decisive for a user to provide personal information. Studies found that users would more willingly share personal data for Web site administration than for marketing contacts. P3P defines twelve usage purposes such as "site and system administration", "one-time tailoring", "pseudonymous analysis", "individual analysis", or "contacting visitors for marketing of services or products". For the present investigation, personalization (or "tailoring") is the primary purpose.

Users' willingness to disclose data is also influenced by the recipient of their data. For example, studies have found that users are less concerned about data sharing when they interact with a government rather than with an e-commerce site [10]. We consider the user's attitude towards the data recipient as another external factor that cannot be influenced directly in the short-term.

A factor that is certainly under the site operator's control is the design and presentation of personalization benefits and privacy policies. We will call this factor *communication design*. It is obviously central for the management of customer relations. It is also very interesting from a consumer point of view because it can not only be investigated with respect to the possibilities of more information, but also with respect to the dangers of manipulation.

1.4 Site Communication Design

The problems arising from the attempt to communicate via privacy statements have inspired a number of proposals for improvement. A non-technical approach to privacy communication has been proposed by Abrams [1]. As an alternative to lengthy and legalistic privacy statements, he suggests a layered approach which includes one short concise notice with standardized vocabulary that is easy to follow and highlights the important information, and a long, complete policy that includes the details.

Patrick and Kenny [36] are concerned with the communication of privacy choices under the European Data Protection Directive. From the principles of this Directive, the authors derived four HCI guidelines for effective privacy interface design: comprehension, consciousness, control, and consent.

Several browser-based approaches for privacy communication have been suggested. The AT&T Privacy Bird (http://www.privacybird.com/, see also [13]) allows users to specify privacy preferences, compares them with a site's P3P-encoded privacy policy and alerts them when this policy does not meet their standards.

Recent versions of common browsers (e.g., Mozilla, Internet Explorer) allow users to specify certain limited privacy preferences and to compare them with the P3P policies of visited Web sites.

However, all these approaches suffer from a number of shortcomings:

(1) They require users to make privacy decisions upfront, without regard to specific circumstances in the context of a particular site or of individual pages at a site.
(2) They do not enhance users' *understanding* of basic privacy settings. For example, most users still do not know what a cookie is and what it can do.
(3) They do not inform about the *benefits* of providing the requested data.

The third shortcoming in particular appears to be of central importance, as the overview of studies in Section 1.2 suggests. These questionnaire-based results are also supported by the results of a laboratory study of online shopping: Spiekermann et al. [42] found that although most users described themselves as highly privacy-conscious in a questionnaire, they disclosed a large amount of highly personal data without any apparent need to do so when interacting with an electronic shopping agent. A closer look at the details of communication design [5] suggests the presence of a *framing* effect: While the questionnaire focused on the loss of privacy that arises from data disclosure, the electronic shop emphasized the benefits to be had in exchange. However, this conjecture was not tested experimentally in that study. The experiment described in the next section did just this by an experimental manipulation of communication design.

2 Analyzing Users: An Experimental Evaluation of the Influence of Communication Design on Data Disclosure

Kobsa and Teltzrow [45] proposed a new user-interface design approach that explicates the privacy practices of a Web site in a contextualized manner and clearly explains users' benefits of providing personal data. To test the merits of this approach, an experiment was carried out that compared two versions of a personalized Web bookstore: one with a traditional global description and one that also provided contextualized explanations of privacy practices and personalization benefits.

Contextual explanations broke long privacy policies into smaller, more understandable pieces and referred more concretely to the current context. This allowed users to make situated decisions regarding the disclosure of their personal data, considering the explicated privacy practices and personalization benefits. The contextualized explanations were complemented by the full privacy statement kept in the background for legal reference and protection.

The effects of contextual explanation were tested in a between-subjects design: Participants in two groups (n_1=n_2=30) were asked to answer 32 questions. These items were demographic questions asking for personal data, e.g., "what is your monthly income?"; questions asking for demographic information that had a relation to books, e.g. "what are your hobbies?"; directly product-related questions, e.g. "who is your favorite author?"; usage-oriented, and purely private questions.

Results indicated that tailored in-context explanation of privacy practices and personalization benefits addressed users' privacy concerns much better than global contextless disclosures: In the contextualized-explanation condition, participants

(a) answered 8.3% more questions (gave at least one answer),
(b) gave 19.6% more answers,
(c) purchased 33% more often,
(d) stated that their data had helped the Web store to select better books (even though the recommendations were static and identical for both groups).

These differences were highly significant ($p<0.001$) in (a) and (b), significant ($p<0.035$) in (d), and they approached significance ($p<0.07$) in (c). The type of questions did not significantly influence participants' willingness to share data. In both conditions, the willingness to answer demographic questions was almost equal to the willingness to answer product-related questions.

One question concerned the acceptance of cookies. Acceptance of cookies was higher in the contextualized condition. All users in the contextualized condition accepted cookies, whereas only 80% did so in the non-contextualized condition.

In this study, users were not asked to provide ratings. However, we expect that a contextualized design might also increase the willingness to disclose information in the form of ratings.

Some limitations need to be mentioned: First, the Web site enjoyed a high reputation which led to a high level of data disclosure. This raises the possibility that Web sites with lesser reputation might experience an even stronger effect of contextualized explanation of privacy practices and personalization benefits. Second, the privacy policy of the Web site that had been mimicked was comparatively strict. Chances are that this may change if a site's privacy policy is less customer-friendly. Third, the contextual explanations given were in most cases incomplete since they needed to be short and focused on the current situation to ensure that users would read and understand them. Thus, it is advisable to include in every contextual explanation a proviso to the complete privacy statement for a full disclosure. In such a design, users might be concerned that the Web site is hiding privacy-critical parts of their disclosure in the "small print", and show less willingness to disclose personal data.

The results demonstrate that the contextualized communication of privacy practices and personalization benefits has a significant positive effect on users' data sharing behavior, and on their perception of the Web site's privacy practices as well as the perceived benefit resulting from data disclosure. The additional finding that this form of explanation also leads to more purchases approached significance. It is therefore advisable to define a corresponding interface design pattern that Web retailers can employ to improve the communication of their privacy policies, in order to increase data disclosure which is a prerequisite for successful personalization.

3 Analyzing Algorithms: Experimental Evaluations of the Influence of Data Availability on Personalization Quality

The foregoing discussion has rested on a simple assumption: "The more data, the better the quality of recommendation". The experiment described in the previous section has shown that an adequate communication design can result in "more data."

However, two questions arise: (i) what do “more” and “better” actually mean in this context, and (ii) do more data always lead to better recommendations?

To answer these questions, we first need to ask what factors are likely to influence quality. Besides the amount and kind of available data, one can expect algorithmic choices (for data preprocessing and mining) to play an important role. As the following discussion will show, part of this influence is indirect, via the quality of the preprocessed data. We expect site design to play (i) an indirect role via its influence on individual-user information and (ii) an indirect role via its influence on data quality. Figure 1 summarizes the framework that will be the subject of this section.

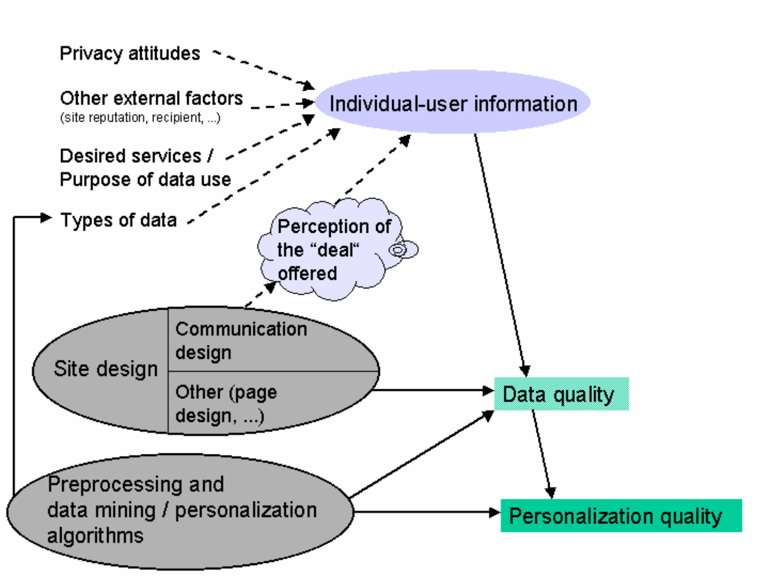

Privacy attitudes
Other external factors
(site reputation, recipient, ...)
Desired services / Purpose of data use
Types of data
Individual-user information
Perception of the “deal“ offered
Site design
Communication design
Other (page design, ...)
Data quality
Preprocessing and data mining / personalization algorithms
Personalization quality

Fig. 1. Objects of evaluation: User and site operator decision parameters (open and bounded ellipses) that influence personalization quality. Influence tested by analyzing users (dashed lines) or algorithms (solid lines).

We presume that for a user, it is not important whether, but by how much, the addition of a particular piece of data increases recommendation quality. Disclosure of private data can be seen as a “deal”: By “paying” with the disclosure of certain data, the user aims at “buying” a certain increase in recommendation quality. This increase can be relative to a zero baseline (in which the site collects no data from the user and recommends the same to every user), or relative to a baseline of recommendations given when another, smaller set of data is transferred. Regardless of what the baseline is, the findings we described in Section 1.2 suggest that the user will want to know how much he gets in return for how much additional disclosure.

Unfortunately, current personalization systems function on an all-or-nothing, or at most discretized, basis. The user can only choose whether to pay nothing (e.g., accept no cookies from the site) or all (accept all cookies), in return for an unspecified increase in recommendation quality. P3P and its variants (such as the contextualized add-on described in Section 2) are limited to a qualitative specification of the exchange relation: In return for data of type *X*, services of type *Y* can be offered. Service types are rather coarsely described in the base P3P schema, see Section 1.3 and the P3P specification [12].[2] Instead, the user should be given the option to accept or reject a deal that can be described as "In return for an amount *a* of data of type *X*, services of type *Y* can be offered *b%* better" (with a well-defined and easy-to-understand description of how quality is measured; a percentage value is one convenient example).

In the following, we will survey a number of studies that present methods allowing us to describe relevant combinations of *X*, *a*, *Y*, and *b*. We concentrate on the disclosure of (or attempt to hide) a persistent Internet-based identity (*X*). Findings from questionnaire studies (see Table 2) as well as – to a certain extent – the experimental findings described in Section 2 indicate that identity disclosure is one of today's Internet users' key concerns. We propose a categorization of levels of identity disclosure that are currently available to Web users (levels of *a*), and methods for defining and measuring the resulting recommendation quality (*Y* and *b*).[3]

3.1 Quantifying Level of Identity Disclosure and Recommendation Quality

We will distinguish five levels, or values *a*, of the variable "identity" (*X*). The criterion for classification will be the scope of the identifiers that a person assumes during their behaviour on the Web. The inspection of a typical Web server log reveals that today, a "normal" level of disclosure involves the transfer of IP address and user agent.[4] We call this the *low* level. A session cookie, or session ID, leads to an *elevated* level. Many sites attempt to increase the level to *high* by asking users to accept persistent cookies that are restricted to the respective site. Increasingly, identifiers that

[2] P3P allows refinements of this description, but this can only be satisfactory if sites *and* user agents agree on common sets and meanings of the extensions – a difficult process.

[3] Identity disclosure was chosen over profile disclosure because the operationalization of the required trade-off (*a* of *X* vs. *b* of *Y*) appears to be much clearer. Available findings on the effects of profile disclosure (e.g., the number of ratings given in collaborative-filtering systems) make it difficult to assign an expected personalization benefit to a user action. Also, while studies have shown that more users providing ratings and thus more background knowledge have a positive effect on recommendation quality [3], studies that investigated the effect of a differential availability of ratings from one individual user on recommendation quality had far less clear-cut results – for a number of algorithms and data sets, more ratings led to a *decrease* in personalization quality [8, 9, 23, 41]. These results require further investigation, in particular a differentiation by the type of information provided (or withheld). In addition, the concentration of the literature on sparsity as a given, and the application of methods for dimensionality reduction as the solution (cf. [4]), lets us suspect that any quantitative impact of more profile revelation induced by an adequate communication design may pale in comparison to the overall sparsity of data.

[4] In addition to these *environment data*, *usage data* like access time and referring pages (and of course the requested pages) are transferred. Session reconstruction methods generally distinguish two components: user / pseudo-user identification, and session boundary detection; for the identification of (pseudo) users, only IP and agent are used. We therefore focus on these data for describing the level of identity disclosure.

extend beyond the limits of a single site are discussed. We call this the *maximum* level. User registration may tie these identifiers to real-world identities, but it may also tie them to pseudonyms which are effectively just names for a cookie identifier. We therefore distinguish between with/without user registration only by adding an index to the level (e.g., $high_2$ / $high_1$). At the other end of the spectrum, the client can configure its software to avoid the transfer of user agent information, and use an anonymizing mix to hide its IP address from the Web server – a *minimum* level.[5]

We will describe findings from a number of studies that have investigated different operationalizations of user-provided data (the a of identity disclosure X) as well as of the other relevant factors shown in Fig. 1: algorithms and site characteristics. The different algorithms used in these studies represent examples of common personalization algorithm families and their associated quality measures. Their values b are interpreted as measures of personalization quality Y.

The current state of this field of research is too premature to draw generally applicable, quantitative conclusions concerning the returns on supplying certain data. We will therefore focus on the described studies' methodologies rather than on their detailed numerical results.

3.2 Level of Identity Disclosure: From *Low* to *High*

In this section, we describe a series of evaluation studies that combined investigations of the effect of changes in (i) the level of identity disclosure, (ii) data preparation algorithms, and (iii) site characteristics [6, 7, 43]. In terms of the framework shown in Fig. 1, these studies investigated the effects of

- individual-user information (*low*: IP + user agent vs. $high_1$: IP + user agent + persistent cookies)
- algorithms (sessionization heuristic: *maximum session duration* vs. *maximum timeout between two consecutive requests* vs. *all requests are issued by clicking a hyperlink on a page in the current session*)
- site design (*frames* vs. *no frames*)

The primary aim of the studies was to investigate the consequences of the need to use heuristics in data preparation for Web usage mining. In particular, most Web usage mining analyses regard *sessions* (also known as visits) as their basic unit of analysis. A session is the sequence of activities of a user from the moment he enters the site to the moment he leaves it. However, most Web server log formats contain neither information on who made a request nor on whether this was the first, last, or an in-between page viewed. Instead, Web server logs only record each request's originating IP address, the time, the requested page and its referring page (if it exists), and information on the user agent.

[5] Note that this is a very coarse description of identity disclosure levels; in general, many more combinations of user and usage data (and the usage regularities derived from them) and environment data are conceivable. For example, we may wish to further differentiate between a pseudonymous cookie that extends over several sites but records only navigation information, an Internet-wide pseudonym with navigational and demographic information attached to it, and an identity that is associated with a person's real name (cf. [27]).

In sessionization [11], this information is used to partition the log into (reconstructed) sessions. Common heuristics for reconstruction are based on assumptions about users' navigational behaviour. However, while these assumptions are based on empirical studies, they are not always true, and thus the session reconstruction heuristics based on them introduce a certain amount of error into the data that are then used for mining (and, in personalization, for deriving recommendations).

This problem is well-known, and a common response throughout the last years has been the call for employing (persistent) cookies or application-server logging to identify visitors in a reliable way (e.g., [28]). However, as argued above, the *high* level of identity disclosure that is enforced by cookies has come under increasing scrutiny. This implies that sites may record a sizeable amount of data without cookies (from users who rejected them), and that it may be advisable for them to weigh the "image costs" of encouraging or even requiring the acceptance of cookies against the "data-accuracy benefits" expected from the use of cookies.

In [6, 11] a formal framework for measuring the quality of session reconstruction was proposed, with different measures of recall, precision, and overlap that reflect the needs of different data mining methods and questions. Using a dataset that contained full information on both users and session boundaries, the error incurred by applying the different heuristics was computed. For each heuristic, the different levels of identity disclosure and site design were compared. Measures were proposed for determining the effects of this error on subsequent data mining analyses, including personalization based on the clustering of sessions. Results indicate that the simple temporal heuristics perform very well in reconstructing sessions, and that the increase in reconstruction quality provided by the use of cookies is smaller than expected.

As an illustration, Fig. 2 shows two recall measures (how many real sessions were identically reconstructed, how many were reconstructed with the correct entry page) and a similarity measure (which averages over the maximal degree of overlap between a real and a reconstructed session). The figure shows that on average, each real session had a reconstructed counterpart which overlapped it by nearly 70% – even at a low level of identity disclosure. Depending on the choice of reconstruction heuristic, the introduction of cookies raised the quality measures by between 12 and 18 percentage points. How high the increase really is for a given site (i.e., whether the benefit is 12-18%, or lower) will, among other factors, depend on the proportion of users who reject or periodically delete cookies. These proportions are reportedly high (see Table 2 and [34]). Whether the eventual increase is substantially significant for the site (i.e., whether it yields a net benefit) will depend on the relation between this increase on the one hand and image costs or further costs on the other.

With regard to privacy issues, these results indicate that users' wishes to reject cookies (to remain at a *low* level of *identity disclosure*) can be accommodated at comparatively little cost to data quality.

How does this increase in data quality translate into an increase in recommendation quality? In [6], PACT was used as personalization algorithm (Profile Aggregation based on Clustering Transactions [31]). In PACT, k-means is performed on a representation of each session as a vector of weighted pageviews. Cluster centroids are thresholded by a minimum frequency of visiting a page in the cluster, and the resulting vector is treated as a user profile. Predictive power was measured by WAVP (weighted average visit percentage). This evaluates each profile individually in terms of the degree to which this user model is representative of actual users. WAVP is computed by calculating the average similarity of a profile to each session that has a non-empty intersection with it (their scalar product), averaging over sessions, and normalizing by the sum of weights in the profile [31].

Recommendations based on profiles built from reconstructed sessions were evaluated as follows: Given that a user can be assigned to a particular heuristic profile, the pages in that profile are assumed to describe that user's interests – i.e., predict what other pages he will visit. To what extent do they really describe his interests, i.e. how much of that profile can also be found in his real session? If all profile weights are equal, this translates into the average percentage of pages in the profile that are visited in a session. When weights are different, pageviews are considered as more or less important depending on their frequency in the sessions that created the profile. Therefore, the presence of an important pageview in the intersection counts more than the presence of an unimportant pageview.

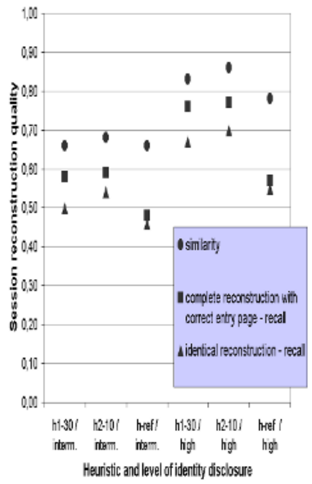


Fig. 2. The effect of the level of identity disclosure on session reconstruction quality. Heuristics: h1-30: total session duration is at most 30 minutes; h2-10: 10 minutes inactivity signal the end of a session; h-ref: a referrer that is not part of the current session signals the start of a new session.

The constructed sessions were clustered, using PACT, to obtain heuristic profiles, and WAVP values of applying these profiles to the original set of real sessions were computed. For each heuristic, the 15 profiles that were best in terms of WAVP were investigated. The value of k that gave rise to the best WAVP values was identified. The analysis showed that data quality does not translate directly into recommendation quality – the ranking of the different session reconstruction heuristics changed in relation to Fig. 2, with h1 and h-ref better than h2. All heuristics, when averaged over the best profiles, produced a drop of about 15-20 percentage points in WAVP relative to the best possible value. This corresponds to a drop in the average percentage of pages in the profile (i.e., in the recommendation set) that users in these sessions will "really" be interested in. However, since WAVP weights pages in the intersection of session and profile according to their importance, the absolute values cannot be translated directly into intersection percentages.[6]

It is likely that the quantitative values of session reconstruction and personalization quality will vary across sites, for example depending on the proportion of site users who share AOL or similar proxies. Thus, the target of this research is to provide a globally applicable framework for quality estimation that an individual site can then employ locally to provide its users with estimates of how much more quality they can expect in return for (e.g.) accepting persistent cookies.

3.3 Level of Identity Disclosure: From *High* to *Maximal*

In this section, we describe a series of evaluation studies that combined investigations of the effect of changes in (i) the level of identity disclosure, (ii) a further aspect of data preprocessing: the data selected, from a given session, to be used for further processing ("data selection"), and (iii) the algorithm operating on the preprocessed data [35], [49]. In terms of the framework shown in Fig. 1, these studies investigated the effects of

- individual-user information (*$high_2$*: IP + user agent + persistent cookies + information package vs. *maximal*: IP + user agent + across-site identity + information package)
- preprocessing algorithms (data selected from a session: *clipping* vs. *sliding window* vs. *full session*)
- personalization algorithms (prediction by *linear regression* vs. *logit model* vs. *classification trees* vs. *neural networks*).

Padmanabhan, Zheng, and Kimbrough [35] compared the effects of the availability of data at a single site to data at a large number of sites (the latter can be regarded as a rough approximation of a Web-wide identity). They framed their analysis in terms of the prediction of whether a user will make a booking or not. This target variable was motivated by two studies by Moe and Fader [32, 33] who showed that a consumer's history and purchasing threshold are highly predictive of purchasing propensity in a given session, and that consumers' searching behavior evolves with accumulated experience. The analysis could easily be transformed to the prediction of other personalization-related events.

In addition, the predicting variables contained an information package consisting of demographic data and information concerning the category of the site. These

[6] In these experiments, session IDs were used to operationalize the "real sessions". I.e., it was assumed that an *elevated* level of identity disclosure leads to a (near-)perfect representation of actual sessions. Viewed like this, the baseline were recommendations possible at an *elevated* level of identity disclosure, and these were *better* than those possible at a *high* level (persistent cookie + sessionization heuristic)! The main reason is that PACT/WAVP focused on recommendations based on the contents of the current session. This shows that depending on the algorithm, a setting that appears to deliver "more" data when viewed out of context (see Section 3.1) may lead to "worse" recommendations.

variables were constant across the two conditions compared. Assuming that there are no significant interaction effects, we therefore disregarded them.

Two kinds of tasks were investigated: (1) at any given point within a user's session, predict whether the user will make a booking in the remainder of the current session, and (2) after any given session, predict whether a user at a site will make a purchase in any future session. Only one kind of information selection method (probabilistic clipping) was used. Four classification algorithms were evaluated w.r.t. *prediction accuracy* and *lift curves*. Results showed that (i) session-level data were all but ineffective in prediction task (1), (ii) user-level data were considerably better in prediction task (1), (iii) both types of data generated good predictions in task (2). In task (2), user-level data gave only slightly better results than session-level data, but the choice of classification algorithm had a noticeable effect (see Fig. 3).

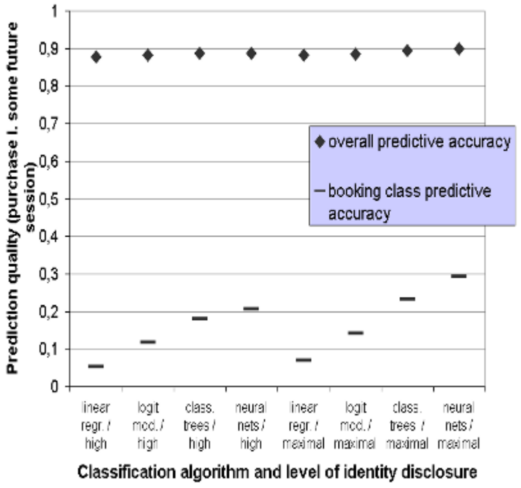


Fig. 3. The effect of the level of identity disclosure on prediction quality. Data from Padmanabhan et al. [35]; averaged over runs. The accuracy for the class of sessions that resulted in a booking is shown separately from the overall accuracy (booking and non-booking sessions); 12% of the overall points were booking records.

Padmanabhan et al. [35] employed a comparatively unusual data selection method – probabilistically sampled fragments of different lengths from whole sessions, used to predict whether there would be a booking in the remainder of that session. In their subsequent work [49], the authors investigated the effect of data selection. They compared the probabilistic clipping method used in the earlier study with the more common sliding window method – using statistics based on the previous five clicks to predict whether there would be a booking in the remainder of the session. They compared these two data selection methods with the use of the full session information. For each method, they computed lift curves using the four classification algorithms from the first study. The level of identity disclosure was fixed as *high*.

Results show that full session information performs best, followed by the sliding window method. (The full session information method obviously needs further processing to yield online recommendations when only a part of the session is available – a method that predicts according to the maximal degree of overlap with profiles previously computed offline would be conceivable.) Probabilistic clipping only yielded results close to chance. We interpret the clear pattern of results (the lift curves lie everywhere above one another in the same order for every classification algorithm) as suggesting that with a sliding window method, the same result pattern would be obtained in a comparison of levels of identity disclosure as shown in Fig. 3. This needs to be tested, and the quantitative relations remain to be established.

Regarding the prediction of a purchase as a proxy for the prediction of some other personalization-relevant event, we can interpret these results as indicating that if a site is interested in predicting behaviour over a longer term (this could be the case in relations with repeat users), the site will be able to deliver the same personalization quality regardless of whether it asks its user for a high or a maximal level of identity disclosure – i.e., there is little or no gain from more data beyond a certain level. However, if a site is interested in predicting short-term behaviour (one-time users), going from a high level of identity disclosure (e.g., a site-related cookie) to a maximal level (e.g., a pseudonym within a network of sites) can increase personalization quality. As discussed at the end of Section 3.3, it is important to note that this presents a methodology rather than a set of quantitative results that can be generalized to all sites in a straightforward manner.

3.4 Reconstructing Data Revisited: From *Minimal* to *Low* (or Higher)

The minimum level of identity disclosure could be used to identify a lower baseline of recommendation quality as follows. The use of an anonymizing mix (e.g., JAP, see http://anon.inf.tu-dresden.de) implies that the requests a site receives from one real session (i) will not disclose the real IP address of the user and, in addition, (ii) may each appear to originate from a different client. In addition, it is likely that privacy-conscious mix users will (iii) configure their browser software so as to avoid the transmission of referrer and user agent information. This is intended to make the reconstruction of the user's session impossible; so at most "non-personalized" recommendations like the most popular item bought can be given and, at the same time, serve as a baseline.[7]

4 Conclusions and Future Work

The comparison between the studies described in the previous section shows that although they dealt with related questions, they operationalized variables in very

[7] Another question is whether it is possible, for the privacy-conscious user, to guarantee this minimal level. There are various attacks against e-mail or Web anonymizers, revealing probability values of who was connected to whom (for an overview, see [30]). However, the security of anonymizers can be considered sufficient for most realistic attackers, In particular, if the attackers are corporations gathering customer data, the barrier will be more than high enough.

different ways – a comprehensive framework is still lacking. Nonetheless, the discussion has shown that user- and privacy-oriented measures of influences on user data provision and their effects can be integrated with common evaluation methods for algorithms. Future work should design studies that use consistent operationalizations and simultaneously vary all three relevant types of factors: the data supplied, but also the algorithms used and the characteristics of the site. The site owner will be interested in all of them, and all are important because she needs to see changes that are possible by parameters under her control.

To the user, sites should offer detailed information concerning the effects of changing the parameters under *the user's* control, i.e. the expected returns on the provision of personal data. First, as we have shown in Section 2, users welcome this kind of information, and it affects their disclosure behaviour. Second, as we have argued in the introduction of Section 3, it is desirable to describe the relation between disclosure and expected returns in more detail than is done today. Third, as we have shown in Sections 3.2 and 3.3, it is possible to measure the increase in personalization quality that arises when certain data are supplied. These measures generally only cover one aspect of personalization quality (e.g., accuracy relative to a given test set), and they must be seen as estimates (in the statistical sense) that involve certain error margins. It is therefore an interesting challenge for user-interface design to find ways of effectively communicating these non-trivial relations. The reward of these combined efforts of data mining and HCI experts will be better-informed and more satisfied users and site owners. Our vision is that integrating a detailed description of what impacts data quality and thus personalization quality into a site's communication design will constitute a feedback loop, clarifying the "perception of the deal offered" in Fig. 1 and contributing to the willingness to disclose individual-user information.

Many open questions remain. The first is the adequacy of evaluating personalization quality based on historical data, as done in the studies mentioned in Sections 3.2 and 3.3. As Baldi, Frasconi, and Smyth [4], p. 214, point out, "a system that recommends item *X* for individual *a* will be rewarded if in the test data individual *a* did indeed purchase that item. However, [it is] entirely possible that the individual would have purchased the item anyway (whether recommended or not), or indeed the perverse situation might arise where the recommendation might actively discourage individual *a* from purchasing a product that they would have otherwise purchased[.]" Experiments that assess users' perceptions of the utility of a given recommendation at a given time are proposed as a better yardstick of personalization quality.

However, we expect that such experiments are likely to remain a complement rather than a substitute of historical-data testing. Reasons include the cost and time needed to carry out user tests versus the nearly unbounded possibilities of varying and comparing conditions in algorithmic experiments. Also, it should be kept in mind that some of the findings discussed in the present paper (viz, the influences on data quality) are independent of user satisfaction.

Regardless of whether a measure of the degree of usefulness of personalization is derived from tests on historical data or reactions of users, an important HCI question is to find out what types of information on recommendation quality users find understandable and helpful – in terms of Section 3.1: What value ranges of *b* are meaningful to a user? A first question concerns the adequacy of numerical quality measures on a scale from 0 to 1: this may be too detailed. Also, the meaning of this number may need to be made clearer – one solution would be a re-representation, or at least require a re-representation in terms of how many recommendations out of 1000 (etc.) were considered useful (cf. [20]).

In addition to this question of the presentation of a measure's values, the question is whether the measure itself is adequate – in terms of Section 3.1: What *Y* are meaningful to a user? In the machine learning and personalization communities, simple one-dimensional measures such as accuracy have come under increasing scrutiny, and the use of ROC curves and other more complex measurement methods is advocated (see [39] and [24]).[8] The WAVP measure for user profiles is an example of a complex measure specifically tailored for personalization. However, its very complexity constitutes a challenge for interpretation: It combines two elements into one value, the percentage of recommendations that are correct predictions of user interest, and the importance (weight) of items in a profile. To our knowledge, the understandability and perceived usefulness of this kind of measure have not been tested empirically. Herlocker, Konstan, Terveen, and Riedl [24] have given a comprehensive overview of measures used for evaluating collaborative filtering recommender systems, and discussed their appropriateness for different settings. They have also systematically investigated the influence of algorithmic parameters on these measures. This type of study would also be very useful for evaluating the effects of different levels of identity disclosure or other privacy-related data disclosure.

Empirical research is also needed to determine how to best describe the options for disclosing personal information – in terms of Section 3.1: What combinations of *X* and *a* are meaningful to a user? Do they need to be limited to a simple choice like "anonymous vs. pseudonymous vs. identified"? Should it be the more fine-grained description afforded by P3P's data elements with a=100% in each case, or can it go beyond this? Which context factors determine the adequacy of particular *X* and *a*?

A last improvement would be to make user agents, and data analysis programs, responsive to unintentional consequences of data collection: the inference or triangulation problem [44]. This arises when data that have not been disclosed, and were not meant to be disclosed, can be inferred from other data. In [46], we have proposed a Web service architecture for business indicator computation that identifies possible triangulations that would violate a company's P3P policy, and blocks the computation of indicators using these unintentionally available data.

In future work, user agents should be extended in similar ways to alert users to the implications of supplying certain data. In addition, the determination of which inferences are possible should become automated and implemented in a distributed fashion throughout the Web, such that business-serving frameworks as well as user agents can automatically and instantly update their knowledge of threats to privacy arising from triangulation.

[8] Note that Padmanabhan et al.'s [35] results do satisfy a criterion that Provost et al. [39] have emphasized as essential for the use of the summarization of quality in terms of an accuracy value: The lift curve for one data constellation was dominant (= had higher values throughout) over the lift curve of the other data constellation.

References

1. Abrams, M.: Making Notices Work for Real People. In: 25th International Conference of Data Protection & Privacy Commissioners, Sydney, Australia, (2003)
2. Ackerman, M.S., Cranor, L., Reagle, J.: Privacy in E-Commerce: Examining User Scenarios and Privacy Preferences. In: Proceedings of the 1st ACM E-Commerce, Denver, Co, (1999) (1-8)
3. Aggarwal, C.C., Wolf, J.L., Wu, K.-L., Yu, P.S.H.: Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering. In: Proceedings of the Fifth ACM SIGKDD International Conference on KnowledgeDiscovery and Data Mining, San Diego, CA, USA, (1999) (201-212)
4. Baldi, P., Frasconi, P., Smyth, P.: Modeling the Internet and the Web. Probabilistic Methods and Algorithms. John Wiley & Sons, Chichester, UK (2003)
5. Berendt, B., Günther, O., Spiekermann, S.: Privacy in E-Commerce: Stated Preferences Vs. Actual Behavior. Communication of the ACM, Vol. forthcoming (2004)
6. Berendt, B., Mobasher, B., Nakagawa, M., Spiliopoulou, M.: The Impact of Site Structure and User Environment on Session Reconstruction in Web Usage Analysis. In: WEBKDD 2002 - Mining Web Data for Discovering Usage Patterns and Profiles, LNAI 2703, Berlin: Springer, (2003) (115-129)
7. Berendt, B., Mobasher, B., Spiliopoulou, M., Wiltshire, J.: Measuring the Accuracy of Sessionizers for Web Usage Analysis. In: Proceedings of the Workshop on Web Mining at SIAM Data Mining Conference 2001, Chicago, IL, (2001) (7-14)
8. Breese, J., Heckerman, D., Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In: Proc. of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), Morgan Kaufmann, San Francisco, (1998) (43-52)
9. Cheung, K.-W., Tian, L.F.: Learning User Similarity and Rating Style for Collaborative Recommendation. In: Advances in Information Retrieval, 25th European Conference on IR Research, Pisa, Italy, (2003) (395-410)
10. Consumer WebWatch: A Matter of Trust: What Users Want from Web Sites. conducted by Princeton Survey Research Associates, (2002) http://www.consumerwebwatch.org/news/1_TOC.htm
11. Cooley, R., Mobasher, B., Srivastava, J.: Data Preparation for Mining World Wide Web Browsing Patterns. Journal of Knowledge and Information Systems,Vol. 1(1) (1999) (5-32)
12. Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M., Reagle, J.: The Platform for Privacy Preferences 1.0 (P3P1.0) Specification. W3C Recommendation 16 April 2002. (2002)
13. Cranor, L.F., Arjula, M., Guduru, P.: Use of a P3P User Agent by Early Adopters. In: ACM Workshop on Privacy in the Electronic Society, Washington, DC, USA, (2002) (1-10)
14. Culnan, M.J., Milne, G.R.: The Culnan-Milne Survey on Consumers & Online Privacy Notices: Summary of Responses. In: Interagency Public Workshop: Get Noticed: Effective Financial Privacy Notices, Washington, D.C., (2001)
15. CyberDialogue: UCO Software to Address Retailers' $6.2 Billion Privacy Problem. (October 2001) http://www.cyberdialogue.com/news/releases/2001/11-07-uco-retail.pdf
16. Department for Trade and Industry: Informing Consumers About E-Commerce. Conducted by MORI, London: DTI, London, (2001) http://www.mori.com/polls/2001/pdf/dti-e-commerce.pdf

17. Earp, J.B., Baumer, D.C.: Innovative Web Use to Learn About Consumer Behavior and Online Privacy. Communications of the ACM, Vol. 46(4) (2003) (81-83)
18. Fox, S., Rainie, L., Horrigan, J., Lenhart, A., Spooner, T., Carter, C.: Trust and Privacy Online: Why Americans Want to Rewrite the Rules. The Pew Internet & American Life Project, Washington, DC, (2000) http://www.pewinternet.org/pdfs/PIP_Trust_Privacy_Report.pdf
19. GartnerG2: Privacy and Security: The Hidden Growth Strategy. Press Release, (August 2001) http://www4.gartner.com/5_about/press_releases/2001/pr20010807d.html
20. Gigerenzer, G., Hoffrage, U.: How to Improve Bayesian Reasoning without Instruction: Frequency Formats. Psychological Review, Vol. 102 (1995) (684-704)
21. Hann, I.-H., Hui, K.-L., Lee, T.S., Png, I.P.L.: Online Information Privacy: Measuring the Cost-Benefit Trade-Off. In: Twenty-Third International Conference on Information Systems, Barcelona, Spain, (2002)
22. Harris Interactive: A Survey of Consumer Privacy Attitudes and Behaviors. Rochester, NY, (2000) http://www.bbbonline.org/UnderstandingPrivacy/library/harrissummary.pdf
23. Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., C.M., K.: Dependency Networks for Inference, Collaborative Filtering, and Data Visualization. Journal of Machine Learning Research, Vol. 1 (2000) (49-75)
24. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems, Vol. 22(1) (2004) (5-53)
25. Hui, K.-L.: Privacy, Information Presentation, and Question Sequence. Working Paper, (2004) http://www.comp.nus.edu.sg/~lung/sequence.pdf
26. Kobsa, A., Ten Year Anniversary Issue. User Modeling and User-Adapted Interaction. Vol. 11 Kluwer Academic Publishers: Dordrecht, Netherlands (2001)
27. Kobsa, A., Koenemann, J., Pohl, W.: Personalized Hypermedia Presentation Techniques for Improving Customer Relationships. The Knowledge Engineering Review, Vol. 16(2) (2001) (111-155)
28. Kohavi, R.: Mining E-Commerce Data: The Good, the Bad, and the Ugly. In: Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, (2001) (8-13)
29. Leathern, R.: Online Privacy. Payments and Transactions, Vol. 1, Jupiter Research, New York, (2002)
30. Mathewson, N., Dingledine, R.: Practical Traffic Analysis: Extending and Resisting Statistical Disclosure. In: to appear in the Proceedings of the 2004 Workshop on Privacy Enabling Technologies, Springer Verlag, Toronto, Canada, (2004)
31. Mobasher, B., Dai, H., Luo, T., Nakagawa, M.: Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. Data Mining and Knowledge Discovery, Vol. 6 (2002)
32. Moe, W., Fader, P.: Which Visits Lead to Purchases? Dynamic Conversion Behavior at E-Commerce Sites. The Wharton School, Working Paper, (2000) http://fourps.wharton.upenn.edu/ideas/pdf/00-023.pdf
33. Moe, W., Fader, P.: Capturing Evolving Visit Behavior in Clickstream Data. The Wharton School, Working Paper, (2000) www-marketing.wharton.upenn.edu/ideas/pdf/00-003.pdf
34. Nielsen//Netratings: Cookies Messen Internetnutzung Ungenau Und Leiten Werbe-treibende Fehl. press release, (2004) http://www.nielsennetratings.com/pr/pr_040929_germany.pdf
35. Padmanabhan, B., Zheng, Z., Kimbrough, S.O.: Personalization from Incomplete Data: What You Don't Know Can Hurt. In: Proc. ACM-SIGKDD 2001 (154-163)

36. Patrick, A.S., Kenny, S.: From Privacy Legislation to Interface Design: Implementing Information Privacy in Human-Computer Interfaces. Third International Workshop, Pet 2003, Dresden, Germany, March 26-28, 2003, Revised Papers, (ed). Dingledine, R. Heidelberg, Germany: Springer Verlag, LNCS 2760, Dresden, Germany (2003)
37. Personalization Consortium: Personalization & Privacy Survey. Personalization Consortium, (2000) http://www.personalization.org/SurveyResults.pdf
38. Privacy Commissioner of New Zealand: Privacy Concerns Loom Large. Conducted by UMR, Issue No. 42, Auckland: PC of New Zealand, Auckland, New Zealand, (2001) http://www.privacy.org.nz/privword/42pr.html
39. Provost, F.J., Fawcett, T., Kohavi, R.: The Case against Accuracy Estimation for Comparing Induction Algorithms. In: Fifteenth International Conference on Machine Learning, (1998) (445-453)
40. Roy Morgan Research: Privacy and the Community. Prepared for the Office of the Federal Privacy Commissioner, Sydney, (2001) http://www.privacy.gov.au/publications/rcommunity.html
41. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce. In: ACM Conference on Electronic Commerce, San Diego, CA, (2000) (158-167)
42. Spiekermann, S., Grossklags, J., Berendt, B.: E-Privacy in 2nd Generation E-Commerce: Privacy Preferences Versus Actual Behavior. In: EC'01: Third ACM Conference on Electronic Commerce, Tampa, FL, (2001) (38-47)
43. Spiliopoulou, M., Mobasher, B., Berendt, B., Nakagawa, M.: A Framework for the Evaluation of Session Reconstruction Heuristics in Web-Usage Analysis. INFORMS Journal on Computing, Vol. 15 (2003) (171-190)
44. Sweeney, L.: Computational Disclosure Control: A Primer on Data Privacy Protection. MIT, Cambridge, http://www.swiss.ai.mit.edu/classes/6.805/articles/privacy/sweeney-thesis-draft.pdf.(2001)
45. Teltzrow, M., Kobsa, A.: Impacts of User Privacy Preferences on Personalized Systems: A Comparative Study. In: Designing Personalized User Experiences for Ecommerce, Karat, C.-M., Blom, J., Karat, J., eds. Kluwer, Dordrecht, Netherlands (2004) (315-332)
46. Teltzrow, M., Preibusch, S., Berendt, B.: SIMT - a Privacy-Preserving Web Metrics Tool. In: IEEE Conference on Electronic Commerce (CEC04), San Diego, (2004)
47. Teo, H.-H., Wan, W., Li, L.: Volunteering Personal Information on the Internet: Effects of Reputation, Privacy Initiatives, and Reward on Online Consumer Behavior. In: 37th Hawaii International Conference on System Sciences, Big Island, Hawaii, USA, (2004)
48. Westin, A.: Testimony before U.S. House of Representatives, Committee on Energy and Commerce, Subcommittee on Commerce, Trade, and Consumer Protection, Hearing on "Opinion Surveys: What Consumers Have to Say About Information Privacy". (2001) http://energycommerce.house.gov/107/Hearings/05082001hearing209/hearing.htm
49. Zheng, Z., Padmanabhan, B., Kimbrough, S.O.: On the Existence and Significance of Data Preprocessing Biases in Web-Usage Mining. INFORMS Journal on Computing, Vol. 15 (2003) (148 -170)

Case-Based Recommender Systems: A Unifying View

Fabiana Lorenzi[1,2] and Francesco Ricci[2]

[1] Universidade Luterana do Brasil,
Rua Miguel Tostes, 101, RS, Brasil
lorenzi@ulbra.tche.br
[2] eCommerce and Tourism Research Laboratory,
ITC-irst, via Solteri 38, 38100 Trento, Italy
ricci@itc.it

Abstract. This paper presents a unifying framework to model case-based reasoning recommender systems (CBR-RSs). CBR-RSs have complex architectures and specialize the CBR problem solving methodology in a number of ways. The goal of the proposed framework is to illustrate both the common features of the various CBR-RSs as well as the points were these systems take different solutions. The proposed framework was derived by the analysis of some systems and techniques comprising nine different recommendation functionalities. The ultimate goal of the this framework is to ease the evaluation and the comparison of case-based reasoning recommender systems and to provide a tool to identify open areas for further research.

1 Introduction

Recommender systems are being used in e-commerce web sites to help customers in selecting products more suitable to their needs. The growth of Internet and the business to consumer e-Commerce has brought the need for such a new technology [32]. In the past years, a number of research projects have focused on recommender systems [25, 7, 30, 8, 27]. These systems learn about user preferences over time and automatically suggest products that fit the learned user model.

Case-Based Reasoning (CBR) is one of the most successful machine learning methodologies that exploit a knowledge-rich representation of the application domain [1, 36, 2]. Basically, CBR is a problem solving methodology that addresses a new problem by first retrieving a past, already solved similar case, and then reusing that case for solving the current problem. In the most straightforward application of CBR to recommendation generation, the case base models the products to be recommended and the set of suggested/recommended products is retrieved from the case base by searching for products similar to that partially described by the user [7]. In these approaches a case and a product are essentially considered as identical objects. The problem component of the case is typically represented by a set of product features, those specified by the user, and the solution component of the case is the product itself.

In the basic usage scenario, the customer is looking for some product to purchase. He/she makes explicit some requirements about the product being searched for and the system searches the case base for products that match the user requirements. The retrieval process is driven by a similarity metric that computes the similarity of the problem description, i.e., the current user requirements and the products in the case base. A set of cases products is then retrieved from the case base and these products are recommended to the user. If the user is not satisfied with these suggestions he/she can modify the requirements in the query and a new recommendation cycle is started.

In a Case-Based Reasoning Recommender System (CBR-RS) the effectiveness of the recommendation is based on: the ability to match user preferences with product description; the tools used to explain the match and to enforce the validity of the suggestion; the range of available functionalities and the graphical interface that support the user in browsing the information content, either the cases or the products to recommend.

In this paper we propose an interpretation framework that models how CBR-RSs behave. We have derived this model by an analysis of the literature, that even if not complete, includes a good number of radically different approaches. In carrying out such an analysis we realized that the literature is fragmented and even contradictory in explaining the effective adoption of the CBR methodology to generate recommendations. The proposed framework: a) specializes the general CBR steps to the recommendation task; b) describes how a recommender system exploits the classical CBR learning loop (retrieve-reuse-revise-review-retain); and c) illustrates how different classes of CBR-RSs further specialize the general CBR model.

To validate the framework we considered several approaches recently developed either as techniques or as full recommender systems. For instance, Entree [7], is a restaurant recommender system that provides recommendations by finding restaurants in a new city similar to restaurants the user knows and likes. The Interest Confidence Values [22] is a recommendation technique where a new product (restaurant) is suggested reasoning on the explicit and implicit user's interests on previously considered products, that are stored in the case base. This approach exploits also a forgetting mechanism that allows the system to distinguish between current and old interests. In the Comparison-Based Retrieval technique [16] is used a preference-based feedback approach that transforms the user's preference into explicit query modifications. In another system, DieToRecs [28, 11], the goal is to help the user to plan a leisure trip. DieToRecs is able to personalize the current recommendation on the base of previously stored recommendation sessions (cases). In the Compromise-Driven Retrieval technique cases are retrieved and grouped according to the compromises done by the system, i.e., the user requirements which cannot be satisfied, and therefore are relaxed by the retrieval algorithm [17]. In the Order-Based Retrieval technique [4] a number of different order relationships among cases are managed to optimally sort the recommended products.

In summary, the goal of this paper is to present a good sample of CBR-RSs and explain the proposed methods using a unifying notation that will ease the reader in understanding their relationships, respective benefits and shortcomings. We aim at offering this research as a starting point for further analysis, identifying still unexplored solutions and therefore motivating new research in the field.

The paper is organized as follow. A brief overview of recommender system technologies is given in the next section. Section 3 describes the whole Case-Based Reasoning process and how it is used in recommender systems. Section 4 discusses the proposed framework. Section 5 illustrates the chosen examples of recommendation techniques and discusses how they fit in the interpretation framework. Section 6 presents a summary comparison of the techniques presented in the previous section. Finally, Section 7 summarizes the conclusions of the paper.

2 Recommender Systems

In the past years, a number of research projects have focused on recommender systems [25, 31, 32]. These systems try to learn user preferences over time and to automatically suggest products that fit the learned user model. E-commerce sites are using recommender systems to suggest products to their customers and improve the look to buy ratio.

The most popular recommendation technique is collaborative filtering that aggregates data about customer's preferences (products' ratings) to recommend new products. Amazon.com is a very popular example of an e-Commerce site that exploits a collaborative-filtering approach. In its book section for instance, the system encourages direct feedback from customers about books they already read [32]. After this, the customer may request recommendation for books that he/she might like. Another notable example is MovieLens [19], a well-known movie recommender system that bases its recommendations on collaborative filtering as well.

Content-based filtering is another recommendation technique that basically exploits the preferences (past and current) of a specific customer to build new recommendations to the customer. NewsDude [3], for instance, observes what online news stories the user has read and not read and learns to present the user with articles he/she may be interested to read. Content-based systems are usually implemented as classifier systems based on machine learning research [37].

In collaborative filtering the recommendation depends on customers' information, and a large number of previous user/system interactions are required to build reliable recommendations. In content-based systems only the data of the current user are exploited in building a recommendation. It requires a description of user interests that is either matched in the items' catalog or provided as input for the learned user model to output a recommendation. Both approaches, if not trained with lot of examples (product ratings or pattern of user preferences), deliver poor recommendations. This limitation mostly motivated a third approach, knowledge-based, that tries to better use preexisting knowledge specific of the application domain (e.g. travels vs. computers) to build a more accurate model requiring less training instances.

The knowledge-based approach is considered complementary to the other approaches [7]. In this approach, knowledge about customers and the application domain are used to reason about what products fit the customer's preferences. The most important advantage is that this approach does not depend (exclusively) on customer's rates, hence avoiding the mentioned difficulty in bootstrapping the system. Knowledge can be expressed as a detailed user model, a model of the selection process or a description of the items that will be suggested. However, the usually complex and error prone process required for extracting the required knowledge and building the needed models (knowledge representation), is seen as a limitation of this approach.

Knowledge-based recommender system can exploit similarity metrics. For example, in the e-commerce portal site recommender.com [7] the system uses knowledge about the customer (the movie's name that the user liked) to search in the database (catalog) for similar movies. The retrieved set is sorted by the similarity to the input movie and the top candidates are recommended to the user. We will further discuss this knowledge-based approach in the next Section.

3 Case-Based Reasoning

Case-based reasoning (CBR) is a problem solving methodology that tries to solve new problems by re-using specific past experiences stored in example cases [12]. A case models a past experience, storing both the problem description and the solution applied in that context. All the cases are stored in the case base. When the system is presented with a new problem to solve, it searches for the most similar case(s) in the case base and reuses an adapted version of the retrieved solution to solve the new problem.

CBR is a cyclic and integrated problem solving process (see Figure 1) that supports learning from experience [1] and has four main steps: retrieve, reuse, adaptation and retain [12]. The adaptation phase is split into two sub-steps: revise and review. In the revise step the system adapts the solution to fit the specific constraint of the new problem. Whereas in the review step the constructed solution is evaluated by applying it to the new problem, understanding where it fails and making the necessary corrections.

In a diagnosis task, for instance, the system acquires the patient symptoms (new problem) and tries to give the final diagnosis based on past patient examples (stored in the case base). Sometime the solution retrieved can be straightforwardly reused in the new problem, but in the majority of the situations the retrieved solution is not directly applicable and must be adapted to the specific requirements of the new problem. After this adaptation the system creates a new case and could retain it in the case base (learning).

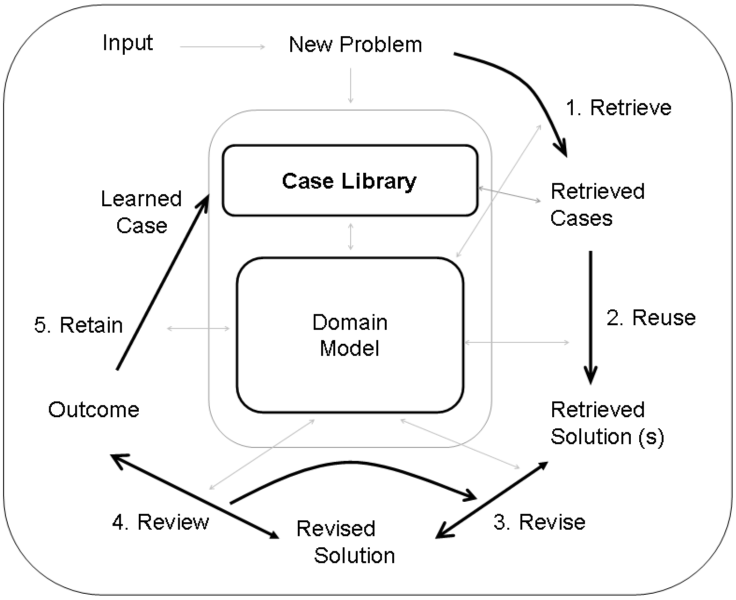


Fig. 1. Case-Based Reasoning problem solving cycle [2]

A fundamental issue in CBR is the case model. This must account for both the problem and solution components. It is necessary to decide which attributes should compose a case and what representation language is better suited to represent the particular knowledge involved in the problem solving process. Hence, the case representation task is concerned with (1) the selection of relevant attributes, (2) the definition of indexes and (3) structuring the knowledge in a specific case implementation. Indexing is related to the creation of additional data structures that can be held in the memory to speed up the search process focussing on the most relevant dimensions. The indexes identify the case attributes that should be used to measure case similarity. Moreover, indexes can speed up the retrieval process by providing fast access to those cases that must be compared with the input case problem. For instance, in a medical diagnosis system, if the system must produce an infection diagnosis then attributes such as profession, gender or age are probably less important than the attributes describing the symptoms.

4 Methodology for the CBR Recommender Systems

In this section we shall illustrate how the generic steps of the CBR problem solving cycle are specialized in a CBR Recommender System, hence providing a unifying description of various systems or techniques. Whereas in the next Section we shall illustrate some real CBR-RSs and the techniques exploited to generate the recommendations.

In the simplest recommendation process, the user is supposed to be looking for some product to purchase and therefore is asked by the system to provide some product requirements, those that he/she considers as the most important. In reply, the system initiates a search in the case base to identify products that should be recommended, i.e. those that satisfy these requirements. In this process we can identify some basic elements, such as, the input (where the user provides his/her requirements), the products retrieval (where the system searches the products according to user requirements) and the output, where some recommendation is given to the user.

As described in the previous section this flows is very similar to that of a generic CBR system. This starts with a new problem, retrieves similar cases from the case base, shows the retrieved solution to the user or adapts it to better solve the new problem and terminates the process retaining the new case.

We have therefore analyzed four CBR recommender systems and six recommendation techniques and through this analysis we created a condensed model of these recommendation approaches. This model is a general framework for accommodating the description of the specific tasks/functionalities available in the considered systems for product recommendation. Using the framework a number of approaches can be described as specific instantiation of the different steps of the CBR cycle and the evaluation and the comparison of CBR-RSs can be eased.

The CBR recommender systems that we have analyzed are:

– Entree (EN): a recommender system that exploits query tweaking to recommend restaurants to the user.
– DieToRecs (DTR): a travel recommender system that suggests both single travel services (e.g. hotel or an event) and complete travel plans comprising more that one elementary service.
– First Case (CDR): a prototype system that uses the Compromise-Driven Retrieval technique to retrieve and group cases according to the alternative compromises found by the system.
– Expertclerk (EC): a tool for developing dialogue-based recommender systems for e-commerce websites.

The recommendation techniques that we have analyzed, either included in some of the previously mentioned systems or not yet exploited in any prototype, are:

– Interest Confidence Value (ICV): a similarity-based retrieval technique that is used to predict the interest of a user in a product. This technique introduces also a mechanism to progressively forget old not useful cases.
– Single Item Recommendation (SIR): a recommendation technique introduced in the DieToRecs system (DTR) to recommend a single item (product).
– Seeking for Inspiration (SI): this technique, used in DieToRecs, updates travel plans recommendations according to explicit user feedbacks.
– Travel completion (TC): a recommendation technology introduced in DTR to recommend a complete travel a partially defined plan.

- Order-based retrieval (ODR): a retrieval technique based on the application of partial order operators to the case base.
- Comparison-based Retrieval (COB): this technique transforms user's preferences into explicit query modifications.

Figure 2 shows the framework including the classical five steps of the CBR problem solving cycle plus an additional "iterate" step. The iterate step models a peculiar feature of many RSs, i.e., to incrementally update the current set of recommendations acquiring new input from the user, usually in the form of critics or feedbacks. In each stage we list in bold face a general description of the technique or data and then the recommendation techniques or systems that exploit such general technique or data. For instance, in the "Input" box we have "product features" used by the SIR (Single Item Recommendation) technique of the DieToRecs system. All the details and acronyms mentioned in this figure will be explained in the following sections. We provide here a general description of this framework as an introduction to the description of each single technique or system.

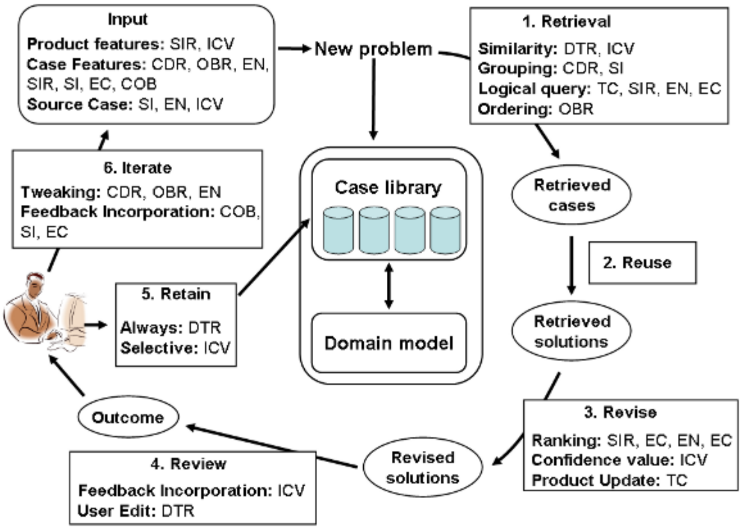


Fig. 2. CBR recommender systems framework

The first stage of many recommendation techniques is the **input** where the system interacts with the user to capture her preferences. According to [24] there are different strategies for interacting with the user. The most popular strategy is dialog-based, where the system offers guidance to the user by asking questions and presenting products alternatives, to help the user to decide. Several CBR recommender systems ask the user's requirements to have an idea of what the user is looking for. In the Compromise-Driven Retrieval [18], for instance, the user provides the (case) features of a personal computer that he/she is looking for, such as, type, price, processor or speed. Expertclerk [34] asks directly to the user to answer some questions and hence to provide case features as replies to these questions.

Usually when the user searches for a product, three situations can occur [6, 24]:

– the user knows exactly what he/she wants;
– the user has a desire but does not know the name of the product;
– the user does not know precisely what he/she is looking for.

In each of these situations, to recommend suitable alternative products the system requires some kind of knowledge. In CBR-RSs the knowledge is mainly stored in the case base and analyzing the existing CBR-RSs we noticed that the knowledge contained in a case can refer to many characteristics of the problem domain. In fact, in the systems that were considered for this study, a case stores information about: the products recommended (or to be recommended), the user to whom the recommendation was supplied, and contextual information about the recommendation session when the recommendation was provided. Actually, the systems exploits these basic ingredients to define their own specific case model as a mix of these. Hence, for instance, in one case model we may find a description of the recommended products and user who received the recommendation or the user together with his recommended products' evaluation.

To compare different CBR-RSs we propose here an artificial case model that includes case components found in these RSs. In this perspective a case base CB, can be decomposed in four sub-components:

$$CB \subseteq X \times U \times S \times E$$

where X is the product/content model, U is the user model, S is the session model, and E is the evaluation model (more details on these models are provided below). This means that a general case $c = (x, u, s, e) \in CB$ in a generic CBR-RS consists of four (optional) sub-elements x, u, s, e which are instances of the spaces X, U, S, E respectively. Each CBR-RS adopts a particular model for the spaces X, U, S, E. These spaces could be empty, vector, set of document (textual), labelled graphs, etc. Let us now describe each model separately.

– *Content model*(X): the content model describes the product recommended or to be recommended, and usually adopts a feature-based representation of the product (feature vector). In Compromise-Driven Retrieval [18], for instance, a case is modelled (only) by the content component, which is an n-dimensional vector space $X = \prod_{i=1}^{n} X_i$. Each X_i represents the set of possible values for a product attribute. For instance, when the products are computers, an attribute (symbolic) could be the computer type, or the price of the computer (numeric).
– *User model*(U): the user model usually contains personal user information, such as, name, address, age or information about the user past system usage,

such as his/her preferred products. Very few CBR-RSs have exploited this component.

- *Session model*(S): the session model is introduced to collect information about the special recommendation session (problem solving loop). In DieToRecs, for instance, a case describe a recommendation session and stores all the user queries and product selected in that session.
- *Evaluation model*(E): the evaluation model describes the outcome of the recommendation, i.e., if the suggestion was appropriate or not. This could be a user a-posteriori evaluation, or, as in [22], the outcome of an evaluation algorithm that guesses the goodness of the recommendation (exploiting the case base of previous recommendations).

Actually, in CBR-RSs, as we noticed above there is a large variability in what a case really models and therefore what components are really implemented. There are systems that use only the content model, i.e, they consider a case as a product, and other systems that focus on the perspective of cases are recommendation sessions. The example systems described in the following sections will illustrate this variability in case structure.

Going back to the problem solving cycle, let us now consider more in detail how cases are managed by different approaches. The first step of the recommendation cycle is the **retrieval** phase. This is typically the main phase and the majority of CBR recommender systems can be described as sophisticated retrieval engines. For example, in Order-Based Retrieval [4] the system uses special operators to retrieve a lattice of cases, or in the Compromise-Driven Retrieval [18] the system retrieves similar cases from the case base but also groups the cases, putting together those cases that offer the same compromise to the user and presents to the user just a representative case for each group.

After the retrieval, in the **reuse** stage the case solution is considered and the system evaluates if it can be reused in the current problem or what part of the case can be reused. In the simplest CBR-RSs, the system reuse the retrieved cases/products showing them to the user. In more advanced solutions, such as in ICV [22] or DTR [28], the retrieved cases are not recommended but used to rank candidate products identified with other approaches, for instance in DTR, with an interactive query management component.

In the next phase **revise** the reused case is adapted to better fit the new problem. The **review** phase in CBR-RSs is implemented by allowing the user to customize the retrieved set of products. For instance in DieToRecs the user can add to the current case other products either using the CBR functionality or by using other system functions (e.g. browsing the product catalogue).

The iterate step is implemented very often in conversational systems. For example, In Entree [7] the system allows the user to tweak the initial query and search for products having marginal differences with those already shown, with respect to some of the product features (e.g. cheaper products). In Comparison-based Retrieval [16] the system asks the user to provide feedback, either positive or negative, about the retrieved product and automatically updates the user query using this information.

The last step of the CBR recommendation cycle is the **retain** phase (or learning), where the new case is retained in the case base. In DieToRecs, for instance, all the user/system recommendation sessions are stored as a new cases in the case base.

The next subsections describe some representative CBR-RSs, focusing on their peculiar characteristics.

5 CBR Recommendation Techniques and Systems

5.1 Entree - EN

Entree is a restaurant recommender system that provides recommendations by finding restaurants in a new city similar to restaurants the user knows and likes or those matching some user goals (case features)[7].

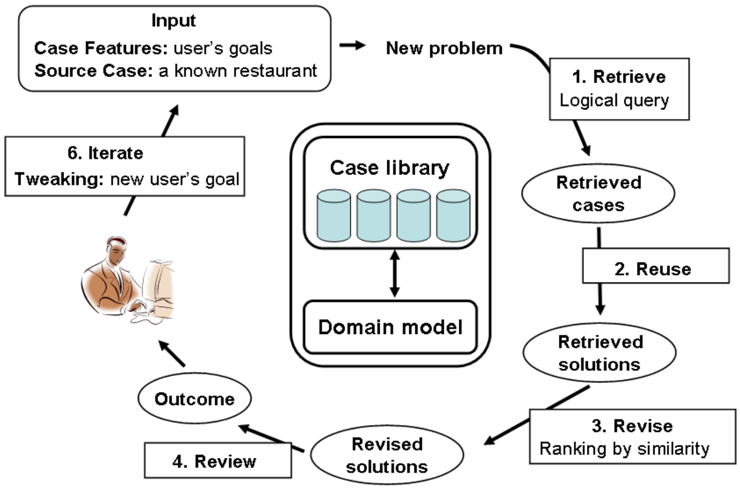


Fig. 3. Entree recommender system

The user starts the interaction with Entree, as showed in Figure 3, either by: mentioning a known restaurant in some place (source case) and asking for a similar one in a give city; or selecting a set of high-level features (case features) and searching for a restaurant that matches those features. With this input information, the system first selects from the database, which physically stores the cases, the set of all restaurants that satisfy the largest number of logical constraints generated by considering the input features type and value. The system, if necessary, implicitly relaxes the lowest important constraints until some restaurants could be retrieved.

Then Entree sorts the retrieved cases using a similarity metric. This similarity metric assumes that the user goals, corresponding to the input features (or the features of the source case), could be sorted to reflect the importance of such goals from the user point of view. Hence the global similarity metric sorts the products first with respect the most important goal and then iteratively with respect to the remaining goals (multi-level sort).

If the recommended restaurants satisfies the user then the interaction finishes. But if the user is not satisfied, because of the values of some features of the proposed restaurant, then he can criticize them. This failure situation is determined by the fact that in the similarity retrieval it is possible that the recommended restaurant does not match 100% the good example provided by the user as input. If for instance, the price is too high and the user is looking for something cheaper, then he/she can "tweak" the original request and provide a new input explicitly mentioning that the result must have a cheaper price. This starts a new recommendation cycle and the criticized features is considered the most important user goal.

In Entree the reuse step is trivially implemented, i.e. the products retrieved are passed to the revise step for ranking. The review and retain steps are not implemented.

5.2 Interest Confidence Value - ICV

Montaner et al. (in [22]) assume that the user's interest in a new product is similar to the user's interest in similar past products. This means that when a new product comes up, either selected by a user's query or by another method, then the recommender system predicts the user's interest in this product based on the interest attributes/evaluation of similar products.

A case is modelled by objective attributes describing the product (content model) and subjective attributes describing implicit or explicit interests of the user in this product (evaluation model). Formally, the case c is defined as $c \in X \times E$.

The content model (X), in this system, is represented by a vector space $\mathrm{X} = \prod_{i=1}^{n} X_i$ where, for instance, x_1 is the restaurant code (*integer*); x_2 is the restaurant name (*string*); x_3 is the restaurant address (*string*); x_4 is the cuisine type (*string*); x_5 is the approximate price (*real*); x_6 is the capacity (*integer*) and x_7 is the air-conditioning (*boolean*).

The evaluation model (E) is also an heterogeneous vector space $E = \prod_{i=1}^{m} E_i$ where some $e_i \in E_i$ describe explicit interests attributes like a general evaluation of the product provided by the user or a quality price ratio. Some other e_i describe implicit evaluation attributes like the rate of time spent by the user to read product information. There is also a special attribute, called drift attribute, that measures how recently the user expressed his interest in the product. When this drift attribute becomes very small the system tends to reduce the importance of the information contained in the case associated to that product, and eventually can discard the case.

As Figure 4 shows, the recommendation process starts with the user providing some preferences about a new restaurant he/she is looking for and with a new restaurant r (source case). The goal of the system is to evaluate if this new

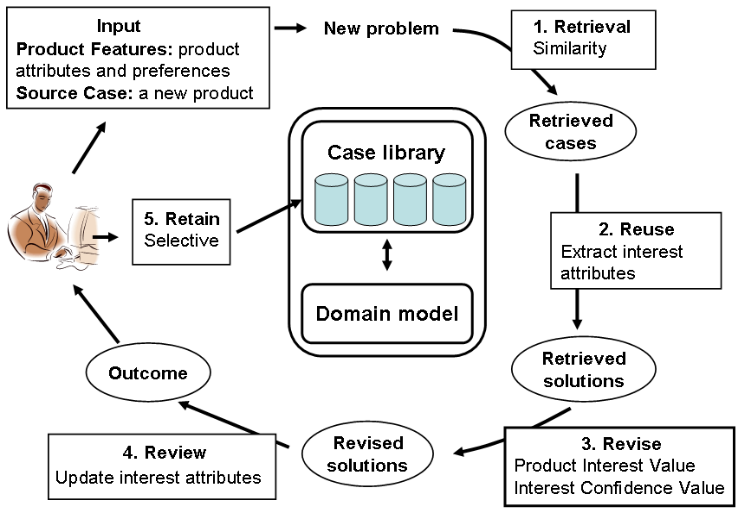


Fig. 4. The Interest Confidence Value technique

restaurant r could be interesting for the user. With these preferences and input product the system searches similar restaurants in the case base (**retrieval phase**) to find restaurants that could be used to compute the interest prediction of the new restaurant r.

In the **reuse** phase the system basically extract from the retrieved cases (c_i, $i = 1, \ldots, k$) the interest attributes, or in our terminology the evaluation model. In the **revise** phase the system assumes that the user's interest in the new restaurant r is similar to his/her interest in the retrieved restaurants, hence, for each retrieved case c_i it extracts the interest attributes (evaluation model) and compute a global interest value $V(i)$ for each retrieved case. $V(i)$ is a weighted average sum of the interest attributes multiplied by the drift attribute [22].

Then a global interest confidence value $I(r)$ for the product r is computed as a weighted average of the interest values of the retrieved cases:

$$I(r) = \frac{\sum_{i=1}^{k} V(i) Sim(r, c_i)}{\sum_{i=1}^{k} Sim(r, c_i)}$$

If the interest confidence value of the new restaurant is greater than a certain value (a confidence threshold), then the new restaurant is recommended to the user. Otherwise, the CBR cycle terminates with no recommendation and the system just provides a negative advice to the user about the queried restaurant.

The **review** phase is implemented by asking the user for the correct evaluation of the restaurant and after that a new case (the product and the evaluation) is **retained** in the case base. Also implicit evaluation indicators are retained as derived from the analysis of the user/system interaction.

We stress that in this approach the recommended product is not retrieved from the case base, as we saw before in Entree, but the retrieved cases are used to estimate the user interest in a generic new restaurant. The new restaurant could be generated in many ways, including a search in the case base.

5.3 DieToRecs - DTR

DieToRecs is a case-based travel planning recommender system, that helps the user to plan a leisure travel in a selected destination [11]. Three different recommendation techniques were implemented in DieToRecs: the single item recommendation (SIR), the travel completion (TC) and seeking for inspiration (SI).

In DieToRecs, a case represents a user interaction with the system and it is built incrementally during the recommendation session [28]. A case comprises the following main components:

– *Collaborative Features (clf)* are features that describe general user's and travel characteristics, wishes, constraints or goals (e.g. desire to relax or to practice sports). They capture preferences relevant to the user's decision-making process, which cannot be directly mapped into product attributes stored in the electronic catalog. These features are used to measure case (session) similarity. A knowledge of the domain and the decision process is essential to select the right collaborative features [26]. The collaborative features belong to the user and session models.
– *Content Queries (cnq)* are queries posed over the catalogs of products. Content queries are built by constraining (content) features that describe products listed in the catalogs. Products may belong to different types (e.g. an accommodation or an event). The content queries belong to the session model.
– *Cart* contains the set of products chosen by the user during the recommendation session represented by the case. A cart represents a meaningful (from the user's point of view) bundling of different products. For instance, a travel cart may contain some destinations, some accommodations, and some additional attractions. The cart component belongs to the content model.
– *Rate* is a collection of rates given by the user to the products contained in the cart. It represents the user evaluation of the products and therefore belongs to the evaluation model.

In the single item recommendation technique (SIR), the user interacts with the recommender system by querying recommendations about a product type (e.g., a destination). The whole process is shown in Figure 5. In SIR the systems asks the user both some general preferences (the clf) that are used to generate a case (current case or source case) and some specific product preferences that are used to query (cnq) the product catalogue. The system uses the content queries to search in the catalog for products that (logically) match these preferences and computes a result set.

The system supports an interaction flow that allows the user to eventually refine the initial content query. That is depicted in Figure 5 as a set of parallel arrows from the product features to the product catalogue. In fact, If too many products matches the input query, then a tightening function suggests to the user some additional features he may use to further constrain the search [21].

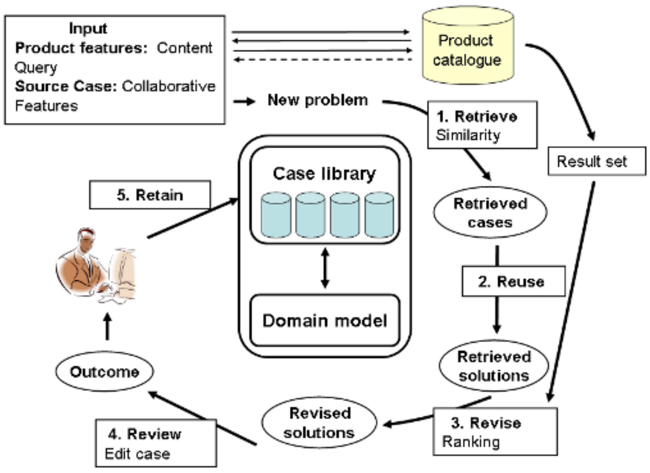


Fig. 5. DieToRecs - Single Item Recommendation technique

Conversely if no result can be found then the system explains to the user the cause of the failure, i.e., it lists those constraints that if relaxed would allow the query to return some results (relax function) [20]. When the number of items retrieved is satisfactory then the system proceeds with the **revise** phase to rank the result set.

In parallel, the collaborative features are used to retrieve the ten most similar cases, and in the reuse phase the products contained in these case are extracted. In the revise phase the results set is ranked with a double similarity process where the product contained in the result set that are more similar to products contained in similar cases obtain a higher rank [28].

Finally the user can edit the current case adding new products, using one of the available recommendation techniques (review). The case is always stored in the case base and it is updated every time the user changes some of its components (for instance adding a new travel product to the cart).

The second recommendation technique introduced in DieToRecs is called Travel Completion (TC). Here the system recommends additional travel products or services to complete the current travel plan of the user. In TC the cycle starts with the current case as source case. This is used by the system to retrieve from the case base similar cases, i.e., travel plans built by other travellers that match the collaborative features of the source case (see Figure 6). Some of the collaborative features are used to generate some logical constraints, hence the retrieval combines a similarity-based one piped after a logical filter.

Before recommending to the user the products contained in the retrieved cases (solutions), the system in the **revise** stage updates, or replace, the travel products contained in the cases exploiting up-to-date information taken from the product catalogs. This adaptation stage is constrained-based and the constraints are stored as structural properties of the travels. For instance if the destination

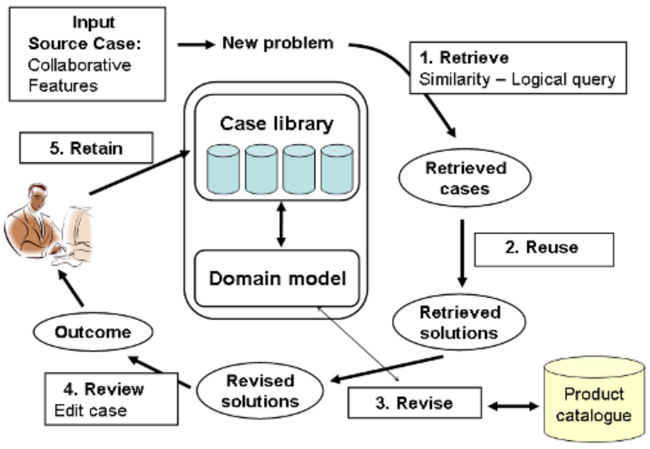


Fig. 6. DieToRecs - Travel Completion technique

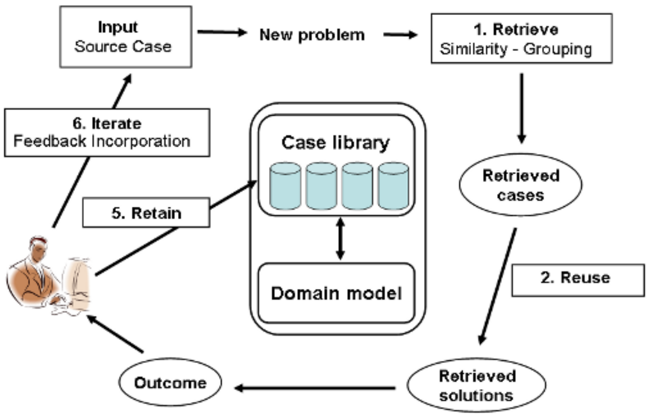


Fig. 7. DieToRecs - Seeking for Inspiration technique

is x, then the system cannot recommend an accommodation in y where the distance of x and y is larger than a given threshold.

In the **review** phase the system allows the user to reconfigure the recommended travel plan. The system allows the user to replace, add or remove items in the recommended travel plan. If the user accepts the outcome (the final version of the recommendation showed to the user), then the system **retains** this new case in the case base.

In the third recommendation technique introduced in the DieToRecs system, that is Seeking for Inspiration (SI) [29], the user is prompted with complete travel recommendations to choose. SI proceeds as a loop, which is initiated with a source case, and terminated when the user selects one of the recommended case. At each loop six cases are shown. The initial probe is randomly selected by the system if the user has not created yet any case (e.g. with any of the other recommendation techniques). These six displayed cases are computed by, first **retrieving** k most similar cases to the source case, and then selecting with a greedy algorithm the six more diverse among the k retrieved. When these six cases are shown the user can provide some feedback, by checking a "I like this" option. The system then iterates the process using the liked case as the current source case. As in the other techniques when the recommendation process terminates the newly generated case is stored in the case base.

5.4 Order-Based Retrieval - OBR

The Order-Based Retrieval (OBR) technique integrates different type of sorting criteria [4]. Using ORB the authors have developed a prototypes that helps a user to find a place to rent in London. A case is modelled using only the content model, $c = (x)$. The content model X is a vector space $X = \prod_{i=1}^{n} X_i$. Typical attributes are: the price of the apartment (a real number); the number of bedrooms (integer); the number of bathrooms (integer); the location (string); the type of the property (string) and whether it is furnished or not (boolean). The features were classified as ordered or unordered values.

The recommendation process starts when the user provides some preferences, as ideal values, or as maximum or minimum values of the searched case. In the **retrieval** phase, the system converts the user input into order relationships on the attributes. Then all these order relationships are combined in a pre-order to produce a lattice of products. The operators used in this task were defined in [5].

Figure 8 illustrate how ORB can be described in the proposed framework. In the **reuse** phase the system extracts from the lattice the maximal products. The last implemented step is the **iterate**, where the system waits for additional

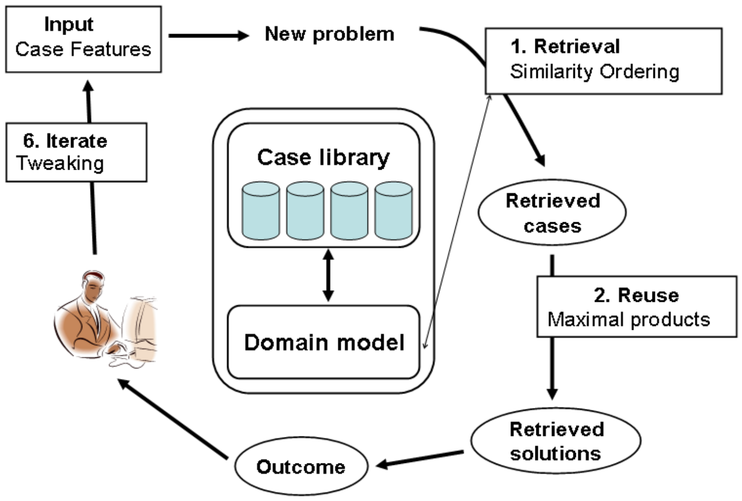


Fig. 8. The Order-Based Retrieval technique

preference constraints from the user to refine the lattice structure. The user can provide modifications to the query, and these are encoded as filters and finally converted into orders using Filter-Ordering (FO) operators. The system will recommend products that satisfy the filter but will not eliminate products that do not satisfy the filter. There are no **revise**, **review** and **retain** tasks implemented in this technique.

The advantage of OBR, compared to pure similarity-based retrieval, is that it allows the user to provide some soft constraints. Suppose the user wants to define the following query: "Rent a property in Clapham with 2 bedrooms but the rent cannot be more than 400". The system takes this upper bound condition ("not more than 400") and build an unary predicate. But instead of filtering away products it builds an ordering from the products using the Filtering-Ordering (FO) operator.

$$<_{FO(\lambda x[price(x) \leq 400])}$$

The values that satisfy the predicate are higher in the ordering than ones that do not.

5.5 First Case - CDR

First Case is a CBR-RS that uses the Compromise-Driven Retrieval (CDR) technique to recommend computers to the user. First Case models a case exploiting only the content component [18]. The content model X is a vector space $X = \prod_{i=1}^{n} X_i$ where n is the number of attributes. Typical attributes are: the computer type (string); the price (real); the manufacturer (string); the processor (string); the speed (integer); the monitor size (integer); the memory (integer) and the hard disk size (integer). In CDR, if a given case c_1 is more similar to the target query than another case c_2, and differs from the target query in a subset of the attributes in which c_2 differs from the target query, then c_1 is more acceptable than c_2.

As showed in Figure 9, the CBR recommendation cycle starts with the user providing his/her requirements in a query. The user can specify how many requirements he/she wants. For example, let's consider a query q, defined by the following conditions: Intel Pentium; speed = 900 (or higher); 17" monitor size; desktop or tower and price not higher than 1400. Let us further imagine that the hard disk and memory are not important for the user.

In the CDR **retrieval** algorithm the system sorts all the cases in the case-base according to the similarity to a given query. The combination of attributes in which the case differs from the user query is important and not just the number of attributes that differ.

For any case c and query q, the author defines the set of compromised attributes as:

$$compromises(c, q) = \{a \in A_q : \pi_a(c) \text{ fails to satisfy the user preference}\}$$

where A_q is the set of attributes constrained in the query, a is an attribute, and $\pi_a(c)$ is the value of attribute a in c.

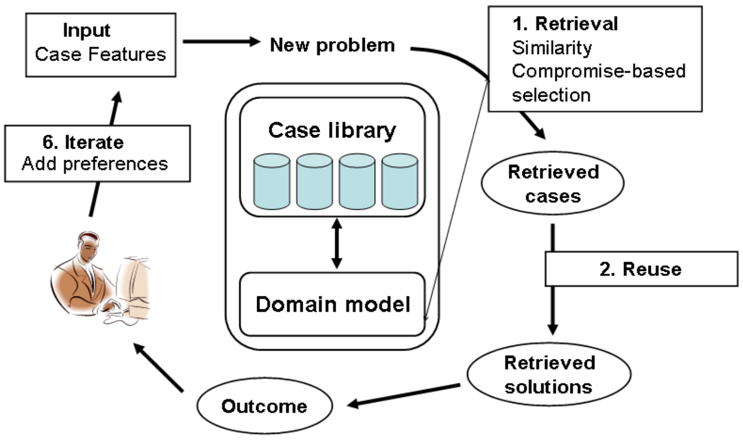


Fig. 9. First Case recommender system

In a second step the algorithm groups together the cases making the same compromise (do not match a user preferred attribute value) and builds a reference set with just one case for each compromise group. In the **reuse phase** the reference set is recommended to the user without modifications and the user has immediate access to it.

The user can also refine (**iterate**) the original query, accepting one compromise, and adding some preference on a different attribute (not that already specified). The system will further decompose the set of cases corresponding to the selected compromise. The **revise**, **review** and **retain** phases are not implemented in CDR.

In this approach similarity and compromise play complementary roles, increasing the probability that one of the retrieved cases will be acceptable to the user. CDR shows alternative compromises and groups cases according to the compromise that is done. In this way it helps the user to immediately grasp the alternatives available and therefore increase the diversity of recommendations. Other researches had proposed different ways to retrieve a diverse set of cases [35]. Mongouie et al [23] have also studied the concept of generalized cases and presented a method to build a retrieval set of cases that are enough different and are also representative for a set of similar cases.

5.6 Comparison-Based Retrieval - COB

According to McGinty and Smyth [14], a key feature that differentiates recommender systems from more conventional information retrieval systems, such as search engines, is their conversational character.

As we saw before, the majority of the recommendations techniques support a cycle that initiates with a first query/problem of the user. Then, each feedback provided by the user during the recommendation cycle is used to update the user's query. The goal is to refine the query so that user needs are better captured and a better recommendation can be produced.

Some researches have focused on different strategies for capturing the user feedback [15, 33]. We have:

– *Value-elicitation*: Where the user is asked to provide a specific value for a specific feature of the recommended product;
– *Tweaking*: Where the user is asked to provide a directional preference for a particular feature;
– *Rating-based*: Where the user is asked to rate the recommended cases according to his/her preferences;
– *Preference-based*: Where the user is asked to select one of the current recommendations, that is closest to his/her requirements.

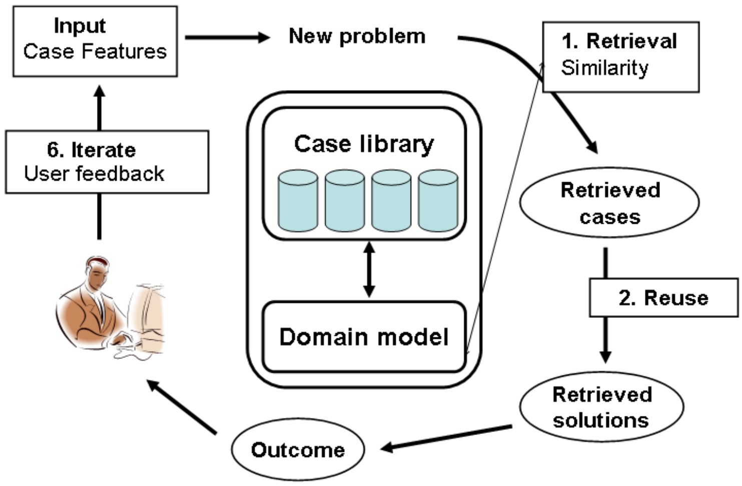


Fig. 10. Comparison-Based Retrieval

These strategies have been classified (in [33]) as: *navigation by asking*, when the system can ask to the user to specify individual feature values as search criteria or *navigation by proposing* when the system invites the user to rate recommendation as relevant or not relevant.

The comparison based retrieval technique (COB) is a navigation by proposing [34] recommender system that exploits preference feedbacks by transforming the user's preference into explicit query adaptations [15]. McGinty and Smyth have used COB to build a prototype aimed at supporting the user in selecting a computer. In this prototype the case model includes only the content model.

The recommendation process starts with the user providing his/her requirements as attribute-value pair of the preferred case. In the **retrieval** phase the system retrieves the cases with a traditional similarity-based process. The retrieved cases are shown to the user in the **review** phase.

Figure 10 shows the recommendation cycle in COB. In the **iterate** phase the user selects a preference case as feedback (positive if the user likes the case recommended or negative if he/she does not like it). This feedback are interpreted as a user evaluation of the difference between the selected product and the products not chosen. This information is used to learn from the user's feedback and update the current query. Some update strategies are used in the review phase, such as, *More-Like-This* that takes each feature of the preferred case as a new query feature, or *Partial-More-Like-This* that transfers a feature value from the preference case if none of the rejected cases have the same feature value. The process terminates when the user is presented with an acceptable item or when he/she gives up. In this approach there are no **revise**, **review** and **retain** phases.

In another paper of the same authors the COB approach is improved using the Adaptation Select technique, that adapts the way the new products are selected, making the preference-based feedback more efficient [16].

5.7 ExpertClerk - EC

Expertclerk is a general recommendation methodology aimed at implementing virtual salesclerk systems as front-end of an e-commerce website [34]. The system implements a question selection method (decision tree with information gain). Using navigation-by-asking, the system starts the recommendation session by asking the user some questions. The questions are nodes in a decision tree. A question node subdivides the set of answer nodes and each one of these represents a different answer to the question posed by the question node. The system concatenates all the answer nodes chosen by the user and then builds the SQL retrieval condition expression.

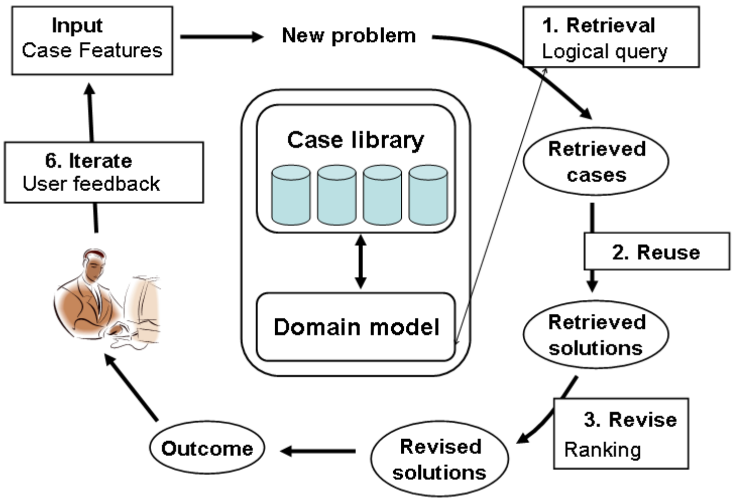


Fig. 11. ExpertClerk

In EC the user can answer the question by choosing an answer node or ignore the question (Figure 11). The system concatenates all the answer nodes chosen by the user and then constitutes the SQL retrieval condition expression. This query is applied to the case base to **retrieve** the set of cases that best match the user query.

The system ranks the products in the **revise** phase, recommends three sample products to the user, and explains their characteristics (positive and negative).

In the **review** phase, the system switches to the navigation-by-proposing conversation mode and allows the user to refine the query. After refinement, the system applies the new query to the case base and retrieves new cases (iterate). These cases are ranked and shown to the user. This cycle continues until the user finds a good product. In this approach the **retain** phases is not implemented.

6 Comparison

In this Section we present a couple of summary tables to quick compare the CBR-RSs that we have illustrated. From the analysis of these systems or techniques we have detected some common patterns. Tables 1 and 2 show a cross-dimension analysis of the systems, taking into account the features described in the proposed CBR framework .

In Table 1 we summarize the analysis of the case model adopted in the various approaches (techniques or systems). In the majority of them the case includes essentially the content model. In other words a case is considered equal to the product to be recommended. This has the implication that no real learning process is supported by these systems. This is severe limitation which is overcome only by two systems: ICV, where they suggested a case representing the user interest and DieToRecs where the authors proposed a case representing a user interaction with the system. Hence in our opinion CBR-RSs research must devote much more attention on the case model and particularly in finding a more comprehensive way to include in the case information and knowledge about the user, the recommendation process and especially the outcome of such a process.

Referring to the application of the CBR cycle to recommendation (see Table 2), the majority of the CBR-RSs stress the importance of the retrieval phase. Some systems perform retrieval in two steps. First, cases are retrieved by

Table 1. Cross-dimensional analysis of the case model

Approach	**Case model**
Entree EN	Content
Interest Confidence Value-ICV	Content, Evaluation
DieToRecs DTR	All
Order-Based Retrieval OBR	Content
Compromise-Driven Retrieval CDR	Content
Comparison-Based Retrieval COB	Content
ExpertClerk EC	Content

Table 2. Cross-dimensional analysis of the selected approaches (Retrieval, Reuse, Revise, Review and Retain)

Technique	**Retrieval**	**Reuse**	**Revise**	**Review**	**Retain**	**Iterate**
EN	Sim, Logic	All	Ranking	none	none	Tweaking
ICV	Sim	Eval model	ICV computation	feedback	selective	none
SIR	Sim	Content	Ranking	User edit	All	none
TC	Sim, Logic	Content	Constraints	User edit	All	none
SI	Sim, Grouping	All	none	none	All	Feedback
OBR	Sim, Ordering	All	none	none	none	Tweaking
CDR	Sim, Grouping	All	none	none	none	Tweaking
COB	Sim	All	none	none	none	Feedback
EC	Sim	All	none	none	none	Feedback

similarity, then the cases are grouped or filtered. The use of pure similarity does not seem to be enough to retrieve a set of cases that satisfy the user. This seems to be true especially in those application domains that require a complex case structure (e.g. travel plans). Hence in these domains similarity is piped after a first retrieval performed with a more efficient logic based filtering (e.g. in SQL on a data base implementation of the case base).

The default reuse phase is used in the majority of the CBR-RSs, i.e, all the retrieved cases are recommended to the user. ICV and SIR have implemented the reuse step in a different way. In SIR, for instance, the system can retrieve just a component of the case (e.g. the destination of a travel and discard the selected accommodation). The same systems that implemented non-trivial reuse approaches, have also implemented both the revise phase, where the cases are adapted, and the retain phase, where the new case (adapted case) is stored.

Not all the CBR-RSs analyzed implement the review phase, allowing the user to modify/configure the proposed solution (recommendation), or implementing an automatic solution to product reconfiguration. Conversely, many systems cycle the recommendation process, either letting the user to tweak the original query or incorporating (system-driven) explicit or implicit feedbacks collected during the interaction.

7 Conclusions

Looking for products on the Internet is not an easy task. There is a huge quantity of information and on-line there is no human advisor that can help customers to identify what products better fit their preferences. The CBR methodology has been used extensively and successfully to build intelligent applications helping users to cope with these problems.

In this paper we have presented a partial review of the CBR recommender systems literature. We have found that it is often unclear how and why the proposed recommendation methodology can be defined as case-based. In fact, the classical CBR problem solving loop, most of the time, is implemented only partially and sometime is not clear whether a CBR stage (retrieve, reuse, revise, review, retain) is implemented or not. For this reason, we have proposed the unifying framework illustrated in this paper to make possible a coherent description of different CBR-RSs. This framework helps to describe to what extent a recommender system exploits the classical CBR cycle.

We believe, that with such an initial common view it will be easier to understand what the research projects in the area have already delivered, how the existing CBR-RSs behave and which are the topics and the features that could be improved in future systems. We plan to extend this work analyzing more CBR recommendation techniques and to formalize the case model representation using a case representation language such as CBML [10, 9] or CASUEL [13].

References

1. A. Aamodt and E. Plaza. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.
2. D. W. Aha. The omnipresence of case-based reasoning in science and application. *Knowledge-Based Systems*, 11(5-6):261–273, 1998.
3. D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling, UM '99*, Banff, Canada, 1999.
4. D. Bridge and A. Ferguson. Diverse product recommendations using an expressive language for case retrieval. In S. Craw and A. Preece, editors, *Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002*, pages 43–57, Aberdeen, Scotland, 4 - 7 September 2002. Springer Verlag.
5. D. Bridge and A. Ferguson. An expressive query language for product recommender systems. *Artificial Intelligence Review*, 18:269–307, 2002.
6. M. L. Brigitte Bartsch-Sporl and A. Hbner. Case-based reasoning - survey and future directions. In *XPS 99*, pages 67–89. Springer Verlag, 1999.
7. R. Burke. Knowledge-based recommender systems. In J. E. Daily, A. Kent, and H. Lancour, editors, *Encyclopedia of Library and Information Science*, volume 69. Marcel Dekker, 2000.
8. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
9. L. Coyle, D. Doyle, and P. Cunningham. Representing similarity for cbr in xml. In P.Funk and P. Calero, editors, *Advances in Case-Based Reasoning, Proceedings of the 7th European Conference on Case Based Reasoning, ECCBR 2004*, pages 119–127, Madrid, Spain, 2004. Springer Verlag.
10. L. Coyle, C. Hayes, and P. Cunningham. Representing cases for cbr in xml. In *Proceedings of 7th UKCBR Workshop*, Peterhouse, Cambridge, UK, 2002.
11. D. R. Fesenmaier, F. Ricci, E. Schaumlechner, K. Wöber, and C. Zanella. DIETORECS: Travel advisory for multiple decision styles. In A. J. Frew, M. Hitz, and P. O'Connors, editors, *Information and Communication Technologies in Tourism 2003*, pages 232–241. Springer, 2003.
12. J. Kolodner. *Case-Based Reasoning.* Morgan Kaufmann Publishers, San Mateo, CA, 1993.
13. M. Manago, R. Bergmann, N. Conruyt, R. Traphoner, J. Pasley, J. L. Renard, F. Maurer, S. Wess, K.-D. Althof, and S. Dumont. *CASUEL:A Common Case Representation Language.* Esprit Project 6322, Deliverable D1.

14. L. McGinty and B. Smyth. Comparison-based recommendation. In S. Craw and A. Preece, editors, *Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002*, pages 575–589, Aberdeen, Scotland, September 2002. Springer Verlag.
15. L. McGinty and B. Smyth. Deep dialogue vs casual conversation in recommender systems. In F. Ricci and B. Smyth, editors, *Recommendation and Personalization in eCommerce, Proceedings of the AH'2002 Workshop*, pages 80–89, Malaga, Spain, May, 28th 2002. University of Malaga Press.
16. L. McGinty and B. Smyth. On the role of diversity in conversational recommender systems. In A. Aamodt, D. Bridge, and K. Ashley, editors, *ICCBR 2003, the 5th International Conference on Case-Based Reasoning*, pages 276–290, Trondheim, Norway, June 23-26 2003.
17. D. McSherry. Diversity-conscious retrieval. In S. Craw and A. Preece, editors, *Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002*, pages 219–233, Aberdeen, Scotland, 4 - 7 September 2002. Springer Verlag.
18. D. McSherry. Similarity and compromise. In A. Aamodt, D. Bridge, and K. Ashley, editors, *ICCBR 2003, the 5th International Conference on Case-Based Reasoning*, pages 291–305, Trondheim, Norway, June 23-26 2003.
19. B. Miller, I. Albert, S. Lam, J. Konstan, and J. Riedl. Movielens unplugged: Experiences with an occasionally connected recommender system. In *Proceedings of ACM 2003 International Conference on Intelligent User Interfaces (IUI'03).* ACM Press, 2003.
20. N. Mirzadeh, F. Ricci, and M. Bansal. Supporting user query relaxation in a recommender system. In *E-Commerce and Web Technologies, Proceedings of the 5th International Conference, EC-Web 2004*, LNCS 3182, pages 31–40, Zaragoza, Spain, August/Semptember 2004. Springer.
21. N. Mirzadeh, F. Ricci, and M. Bansal. Feature selection methods for conversational recommender systems. In *Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Services*, Hong Kong, 29 March - 1 April 2005. IEEE Press.
22. M. Montaner, B. López, and J. L. de la Rosa. Improving case representation and case base maintenance in recommender systems. In S. Craw and A. Preece, editors, *Advances in Case-Based Reasoning, Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002*, pages 234–248, Aberdeen, Scotland, 4 - 7 September 2002. Springer Verlag.
23. B. Mougouie, M. M. Richter, and R. Bergmann. Diversity-conscious retrieval from generalized cases: a branch and bound algorithm. In A. Aamodt, D. Bridge, and K. Ashley, editors, *ICCBR 2003, the 5th International Conference on Case-Based Reasoning*, pages 319–331, Trondheim, Norway, June 23-26 2003.
24. S. S. Ralph Bergmann and A. Stahl. Intelligent customer support for product selection with case-based reasoning. In P. S. S. Javier Segovia and M. Niedzwiedzinski, editors, *E-Commerce and Intelligent Methods*, pages 322–341. Physica-Verlag, 2002.
25. P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.
26. F. Ricci, B. Arslan, N. Mirzadeh, and A. Venturini. ITR: a case-based travel advisory system. In S. Craw and A. Preece, editors, *6th European Conference on Case Based Reasoning, ECCBR 2002*, pages 613–627, Aberdeen, Scotland, 4 - 7 September 2002. Springer Verlag.

27. F. Ricci and B. Smyth, editors. *Recommendation and Personalization in eCommerce, Proceedings of the AH'2002 Workshop*, Malaga, Spain, May, 28th 2002. Universidad de Malaga.
28. F. Ricci, A. Venturini, D. Cavada, N. Mirzadeh, D. Blaas, and M. Nones. Product recommendation with interactive query management and twofold similarity. In A. Aamodt, D. Bridge, and K. Ashley, editors, *ICCBR 2003, the 5th International Conference on Case-Based Reasoning*, pages 479–493, Trondheim, Norway, June 23-26 2003.
29. F. Ricci, K. Woeber, and A. Zins. Recommendations by collaborative browsing. In *Information and Communication Technologies in Tourism 2005*, pages 172–182. Springer Verlag, Wien - New York, 2005.
30. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of WWW10 Conference*, pages 285–295, Hong Kong, May 1-5 2001. ACM.
31. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *ACM Conference on Electronic Commerce*, pages 158–167, 2000.
32. J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1/2):115–153, 2001.
33. H. Shimazu. ExpertClerk: Navigating shoppers buying process with the combination of asking and proposing. In B. Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001*, pages 1443–1448, Seattle, Washington, USA, August 4-10 2001. Morgan Kaufmann.
34. H. Shimazu. Expertclerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops. *Artificial Intelligence Review*, 18:223–244, 2002.
35. B. Smyth and P. McClave. Similarity vs diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning*, Springer-Verlag, 2001.
36. I. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems.* Morgan Kaufmann, 1997.
37. I. H. Witten and E. Frank. *Data Mining.* Morgan Kaufmann Publisher, 2000.

Improving the Performance of Recommender Systems That Use Critiquing*

Lorraine McGinty and Barry Smyth

Adaptive Information Cluster, Smart Media Institute,
Department of Computer Science, University College Dublin (UCD), Ireland
{firstname.lastname}@ucd.ie

Abstract. Personalization actions that tailor the Web experience to a particular user are an integral component of recommender systems. Here, product knowledge - either hand-coded or "mined" - is used to guide users through the often overwhelming task of locating products they will like. Providing such intelligent user assistance and performing tasks on the user's behalf requires an understanding of their goals and preferences. As such, user feedback plays a critical role in the sense that it helps steer the search towards a "good" recommendation. Ideally, the system should be capable of effectively interpreting the feedback the user provides, and subsequently responding by presenting them with a "better" set of recommendations. In this paper we investigate a form of feedback known as critiquing. Although a large number of recommenders are well suited to this form of feedback, we argue that on its own it can lead to inefficient recommendation dialogs. As a solution we propose a novel recommendation technique that has the ability to dramatically improve the utility of critiquing.

1 Introduction

Personalized recommender systems combine ideas from information retrieval, AI, user modelling and interface design to provide users with a more adaptive environment for information retrieval. *Reactive* recommender systems are designed to make recommendations based on a user's query; for example, Entree [4] makes restaurant recommendations based on a query that specifies features such as *cuisine type*, *cost*, etc. In contrast, *proactive* recommenders operate without the need for an explicit query, proactively pushing recommendations to a user; for example, PTVPlus makes TV program suggestions to a user based on their learned viewing preferences [17].

User feedback is a vital component of most recommenders, allowing them to adapt precisely to the needs of target users, and setting recommender systems apart from more traditional information retrieval systems. To date researchers

* The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

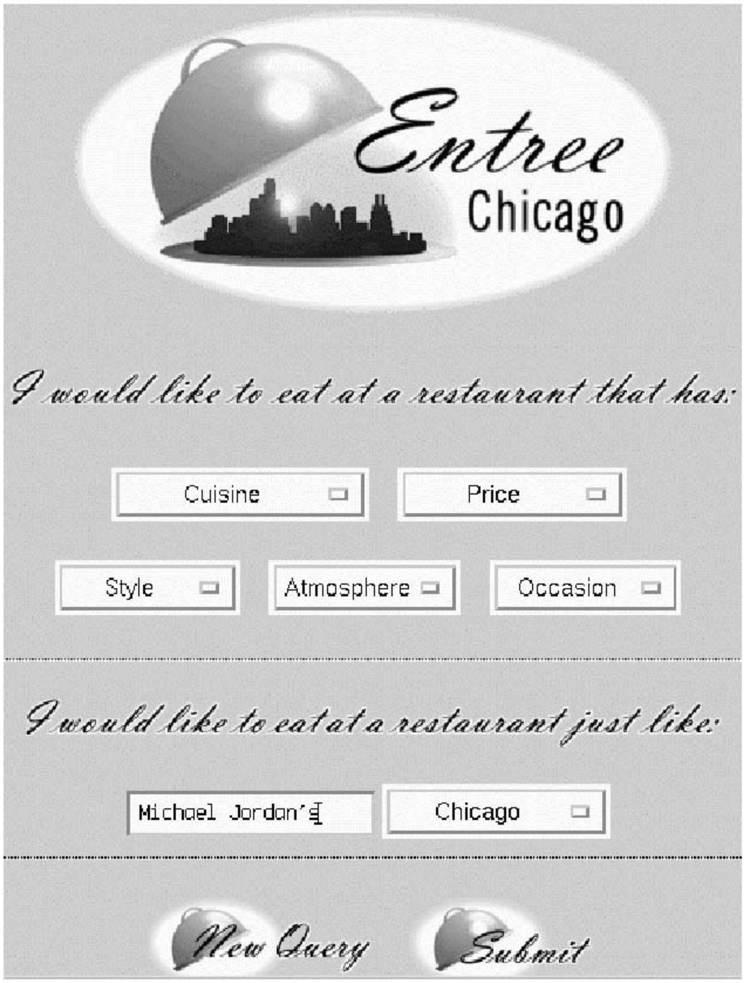
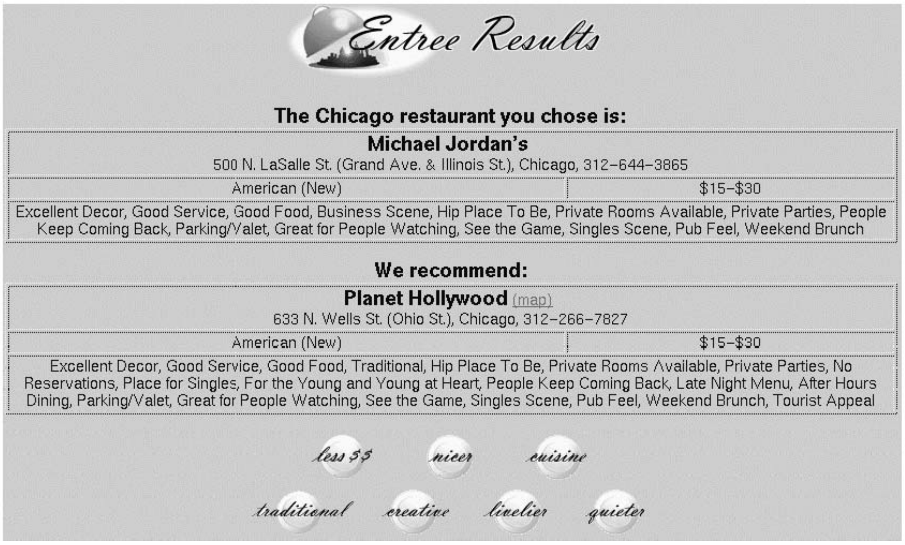

Entree
Chicago
I would like to eat at a restaurant that has:
Cuisine
Price
Style
Atmosphere
Occasion
I would like to eat at a restaurant just like:
Michael Jordan's
Chicago
New Query
Submit

Fig. 1. The Entree Restaurant Recommender: the user specifies their query on the opening screen

have focused on *value elicitation* and *ratings-based* feedback, but recently two alternatives, *critiquing* (or *tweaking*) and *preference-based* feedback, have become increasingly relevant in many important application scenarios.

In this paper we focus on reactive recommenders that use critiquing as their primary source of feedback, as epitomised by the so-called FindMe recommender systems [4,3]. For instance, in the Entree Restaurant Recommender users tend to have limited knowledge about the restaurants being recommended, but are capable of recognising 'what they want when they see it'. Figure 1 shows the initial screen at the start of a new session. The user has two options: (1) they can specify their restaurant preferences by selecting specific features (for instance,



The Chicago restaurant you chose is:

Michael Jordan's

500 N. LaSalle St. (Grand Ave. & Illinois St.), Chicago, 312-644-3865

American (New) | $15-$30

Excellent Decor, Good Service, Good Food, Business Scene, Hip Place To Be, Private Rooms Available, Private Parties, People Keep Coming Back, Parking/Valet, Great for People Watching, See the Game, Singles Scene, Pub Feel, Weekend Brunch

We recommend:

Planet Hollywood (map)

633 N. Wells St. (Ohio St.), Chicago, 312-266-7827

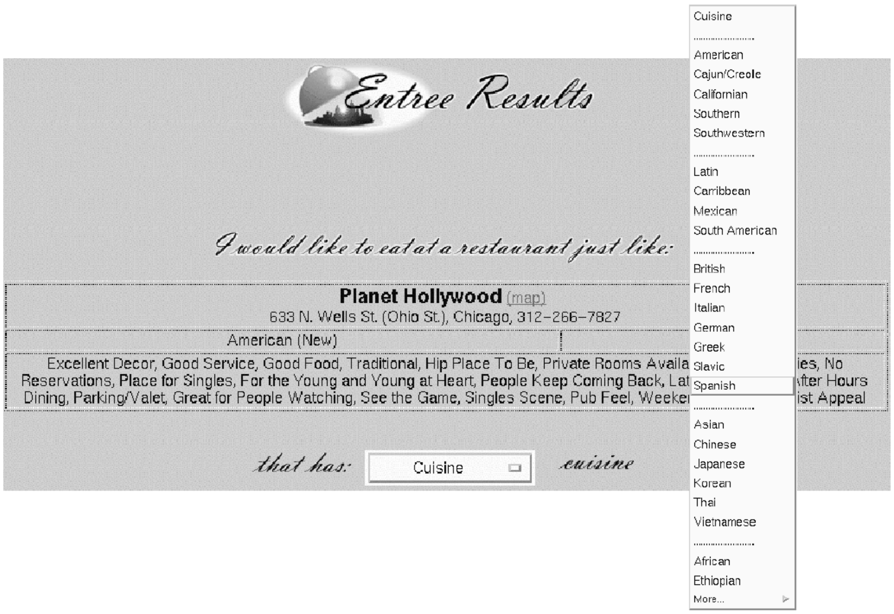
American (New) | $15-$30

Excellent Decor, Good Service, Good Food, Traditional, Hip Place To Be, Private Rooms Available, Private Parties, No Reservations, Place for Singles, For the Young and Young at Heart, People Keep Coming Back, Late Night Menu, After Hours Dining, Parking/Valet, Great for People Watching, See the Game, Singles Scene, Pub Feel, Weekend Brunch, Tourist Appeal

Fig. 2. The Entree Restaurant Recommender: when a recommendation is returned the user can apply any one of a set of pre-defined critiques

preferred location, cuisine, atmosphere, price, etc.) from the drop-down lists provided, or (2) specify the name of a restaurant they already know and like, and use its description as a query. When the user submits their query to the Entree recommender engine the system responds by presenting the user with the most similar restaurant description available (see Figure 2). The user can then either accept this recommendation, or critique it by asking for another that is different in relation to one of seven pre-defined features (e.g., less expensive, nicer, quieter etc.). Figure 3 shows how a user would go about selecting a critique over the cuisine feature. The applied critique then acts as a filter in the retrieval process, such that any restaurant candidates that so not satisfy the critique are not considered for the next recommendation cycle. Importantly, after the first page, no action requires more than one click, and the user is in control of the direction of the search.

Notably, critiquing strikes a appealing balance between the effort that a user must invest in providing feedback and the information content of the feedback; see Section 2. Critiques are relatively easy to apply and they don't require the user to have detailed domain knowledge, and yet they provide valuable information for the recommender. We argue that this makes critiquing an especially useful form of feedback for many practical applications. However, we also highlight how critiquing on its own can lead to protracted recommendation dialogs. As a solution, we describe and evaluate a novel approach to recommendation, called *adaptive selection*, which can be used to significantly enhance the performance of recommenders that use critiquing, shortening recommendation dialogs by up to 60%. Before continuing it is worth noting that in related research we



Planet Hollywood (map)
633 N. Wells St. (Ohio St.), Chicago, 312-266-7827
American (New)
Excellent Decor, Good Service, Good Food, Traditional, Hip Place To Be, Private Rooms Availa ies, No Reservations, Place for Singles, For the Young and Young at Heart, People Keep Coming Back, Lat ıfter Hours Dining, Parking/Valet, Great for People Watching, See the Game, Singles Scene, Pub Feel, Weeke ist Appeal

Fig. 3. The Entree Restaurant Recommender: example of a critique. The user wants to see another recommendation *like* the current one, *but* with a different cuisine.

have demonstrated how this approach can also be used with other forms of feedback such as preference-based feedback [10,7,9].

2 Background

Reactive, content-based recommendation systems typically adopt a *conversational* style of interaction with a user, selecting items for recommendation and eliciting feedback from the user before making the next batch of recommendations (see for example [1,4,3,6,14,15]). Feedback allows a recommender to make better suggestions by adapting its current understanding of the user's requirements. We are especially interested in developing personalized product recommenders for consumer markets, and for use across a variety of devices, from traditional PCs to much more limited devices such as Internet-enabled mobile phones. The ideal feedback method should be low-cost and provide unambiguous information regarding the user's intentions. It should also be applicable across a wide range of user types, even those with limited domain expertise. Finally, it should be possible to capture user feedback on even the most limited of devices.

2.1 User Feedback in Personalized Recommender Systems

As mentioned above, we can usefully categorise feedback strategies in terms of: the *cost* to the user of providing the feedback; the level of *ambiguity* inherent

Feedback	Cost	Ambiguity	Expertise	Interface
Value Elicitation	***	*	***	***
Ratings	**	***	**	*
Critique	**	**	**	*
Preference	*	***	*	*

Key: * Low ** Moderate *** High

Fig. 4. A comparison of feedback strategies

in the feedback; the level of domain *expertise* required by the user to provide the feedback; and the type of user *interface* needed to capture the feedback; see Figure 4. For example, value elicitation (e.g. "I want a 1GHz Pentium PC"), perhaps the most common form of feedback, is a rich source of information; knowing that the user is interested in items with a particular feature allows the recommender to eliminate many irrelevant items from consideration. However, to provide this feedback the user needs a high level of domain expertise to be able specify a reasonable feature value. In addition to this the user must be willing to answer the direct and specialised questions posed by recommenders. Finally, providing detailed feature-level feedback demands a sophisticated user interface (e.g. text-entry, check boxes, drop lists), thus limiting its use on some mobile devices, for example.

In contrast, preference-based feedback (e.g. "I prefer PC1") is an low-cost form of feedback and can be provided by users, through a simple interface, and with only a rudimentary understanding of the domain. It is low-cost because the user can indicate their preference in a single *click*. Contrast this with value elicitation where the user must select a specific feature of a specific case and provide a specific value for that feature. However, on its own a simple preference for an item is inherently ambiguous with respect to the user's intent. Preference-based feedback provides no information about the particular features that have motivated the user's preference or choice. Thus it has only a limited capacity to guide the recommendation process; see [8,10] for recent advances with this form of feedback.

2.2 The Promises and Pitfalls of Critiquing

Neither value elicitation nor preference-based feedback are an ideal form of feedback (i.e. low-cost, unambiguous, minimal expertise, and simple interface), but critiquing appears to come close. It is a special case of value elicitation - instead of providing a specific feature value, the user indicates a feature *critique* (or *tweak*) to constrain the value range of that feature. For example, the tweak,

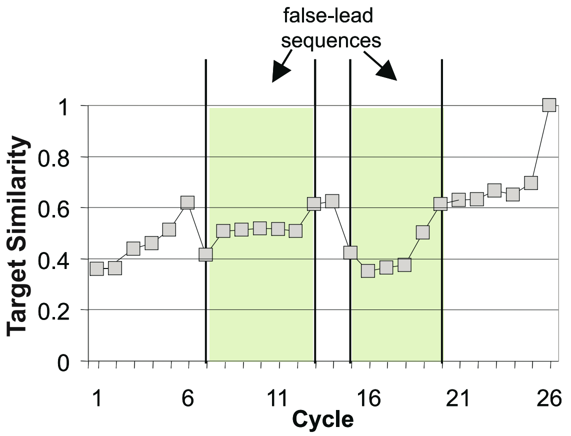


Fig. 5. An example of a recommendation session similarity profile for a sample query

"Show me more like PC1 but cheaper", allows the recommender to focus on products that are less expensive than PC1 in the next recommendation cycle without requiring the user to provide specific price details.

The benefits of critiquing are compelling. It provides a relatively unambiguous indication of the user's current requirement. It is low-cost, easy to implement with the simplest of interfaces, and it can even be applied by users with only a moderate understanding of the product-space. However, some concerns remain. There has been relatively little evaluation work carried out on critiquing, especially with respect to its impact on recommendation efficiency. How effective are individual critiques at focusing the recommendation dialog? Personalized recommender systems that utilise preference-based feedback have a tendency to overfit to the current preference misleading the recommender to follow *false-leads* into an irrelevant portion of the item space [8]. Does this problem exist with critiquing and if so how can it be solved?

Consider a recommender system that presents a user with a set of 3 recommendations during each recommendation cycle and expects the user to guide the next recommendation cycle by critiquing one of the features of the preferred item. Figure 5 graphs the *similarity profile* of such a recommender for a single query taken from the Travel domain (see Section 4.2). The graph plots the similarity between the critiqued item and some predefined ideal target (see Section 4.2). Here we see that 26 cycles are required to locate the target item for this query. For the sake of clarity we have chosen a particularly *difficult* query that results in a protracted recommendation dialog. Obviously it is unlikely that users will tolerate such a lengthy dialog; in fact it is quite likely that most users will give up after 5 to 10 cycles. Nevertheless such lengthy dialogs can occur with critiquing as a source of feedback.

Looking at this similarity profile in more detail shows that good progress is made towards the target during the first 6 cycles, as indicated by the increasing similarity of the preferred case to the target. However, recommendations made in the 7th cycle are false-leads since none of them are closer to the target problem than the preferred case from the previous cycle. In fact the best of these false-leads, which is the preference case for the 7th cycle, has a target similarity of only 0.4 which is less than the 0.6 target similarity for the previous preference case; essentially this set of recommendations has led the user astray. Another false-lead is followed from cycle 15 and, in all, these false-leads contribute 11 extra cycles to the session resulting in a lengthy recommendation dialog that does not guarantee to present the user with increasingly relevant recommendations. We have found this type of similarity profile to be common in critiquing-based recommenders, especially for queries that are initially vague or difficult to satisfy. And, unless false-leads can be detected and controlled, this issue is likely to limit the practical success of such recommender systems .

3 Critiquing in Comparison-Based Recommendation

Comparison-based recommendation [8] is a framework for reactive, content-based recommendation systems that emphasises the importance of feedback during each recommendation cycle. Comparison-based recommendation was originally proposed as a test-bed for preference-based feedback strategies but is equally applicable to other forms of feedback, especially critiquing. Hence, in this paper we investigate its use with critiquing, and propose and evaluate adaptive selection (AS) as a recommendation strategy capable of addressing the false-lead problem.

3.1 A Review

Comparison-based recommendation is a conversational recommendation process whereby the user engages in an extended dialog made up of a sequence of recommendation cycles. The basic algorithm (fully described in Figure 6) consists of 3 key steps: (1) new items are *recommended* to the user based on the current query; (2) the user *reviews* the recommendations and indicates a preference case as *feedback*; (3) information about the difference between the selected item and the remaining alternatives is used to *revise* the query for the next cycle [7]. The recommendation process terminates either when the user is presented with a suitable item or when they give up.

Ordinarily the **ItemRecommend** step (line 4 of Figure 6) retrieves the k most similar items to the query, the user indicates that case from among the recommended items which best matches their needs in the **UserReview** step (line 5 of Figure 6), and the **QueryRevise** step (line 6 of Figure 6) focuses on updating the current query based on what can be learned from the user's feedback. Previous work has examined a number of query update strategies for use with preference-based feedback (e.g. "show me *more like this* (MLT)") whereby the query is elaborated with features from the preferred case that best reflect the user's implicit preferences. Further details on this topic can be found in [7].

```
define Comparison-Based-Recommend(q, CB, k)
  $i_{p-1}$ , $i_p$ ← null
  do
    R ← ItemRecommend(q, CB, c, k, $i_p$, $i_{p-1}$)
    <$i_p$,c> ← UserReview(R, CB)
    Q ← QueryRevise(q, $i_p$, R)
    $i_{p-1}$ ← $i_p$
  until UserAccepts($i_p$)

define QueryRevise(q, $i_p$, R)
  R′ ← R - {$i_p$}
  q ← $i_p$
return q

define UserReview(R, CB)
  $i_p$ ← user’s preferred case from R
  c ← user critique for some f ∈ $i_p$
  CB ← CB - R
return <$i_p$,c>

define ItemRecommend(q, CB, c, k, $i_p$, $i_{p-1}$)
  CB′ ← {i ∈ CB | Satisfies(i,c)}
  if($i_p$ != null) && ($i_p$ == $i_{p-1}$)
     CB′′ ← sort CB′ in by decreasing sim to q
     R ← top k items in CB′′
  else R ← BoundedGreedySelection(q, CB, k, b)
return R

define BoundedGreedySelection (q, CB, k, b)
  CB' := bk items in CB that are most similar to q
  R := {}
  For j := 1 to k
    Sort CB' by Quality(q,i,R) for each case i in CB‘
    R  := R + First(CB')
    CB′ := CB′ - First(CB')
  EndFor
return R
… where Quality(q,i,R) = $\alpha$.Sim(q,I)+(1- $\alpha$)RelDiv(i,R)
        RelDiv(i,R)= $\sum_{j=1..m}$(1-Sim(i,$r_j$))/m if R={}; = 1 otherwise
```

Fig. 6. The comparison-based recommendation algorithm adapted for critiquing with adaptive selection

3.2 Adapting Comparison-Based Recommendation for Critiquing

Adapting comparison-based recommendation for critiquing is relatively straightforward. For example, in the Travel domain suppose a user is recommended a \$2000, two-week vacation in Venice, Italy in a three-star hotel, but indicates that they are looking for something similar but cheaper (a $<$ \$2000 critique on the price feature). During the next cycle only those items whose price is less than \$2000 will be considered for selection. Thus **ItemRecommend** becomes a two step process. In step one, all items that fail to satisfy the critique (those with price features $\geq$ \$2000) are eliminated. In step two, the remaining items are ranked according to their similarity to the updated query, and the top k are selected. Note that in this work we will assume that query revision follows the simple more like this (MLT) strategy in which the current preferred item is used as the new query in the next recommendation cycle (see [8]). We make use of four basic types of critiques: $<$, $>$, $=$, and $\neq$. Obviously, $<$ and $>$ can only be applied to numeric features.

3.3 Adaptive Selection

As mentioned earlier, the standard item selection strategy selects the k most similar items to the query, but recent work suggests that factors other than similarity should be considered during selection. Research has shown that similarity-based methods tend to reduce recommendation diversity if the selected items are similar to each other, as well as similar to the query [2,12,15,16,18]. By increasing the degree of diversity amongst recommended items it may be possible to cover more of the item space, in a given cycle, and as a consequence perhaps increase recommendation efficiency.

A number of diversity enhancing mechanisms have already been proposed in the literature [2,12,15,16,18], but these techniques introduce diversity in a static or uniform way. While increasing selection diversity can offer a recommender an improved ability to cover the item space, it may lead to new inefficiencies [9]. By design, diversity enhancing selection methods pass over certain items that might otherwise be selected, and this introduces a problem if the target happens to be one of these passed-over items; in this case the correct target item was passed over in the blind pursuit for improved recommendation diversity. A more flexible mechanism for introducing diversity seems appropriate. If the recommender is focused in the region of the target problem, similarity should be emphasised, otherwise diversity should play a role. However, it is not immediately obvious how to judge when the recommender is focused in the target region.

The key idea in *adaptive selection* (AS) is that by determining whether the most recent recommendations represent an improvement on those made in the previous cycle it is possible to judge whether or not the recommender is correctly focused in its search, and thus whether diversity should be introduced into item selection. This is achieved by making two modifications to the standard similarity-based item selection strategy (i.e. **ItemRecommend** method outlined in lines 22-27 of Figure 6).

Firstly, instead of making k new recommendations in each new cycle, the current preference item (the critiqued item) is added to $k-1$ new recommendations; we refer to this as *carrying the preference* (CP). On its own this modification introduces redundancy, in the sense that a previously seen item is repeated in one or more future cycles. However, including the previous preference makes it possible to avoid the false-lead problem mentioned above. If none of the $k-1$

new items are relevant then the user can continue to critique the carried preference instead of being forced to critique a less relevant recommendation. For example, the similarity profile for this method is included in Figure 7, with 14 cycles needed, and a number of similarity *plateaus* in evidence (cycles 4-6, 8-10 and 11-12) where the user reselects preference items.

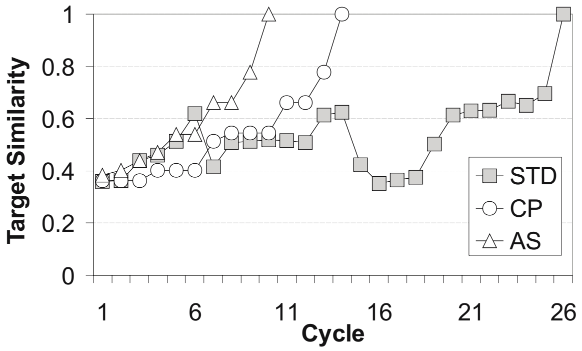


Fig. 7. Preference case similarity profiles for three different versions of the critiquing strategy; standard critiquing (STD), critiquing with carrying the preference (CP), and critiquing with adaptive selection (AS)

If the user critiques the carried preference then it suggests that the other $k-1$ items are less relevant than the carried item, and thus that the recommender has failed to make positive progress towards the target. If the user chooses to critique one of the newly recommended items, then it must be because it is closer to the target, and thus positive progress has been made. The second part of the AS strategy takes advantage of this idea by introducing diversity into the next round of selection if and only if the user has critiqued the carried preference. The similarity profile for AS is also presented in Figure 7. Once again a significant reduction in the number of cycles is evident, without any drops in target similarity; this time only 10 cycles are needed, with the carried preference reselected in cycle 6 and 8.

4 Experimental Evaluation

The success of any conversational recommender depends critically of the efficiency of its dialogs. Generally, shorter dialogs that present the user with increasingly good recommendations are more likely to lead to greater success than longer ones [5,11]. In this section we describe the results from a comprehensive evaluation of critiquing focusing on the benefits of our new CP and AS techniques over standard critiquing.

4.1 Basic Assumption

We assume that the user has some product requirement and they are looking for a product that meets this requirement. However, we further assume that this requirement is sufficiently vague and that the user cannot simply list the features and search from there. Rather, the user provides some initial features and then is helped to locate a good product by critiquing additional features from intermediate recommendations. These intermediate recommendations help to influence the user's requirements and elaborate their initial query.

4.2 Setup

In this evaluation we compare the performance of three critiquing techniques in a comparison-based recommender (with $k = 3$):

1. *STD* uses basic critiquing and is our *benchmark*,
2. *CP* implements preference carrying, and
3. *AS* implements adaptive selection.

Data-Sets. Two public data sets, from Travel and PC domains are used for the evaluation. The familiar *Travel* case-base contains 1024 cases, each describing a specific vacation in terms of features such as *region*, *duration*, *accommodation*, *price* etc (see Figure 8). The *PC* case-base contains a set of 120 cases, describing a unique PC in terms of features such as *manufacturer, processor type, processor speed, etc* (see Figure 9).

HolidayType	Bathing
Price ($)	2498
# of People	2
Region	Egypt
Transportation	Flight
Duration (Days)	10
Season (Month)	April
Accommodation	Two Stars
Hotel	Hotel White House

Fig. 8. Sample PC case

Manufacturer	Compaq
Processor Type	Intel Pentium III
Processor Speed (Mhz)	1300
Monitor (Inches)	14
Type	Laptop
Memory (MB)	256
Drive Capacity (GB)	20
Price ($)	2300

Fig. 9. Sample TRAVEL case

Methodology. Using a leave-one-out methodology, each item (*base*) of a data set is temporarily removed and used in two ways. First it serves as the basis for a set of queries constructed by taking random subsets of item features. Second, we select the item that is most similar to the original base. These items serve as the recommendation *targets* for the experiments. Thus, the base represents the ideal query for a user, the generated query is the initial query that the user provides to the recommender, and the target is the best available item for the user based on their ideal. Each generated query is a test problem for the recommender, and in each recommendation cycle a tweak is applied to the item that is most similar to the known target item; the tweak is based on a random feature of this item.

For each data set, three different groups of queries are generated of varying degrees of difficulty (*easy*, *moderate*, *difficult*); difficulty is based on the number of cycles required by STD. Because of the complexity differences between the PC and Travel data sets the cycle threshold that corresponds to a particular level of difficulty is not the same for both data sets and, in general, PC queries are less difficult than Travel queries.

4.3 Recommendation Efficiency

To test recommendation efficiency the leave-one-out method outlined above is used for each query, from both data sets, across the three recommenders, and the average number of cycles and unique items presented to the user are measured.

Results. The results for Travel and PC are summarised in Figure 10(a-d) as graphs of mean cycles and items for each algorithm and query group. They clearly indicate the potential benefits of the CP and AS strategies relative to standard critiquing (STD), especially as query difficulty increases. In Travel we find that for simple queries it takes an average of 9.52 recommendation cycles (and 28.55 unique items) for standard critiquing to locate the target items. In contrast, CP takes 6.93 cycles (and 15 items) and AS takes 6.7 cycles (and 14 items). In other words, the AS strategy achieves a relative reduction in cycles of 30% and, in items, of 50%, compared to STD. Due to the reduced complexity of the PC domain (i.e. fewer features and fewer cases) the benefits across simple queries are less clear than in Travel, with minor improvements in the number of unique items presented to the user (8% and 12% for AS and CP over STD), but minor increases in the number of cycles needed.

The benefits of CP and AS are further enhanced in both domains with increasingly difficult queries. In the Travel domain, the number of cycles and items required by STD increases to 15.53 and 46.59, for the moderate query groups, and in the PC domain the corresponding values are 5.81 and 17.44. However, the increase in cycles and items proceeds at a much slower rate for the CP and AS methods, leading to incremental improvements in their benefits relative to STD. For example, in the Travel domain, for the difficult queries a dramatic 60% reduction in the number of items (compared to STD) is achieved for AS, and a 52% reduction for CP. Similarly, in the PC domain, for the difficult queries a 44% reduction in items is achieved for AS with a 31% reduction for CP.

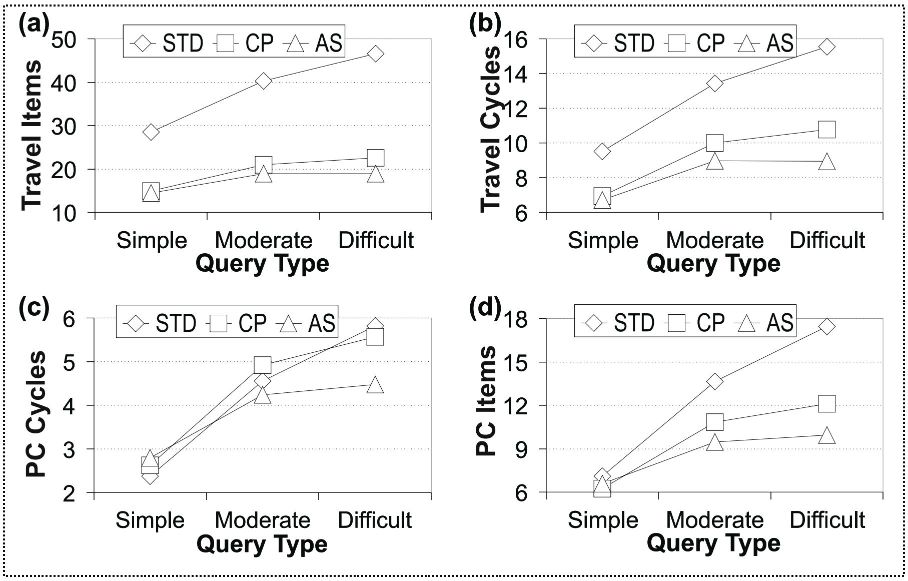


Fig. 10. Efficiency results from the Travel and PC domains

Discussion. These results demonstrate that CP and AS can have a significant positive impact on the efficiency of critiquing. A consistent benefit is found for AS over CP and the CP and AS benefits are related to increasing query difficulty - fewer STD cycles mean less opportunity for improvements by CP and AS. The strength of this relationship can be assessed by measuring the correlation between the cycles needed by STD and the corresponding benefit enjoyed by CP and AS (in terms of cycles or items). These correlations turn out to be in excess of 0.85 for the various possible combinations. For example, combining the benchmark cycles across both PC and Travel and measuring the correlation between these cycles and the associated benefit for AS, in terms of a reduction to the number of items, yields a correlation of 0.87. In practice then, the above results indicate that significant reductions in the number of items can be achieved by AS for queries that ordinarily demand about 3 or more cycles, with reductions in cycles available for queries that require about 5 or more cycles.

4.4 Preference Tolerance

The above results assume that the recommendation dialog ends when the predetermined target item is selected. This is analogous to a user seeking out a very specific item but, in reality, users are likely to be more flexible in their requirements, accepting sub-optimal items that are close to the target.

The experiment is repeated, but instead of terminating a recommendation dialog when the target has been found, it is terminated once an item is found that is within a specific similarity of the target. We test similarity thresholds

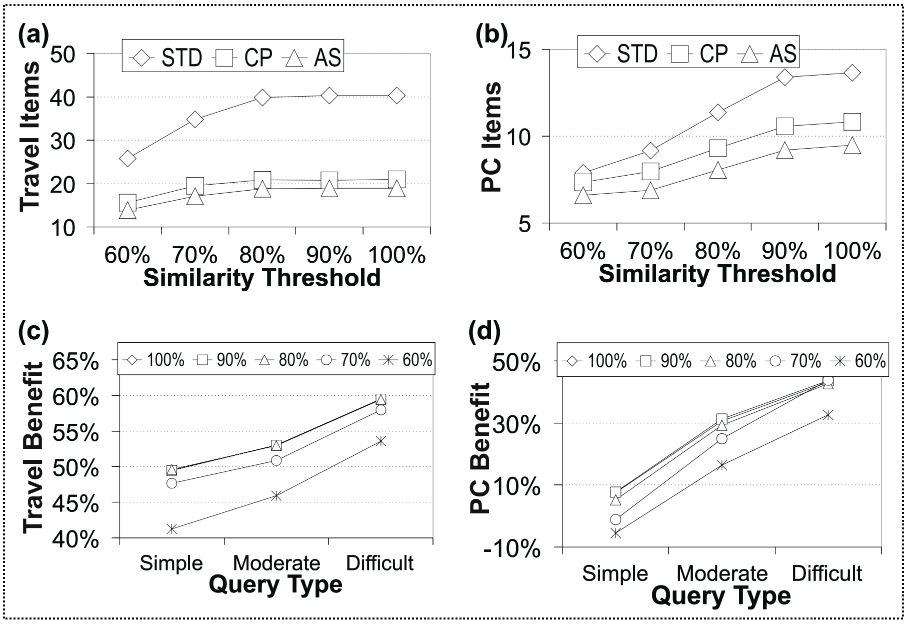


Fig. 11. Preference tolerance results from the Travel and PC domains

from 60% to 100%; 100% corresponds to the previous setup where the dialog terminates with the optimal target case.

Results. The results are presented in Figure 11(a-d)in a number of ways. First, Figure 11(a&b) present the results for Travel and PC, graphing the mean items per recommendation dialog for different similarity thresholds. In these graphs, for reasons of brevity, the results for the moderate queries in each domain are shown with qualitatively similar results observed for the other query groups. Unsurprisingly, as the success criteria becomes more rigid the number of unique items presented to the user tends to increase. For example, in the Travel domain, the number of unique items presented to the user by STD increases from 25.76 (60% threshold) to just above 40 (100% threshold) with a similar increase for PC (7.8 items increases to more than 13 items).

Once again we notice significant benefits for CP and AS, with AS consistently out-performing CP. For example, in Travel the number of items required by AS at the 60% threshold is 13.92 (a reduction relative to STD of 45.94%), which increases to 18.93 at the 100% threshold (a reduction of 53.01%). The AS (and CP) benefit is seen to increase with the similarity threshold, at least in the case of this query group. Figure 11(c&d) confirms this across all query groups by graphing the relative (items) benefit of AS (relative to STD) against query difficulty. In Travel the benefits increase from a minimum of 41% to a maximum of 60%, and in PC domain from -5% to 44%; once again it is worth noting that the -5% is associated with a mean number of cycles of 1.55 at the 60% similarity threshold and is the only situation where a negative impact is found in the number of items metric for AS.

Discussion. In summary, the previously observed benefits for CP and AS are maintained under more flexible termination conditions. Indeed the AS benefits increase as the termination condition becomes more rigid, a result that is likely to be especially important in domains where users tend to seek out optimal or near-optimal products and items; for example, domains with high-cost or highly specialised products (e.g. holidays, jewellery, PCs, high-end audio-visual equipment etc.).

It is worth noting that it is actually possible to achieve optimal (100% similarity threshold) recommendations using AS with the same number of items, or fewer, than the STD method requires to achieve recommendations that are only 60% optimal, in Travel, or 70% optimal, in PC. For example, for the moderate query group in Travel (see Figure 11(a)), AS requires an average of 18.93 items at the 100% similarity threshold, whereas STD requires 25.76 items at the 60% similarity threshold. Comparable results are found, but not reported here in detail, for other query groups, although they are less pronounced for the PC domain.

4.5 Preference Noise

So far we have assumed that in each cycle the user critiques the item that is most similar to the target. If we break this assumption what will be the impact on the observed benefits of CP and AS? To investigate this issue we repeat the original experiment except that noise is introduced into the critiquing stage by perturbing the similarities, between each recommended case and the target, by some random amount within a set noise limit.

For example, a noise limit of 10% means that each similarity value will change by a random amount up to +/-10% of its actual value. This will potentially change the ordering of recommended items during each cycle so that, depending on the closeness of the original similarity values and the size of the noise limit, the second or even third most similar item to the target may be critiqued instead of the most similar item. This approach mimics the situation where users make preference mistakes more frequently when recommended items are all similar to the target to a more or less equivalent extent.

Results. Figures 12(a&b) graph the mean number of items presented to the user versus the noise limit for moderate queries in the Travel and PC domains; similar results are observed for the simple and difficult queries. These results are somewhat surprising in terms of their consistency across the various noise levels. For example, no significant change in the mean number of items is found for STD, CP or AS for the noise levels examined, which in turn leads consistent benefits being observed for AS in terms of the number of items presented to

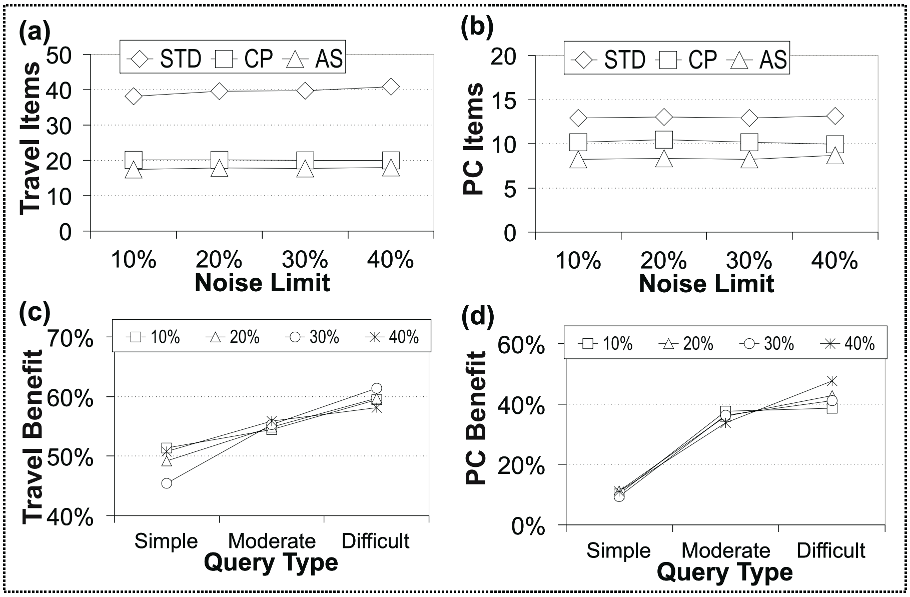


Fig. 12. Preference noise results from the Travel and PC domains

the user; see Figure 12(c&d). For example, in the PC domain we find that the AS benefits are virtually identical for different levels of noise for simple and moderate queries, and not significantly different even for the difficult queries. For example, in PC, the AS benefit for simple queries is consistently about 10% for all levels of noise, rising to about 36% for moderate queries, and 41% for the difficult queries; broadly similar pattern is found in the Travel domain. Similar results are observed for the CP technique and with respect to numbers of cycles, but are not reported here in detail for reasons of brevity.

Discussion. In summary, we find that the benefits of AS (and CP) are maintained across noise levels. But perhaps the most notable feature of these results, is the lack of observed sensitivity to different levels of noise. In recent work on pure preference-based feedback, when a user selects an item other than the one that is most similar to the target, recommendation efficiency degraded significantly [10]. It appears that critiquing offers some protection against this type of noise, compensating for sub-optimal preferences. The level of protection probably depends on differences in similarity to the target between the preferred/critiqued item and the other items in the recommendation cycle. In our experiments the difference in similarity to the target between the closest and furthest item from the target, in a single recommendation cycle, is approximately 25-40% of the closest item's similarity to the target. This is a significant difference and no doubt explains the degradation in recommendation efficiency observed when preference-based feedback is used on its own. However, our results indicate that critiquing has the ability to overcome susch a potential drop in target-similarity for the preferred item; presumably because the filtering of items by the chosen critique helps to maintain the relevancy of remaining items for the next cycle.

5 Conclusions

Feedback plays a critical role in many personalized recommender systems and different types of feedback strike a different balance with respect to user cost, feedback ambiguity, user expertise, and interface requirements. Consumer-oriented product recommenders, in particular, are well suited to critiquing. However, on its own this form of feedback can lead to inefficient recommendation dialogs.

As a solution we have described adaptive selection, a novel approach to item recommendation that effectively balances the influence of similarity and diversity during item selection. Adaptive selection has been shown to significantly improve recommendation efficiency under a variety of experimental conditions. Moreover, the benefits of AS increase with query difficulty, and reductions of up to 60% in the number of items presented to the user (relative to standard critiquing) have been reliably observed.

It is worth noting that adaptive selection is not limited to recommenders that use critiquing as their primary form of feedback. As mentioned earlier, related work has looked at how adaptive selection can also significantly enhance the effectiveness of preference-based feedback (PBF)[10,19]. Indeed in a comparison between a recommender that implements standard critiquing and an equivalent recommender that integrates preference-based feedback with adaptive selection, we were surprised to find the latter to be the more efficient approach, delivering dialog reductions of between 37% and 52%. In other words adaptive selection was able to improve the performance of preference-based feedback to such an extent that it was able to compete with critiquing, a far more inherently powerful form of feedback.

Finally, the AS approach is just one way to improve the utility/efficiency of the standard critiquing approach, by adapting the strategies used to retrieve items in line with the recommendation focus. Another alternative is to look at improving critiquing utility/efficiency by adapting how recommenders, using this feedback mechanism, interface with the user. As described earlier, the standard critiquing approach presents users with a set of fixed critiques, where each critique usually operates over a single feature [3,4]. Recent related research has looked at ways of automatically generating compound critiques during each recommendation cycle, and proposes a strategy for selecting a small number of high-quality compound critiques for presentation to the user [13]. Very briefly, compound critiques are *dynamically* generated on a cycle-by-cycle basis by data mining the feature patterns of the remaining product cases. These compound critiques are then filtered and a high-quality subset is presented to the user along with the standard (fixed) unit critiques; in this way the feedback offered by the recommender system interface can be adapted to the current recommendation session and cycle. Initial experiments indicate that this critiquing strategy has the potential to offer significant performance improvements—reducing session length by up to nearly 70%—as well as offering some user-interaction benefits such as improved explanatory power[13].

References

1. D.W. Aha, L.A. Breslow, and H. Muoz-Avila. Conversational case-based reasoning. *Applied Intelligence*, 14:9–32, 2000.
2. D. Bridge. Product Recommendation Systems: A New Direction. In D. Aha and I. Watson, editors, *Workshop on CBR in Electronic Commerce at The International Conference on Case-Based Reasoning (ICCBR-01)*, 2001. Vancouver, Canada.
3. R. Burke. Knowledge-based Recommender Systems. *Encyclopedia of Library and Information Systems*, 69(32), 2000.
4. R. Burke, K. Hammond, and B.C. Young. The FindMe Approach to Assisted Browsing. *Journal of IEEE Expert*, 12(4):32–40, 1997.
5. M. Doyle and P. Cunningham. A Dynamic Approach to Reducing Dialog in On-Line Decision Guides. In E. Blanzieri and L. Portinale, editors, *Proceedings of the Fifth European Workshop on Case-Based Reasoning, (EWCBR-00)*, pages 49–60. Springer, 2000. Trento, Italy.
6. M. Goker and C. Thompson. Personalized Conversational Case-based Recommendation. In E. Blanzieri and L. Portinale, editors, *Proceedings of the Fifth European Workshop on Case-based Reasoning, (EWCBR-00)*, pages 99–111. Springer-Verlag, 2000.
7. L. McGinty and B. Smyth. Comparison-Based Recommendation. In Susan Craw, editor, *Proceedings of the Sixth European Conference on Case-Based Reasoning (ECCBR-02)*, pages 575–589. Springer, 2002. Aberdeen, Scotland.
8. L. McGinty and B. Smyth. Deep Dialogue vs Casual Conversation in Recommender Systems. In F. Ricci and B. Smyth, editors, *Proceedings of the Workshop on Personalization in eCommerce at the Second International Conference on Adaptive Hypermedia and Web-Based Systems (AH-02)*, pages 80–89. Springer, 2002. Universidad de Malaga, Malaga, Spain.
9. L. McGinty and B. Smyth. The Role of Diversity in Conversational Systems. In D. Bridge and K. Ashley, editors, *Proceedings of the Fifth International Conference on Case-Based Reasoning (ICCBR-03).* Springer, 2003. Troindheim, Norway.
10. L. McGinty and B. Smyth. Tweaking Critiquing. In *Proceedings of the Workshop on Personalization and Web Techniques at the International Joint Conference on Artificial Intelligence (IJCAI-03).* Morgan-Kaufmann, 2003. Acapulco, Mexico.
11. D. McSherry. Minimizing dialog length in interactive case-based reasoning. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 993–998. Morgan Kaufmann, 2001. Seattle, Washington.
12. D. McSherry. Diversity-Conscious Retrieval. In Susan Craw, editor, *Proceedings of the Sixth European Conference on Case-Based Reasoning (ECCBR-02)*, pages 219–233. Springer, 2002. Aberdeen, Scotland.
13. J. Reilly and K. McCarthy and L. McGinty and B. Smyth. Dynamic Critiquing. *Submitted to the European Conference on Case-Base Reasoning (ECCBR-04).* To be held in Madrid, Spain.

14. S. Rogers, P. Langley, B. Johnson, and A. Liu. Personalization of the automative information environment. In J. Herrmann R. Engels, B. Evans and F. Verdenius, editors, *Proceedings of the workshop on Machine Learning in the real world; Methodological Aspects and Implications*, pages 28–33, 1997. Nashville, TN.
15. H. Shimazu. ExpertClerk : Navigating Shoppers' Buying Process with the Combination of Asking and Proposing. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 1443–1448. Morgan Kaufmann, 2001. Seattle, Washington, USA.
16. H. Shimazu, A. Shibata, and K. Nihei. ExpertGuide: A conversational case-based reasoning tool for developing mentors in knowledge spaces. *Applied Intelligence*, 14(1):33–48, 2002.
17. B. Smyth and P. Cotter. A Personalized TV Listings Service for the Digital TV Age. *Journal of Knowledge-Based Systems*, 13(2-3):53–59, 2000.
18. B. Smyth and P. McClave. Similarity v's Diversity. In D. Aha and I. Watson, editors, *Proceedings of the International Conference on Case-Based Reasoning*, pages 347–361. Springer, 2001.
19. B. Smyth and L. McGinty. The Power of Suggestion. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03)*. Morgan-Kaufmann, 2003. Acapulco, Mexico.

Hybrid Systems for Personalized Recommendations

Robin Burke

School of Computer Science, Telecommunications and Information Systems, DePaul University, 243 S. Wabash Ave., Chicago, Illinois
rburke@cs.depaul.edu

Abstract. A variety of techniques have been proposed and investigated for delivering personalized recommendations for electronic commerce and other web applications. To improve performance, these methods have sometimes been combined in hybrid recommenders. This chapter surveys the landscape of actual and possible hybrid recommenders, and summarizes experiments that compare a large set of hybrid recommendation designs.

1 Introduction

Recommender systems are personalized information agents that provide recommendations, suggestions for items likely to be of use to a user [17, 25, 26]. They have been applied to many information access problems from news and email filtering to e-commerce product selection.

Recommendation techniques can be distinguished on the basis of their knowledge sources: where does the knowledge needed to make recommendations come from? In some systems, this is the knowledge of other user's ratings of the items in question. In others, it is ontological or inferential knowledge about the domain, added by a human knowledge engineer.

Specifically, recommender systems have *background data* that the system has before the recommendation process begins; *input data* that the user must communicate to the system in order to generate a personalized recommendation; and an *algorithm* that combines background and input data. On this basis, we can distinguish three commonly-used recommendation techniques: collaborative, content-based, and knowledge-based.[1]

1.1 Collaborative Recommendation

Collaborative recommendation is probably the most familiar, most widely implemented and most mature of the technologies. As shown in Figure 1, the collaborative recommender uses a profile from the current user and a profile database

[1] My earlier survey [7] includes two additional techniques: demographic and utility-based. Utility-based recommendation is a special case of knowledge based recommendation. Demographic recommendation is rarely used in a web context because users are generally reluctant to provide the personal data that would make such a technique effective.

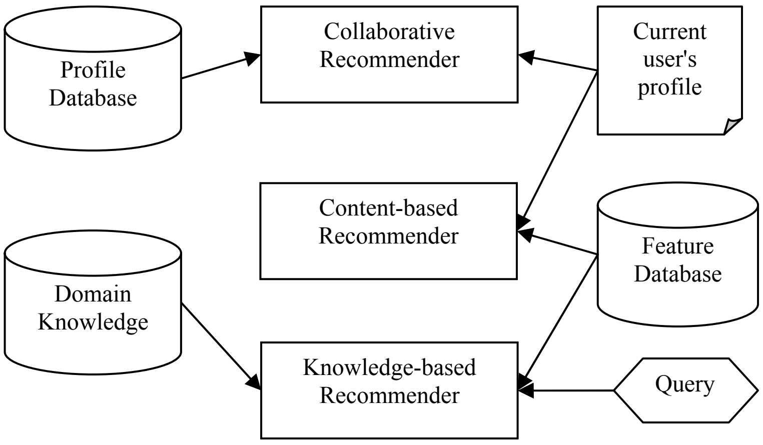

Profile Database
Collaborative Recommender
Current user's profile
Content-based Recommender
Feature Database
Domain Knowledge
Knowledge-based Recommender
Query

Fig. 1. Recommendation techniques and their knowledge sources

of other users. The system recognizes commonalities between the current user and those in the profile database, and generates new recommendations based on inter-user comparisons. A typical user profile in a collaborative system consists of a vector of items and their ratings, continuously augmented as the user interacts with the system over time. Some of the most important systems using this technique are GroupLens/NetPerceptions [25], Ringo/Firefly [30], Tapestry [14] and Recommender [17].

The greatest strength of collaborative techniques is that they are completely independent of any machine-readable representation of the objects being recommended, and work well for objects whose qualities are relatively intangible such as music and movies where variations in taste are responsible for much of the variation in preferences.

1.2 Content-Based Recommendation

Content-based recommendation is an outgrowth and continuation of information filtering research and is based on the idea of recommendation as classification [3]. Figure 1 shows the content-based recommender drawing from the user's profile and also from a feature database. For example, text recommendation systems like the newsgroup filtering system NewsWeeder [20] use the words of their texts as features. A content-based recommender learns a profile of the user’s interests based on the features present in objects the user has rated. The type of user profile derived by a content-based recommender depends on the learning method employed. Decision trees, neural nets, and vector-based representations have all been used. As in the collaborative case, content-based user profiles are long-term models and updated as more evidence about user preferences is gathered.

1.3 Knowledge-Based Recommendation

Knowledge-based recommendation is similar to content-based recommendation in that it draws from the features of the recommended objects, but a knowledge-based recommender also makes use of an explicit query formulated by the user. The knowledge-based recommender uses the query to make recommendations based on inferences about a user's needs and preferences. In some sense, all recommendation techniques could be described as doing some kind of inference. Knowledge-based approaches are distinguished in that they typically have some form of functional knowledge: they have knowledge about how a particular item meets a particular user need, and can therefore reason about the relationship between a need and a possible recommendation. By contrast, a content-based system has only ratings on which to base its conclusions. The Entree system (described below) and several other recent systems (for example, [29]) employ techniques from case-based reasoning for knowledge-based recommendation.

1.4 Strengths and Weaknesses

All recommendation techniques have strengths and weaknesses. Perhaps the best known weakness of the collaborative filtering technique is the "ramp-up" or "cold-start" problem [19]. This term actually refers to two distinct but related problems.

- **New User:** Because recommendations follow from a comparison between the target user and other users based solely on the accumulation of ratings, a user with few ratings becomes difficult to categorize.
- **New Item:** Similarly, a new item that has not had many ratings cannot be easily recommended.

Collaborative recommender systems depend on overlap in ratings across users and have difficulty when the space of ratings is sparse: few users have rated the same items. The sparsity problem is somewhat reduced in model-based approaches, such as singular value decomposition, which can reduce the dimensionality of the space in which comparison takes place [12, 27].

Content-based techniques also have a start-up problem in that they must accumulate enough ratings to build a reliable classifier. Relative to collaborative filtering, content-based techniques also have the problem that they are limited by the features that are explicitly associated with the objects that they recommend. For example, content-based movie recommendation can only be based on written materials about a movie: actors' names, plot summaries, etc. because the movie itself is opaque to the system. This puts these techniques at the mercy of the descriptive data available. Collaborative systems rely only on user ratings and can be used to recommend items without any descriptive data. Even in the presence of descriptive data, some experiments have found that collaborative recommender systems can be more accurate than content-based ones [1].

Knowledge-based recommenders do not have ramp-up or sparsity problems, since they do not base their recommendations on accumulated statistical evidence. Knowledge-based recommender systems are prone to the drawback of all knowledge-based systems: the need for knowledge acquisition. There are three types of knowledge that are involved in such a system: catalog knowledge, knowledge about the objects being recommended and their features; functional knowledge, needed to map between the user's needs and the object that might satisfy those needs; and, user knowledge, knowledge of the user's need and preferences, sufficient to drive the recommendation process.

Despite this knowledge acquisition problem, knowledge-based recommendation has some beneficial characteristics. It does not involve a start-up period during which its suggestions are low quality. While a knowledge-based recommender cannot "discover" user niches, the way collaborative systems can, it can make recommendations as wide-ranging as its knowledge base allows.

These considerations show that no recommendation technique is perfect:

- Collaborative techniques have the unique capacity to identify cross-genre niches and can entice users to jump outside of the familiar. Knowledge-based techniques can do the same but only if such associations have been identified ahead of time by the knowledge engineer.
- Both of the learning-based techniques (collaborative and content-based) suffer from the ramp-up problem in one form or another. The converse of this problem is the stability vs. plasticity problem for such learners. Once one's profile has been established in the system, it may be difficult to change one's preferences.
- The learning-based technologies work best for dedicated users who are willing to invest some time making their preferences known to the system. Knowledge-based systems have fewer problems in this regard because they do not rely on having historical data about a user's preferences.

1.5 Hybrid Recommendation

The desire to avoid the weaknesses of individual recommendation methods has led researchers to consider hybrid recommendation systems, systems which combine techniques of different types with the expectation that the strengths of one will compensate for the weaknesses of another. Researchers have typically evaluated these hybrids by showing their benefits over non-hybridized systems, but there is little work that compares different hybrid designs against each other.

This chapter summarizes the results from a comparative study [8] that examines a large subset of the hybrid recommendation design space using a single data set. The next sections conform to the following outline. First, a taxonomy of recommendation hybrids is outlined and the landscape of possible hybrid designs articulated, based on the survey in [7]. Then, the data set and experimental methodology are described, followed by a summary of results from the large-scale study, and conclusions.

2 Hybrid Recommender Systems

There are seven basic ways that recommender systems can be combined to build hybrids:

- Mixed: Results for different recommenders are presented together either in a combined presentation or in separate lists.
- Weighted: Scores from the recommenders are combined using weights to derive a single score.

Table 1. Two-Component Hybrid Recommendation Designs

	Weighted	Switching	Feature Combination	Cascade	Feature Aug.	Meta-level
CF/CN						
CF/KB			■			
CN/CF	▨	▨	▨			
CN/KB			■			
KB/CF	▨	▨	▨			
KB/CN	▨	▨	▨			

(CF = collaborative, CN = content-based, KB = knowledge-based)

▨	Redundant
■	Not possible

- Switching: The system uses some decision criteria to choose a recommender based on the context and uses the results from only the chosen source.
- Cascade: One recommender refines the recommendations produced by another.
- Feature Combination: Data from different source types are combined together and treated using one recommendation algorithm.
- Feature Augmentation: The output from one technique is used as an input feature to another.
- Meta-level: One recommender produces a model, which is then used as input for the second recommender.

If we consider these possible hybridization strategies and the three recommendation techniques discussed above, we can derive a matrix of possible hybrid recommender designs. Table 1 shows this matrix. For the sake of simplicity, the table omits "Mixed" hybrids, which do not combine evidence from their components and are not really comparable to other hybrids.

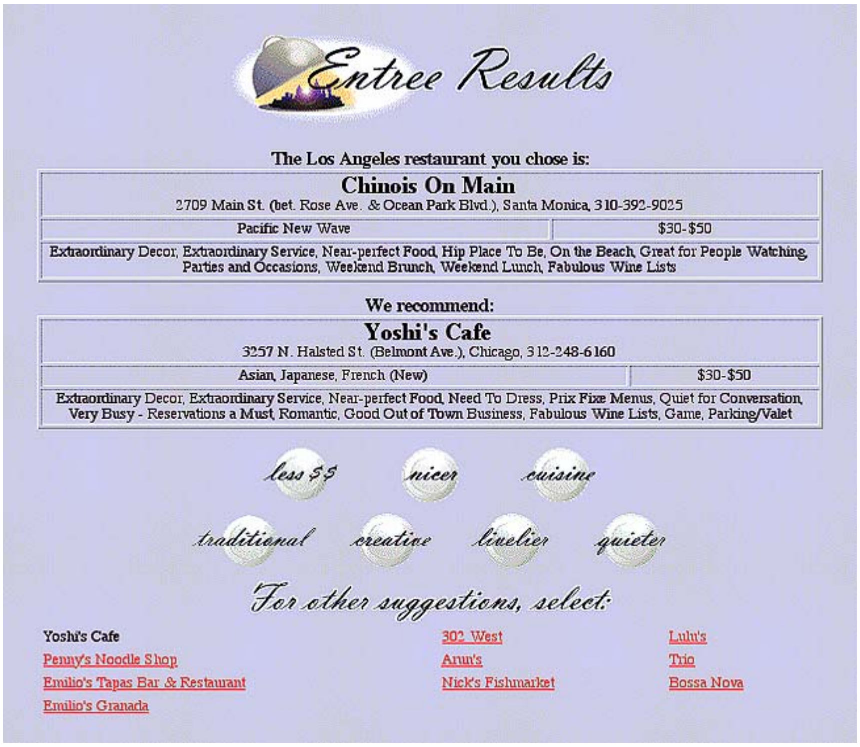
There are three hybridization techniques that are order-insensitive: Weighted, Switching and Feature Combination. With these hybrids, it does not make sense to talk about the order in which the techniques are applied: a CN/CF weighted system would be no different from a CF/CN one. The redundant combinations are marked in gray.

The cascade, augmentation and meta-level hybrids are inherently ordered. For example, a feature augmentation hybrid that used a content-based recommender to contribute features to be used by a second collaborative process, would be quite different from one that used collaboration first. With these techniques all permutations must be considered and these columns do not contain any redundancies.

There are 27 non-redundant spaces in the table, but some combinations are not possible. Since a knowledge-based technique may take into account any kind of data, feature combination does not really represent a possible hybrid. The illogical hybrids are marked in black. The white areas of the table enumerate 25 different possible hybrid recommender systems.

3 Experiments with Hybrid Recommendation

This paper examines some of the results from a large comparative study of hybrid recommender systems [8], looking at a large subset of Table 1 using the same data and recommendation problem. To understand the evaluation methodology employed in this study, we will need to examine the characteristics of the Entree data set. Entree [4, 5] is a knowledge-based restaurant recommendation system that uses case-based reasoning [18] techniques to select and rank restaurants. It was implemented to serve as a guide to attendees of the 1996 Democratic National Convention in Chicago and operated as a web utility for approximately three years. The system is interactive, using a critiquing dialog [9, 31] in which users' preferences are elicited through their reactions to examples that they are shown.



The Los Angeles restaurant you chose is:

Chinois On Main 2709 Main St. (bet. Rose Ave. & Ocean Park Blvd.), Santa Monica, 310-392-9025	
Pacific New Wave	$30-$50
Extraordinary Decor, Extraordinary Service, Near-perfect Food, Hip Place To Be, On the Beach, Great for People Watching, Parties and Occasions, Weekend Brunch, Weekend Lunch, Fabulous Wine Lists	

We recommend:

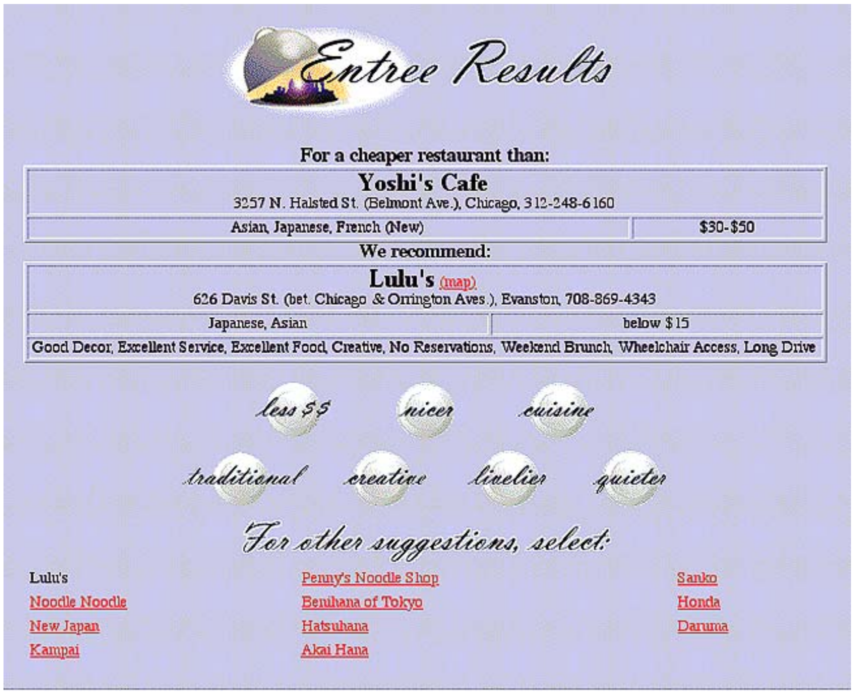
Yoshi's Cafe 3257 N. Halsted St. (Belmont Ave.), Chicago, 312-248-6160	
Asian, Japanese, French (New)	$30-$50
Extraordinary Decor, Extraordinary Service, Near-perfect Food, Need To Dress, Prix Fixe Menus, Quiet for Conversation, Very Busy - Reservations a Must, Romantic, Good Out of Town Business, Fabulous Wine Lists, Game, Parking/Valet	

Yoshi's Cafe	302 West	Lulu's
Penny's Noodle Shop	Arun's	Trio
Emilio's Tapas Bar & Restaurant	Nick's Fishmarket	Bossa Nova
Emilio's Granada		

Fig. 2. Results of a query to the Entree restaurant recommender

Figures 2 and 3 show a user's interaction with the system. An initial query based on a favorite restaurant yields a similar Chicago establishment. When this is deemed too expensive, a click on the "Less $$" critique navigates to a less expensive option.

The data set produced by this interaction is in the form of user histories: $<u, h>$ where each history h is a set of pairs $<r_i, s_i>$ where r_i is a restaurant, and s_i is one of 9 possible responses: entry point, exit point, or one of the seven critiques. This data set has some substantial differences from the standard collaborative-filtering data sets frequently used in the recommendation literature, and is more similar in some respects to the log data used in web personalization [22]. The sessions are short, with only a small percentage containing more than a dozen interactions. User tracking technology was not employed when the system was deployed, but it is possible to heuristically join sessions into long-term user profiles of larger size. However, the task constraints of the restaurant search problem are such that long-term profiles are less likely to be valuable – an intuition borne out by experiment. With such short sessions, we cannot expect to get a large amount of information about a user, and this limits how well a recommender can be expected to perform.



For a cheaper restaurant than:

Yoshi's Cafe
3257 N. Halsted St. (Belmont Ave.), Chicago, 312-248-6160

Asian, Japanese, French (New)	$30-$50

We recommend:

Lulu's (map)
626 Davis St. (bet. Chicago & Orrington Aves.), Evanston, 708-869-4343

Japanese, Asian	below $15

Good Decor, Excellent Service, Excellent Food, Creative, No Reservations, Weekend Brunch, Wheelchair Access, Long Drive

Lulu's	Penny's Noodle Shop	Sanko
Noodle Noodle	Benihana of Tokyo	Honda
New Japan	Hatsuhana	Daruma
Kampai	Akai Hana	

Fig. 3. Results of the "Less $$" critique

Explicit rating data and standard web mining data such as dwell time are not available. However, we do have the evidence of the user's critiques. The critiques can be interpreted as negative ratings; they result in the user moving away from the suggestion that is shown. There are few actions that can be interpreted as positive ratings. The system allows a user to input a favorite restaurant as a starting point for recommendation, which can definitely be considered positive. However, this only occurs in about 10% of the sessions. End points may constitute either successful recommendations (which should be positive ratings) or abandoned sessions (noise). To determine the effects of this noise, we experimented using only the definite positive (starting point) ratings in the subset of the data in which these ratings are available and compared these results with the experiments treating end points also as positive ratings.

Because this data is much small than the full one, the algorithms are less accurate, but the correlation between the two conditions was extremely high (0.92). In the experiments described here, both start and end points are used as positive ratings, with the understanding that there is some noise in the positive rating data.

Despite its problems, the Entree data has the unique advantage in that a working knowledge-based recommender exists for the restaurant domain (the Entree recommender itself), the hard work of knowledge acquisition having already been performed. This means that it is possible to compare the designs shown in Table 1. Without a knowledge-based component, only nine of these designs can be evaluated. In this article, we will not discuss the six possible meta-level hybrids, which are somewhat more complex in design and for which experimental results are still incomplete. See [8] for additional details.

3.1 Evaluation Metrics

Herlocker and colleagues have studied a variety of evaluation techniques for collaborative filtering systems [16]. Different criteria may be important in different contexts. In the restaurant recommendation domain, "Find Good Recommendations" is the appropriate task context. This work identifies three basic classes of evaluation measures: discriminability measures (such as ROC-derived measures) , precision measures (such as mean absolute error) and holistic measures (that work best when all user ratings and system predictions are pooled and evaluated as a group). In each of these groups, a wide variety of different metrics were found to be highly correlated, effectively measuring the same property. For the restaurant recommendation task, we are interested in a precision-type measure, and Herlocker's results tell us that we need not be extremely picky about how such a measure is calculated.

With short sessions and a dearth of positive ratings, there are some obvious constraints on how the Entree sessions can be employed and recommendations evaluated. An evaluation technique that requires making many predictions for a given user will not be applicable, because there would not be enough of a profile left for a collaborative system to use. This rules out such standard metrics as precision/recall and mean absolute error. Ultimately, in order to find good recommendations, the system must be able to prefer an item that the user rated highly. How well the system can do this is a good indicator of its success in prediction, so our evaluation will concentrate on those items the user likes. What we do is to record the rank of a positively-rated test item in the recommendation set returned by a given recommender. Averaging over many trials we can compute the "average rank of the correct recommendation" or ARC. The ARC measure provides a single value for comparing the performance of the hybrids, focusing on how well each can discriminate the item liked by the user from the others.

The evaluation of the recommenders proceeded as follows. The set of sessions was divided randomly into training and test parts of approximately equal size. This partition was performed five times and results from each test/training split averaged. Each algorithm was given the training part of the data as its input, each handling this data in its own way, and in some cases, such as with the knowledge-based recommender, it was ignored. Evaluation was performed on each session of the test data, simulating the interaction of the system with a single user. From the session, a single item with a positive rating was chosen.[2] This item was the test item for which the ARC value was calculated. All of the other ratings were considered part of the user profile.

The recommendation algorithm was then given the user profile without the positively-rated test item, and made its recommendations. The result of the recommendation process was a ranked subset of the product database containing those items possibly of interest to the user. From this ranked list, the rank of the test item was recorded.

There are two variables that can be manipulated with respect to session size. One is the number of ratings the user has provided; the other is whether the profile consists of a single session or multiple interactions. There are only a small percentage of single sessions of size 15 or greater, so for long profiles, we must turn to multi-session ones. To examine how the hybrids fared on both types of profiles, we examine each system's performance with single sessions of size 5, 10 and 15 and multi-session profiles of 10, 20 and 30. These conditions are indicated as "5S", "10S", "15S" and "10M", "20M", "30M", respectively.

The content-based and collaborative components fit well into this evaluation paradigm. They are designed to accept a profile as input and produce a recommendation. A knowledge-based component is different. The Entree recommender needs a query in order to produce output, and in order to use such a component we must decide where its queries will come from. One possibility would be to use the features of all of the restaurants that appear in the profile and derive a composite representation that would serve as a query. This did not work well in practice, no doubt because the Entree interface encourages users to explore, and cumulative profiles contain many digressions. A better alternative is to pick the last item in the profile and use it as the query. This item represents in some sense the user's progress toward the final recommendation destination, and perhaps the convergence of their preferences. The last item in the profile is not necessarily the critique immediately prior to the end point, since we evaluate profiles at fixed sizes as discussed above and extra (negative) ratings are truncated.

4 Results

To provide a starting point for analysis of the hybrids, we can examine the four basic algorithms, including the performance of the "average" recommender, which recommends restaurants based on their average rating from all users, and does not take the user profile into account. The three basic algorithms are CF, a correlation-based collaborative algorithm using Pearson's r [15]; CN, a content-based recommender implemented using the Naive Bayes technique [13]; and KB, the Entree recommender system used as a knowledge-based component.

Figure 4 shows the average rank of the correct recommendation (ARC) for each of the basic algorithms over the six different session size conditions. We should note that this recommendation task is actually rather difficult, especially for the multi-session profiles. The best any of these basic algorithms can manage is average rank of

[2] If there are no positive ratings, the session is discarded. We cannot evaluate the quality of recommendation if we have no information about what the user prefers.

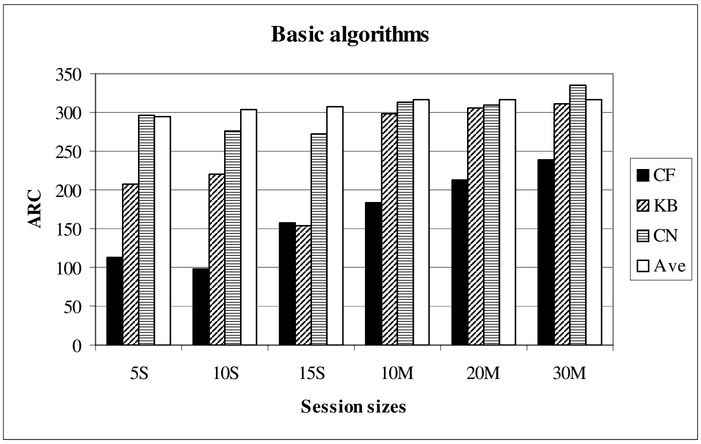


Fig. 4. Average rank results for the basic recommenders

approximately 100 for the correct answer, not inspiring in an e-commerce context where the user might be expected only to look at the first dozen results or so. The techniques vary widely in their performance on the Entree data. On the multi-visit profiles, all of the algorithms do worse, with the knowledge-based technique in particular falling back considerably from its single-session performance.[3]

4.1 Weighted

Perhaps the simplest design for a hybrid system is a weighted one. Each component of the hybrid scores a given item and the scores are combined using a linear formula [11]. Entree returns integer scores,[4] which are normalized to the range 0..1. To derive the optimum weighting, we examine all possible weightings (in discrete increments) and determine which weighting, over the training data, would yield the best ARC value.

Three weighted combinations of the algorithms were possible: CF/CN, CF/KB, CN/KB. The results for this hybrid were rather surprising. Figure 5 shows the average rank results. In all but five of the 18 conditions, the performance of the combined recommenders was worse than the stronger recommender alone. In two of the conditions, the weighted hybrid was either the same or worse than the weakest recommender of the hybrid pair. Only in five conditions is the weighted hybrid superior to its components separately, with the remaining conditions showing performance in-between the two components.

[3] The content-based algorithm may suffer due to the skewed nature of the profiles: with many negative ratings and few positive ones, but recall that the ARC metric measures where a recommendation is placed relative to others in the list and all items are effected by the negative baseline. More likely the paucity of data is responsible for the poor performance of the content-based technique. Because the evaluation paradigm for the knowledge-based technique assumes consistency in preferences, it is particularly affected in the multi-session case.

[4] A peculiarity dictated by the system's similarity metrics [4].

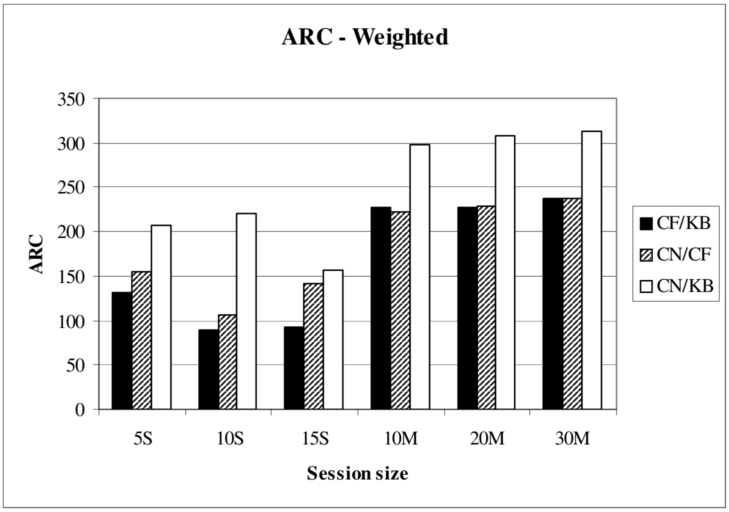


Fig. 5. ARC results for weighted hybrids

There are several reasons why this result might occur. One is that the recommendation components are not sufficiently independent for their combination to be different that any one alone. However, we know this is not the case since the measures use very different knowledge sources and as we will see below, other types of hybrids using these components are successful. More likely, the problem is that the recommenders, especially KB and CF, do not have uniform performance across the product and user space and the underlying assumption behind a linear weighting scheme is at fault. This suggests the application of the next type of hybrid, one in which the hybrid switches between its components depending on the context.

4.2 Switching

A switching hybrid is one that chooses a single recommender from among its constituents in each recommendation situation. In order to implement such a hybrid, there must be some criterion available to enable the switching decision. We can think of this value as a "confidence" value.

Ideally, we would survey the confidence values computed by each algorithm and choose the most confident. However, this would assume comparability between confidence values computed in different ways, and experiments showed this was not a valid assumption. An alternative is to select one component of the hybrid as the primary recommender and let it determine the confidence in its own prediction. If the primary recommender has confidence above some threshold, its recommendation will be used; otherwise, the secondary recommender takes over.[5] This distinction between

[5] There are other possibilities for implementing a switching hybrid. The system might look at the difference between confidence values or their ratio, for example. An alternative is to have a third metric outside of the hybrids as the switching criterion. Mobasher and Nakagawa [23] describe a system in which a metric of site connectivity is used to determine which of two usage mining techniques to employ.

primary and secondary recommenders makes the switching hybrid an order-sensitive system. Like the weighted hybrid, all possible switching thresholds are tried and the threshold that maximizes ARC performance on the training set is used.

Each of the recommenders in the experimental set required a different confidence calculation, normalized to the range [0-1]. For the collaborative algorithms, the confidence value is computed using the inverse of the average distance between these peer profiles and the user, since the closer the peers to the user, the more likely it should be that they are good predictors. For the naive Bayes algorithm, the choice of confidence metric is fairly straightforward – the value returned by the naive Bayes classifier is supposed to represent the probability that the classified object is a member of the given class. For the knowledge-based algorithm, the confidence is computed by finding the overlap in features between the top recommendation and the query. The intuition here is that if the knowledge-based system did not have to go far afield (and thereby making many inferences) to make its retrieval, then the results returned will be more confident.

To be a good primary recommender in this switching paradigm, an algorithm must have a reliable assessment of its own accuracy. Otherwise, it will turn over control to the secondary recommender when its own results might be more correct and make recommendations when the secondary one might be better. In some cases, when a range of confidence thresholds were attempted, the best accuracy was achieved when the primary algorithm had a threshold equal to 1.0, meaning that the recommender would have to compute a confidence > 1 in order for its recommendations to be used. This is an impossibility, and so in these degenerate cases, the primary recommender is essentially ignored and the recommender becomes a non-hybrid made up only of the secondary component. This happens when the primary recommender is particularly weak.

For the sake of brevity, Figure 6 and the rest of the graphs only show those hybrids that achieve synergy: that is, the hybrid together performs better than either of its

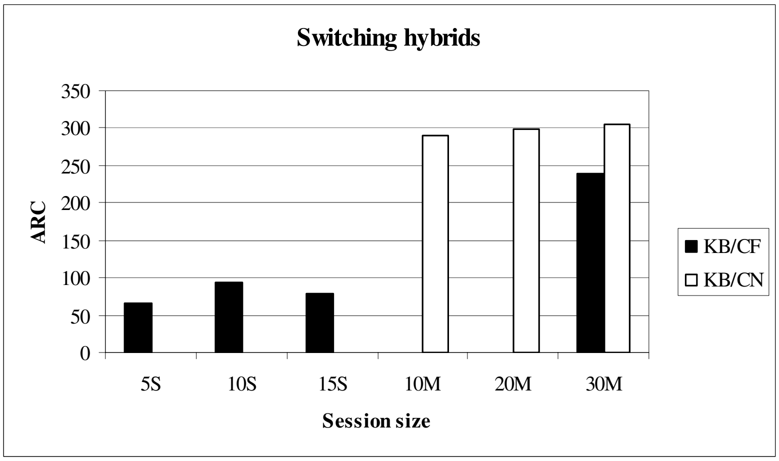


Fig. 6. ARC results for switching hybrids

components on their own. Also, for switching hybrids, the degenerate (threshold = 1.0) cases have been omitted, so this chart shows only a subset of the systems tested. Only in the KB/CF case do we see true synergy: the CF algorithm achieved an ARC around 150 in the 15S condition, but the KB/CF hybrid is close to half that value in the same condition.

The relatively poor performance of these hybrids indicates the difficulties presented in accurately computing the confidence that should be associated with an algorithm's prediction. See [10] for a discussion of the difficulties of calculating confidence in a prediction context.

4.3 Cascade Hybrids

The idea of a cascade hybrid is to create a strictly hierarchical hybrid, one in which a weak recommender cannot overturn decisions made by a stronger one, but can merely refine them. A cascade recommender uses a secondary recommender only to break ties in the scoring of the primary one.

Many recommendation techniques have real-valued outputs and so the probability of identical scores is small. This would give the secondary recommender in a cascade little to do. In fact, the literature did not reveal any other instances of the cascade type at the time that the original hybrid recommendation survey [7] was completed. However, the cascade hybrid raises the issue of the appropriate confidence interval for the score returned by recommendation algorithms, presumably much less than the full 32 bits in the computational representation. And, if the scoring of our algorithms is somewhat less precise, then there may be space in which a cascade can operate. Figure 7 shows the ARC graph comparing the CF and CN recommenders with full 32-bit precision against the same algorithms truncated to two digits of precision (the LP versions). We see that overall the differences are very small and not always in the

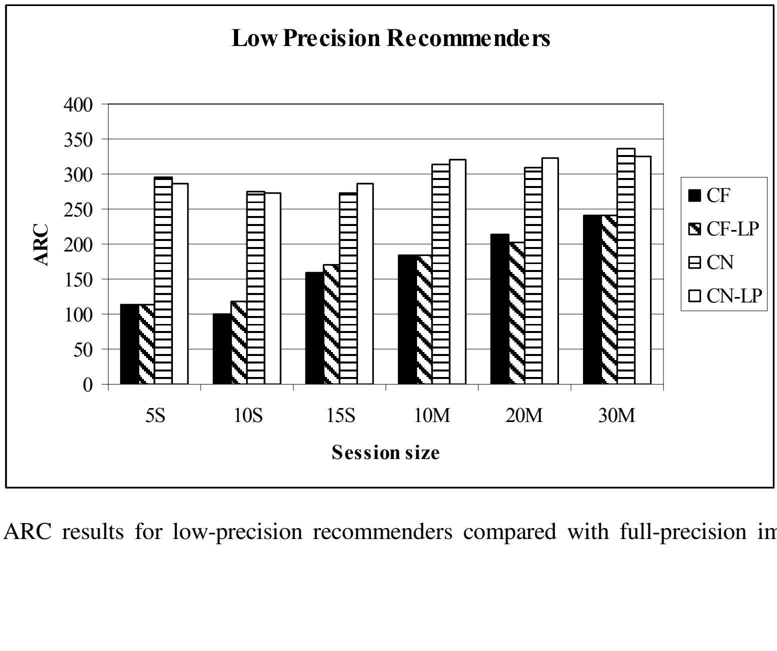


Fig. 7. ARC results for low-precision recommenders compared with full-precision implementations

favor of the higher-precision algorithm. Our cascade hybrids therefore use these low-precision versions of the algorithms, generating ties that a secondary recommender can break.

With this result in mind, we can turn to the cascade recommenders. Figure 8 shows the ARC results for these hybrids. Again, only those recommenders demonstrating synergy are shown. The successful cascade combinations are very good. (Note the change in scale in the y-axis.) Most significant is the behavior on the multi-profile sessions, for which most systems examined so far have been inadequate.

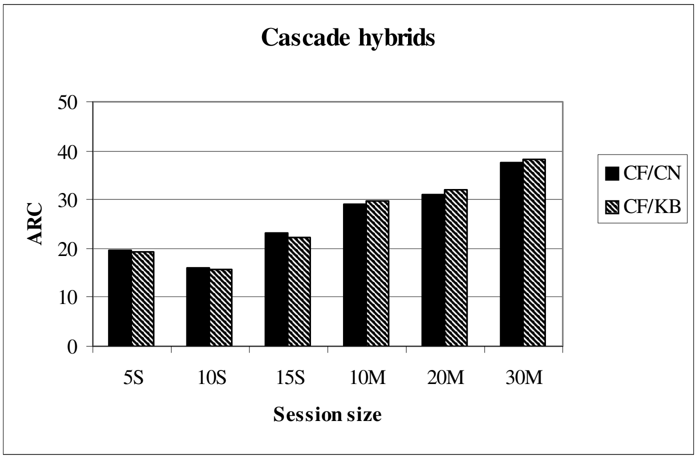


Fig. 8. ARC results for cascade recommenders

4.4 Feature Combination Hybrids

The idea of feature combination is to inject features associated with one recommendation type (such as collaborative recommendation) into an algorithm designed to process data with a different source (such a content-based recommendation). This is a way to expand the capabilities of a well-understood and well-tuned system, by adding new kinds of features into the mix [2, 24].

The content-based recommender with a contributing collaborative part (CF/CN) was built by augmenting the representation of each restaurant with new features corresponding to the reaction of each profile in the training data to that restaurant. For example, if profiles A and B had negative ratings for restaurant X and profile C had a positive rating, the representation of restaurant X would be augmented with three new features, which can be thought of as A-, B- and C+. Now an ordinary classification algorithm can be employed using the test user's profile to learn a model of user interests, but this model will now take into account similarities between restaurants that have a collaborative origin.

A feature combination hybrid uses only one recommendation algorithm, but this should not disqualify it from consideration as a hybrid. The original taxonomy presented above and illustrated in Figure 1 defines recommendation techniques based on their knowledge sources: what it is that they need to know in order to generate recommendations. This is a most important consideration for system design and implementation, because it determines what data the system must be able to store and track and what outside data must be supplied to it. In this definition, a content-based recommender is one that is aware of one user's profile and the product database. A collaborative hybrid in which the entire profile database is also brought to bear is a very different type of system, regardless of the algorithm by which it is achieved.

The naive Bayes implementation performed poorly with this augmented model, including as it does over 5,000 new collaborative features in addition to the 256 content ones. So, for this hybrid only, I examined the Winnow algorithm [21], another classification learning algorithm that scales better to large numbers of features. (Naives Bayes outperformed Winnow in the non-hybridized case.)

A collaborative recommender with a contributing content-based component turns this process around and creates artificial profiles corresponding to particular content features; these are sometimes called "pseudo-users" [28]. For example, all of the restaurants with Tex-Mex cuisine would be brought together and a profile created in which the pseudo-user likes all of the Tex-Mex restaurants in the database. Similar profiles are generated for all the other content features.

Other possibilities for feature combination hybrids turn out to be either illogical or infeasible. The features that the knowledge-based recommender uses are the same as those used by the content-based recommender; they are just used in a different way. A feature combination hybrid with a knowledge-based contributing part would therefore be no different from the content-using one described above. A knowledge-based hybrid with a collaborative contributing recommender would be theoretically possible: a knowledge engineer could write rules that make inferences about the preferences of the users in our test set, but such an enterprise would be wholly impractical in any fielded application and would run counter to the intent that as little as possible extra work would be done on the base recommenders in order to implement the hybrids.

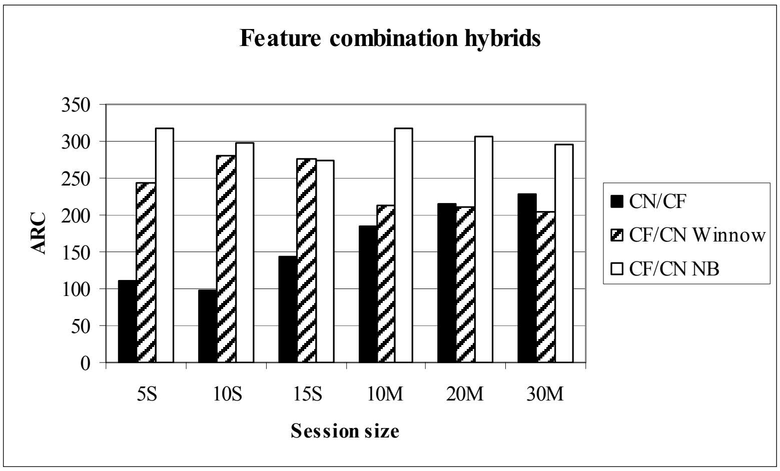


Fig. 9. ARC results for feature combination hybrids

Figure 9 shows the average rank data for these hybrids. The CN/CF hybrid is nearly identical to the unhybridized CF algorithm. The CF/CN hybrid with Winnow shows modest success for the multi-session profiles.

4.5 Feature Augmentation Hybrids

Feature augmentation is a strategy for hybrid recommendation in which a contributing recommender generates a new feature or set of features for each item, augmenting the data for the primary recommender with its own contribution. The augmentation can usually be done off-line, making this approach attractive when trying to strengthen an existing recommendation algorithm by adjusting its input.

The integration of components in a feature augmentation arrangement is somewhat trickier than in the hybrids seen so far. The contributing recommender must actually modify the input of the primary recommender and this is different than merely producing a score. So, the KB component must produce features associated with items that the other components can use, in order to be a contributing component, and it must reason with the features produced by the other components in order to be a primary recommender.

The feature generation technique used in these experiments is clustering. To use clustering techniques, we can envision the set of user profiles as a set of sparse vectors. Each user is a row and each restaurant a dimension. Each user will only have rated a few restaurants, which is what makes each vector sparse. Clustering these vectors will group similar users together. An alternate representation would be one in which there is a vector for each restaurant with the users being the dimensions. Clustering the restaurant vectors yields information on what restaurant are similar to others, based on their patterns of preference across users. The result of a clustering computation is a cluster id, which is effectively a collaboratively-derived feature that can be associated with each restaurant. Restaurants that share a cluster will have the same id, and this can be part of the input to the CN component, which is expecting input consisting of restaurants and their features.

When the KB component is the primary recommender, we use these ids in a very simple manner: an additional similarity metric is added to the recommender that prefers restaurants in the same cluster. This does require modification of the Entree recommender, but it is the minimal amount required to make use of the new features.

When the KB component is the contributing recommender, there is no underlying data that can be used as features for this type of hybrid. If we want to build a KB/CN feature augmentation query, we need the KB component to contribute restaurant features derived from its similarity knowledge. Sticking with the idea of clustering, we perform the following operation. For each restaurant, use the KB component to compute the most similar restaurant. Build a "user profile" for each restaurant, where each similar restaurant is given a positive rating. These profiles can then be input to the clustering algorithm described above, yielding a cluster id based on the system's similarity knowledge, which are then added to the features used by the CN component.

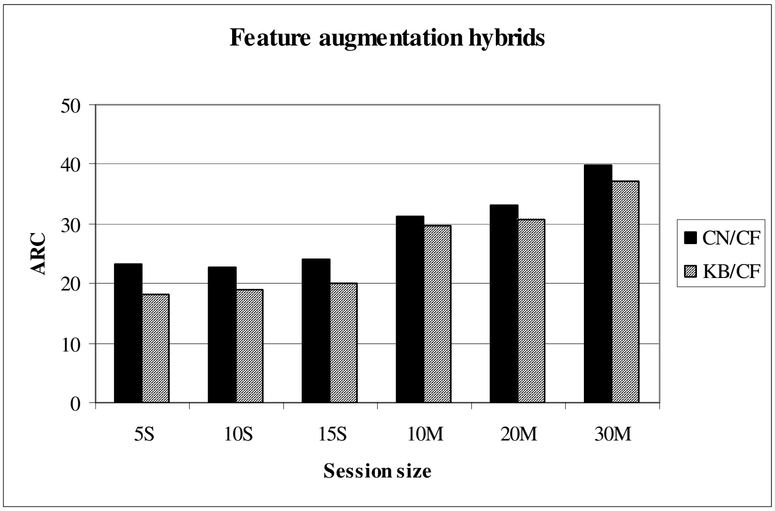


Fig. 10. ARC results for feature augmentation hybrids

The KB/CF feature augmentation hybrid is slightly different. The features that the CF component "understands" are profiles, lists of restaurants/rating pairs. A KB/CF hybrid can be achieved through pseudo-users created through retrieval. We perform retrieval based on the user profile and then create a pseudo-user, who gives positive ratings to all the restaurants returned by the query and negative ratings to the others.

The results for these recommenders shown in Figure 10 are impressive for the both of the cases shown. While other combinations did not achieve synergy, the successful implementations were those in which the collaborative component was the one receiving augmented data. Here we see very strong performance, especially the KB/CF recommender which has an ARC below 40 throughout all of the profile types. This follows the pattern seen in the cascade results, where a combination of a collaborative and a knowledge-based component produced the best results.

5 Conclusion

The experiments outlined in this paper cover a large portion of the space of possible two-component hybrid recommender systems based on three basic recommendation algorithms: content-based, collaborative, and knowledge-based and five types of combinations: weighted, switching, cascade, feature combination, and feature augmentation. The work surveyed 19 different systems in all, including 10 designs without previous extant examples.

Of course, any such study is by its nature limited by the peculiarities of the available data and the recommendation domain. Of course, any such study is by its nature limited by the peculiarities of the available data and the recommendation domain. The Entree data set is relatively small (just over ¼ million ratings), the profiles are short and the ratings are implicit and heavily skewed to the negative. It would be valuable to repeat this study in a different recommendation domain with different products and a set of user profiles with different characteristics. The most substantial hurdle to such a study is the availability of a knowledge-based recommendation component.

Three results, however, can be seen, which may have general applicability and are worthy of further study. First is the utility of a knowledge-based recommender itself. Such recommenders have been explored as a knowledge-based solution to the problem of product search, and have been deployed in a number of e-commerce applications [6, 27]. The experiments presented here show that a knowledge-based recommendation engine may also be combined in numerous ways to build hybrids and in fact, some of the best performing recommenders seen in these experiments were created by using a knowledge-based component.

In examining the hybrids themselves, we see that cascade recommendation, although rare in the hybrid recommendation literature, turns out to be a very effective means of combining recommenders of differing strengths. Adopting this approach requires treating the scores from a primary recommender as rough approximations, and allowing a secondary recommender to fine-tune the results. In addition, feature augmentation also is shown to be highly effective, and this technique has the added efficiency that the contributing recommender can produce its augmenting features off-line.

Acknowledgements

Entree was developed at the University of Chicago in collaboration with Kristian Hammond, with the support of the Office of Naval Research under grant F49620-88-D-0058. Parts of the experimental design were done with the assistance of Dan Billsus at the University of California, Irvine. Thanks to Michael Pazzani and Bomshad Mobasher for helpful comments on an early draft of this paper.

References

[1] Alspector, J., Koicz, A., and Karunanithi, N.: 1997, 'Feature-based and Clique-based User Models for Movie Selection: A Comparative Study'. *User Modeling and User-Adapted Interaction* 7, 279-304.

[2] Basu, C., Hirsh, H. and Cohen W.: 1998, 'Recommendation as Classification: Using Social and Content-Based Information in Recommendation'. In: *Proceedings of the 15th National Conference on Artificial Intelligence*, Madison, WI, pp. 714-720.

[3] Belkin, N. J. and Croft, W. B.: 1992, 'Information Filtering and Information Retrieval: Two Sides of the Same Coin?' *Communications of the ACM* 35(12), 29-38.

[4] Burke, R.: 1999, 'The Wasabi Personal Shopper: A Case-Based Recommender System'. In: *Proceedings of the 11th National Conference on Innovative Applications of Artificial Intelligence*, pp. 844-849.

[5] Burke, R.: 2000, 'Knowledge-based Recommender Systems'. In: A. Kent (ed.): *Encyclopedia of Library and Information Systems*. Vol. 69, Supplement 32.

[6] Burke, R.: 2001, *Case-Based Reasoning in Electronic Commerce*. Workshop held at ICCBR 2001, Vancouver, Canada. Proceedings available at http://www.aic.nrl.navy.mil/-papers/2001/AIC-01-003/ws3/ws3.htm

[7] Burke, R.: 2002, 'Hybrid Recommender Systems: Survey and Experiments'. *User Modeling and User Adapted Interaction*, 12 (4), 331-370.

[8] Burke, R.: in preparation, 'Hybrid Recommender Systems: Comparative Studies'.

[9] Burke, R., Hammond, K., and Young, B.: 1997, ‘The FindMe Approach to Assisted Browsing’. *IEEE Expert*, 12 (4), 32-40.

[10] Cheetham, W and Price, J.: 2004, 'Measures of Solution Accuracy in Case-Based Reasoning Systems'. In: P. Funk and P.A. Gonzalez-Calero (eds.) *Advances in Case-Based Reasoning (ECCBR 2004)*, pp. 106-118.

[11] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M.: 1999, ‘Combining Content-Based and Collaborative Filters in an Online Newspaper’. *SIGIR ’99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA. <URL: http://www.cs.umbc.edu/~ian/sigir99-rec/papers/claypool_m.ps.gz>

[12] Foltz, P. W.: 1990, ‘Using Latent Semantic Indexing for Information Filtering’. In: R. B. Allen (ed.): *Proceedings of the Conference on Office Information Systems*, Cambridge, MA, pp. 40-47.

[13] Friedman, N., Gieger, M., and Goldszmidt, M.: 1997, 'Bayesian Network Classifiers'. *Machine Learning*, 29, 131-163.

[14] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D.: 1992, 'Using collaborative filtering to weave an information tapestry'. *Communications of the ACM*. 35 (12), 61-70;

[15] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. T.: 1999, 'An Algorithmic Framework for Performing Collaborative Filtering'. In *ACM SIGIR 1999*, pp. 230-237.

[16] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T.: 2004, 'Evaluating Collaborative Filtering Recommender Systems'. *ACM Transactions on Information Systems*. 22 (1).

[17] Hill, W., Stead, L., Rosenstein, M. and Furnas, G.: 1995, ‘Recommending and evaluating choices in a virtual community of use’. In: *CHI ’95: Conference Proceedings on Human Factors in Computing Systems*, Denver, CO, pp. 194-201.

[18] Kolodner, J.: 1993, *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann.

[19] Konstan, J. A., Riedl, J., Borchers, A. and Herlocker, J. L.: 1998, ‘Recommender Systems: A GroupLens Perspective.’ In: *Recommender Systems: Papers from the 1998 Workshop (AAAI Technical Report WS-98-08)*. Menlo Park, CA: AAAI Press, pp. 60-64

[20] Lang, K.: 1995, ‘Newsweeder: Learning to filter news’. In: *Proceedings of the 12th International Conference on Machine Learning*, Lake Tahoe, CA, pp. 331-339.

[21] Littlestone, N. and Warmuth, M.: 1994, ‘The Weighted Majority Algorithm’. *Information and Computation* 108 (2), 212-261.

[22] Mobasher, B.: 2004, 'Web usage mining and personalization'. In *Practical Handbook of Internet Computing*, Munindar P. Singh (editor), Chapman Hall & CRC Press.

[23] Mobasher, B. and Nakagawa, M.: 2003, ' A Hybrid Web Personalization Model Based on Site Connectivity'. In *Proceedings of the WebKDD Workshop at the ACM SIGKKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, August 2003.

[24] Mooney, R. J. and Roy, L.: 1999, ‘Content-Based Book Recommending Using Learning for Text Categorization’. *SIGIR ’99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA. <URL: http://www.cs.umbc.edu/~ian/sigir99-rec/papers/mooney_r.ps.gz>

[25] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.: 1994, ‘GroupLens: An Open Architecture for Collaborative Filtering of Netnews’. In: *Proceedings of the Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, pp. 175-186.

[26] Resnick, P. and Varian, H. R.: 1997, ‘Recommender Systems’. *Communications of the ACM*, 40 (3), 56-58.

[27] Rosenstein, M. and Lochbaum, C.: 2000, ‘Recommending from Content: Preliminary Results from an E-Commerce Experiment.’ In: *Proceedings of CHI'00: Conference on Human Factors in Computing*, The Hague, Netherlands.

[28] Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, J. Miller, B. and Riedl, J.: 1998, ‘Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System’. In: *Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work*, Seattle, WA, pp. 345-354.

[29] Schmitt, S. and Bergmann, R.: 1999, ‘Applying case-based reasoning technology for product selection and customization in electronic commerce environments.’ *12th Bled Electronic Commerce Conference*. Bled, Slovenia, June 7-9, 1999.

[30] Shardanand, U. and Maes, P.: 1995, ‘Social Information Filtering: Algorithms for Automating "Word of Mouth"’. In: *CHI ’95: Conference Proceedings on Human Factors in Computing Systems*, Denver, CO, pp. 210-217.

[31] Shimazu, H.: 2001, 'ExpertClerk: Navigating Shoppers' Buying Process with the Combination of Asking and Proposing'. In B. Nebel, (ed.) *Proceedings of the Seventeenth International Joint Conference on Artification Intelligence*, pp. 1443-1448.

Collaborative Filtering Using Associative Neural Memory

Chuck P. Lam

Lama Solutions LLC, 408 Carl Street, San Francisco, CA 94117
chuck.lam@lama-solutions.com

Abstract. This paper introduces a collaborative filtering (CF) neural-network algorithm for recommending items. This algorithm connects the study of collaborative filtering with the study of associative memory, which is a neural network architecture that is significantly different from the dominant feedforward design. There are two types of CF systems – user-based and item-based, and we show that our CF system can have both interpretations. We further prove that, given a random subset of all users, our CF system is an unbiased estimator of predictions made from all users, thus theoretically justifying random sampling. We further apply standard neural network techniques, such as magnitude pruning and principle component analysis, to improve the system's scalability. Results from experiments with the MovieLens dataset are shown.

1 Introduction

With hundreds of thousands of products available for sale, today's retailers (including e-tailers, mail-order companies, and brick-and-mortar channels) are looking for ways to help their customers find the desired products more easily. For example, electronic retailers often help customers evaluate the desirability of a product by providing reviews, testimonials, and numerical ratings from other real customers. Powerful search engines can help customers locate products with specific descriptions. For some commerce categories, such as travel reservations, cars, and dating, search engines can also be optimized to the unique aspects of each category. For example, search engine for travel reservation can suggest near-by airports that may have cheaper flights.

Computational intelligence can play many additional roles in helping shoppers find desired products. One of which is to learn the preference and purchasing behavior of previous shoppers, and using that knowledge to make personalized recommendations to the current, active shopper. Such applications are generally known as collaborative filtering (CF).

Many techniques have been tried to create collaborative filtering systems. The algorithm introduced in this paper is novel in applying *associative memory* to collaborative filtering. Like feedforward neural networks, associative memory was inspired by neuropsychological principles and had been studied extensively. This paper will demonstrate that many of the theories and techniques well-known to researchers of associative memory can now be fruitfully applied to collaborative filtering.

2 Background

2.1 Collaborative Filtering

To the user, recommendations can be given in several ways depending on the specific sales scenario. One way is *prediction*, in which the system tries to predict a user's preference on specific products. Another way is *ranking*. Here the system tries to predict the user's ranking of products and recommends the top N products to the user. Many of the initial works in collaborative filtering tend to focus on prediction [1,2]. However, ranking is more used in commercial settings, and much of recent work, including ours, focuses on ranking.

There are two major approaches to collaborative filtering: user-based (also called memory-based) and model-based (also called item-based). In user-based collaborative filtering, the similarities between the current shopper and all previous shoppers are measured. Those similarity measurements are used in calculating a weighted sum (or average) of other users' preferences on items the active user has not seen. Sometimes only a subset of close neighbors is used in the weighted sum. Conceptually user-based systems are similar to weighted nearest-neighbor algorithms.

In model-based systems, some model is built to represent the relationships between different products. The active user's preference on some products is used to predict her preference on other products. Various models, such as Bayes nets [3], association rules [4], and item-based systems [5,6], have been applied and reported in the literature. Our use of associative memory will first be explained as an item-based system. However, we will later demonstrate its user-based interpretation.

2.2 Notations

In collaborative filtering, one is given the preference ratings of a particular user, called the *active user*, on some items. The prediction task attempts to predict what the active user's rating will be on items she has not yet rated. The ranking task is to recommend a list of N new items to her. The goal is to have as many highly desired (to the active user) items as possible in those N recommendations. A database of other users' preference ratings is available. The assumption is that the active user's preference on new items will be close to that of other users who have had similar preferences in the past, and recommendation is computed by formalizing that assumption.

More specifically, we have I, a set of items that can be rated. We also have r_a, a vector of ratings from the active user. The jth element $r_{a,j}$ denotes the active user's rating on item j and is denoted $\perp$ if she has not rated the item. Similarly, there exists R, a matrix of ratings in which the element $r_{i,j}$ denotes user i's rating on item j. Furthermore, define I_i as the set of items for which user i has rated. The average rating for user i is

$$\overline{r}_i = \frac{1}{|I_i|} \sum_{j \in I_i} r_{i,j}$$

and the average rating for the active user, $\overline{r}_a$, is defined similarly.

A typical user-based collaborative filtering algorithm [3] for *predicting* the active user's rating on item j is

$$p_j^a = \overline{r}_a + \kappa_j \sum_{i:r_{i,j} \neq \perp} w(a,i)(r_{i,j} - \overline{r}_i) \tag{1}$$

where $w(a,i)$ is the weight of user i in predicting the active user's preference, and κ_j is a normalizing constant such that the absolute values of the weights used in the summation add up to one. That is,

$$\kappa_j = \frac{1}{\sum_{i:r_{i,j} \neq \perp} |w(a,i)|}$$

Two of the most widely used weighting schemes for $w(a,i)$ are Pearson correlation coefficient and vector similarity [3,2,7]. One variation on Equation 1 is to limit the summation to the k users with the highest weights $w(a,i)$. In this paper we will have no such limitation. That is, k can be the size of the entire training set.

Equation 1 is designed originally for the *prediction* task, but its output p_j^a can easily be applied to the ranking task. However, a couple modifications can be made to simplify the equation for ranking. The obvious change is to remove the term $\overline{r}_a$. It has no effect on the ranking of one item above another for the active user. A more subtle change is to remove the normalization effect of κ_j by setting it to one. The original role of this normalization is to shrink the summation down within the range of rating values, a reasonable and necessary process for the task of prediction. For the ranking task, however, we are no longer concerned with having the predicted value in some limited range. Furthermore, qualitatively speaking, the general range of the summation is affected by two factors: the distribution of $w(a,i)$ (a "main-stream" active user can have high weights with many other users) and the popularity of item j (there will be more terms in the summation if many people have rated item j.). Normalization removes those factors from consideration. Yet for ranking we would like to take popularity into account. That is, we want to recommend items that *more people* have expressed higher preferences for. Given those modifications, the user-based *ranking* algorithm becomes choosing the N highest scores of[1]

$$p_j^a = \sum_{i:r_{i,j} \neq \perp} w(a,i)(r_{i,j} - \overline{r}_i) \tag{2}$$

Another approach to collaborative filtering is the item-based system [6,5]. We refer the reader to Karypis [6] for a detailed description of such systems. The general idea is to first build a database of similar items. When an active user has expressed preference for a set of items, the item-based algorithm selects other items similar to the ones in that set and recommends the more popular items in that selection. As will be seen later, linear associative memory performs the task in a conceptually similar fashion. The major advantage of this associative memory framework is that it is well established in the neural network community and many of its properties are known and proven.

[1] We have actually run our ranking experiments with the κ_j normalization, but the results have been consistently bad.

Its conceptual basis in biological memory also provides a plausible (though simplistic) explanation for how humans may recommend items to each other.

2.3 Linear Associative Memory

Associative memory is any neural network that deals with "associative learning and retrieval of information"[8]. It remembers patterns of data it has been given, and one retrieves those patterns by giving it similar and/or partial data that have been associated with the different patterns. This fits the notion that biological memories remember and recall by association (e.g., associate "alarm" sounds with danger) rather than by *explicit addresses* (e.g., get memory stored at disk 3, track 4, sector 5).

Traditionally, the emphasis of associative memory has been learning and *exact* recall of data when given only partial input. Say, for example, two four-bit patterns (1110 and 0001) have been given to an associative memory for it to remember, and we call the first two bits 'input' and the last two bits 'output'. One can later show the network an input pattern such as 11, 1X, or X1 (in which X stands for a null state), and it should recall the remembered output pattern 10. Similarly, an input pattern of 00, 0X, or X0 should recall the output pattern 01. In this example, one usually does not care what the outputs are for input patterns 10 and 01.

Linear associative memory (LAM) is one simple implementation of associative memory [9,10,8,11]. Consider the column vector $\boldsymbol{x}$ as input and the column vector $\boldsymbol{y}$ as output. LAM *recalls* the output by a simple linear multiplication

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$$

where $\boldsymbol{W}$ is the *memory matrix* that specifies the connection weights of the associative memory network. This network is illustrated in Figure 1. It has an input layer of neurons, an output layer of neurons, and a set of weighted synapses to connect the two. Given a set of data, $\{\boldsymbol{x}^i, \boldsymbol{y}^i\}, i = 1, \dots, m$, one way to define (that is, train) the memory matrix is by correlation

$$\boldsymbol{W} = \sum_i \boldsymbol{y}^i \cdot (\boldsymbol{x}^i)^T \qquad (3)$$

Another name for this type of neural network is thus Correlation Matrix Memory [9]. This correlation learning mechanism is considered a generalization of *Hebb's postulate of learning*, which was studied extensively in neuropsychology. Basically the postulate says that the connection between two neurons will become stronger if the two neurons are often stimulated together. Over time, the connection will be strong enough that stimulating just one of the neurons will automatically activate the other one. Repetition is thus one way of learning. (See [9] for more details.)

A common preprocessing scheme is to scale all input and output vectors to unit-length. It is a simple proof to show that if the input vectors, $\{\boldsymbol{x}^i\}, i = 1, \dots, m$, form an orthonormal set, then the memory system can recall the output $\boldsymbol{y}^i$ exactly when given the input $\boldsymbol{x}^i$ again.

When one forces $\boldsymbol{x} = \boldsymbol{y}$, the resulting associative memory is considered *auto-associative*. That is, the memory associates the data with itself. The main use of such memory is to recall patterns when only a partial pattern is available. The focus of our work is auto-associative LAM.

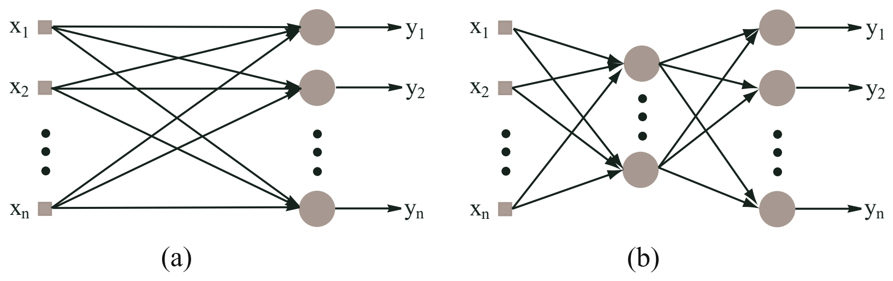


Fig. 1. Figure (a) illustrates a Linear Associative Memory (LAM) neural network. It is simply a two-layer neural network with linear nodes. While it looks like a standard feedforward neural network in terms of its node connections, its learning principles and its memory application are quite different. Furthermore, when it is used auto-associatively, as is done in the CLAM algorithm, the input and output nodes are tied back together, which is not done in standard feedforward neural networks and introduces a new learning dynamic. Figure (b) illustrates a PCA version of LAM. It is a neural network with a hidden layer and all nodes are still linear. The weights are set by principal component analysis.

3 Collaborative-Filtering by Linear Associative Memory (CLAM)

Since associative memory is modelled after biological memories, applying it to collaborative filtering forms a plausible first attempt at explaining how human minds make recommendations. In terms of implementation, we define the column vectors $\boldsymbol{x}^i$ as patterns to the system in which the jth element is

$$\boldsymbol{x}^i_j = \begin{cases} r_{i,j} - \overline{r}_i \ , \ r_{i,j} \neq \perp \\ 0 \qquad\quad\ , \ r_{i,j} = \perp \end{cases} \tag{4}$$

and $r_{i,j}$ and $\overline{r}_i$ are user ratings as defined in Section 2.2. The definition for $\boldsymbol{x}^a$ is analogous. The memory matrix is modified from Equation 3 to be

$$\boldsymbol{W} = \sum_{i=1}^{m} \frac{\boldsymbol{x}^i \cdot (\boldsymbol{x}^i)^T}{\|\boldsymbol{x}^i\|^\alpha} \tag{5}$$

The main difference between Equation 3 and Equation 5 is that Equation 5 describes an auto-associative memory in which the normalization of the input patterns is expressed as part of building $\boldsymbol{W}$ rather than as a separate pre-processing step. For no normalization, $\alpha = 0$. In a typical application of linear associative memory, in which all patterns are normalized to unit-length, $\alpha = 2$. We also leave open the possibility of $\alpha = 1$ for later evaluation.

The process of making recommendations is the same as that of memory recall. We calculate the column vector $\boldsymbol{p}^a$ as

$$\boldsymbol{p}^a = \boldsymbol{W}\boldsymbol{x}^a$$

We then recommend the top N items based on the elements of $\boldsymbol{p}^a$, not including those items for which the active user has already rated.

We will simply refer to collaborative-filtering using linear associative memory as CLAM from now on. The first observation we make is that $\boldsymbol{W}$ completely represents the underlying model and is an $n \times n$ symmetric matrix, in which n is the number of items. Its complexity is independent of m, the number of users. In general, $m \gg n$, so the difference can be significant.

Another notable property about CLAM is its ability for incremental learning. Other model-based CF algorithms tend to support only batch learning, and their models have to be completely rebuilt when more data is available. It is a simple observation that if $\boldsymbol{W}(m)$ stands for the memory matrix given m users, and the data for a new, $(m+1)$th, user become available, then the memory matrix can be updated by $\boldsymbol{W}(m+1) = \boldsymbol{W}(m) + \boldsymbol{x}^{m+1} \cdot (\boldsymbol{x}^{m+1})^T / \|\boldsymbol{x}^{m+1}\|^\alpha$. Other simple updating rules can also be devised for the cases where existing users in the training set decide to add, modify, or delete their ratings.

3.1 Effect of Training Set Size

The fact that each user's information is additive in creating the memory model also enables some theoretical analysis. One can imagine there to be a universe of all users that one can get ratings from (e.g., the population of all shoppers). Say the size of this universe is M. If one can actually collect the ratings from all M users, then one can create the *ideal CLAM model*,

$$\boldsymbol{W}^* = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{x}^i \cdot (\boldsymbol{x}^i)^T$$

Here we assume $\alpha = 0$, and the $\frac{1}{M}$ multiplier has no impact on the recommendation but will add clarity to our analysis.

Of course, one does not have the ratings from all possible users, so we assume what one has is a *random subset* of size m. The CLAM model built from this subset is

$$\boldsymbol{W}(m) = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{X}^i \cdot (\boldsymbol{X}^i)^T \tag{6}$$

(Again the multiplicative constant has no effect on recommendation, and we capitalize $\boldsymbol{X}$ just to emphasize it as a random sample.) Under this view, one is effectively trying to estimate $\boldsymbol{W}^*$ by sampling a population of size m. It is well known that this is an unbiased estimator and the expectation of $\boldsymbol{W}(m)$ is in fact $\boldsymbol{W}^*$. Another known result [12] for this estimator is that the variance of each element of $\boldsymbol{W}(m)$ is proportional to

$$\frac{1}{m}\left(1 - \frac{m-1}{M-1}\right)$$

For $m \ll M$, as is usually the case, the variance of the elements of $\boldsymbol{W}(m)$ is approximately proportional to $\frac{1}{m}$ and the standard error, the more useful measure, is approximately proportional to $\frac{1}{\sqrt{m}}$. Thus a larger sample population of users will make a better model with lower standard error, although the rate of improvement will be decreasing. This fits the intuition of many practitioners, but to our knowledge this is the first formal analysis of such phenomenon for a collaborative filtering algorithm.

3.2 User-Based Interpretation of CLAM

Earlier we described the user-based collaborative filtering algorithm for ranking in Equation 2. Let $\boldsymbol{x}^i$ be a column vector with its elements as defined in Equation 4. Let $\boldsymbol{p}^a$ be a column vector whose jth element is p^a_j of Equation 2. The user-based ranking algorithm can be written in vector form as

$$(\boldsymbol{p}^a)^T = \sum_{i=1}^{m} w(a,i) \cdot (\boldsymbol{x}^i)^T$$

Several forms for the weighting function $w(a,i)$ exist [2]. One popular form takes the *vector similarity* of user i and the active user's rating vectors. It is defined as

$$w(a,i) = \frac{(\boldsymbol{x}^a)^T \cdot \boldsymbol{x}^i}{\|\boldsymbol{x}^a\|\|\boldsymbol{x}^i\|}$$

Plugging into the user-based ranking algorithm, we get

$$\begin{aligned}(\boldsymbol{p}^a)^T &= \sum_{i=1}^{m} \frac{(\boldsymbol{x}^a)^T \cdot \boldsymbol{x}^i \cdot (\boldsymbol{x}^i)^T}{\|\boldsymbol{x}^a\|\|\boldsymbol{x}^i\|} \\ &= \frac{(\boldsymbol{x}^a)^T}{\|\boldsymbol{x}^a\|} \cdot \sum_{i=1}^{m} \frac{\boldsymbol{x}^i \cdot (\boldsymbol{x}^i)^T}{\|\boldsymbol{x}^i\|}\end{aligned}$$

The division by $\|\boldsymbol{x}^a\|$ can be dropped because it affects the predicted value of every item equally and thus has no effect on the ranking of items. The summation is simply $\boldsymbol{W}$ defined in Equation 5 with α set to 1. Noting the symmetry of $\boldsymbol{W}$, we have

$$\begin{aligned}(\boldsymbol{p}^a)^T &= (\boldsymbol{x}^a)^T \boldsymbol{W} \\ \boldsymbol{p}^a &= \boldsymbol{W}\boldsymbol{x}^a,\end{aligned}$$

which is exactly our CLAM algorithm.

4 Experiment

4.1 Methodology

To test the effectiveness of the CLAM algorithm we ran some experiments on the MovieLens dataset [13], collected by the GroupLens Research Project at the University of Minnesota. The dataset consists of 100,000 ratings, in the range of one to five, from 943 users on 1682 movies. Each user has rated at least 20 movies, and each movie has been rated at least once. The ratings were gathered at a Web site during a seven-month period from 1997 to 1998.

For each run of an experiment, we randomly divide the dataset such that 90% of the users are in the training set and the other 10% are the testing set. Each user in the testing set in turn is considered the active user. The mean is subtracted from the active user's ratings. For the active user, five of her ratings are randomly chosen to be withheld. That is, I_a, the set of items the active user has rated on, is divided into $I_a^{withheld}$ and I_a^{test}, with $|I_a^{withheld}| = 5$. We apply various collaborative filtering algorithms to the ratings in I_a^{test} and the training set to make recommendations. The systems return the top N products, where N is ten in our experiments. This set is denoted I_N. Obviously none of the items in I_a^{test} is included in I_N.

A maximum sum, $maxSum$, is defined as the sum of all the active user's withheld ratings that are positive. That is,

$$maxSum = \sum_{j \in I_a^{withheld}} \max(0, x_{a,j})$$

Another value, $sumNoPenalty$, is the same summation but limited to only the products returned by the top N recommendations. That is,

$$sumNoPenalty = \sum_{j \in \{I_a^{withheld} \cap I_N\}} \max(0, x_{a,j})$$

We report $sumNoPenalty/maxSum$ as one of the evaluation scores. So for example, if the five movie ratings withheld from the active user are as follows: Scarface = 2, Fist of Legend = 1, The Godfather = 1, American Pie = -1, and Sleepless in Seattle = -1, then the $maxSum$ of 4 would be the sum of the ratings for Scarface, Fist of Legend, and The Godfather. To get a maximum score of 100%, the system must return those three movies as part of the top N recommendations. Say, if only Scarface is included in the top N recommendations, then the score is $2/4$, or 50%.

The above evaluation metric may be considered a bit generous, since it does not penalize recommending a product that the active user has a negative rating on. We therefore also calculate a penalized sum, $sumPenalty$, that is analogous to $sumNoPenalty$ but includes the negative ratings of products in the top N recommendation. That is,

$$sumPenalty = \sum_{j \in \{I_a^{withheld} \cap I_N\}} x_{a,j}$$

We denote the ratio $sumPenalty/maxSum$ as the penalized score and call the ratio $sumNoPenalty/maxSum$ the non-penalized score. We note that the penalty under the penalized score is a bit onerous for our experiments, since in our pre-processing of the MovieLens data, a negative rating only denotes a below-average preference, which does not necessarily mean a dislike for the movie. The penalized score also cannot be neatly interpreted as a percentage, as a negative score is now possible. In any case, we include both scores in reporting our results.

4.2 Results

Basic Performance of CLAM. We ran experiments to test the CLAM algorithm with $\alpha = 0$, 1, and 2. As discussed earlier, CLAM at $\alpha = 1$ is equivalent to a user-based

Table 1. Comparison of the Naive algorithm versus CLAM with different normalizations (α). The Naive method simply recommends the items with highest sum of user preferences (without, of course, the ones the active user has already rated). Each experiment (cell) is the result of 100 runs. Shown are the averages with the standard deviations within parentheses.

	Evaluation score	
	penalized	non-penalized
Naive	0.056 (0.106)	0.129 (0.025)
CLAM ($\alpha = 0$)	0.132 (0.068)	0.198 (0.034)
CLAM ($\alpha = 1$)	0.128 (0.085)	0.204 (0.031)
CLAM ($\alpha = 2$)	0.137 (0.068)	0.220 (0.032)

collaborative filtering algorithm. Since our ranking task also takes into account the popularity of movies, we need to ensure that popularity is not an overwhelming factor and that personalization still plays a significant role. To investigate this bias, we examined a naive algorithm that recommends the top N movies (that the active user has not rated) based on just the sum of all training users' ratings. That is,

$$p_j^a = \sum_{i:r_{i,j} \neq \perp} (r_{i,j} - \overline{r}_i)$$

The naive algorithm is unpersonalized and just attempts to recommend products more users have given higher ratings to. We use that as the base case to demonstrate the enhancement that CLAM does through personalization. For each experiment, there were 100 runs, and the average and standard deviation are shown in Table 1.

As can be seen in the table, the CLAM algorithm performs significantly better than the non-personalized naive method, both for the penalized and the non-penalized scoring. However, the improvement in non-penalized scoring is more pronounced as the standard deviation is smaller. The difference for different α's is relatively small. Since $\alpha = 2$ has performed slightly better, we have decided to use this setting for all subsequent experiments of CLAM. It is also the case that normalizing both input and output (i.e., $\alpha = 2$) is standard practice in using associative memory.

Dimensionality Reduction on CLAM. The model built by CLAM is an $n \times n$ symmetric matrix. Even though it is independent of the number of users, it is still fairly demanding of memory and computation when n is large. Reducing the size of the model further is therefore desirable. Many techniques for dimensionality reduction exist, and we will report our investigation into two of them. The first one is the general technique of principal component analysis (PCA). The second one is the basic technique of magnitude pruning often used in neural networks.

Principal component analysis reduces the effective dimension of $\boldsymbol{W}$ by replacing it with a lower-rank approximation. First one decomposes $\boldsymbol{W}$ by writing

$$\boldsymbol{W} = \boldsymbol{U\Sigma U^T}$$

in which the columns of $\boldsymbol{U}$ are unit-length eigenvectors of $\boldsymbol{W}$ and Σ is a diagonal matrix of the eigenvalues of $\boldsymbol{W}$. Each eigenvalue corresponds to a *principal compo-*

Table 2. Comparison of standard CLAM with $\alpha = 2$ versus the reduced dimensionality versions of it. The dimensionality reduction is done through principal component analysis. Each row represents the number of dimensions (components) in the memory model. Note that the original CLAM model has 849 components. (I.e., its rank is 849.)

	Evaluation score	
	penalized	non-penalized
CLAM	0.137 (0.068)	0.220 (0.032)
PCA (100 components)	0.122 (0.083)	0.213 (0.033)
PCA (50 components)	0.139 (0.070)	0.218 (0.033)
PCA (25 components)	0.114 (0.113)	0.206 (0.031)
PCA (10 components)	0.114 (0.071)	0.202 (0.033)
PCA (5 components)	0.101 (0.066)	0.192 (0.028)

nent. One gets an approximation to $\boldsymbol{W}$ by setting small elements of Σ to zero. The corresponding columns of $\boldsymbol{U}$ would then become useless. If one is retaining k principal components, one can get a tighter representation of the approximation of $\boldsymbol{W}$ if one defines $\boldsymbol{U'}$ as an $n \times k$ matrix of the k principal eigenvectors and $\boldsymbol{\Sigma'}$ as a $k \times k$ diagonal matrix of the *square root* of the principal eigenvalues and

$$\begin{aligned}\tilde{\boldsymbol{W}} &= \boldsymbol{U}'\boldsymbol{\Sigma}'\boldsymbol{\Sigma}'^T\boldsymbol{U}'^T \\ &= \boldsymbol{V}\boldsymbol{V}^T\end{aligned}$$

In the last step we simply represented $\boldsymbol{U'}\boldsymbol{\Sigma'}$ with $\boldsymbol{V}$. Using this approximation, the prediction algorithm is now

$$\boldsymbol{p}^a = \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{x}^a$$

Note that our model is now completely represented by $\boldsymbol{V}$, an $n \times k$ matrix, rather than by $\boldsymbol{W}$, an $n \times n$ matrix. Of course, k is always less than n and generally $k \ll n$. In terms of neural networks, this representation adds a hidden layer of k neurons with the weights now represented by $\boldsymbol{V}$ (Figure 1(b)).

Using the same experimental set up as before, we compare the performance of CLAM ($\alpha = 2$) versus various number of components for the PCA network. The results are shown at Table 2. Note that the PCA representation using 849 components is equivalent to the original CLAM algorithm, since in our experiment the rank of $\boldsymbol{W}$ (i.e., the number of eigenvalues) is 849. Looking at the results, dimensionality reduction does achieve the graceful degradation of performance that we expected as the number of principal components used is reduced. It is especially encouraging to see the results of only five principal components can still significantly outperform the naive (non-personalized) method of recommendation.

Another dimensionality reduction technique we tried was magnitude pruning, which is the setting of a weight to zero when its absolute magnitude is below some threshold. This operation makes the memory model $\boldsymbol{W}$ more sparse. The model can then be stored using more efficient data structures, storing just the non-zero elements of $\boldsymbol{W}$. The general rationale for this heuristic is that neural networks are observed to be

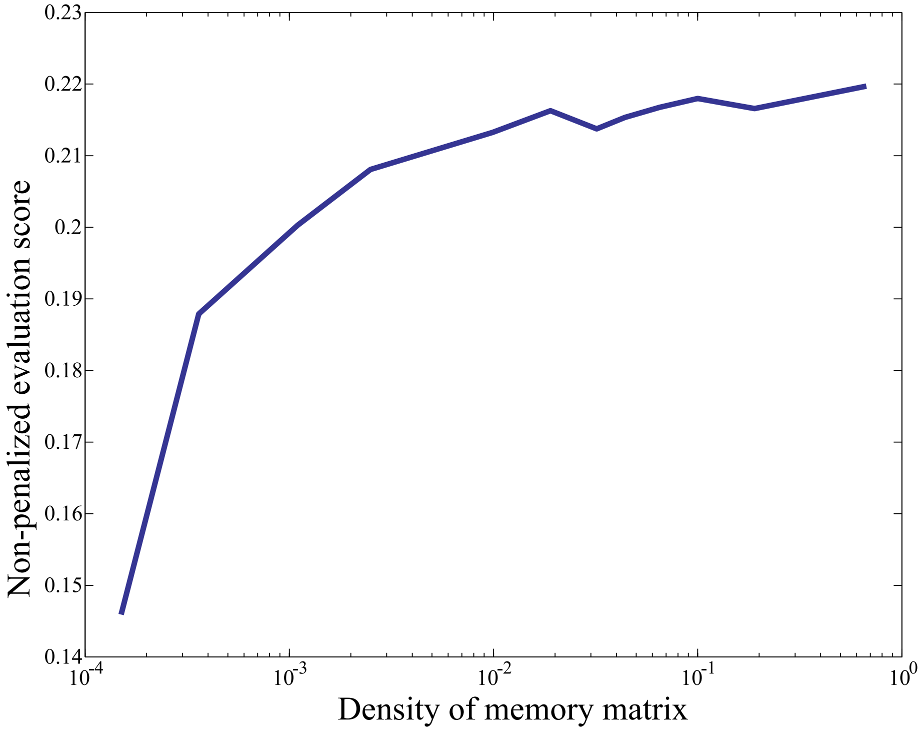


Fig. 2. Result of pruning CLAM. The horizontal axis denotes the density of the memory model, which is the fraction of non-zero entries in the memory matrix. Note that the density is in logarithmic scale. The vertical axis denotes the non-penalized score. The CLAM is pruned at various thresholds, and the resulting density and score are then plotted. Each point is the average of 100 runs.

fairly robust to small perturbations of their weights[2], thus it should degrade gracefully when fairly small weights are set to zero.

Using the same experimental set up as before, we tested the performance of CLAM when its weights are pruned under various thresholds. We define the *density* of a matrix as its fraction of non-zero elements. Pruning with a higher threshold thus makes the memory matrix less dense. In Figure 2 we plotted the density of the memory matrix versus the resulting non-penalized score as we varied the threshold. Note that the density dimension in the figure is in logarithmic scale, and the original (unpruned) version of CLAM has a density of 67%. The surprising find is how much one can prune and still have decent performance. Given its simplicity and its performance, magnitude pruning is a better choice, in our opinion, for dimensionality reduction than principal component analysis.

[2] In fact, in Kohonen's original paper on correlation matrix memories [11], he analyzed the case in which the weights are pruned *randomly* and he only used the randomly pruned version for his experiments.

5 Related Work

Karypis [6] and Sarwar et al. [5] first documented the item-based approach to collaborative filtering. The reasoning was that each item has a set of similar items as neighbors, and recommendation would be some weighted sum (average) of those similarities. This reasoning is exactly analogous to traditional user-based CF algorithms but now neighborhoods are formed around items rather than users. The CLAM algorithm with pruning ends up being one form of their algorithms, although this form is particularly well motivated by neuropsychological principles and led to theoretical results and extensions that are not immediately applicable to other forms.

Billsus & Pazzani [14] also approached collaborative filtering from a neural network framework. They used a multi-layer feedforward neural network to predict one's preference on an item based on other people's preferences on the same item. The neural network was trained from ratings on other items such that the network implicitly models users' similarity to each other. The feedforward neural network design they used is very different from the associative memory design this paper is based on [9,8]. The dominant learning principle for feedforward neural network is gradient descent, whereas for associative memory it is correlation. Tieing the input and output together, as used in CLAM, is a very popular technique in associative memory but not often used in static[3] feedforward neural networks.

Pennock et al. [7] presented an algorithm that can be interpreted as both user-based and model-based. The model is based on the idea of personality and that people with similar personalities share similar preferences for different items. A strength of model-based algorithms is that they tend to be more amenable to theoretical analysis. Pennock et al., for example, suggested a value-of-information analysis for their model that may be too difficult to pursue for user-based models.

Sarwar et al. [15], Goldberg et al. [16], and Hofmann & Puzicha [17] have all used dimensionality reduction techniques in collaborative filtering. The technique is well known and is used to solve many different problems. Sarwar et al., for example, used it to mitigate the effect of sparsity in the ratings matrix. Goldberg et al. used dimensionality reduction techniques to create a constant-time CF algorithm. Hofmann and Puzicha used the technique as a theoretically sound way to model latent attributes.

There are also many other pruning techniques in the neural network literature other than PCA and pruning of small weights. Lam & Stork [18] has a survey of many of these techniques.

6 Future Work

We have already noted that using the $n \times n$ memory matrix $\boldsymbol{W}$ can be a substantial cost in memory and computation if n is large, and we found that dimensionality reduction techniques such as pruning and PCA analysis can reduce such costs. However, such techniques only reduce the costs for deployment. In precomputing the (reduced

[3] We note that feedforward neural networks used for time-series prediction often do have the input and outputs tie together *over time*, but in collaborative filtering one is only making a static prediction.

dimensionality) model, one must still compute every element of $\boldsymbol{W}$ before applying pruning or PCA techniques. When n is often in the tens of thousands and m is in the tens of millions, this precomputation can incurred substantial costs. One of our ongoing research efforts is to modify pruning techniques such that one can approximately decide what elements to prune before committing substantial effort to compute the actual values of those elements. For example, one may devise a “cheap” distance metric to quickly discern and prune highly uncorrelated items [19].

Note that in our definition of $\boldsymbol{x}_j^i$ in Equation 4, an unrated item is given a zero rating. Examining the CLAM algorithm as expressed in Equation 6, we can denote the (j, k)th element of $\boldsymbol{W}(m)$ as

$$\begin{aligned} W_{j,k}(m) &= \frac{1}{m}\sum_{i=1}^{m} X_j^i X_k^i \\ &= \frac{\sum_{i=1}^{m} I(X_j^i \neq \perp, X_k^i \neq \perp)}{m} \cdot \\ &\quad \frac{\sum_{i=1}^{m} X_j^i X_k^i}{\sum_{i=1}^{m} I(X_j^i \neq \perp, X_k^i \neq \perp)} \\ &= \tilde{P}(X_j \neq \perp, X_k \neq \perp) \cdot \\ &\quad \tilde{E}[X_j X_k | X_j \neq \perp, X_k \neq \perp] \\ &= \tilde{P}(X_k \neq \perp) \cdot \tilde{P}(X_j \neq \perp | X_k \neq \perp) \cdot \\ &\quad \tilde{E}[X_j X_k | X_j \neq \perp, X_k \neq \perp] \end{aligned}$$

in which $\tilde{P}$ and $\tilde{E}$ stand for estimated probability and expectation, respectively, and $I(\cdot)$ is the identity function. This decomposition has an intuitive interpretation. The weight between item j and k is the product of the popularity of item k ($\tilde{P}(X_k \neq \perp)$), the popularity of “purchasing” item j with item k ($\tilde{P}(X_j \neq \perp | X_k \neq \perp)$), and the actual correlation of ratings between item j and item k from users who have “purchased” both items ($\tilde{E}[X_j X_k | X_j \neq \perp, X_k \neq \perp]$). Of the three terms, the item-k popularity is the most debatable in terms of how it should contribute to a system’s recommendation. Almost by definition, most people tend to like popular items. However, many have argued that traditional marketers already spend most of their effort on promoting popular items, and the real opportunity for computerized recommendation systems is the discovery of hidden “gems” that one may not otherwise learn about from traditional promotional channels. Another problem with popularity is that new items will score low on the popularity scale, simply because people have not had enough time yet to learn about the item and purchase it. Thus one may want to scale popularity to account for factors such as time. In any case, having shown how to isolate popularity’s effect in CLAM’s recommendation, we plan to investigate further various ways to modify it to make better collaborative filtering systems.

The user input to recommendation systems are generally divided into *implicit* and *explicit* ratings. Implicit rating is just whether one has “purchased” an item, whereas explicit rating is one’s actual preference for the item after having experienced it. Explicit rating is much more expensive to obtain than implicit rating, since an implicit rating can often be gathered through passive observation while the user must be queried to elicit an explicit rating. Explicit rating is also more informative since its existence automatically *implies* the purchase of the item by the user. In many applications both types of information are available. For example, a retailer may have extensive purchasing records (implicit ratings) plus some comparatively smaller surveys of customer preferences (explicit ratings). None of the collaborative filtering algorithms we know of can handle both data types simultaneously. However, looking at our decomposition of $W_{j,k}(m)$ above, the term $\tilde{P}(X_j \neq \perp | X_k \neq \perp)$ depends only on implicit ratings, while the term $\tilde{E}[X_j X_k | X_j \neq \perp, X_k \neq \perp]$ can be calculated separately from explicit ratings. This suggests the possibility of developing an algorithm that can combine both implicit and explicit ratings in computing its recommendation. We hope to explore this direction in our future work.

The analysis of the effects of training set size in Section 3.1 is very preliminary and much more work needs to be added. For example, the analysis assumes a single static set of ratings from each user, when in fact each user only gives a random subset of her ratings. Since both the number of users and the number of ratings from each user can be influenced through different data collection methods, one ought to take both into account when analyzing their effects on improving model estimation. Furthermore, we must analyze the effect of variance in model estimation on the actual *ranking* outcome. Depending on the value of $\boldsymbol{W}^*$ and $\boldsymbol{x}^a$, small reduction in the variance of the estimated model may have a big or no influence on the resulting item ranking.

In situations where one can request users to rate specific items (i.e., active learning), one may ask for ratings that will most likely reduce the variance of certain matrix elements. Take movies as an example, if all the users in one's database who have rated Star Wars highly have also rated The Phantom Menace highly, and vice versa, then one is unlikely to build a better model by asking a new user what he thought of those two movies. On the other hand, when requesting ratings from the active user, the system would want to ask about "archetypal" movies that can quickly locate his interest rather than movies that most people already like. Boutilier et al. [20] had performed some value of information analysis for active collaborative filtering, and we are interested in applying similar analysis to the CLAM algorithm.

Finally, we must note that other forms of associative memory exist, such as nonlinear PCA and Boltzmann machine. Duda, Hart, and Stork [22] and Haykin [9] list many examples. These associative memories also have many known interesting properties. Their appropriateness for collaborative filtering remains to be investigated.

7 Conclusion

In this paper we have developed a form of item-based collaborative filtering algorithm based on linear associative memory, which is itself motivated by neuropsychological principles. This algorithm thus can serve as a primitive first step at explaining how human memory performs the recommendation function. We elaborated on the statistical estimation aspect of the algorithm, and showed that the benefit of the number of users in the database to CLAM's model estimation is proportional to $\frac{1}{\sqrt{m}}$. We also showed that the algorithm has a user-based interpretation. In addition, the complexity of this algorithm is independent of the number of users, and we applied pruning and PCA analysis to further reduce complexity. Finally, we ran various experiments to show its performance.

References

1. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating "word of mouth". In: Proceedings of the ACM CHI '95 Conference on Human Factors in Computing Systems. (1995) 210–217
2. Herlocker, J., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual international ACM SIGIR conference (SIGIR '99). (1999) 230–237
3. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98). (1998)
4. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for e-commerce. In: Proceedings of the 2nd ACM E-Commerce Conference (EC'00). (2000)
5. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommender algorithms. In: Proceedings of the WWW10 Conference. (2001)
6. Karypis, G.: Evaluation of item-based top-N recommendation algorithms. Technical Report TR00-046, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN (2000)
7. Pennock, D.M., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI 00). (2000)
8. Hassoun, M.H.: Fundamentals of Artificial Neural Networks. The MIT Press (1995)
9. Haykin, S.: Neural Networks: A Comprehensive Foundation. Macmillan College Publishing (1994)
10. Hassoun, M.H., ed.: Associative Neural Memories: Theory and Implementation. Oxford University Press (1993)
11. Kohonen, T.: Correlation matrix memories. IEEE Transactions on Computers **C-21** (1972) 353–359
12. Rice, J.A.: Mathematical Statistics and Data Analysis. Duxbury Press (1995)
13. University of Minnesota: (MovieLens Data Set) http://www.grouplens.org/data/.
14. Billsus, D., Pazzani, M.J.: Learning collaborative information filters. In: Proceedings of the 15th International Conference on Machine Learning (ICML 98). (1998)
15. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.T.: Application of dimensionality reduction in recommender system – a case study. Technical Report CS-TR 00-043, Computer Science and Engineering Dept., University of Minnesota, Minneapolis, Minnesota (2000)
16. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Technical Report UCB ERL M00/41, Electronics Research Laboratory, University of California at Berkeley, Berkeley, CA (2000)
17. Hofmann, T., Puzicha, J.: Latent class models for collaborative filtering. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI '99). (1999)
18. Lam, C.P., Stork, D.G.: Learning network topology. In Arbib, M.A., ed.: The Handbook of Brain Theory and Neural Networks. 2nd edn. MIT Press, Cambridge, MA (2003)
19. McCallum, A., Nigam, K., Ungar, L.H.: Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of KDD-2000. (2000)

20. Boutilier, C., Zemel, R.S., Marlin, B.: Active collaborative filtering. In: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI'03). (2003)
21. Lam, C.P.: SNACK: Incorporating social network information in automated collaborative filtering. In: Proceedings of the 5th ACM Conference on Electronic Commerce (EC'04),, New York, New York (2004)
22. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons (2001)

Scaling Down Candidate Sets Based on the Temporal Feature of Items for Improved Hybrid Recommendations

Tiffany Ya Tang[1], Pinata Winoto[2], and Keith C.C. Chan[3]

[1, 3] Department of Computing, Hong Kong Polytechnic University, Hong Kong
{cstiffany, cskcchan}@comp.polyu.edu.hk
[1, 2] Department of Computer Science, University of Saskatchewan, Canada
piw410@mail.usask.ca

Abstract. The intensive information overload incurred by the growing interest in the Internet as a medium to conduct business has stimulated the adoption of recommender systems. However, scalability still remains an obstacle to applying recommender mechanism for large-scale web-based systems where thousands of items and transactions are readily available. To deal with this issue, data mining techniques have been applied to reduce the dimensions of candidate sets. In this chapter in the context of movie recommendations, we study a different kind of technique to scale down candidate sets by considering the temporal feature of items. In particular, we argue that movies' production year can be regarded as a "*temporal context*" to which the value (thus the rating) of the movie can be attached; and thus might significantly affect target users' future preferences. We call it the temporal effects of the items on the performance of the recommender systems. We perform some experiments on the MovieLens data sets. The results show that the temporal feature of items can not only be exploited to scale down the candidate sets, but also increase the accuracy of the recommender systems.

1 Introduction

When the World Wide Web becomes an increasingly popular medium, information overload intensifies: users are overwhelmed by the information pouring out from the web, and are usually confused by which information should be consumed. Fortunately, recommender system offers a feasible solution to this issue. However, since the volume of transactions and web activity are increasing, it is not trivial to make useful recommendations. As such, how to select candidate items for personalized recommendations becomes critical. Recommender systems perform personalized information filtering based on either the content features of items (content-based filtering) or the "word of mouth" social ratings of users (collaborative filtering). Collaborative filtering and content-based filtering are two most commonly used approaches in many recommender systems (other approaches include utility based and rule-based systems). Although each approach has both advantages and disadvantages in providing high quality recommendations, a hybrid recommendation mechanism incorporating components from both of the methods would yield satisfactory results in many situations.

Recently, it is observed that there is a growing interest concerning designing efficient recommendation algorithms on e-commerce system. In spite of these, many of these methods still suffer problems in terms of scalability due to the size of the data involved. Hence, several methods have been proposed to address the scalability issue [21, 29].

In this chapter, we will show a solution to the scalability issue from another perspective, i.e. scaling down candidate sets using the temporal features of products or services (in the context of this chapter, products or services refer to movies). Intuitively, if only considering recent movies as candidate sets would not reduce the accuracy of recommendations, it would then not be necessary to include old movies for recommendations, which can significantly prune the candidate sets space, and thus, select recent items only to generate recommendations. Although there has been extensive research on recommender systems [3, 6, 8, 15 23], as far as we know, there is little research addressing this particular issue. It is our hope that our work here would shed light on future research on recommender systems. For instance, one future research could possibly analyze the temporal trend of the user preferences on the web based upon a temporal analysis of their web activities and the products they purchased or the services they received.

The rest of this chapter is arranged as follows. In Section 2, we briefly present background information concerning recommender system. We then introduce related work and explain the motivation of our work through an example in Section 3. In Section 4, we discuss in details the experiments we conducted on the MovieLens data set. We conclude this chapter by pointing out our future research and the potential impact of our study on future recommender systems.

2 Background

There are two common methods to provide personalized recommendations: *content-based filtering* and *collaborative filtering* [11]. Content-based filtering systems build user models that link the contents of the information a user has browsed to the preferences of the user concerning those artifacts. Collaborative filtering systems build user models that link the information preferences of a user to those of other users with similar tastes or preferences.

2.1 Content-Based Filtering Systems

As its name signifies, content-based filtering systems recommend items based on the contents of the items a user has experienced before. News Dude [4] performs a content-based filtering to learn users' news-reading preferences and recommend a new story. To accomplish this, the system builds and maintains two kinds of user models. The first is a short-term user model that measures the interestingness of a story by how close it is to the stories that the user has read before. In this case, similarity measurement is based on the *co-occurrences of words* appearing in these stories. The second user model carries a probabilistic classifier that assigns a probability of interest to a new story by comparing how frequently its words occur in those stories the user regards as interesting to those the user regards as of no interest. When a new article comes, News Dude will first consult the short-term user model to predict whether the user will be interested in it; if not, the second user model is consulted. Both the user models would match user profiles with text documents. Results show that this kind of user model performs better than either model in isolation. WebWatcher [12] is another content-based filtering system, in which on entering the site, a user is accompanied by a learning agent who can learn from what the user has stated when first entering the system, and the subsequent contents of the pages he/she has visited. WebWatcher would try to link the contents of a page to the user's interest. Experimental results show, however, that WebWatcher's ability to recommend hyperlinks is relatively low (only 48% [12]) due to the drifting interests of users.

Since user profiles in the content-based filtering system are built through an association with the contents of the items, one of the disadvantages of this method is that it tends to be quite narrowly focused with a bias towards highly scored items. Thus, a user might be restricted to items that are very similar to the ones he/she has 'consumed' before. Another disadvantage of the content-based filtering lies in the fact that it only considers the preferences of a single user. For News Dude, even if multiple users' preferences are given (in the form of the co-occurrences of words best representing the preferences of the users), the system cannot learn across this cluster of users to inform the prediction process. Collaborative filtering is an approach capable of exploiting information about other similar users.

2.2 Collaborative Filtering Systems

Collaborative filtering systems make recommendations by observing like-minded groups. These systems work by first building a database of customer preferences over items. Then, when a customer arrives, he/she will be matched against the database to find his/her *neighbors* who have historically had similar preferences to him/her. Finally, the systems will recommend item(s) that highly rated by his/her neighbors.

GroupLens is a pioneer *automated* recommendation system which successfully adopts the collaborative filtering approach [18]. Developed for Usenet news and movie recommendation, GroupLens is a collaborative filtering system where user profiles are built based on their ratings of products or services. It determines similarity between users and predicts how well users will like new articles based on the ratings from similar users. Firefly is another collaborative filtering system which provides recommendations for music albums and artists [24]. Results show that Firefly is surprisingly good at predicting a user's future music tastes; even when it ignores the features of the music albums and simply matches one user against cluster of similar users.

Compared to content-based filtering approach, collaborative filtering is more popular in providing personalized recommendations [20]. However, there are several drawbacks with collaborative filtering approach, among them:

- Relying on explicit ratings of users
  The performance of collaborative filtering systems relies heavily on explicit ratings of users. Hence, lack of explicit user ratings and the sheer amount of data could pose serious problems to the systems. Besides, since ratings are subjective opinions of users themselves, thus are prone to biases. Therefore,

the issue of trust towards the system also emerges as an important issue [11], since recommendations derived from explicit ratings is vulnerable to external manipulation.

- Making recommendations for homogeneous types of items or services
  Systems like GroupLens, Firefly, MovieLens make recommendations on homogeneous types of simple products, i.e. CDs or Movies. In this case, the more items two users have rated similarly, the closer they are in terms of the preferences inferred from the ratings. However, they would fail to be directly applied to heterogeneous environments or even complex product where more than one aspects of the product should be considered [30].
- Failing to measure the significance of the correlation between ratings [4]
  Ratings towards each item in the system are treated evenly, which is reflected into an equal weight of each item in finding the neighborhood of a target user. But in many cases, some items might be more valuable than others in clustering the users into similar groups; thus, promoting a higher accuracy of the recommendation. A pure collaborative filtering fails to utilize this feature.

2.3 Hybrid Approach

A pure content-based filtering approach only considers the preferences of a single user, and concern only the significant features describing the content of an item; whereas, a pure collaborative filtering approach ignores the contents of the item, and only makes recommendations based on the comparison of a user against clusters of other similar users. Consider, however, that item information (features that would best categorize the most-preferred items) can be obtained through content-based filtering and user information (relative distance of the user to other clusters of users, and users' opinions) can be obtained from collaborative filtering. Then, by combining these two techniques, perhaps we can have both individual as well as collective experiences with respect to the items being recommended.

Fab [2] can be regarded as a two-layered filtering system. The first layer is a content-based filtering, which ranks documents by the topic, and then ranked documents are sent to a user's personal filter. In the second layer, a user's relevance feedback is used to modify both the personal profile filter and the topic filter. It is obvious that only filtered documents are added to the list of candidate documents to be recommended. In the case when there are no ratings for a target item, Fab can still recommend it appropriately based on content-based filtering. Other systems which apply hybrid approach to make movie recommendations include [3, 7].

Although hybrid approaches have achieved success both in research and practice, unfortunately, as far as we know, most hybrid approaches have focused on the contents of items rather than the temporal feature of them (e.g. movies' production year).

3 Our Proposed Approach

It is recognized that scalability remains a big challenge for recommender system [20, 29]. In this study, we attempt to investigate whether or not items' temporal feature could be exploited to scale down the candidate sets for improved hybrid recommendations. For instance, if by only considering recent movies as candidate sets would not reduce the accuracy of recommendations, it is desirable to only include these recent movies because of a lower computational cost.

This temporal-constraint collaborative filtering works on the hypothesis that a movie's temporal feature, regarded as a "*temporal context*" to which the value (thus the rating) of the movie can be attached, reflects the situational environment where the movie was produced and might significantly affect the target users' future preferences. The newer the movies he/she has liked/disliked, the greater they will affect his/her future preferences. We call it the *temporal effects* of the items on the performance of the recommender systems. In addition, users would prefer to receive new items they like rather than some relatively old items from the recommender system. Therefore, our focus is to study whether the temporal feature of an item, if considered, would have an effect on the performance of a recommender system or not. If these temporal features can be exploited to increase the scalability of the recommender system without sacrificing the accuracy of the recommendation made, it would then greatly assist the recommender system in locating the most informative candidate sets and thus would greatly reduce system's response time to provide on-line "just-in-time" personalized recommendations. In this section, we will explain our main motivation through an example.

3.1 A Motivational Example

Intuitively, a thriller released in 1987 would be quite different from a thriller released in 1997, because, the contextual features (the technology used, the story itself, etc.) involved in these two movies would be significantly different. For example, *The 39 steps*, a thriller released in 1935, is certainly different from *The Bourne Identity*, a thriller released in 2002. To better illustrate how the temporal features of products or services might affect the prediction upon which a recommender system makes, let us consider the following example.

Suppose there are four movies, A, B, C, and D, with their production year and genre shown in Table 1; in addition, two users John and Alex have rated these four movies as shown in Table 2.

Table 1. Movies and their features (production year, and genre)

	Movie A	Movie B	Movie C	Movie D
Production year	1970	1991	1993	1995
Genre	Action	Action	Thriller	Action

Table 2. Movies and their ratings from John and Alex

	Movie A	Movie B	Movie C	Movie D
John	4	5	3	4
Alex	1	5	3	N/A

where rating 1=worst and 5=best. If John is considered as Alex's neighbor, then the system may recommend movie D to Alex, because John highly rated it (equals to 4). But, the question is whether or not John is considered as Alex's neighbor. Suppose the system only consider the dissimilarity of the ratings between them, i.e. they are neighbor if the total dissimilarity ≤ 2. Then, John will not be Alex's neighbor, because their total dissimilarity equals to $4 - 1 = 3$, which comes from movie A. But if we could ignore movie A, which released in 1970, then John is considered as Alex's neighbor. But can we ignore it? Intuitively, we can due to the following two reasons:

- The situational environment of the movie can change a lot due to the movie's contextual existence in the world. Features most affecting the tastes of a movie-goer include the plot of the movie, the contextual environment when the movie was released, the popularity of the cast and crew, etc. Therefore, a movie released in 2002 could be very different from a movie released in 1970 in many aspects.
- More recent movies a user has seen could have greater impact on his/her future preferences, which is similar to the notion discussed in [28]. Some people may watch an old movie long time ago while others watch it recently, but they rate it recently. Thus, the ratings of an old movie may not be as accurate as the ratings of a new movie, since some people may forget the details of the old movie.

A more accurate prediction can be made if we know when they first watch the movie. Unfortunately, all movie databases do not record this information. Thus, if the system makes recommendations only based on recently released movies, the candidate space could be significantly pruned; therefore, we hypothesize that when making recommendations, if the system can only consider those recently released movies in order to make recommendations faster without compromising the accuracy of the overall performance, it is very desirable especially for on-line decision making environment. Technically, if only recently released movies are considered, then the candidate sets can be pruned significantly, which makes the overall recommendation-making more efficient in terms of space and time complexity. In the rest of this chapter, we will show through experiments how pruning the candidate sets will affect the recommendation accuracy.

Moreover, movies, like news articles, or research papers, are situated in a context [16] where the elements within each context might greatly affect the values of the items being recommended. Indeed, recommending old items is different from recommending new items, which become another interesting issue in the recommender system. Concerning user satisfaction, which according to Herlocker *et al.* [9] still remains the bottom-line measurement of the successfulness of recommender systems, most users would like to see more recent items instead of old items. Thus, the successfulness of recommender system in recommending new items is more interesting than recommending old items. Moreover, since more users watch new items rather than old items, sparse ratings are expected to be observed in old movies rather than in new movies. Although this issue sounds quite intuitive, to the best of our knowledge, few papers have addressed it.

3.2 Our Proposed Hybrid Recommender Mechanism

3.2.1 Collaborative Filtering Aspect

One of the key steps in the classic collaborative filtering is to form *neighbors* for a target user. The process of neighborhood formation can formally be expressed in the following way.

Given a target user a, the recommender system should find an *ordered* list of h neighboring users $N=\{N_1, N_2, \ldots N_h\}$, such that $a \notin N$ and $sim(a, N_1) \geq sim(a, N_2) \geq \ldots \geq sim(a, N_h)$, where $sim(a, N_i)$ denotes the similarity of user model a to its neighbor N_i.

There are two commonly adopted similarity measurements in the literature: *Pearson-correlation based* and *Cosine-based similarity*. Since Pearson-correlation approach outperforms the Cosine-based approach as found in [5], we apply the former in our study. Specifically, the Pearson correlation between users a and b is given by:

$$W(a,b) = \frac{\sum_K (V_{a,k} - \overline{V}_a)(V_{b,k} - \overline{V}_b)}{\sqrt{\sum_K (V_{a,k} - \overline{V}_a)^2 \sum_K (V_{b,k} - \overline{V}_b)^2}}$$

Where $V_{i,k}$ is the rating by user i on item k, $\overline{V}_i$ is the mean rating by user i, and K is the set of items co-rated by both a and b. The value of Pearson correlation is between 0 and 1. As suggested in [8], we devalue the Pearson correlation by $|K|/50$ if $|K| < 50$. The reason is that we are less interested in a neighbor whose number of co-rated items is relatively small. In other words, we are only interested in those neighbors who have rated many similar items with the target user.

After computing the (adjusted) Pearson correlation between the target and each of the user profiles, the system will select 30 most correlated users as the neighbors of each target user. After that, it can then calculate the predicted rating of target user a on item j using the following equation:

$$P_{a,j} = \overline{V}_a + \frac{\sum_B W(a,b)(V_{b,j} - \overline{V}_b)}{\sum_B W(a,b)}$$

Where B is the set of all neighbors being considered, i.e. the 30 closest neighbors, in our study. The mechanism above is commonly used in the literature, e.g. [8, 15, 29].

In addition, instead of performing collaborative filtering on the whole pool of the candidate sets, we proposed to perform the operation on only a subset of the candidate items; in other words, items[1] are first ordered temporally, then the system will select neighbors based on those recent movies only. By doing so, it is obvious that the candidate space can significantly be pruned for faster, just-in-time recommendations, which is especially desirable for large-scale on-line systems. In order to show that our proposed temporal value-based collaborative filtering mechanism works well without compromising the accuracy of the overall recommendations, we conducted a series of experiments which will be described in more details in the next section.

[1] In the context of this paper, items refer to the movies; however, they can cover others such as books, CDs, papers, or even users' path traversal sequences as described in [28].

3.2.2 Content Aspect

In our approach we explicitly consider movies' production year in finding neighbors. Therefore, the computation of Pearson correlation is restricted for recent movies only. However, specifying the range of production years becomes another important issue, i.e. which old movies should be excluded from the consideration? Rather than looking for the optimal duration (production years), this chapter intends to show the general effect of items' temporal feature in collaborative filtering, because we believe that the optimal duration setting may vary for different data sets.

3.3 Related Work

3.3.1 Temporal Effect of Items

In fact, a movie's production year is essentially an attribute of the movie entity just like other attributes such as Genre. Using the Genre within the recommendation process has been studied previously [3, 27]. However, we believe that the production year of the movie might induce a non-negligible role on predicting users' preferences. We refer to each movie's production year as the temporal feature, and attempt to study how this temporal feature would affect the global performance of recommender systems. One recent study by Adomavicius and Tuzhilin [1] is similar to ours in the sense that movies' releasing year is considered when making recommendations, enabling the system to make multidimensional recommendations as opposed to traditional pair-wise recommendations (user/item). For example, recommending the top 3 romantic/drama movies starring Julia Roberts that were released within the last 5 years. While their focus is on the multidimensional architecture of the system, ours is to investigate whether or not movies' temporal feature will have an effect on the predictions of ratings. Yang *et al.* [28] speculate that "*pages accessed recently have a greater influence on pages that will be accessed in the near future*", and experimental results confirmed their hypothesis. The premise on which [28] is based is similar to ours in the sense that our proposed collaborative filtering algorithm will only focus on those temporally constrained items (newer items), therefore it is different from the majority of other studies in the area where collaborative filtering is performed on the whole pool of data.

To the best of our knowledge, in the collaborative filtering research community, few researches have addressed this issue, although temporal feature of items has been recognized as an important indicator in its own right. For example, Paepcke *et al.* [16] argue, in the context of document retrieval, that '*the time at which the document was published*' will contribute to the value of the document. Popescul *et al.* [17] also consider the publication year of each scientific literature for CiteSeer to analyze the '*temporal trends in hyper-linked document databases*'.

3.3.2 Dimension Reduction

In response to the growing needs of e-commerce systems, improving recommender systems' scalability becomes vital. Data clustering has been proposed to scale down candidate sets [27]. However, Breese *et al.* [5] point out that under some circumstances, the cluster-based method have worse accuracy than the commonly adopted nearest neighbor algorithm. In [29], Yu and his colleagues study the scalability issue from another approach, namely information theoretic approach, to measure the relevance of a target user for predicting the preference of a target item. Sarwar *et al.* [21] applied Singular Value Decomposition (SVD) to reduce the dimensionality of the recommender system databases. Experiment results show that only under some conditions will the SVD-based approach yield better results than a traditional collaborative filtering algorithm. In addition, as pointed out in the paper, the computation of the SVD is very expensive; therefore, it might not possibly be done online. From this perspective, the SVD-driven dimension reduction approach is not feasible for on-line real time recommender systems.

Our approach is different from the above research in that we will not pool the whole data set; instead, we will only select temporal-constrained items as candidate sets, and perform the collaborative filtering approach. In the next section, we will describe in details the experiments we conducted to validate it.

4 Experiments

Two groups of experiment are performed in our study, and each group consists of three independent experiments for three different data sets. All experiments use the same methodology as described in section 3.2 except the data set. In the first group of experiments, the system recommends movies with all production years, while in the second group the system considers the recommendation of new movies only.

4.1 Data Set

The experiments were conducted on MovieLens data set (available from http://movielens.umn.edu) consisting of 100,000 ratings from 943 users on 1,682 movies. Movies' releasing time spans from 1922 to 1998, and rating scales range from 1 to 5.

Table 3. Number of movies and their ratings for various movie-releasing time

	1922-1939	**1940-1949**	**1950-1959**	**1960-1969**	**1970-1979**	**1980-1989**	**1990-1994**	**1995-1998**
#movies	32	45	57	46	56	110	455	881
#ratings	1561	2249	3527	3909	6451	12834	22084	47385

Table 3 shows the number of movies and ratings for various movie-releasing time periods. Since our goal is to study the cost/benefit of data set reduction based on movies' temporal property, we generate three data sets based on movies' releasing time. In the first experiment, we assume that the recommended movies are recent movies released from 1985 to 1998. Thus, we generate the first data set by excluding all ratings for movies before year 1985, resulting in a data set of 1409 movies released from 1985 to 1998. In total 24.6% ratings are discarded. Then, we clean the data set by excluding all users who rate less than 20 movies, or 6.7% users are removed from our consideration. In the rest of this chapter, we call this data set '1985'.

The second data set is generated by adding to the first data set all ratings for movies released from 1980 to 1985, which are rated by those users in the data set '1985'. We call this data set '1980'. The number of ratings in this data set is approximately 8% more than those in '1985'. Finally, the third data set is generated by including all ratings by the same users for all movies regardless of their releasing years. We refer this data set as 'ALL'. We use the same users for all three data set for comparison purpose.

From the data set '1985', we randomly pick up 90 users as our target users (test users). Then, based on the number of movies rated by each target user, we randomly hide 2 to 6 ratings by him/her. If the number of movies rated by a target user is greater than 50, then we hide 5 ratings; however, if the number of movies rated is around 20, then we only hide 2 ratings. The total number of hidden ratings is 451, whose movie production years range from 1985 to 1998. And we use these hidden movies in our first group of experiments (data sets 1985, 1980 and ALL).

As stated earlier in this chapter, we are also interested in the study of recommending new items. Therefore, in the second group of experiments, we use identical data sets and target users except that we choose the most recent five movies as the hidden movies, resulting in 450 hidden movies (mostly produced in 1997 and 1998).

4.2 Evaluation Metrics

According to Herlocker *et al.* [8], there are two key elements to measure the quality of a recommender system, the coverage metrics and accuracy metrics.

The coverage metrics mainly measures the percentage of items for which a recommender system is capable of making predictions, and the accuracy metrics measures the accuracy of the recommender system in predicting the user's preferences. In our analysis, we will only consider the accuracy metrics.

Herlocker *et al.* [8], further divide the accuracy metrics into two main categories: *statistical accuracy* metric and *decision-support accuracy* metric.

Statistical accuracy metric mainly compares the numerical prediction scores against actual user ratings for the user-item matrix in the test data set.

Several statistical accuracy metrics have been adopted previously, including Mean Absolute Error (MAE) (e.g. [15, 22, 24]), root mean squared error [19], correlation between predictions and ratings [10, 13]. Since most of these metrics generally support similar conclusions [6, 8], we only report MAE in our study, where the error is the difference between the predicted rating and the actual rating.

Decision-support accuracy metrics measure how well a recommender system can make predictions that help users select *high-quality* items. Suggested by Herlocker *et al.* [8], and widely adopted in the research community (e.g. [6, 15, 23]) is the *ROC (Receiver Operating Characteristic) sensitivity*. ROC is made based on the assumption that filtering is a binary problem—either users will view items or they will not. As such, when adopted for measuring the accuracy of a recommender system, a predictor is treated as a filter, where an item's high rating equals to user's acceptance of it; while an item's low rating equals to user's rejection of it. The table below shows the 2x2 confusion matrix of the measurement in binary test, where *sensitivity* = TP/(TP+FN), and *specificity* = TN/(TN+FP).

	Actual Rating Good	Actual Rating Bad
Predicted Rating Good	True Positive (TP)	False Positive (FP)
Predicted Rating Bad	False Negative (FN)	True Negative (TN)

ROC curve is a curve plotting the *sensitivity* (vertical axis) vs. *1-specificity* (horizontal axis) of the test [26]. *ROC sensitivity* only covers the area under the ROC curve. Generally, the probability that a randomly selected good item being accepted by the user is referred to as *sensitivity*; while the probability that a randomly selected bad item being rejected by the user is referred to as *specificity* [6]. Hence, a ROC curve demonstrates the tradeoff between *true positive rate* and *false positive rate* in recommender systems. Usually the curve is a non-decreasing line. The more the curve skews to the left-upper corner, the better the performance of the system. In our study, we adopt ROC curve to measure the decision-support aspect of accuracy. In order to make it work, we must differentiate between 'good' items and 'bad' items. We treat items with ratings of 4 and 5 as "good", while items with ratings of 1, 2 and 3 as "bad". These settings are also adopted in [6, 15, 23]. In addition to MAE and ROC curve, we also use positive predictive values of recommendation as our metric, which measures the rate of true positive for all recommended items, i.e. equals to TP/(TP+FP).

4.3 Experiment Results

Fig. 1 shows the average MAE for three data sets 1985, 1980 and ALL from our first group of experiments. The t-statistic test result shows significant differences between the average MAE from data set 1985/1980 and ALL (0.04 and 0.06 respectively). Thus, we may conclude that based on statistical analysis, using data set 1985/1980 will generate less error than using data set ALL.

Fig. 2 shows the average positive predictive value of recommendation by increasing the threshold value of the filter (predicted ratings by neighbors). Higher threshold signifies that the system becomes more selective, i.e. only recommending movies whose predicted ratings are higher than the threshold. As shown in the figure, data sets 1980 and 1985 have higher average positive predictive values compared to those of data set ALL. Indeed, the maximum predictive values for data set 1980 and 1985 is on threshold value 3.9. In the case when threshold value equals to 4, the predictive values of data sets 1980/1985 and ALL are not significantly different. But for other threshold values, the gaps are around 2% to 5%.

Fig. 3 shows a linear relationship between the threshold values and number of recommended movies. If we adjust the threshold from 3.5 to 4 then the reduction of number of movies is approximately 50% (half), which is a significant loss. As shown in the figure, the differences of the loss for data set 1985, 1980 and ALL are not significant. Thus, in terms of the capability of recommending movies, there is no significant difference between reduced data sets (1985/1980) with non-reduced data set (ALL).

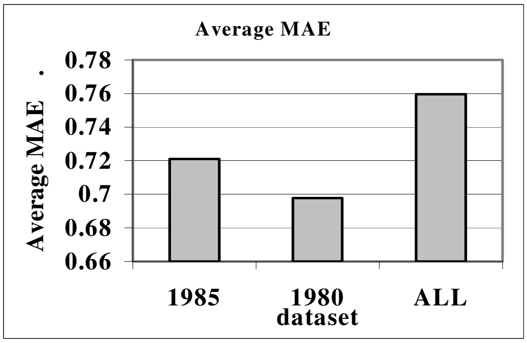


Fig. 1. Average MAE from the first group of experiments

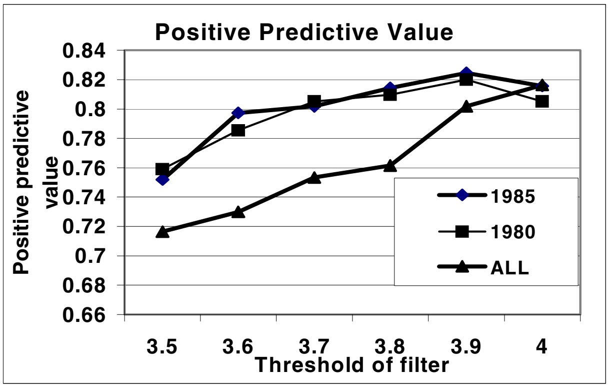


Fig. 2. Positive predictive values of recommendation for various thresholds in the first group of experiments

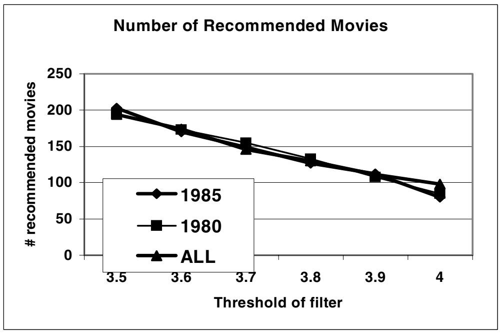


Fig. 3. Number of recommended movies for various thresholds in the first group of experiments

Fig. 4 shows the ROC curve for all three data sets. The curve of data sets 1980/1985 are further from the diagonal line compared to data sets ALL, which means a better performance is gained by data sets 1980/1985. The fact that the selection of data sets based on their temporal feature may increase the accuracy of recommendation in terms of both MAE and ROC justifies our hypothesis that temporal feature of items should be considered in making recommendation. It can not only reduce the search space, but also increase the accuracy of recommendation.

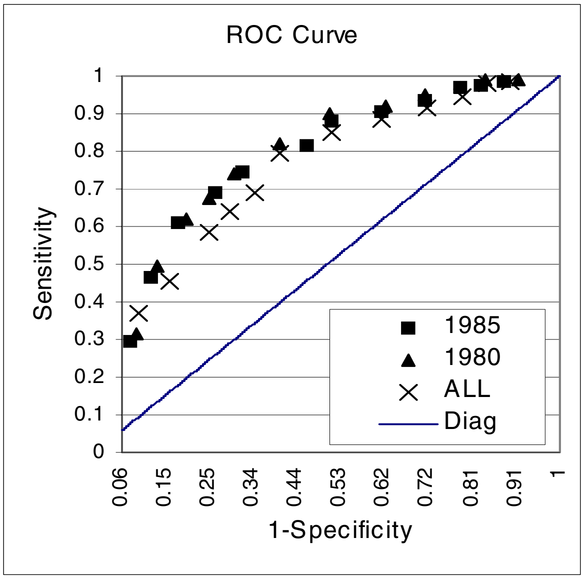


Fig. 4. ROC curves for three data sets in the first group of experiments

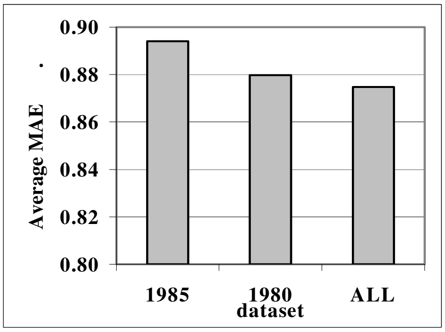


Fig. 5. Average MAE from the second group of experiments

Fig. 5 until Fig. 7 show the results of our second group of experiments. Fig. 5 shows a slightly better MAE of data set ALL than that of data sets 1980/1985 (less than 0.02). Fig. 6 shows that the positive predictive values of data set ALL is mostly better than those of data set 1985, but worse than those of data set 1980.

And Fig. 7 shows that the ROC curves of all three data sets are almost the same. From those three figures we may conclude that there is no significant differences between choosing data set ALL and 1980, except for data set 1985 in terms of the accuracy of recommendation. In other words, if the system is required to recommend new movies, then it might be better to compute the neighborhood based on non-reduced data set. However, if we compare the benefit of 24.6% reduction of data set (from ALL to 1985), then the gain of 2% predictive value may not be so important.

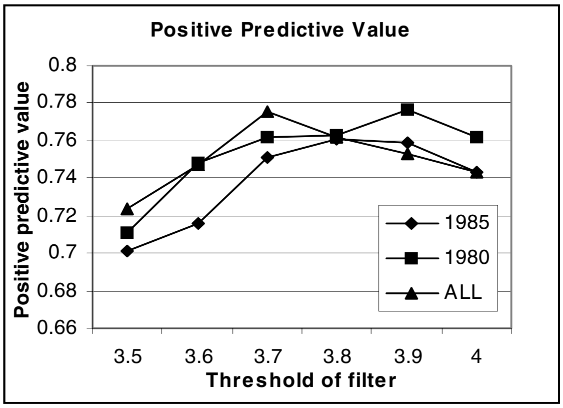


Fig. 6. Positive predictive values of the second group of experiments

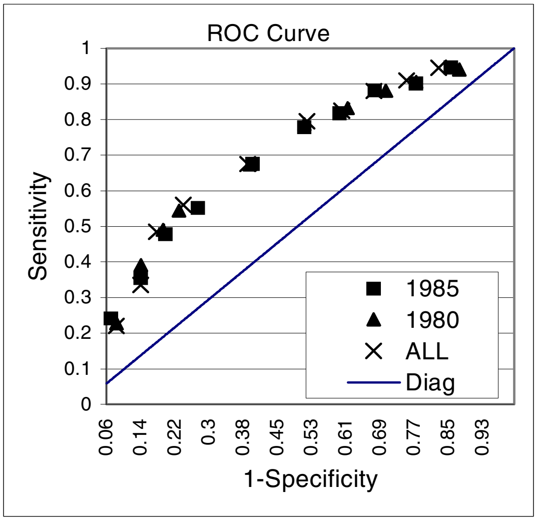


Fig. 7. ROC curves for three data sets in the second group of experiments

If we compare the results of the two groups of experiments, less accuracy is observed for the system to recommend only new movies than to recommend all movies. For example, the MAEs of the first group (0.698 – 0.760) are lower than those of the second group (0.875 – 0.894). And the ROC curves of the first group are more convex to the left-upper corner than those of the second group. This result supports our conjecture that using older movies for recommending newer movies (group 2) is less accurate than using any releasing time (including newer) movies in recommendation (group 1).

From the two groups of experiments conducted, we conclude that the temporal feature of items is very important in providing recommendations. We believe that the temporal feature of items can be exploited to not only scale down the huge amount of data set, especially for web-based recommender system, but also allow us to quickly select high quality candidate sets to make more accurate recommendations.

5 Discussions

The remaining question is how the temporal feature can affect the result. One of the explanations is that old movies are rated sparsely, while newer movies are not. Another explanation is that there is bias towards the ratings of old movies. Intuitively, only those good movies are still surviving today (watched and rated). Therefore, they may get higher ratings and according to the suggestion by Herlocker *et al.* [8], they cannot be used to differentiate user taste; thus, reducing the accuracy of recommendation. If we analyze the ratings in MovieLens data set, it is not true that old movies are rated sparsely, but it is true that they receive higher average ratings (see Table 4). As shown in the second row of Table 4, movies released between 1970 and 1989 are rated more frequently than those released before and after that. Movies released between 1990 and 1998 are rated sparsely (as sparse as those released before 1949). However, approximately 79.4% of movies are released between 1990 and 1998. Thus, we conclude that the sparsity of old movie is not the primary factor in reducing the accuracy of recommendation. However, the average ratings of old movies are higher than the average ratings of newer movies, as shown in the last row of Table 4. Therefore, it is very likely that the bias of ratings towards old movies explains the result of our experimentation. This result supports the suggestion made by Herlocker *et al.* [8] that movies that are liked by most people cannot be used to classify an individual user taste; therefore, these items should be under-weighted (or excluded) from the recommendation system.

Another challenging issue regarding recommender system is the evaluation approach. So far, current research on evaluating recommendation approach have largely followed suggestions made in [8], which focuses mainly on the accuracy of collaborative filtering algorithms, and its potential as a good filter to support decision

Table 4. Average number of ratings for movie released in various years and average ratings of each of them

	1922-1939	**1940-1949**	**1950-1959**	**1960-1969**	**1970-1979**	**1980-1989**	**1990-1994**	**1995-1998**
ave. #rating	48.8	50.0	61.9	85.0	115.2	116.7	48.5	53.8
ave. rating	3.91	4.01	3.94	3.88	3.87	3.77	3.50	3.34

making. However, as argued recently in [9], more research should be conducted to study a wider range of non-accuracy metrics, e.g. '*the novelty and serendipity of recommendations, and user satisfaction and behavior in the recommender system*'. In spite of these, it is interesting to note that no matter what kind of goals an evaluation aims at, the candidate data sets still include all the available data sets in the databases. In other words, users are still receiving recommendations from the pool of all data sets appeared in the system. It is obvious that by doing so, users are continuing to receive personalized items, especially old ones.

Moreover, as argued in [9] that there remains a significant challenge in deciding what measures to use in comparative evaluation, and in different domains, although there is a bottom-line measure of recommender system success by user satisfaction. However, user satisfaction, by itself, is not a trivial evaluation goal. There could be many different dimensions of user satisfaction. For example, how would the ability of recommenders to explain their recommendations to users [25], the degree to which recommendations made are not-so-obvious [14].

One implication of our study is to investigate the degree to which a user would like to receive personalized old items, newer items, or both respectively. It is shown through the experiment that recommending newer item is more difficult (less accurate) than recommending older item. However, most of the ratings are for newer movies (cf. Table 3), which explain the fact that most people prefer to watch and rate new movies rather than old ones. Thus, improving recommendation techniques for the recommendation of newer movies would become more important.

6 Conclusions and Future Work

In this chapter, we investigate the temporal effect of items on the performance of the recommender systems, in the context of movie recommendation. In particular, we argue that movies' production year, sometimes reflecting the situational environment where the movies were filmed, could be used to scale down candidate set for recommendations. We perform some experiments on MovieLens data sets, and satisfactory results were obtained from our experiment. The experiment results show that the temporal feature of items can be exploited to scale down the candidate sets, without compromising the accuracy of the recommender systems.

We speculate that not all movies of all genres would have the same degree of sensitivity to the temporal effects. For example, certain category of movies, including romantic movies, would somehow survive this temporal selection. For example, many users, if they like the movie *Breakfast in Tiffany's*, released in 1961, they would probably like *Notting Hill*, released in 1999. In this case, the majority of users who like romantic movies believe that a good romantic movie is for all generations. Hence, it might not matter that whether we select recent romantic movies or not. As our future work, we will study different genres of movies' sensitivity to the temporal effects, and how would that in turn affect the recommendations made.

In addition, currently, the study reported here is conducted in the movie domain. It is apparently that the proposed approach can be generalized into many other domains, and it is especially desirable for online real-time recommendations in large-scale e-commerce systems.

Acknowledgments

The majority of this work was performed while the first author was still with Department of Computer Science at the University of Saskatchewan and was supported by NSERC. We would like to thank Pendopo Winoto for developing the testbed.

References

1. Adomavicius, G. and Tuzhilin, A. Extending recommender systems: a multidimensional approach. *Workship on Intelligent Techniques for Web Personalization, 17th International Joint Conference on Artificial Intelligence*, Seattle, Washington. (2001).
2. Balabanović, M. and Shoham, Y. Fab: content-based collaborative recommendation. *Communications of the ACM*, 40(3): 66-72. (1997).
3. Basu, C., Hirsh, H. and Cohen, W. W. Recommendation as classification: using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI/IAAI 1998)*, Madison. (1998) 714-720
4. Billsus, D. and Pazzani, M. A hybrid user model for news story classification. *In Proceedings of the 7th International Conference on User Modeling (UM'99).* (1999) 99-108
5. Breese, J., Heckerman, D. and Kadie, C. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI 1998).* (1998)
6. Good, N., Schafer, J. B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J. and Riedl, J. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI'99).* (1999) 439-446
7. Grant, S. and McCalla, G. I. A hybrid approach to making recommendations and its application to the movie domain. In *Proceedings of 2001 Canadian Artificial Intelligence Conference*. Stroulia, E. and Matwin, S.(eds) LNAI 2056. (2001). 257-266
8. Herlocker, J., Konstan, J., Borchers, A. and Riedl, J. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, Berkeley, USA. (1999) 230-237
9. Herlocker, J., Konstan, J., Loren G. Terveen, L. G. and John Riedl, J. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1): 5-53. January (2004)
10. Hill, W., Stead, L., Rosenstein, M. and Furnas, G. Recommending and evaluating choices in a virtual community of use. In *Proceedings of ACM Conference on Human Factors in computing Systems (ACM CSCW'95).* (1995) 194-201
11. Jameson, A., Konstan, J. and Riedl, J. AI techniques for personalized recommendation. Tutorial notes, *18th National Conference on Artificial Intelligence (AAAI-02)*, Edmonton, Canada. (2002)
12. Joachims, T., Freitag, D. and Mitchell, T. WebWatcher: a tour guide for the world wide web. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97).* (1997) 770-775
13. Konstan, J., Miller, B.N., Maltz, D., Herlocker, J., Gordon, L.R. and Riedl, J. GroupLens: applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3): 77-87, March (1997)

14. McNee, S.M., Albert, I., Cosley, D., Gopalkrishnan, P., Lam, S. K., Rashid, A. M., Konstan, J. and Riedl, J. On the recommending of citations for research papers. In *Proceedings of ACM Conference on Computer Supported Collaborative Work (CSCW'02)*. ACM New York. (2002)
15. Melville, P., Mooney, R. and Nagarajan, R. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-2002)*, Edmonton, Canada. (2002) 187-192
16. Paepcke, A., Garcia-Molina, H., Rodriguez-Mula, G. and Cho, J. Beyond document similarity: understanding value-based search and browsing technologies. *SIGMOD Records*, 29(1): 80-92, March (2000)
17. Popescul, A., Flake, G.W., Lawrence, S., Ungar, L. H. and Giles, C.L. Clustering and identifying temporal trends in document databases. In *Proceedings of the IEEE Advances in Digital Libraries (IEEE ADL 2000),* Washington, DC. May, (2000) 173-182
18. Resnick, P., Iacouvou, N., Suchak, M., Bergstrom, P. and Riedl, J. GroupLens: an open architecture for collaborative filtering of Netnews. In *Proceedings of ACM Computer-Supported Collaborative Work (CSCW'94)*, Chapel Hill, NC. Addison-Wesley. (1994) 175-186
19. Sarwar, B., Konstan, J., Borchers, A., Herlocker, J., Miller, B. and Riedl, J. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of ACM Conference on Computer Supported Collaborative Work (ACM CSCW'98),* Nov. (1998)
20. Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. Analysis of recommendation algorithms for e-commerce. In *Proceedings of ACM Conference on Electronic Commerce (ACM EC'2000),* Minneapolis. (2000) 158-167
21. Sarwar, B., Karypis, G., Konstan, J. and Riedl., J. Application dimensionality reduction in recommender system—a case study. In *WebKDD Web Mining for E-Commerce Workshop*, *ACM International Conference on Knowledge Discovery and Data Mining (KDD 2000)*. (2000)
22. Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. Item-based collaborative filtering recommender algorithms. In *Proceedings of ACM International World Wide Web Conference (WWW10),* Hong Kong. (2001)
23. Schein, A., Popescul, A., Ungar, L.H. and Pennock, D. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02),* Tampere, Finland, August. (2002)
24. Shardanand, U and Maes, P. Social information filtering: algorithms for automating 'word of mouth'. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, (ACM CHI'1995)*, Denver. (1995) 210-217
25. Sina, R and Swearinggen, K. The role of transparency in recommender systems. In *CHI 2002 Conference Companion*. (2002)
26. Swets, J.A. Measuring the accuracy of diagnostic systems. *Science* 240:1285-1289. (1988)
27. Ungar, L.H. and Foster, D.P. Clustering methods for collaborative filtering. In Workshop on Recommender Systems, *15th national Conference on Artificial Intelligence (AAAI-98).* (1998)
28. Yang, H, Parthasarathy, S and Reddy, S. On the use of constrained association rules for web mining. In *WEBKDD'2002---MiningWeb Data for Discovering Usage Patterns and Profiles*, Springer-Verlag, LNCS 2703. (2002) 100-118
29. Yu, K, Xu, X., Ester, M and Kriegel, H-P. Selecting relevant instances for efficient and accurate collaborative filtering. In *Proceedings of ACM International Conference on Information and Knowledge Management* (*CIKM'01)*. (2001) 239-246
30. Yu, P. Data mining and personalization technologies. In *Proceedings of the International Conference on Database Systems for Advanced Applications*. Taiwan. 1999.

Discovering Interesting Navigations on a Web Site Using SAM^I

Birgit Hay, Geert Wets, and Koen Vanhoof

Limburg University Centre, Faculty of Applied Economic Sciences,
B-3590 Diepenbeek, Belgium
{birgit.hay, geert.wets, koen.vanhoof}@luc.ac.be

Abstract. In this article, a new algorithm called Sequence Alignment Method extended with an Interestingness Measure (SAM^I) is illustrated for mining navigation patterns on a web site. Through log file analysis, SAMI distinguishes interesting patterns (i.e. unexpected, surprising patterns contradicting with the structure of the web site or direct hyperlinks between web pages) from uninteresting patterns (i.e. expected, known, obvious patterns resulting from the structure of the web site or direct hyperlinks between web pages) and provides information about the order of visited web pages. The algorithm is validated using real data sets of the Music Machines web site http://machines.hyperreal.org, home of musical electronics on the web. Empirical results show that SAMI identifies profiles of visiting behavior, which may be used for web personalization techniques and for optimizing the layout of the web site through structuring of page-links.

1 Introduction and Background

One of the fundamental elements in modern society is the World Wide Web (or Web), which creates a universal space of information that can be accessed by companies, government, universities, students, teachers, business people and other individuals. A web site represents a set of interconnected web pages on the Web and is developed and maintained by a person or organization. While web sites constitute a medium for communication, publicity and commerce, Web Mining studies discover and analyze useful information from the Web [1]. In [2], Web Mining is a common term for the application of data mining techniques to web access logs. Data Mining is the process of non-trivial extraction of implicit previously unknown and potentially useful information from data in large databases [2], [3].

In general, Web Mining covers three knowledge discovery domains: Web Content Mining, Web Structure Mining and Web Usage Mining [1], [2]. Web Content Mining is the process of extracting knowledge from the content of documents and their descriptions. Web Structure Mining is the process of inferring knowledge from the World Wide Web organization and links between pages in the Web. Finally, Web Usage Mining focuses on analyzing visiting information from logged data in order to extract previously unknown and interesting usage patterns [4]. In [5] Web Usage Mining is described as the application of data mining techniques on web access logs allowing management to optimize the site for the benefit of visitors. In this study, we will focus on Web Usage Mining.

Studies show that, for mining visiting behavior on web sites, different techniques are used. For example, in [6] a tool for real-time knowledge discovery from users' web page navigation called INSITE is presented. The system tracks users' navigation through a web site and discovers real-time, scalable and adaptive clustering of navigation paths. A role-based recommendation engine allows for the web site to react to the user in real-time with customized information e.g. in target advertisement. Another technique is described in [7] where a Web Utilization Miner (WUM) discovers interesting navigation patterns. The system consists of two modules. The Aggregation Service prepares the logged data for mining while the MINT-Processor performs the mining. MINT supports the specification of criteria of statistical, structural and textual nature. Also, an innovative aggregated storage representation for the information in the web server log is exploited by WUM.

Interesting patterns on a web site are unexpected, surprising patterns contradicting with the structure of the web site or direct hyperlinks between web pages. *Uninteresting patterns* on a web site are expected, known, obvious patterns resulting from the structure of the web site or direct hyperlinks between web pages.

As far as we know, no technique in Web Usage Mining studies exists that includes both a measure for distinguishing interesting patterns from uninteresting patterns and a measure for the order in which pages are visited within interesting patterns. However, mining navigation patterns based on interestingness and order of visited pages offers important information for the purpose of supporting and increasing customer satisfaction. For example, optimizing the layout of the web site through structuring of page-links. Therefore, we will concentrate in this study on a measure for Web Usage Mining that discovers knowledge about interesting navigations representing also structural information (or the order in which pages are visited). To this end, we will introduce Sequence Alignment Method extended with an Interestingness Measure (SAM^{I}).

The focus of our paper is concentrated on developing and applying SAM^{I}, rather than examining the various issues with regard to pre-processing server log data (i.e. application of specialized algorithms for sessionizing, identification of users by means of cookie registration etc.).

The article is organized as follows. First, Sequence Alignment Method (SAM) is described. Then, an Interestingness Measure, based on Baldwin's support logic [8] as well as a support logic framework for Web Usage Mining [9], is explained. In section 4, the algorithm of SAM^{I} is described. In section 5, SAM^{I} is applied to a real data set storing usage behavior on the web site http://machines.hyperreal.org. Finally, in section 6, conclusions and topics for future research are given.

2 Sequence Alignment Method (SAM)

SAM is explained and illustrated in [10], [11], [12]. In this section we give a short overview of the algorithm.

SAM is a distance (or similarity) measure between sequences reflecting the amount of work that needs to be done to convert one sequence into the other. The higher/lower SAM distance, the more/less effort it takes to equalize the sequences.

The amount of work or effort is expressed by the following operations: insertion, deletion and reordering. Insertion and deletion operations are applied to unique elements; reordering operations are applied to common elements. *Common elements* appear in both of the compared sequences whereas *unique elements* appear in either one of them. Furthermore, *insertion* adds an element into the source (first) sequence; *deletion* removes an element from the source (first) sequence. Moreover, *reordering* changes the order of elements.

In particular, the SAM distance measure between two sequences S_1 and S_2 is calculated using the following formula [13], [14]:

$$d_{SAM}(S_1, S_2) = \min[\,(w_d D + w_i I) + \eta R\,]\,. \tag{1}$$

where

d_{SAM} is the distance between two sequences S_1 and S_2, based on SAM;
w_d is the weight value for the deletion operations, a positive constant not equal to 0, determined by the researcher ($w_d > 0$);
w_i is the weight value for the insertion operations, a positive constant not equal to 0, determined by the researcher ($w_i > 0$);
D is the number of deletion operations;
I is the number of insertion operations;
R is the number of reordering operations;
η is the reordering weight, a positive constant not equal to 0, determined by the researcher ($\eta > 0$);

Equation (1) indicates that the score between two sequences, represented by SAM, consists of the minimum of costs for deleting and inserting unique elements and reordering common elements.

To illustrate SAM, consider the following example. Sequences s_1 and s_2 represent sequentially ordered visited pages (each element or page within the sequence is represented by an identification number) on a web site.

Suppose: $w_d = w_i = 1$ and $\eta = w_d + w_i$
s_1 {1, 4, 7, 8}
s_2 {4, 2, 3, 1, 5}

First, common elements, which do not occur in the same order, are reordered. In s_1, page (or element) 4 must precede page (or element) 1 so as to be equal to the order in s_2. The result of this reordering operation is:

s_1 {4, 1, 7, 8}
s_2 {4, 2, 3, 1, 5}

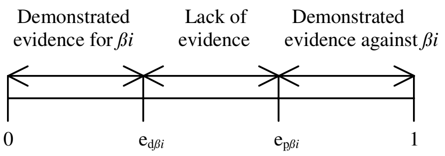
Then, unique elements in s_1 (i.e. pages 7 and 8) are deleted from s_1 while unique elements in s_2 (i.e. pages 2, 3 and 5) are inserted into s_1. Insertion in s_1 is done following the order of pages in s_2. Finally, 1 reordering, 2 deletions and 3 insertions is the amount of work done to equalize s_1 to s_2, resulting in a SAM distance measure $d_{SAM}(s_1, s_2) = 7$.

3 Interestingness Measure

The support logic framework, used within SAM^{I}, starts with the principles of Baldwin's support logic [8]. The framework is constructed with beliefs for Web Usage Mining [9].

3.1 Baldwin's Support Logic

Baldwin's support logic [8] values each piece of information, also called *belief*, by the *evidence for* and *evidence against*. For each such type of evidence, two kinds of evidence definitions exist. *Demonstrated evidence* is evidence that is proven or shown by the data and known by the researcher. *Possible evidence* is evidence that is not proven by the data. The researcher may have an idea about the existence of such evidence but it is not known for sure.


Demonstrated evidence for βi
Lack of evidence
Demonstrated evidence against βi
0
$e_{d\beta i}$
$e_{p\beta i}$
1

where
βi = belief i with $i = 1, 2, \ldots$ B;
B = total number of beliefs;
$e_{d\beta i}$ = demonstrated evidence for, in support of, βi;
$e_{p\beta i}$ = possible evidence for, in support of, βi;
$(1-e_{d\beta i})$ = possible evidence against βi;
$(1-e_{p\beta i})$ = demonstrated evidence against βi;
$(e_{p\beta i}-e_{d\beta i})$ = lack of evidence for or against βi;
$[e_{d\beta i}, e_{p\beta i}]$= evidence pair of βi;
$e_{d\beta i} \geq 0$; $e_{p\beta i} \geq 0$; $e_{d\beta i} + (1-e_{p\beta i}) \leq 1$;

Fig. 1. Conceptual frame of evidence

Figure 1 illustrates the conceptual frame of evidence [8], [9]. For each belief βi, demonstrated evidence $e_{d\beta i}$ and possible evidence $e_{p\beta i}$ is represented by the evidence pair $[e_{d\beta i}, e_{p\beta i}]$. Furthermore, possible evidence against βi, demonstrated evidence against βi and lack of evidence with regard to βi are represented in the framework by respectively $(1-e_{d\beta i})$, $(1-e_{p\beta i})$ and $(e_{p\beta i}-e_{d\beta i})$. Demonstrated as well as possible evidence must be nonnegative. Finally, summing demonstrated evidence supporting βi with demonstrated evidence against βi must not be greater than one.

Following Baldwin's support logic programming [8], for every belief βi, evidence pairs $[e_{d\beta i}^{1}, e_{p\beta i}^{1}]$ and $[e_{d\beta i}^{2}, e_{p\beta i}^{2}]$, coming from two different sources, are combined into one evidence pair $[e_{d\beta i}^{c}, e_{p\beta i}^{c}]$ as follows:

$$K = 1 - e_{d\beta i}^{1}\,(1 - e_{p\beta i}^{2}) - e_{d\beta i}^{2}(1 - e_{p\beta i}^{1}) \,. \tag{2}$$

$$e_{d\beta i}^{c} = [e_{d\beta i}^{1}\, e_{d\beta i}^{2} + e_{d\beta i}^{1}\,(e_{p\beta i}^{2} - e_{d\beta i}^{2}) + e_{d\beta i}^{2}\,(e_{p\beta i}^{1} - e_{d\beta i}^{1})] / \, K \,. \tag{3}$$

$$e_{p\beta i}{}^{c} = -[[(1-e_{p\beta i}{}^{1})(1-e_{p\beta i}{}^{2}) + (e_{p\beta i}{}^{1}-e_{d\beta i}{}^{1})(1-e_{p\beta i}{}^{2}) + (e_{p\beta i}{}^{2}-e_{d\beta i}{}^{2})(1-e_{p\beta i}{}^{1})]/K] + 1. \quad (4)$$

Interesting results are defined as either a belief with a combined evidence pair that is significantly different from (conflicting with) one of the original evidence pairs, or original evidence pairs that are significantly different from (conflicting among) each other. *Significantly different* is determined by setting a threshold value τ for the differences between the evidence pairs. Ultimately, a belief *ßi* is interesting if:

$$\tau \leq IM_{\beta i}\,. \quad (5)$$

where
$IM_{\beta i} = \sqrt{(e_{d\beta i})^2 + (e_{p\beta i})^2}$ = interestingness measure for *ßi*;
$e_{d\beta i} = |e_{d\beta i}{}^{1} - e_{d\beta i}{}^{2}|$;
$e_{p\beta i} = |e_{p\beta i}{}^{1} - e_{p\beta i}{}^{2}|$;

3.2 Support Logic Framework for Web Usage Mining

In order to define which beliefs are interesting and which are not, we will use the two different sources of data. Structure data provide for each belief *ßi structure evidence* [$e_{d\beta i}{}^{s}$, $e_{p\beta i}{}^{s}$]. Usage data provide for each belief *ßi usage evidence* [$e_{d\beta i}{}^{u}$, $e_{p\beta i}{}^{u}$]. Evidence is calculated for pages being related on a web site. A belief *ßi* is interesting if the difference between its structure and usage evidence pairs $\geq \tau$ or if the difference between its structure (usage) and combined evidence pairs $\geq \tau$. We may also say that a belief *ßi* is interesting if $IM_{\beta i} \geq \tau$, following equation (5).

Calculating Structure Evidence. In [9], a method for automatically calculating structure evidence pairs for beliefs of related web pages is given. Two factors define $e_{d\beta i}{}^{s}$. The *link factor* (lfactor) is a normalized measure for the number of links present among the pages of an item set. The *connectivity factor* (cfactor) is a measure for the strength of the topological connection among the pages in an item set. Structure evidence for a belief *ßi* is defined as follows:

$$e_{d\beta i}{}^{s} = \text{lfactor x cfactor}\,. \quad (6)$$

where
lfactor = L / [P (P-1)];
P = total number of pages in the item set;
L = number of direct hyperlinks between the pages in the item set;
cfactor = 1 if the graphical presentation for the pages in the item set is connected, which means that minimum one direct hyperlink must exist between every pair of pages in the item set;
cfactor = 0 otherwise;

$e_{p\beta i}{}^{s}$ may be set anywhere between $e_{d\beta i}{}^{s}$ and 1, depending on the degree of lack of evidence . (7)

Calculating Usage Evidence. In [9], mined results from server session analyses, in the form of *frequent item sets*, representing frequently visited pages, are used to provide usage evidence for pages being related. Two measures are calculated for frequent item sets. *Support* (s) calculates the fraction of transactions that contain all of the items in the item set while *coverage* (c) measures the fraction of transactions that contain at least one of the items in the item set.

$$s = \text{count}\ (i_1 \wedge i_2 \ \dots \wedge i_P)\ / N . \tag{8}$$

$$c = \text{count}\ (i_1 \vee i_2 \ \dots \vee i_P)\ / N . \tag{9}$$

where

count (predicate) is the number of transactions containing the predicate;
i is a web page in the item set;
P is the total number of pages in the item set;
N is the total number of transactions or server sessions;

Note that support and coverage are both highly dependent on the total number of transactions. By taking the ratio of *support-to-coverage* (SCR), this dependency is eliminated. SCR gives a single measure of the strength of a frequent item set independent of the total number of transactions in the data set. Finally, $e_{d\beta i}^{u}$ is calculated as follows:

$$e_{d\beta i}^{u} = \text{SCR} . \tag{10}$$

where
SCR = s / c;

$$e_{p\beta i}^{u} \text{ may be set anywhere between } e_{d\beta i}^{u} \text{ and } 1, \text{ depending on the degree of lack of evidence} . \tag{11}$$

Combining Structure and Usage Evidence. [9] noticed the problem of scaling when combining structure and usage evidence into the support logic framework. Since the two sets of evidence are derived in different manners from different data sets, the scales do not necessarily match. This means that, if the number of related pages in a belief increases, the less likely it is that a corresponding frequent item set will be discovered. To deal with this, [9] scales usage evidence based on the number of pages in the item set:

$$e_{d\beta i}^{u} = \text{SCR x sfactor} . \tag{12}$$

where
sfactor = number of pages in the item set;

4 Sequence Alignment Method Extended with Interestingness (SAM^I)

Beliefs of related pages that are declared interesting are integrated into the SAM algorithm. Hence, interesting visiting patterns, providing order based information, are automatically discovered. In particular, SAM^I distance between two sequences S_1 and S_2 is calculated using the following formula:

$$d_{SAM}^{I} (S_1, S_2) = \min\ [(w_d D^I + w_i I^I) + \eta\ R^I . \tag{13}$$

where

$d_{SAM}{}^I$	is the similarity or distance for interesting pages between two sequences S_1 and S_2, based on SAM;
w_d	is the weight value for the deletion operations, a positive constant not equal to 0, determined by the researcher ($w_d > 0$);
w_i	is the weight value for the insertion operations, a positive constant not equal to 0, determined by the researcher ($w_i > 0$);
D^I	is the number of deletion operations for interesting pages;
I^I	is the number of insertion operations for interesting pages;
R^I	is the number of reordering operations for interesting pages;
η	is the reordering weight, a positive constant not equal to 0, determined by the researcher ($\eta > 0$);

Equation (13) indicates that the score between two sequences, represented by SAM^I, consists of the minimum of costs for deleting and inserting unique interesting elements and reordering common interesting elements.

Table 1. Sequence comparison based on SAM^I

Interesting beliefs of related pages	Source sequence: $s_1 = 2, 4, 5, 1$	$w_i = 1$ $w_d = 1$ $\eta = 2$	Source sequence based on int. bel. of related pages: $s_1 = 2, 1$
(1, 2)	Target sequence:		Target sequence
(7, 8)	$s_2 = 1, 3, 1, 8, 2$	$d_{SAM}{}^I$	based on int. bel. of
(2, 8)		(s_1, s_2)	related pages:
(1, 2, 3)		$= 5$	$s_2 = 1, 3, 1, 8, 2$

To give a clear understanding of how SAM^I works, the algorithm is illustrated with an example in table 1. The interesting beliefs of related pages, or interesting frequent item sets, discovered by the support logic framework for Web Usage Mining, are given in the first column. We remind that the order in which elements occur in frequent item sets is irrelevant. The second column presents two sequences s_1 and s_2 representing server sessions holding interesting and uninteresting combinations of pages. In the third column, SAM^I between s_1 and s_2 is presented. Finally, in the last column, the original source and target sequences s_1 and s_2 are changed into sequences holding only interesting combinations of pages, respecting the order in which pages occur. Combinations of pages that are not interesting are filtered out of the sequences.

5 Empirical Analysis

5.1 Proposed Approach

The empirical analysis reported in this article concerns the question whether interesting navigations providing structural information (sequential relationships or order of visited pages), embedded in web-clickstream data, are well reflected by SAM^I. Hence, we propose the following approach.

Calculate SAM^I Distances. First pair wise SAM^I distances are calculated between server sessions. A *server session* represents the click stream of page views for a single user to a web site. SAM^I uses its most common parameters (i.e. $w_d = w_i = 1$; $\eta = 2$).

Apply Clustering on Distance Matrix. Second, the pair wise SAM^I distance measures are inserted into a distance matrix and Ward hierarchical clustering is invoked on the matrix. In order to define the best solution for the number of clusters, a consensus among the following criteria is used. *R-squared* is used as a goodness-of-fit measure during clustering processing and equals the proportion of variation explained by the model. Furthermore, [15] have compared thirty methods for estimating the number of clusters using hierarchical clustering methods. The criteria that performed best in these simulation studies were *pseudo F statistic (PSF)* and *T-squared statistic (TST)*. Relatively large values given by the PSF indicate a stopping point. A general rule for interpreting the values of TST is to move towards joining of clusters and find values markedly larger than previous values. Finally, another method for judging the number of clusters in a data set is the *root mean squared standard deviation (RMSSTD),* which provides a measure of homogeneity for the cluster solution. The smaller this value, the more homogeneous are the data.

Calculate Support and Confidence. Third, for each cluster, support and confidence are calculated for every combination of the order of pages within interesting frequent item sets. Here, support and confidence take into account the order of pages. *Support* is specified as the fraction of server sessions within a cluster presenting the interesting order based frequent item set. *Confidence* expresses the probability that, if a server sessions in a cluster contains all but the last page (in respective order) of the order based interesting frequent item set, the server session will also hold the last page. For each cluster, the five highest support values are used for graphically depicting interesting navigations.

5.2 Data

For this application, log files registering visiting behavior from 01/02/1999 till 28/02/1999 on the web site http://machines.hyperreal.org are analyzed. After pre-processing the data using the method described in [10], [11] a total number of 75,855 server sessions, showing navigations through web pages with 1,159 different logged URL addresses, are identified. In a preceding step, before the actual SAM^I application took place, 539 beliefs of related web pages (or frequent item sets), consisting minimum of 2 and maximum of 4 related pages (or items), with a minimum support of 0.1% are defined from the usage data. For each belief, usage, structure and combined evidence pairs are calculated using equations (2) to (4) and (6) to (12). Note that no lack of evidence is tolerated in the analysis, which means that demonstrated evidence always equals possible evidence. An interestingness threshold value of $\tau = 0.75$ in equation (5) is used to filter out interesting beliefs of related pages. By setting the value of τ very high, related pages of the highest interest are discovered. Usually, a τ-value of 0.5 is satisfactory [9]. From the analysis, 91 beliefs of related pages were declared interesting.

The analysis starts with applying SAM^I, taking into account the 91 interesting beliefs of related pages, to the server sessions. While calculating SAM^I distance measures, the original number of server sessions is reduced to 7,266, due to the fact that SAM^I selectively aligns sequences based on interesting beliefs of related pages. This means that 68,589 server sessions do not hold interesting beliefs of related pages and therefore, are not considered for further analysis.

From the data, 4 clusters are defined, following the criteria for defining the number of clusters. For each cluster, support and confidence are calculated for every combination of the order of pages within interesting beliefs of related pages. Out of a total number of 91 interesting beliefs of related pages, 278 different combinations of the order of pages are defined. For each cluster, the 5 highest support values are used for presenting interesting navigations.

5.3 Results

Graphs. Figures 2 and 3 present interesting navigations providing information about the order of visited pages on the web site http://machines.hyperreal.org. Parts of the structure of the web site, along with direct hyperlinks between pages are graphically depicted in each cluster. For each page, the page_id is given along with (a part of) the URL address of this particular page, which is written under the page_id inside the rectangle. The complete URL address of each page can be read taking into account the level in the web site structure and the links. For example, page 657 constitutes the main page with URL address http://machines.hyperreal.org. Going one level downwards, 4 different web pages appear. The complete URL address of page 349 is http://machines.hyperreal.org/manufacturers. Proceeding towards, for example, page 868, the URL address http://machines.hyperreal.org/manufacturers/ARP/Odyssey is given. The dashed rectangles originated from different logged URL addresses in the files. However, the content of the web page appears to be exactly the same as the one given by the solid rectangles. Further analysis revealed that the log files also stored information of people who used the URL address http://www.hyperreal.org and navigations from this main page on. For example, page 159 appears to be exactly the same as page 815. The only difference is that page 159 is navigated through http://www.hyperreal.org/guide and page 815 is navigated through http://machines.hyperreal.org/guide. We would like to keep this distinction in our analysis because these navigations are considered interesting. Links between pages are drawn by *thin black solid arrows*, while interesting navigations, including order based information, are given by the *bigger dashed arrows*. For example, from page 657, people can go to pages 349, 815, 810, 813 and from each of these pages a link points back to the home page. Also, from page 349 other pages may be visited like 857, 984, 882, 1082 as well as 657, 815 and 810.

Support (s) and confidence (c) values are written next to, above or under the arrows of the interesting navigations. For example, in cluster 1, 22.29% of the server sessions visited page 163 before page 349. The confidence value indicates that, if people visit page 163, the chance that they will visit page 349 thereafter is 67.27%. For evaluation purposes, distribution of server sessions is given in the upper left corner of every cluster. For example, 18.40% (i.e. 1,337 out of 7,266) of the server sessions are grouped in cluster 1.

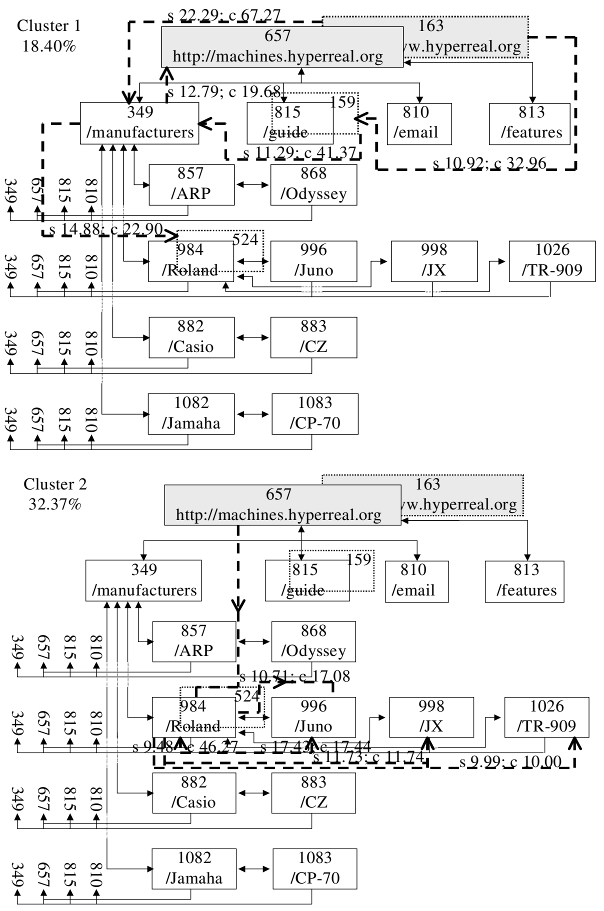


Fig. 2. Cluster 1 and 2: Interesting navigations on http://machines.hyperreal.org

In order to avoid complex drawings of arrows making figure 2 unclear, some modifications are made. First, with regard to the links between pages, some arrows point towards a particular page_id. For example, from pages 857, 984, 882, 1082 one may proceed to pages 657, 815 and 810. Likewise, from pages 868, 996, 998, 1026, 883, 1083 one may proceed to pages 349, 657, 815 and 810. Second, the dashed parts of the links indicate that there is no intersection with other links.

If there were no dashed parts, the links could be misinterpreted, saying, for example, that from page 984 a link points to page 882. Third, with regard to the presentation of interesting navigations, lines showing arrows in the middle of navigations, instead of at the beginning or at the end, may appear. For example, in cluster 2, when navigating from page 657 to page 984 and from page 984 to page 996, somewhere in the middle of both navigations, an arrow is drawn. These arrows are used for interpreting order based interesting beliefs of related pages with more than 2

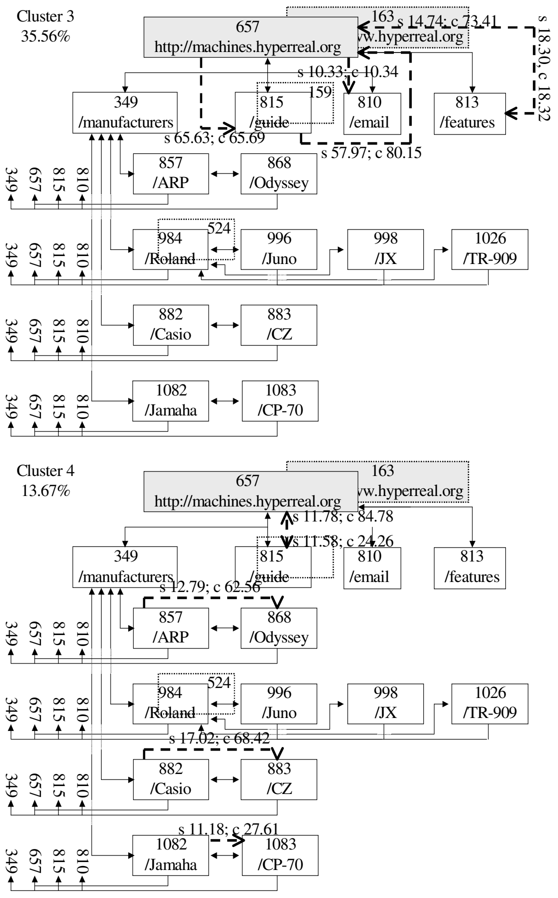


Fig. 3. Cluster 3 and 4: Interesting navigations on http://machines.hyperreal.org

pages. Support and confidence values are given next to or above the arrow of the last navigation. For example, in cluster 2, an interesting navigation appears in the following order: 657, 984, 996 with support and confidence values of 10.71% and 17.08%. Fourth, with regard to the magnitude of the structured web site with interesting related pages, for each cluster, only part of the site is given that is relevant for describing the interesting navigations.

Clusters. The empirical results provide 4 clusters, showing profiles of interesting navigations. Cluster 1 mainly represents navigations to and from the *'manufacturers'* page. Clusters 2 and 3 represent navigations that are concentrated around the *'Roland'* and *'home'* page respectively. In cluster 4, interesting navigations to several pages are presented: *'guide'*, *'ARP'*, *'Odyssey'*, *'Casio'*, *'CZ'*, *'Jamaha'* and *'CP-70'*.

The clusters are well separated with regard to the order of visited pages since all of the support values of the navigations in figure 2 are below 1% for other clusters. For example, cluster 1 represents navigating from page 163 to 349. The support for this navigation in clusters 2, 3 and 4 is respectively 0.17%, 0.19% and 0.00%. One exception is made for navigating from page 657 to 815 and from page 815 back to 657. These navigations are strongly related with other interesting navigations in cluster 3 and 4. This means that server sessions in cluster 3, holding navigations 657, 815 and 815, 657, also hold navigations 657, 810; 657, 813 and 813, 657. Server sessions in cluster 4, holding navigations 657, 815 and 815, 657, also hold navigations 857, 868; 882, 883 and 1082, 1083.

5.4 Deploying the Results

Interesting navigations may provide useful information for link optimization studies. In order to develop a web site structure conform to visiting behavior of users, links between pages that are not optimally used may be deleted or pages may be moved elsewhere in the structure of the web site. Given the analysis of web usage behavior on http://machines.hyperreal.org, clustering server sessions based on SAM^{I} discovered interesting navigations providing order based information of visited pages that are used together *less* then would be expected from the structure of the web site. Some suggestions for improving the structure of the web site, based on figure 2, along with usage, structure and combined evidence, are given in table 2. Due to lack of space, we were not able to give a list of all the suggestions. In the last column, the Interestingness Measure (IM) is given. This measure may give an indication of the 'urgency' of reacting to the behavior of web users. The higher IM, the more urgent it is to respond to visiting behavior by optimizing the structure of the web site. For example, navigating from page 349 to 657 occurs less frequent as expected. Therefore, the direct hyperlink from page 349 to 657 may be deleted. An IM of 1.39 notifies that reaction to this behavior is most urgent.

Table 2. Suggestions for reorganizing pages or deleting direct links

From	To	Usage evidence	Structure evidence	Combined evidence	IM
349 http://machines.hyperreal.org /manufacturers	657 http://machines.hyperreal.org	[0.0167; 0.0167]	[1.000; 1.0000]	[1.0000; 1.0000]	1.3905
657 http://machines.hyperreal.org	813 http://machines.hyperreal.org /features	[0.0345; 0.0345]	[1.0000; 1.0000]	[1.0000; 1.0000]	1.3654
813 http://machines.hyperreal.org /features	657 http://machines.hyperreal.org	[0.0345; 0.0345]	[1.0000; 1.0000]	[1.0000; 1.0000]	1.3654
163 http://www.hyperreal.org	159 http://www.hyperreal.org /guide	[0.0748; 0.0748]	[1.0000; 1.0000]	[1.0000; 1.0000]	1.3084
…	…	…	…	…	…
1082 http://machines.hyperreal.org /manufacturers/Jamaha	1083 http://machines.hyperreal.org /manufacturers/Jamaha CP-70	[0.0995; 0.0995]	[1.0000; 1.0000]	[1.0000; 1.0000]	1.2734
984 http://machines.hyperreal.org /manufacturers/Roland	1026 http://machines.hyperreal.org /manufacturers/Roland /TR-909	[0.1151; 0.1151]	[1.0000; 1.0000]	[1.0000; 1.0000]	1.2514
…	…	…	…	…	…

6 Conclusions and Future Research

In this article, SAM is extended with an Interestingness Measure (SAM^I) to discover interesting navigation patterns, providing information about the order of visited pages on a web site. Navigations are defined interesting if they are unexpected, surprising or contradicting with the structure of the web site or direct hyperlinks between web pages. The new algorithm SAM^I is tested on a real data set of web usage data on http://machines.hyperreal.org and discovered interesting navigations that are used together *less* frequent than would be expected from the structure of the web site. This indicates that links between web pages are not optimally used and suggestions for reorganizing pages or deleting direct hyperlinks on http://machines.hyperreal.org may be given, along with an 'urgency' measure. Finally, given the results of our study, we may conclude that interesting navigations providing structural information (sequential relationships or order of visited pages), embedded in web-clickstream data, are well reflected by SAM^I.

Topics for future research should include sensitivity into IM with regard to the 'depth' of pages in the web site structure. Studies must verify whether the a-priori probability of finding related pages which are situated 'deep' in the web site structure is smaller then the probability of finding related pages which are situated at the 'top' in the web site structure. Finally, instead of using two values for the cfactor, indicating whether the graphical presentation for the items in the item set is connected or not, it might be a good idea to use a range of values for the cfactor (i.e. $0 \leq$ cfactor ≤ 1), indicating how strong the graphical presentation for the items in the item set is connected.

References

1. Cooley, R., Mobasher, B., Srivastava, J.: Web Mining: Information and Pattern Discovery on the World Wide Web. A survey paper. In: Proc. ICTAI-97 (1997)
2. Zaïane, O.R.: Conference Tutorial Notes: Web Mining: Concepts, Practices and Research. In: Proc. SDBD-2000. (2001) 410-474
3. Piatetsky-Shapiro, G., Fayyad, U., Smith, P.: From data mining to knowledge discovery: An overview. In: Fayyad, U., Piatetsky-Shapiro, G., Smith, P., Uthurusamy, R. (eds): Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press. (1996) 1-35
4. Cooley, R., Mobasher, B., Srivastava, J.: Data Preparation for Mining World Wide Web Browsing Patterns. Knowledge and Information Systems, Vol. 1(1). (1999a) 5-32
5. Foss, A., Weinan, W., Zaïane, O.R.: A Non-Parametric Approach to Web Log Analysis. In: Proc. Workshop on Web Mining SDM-01 (2001) 41-50
6. Shahabi, C., Faisal, A., Kashani, F.B., Faruque, J.: INSITE: A Tool for interpreting Users? Interaction with a Web Space. In: Proc. VLDB-00. (2000) 635-638
7. Spiliopoulou, M., Faulstich, L.: WUM: a Tool for Web Utilization Analysis. In: Proc. Workshop WebDB-98, Extended version in LNCS. Vol. 1590. (1998) 84-103
8. Baldwin, J.F.: Evidential support logic programming. Fuzzy sets and systems, Vol. 24(1). (1987) 1-26
9. Cooley, R., Tan, P.-N., Srivastava, J.: Discovery of interesting usage patterns from web data. Technical Report TR 99-022 University of Minnesota (1999b)

10. Hay, B., Wets, G., Vanhoof, K.: Web Usage Mining by means of Multidimensional Sequence Alignment Methods. In: Proc. Workshop WEBKDD-01 (2002) 44-52
11. Hay, B., Wets, G., Vanhoof, K.: Mining Navigation Patterns using a Sequence Alignment Method. Knowledge and Information Systems, in press (2003a)
12. Hay, B., Wets, G., Vanhoof, K.: Segmentation of visiting patterns on web sites using a Sequence Alignment Method. Journal of Retailing and Consumer Services in press (2003b)
13. Joh, C.H., Arentze, T.A., Timmermans, H.J.P.: A position-sensitive sequence alignment method illustrated for space-time activity diary data. Environment and Planning A, Vol. 33(2). (2001) 313-338
14. Sankoff, D., Kruskal, J.B. (eds.): Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparison. Addison-Wesley, Reading, MA (1983)
15. Cooper, M.C., Milligan, G.W.: The effect of error on determining the number of clusters. In: Proc.Workshop on Data Analysis, Decision Support and Expert Knowledge Representation in Marketing and Related Areas of Research (1988) 319-328

Personalisation of Web Search

Kevin Keenoy and Mark Levene

School of Computer Science and Information Systems,
Birkbeck, University of London
{kevin, mark}@dcs.bbk.ac.uk

Abstract. The availability of web search has revolutionised the way people discover information, yet as search services maintain larger and larger indexes they are in danger of becoming a victim of their own success. Many common searches can return a vast number of web pages, many of which will be irrelevant to the searcher, and of which only about ten or twenty of the top-ranked results will be browsed. The problem is that while pages returned by a search may be relevant to the keywords entered, the keywords generally give only a partial expression of the searcher's information need. Personalised web search takes keywords from the user as an expression of their information need, but also uses additional information about the user (such as their preferences, community, location or history) to assist in determining the relevance of pages.

There are many approaches to providing personalised web search, each with the aim of returning the results most relevant to the user ranked highest. The features that distinguish the approaches are the kind of information about the user that is used, the level of interaction with the user (explicit or implicit collection of data), how the information is stored (client-side or server-side), the algorithm used to incorporate the information about the user into the search and how information is presented to the user (mobile devices present some unique challenges in this respect). Some of these personalisation methods stem from techniques previously used in traditional information retrieval, whilst others are unique to the web environment.

This chapter describes the many techniques that have been applied to adapt the web search process to the individual user. We also present a novel system that we are developing, which uses a client-side user profile to provide a personalised ranking of results from multiple search portals. We conclude with a brief consideration of the future of personalised search and how it may affect the development of the web.

1 Introduction

Current web search engines are very successful and are increasingly big business, with much of the revenue coming from advertising related to the user query. The algorithms used to generate search results are very good at discovering pages relevant to a (typically keyword-based) query, but in doing so they do not consider the *user* who submitted the query. Relevance is hard to measure on the web (the series of TREC Text Retrieval conferences has devised useful measures of relevance for the evaluation of retrieval from various fixed corpora, but is yet to produce anything that is truly representative of performance on the vast and ever-changing web). However, if users are satisfied with the web search service they are using we can infer that the results they are getting are somewhat relevant to their information need. Hence, the level of user satisfaction with a search service can be taken as a rough measure of the relevance of the results provided. It is our belief that the biggest increases in the relevance of web search results to users will come not by fine-tuning existing search algorithms nor from defining completely new algorithms based on web content and structure alone – the biggest increase in user satisfaction will come when the results returned are tailored to the individual as well as to the question asked. Search systems will know who the person is and what their particular preferences and idiosyncracies are, and this will enable them to better provide the user with what they were looking for.

Web search personalisation is still in its infancy – a recent survey of sixty publicly available search engines on the web found "little or no personalization of search functions, an expected characteristic of search engine personalisation" [35], and we have not encountered much published academic work specifically about personalising web search. However, work in closely related domains can provide insight into some of the techniques that could be applied to personalisation of web search. We will therefore also consider other work on web personalisation, not specifically related to search – things such as web browsing assistants and web-page and news recommender systems. Such systems are not, strictly speaking, personalising web search, but we believe that some of the methods developed in these areas could be applied to search as well.

The techniques used for personalisation of results mainly take one of three approaches, each of which has its own potential drawbacks: *re-ranking*, the re-ordering of search results to tailor them to the user (note the problem of needing good quality results to re-rank); *filtering*, the removal of results deemed to be irrelevant to the user (note the problem of potentially excluding relevant results all together); and *query expansion/modification*, the augmentation of the user's keyword-based query (note the problem of getting irrelevant results unless it is done very carefully).

Real-world systems that claim to be doing personalisation are often actually offering what we would call customisation – the ability for users to build profiles of preferences for the content they want to see and the layout it should be displayed in, with users typically choosing from a set number of possibilities. For example, My Yahoo! [46] allows users to specify which news, stock prices, weather and sports scores they want displayed on their My Yahoo! web pages, and these preferences are stored in a profile that is used to create the pages each time the user visits. This is what we call customisation rather than personalisation (although see Section 4.6 for details of some genuine personalisation done by My Yahoo!, which [35] found to be the service currently providing the most extensive personalisation).

Another approach to improving the user's search experience is to try to improve upon the standard ranked-list presentation of results. The presentation of search results to users can be almost as important in helping the user quickly find what they want as the actual pages retrieved, and techniques that present results in a format other than an ordered list can be useful. Result pages can be clustered into groups of similar pages and the clusters labelled to help users quickly identify which groups of pages they may be interested in. The meta-search engine Grouper [70] takes this approach, clustering the results from several different search engines, as does the Vivísimo [62] search engine. The Scout [39] meta-search system offers labelled clusters of results as one of four presentation options (the others being a plain list, a re-ranked list and an "index-based" representation).

The main distinguishing factor between different clustering interfaces is the quality of the labels provided for the clusters. Labels accurately describing the content of clusters can make it much easier for the user to pinpoint the pages they will be interested in, but the danger is that the labels generated give an incomplete or misleading picture of the pages within the cluster.

While the clustering of web search results is based solely on the content of the pages returned (or in some cases just their summaries), user interfaces can be made to adapt to the individual. Liu et al. [44] describe how an adaptive user interface (AUI) can monitor the user's interactions with almost any software application to identify common patterns of behaviour (that they call "episodes"). The AUI can then try to recognise when the user is about to enact one of these episodes and adaptively assist the user in carrying it out (e.g. by semi-automating the sequence of actions). They have realised an AUI in their Personalized Word Assistant (PWA), which can automate the input of repeated phrases and text-formatting tasks in Microsoft Word. Automatic user interface adaptation is an area with much to offer in terms of increasing the usability of systems, and hence user satisfaction, especially with the increasing use of hand-held computing devices with limited screen space. Although the ranked list of search results is an effective mode of presentation, we think there is certainly scope for the development of adaptive interfaces to web search. See [63] for a discussion on the customisation and personalisation of user interfaces, and a review of some of the work that has been done in this area.

2 Web Information Retrieval

Web information retrieval is a technical term describing what a web search engine does. "Traditional" information retrieval (IR) addresses the problem of determining the relevant documents from a (usually fixed) collection of electronic documents, based on a query presented by the user. Web IR extends this field by taking the web as its document collection. The web's peculiarities provide both challenges (because of it's scale and mutability) and opportunities for innovation in IR techniques (such as exploitation of the link structure of the web).

The traditional method used for IR is to build an index of the document collection (in the case of web search this is the entire web) and use this to look up the documents that include keywords submitted as a query. This is a purely content-based IR method – retrieval is based solely on the content of the documents in the collection and does not consider the users of the system or any inter-document structure (such as hyperlinks) during the retrieval process. The most commonly used model to represent the document space (especially among web search engines) is the *vector space model* [4], which represents each document as a vector of term weights with the weight (i.e. importance) of a term within a document being determined by metrics such as *Term Frequency-Inverse Document Frequency* (TF-IDF) [57]. A similar vector is created to represent the query terms submitted, and a "score" for each document is generated by calculating the similarity of the document vector to the query vector. The cosine of the angle between the two vectors is often used to generate the similarity score, as the closer together the two vectors are the more likely they are to contain the same terms.

The effectiveness of an IR system is frequently measured using the *precision* and *recall* performance measures [4]. The precision of a set of results is the ratio of the number of relevant documents retrieved to the total number of documents retrieved, and is a measure of how well the system retrieves only relevant results. Recall is the ratio of the number of relevant documents retrieved to the total number of relevant documents in the corpus, and is a measure of the exhaustiveness of the results. On the web recall is less important than precision, as users are not so worried about retrieving all documents relevant to their query, but do want high precision in the top ranking results.

The traditional IR process relies solely on the content of the documents (i.e. the keywords they contain) to decide the relevance of a page to a query. In hypertextual environments such as the web, additional metrics can be introduced that are based on the link structure between pages. The most well known of these are Kleinberg's hub's and authorities, which is also known as Hyperlinked Induced Topic Search (HITS) [37], and PageRank, which is one of the important components of the retrieval algorithm used by Google [9]. Link-based scoring metrics assume that links are included in a web page when the author considers the linked page to be of high quality. Hence, pages that are linked to by many other pages are more likely to contain high quality information than pages with few inlinks. If a page is linked to by many other pages on a particular topic then it is considered to be an *authority* on that topic. *Hub* web pages, on the other hand, are pages that link out to many authorities. Hubs play the role of a resource index to informative pages on the web, for example the pages of a web directory such as Yahoo! [68] are hub pages. For a review of algorithms using the link structure of the web to determine the relative rankings of pages the reader is referred to [8].

We explore attempts to modify both content-based and link analysis-based web retrieval techniques to also take account of the individual user's preferences, hence providing personalised web search.

3 Profiles for Personalisation

Any system providing long-term (i.e. between session) personalisation services will need to store some information about the user in order to function. In the case of web search this will be information that aids the system in deciding which web pages are more likely to be of interest to the user. This is what we call a *user profile*, or simply a *profile* for short. The profile will generally differ from system to system, storing different pieces of information in various proprietary formats depending on the exact functionality and the algorithms used.

Collection of data for the profile can be done in various ways, the simplest being to collect users' preferences *explicitly*, by asking them to submit the necessary information manually before any personalisation can be provided. This could be as basic a process as ticking a box to mark a subject area as being of interest to them, or as detailed as filling out long forms of personal information. Explicitly entered profile information is potentially "high quality" and was the choice for many of the earlier systems providing personalisation-like services. However, studies and anecdotal evidence show that users generally dislike having to spend time and effort submitting data to any system, especially when the benefits may not be immediately obvious. There are also often concerns about privacy, and users might not be happy supplying personal information to anonymous servers. This natural combination of a lack of motivation to provide information and caution due to the uncertainty of how their information will be used can make the explicit collection of sufficient data for the profile difficult.

The alternative is to attempt to collect profile data *implicitly* – this means inferring preferences from users' normal interactions with the system. The interactions most relevant to the discovery of web page preferences are things such as visiting a page (and the time spent viewing it), following a hyperlink, scrolling down a page and 'bookmarking', saving or printing a page. An application on the user's machine can monitor such behaviours in the background by 'listening' to browser- or system-level events. The advantage of using implicitly collected data to form a profile is that the user is relieved of the burden of having to supply (and possibly regularly update) the necessary information. Implicit measures of interest are generally thought to be "lower quality" than explicitly gathered preferences [50], although White et al. [65] found there to be no statistical difference between the results they obtained using explicit and implicitly gathered relevance judgements for web pages. Implicit and explicit data collection methods can of course be used in conjunction with one another, potentially giving the best of both worlds; The user can supply as much or as little high quality information as they like, and this can be augmented with information inferred from the large quantities of implicit data that can be automatically generated with no further user effort. The reader is referred to [34] for a review and classification of various methods for inferring user preferences from implicit feedback.

In most search personalisation systems the user profile is stored on the web search server (or on a server that proxies a web search server), the idea being that the search engine (or proxy) can make results more relevant to the user if it has direct access to the profile and so knows something about the individual submitting queries. It is also possible for the profile to be stored on the user's machine (client-side). In this case relevant parts of the profile can be sent to the server when necessary, or the personalisation can be performed on the client machine itself.

Current search personalisation on the web is, in general, supported by static questionnaire-based profiles. The profile used by Yahoo! is stored server-side, and that used by Google Personalized beta [26] (that made its debut late in March 2004) is stored client-side in a cookie. Static questionnaire-based profiles are the easiest to implement but are the most limited kind, as they are not fine-grained enough to represent the range of possible users, they require effort from the user to set up, and they soon become out-of-date unless the user is diligent in updating their profile regularly as their interests change.

4 Approaches to Personalisation

The preceding sections discuss the main elements needed for personalised search: web search techniques and data about users in the form of profiles. This section reviews a number of different approaches to using these elements together and gives details of some systems that have applied the approaches. In 4.1 we look at the use of *relevance feedback*, a technique from IR that uses additional information from the user to aid the retrieval process. Although this is not, strictly speaking, personalisation, ideas from relevance feedback (such as query modification and the use of both explicit and implicit indicators of user interest) form the basis of many of the personalisation systems discussed subsequently. The content-based personalisation systems discussed in 4.2 try to match web pages to users by comparing the content of pages to a profile containing information about the content preferred by the user. The recommender systems discussed in 4.3 similarly use content-based techniques to match documents to users, and although in this case the process is not driven by user queries many of the techniques are relevant to search personalisation. Link-based approaches and systems, discussed in 4.4, take a different tack and attempt to leverage the topology of the web (possibly in combination with page content) to personalise search results. In 4.5 we consider yet another approach to page recommendation: collaborative systems take the behaviour of a whole group of users into consideration when recommending pages to individuals, ranking highly those pages that are popular with other users in the group. Search on mobile devices and location-aware search is discussed in 4.6, and finally we bring together the lessons learned about requirements for personalised search systems in 4.7.

4.1 Relevance Feedback and Query Modification

Work on relevance feedback and query expansion does not generally describe itself as providing personalisation. This is because the techniques are usually designed to be used during the course of a single search session to improve the relevance of the set of documents retrieved from a corpus. This can be seen as short-term (single session) personalisation, however, and the processes would become personalised search if the user's relevance judgements (however they are collected) were to be recorded in a user profile and used across search sessions to improve document retrieval and search result rankings in future, rather than only on a per-session basis.

Relevance feedback is the process of collecting user assessments of retrieved documents and using these assessments to try to improve search results. The user's relevance judgements are generally used to inform query modification, but sometimes they can be used to modify the document representations within the IR system instead [4].

Relevance feedback systems view information seeking as an iterative process rather than as a 'one-shot' search. The user's query is a first attempt at defining their information need, to be refined later in the light of their judgements about the relevance of documents retrieved so far. The sequence of events proceeds as follows: after the initial query is submitted, each document retrieved (or possibly a subset of them) is examined by the user and marked as 'relevant' or 'not relevant'. Keywords are automatically extracted from the rated documents and used to modify the original query – query terms can be added or removed, or in more sophisticated systems the weights of terms in the query vector can be adjusted so that relevant terms are more highly weighted. This modified query is re-submitted to the IR system, a new set of documents is retrieved and the process continues. Some systems, rather than automatically updating and resubmitting the query, suggest extra query terms to the user that can be interactively selected for inclusion in a modified query. The latter option allows users more control over the search process.

In systems where the vector space representation of documents and queries is used the relevance feedback process can be seen as an attempt to move the vector representing the query closer to the vectors representing the relevant documents, the underlying assumption being that documents relevant to the query will have vector representations that lie in a localised region of the vector space. Moving the query vector closer to this region should lead to more of the relevant documents being retrieved [6].

Many of the algorithms that have been used to do this query modification are variations on that originally proposed by Rocchio [56], which has generally shown to have quite positive results [28,13,32] at least for the first few iterations – the amount that the results are improved after four to five iterations (by the standard recall and precision measures of retrieval performance) falls off quite sharply after this. Rocchio's and related algorithms are *linear additive* algorithms, where a linear combination of the vectors representing relevant documents is added to the existing query vector. They are simple and give good results, but may be slow to converge to the target user preference relation. Other approaches that attempt to speed up this convergence are possible: a *linear gradient descent* procedure is described in [66]. A *multiplicative adaptive* algorithm (MA), used in the Multiplicative Adaptive Retrieval System (MARS) meta-search engine, is suggested in [12], and a *multiplicative gradient descent* algorithm (MG) is used in the MAGRADS (Multiplicative Adaptive Gradient Descent Search) system [47]. It is claimed that the multiplicative algorithms converge more quickly (i.e. with fewer iterations) than linear algorithms. The MG algorithm increases the weight of individual terms in the query vector (i.e. not complete document vectors) exponentially after each feedback iteration. It has been found to give significant improvements in relative recall and precision with few (three to five) feedback iterations.

The Scout meta-search engine [39] mentioned in Section 1 also allows users to provide explicit per-session relevance feedback on individual result summaries and on cluster and index labels. When the user chooses to provide feedback a modified query based on the user's relevance judgements is suggested for re-submission to the meta-search.

Query expansion terms are usually determined by considering the term co-occurrences within relevant documents, but the co-occurrences of terms in the user queries as well as the documents inspected can also be useful [15]. The correlations between query and document terms can be automatically extracted by analysing user logs – the terms appearing in documents clicked by users are considered to be relevant to the query submitted. These relationships can then be used in future to suggest query expansion terms.

The Inquirus 2 meta-search engine [24] has a slightly different take on query expansion. It allows users to specify a category of information need along with their query terms in order to more clearly indicate what it is they are looking for (such as recent news, home pages or research papers). Based on the type of information need it selects which sources to meta-search, so the same query may be submitted to a different set of search sites depending on the kind of page required. The query modifications made are also based on the category of information need being addressed; search engine-specific options may be added and additional search terms may be appended or prepended to the user's original query. The query modifications used are simply the addition of fixed extra terms depending on the information need, for example `links resources` is appended to the query when a general purpose resource is required and `abstract keywords references` when the aim is discovery of research papers. Multiple queries can be submitted to each search engine, always including the unmodified original query to avoid 'losing' results that do not satisfy the modified queries but that are otherwise highly ranked by the search engine. In later work on Inquirus 2 [20] the system is extended so that effective query modifications are automatically generated, rather than hard-coded. The rules for generation are extracted using nonlinear support vector machines trained to classify documents. The case for submitting multiple modified queries to the same search engine is also strengthened – it is argued that a single query modification may improve precision, but this will be at the expense of recall (i.e. pages that would have occurred in the results from the user's original query will be excluded – this is always a danger when using query modification). By submitting multiple modified queries the system is more likely to achieve high precision while maximising recall.

Another interesting approach to the processing of relevance feedback in web search is taken by White et al., who initially designed a system to investigate the difference in effectiveness between using explicit feedback and implicit evidence for relevance feedback [65], and who later extended their work to investigate the usefulness of displaying top-ranking sentences (rather than documents) to users [64]. In order to evaluate the effectiveness of using implicit evidence of interest they have built two versions of a web document retrieval system that uses relevance feedback techniques to suggest new documents from the result set to the user – the system is unusual in that it uses relevance feedback to update the information displayed to the user, not to do query modification for further retrieval iterations. The system does not do web IR itself, but instead acts as an interface to existing web search engines (Google and AltaVista are used in the studies). The top thirty results are retrieved and their titles presented to the user. When the mouse pointer is moved over one of the titles a summary of the corresponding result page is shown in a pop-up window. Additionally, a list of the top ranking sentences from within the retrieved pages is also shown as a representation of document content. This method of presentation ranks the retrieved content at the sentence level rather than the document level, as only parts of a document may be relevant.

Relevance feedback techniques are used to alter which sentences are shown and the order in which they are presented. The two versions of the system differ only in the way that feedback is gathered from the user: in one system the user must manually mark-up documents as being relevant or non-relevant by clicking a check-box, and in the other only the implicit evidence of holding the mouse pointer over the link (to make the summary visible) is used. Summaries from the (assessed or assumed) relevant documents are used to generate a list of possible query expansion terms. Expansion terms are selected from these and used along with the original query to generate an updated top-ranking sentence list for presentation to the user. They found there to be no statistical significance in the differences in effectiveness of the two systems and conclude that implicit evidence can be a useful substitute for explicit assessments for relevance feedback systems. The biggest drawback of the system presented is that the download, sentence extraction and summarisation takes about seven seconds in total. This is quite a long time for a web application, where users generally demand sub-second response times.

4.2 Personalisation by Content Analysis

As we have already mentioned, relevance feedback is usually done on a per-session basis to try to focus the user's query. Personalisation, on the other hand, occurs across search sessions by using a user profile as well as query terms to decide which pages will be relevant to the user. In this section we discuss systems that try to determine relevant search results by matching the content of web pages to a user profile that somehow records the type of content that the user is generally interested in.

The user profile employed in the OBIWAN system [53] is structured as a concept hierarchy of 4,400 nodes, with each node weighted to represent the strength of user interest in the topic. Each node of the hierarchy is annotated with a set of terms from ten pages that represent the content of that node, stored as a vector of keyword weights (with weights based on the TF-IDF measure). The profile, which is stored client-side thus avoiding the need for a profile server, is used to both filter and re-rank search results. Information about which topics the user is interested in is implicitly gathered by analysing the pages browsed by the user. The visited pages are categorised as follows: for each browsed page a keyword vector is generated and the similarity of this to the node keyword vectors in the profile is calculated; the most similar nodes (and hence the topics they represent) are assumed to be related to the content of the page. Profile construction is done off-line by periodically categorising the pages in the browser's cache folder and incrementing the weights of the top five category matches by an amount calculated by taking into account the original weighting for that category, the length of time spent viewing the page and the length of the page itself. Experiments show that the number of categories represented in a user's profile soon converges.

Building the profile off-line in this way means that the system cannot adapt to novel user interests exhibited during a single search session, and will only successfully be able to reflect medium- and long-term interests. The experiments reported do not show how the profile will cope as interests change over time – negative feedback is not incorporated in the system at all, and there is no mention of the possibility of "aging" categories that have not been visited for a long time by decreasing their weighting in the profile.

The profile is used to re-rank the results returned by the search engine. The summary of the page is categorised (the system does not download the pages themselves), and the interest of the user in the page is judged by their interest in the categories that the result falls into. Five re-ranking algorithms have been evaluated, all of which take into consideration (to a greater or lesser extent) the original ranking of the page by the search engine, the interest of the user in the top four categories of the document and how well the category matches the document. Re-ranking results gave an increase in precision of up to 8% at low recall, but in experiments with filtering search results (i.e. removing results that the system decides will not be relevant to the user) incorrectly filtered results were up to 12% of the documents excluded, suggesting that the system performs better in ranking than filtering.

The system developed by Liu et al. [43] attempts to disambiguate query terms by associating each term with a small set of categories and augmenting the user's queries with other category-specific terms before submission to a search engine. The system uses a profile that could theoretically be based on the user's search history (the queries they submit, the set of categories related to the query and a list of relevant documents), although for their experiments an artificial "search history" of this sort was generated manually by each subject to be used for profile generation. The profile generated from the search history consists of a set of categories that are each represented by a weighted vector of terms significant to that category for the particular user, so the same term could appear weighted highly in one category but low in another, depending on the user's interests within the categories. The user profile is used along with a general profile of all categories (derived from the ODP (Open Directory Project) [49] hierarchy) to categorise user queries. The system has two modes of operation: "semi-automatic", where the user can select which category the query belongs to from the top three category matches, and "automatic", where the system automatically chooses the best matching category or two to use without any further input from the user. Multiple queries are submitted to the search engine being used (in their study this is Google's Web Directory [25]): the query as submitted by the user and the query submitted with category information. The results are collated and combined into a single list for presentation to the user.

They tested four algorithms that do not allow adaptation over time (where the profile is generated from an initial "search history" and then remains unchanged), and one that does (where the category vectors in the user profile are updated with new information from the users periodically). All non-adaptive systems face the problem that results will deteriorate over time as the user's interests and aims change, making this last algorithm the most promising. However, there is an issue that may restrict automatic profile generation: The approach depends fundamentally on knowing which categories each page falls into. They get around this by using the Google Web Directory as the underlying search system – any other categorised directory such as Yahoo! or ODP would serve as well. However, directories such as this are orders of magnitude smaller than the biggest search engines (on 5th May 2004 ODP was reporting about 4.4 million web pages, whereas Google's main search site claimed to index about 4.3 billion web pages), so any personalisation system that depends on them will exclude most of the web from its results. To solve this problem the personalisation engine could be augmented with an automatic page classifier – obviously this would be non-trivial, but it is one possible way that the approach could be scaled for searching the unclassified web at large.

Outride [52] is another personalisation engine that acts as an intermediary between the user and a search engine. Query modification in Outride is done based on the user profile as queries are entered, and the returned results are filtered or re-ranked before being presented to the user. The user interface is a sidebar of the browser, including the user's bookmarks and surf history along with a web directory (based on the ODP), and search results. The user profile is initially derived from the bookmarks imported by the user, and then updated with information extracted from pages browsed by the user. Re-ranking of search results is based on vector space methods that compare the titles and other page metadata with the user profile. Experiments with the system show that users find the information they are looking for more quickly using Outride than using only the underlying search engine. Outride was acquired by Google in 2001, but to date there has been no release of the technology.

4.3 Recommender Systems

Recommender systems are systems designed to recommend content based on some learning mechanism. The user's long-term information needs that the systems try to cater for can be seen as a query (or set of queries), so although these systems are not overtly query-driven, the techniques they employ are very relevant to personalised web search. Anticipating information needs to actively suggest items to users is akin to anticipating the queries that a user would want to formulate (much as systems employing query modification do).

Letizia [41,42] is a client-side agent that accompanies the user as they browse the web, recommending links from the current page being browsed when asked to do so. When the user navigates to a web page Letizia begins to download and analyse the pages linked to from the current page in breadth-first order. The content of the pages is compared to a keyword-based user profile and the links are ranked according to the similarity between the page content and the profile.

WebWatcher [2,33] acts as a "tour guide" in a similar way to Letizia, but requires the user to explicitly specify a goal at the outset, whereas Letizia tries to infer the user's current goals implicitly from their browsing behaviour. WebWatcher is a server-based system, so unlike Letizia it has access to the browsing histories of multiple users to draw on in deciding which pages are likely to be relevant to the current user. URLs are annotated with the goal descriptions of users who visited the page on a successful "tour" (users are asked to give explicit feedback on whether their goal has been achieved at the end of a session). During a session, hyperlinks in the current page are scored against the current user's interest using the cosine similarity metric, and those scoring highly are suggested to the user.

Syskill & Webert [51] is another agent that assists in browsing the web. The user must provide explicit feedback by rating pages visited on a three-point scale. The user profile consists of a set of topic-specific profiles, with one weighted term vector for each distinct topic that the user is interested in. These profiles are learned using a naïve Bayesian classifier, comparing the user's relevance judgements with the content of the rated pages. The profile is then used to suggest web pages that the user might wish to visit from a topic-specific 'index-page' (containing links to many pages on the topic) that the user must supply the URL for. The profile can also be used to generate queries for submission to a search engine.

WebMate [11] similarly stores a separate section of user profile for each user interest, which it learns automatically rather than requiring the user to explicitly supply a list of topics. The profile is stored as TF-IDF weighted vectors of terms which are matched to pages using the cosine similarity measure. WebMate uses the profile to suggest documents that match the user's interests, and compiles a "personalised newspaper" from various news sources on the web. It also aids the user in refining their search terms.

ICPF [71] provides the basis for a client-side recommender system. The system is initially trained using manually annotated logs of session histories to identify *information content* (IC) pages – pages that contain information relevant to the user – based on browsing behaviour (such as following a link or going back a page). This gives it a model of general web users. When an individual begins a browsing session the system uses this general model to try to identify which terms on the pages visited will be IC terms (i.e. terms that will appear on the final IC page that satisfies the user information need) using a naïve Bayesian classifier. Experiments have shown that the system can effectively predict the words that are relevant to the user. It remains for the system to be extended to use these predictions to recommend pages to the user. This could possibly work in a similar way to Letizia, checking the links ahead of the current page, or it could submit the predicted IC words to a web search engine and evaluate the pages returned for relevance to the user.

Quite a lot of work has been done on recommender services for internet news. There is now more news available on the web than anyone has time to read, so news aggregation and filtering according to user interests is highly desirable. News recommendation is a slightly more manageable problem than web page recommendation (or personalised search), as it only needs to deal with perhaps a few thousand sources of mostly high quality content, rather than upwards of four billion web pages of varying quality. It appears to be a slightly different problem from that of web search as news filtering is not generally query-driven, a range of topics will be of interest to the user, and the novelty of the stories presented is particularly important. However, many of the techniques used may be transferrable to personalised web search.

An early attempt to provide a personalised ranking of news articles is presented in [31]. A user model neural network based on the significant features of the articles read and those explicitly rejected by the user is used. This adapts over time to represent the user's interests, and is used to rank incoming USENET news articles. Common features of articles read by the user are represented by nodes of the neural network, and connections are made between co-occurring terms. The system also allows augmented keyword search of the news archive in conjunction with the user model neural net. Article ranking is done using a combination of query terms and activation of the neural network by the features of the article. When keywords are submitted for search the energies of the nodes representing the keywords are substantially increased (and if a node is not already present it is added as a temporary unconnected node with high energy). Once the search is complete the network reverts to its previous state (so submitting queries does not permanently change the user profile; only read and rejected articles do this). Update of the profile is done off-line at the end of a news browsing session on the news server, but the updated profile can then be stored on the client machine for processing searches, to try to share computational cost among a large group of users.

Adaptive Information Server (AIS) [7] is a client-server agent for adaptive news access. It uses a hybrid user model, with separate models for long-term and short-term interests. These are acquired using a multi-strategy machine learning algorithm (the nearest neighbour algorithm is used for short-term interests, and a naïve Bayesian classifier for long-term) based on both implicit and explicit user feedback. The algorithm used tries to address the fact that a user's information need can change as a result of interaction with new information. News stories are collected and stored on a central server. This allows for collaborative as well as individualised techniques to be used in choosing news stories to recommend. Users have ubiquitous access to their profile via two different client agents – one for web access (Daily Learner) and one for mobile access (Daily Learner Palm VII^{TM} Edition). The system gives users the option of providing explicit feedback if they like, allowing stories to be rated as interesting, uninteresting or 'I knew that already'. If keyword-based search is used then the ranking of news stories is based on the user's interest profile.

The system presented by Sunderam in [60] is a client-side application that uses a profile based on implicit user feedback to retrieve, filter and present personalised news articles. The system acts as a proxy server on the user's computer, monitoring the user's Internet news reading patterns to track interests. The profile is a vector of term weights and the cosine similarity metric of the vector space model is used to judge the similarity of the profile to incoming news stories. While the user is browsing news the system monitors how quickly links are followed after reading a headline, the time spent reading an article, the number of embedded links followed and the percentage of topical articles read. All of these factors are considered when allocating weights to terms in the profile vector. Every 15 minutes new stories (URLs) are analysed for relevance to the user and high-scoring articles are added to a personalised news web page for viewing by the user later on. The client-side implementation allows the user to obtain personalised news from multiple news sites and sources – for the reported experiments the sites chosen were `cnn.com` and `dailynews.yahoo.com`. Evaluation was carried out over two weeks by creating several user profiles trained with differing numbers of documents. High precision was achieved with only five training articles, and the system also appears to adapt well to changing user interests, the process of replacing one interest completely with another in the profile taking around 35 training articles to achieve.

Two personalised news services currently available on the web are Findory [19] and MSN Newsbot (beta) [48]. Neither give much detail of the personalisation algorithms that they use: Findory's algorithm appears to be a hybrid, combining both content-based and collaborative filtering methods to suggest news stories. MSN Newsbot clusters stories into categories for presentation, and although no details are given it appears that the stories suggested to a user may be based on the categories they have previously viewed.

4.4 Personalisation by Link Analysis

Just as pages relevant to a query can be determined by analysing the link structure of the web, so can pages that will be relevant to a particular user. In this section we discuss approaches that could be used to combine information about the topology of the web with information about the user to provide personalised search results. There are few systems that actually provide personalisation in this way, but some variations on the PageRank algorithm have the potential to be used for personalisation.

The topic-sensitive PageRank [29] algorithm firstly categorises query terms and then calculates PageRank scores favouring the topic of the query. Whereas the original PageRank algorithm computes a single vector based solely on the link structure of the web [9], topic-sensitive PageRank precomputes a set of PageRank vectors, each biased towards a different topic. At query time the (likely) topic of the query is ascertained, and pages are scored according to the topic-biased PageRank vector. Although Google has not disclosed the algorithm used as yet, it is possible that this is the basis of the method of personalisation used by the recently introduced Google Personalized beta service from Google labs, which allows users to specify a set of categories that they are interested in – this would work by calculating results using a PageRank vector sensitive to the topics in the user profile (rather than to the topic of the query).

Personalised PageRank, a variation of the idea of topic-sensitive PageRank, is presented in [30]. This suggests the possibility of biasing the scores of pages according to a user-specified set of interesting pages (for example their bookmarks – which in this case form the user profile). Personalised PageRank Vectors (PPVs) are not generated from scratch each time a user enters a query, as this would be too computationally expensive to be a scalable solution. Instead, a set of partial basis PageRank vectors representing hub pages is initially computed from the web graph, and the PPVs are calculated as a linear sum of these basis vectors, thus avoiding the need to do the link analysis necessary for the generation of PPVs at query time. The prototype PROS system [14] implements personalised PageRank, choosing hub pages to form the basis of the profile automatically from the user's bookmarks and most visited web pages. It has so far only been tested on a small crawl of 3 million web pages, but improvements in the relevance of retrieved documents over (unpersonalised) PageRank have been found.

The Intelligent Surfer [55] is another system that precomputes some data (in this case at crawl time) in order to efficiently calculate, at query time, PageRank vectors biased to reflect the relevance of a page to a query. It does this using an algorithm called query-dependent PageRank (QD-PageRank). While this is not exactly personalisation, it is easy to see how the approach could be used in conjunction with a profile consisting of queries that the user is interested in to provide personalised QD-PageRank scores for pages.

The Compass Filter [38] is one system that does do genuine personalisation based on link analysis. It uses the concept of *web communities* [23,21] and their neighbourhoods in providing personalised web search results. A web community is a small set of highly connected (i.e. hyperlinked) web pages, which are generally formed of hubs and authorities in a specialised subject area. The *neighbourhood* of such a community is defined as the set of pages that either inlink to community pages or that are the target of outlinks from community pages. The system works by first extracting the web communities and their neighbourhoods from the corpus (1.33 million documents from the Greek part of the web – a very small corpus in today's web search terms). The user profile consists of weightings for each community neighbourhood, based on the number of times that pages belonging to the neighbourhood have been visited by the user in the past. A re-ranked list of results is generated by calculating a score for each URL as the sum of the profile weights for each community that the page belongs to. Initial results are of limited scope (only 44 queries had their results re-ranked), but they do show a statistically significant increase in the user success metric used for evaluation. However, the re-ranking algorithm seems to us to be too simplistic as the final ordering depends solely on how well the pages match to the profile. We believe that the best personalisation algorithms will be those that additionally take some account of the relevance of the pages to the query as well as to the user's profile (and not just when it is necessary to "break ties", as the Compass Filter does).

4.5 Social Search Engines

A different approach to locating relevant web pages is taken by systems that consider the behaviour of other users of the system when generating search results and recommendations. Such systems are called *collaborative systems* as they consider the user within a community rather than as an isolated individual. This seems to be a common-sense approach to identifying interesting items once we make the observation that "word of mouth" is probably the most common method of information filtering in everyday life – people tend to read books, watch films, take holidays and so on based on the recommendations of people they trust.

To date, collaborative techniques have mostly been used in recommender, rather than search systems [58,54,17,36]. The usual technique used is to discover users who have profiles similar to that of the current user, often using an algorithm such as k-Nearest-Neighbour (k-NN), and recommend items based on usage patterns of these similar users. Many systems base their recommendations solely on the user profiles and usage records, and the content of the items is not considered. However, it is possible to create hybrid systems that take both content and usage into consideration [5], in theory getting the best of both worlds. As well as a plethora of recommendation systems, there are some examples of systems where collaborative techniques have been applied to search.

The now defunct DirectHit [16] search engine used a popularity-based search algorithm, ranking URLs in order of popularity, with the pages visited most by other users ranking highest in their search results. This strategy is based on the assumption that the most relevant pages of a topic are those that have been most visited. The popularity of a page was ascertained from clickthrough data in their query log files, and the system considered the time spent by users on the pages as well as the number of clicks on the URL.

A similar approach is taken by I-SPY [22], which is a popularity-based search interface designed to be used in conjunction with specialised search engines, with communities of users having similar interests and information needs, rather than by the highly heterogeneous web community at large. The system acts as an interface to existing search services and re-ranks the results according to the preferences of a community of users, as judged from a query-to-page relevance model based on which pages have been visited by users as a result of issuing similar queries in the past. This model will be "tuned" to the domain of interest by the actions of the particular community of users. In experiments using a wildlife website the system produced four relevant results out of the top five, compared with one out of five in the results from HotBot [45], and achieved a much better recall at low result-list sizes (with up to 10 results). In their "live user" study, participants without I-SPY's re-ranked results on average needed to examine 15% more pages than those using the system, and the average ranking of results followed by I-SPY users was 2.24, compared with 4.26 for those without the re-ranked results.

An experimental algorithm called Link Fusion [67] augments the basic HITS link analysis algorithm by adding a "user popularity" attribute to it (where 'popular' users are those that always visit good hub and authority pages). This is done by identifying the hub and authority pages as in HITS, and then extracting the popularity component from the overlap between the usage data in server log files and these hubs and authorities. Experiments using a 10-day log from a proxy server at Microsoft found that adding this user popularity attribute enabled the Link Fusion algorithm to outperform HITS by 24.6% for 10 sample queries.

A commercial system called Eurekster [18,59] provides search results personalised according to the other people that the user knows (all users must create a *social network* of the other Eurekster users that they know). It acts as an interface to other search engines, storing a history of users' searches. When results are returned via Eurekster, pages that have been viewed or highly rated by other members of the user's social network have their ranking boosted. This technique, like I-SPY, will probably be most useful within a community of people who share a common interest or goal. To this end, Eurekster allows the user to divide their contacts into subgroups ("SearchGroups") that each share such a common interest. Searches can then be shared with only a subset of these groups. This system does raise additional privacy issues, as search activity is explicitly associated with personal information about the user and who they know.

4.6 Mobile and Context-Aware Searching

Search on mobile and hand-held devices presents particular challenges due to the technical constraints of a small screen size, low bandwidth and cost. These constraints make effective personalisation and good interface design even more valuable than on the web in general – long lists of results are no good for hand-held devices because users may then need to scroll through pages of results. Much of the work done on personalising search for mobile devices concentrates on the interface rather than on the results themselves, the exception to this being services that take the user's location into consideration when suggesting results. Other contexts that could be taken into consideration include the time of day it is and whether the user is at work, at home or out and about somewhere [40].

Hyponym [3] organises search results by topic. The topics are found by analysing the wireless usage logs collected at a proxy server. Sequences of pages visited by users when searching are discovered, and queries leading users to similar pages are deemed to be topically related. This allows queries to be clustered and these clusters in turn are used to structure the search results display. Although the system currently uses aggregate usage data from all users to decide on the topics, the model can be extended to include user-level information such as location, and to use pre-existing hierarchies of pages such as those used by Yahoo! or ODP.

WebClipping2 [10] is a mobile interface for news filtering. It uses a Bayesian classifier to rate news stories based on the user profile. Keywords from news stories are matched with keywords in the user profile that represent the user's interests. A list of interests must be provided when first using the system to allow generation of a personal keyword database, but subsequently user behaviour is monitored and the information is used to adapt the initial profile by increasing or decreasing the weight of keywords in the profile. In experiments where users were allowed to build their own profiles they expressed satisfaction with the system after using it for only 1-2 days. To test how well the profile adapts to changing user preferences some users were made to begin using the system with a profile of interests opposite to their actual interests. In this case it took 8-10 days before the profile had adapted enough for the users to be happy with the news stories they were presented with.

Although possibly at its most useful in mobile scenarios, search biased towards the location of the user also has a place on the web. One of the earliest (and ongoing) "local search" systems that we have come across is My Yahoo! [46]. When the user supplies the system with their US zip code it is stored in their profile and some search result pages are then optimised to supply local results for certain types of search – the examples given are for cinema showing times and restaurants. The user must explicitly change the information held about their zip code if they change their location – the system does not adapt automatically.

Google Local [27] and Yahoo! Local [69] allow US-based users to search for local businesses and services by entering their Zip code, city or address along with their search terms. This sort of service could be easily adapted for mobile search, taking the location of the phone as input along with a query from the user.

Topix.net [61] provides local search over news stories (again, mainly for the US at present), and there are plans to add additional personalisation functionality in the near future. Topix crawls, clusters and categorises articles, first by location and then by subject. As well as the URL, title and summary of articles, Topix stores the latitude and longitude of both the news source and the story itself, the prominence of the source and the subject categorisation.

4.7 Requirements for Personalised Search

We have examined a representative sample of personalisation-type systems that have been developed to date. They are a diverse collection, and do not bear direct comparison with one another as they are generally trying to achieve different aims. We can, however, identify several distinguishing features of personalised search and similar systems, and identify some requirements for an "ideal" personalised search system:

- *User data collection method* – Is the data collected explicitly from the user or implicitly inferred from their normal interaction with the system? We saw in 4.1 that either method may be equally effective, although for our "ideal" requirements we favour the use of implicit data collection (possibly in combination with optional explicit collection), so the user is not burdened with submitting data or feedback to the system.
- *Profile storage* – Is the user data stored client-side on the user's machine or at a server? Again, there are pros and cons to each of these approaches. Client-side systems provide privacy and security, but profile portability may be an issue. Collaborative filtering type techniques are easier in a server-side environment, but if all profile processing is server-side then scalability can become an issue. The best solution may be to do as much profile collection and processing as possible on the client, but share data with servers when necessary.
- *Adaptivity* – Does the system automatically adapt to the user over time? If not, can the user manually affect the behaviour of the system (e.g., by updating a static profile)? Ideally, personalisation systems should adapt to the user to reflect their current interests and preferences.
- *Profile construction* – Is the user profile updated on- or off-line? On-line systems are preferable as they can immediately adapt to behaviour during a session.
- *Profile data* – What exactly does the profile store? We have seen examples where the profile consists of weightings that represent user interest in a range of topics, and where features extracted from page content are stored explicitly. The suitability of the profile will be determined by the algorithms used.
- *Personalisation method* – Does the system re-rank or filter third party search results, create modified queries to submit to third party search services, or perform it's own personalised retrieval? All of these approaches have their merits, although we think it makes sense to personalise the results from existing search services, thus taking advantage of the work done to date producing good web search results rather than re-inventing the wheel. Good personalisation could easily suffer from bad basic information retrieval or a small index size. The link-based personalisation algorithms in 4.4, however, rely on knowledge of the web topology and so require specialised indexes to perform their personalised retrieval.
- *Algorithm used* – What algorithm(s) are used to create personalised results?
- *Interface* – How are the personalised results presented? This could be across the web to a browser, to a special-purpose or customised client application or to a mobile interface.

5 PResTo!

We are currently developing and testing a Personalised Results Tool (PResTo!) that provides personalised web search results by re-ranking third party results according to a user profile. Of the systems described in Section 4 PResTo! is most similar to OBIWAN and Outride, in that it is a client-side agent that re-ranks third party search results. It differs from these systems mainly in the profile content and structure, and the algorithm used for re-ranking.

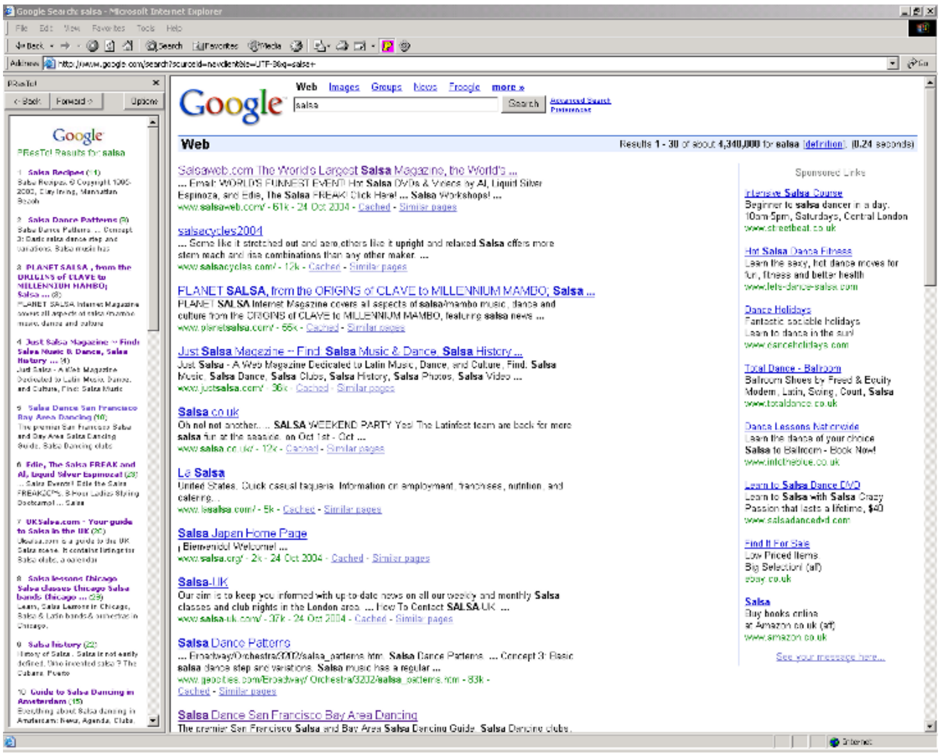
PResTo! is a plug-in for Internet Explorer that maintains a profile based on the user's interactions with web search sites (currently Google, Yahoo!, Yahoo! UK and CiteSeer are supported, but any other search sites can easily be added). It learns the user profile transparently, without intervening in the user's normal browsing behaviour (although we plan to incorporate optional explicit feedback mechanisms at a later date). The profile is used to re-rank the results returned, and the new ranking is displayed in a sidebar of the browser window alongside the original results page from the search engine, as can be seen in Fig. 1. Users can browse results by clicking in PResTo!'s interface or on the main results page. The re-ranked result list remains visible in the sidebar while users browse the pages.

The user profile is kept private to the user – it is stored client-side and all personalisation takes place locally. Logging of search activity and re-ranking of results can be turned on and off independently at will – when logging is on, the profile is incrementally updated on-line as the user searches for information on the web, allowing the profile to adapt to the changing and developing interests of the user.

The model used is of a user interacting with a search engine, and the user profile consists of data about these interactions. For each query submitted to a supported search engine PResTo! records the *query* and the *URLs* visited as a result of issuing the query; For each URL visited the *keywords* associated with the page (extracted from the search engine's summary of the page), the *number of user visits* to the page and a *timestamp* are recorded. Our approach assumes that past search behaviour is an indicator of the user's future behaviour. We avoid the need for collecting a large amount of data from the user before a model can be created by incrementally updating the profile as data is (implicitly) made available. When data relevant to the current query exists in the profile (i.e. data collected when the user previously submitted at least one of the current query terms) it is used to re-rank, but if no relevant data is available then the ranking presented to the user is the same as that generated by the search engine.

Three separate structures are used to store the profile data. The bulk of the data about URLs visited (keywords, timestamp, number of visits) is stored in a flat file with fixed-length records. The index into this file is a collection of *word suffix trees* [1] rooted at the same node, along with a lookup table containing pointers into the inverted file.

The compound word suffix tree is the key to the profile. It contains all the queries submitted by the user – each query is recorded in the suffix tree whether or not any URLs are visited as a result of issuing the query. Each node in the


Google Search: salsa - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://www.google.com/search?sourceid=navclient&ie=UTF-8&q=salsa+
Go
PResTo!
<- Back Forward -> Options
Web Images Groups News Froogle more »
salsa
Search
Advanced Search
Preferences
Internet

Google
PResTo! Results for: salsa

1 **Salsa Recipes** (11)
Salsa Recipes. © Copyright 1995-2003, Clay Irving, Manhattan Beach

2 **Salsa Dance Patterns** (9)
Salsa Dance Patterns. ... Concept 3: Basic salsa dance step and variations. Salsa music has

3 **PLANET SALSA , from the ORIGINS of CLAVE to MILLENNIUM MAMBO; Salsa ...** (8)
PLANET SALSA Internet Magazine covers all aspects of salsa/mambo music, dance and culture

4 **Just Salsa Magazine -- Find: Salsa Music & Dance, Salsa History ...** (4)
Just Salsa - A Web Magazine Dedicated to Latin Music, Dance, and Culture, Find: Salsa Music

5 **Salsa Dance San Francisco Bay Area Dancing** (10)
The premier San Francisco Salsa and Bay Area Salsa Dancing Guide. Salsa Dancing clubs

6 **Edie, The Salsa FREAK and Al, Liquid Silver Espinoza!** (23)
... Salsa Events! Edie the Salsa FREAK2CPS. 8-Hour Ladies Styling Bootcamp! ... Salsa

7 **UKSalsa.com - Your guide to Salsa in the UK** (20)
Uksalsa.com is a guide to the UK Salsa scene. It contains listings for Salsa clubs, a calendar

8 **Salsa lessons Chicago Salsa classes Chicago Salsa bands Chicago ...** (29)
Learn, Salsa Lessons in Chicago, Salsa & Latin bands & orchestras in Chicago.

9 **Salsa history** (22)
History of Salsa . Salsa is not easily defined. Who invented salsa ? The Cubans, Puerto

10 **Guide to Salsa Dancing in Amsterdam** (15)
Everything about Salsa dancing in Amsterdam: News, Agenda, Clubs,

Web Results **1 - 30** of about **4,340,000** for **salsa** [definition]. (**0.24** seconds)

Salsaweb.com The World's Largest **Salsa** Magazine, the World's ...
... Email: WORLD'S FUNNEST EVENT! Hot **Salsa** DVDs & Videos by Al, Liquid Silver Espinoza, and Edie, The **Salsa** FREAK! Click Here! ... **Salsa** Workshops! ...
www.**salsa**web.com/ - 61k - 24 Oct 2004 - Cached - Similar pages

salsacycles2004
... Some like it stretched out and aero,others like it **upright** and relaxed.**Salsa** offers more stem reach and rise combinations than any other maker. ...
www.**salsa**cycles.com/ - 12k - Cached - Similar pages

FLANET **SALSA**, from the ORIGINS of CLAVE to MILLENNIUM MAMBO; **Salsa** ...
PLANET **SALSA** Internet Magazine covers all aspects of **salsa**/mambo music, dance and culture from the ORIGINS of CLAVE to MILLENNIUM MAMBO, featuring **salsa** news ...
www.planet**salsa**.com/ - 55k - Cached - Similar pages

Just **Salsa** Magazine -- Find: **Salsa** Music & Dance, **Salsa** History ...
Just **Salsa** - A Web Magazine Dedicated to Latin Music, Dance, and Culture, Find: **Salsa** Music, **Salsa** Dance, **Salsa** Clubs, **Salsa** History, **Salsa** Photos, **Salsa** Video ...
www.just**salsa**.com/ - 36k - Cached - Similar pages

Salsa.co.uk
Oh no! not another... ... **SALSA** WEEKEND PARTY Yes! The Latinfest team are back for more **salsa** fun at the seaside. on Oct 1st - Oct ...
www.**salsa**.co.uk/ - 12k - Cached - Similar pages

La **Salsa**
United States. Quick casual taqueria. Information on employment, franchises, nutrition, and catering. ..
www.la**salsa**.com/ - 5k - Cached - Similar pages

Salsa Japan Home Page
¡ Bienvenido! Welcome! ...
www.**salsa**.org/ - 2k - 24 Oct 2004 - Cached - Similar pages

Salsa-UK
Our aim is to keep you informed with up-to-date news on all our weekly and monthly **Salsa** classes and club nights in the London area. ... How To Contact **SALSA**-UK ...
www.**salsa**-uk.com/ - 37k - 24 Oct 2004 - Cached - Similar pages

Salsa Dance Patterns
... Broadway/Orchestra/3202/salsa_patterns.htm. **Salsa** Dance Patterns. ... Concept 3: Basic **salsa** dance step and variations. **Salsa** music has a regular ...
www.geocities.com/Broadway/ Orchestra/3202/**salsa**_patterns.htm - 83k -
Cached - Similar pages

Salsa Dance San Francisco Bay Area Dancing
The premier San Francisco **Salsa** and Bay Area **Salsa** Dancing Guide. **Salsa** Dancing clubs,

Sponsored Links

Intensive **Salsa** Course
Beginner to **salsa** dancer in a day.
10am-5pm, Saturdays, Central London
www.streetbeat.co.uk

Hot **Salsa** Dance Fitness
Learn the sexy, hot dance moves for fun, fitness and better health
www.lets-dance-salsa.com

Dance Holidays
Fantastic sociable holidays
Learn to dance in the sun!
www.danceholidays.com

Total Dance - Ballroom
Ballroom Shoes by Freed & Equity
Modern, Latin, Swing, Court, **Salsa**
www.totaldance.co.uk

Dance Lessons Nationwide
Learn the dance of your choice
Salsa to Ballroom - Book Now!
www.intotheblue.co.uk

Learn to **Salsa** Dance DVD
Learn to **Salsa** with **Salsa** Crazy
Passion that lasts a lifetime, $40
www.salsadancedvd.com

Find It For Sale
Low Priced Items.
Big Selection! (aff)
ebay.co.uk

Salsa
Buy books online
at Amazon.co.uk (aff)
www.amazon.co.uk

See your message here...

Fig. 1. Re-ranked results generated by PResTo! for the query 'salsa' on Google

suffix tree contains a *count* of the number of times the sub-query represented by the path from the root to the node has been issued, along with a *list of URLs* that have been visited as a result of issuing the query. The 'hit count' of the number of times the sub-query has been issued gives an indication of the level of user interest in the topic(s) represented by the query, and the list of URLs act as pointers into the lookup table (which is an in-memory hashtable), to retrieve data from the main flat file about the pages visited by the user. Each node also holds a set of internal pointers to enable the location of URLs when a previous query is only partially matched. The overall structure of the profile data structures is shown in Fig. 2.

When re-ranking the results PResTo! uses only the data associated with similar earlier queries, ensuring that only parts of the profile relevant to the current query are used to inform the re-ranking process. The paths from the root of the suffix tree to the nodes representing the query string and its suffixes are followed, and the data relating to the URLs at the nodes found is retrieved from the main data file. The composite suffix tree data structure in PResTo! allows fast searching for these matching terms. It also saves us from having to model separate domains of interest in separate profiles because the structure of the index naturally acts as a kind of 'clusterer' for different areas of

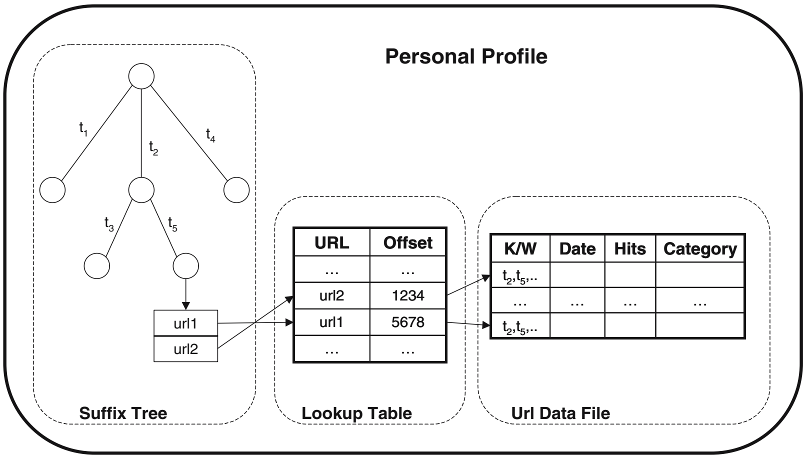


Fig. 2. Overview of the structure of PResTo!'s user profile

interest – by focussing on and only extracting data directly relevant to the current user query, irrelevant parts of the profile that may otherwise confuse the re-ranking process are automatically ignored. If any of the query terms have previously been entered by the user then at least one sub-query match will be found in the suffix tree, providing data for the re-ranking algorithm.

The re-ranking algorithm is based on the vector space model from IR. A set of weighted vectors of terms is created from the keywords associated with the relevant previously visited URLs. The term weights for each URL depend on a combination of:

- the depth in the tree of the node where the URL was found – longer matches are weighted higher;
- the 'hit count' at the node in the suffix tree where the URL was found – queries that have been submitted often will be of higher importance to the user;
- the age of the entry in the URL data file – extra weighting is given to the most recent information in the profile, so the system will adjust quickly to novel behaviour (i.e. new search interests);
- the number of times the URL has been visited previously.

Similar vectors are created for each result using the terms in the title, URL and summary. Terms in the result vectors are weighted according to TF-IDF, where the set of results is taken as the 'corpus' for document frequency calculations. A score for each result is calculated by taking the cosine similarity of the result vector to the vectors created from profile data. This score is adjusted to take the original search engine's ranking into account (this ensures that if there is little or no relevant data in the personal profile PResTo! will simply return the original ranking).

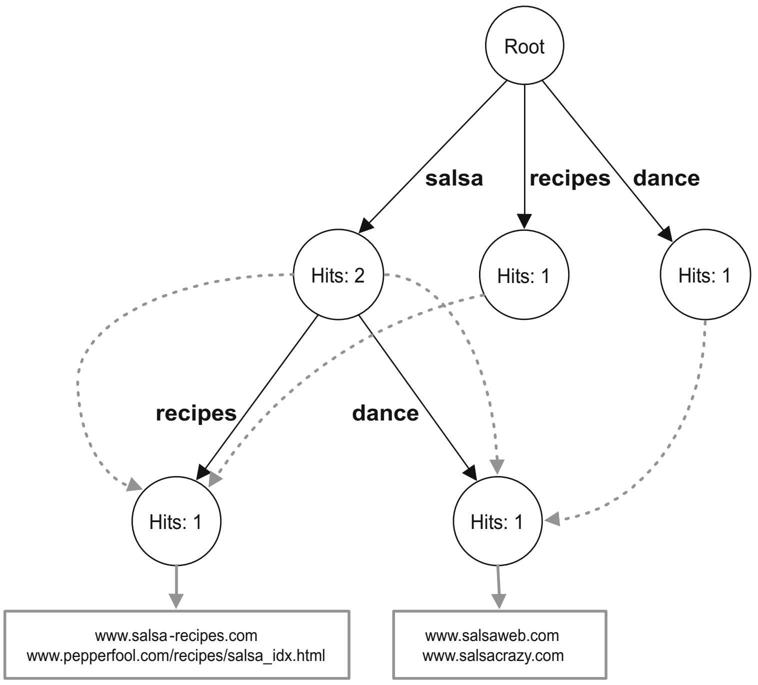


Fig. 3. User profile after the submission of queries ‘salsa dancing’ and ‘salsa recipes’, with two results clicked for each

Fig. 3 shows the suffix tree part of a user profile after the submission of two queries, namely ‘salsa recipes’ and ‘salsa dance’. Two results have been clicked for each of these queries. Fig. 1 shows the re-ranking produced by PResTo! for the query ‘salsa’ based on a profile in this state: a page of *Salsa Recipes* has been brought to the top of PResTo!’s ranking from 11th in Google, reflecting the user’s interest in recipe pages, and Google’s number two result, *salsacycles2004*, has dropped out of PResTo!’s top ten as it is not related to recipes or dancing.

Our client-side approach avoids the problems of privacy and security as personal information is not shared with any servers or search engines. It also has the benefit of not burdening search servers with additional computational load. The support for personalisation of multiple search sites using a single profile means that users are not tied to any particular search tool to be able to get personalised results, and when loyalty changes to a new search tool there is no need to rebuild a profile from scratch.

We are currently evaluating an initial release of PResTo!. Future plans for the development of the system include the addition of category information for each page visited, and a new section of the profile reflecting the weight of user interest in different categories of page. The addition of category information should greatly improve the quality of the re-ranking. Users could also optionally specify a set of categories they are interested in, to reduce the "startup time" before useful personalisation is possible.

The disadvantage of a client-based personalisation system is that the user does not have ubiquitous access to the profile – it may not be available if the user logs in from a different machine. We plan to make the profile easily portable between different machines and devices so the user can take the most up-to-date version of their profile with them wherever they go.

6 Final Remarks

Personalisation technology has matured in the last few years to the degree that large-scale personalised search systems can now be deployed. Various algorithms and techniques have been developed and tested in mostly specialised and restricted domains, some of which have been reviewed here. Relevance feedback, a technique dating to the early days of IR, is being incorporated into result ranking and is being enhanced and adapted to take advantage of the unique features of web search. Moreover, newer systems are beginning to combine several different techniques to improve the user's overall experience of search personalisation.

The challenge facing any personalisation system is that to succeed commercially on a large scale the loyalty and trust of users must be won. We see three main components that must be addressed in overcoming this challenge. Firstly, the system's behaviour should be predictable and its workings transparent. Lack of predictability is an issue for any adaptive system where following the same sequence of actions at different times could lead to different results, as users like to have a degree of knowledge of what to expect. Any lack of predictability should be compensated for by providing an explanation of why the results have been reranked in the order they have, or why a particular site has been recommended. Secondly, the system needs to be highly scalable and very robust. Server-side personalised search systems present scalability problems over and above those of standard web search because the profiles for all users will need to be stored, these must be retrievable quickly, and the additional computational load of running a personalisation algorithm will need to be catered for. No system with slow response times or periods of unreachability due to server overload will gain the loyalty of its users. Finally, issues of privacy and security must be addressed. These issues apply to all personalised systems, but especially to those serving mobile and ubiquitous devices. Users will need to be confident that their personal information will not be shared with any third parties without their prior consent, which requires trust in the company ethos. Users will also need to be confident that their data cannot be stolen or maliciously tampered with, which requires trust in the company's security systems. As Manber et al. say, "any company that collects private information must guard that information with its (business) life" [46].

We are now at the point where standard web search is relatively mature. From here, personalisation becomes of prime importance, and crucial to the strategies of web search engines to keep their users. Google is the first major search engine to offer personalised web search results, but as we know from standard web search (where Google was a relative latecomer) this is no indication of which system will eventually prevail, if any. Web users are notoriously fickle and if something better comes along allegiances can switch almost overnight. With huge advertising revenues at stake, the race to provide quality personalised results is surely now on.

References

1. A. Andersson, N.J. Larsson, and K. Swanson. Suffix trees on words. *Algorithmica*, 23:246–260, 1999.
2. R. Armstrong, D. Freitag, T. Joachims, and T. Mitchell. WebWatcher: A learning apprentice for the World Wide Web. In *AAAI Spring Symposium on Information Gathering*, Stanford, CA, March 1995.
3. K. G. August, M. H. Hansen, and E. Shriver. Mobile web searching. *Bell Labs Technical Journal*, 6(2):84–98, 2002.
4. R.A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* ACM Press and Addison-Wesley, Reading, Ma., 1999.
5. Marko Balabanović. An adaptive web page recommendation service. In *Proceedings of the First International Conference on Autonomous Agents*, pages 378–385. ACM Press, 1997.
6. R.K. Belew. *Finding Out About.* Cambridge University Press, Cambridge, UK, 2000.
7. D. Billsus and M. J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10:147–180, 2000.
8. A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *World Wide Web*, pages 415–429, 2001.
9. S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.
10. R. Carreira, J. M. Crato, D. Gonçalves, and J. Jorge. Evaluating adaptive user profiles for news classification. In *Proceedings of the 9th International Conference on Intelligent User Interface*, pages 206–212, Madeira, Funchal, Portugal, January 2004. ACM Press.
11. L. Chen and K. Sycara. WebMate: A personal agent for browsing and searching. In Katia P. Sycara and Michael Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 132–139, New York, 1998. ACM Press.
12. Z. Chen and X. Meng. MARS: Applying multiplicative adaptive user preference retrieval to web search. In *Proceedings of the International Conference on Internet Computing 2002 (IC2002)*, pages 643–648, Las Vegas, Nevada, USA, 2002.
13. Z. Chen and B. Zhu. Some formal analysis of Rocchio's similarity-based relvance feedback algorithm. In *International Symposium on Algorithms and Computation*, pages 108–119, 2000.
14. P. Chirita, D. Olmedilla, and W. Nejdl. PROS: a personalized ranking platform for web search. In *Proceedings of the 3rd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, Eindhoven, Netherlands, August 2004.

15. H. Cui, J. Wen, Jian-Yun Nie, and Wei-Ying Ma. Query expansion by mining user logs. *IEEE Transactions on Knowledge and Data Engineering*, 15(4), July/August 2003.
16. Direct Hit. Popularity-based search. `http://www.directhit.com/`.
17. P. Edwards, M. Steele, S. E. Clare, and D. M. Johns. Exploiting a web cache for user recommendations. In *Proceedings of Agents 2000 Workshop on Agent-Based Recommender Systems (WARS-2000)*, 2000.
18. Eurekster. `www.eurekster.com`, 2004.
19. Findory.com. Personalized news. `http://www.findory.com`, 2004.
20. G. Flake, E. Glover, S. Lawrence, and C. L. Giles. Extracting query modifications from nonlinear SVMs. In *International World Wide Web Conference*, pages 317–324, Honolulu, Hawaii, May 7–11 2002.
21. G. W. Flake, S. Lawrence, C. L. Giles, and F. Coetzee. Self-organization of the web and identification of communities. *IEEE Computer*, 35(3):66–71, 2002.
22. J. Freyne, B. Smyth, M. Coyle, E. Balfe, and P. Briggs. Further experiments on collaborative ranking in community-based web search. *Artificial Intelligence Review*, 2004. To appear.
23. D. Gibson, J. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia : Links, Objects, Time and Space — Structure in Hypermedia Systems*, pages 225–234. ACM Press, 1998.
24. E. J. Glover, S. Lawrence, M. D. Gordon, W. P. Birmingham, and C. L. Giles. Web search—your way. *Communications of the ACM*, 44(12):97–102, 2001.
25. Google Directory. The web organized by topic into categories. `http://directory.google.com/`.
26. Google Labs. Google personalized (beta). `http://labs.google.com/personalized`.
27. Google Local (beta). Find local businesses and services on the web. `http://local.google.com`.
28. D. Harman. Relevance feedback revisited. In *Proceedings of SIGIR92, the 15th ACM International Conference on Research and Development in Information Retrieval*, pages 1–10, 1992.
29. T. H. Haveliwala. Topic-sensitive PageRank. In *Proceedings of the Eleventh International World Wide Web Conference (WWW2002)*, Honolulu, Hawaii, May 2002.
30. G. Jeh and J. Widom. Scaling personalized web search. In *Proc. WWW 2003*, Budapest, Hungary, May 20-24 2002.
31. A. Jennings and H. Higuchi. A personal news service based on a user model neural network. *IEICE Transactions on Information and Systems*, 75(2):198–209, March 1992.
32. T. Joachims. A probabilistic analysis of the Rocchio algorithm with TF-IDF for text categorization. In *International Conference on Machine Learning*, pages 143–151, 1997.
33. T. Joachims, D. Freitag, and T. M. Mitchell. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 770–777. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1997.
34. D. Kelly and J. Teevan. Implicit feedback for inferring user preference: A bibliography. *SIGIR Forum*, 37(2):18–28, 2003.
35. Y. Khopkar, A. Spink, C. L. Giles, P. Shah, and S. Debnath. Search engine personalization: An exploratory study. *First Monday*, 8(7), July 2003. See `http://www.firstmonday.org/issues/issue8_7/khopkar/`.

36. M. W. Kim, E. J. Kim, and J. W. Ryu. A collaborative recommendation based on neural networks. In Y. Lee et al., editor, *Database Systems for Advanced Applications: 9th International Conference, DASFAA 2004*, volume 2973 of *LNCS*, pages 425–430. Springer-Verlag, 2004.
37. J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
38. A. Kritikopoulos and M. Sideri. The Compass Filter: Search engine result personalisation using web communities. In *Proceedings of the Workshop on Intelligent Techniques for Web Personalization (ITWP '03)*, Acapulco, Mexico, August 9-15 2003.
39. S. Kumar, L. Ertöz, S. Singhal, B. U. Oztekin, E. Han, and V. Kumar. Personalized profile based search interface with ranked and clustered display. Technical report, University of Minnesota, Department of Computer Science, 2001. `https://wwws.cs.umn.edu/tech_reports/listing/tr2001/01-023.pdf`.
40. S. Lawrence. Context in web search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.
41. H. Lieberman. Letizia: An agent that assists web browsing. In Chris S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 924–929, Montreal, Quebec, Canada, 1995. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.
42. H. Lieberman and T. Selker. Out of context: Computer systems that adapt to, and learn from, context. *IBM Systems Journal*, 39, 2000.
43. F. Liu, C. Yu, and W. Meng. Personalized web search for improving retrieval effectiveness. *IEEE Transactions on Knowledge and Data Engineering*, 16(1), January 2004.
44. J. Liu, C. K. Wong, and K. K. Hui. An adaptive user interface based on personalized learning. *IEEE Intelligent Systems*, 18(2):52–57, March/April 2003.
45. Lycos, Inc. HotBot. `http://www.hotbot.com`.
46. U. Manber, A. Patel, and J. Robison. Experience with personalization on Yahoo! *Communications of the ACM*, 43(8):35–39, August 2000.
47. X. Meng, Z. Chen, and A. Spink. A multiplicative gradient descent search algorithm fo user preference retrieval and its application to web search. In *Proceedings of the International Conference on Information Technology: Computers and Communications*, Las Vegas, Nevada, April 28-30 2003.
48. MSN. Newsbot (beta). `http://uk.newsbot.msn.com`, 2004.
49. Netscape. Open Directory Project. `http://dmoz.org`.
50. D. M. Nichols. Implicit ratings and filtering. In *Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering*, pages 31–36, Hungary, 1997.
51. M. J. Pazzani, J. Muramatsu, and D. Billsus. Syskill & Webert: Identifying interesting web sites. In *AAAI/IAAI, Vol. 1*, pages 54–61, 1996.
52. J. E. Pitkow, H. Schtze, T. A. Cass, R Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. M. Breuel. Personalized search. *Communications of the ACM*, 45(9):50–55, September 2002.
53. A. Pretschner and S. Gauch. Ontology based personalized search. In *Proceedings of the 11th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 391–398, Chicago, November 1999.
54. B. Smyth R. Rafter, K. Bradley. Automated collaborative filtering applications for online recruitment services. In *AH 2000*, pages 363–368, 2000.
55. M. Richardson and P. Domingos. The Intelligent Surfer: Probabilistic combination of link and content information in PageRank. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

56. J.J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System - Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Englewood Cliffs, NJ, 1971.
57. G. Salton, editor. *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ, 1971.
58. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating “word of mouth”. In *Proceedings of the CHI’95 Conference on Human Factors in Computing Systems*, pages 210–217, New York, 1995. ACM Press.
59. D. Sullivan. Eurekster launches personalized social search. *SearchEngineWatch*, January 21 2004.
See `http://searchenginewatch.com/searchday/article.php/3301481`.
60. A. V. Sunderam. Automated personalisation of internet news. In P. De Bra, P. Brusilovsky, and R. Conejo, editors, *Proceedings of Adaptive Hypertext 2002 (AH2002)*, volume 2347 of *LNCS*, pages 348–357. Springer-Verlag, 2002.
61. Topix.net. Pick your Topix, get the news. `http://www.topix.net/`.
62. Vivísimo Inc. Vivísimo clustering engine. `http://vivisimo.com`.
63. D. S. Weld, C. Anderson, P. Domingos, O. Etzioni, K. Gajos, T. Lau, and S. Wolfman. Automatically personalizing user interfaces. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI03)*, Acapulco, Mexico, August 2003.
64. R. W. White, J. M. Jose, and I. Ruthven. Using top-ranking sentences for web search result presentation. In *Poster Proceedings of the 12th International World Wide Web Conference (WWW 2003)*, Budapest, Hungary, May 20-24 2003.
65. R. W. White, I. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in web retrieval. In *24th BCS-IRSG European Colloqium on IR Research (ECIR 2002)*, Glasgow, Scotland, UK, March 25-27 2002. Springer.
66. S.K.M. Wong, Y.Y. Yao, G. Salton, and C. Buckley. Evaluation of an adaptive linear model. *Journal of the American Society for Information Science and Technology*, 42:723–730, 1991.
67. W. Xi, B. Zhang, Z. Chen, Y. Lu, S. Yan, W.Y. Ma, and E.A. Fox. Link Fusion: A unified link analysis framework for multi-type inter-related data objects. In *Proceedings of the Thirteenth International World Wide Web Conference (WWW2004)*, New York, U.S.A., 19-22 May 2004.
68. Yahoo! Inc. Yahoo! `http://www.yahoo.com`.
69. Yahoo! Inc. Yahoo! Local. `http://local.yahoo.com/`.
70. O. Zamir and O. Etzioni. Grouper: A dynamic clustering interface to Web search results. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1361–1374, 1999.
71. T. Zhu, R. Greiner, and G. Häubl. An effective complete-web recommender system. In *Proceeings of the Twelfth International World Wide Web Conference(WWW2003)*, Budapest, Hungary, May 20-24 2003.

The Compass Filter: Search Engine Result Personalization Using Web Communities

Apostolos Kritikopoulos and Martha Sideri

Dept. of Computer Science,
Athens University of Economics and Business,
Patision 76, Athens, T.K.10434, Greece
apostolos@kritikopoulos.info
sideri@aueb.gr

Abstract. We propose a simple approach to search engine personalization based on Web communities [14]. User information –in particular, the Web communities whose neighborhoods the user has selected in the past– is used to change the order of the returned search results. We present experimental evidence suggesting that our method indeed improves the quality of the ranking. Our experiments were carried out on a search engine created by our research group and focusing on the Greek fragment of the worldwide Web (1.33 million documents); we also discuss the issue of scaling.

1 Introduction

The worldwide Web has unprecedented size and diversity –both in terms of the *documents* it contains, and in terms of the *users* who access it and depend on it. While search engine technology has advanced tremendously, the criteria used in evaluating the relevance of a document to a particular query do not typically take into account the *user who asked this query* (his/her degree of sophistication, interests and preferences, as evidenced, for example, by the order in which s/he selected the preferred URLs, the groups of URLs that s/he has visited in the past, etc). There are, of course, related domains, such as recommendation systems [4,5,21] and push channel technology [23], in which personalization based on the user's declared or mined preferences is the supreme consideration. See also the next section for three recent approaches to personalization [2,15,17] based on PageRank [6,9,16,25].

Some of the most successful and elegant approaches to Web information retrieval are based on the realization of the importance of the link structure of the Web. In fact, two of the best known and most successful approaches to www information retrieval, Google's page rank [6,9,25] and Kleinberg's hubs and authorities [18] are in principle based exclusively of link structure.

The link structure of the Web has been, of course, the object of extensive study over the past five years [1,9,14,18,19,20]. One of the most interesting and intriguing observations in this study is the existence of abundant *Web communities* [14,20], that is, small sets of documents that are highly connected, (in other words, small-scale, consensual hubs and authorities in a very specialized subject). The importance of the Web communities to the structure and nature of the worldwide Web has often been emphasized [3,13,22].

Web communities are dense directed bipartite subgraphs of the web graph. A bipartite graph is a graph whose node set can be partitioned into sets, F and C. Every edge in the graph is directed from a node u in F to a node v in C. A bipartite graph is *dense* if many of the $|F| \cdot |C|$ possible edges between F and C are present; it is *complete* (or a *bipartite clique*) if all such edges are present. Without mathematically pinning down density, we proceed with the following hypothesis, proposed in [14]: the dense bipartite graphs that are signatures of web communities contain at least one core, where a core is a complete bipartite subgraph with i nodes from F and j nodes from C. Thus, the community core is an $i \times j$ complete bipartite subgraph of the community, for some small integers i and j greater than one.

In this paper we present a novel approach to search engine result personalization based on Web communities. Our method (Figure 1) filters the results of the search engine to a query, based on its analysis of the frequency with which the user asking the query has in the past visited or selected the (neighborhoods of the) various Web communities in the corpus.

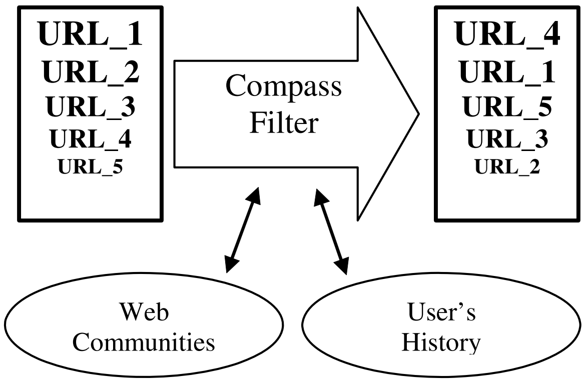


Fig. 1. Reordering the result set, using the Compass Filter

The main idea is as follows: We extract the Web communities (all $i \times j$ complete bipartite graphs in the corpus for all $i, j \geq 2$) and for each such "community core" we also determine its neighborhoods (the documents linked to, or from, documents in the community core; this is a little more general than the original proposal in [14]). When the search engine returns a set of documents in response to a query by the user, we re-evaluate these documents by taking into account the community neighborhoods in which they are involved (and exactly which part of the neighborhood they are involved), and the number of times these community neighborhoods have been visited or selected by the same user in the past. The results are then ordered in decreasing values of this personalized measure of relevance (the original order used to break ties), and presented to the user.

For a hypothetical illustrative example, consider the query "duck". The engine returns the following ranked Web pages, where the ranking depends only on the query terms and the corpus:

Table 1. First result set of the example

	URL	*MAIN THEME OF THE WEB SITE*
1	www.greek_natural_park.gr /crete/duck.htm	NATURAL PARK
2	www.gastronomy.gr/recipes /duck_potatoes.html	RECIPES
3	www.corfu_island.gr /local_animals/duck.html	ISLAND OF CORFU
4	www.ornithologic_home /duck_in_danger.htm	ECOLOGIC ORGANIZATION
5	**www.hunter.gr /duck_spots.html**	HUNTING
6	www.greek_encyclopedia /birds/duck.html	ENCYCLOPEDIA

From the history of the user, however, we know that the user had in the past clicked on a Web page (www.hunting_guns.gr/double-barrel/carbines.htm) that belongs to a 2x2 community (where the set F is are the url's on the first column, and the set C on the second, and in the corpus there are links from both entries of the first column to both entries of the second, see Table 2). We also have found that the fifth page (highlighted in Table 1) belongs to the same hyperlinked community (Table 2).

We next re-rank the result set, adding appropriate weights to the various web pages in a manner explained in Section 3, so that the Web pages appearing in communities, such as the fifth Web page in this example, ends up higher in the order of the presented Web pages:

Table 2. 2x2 Community of the example

2x2 COMMUNITY (main theme :HUNTING)	
www.hunting_guns.gr/double-barrel/carbines.htm	www.bird_chasing.gr/venues.html
www.hunter.gr/duck_spots.html	www.hunting_laws.gr/guns/limitations.html

Table 3. Final result set of the example

	URL	*MAIN THEME OF THE WEB SITE*
5	**www.hunter.gr /duck_spots.html**	**HUNTING**
1	www.greek_natural_park.gr /crete/duck.htm	NATURAL PARK
2	www.gastronomy.gr/recipes /duck_potatoes.html	RECIPES
3	www.corfu_island.gr /local_animals/duck.html	ISLAND OF CORFU
4	www.ornithologic_home /duck_in_danger.htm	ECOLOGIC ORGANIZATION
6	www.greek_encyclopedia /birds/duck.html	ENCYCLOPEDIA

We implemented our method on top of *SpiderWave* [27], a research search engine for the Greek fragment of the Web (about 1.33 million documents, basically the .gr domain) designed by our research group, which can be clicked from the Web site of our University (www.aueb.gr) as an alternative search engine.

SpiderWave totally resides on the server-side, and it was extended to include the capability of tracking the individual user profile (search and navigation history). We call this implementation of our idea "The Compass Filter" (for community-pass). In this paper we present some experimental results to evaluate our method.

Whenever a query is asked, our experiment engine flips a fair coin to decide whether the answer will be filtered through Compass or not. In either case we monitor the user's response (the results clicked, the order in which they were clicked, and the timing of the clicks –even though we do not use the latter data in our evaluation). We evaluate the user's response by a formula that rewards early clicking on high-ranking results, and penalizes extra clicks. Comparison between the three suites (the one without the Compass Filter, the one that was processed successfully by Compass and the one that was processed unsuccessfully due to the fact that the user had not visited any relevant communities in the past), followed by a statistical test, suggests that, our method significantly improves the quality of the returned results.

The main limitation of our experiments has been the difficulty to have our system used by enough users long and intensively enough so that the Compass Filter can intervene meaningfully (we believe that these are problems small academic research groups are bound to face, and they do not limit by themselves the applicability of our method). From over 450 users in the period April 2002 to February 2003, only 18 interacted long enough with the system so our method made a difference in the ranking of the results, and they asked a total of 44 queries. Still, a statistical test (see Section 5) indicates that the Compass Filter improves the quality of the user experience in a statistically significant way.

In the next section we describe recent approaches to personalization, in Section 3 we describe our method, in Sections 4 and 5 the experiments and the results, and in Section 6 the research directions suggested by this work.

2 Recent Approaches to Personalization

Recently, several methods for the personalization of web search engines have been proposed; we briefly review these advances in this section.

The method of Topic-Sensitive PageRank [15] proposes to compute a set of PageRank vectors, in principle one per user, each biased by a set of representative topics (which are taken from Open Directory [24]). By using these precomputed vectors, Topic-Sensitive PageRank generates query-specific importance scores for pages at query time. This technique is modified in [17] so that it scales well with the corpus size and the number of users, and can thus be feasibly implemented. "Partial vectors" are shared across multiple personalized views, and their computation and storage costs scale well with the number of views; on the other hand, incremental computation allows the calculation of personalized rankings at query time.

The ranking proposed in [2] also derives from PageRank; the difference is that it takes into consideration user preferences based on URL features such as Internet domains. For instance, a user might favour pages from a specific geographic region, or pages with topical features also captured in Internet domains, or documents from domains such as academic institutions in which pages are more likely to be monitored by experts for accuracy and quality. Users specify interest profiles as binary feature vectors where a feature corresponds to a DNS tree node or node set, and the method precomputes PageRank scores for each profile vector by assigning a weight to each URL based on the match between the URL and the profile features. A weighted PageRank vector is then computed based on URL weights, and used at query time to rank results.

Paper [3] presents and evaluates a novel ranking technique that combines the content of the objects being retrieved and the interest-based community of the user issuing the search. The theory of Bayesian belief networks is used as the unifying framework of the approach. Interest-based communities are groupings of users that share common interests. They are created using clickthrough data recorded in the form of user surfing behaviour. The method infers communities even for sources that do not explicitly show relationships between the pieces of information provided. The communities that are recognized are not necessarily based on the link information of the Web. Query contextualization is achieved by the juxtaposition of the current user interaction with a set of previous user interactions of all users in a way similar to collaborative filtering.

Other proposals for web search personalization in the recent literature include methods based on syntactic clustering of the Web [7], and on the recording of user preferences for meta-search engine personalization [8]. For two reviews on personalization see [10] and [11].

3 Description of the Method

Web communities are complete bipartite graphs of hyperlinks; the surprising prevalence of Web communities is an important and rather surprising property of the worldwide Web. For example, we shall see in the next section that in our crawl of .gr with 1,329,260 documents we found 1337 communities with a total 11,917 documents –roughly .9% of the crawl.

3.1 Step 1 (Preprocessing): Expand the Communities

We chose to "expand" the communities in a manner very similar with HITS [9,14,18]: we add to the set of the Web pages on either side of the original core community (S_{CG} – **C**ore **G**roup of community, see Figure 2), the group of pages that point to the core (S_{RG} – **R**eference **G**roup of community), and the group of pages that are pointed to by any page in the core (S_{IG} – **I**ndex **G**roup of community). From now on, we understand as "community" the union of the pages in $S_{CG,}$ S_{RG} and S_{IG}. This way, the 11,917 Web pages of the core communities were expanded 30fold to 348,826, almost 32% of the corpus.

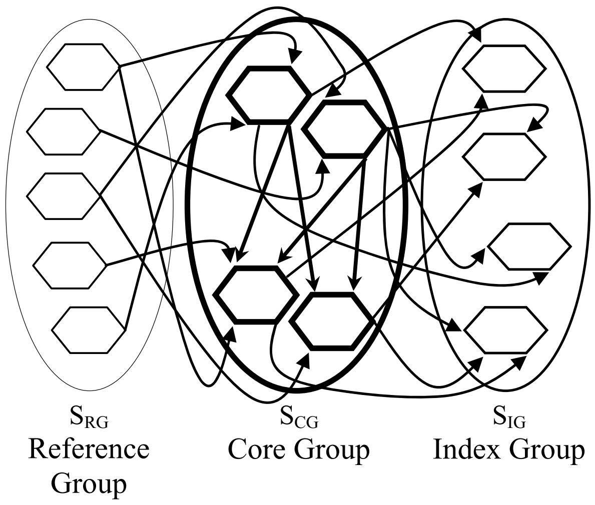


Fig. 2. Link graph of the community groups

3.2 Step 2: Calculate the Community Weights of the User

While a user clicks on query results, we monitor the core, reference, and index groups s/he visits, and we calculate, for each user and each community, the community weight of the user.

We noticed empirically that the influence on relevance of visits by the user to the core, index, and reference group of the same community decreases rapidly in this order. That is, if the user has visited the core group, everything else can be ignored, and if not the core but the index group, then visits to the reference group is not very significant unless they are extremely numerous. We capture this by the following formula, which appears (and is) rather arbitrary, but whose main point is that visits to $S_{\mathbf{CG}}$ are rewarded much more than those to $S_{\mathbf{IG}}$, and those to $S_{\mathbf{IG}}$ much more than those to $S_{\mathbf{RG}}$:

$$\text{COMMUNITY WEIGHT} = \begin{aligned}&(\text{Visits to the SRG Community}) + \\ &(2 * \text{Visits SIG Community})^2 + \\ &(3 * \text{visits to the SCG Community})^3\end{aligned} \tag{1}$$

3.3 Step 3: Reorder the Result Set

Given that the search engine has returned a ranked result set, we apply the outcome of the previous step, and we identify the URLs that belong to any of the expanded communities. The final weight of every URL is the sum of the weights (for the user) of each expanded community it belongs.

$$\text{Weight Of Url} = \begin{aligned}&\text{Sum of Weights of the Communities} \\ &\text{to which it Belongs .}\end{aligned} \tag{2}$$

Finally, we use this to reorder the result set in decreasing weight, using the original order to break ties.

EXAMPLE: A user has searched for the word "Asimov." In the original result set that the search engine produced, all documents from the Web site "www.altfactor.gr" (a leading Greek science fiction site) were ranked very low (31st place and below). Since www.altfactor.gr is part of a science fiction-based Web community, and the user has visited several sites that are referred to by sites of that community (even though s/he had not visited altfactor.gr itself), all pages from altfactor.gr were ranked highest by the Compass Filter.

4 Experimental Set-Up and Evaluation Metric

SpiderWave (http://spiderwave.aueb.gr) is a search engine research project whose aim is to determine the structure of the Greek Web (the .gr domain), and to use it as a test-bed for developing new ideas and methods of searching the Web. The crawl of the .gr domain was made with crawler software developed by a sister research group at the University of Patras [12]. The search engine is based on the ORACLE Intermedia Text processor (we also have implementation of HITS but we did not use it for this experiment). The result to every user's query is a ranked group of Web pages.

We used a process similar to that described in [20] to extract the communities of the Greek Web. This process starts by extracting all $i\times j$ communities (of *i* fans and *j* centers) in which the fans have outdegree exactly *j,* and the centers have indegree exactly *i.* A fan of degree j (pointing to j centers) is part of an $i\times j$ community if we can find $i-1$ other fans pointing to the same centers. For small values of i, j we can check this condition easily. After this first step, we enumerate all remaining communities iteratively as follows: for fixed *j* we find all vertices of outdegree at least *j* (all $1\times j$ communities), then we find all $2\times j$ communities by checking every fan which also cites any center in a $1\times j$, then we compute all $3\times j$ communities by checking every fan which cites any center in $2\times j$, and so on.

The community extraction process traced 1337 communities having in total 11917 Web pages, with dimensions varying from 2x2 to 2x12, and 8x2 to 8x8. Following the first step of the method, we expanded the communities and finally concluded with 1337 expanded communities containing a total of 348826 Web pages. Independently of their use in personalization, these communities seemed to us quite informative: by studying them we discovered that they summarize the "sociology" of the Greek Web, focusing on such diverse topics as Stock Market, Greek music, University issues, Linux, automobiles, literature and movies.

For the experiment we set up an extra interface to our search engine. We asked users to use a login name, which is used to trace each user's selection history. We explained that by doing so they participate in a search engine research project that will log their preferences, and will use them only for the purpose of improving their own search results. Anecdotal evidence tells us that the vast majority of users turned back at this point and selected the plain version. The history of each logged-in user (the weight of the user viz. all expanded communities) was updated with every selection of a document (it follows from the numbers above that roughly one in three clicks resulted in an update). In our early implementation we did the expansion of the communities on-demand, but we now have a full list of the expanded communities for our crawl, and we update it periodically.

Whenever the user asked a query, with probability 50% (the user was unaware of the results of this flip, or even that a flip was taking place), the results were filtered through Compass. The returned results, an ordered set of documents, reordered by Compass or not, were presented to the user, who proceeded to click some of them. We recorded the documents clicked on, and the order in which they were clicked (as well as the timing of each click, even though we did not use it in our evaluation formula).

We then evaluated the user's response using a metric we call SI (for Success Index), a number between 0 and 1:

$$\mathbf{SI} = \frac{1}{n}\sum_{t=1}^{n} \frac{n-t+1}{d_t * n} \tag{3}$$

where: **n** is the total number of the URLs selected by the user

$\mathbf{d_t}$ is the order in the list of the **t**-th URL selected by the user

The SI score rewards the clicking of high items early on. The reverse ranks of the items clicked are weight-averaged, with weights decreasing linearly from 1 down to 1/n with each click. For example, suppose n = 2 and the documents ranked 2 and 10 were clicked. If 2 is clicked first, then the SI score is bigger (27.5%); if second, smaller (17.5%). More controversially, SI penalizes many clicks; for example, the clicking order 2-1-3 has higher score than 1-2-3-4 (see the table below). Absence of clicks (the empty set) are scored zero –even though there were no such instances. Some examples of **$\mathbf{d_t}$** sequences and their SI scores:

Table 4. Examples of the SI score

Selection Order	*1*	*2*	*1*	*3*	*5*	*7*	*10*	*3*	*1*	*2*
SI score	*100%*	*42,59%*			*10,10%*			*38,88%*		

Table 5. Examples of the SI score

Selection Order	*1*	*2*	*3*	*4*	*4*	*3*	*2*	*1*	*5*	*8*	*7*	*2*	*1*
SI score	*40,10%*				*25%*				*15,71%*				

5 Experimental Results

General

Time period of the Experiment:	23 April 2002 - 21 February 2003
Number of logged-in users:	460
Number of users for which the Compass Filter changed the order in a query:	18

Group A) Queries in the control (no Compass Filter) group
(*Note*: These queries were randomly selected not to be treated by Compass)

Number of queries:	508
Average SI score:	48.58%
Variance:	13.98%

Group B) Queries in the group processed unsuccessfully, because Compass had no community information

Number of queries:	476
Average SI score:	46.29%
Variance:	13.01%

Group C) Queries in the group processed successfully by Compass Filter

Number of queries:	44
Average SI score:	57.70%
Variance:	9.86%

For group A the coin flip determined that the answer not be filtered. For groups B and C, in contrast, the engine tried to filter the results, but succeeded only for group C. The Compass changed the order of the results only for group C; users that their queries belonged to group A or B, didn't had the chance to see the results ordered by the Compass Filter.

Table 6. t-Tests Results

t-Test Results	
Groups to compare:	**P-value**
A and B	16.48% (>>5%)
A and **C**	3.74% (<5%)
B and **C**	1.35% (<5%)

Submitting these results to the t-Test (one-tailed) statistical analysis method (see Table 6) tells us that the observed difference between the means is significant, supporting the conclusion that the results of group C are substantial better that the results of the other two groups, and that our method appears to significantly improve the quality of the retrieved information.

6 Discussion

We have proposed a method for using communities to personalize and therefore enhance Web information retrieval, and a metric on click sequences for evaluating user satisfaction. Our experimental results are quite encouraging. Much more *experimental evaluation* of our method, as well as *tuning of its parameters* (especially the calculation of weights), is needed. Our SI metric could also use more refinement and justification.

Our way of extending the communities (not unlike that in Kleinberg's algorithm [18] and HITS [9,14]) results in a wealth of documents, but is not the only possibility. For example, a more modest approach would only include the documents pointing to the authorities (centers) and pointed to by the hubs (fans) as in [20]; the quality and relevance of the resulting group may compensate for the loss of volume. This is worth experimenting with.

We developed and tested our method in the context of a very modest fragment of the Web. This scaled-down experimentation and prototyping may be an interesting methodology for quickly testing information retrieval ideas, and for expanding the realm of research groups, especially academic groups lacking strong industrial contacts, that are in a position to conduct search engine research.

But does our method scale to the whole Web? It is based on the fact that Web communities seem to be prevalent in the Greek Web. Ravi Kumar et al. [20] report 191629 communities in a Web with 200,000,000 documents, comprising a total of 3823783 documents belonging to a community, or 1.91% of the whole (compared to our .9%). The degree structure of the Greek Web is not too different from the Web's, and so a 30fold increase by extending the communities is plausible in the Web as well. Hence, the user's clicking history would again present ample community information. The other premise on which the success of our approach depends is that, in the Greek Web, the queries asked by a user are apparently quite often relevant to the communities visited by the same user in the past. How this phenomenon scales is much harder to predict.

Finally, a very challenging question (for this and many other approaches to Web information retrieval) is to develop a realistic mathematical *user model*, predicting on the basis of few parameters the user's needs, expectations and behaviour. Such a model would help evaluate and optimize novel approaches to personalized information retrieval, and suggest more principled metrics for evaluating a search engine's performance.

Acknowledgments

We are grateful to Stelios Psarakis for his help with the statistical tests. We also thank Christos Amanatidis, Serafeim Karapatis, Kiriakos Tassopoulos, Giorgos Papachrysanthou and Christos Papadimitriou for many useful discussions.

References

1. Dimitris Achlioptas, Amos Fiat, Anna Karlin and Frank McSherry: Web Search via Hub Synthesis. Proc. Symp. On Foundations of Computer Science (2001)
2. Mehmet S. Aktas, Mehmet A. Nacar, Filippo Menczer: Personalizing PageRank Based on Domain Profiles. WebKDD 2004
3. Rodrigo B. Almeida, Virgilio A. F. Almeida: A Community-Aware Search Engine. WWW2004

4. Chumki Basu, Haym Hirsh and William Cohen. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, pages 714-720 (1998)
5. Daniel Billsus and Michael .J.Pazzani: Learning Collaborative Information Filters. In Proc. 15th International Conference on Machine Learning (1998)
6. Sergey Brin, Lawrence Page: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In Proceedings of the WWW7 Conference (1998)
7. A. Broder, S. Glassman, M. Manasse, and G. Zweig: Syntactic clustering of the web. In Sixth International World Wide Web Conference, pages 391-404, 1997
8. Lin Deng, Xiaoyong Chai, Qingzhao Tan, Wilfred Ng, Dik Lee: Spying Out Real User Preferences for Metasearch Engine Personalization. WebKDD 2004
9. Chris Ding, Xiaofeng He, Parry Husbands, Hongyuan Zha , Horst Simon: PageRank, HITS and a Unified Framework for Link Analysis. Lawrence Nerkeley National Lab Tech Report 49371 (www.nersc.gov/~cding.page.ps) (Nov. 2001)
10. S. T. Dumais (1999). Beyond content-based retrieval: Modeling domains, users and interaction. Keynote address at IEEE: Advances in Digital Libraries'99, May 19-21, 1999. Powerpoint slides
11. M.Eirinaki, M.Vazirgiannis, "Web Mining for Web Personalization", ACM Transactions on Internet Technologies (ACM TOIT), Vol.3 Issue 1
12. Final Report of Decision Making in Microeconomics using Data Mining and Optimization Techniques (Project PENED 99, under contract no. 99 ED232): General Secretariat for Research and Technology, Hellenic Ministry of Development, Greece (September 2001)
13. Gary Flake, Steve Lawrence, C. Lee Giles: Efficient Identification of Web Communities. In Proc. of the 6th ACM SIGKDD, pp.150-160 (August 2000)
14. David Gibson, Jon Kleinberg, Prabhakar Raghavan: Inferring Web communities from link topology. Proc. 9th ACM Conference on Hypertext and Hypermedia (1998)
15. Taher Haveliwala: Topic-sensitive PageRank. In Proceedings of the Eleventh International World Wide Web Conference (2002)
16. Taher Haveliwala, Sepandar Kamvar and Glen Jeh: An Analytical Comparison of Approaches to Personalizing PageRank. Stanford University Technical Report (2003)
17. Glen Jeh and Jennifer Widom: Scaling personalized web search. In Proceedings of the Twelfth International World Wide Web Conference (2003)
18. Jon M. Kleinberg.:Authoritative Sources in a Hyperlinked Environment. Journal of the ACM, 46(5):604-632 (1999)
19. Jon M. Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan and Andrew S. Tomkins: The Web as a graph: measurements, models and methods. Proceedings of the 5th International Computing and combinatorics Conference (1999)
20. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan and Andrew Tomkins: Trawling the web for emerging cyber-communities. WWW8 / Computer Networks, Vol 31, p1481-1493 (1999)
21. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan and Andrew Tomkins: Recommender systems: A probabilistic analysis. In Proc. 39th IEEE Symp. Foundations of Computer Science (1998)
22. Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D Sivakumar, Andrew Tomkins and Eli Upfal: Stochastic models for the Web graph. In Proceedings of the 41st Annual Symposium on Foundations of Computer Science, pp. 57-65 (2000)
23. Tie Liao: Global Information Broadcast: An Architecture for Internet Push Channels. IEEE Internet Computing, Volume 4, Issue 4:16–25, (July/August 2000)
24. The Open Directory Project: Web directory. http://www.dmoz.org/

25. Larry Page, Sergey Brin, R. Motwani, T. Winograd: The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford (Santa Barbara, CA 93106, January 1998). http://www.db.stanford.edu/~backrub/pageranksub.ps
26. Skiena, S: "Coloring Bipartite Graphs." §5.5.2 in Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica. Reading, MA: Addison-Wesley, p. 213, 1990
27. SpiderWave , http://spiderwave.aueb.gr

Predicting Web Information Content

Tingshao Zhu[1], Russ Greiner[1], Gerald Häubl[2], and Bob Price[1]

[1] Department of Computing Science,
University of Alberta, Canada T6G 2E1
{tszhu, greiner, price}@cs.ualberta.ca
http://www.web-ic.com
[2] School of Business, University of Alberta,
Canada T6G 2R6
Gerald.Haeubl@ualberta.ca

Abstract. This paper introduces a novel method for predicting the current information need of a web user from the content of the pages the user has visited and the actions the user has applied to these pages. This inference is based on a parameterized model of how the sequence of actions chosen by the user indicates the degree to which page content satisfies the user's information need. We show that the model parameters can be estimated using standard methods from a labelled corpus. Data from lab experiments demonstrate that the prediction model can effectively identify the information needs of new users, browsing previously unseen pages. The paper concludes with an overview of our "complete-web" recommendation system, *W*ebIC, which uses the prediction model to recommend useful pages to the user, from anywhere on the Web.

1 Introduction

While the World Wide Web contains a vast quantity of information, it is often difficult for web users to find the information they are seeking. It is natural to ask whether computational techniques can be used to assist the user in finding useful pages. Research in information retrieval techniques has answered this question definitively in the affirmative: Today, millions of users employ information retrieval techniques in the form of popular search engines to successfully find useful pages.

While present techniques have unquestionably made a considerable contribution to modern society, we observe that users must explicitly perform searches in order to benefit from these techniques and that users must have intuitions about what keywords they should use in these searches to efficiently circumscribe the information they are looking for within the now three billion web pages that search engines typically index. We believe, however, that the time has come for information systems to go beyond simple fulfillment of specific requests. We envision interfaces that anticipate the user's information needs and actively make suggestions for useful content.

Antecedents for the kind of systems we envision can be readily found in the information filtering and routing literature. These systems attempt to learn a user's preferences over time in order to filter a stream of ongoing information for a user. Examples include systems that filter email [9] and systems that select news articles [3]. Typically user judgements (or judgements of a class of related users) on past exemplars are used to build a model of user needs. This class of systems effectively predicts the user's information needs in as much as future information needs of the user are 'like' the user's past needs.

We observe, however, that the user's needs can change dramatically from day-to-day or even in the course of a single browsing session. As the user works on various tasks and subtasks, the user will often require information on various unrelated topics, including topics the user has investigated before. This motivates us to build a system that can predict the user's information need dynamically, based on the current browsing session.

The main contribution of the paper is a method for *passively* identifying the user's current information need from the pages the user visits and the actions that the user applies to the pages. Followed by the introduction of the related work, we describe a simple parameterized model which interprets the user's browsing actions as judgments about the relevance of page content to the user's current needs. A simple procedure is then explained for training the model. The following section describes a laboratory experiment which demonstrates the ability of the model to predict user information need. The next section follows up the laboratory experiment with an implementation of our ideas in the form of a stand alone web browser that can be run on a user's computer to provide on-line recommendations for any topic. We conclude with a discussion of our key contributions and insights.

2 Related Work

Pirolli and Fu [12] construct Information Need based on SNIF-ACT model, which uses production rules to predict the user's information need and then enlarges it through a *spreading activation network* which is derived from Tipster corpus. The production rules in SNIF-ACT are like the patterns that we are supposed to find through learning. Like SNIF-ACT our models uses prior accesses to predict future accesses. Our model, however, makes use of the information in the user's actions together and does not require any prior experience with words appearing on pages.

Blackmon et al. proposed Congnitive Walkthrough for the Web (CWW) [4] models limited WWW interaction. CWW uses Latent Semantic Analysis (LSA) to compute the similarity between goal statement and heading/link texts on web pages, rather than the subjective estimation in original Congnitive Walkthrough. In our case, we want to predict the representative goal, e.g., predicted IC-words, rather than the goal statement from the users.

Choo et al. [6] conducted a experiment to collect feedback from web users' ordinary work to build Information seeking mode for web user. The model that they obtained can predict which category the user belongs to, but it is still not applicable to locate IC-pages. In our research, we want to get the applicable model which can be used in real world applications.

Letizia [10] is an agent that helps the user browsing the Web, and it operates a best-first search augmented by heuristics inferring user interest from browsing behavior. Watson [5] observe users interact with everyday applications and then anticipate their information needs using heurictics, and then automatically form queries to information retrieval systems (e.g., search engines) to get the related information for user. While the heuristics used by Letizia and Watson may represent the user behavior very well, we suspect a model learned from user data, should produce a more accurate user model.

Our research identifies "Information Need" with the distribution over IC-words, which relates to the words that will be in the IC-page. Some other systems define the user's information need as a learned combination of a set of (possibly pre-defined) words — e.g., a NaïveBayes model [7] that classifies each webpage as IC or not, using a set of words as the features [3]. Our method differs as we do not limit the set of words, but instead label each individual word in the user's current session pages with a measure of its likelihood of appearing within an IC-page.

Our previous research of the IC prediction on URL and IC-word [16,**?**] has tested the accuracy of prediction based on "browsing feature", but the results on both papers only concern the training/testing of the classifier. This paper including additional analysis to further demonstrate the applicability of our model, and also to incorporate the model into a real complete-web recommendation system.

These above projects lead us to conclude that there exist general user model, the only problem is how to acquire such model, based on heuristics or learning from data. Models from experience may be very easy for people to understand and manage, but it is not a easy task to maintain such model, such as the updating. In our research, we propose to learn user model from real data, and we conduct user studies to obtain the training data, and using "browsing feature" to abstract the information to proper level that suitable for learning and the learned model is applicable in real application.

3 Web Browsing Behavior Model

Our goal is to build a "passive" system, which can recommend pages relevant to the user's *current* information need without burdening the user with intrusive questions. The challenge, therefore, is to identify relevant pages using only the information available to a web browsing program in the course of an ordinary browsing session. The notion of relevance is difficult to characterize formally, but we feel it is reasonable to believe that the notion could be defined empirically. We therefore apply a supervised machine-learning approach to the problem. Basically, we learn a model that predicts which pages will ultimately satisfy a user's information need from the pages the user has visited and the actions the user has chosen within the current browsing session.

We call pages that satisfy the user's information need *information content pages* or IC-pages for short. On web pages, much of the content of the page is communicated through the words on the page. While the subtle relations that make up the content of the page depend upon the precise grammatical roles of the words and considerable background knowledge, we can roughly approximate the content of the page simply by listing the words that appear on it. This list, or as it is more commonly known, bag of words is often sufficient for discriminating useful pages, that is IC-pages from unuseful or non IC-pages. We therefore call words on IC-pages, *information content words* or IC-words, as they form a crude approximation of the page content. The original goal of predicting IC-pages can then be reduced to the problem of predicting a sufficiently discriminating set of IC-words from the past page visits and actions of the user.

In traditional information filtering systems, the next step would be to collect training examples consisting of web pages and user judgments about the relevance of each page. The web pages would be turned into bags of words and a model would then be trained predict useful pages from the bags of words they contain. This works well when the user has a small set of static information needs. It can be reasonably argued that this is the case for problems like filtering emails or selecting interesting news articles. In task oriented behaviour, however, information needs change quickly and are hard to anticipate. It is therefore infeasible to have the user explicitly label pages as useful or unuseful for specific topics.

A little meta-reasoning can solve our problem. Instead of asking "What is the user's information need", we ask "How does the user signal her information need during a single browsing session?". We can answer the second question by examining the nature of browsing sessions. A typical browsing session has a regular structure. The user views a page and then chooses a browsing action. The action might be following a link, backing up to a previous page or terminating the session. If the session is not terminated, the action will result in a new page and the sequence continues. Within the sequence, we can interpret the user's actions as signals communicating the user's attitude towards the content of the pages the actions are applied to. For instance, backing up out of a page suggests the user did not find the page content useful, while following up a link on a page that leads to pages with similar content suggests that the prior page's content was useful. The sequence of pages the user visits and the actions the user applies to these pages therefore communicates information about the user's information need. Viewing actions as signalling information need solves two problems: the user's information need can be determined even when this need is novel and the signals can be collected unobtrusively by a passive system.

The representation developed for IC-pages above can also be applied to pages viewed earlier in the session. We call these pages *session pages* and the words that appear on them *session words*. As in the case of IC-pages, we view session words as representing the content of their respective pages. In the case of session words, however, we use a slightly richer representation which allows us to express the degree to which a given session word represents the content of its respective page. The representativeness of a session word is determined by features such as the frequency of the word on the page and any special roles assigned to the word such as appearances in titles, bold text or hyperlink anchors.

Some of these session words may also be IC-words. Without even considering the user's actions, we might speculate that session words that appear frequently throughout the pages of the session will also be IC-words. These "session-level" features can be useful for identifying broad themes. We make use of these session level features in our model, however, the key to our approach is explaining how the user's action choices signal which of the session words are IC-words. Some actions are easy to interpret. When a user clicks on a hyperlink, there is a high probability that session words appearing in the anchor text of the link are also IC-words. Other actions require more subtle inference. When the user backs up from a page, it might be interpreted as a judgment that the content of the page was not useful, or the user found what she was looking for and continued browsing. This action is not directed at specific words on the page, however, the significance of the action can be inferred from the difference in content between the page before the action and the page after the action. In general, for any action we can compare the bag of words associated with the proceeding and following pages and determine which words are retained, introduced or discarded. We can hypothesize that introduction of a word or the retention of a word signals that the word is an IC-word and that the omission of a word signals that the word is not an IC-word.

We can also apply background knowledge to interpret user actions. When the user views a search page, we can make use of the fact that links on the page are sorted in order. If the user skips over some items in an ordered list of options, we may infer that the user judged these links to be less attractive. If the user does not pursue links past a certain point in an ordered list, we might conclude that words in these links appear to be less relevant to her search task.

Together, the representativeness of a word on a session page, session level features of a word such as its frequency within the session, action signals associated with a word and specific background knowledge provide evidence about whether the word is an IC-word or not. We can think of each of these sources of information about the word as a feature of the word. In order to convey the idea that these features depend partly on the user's actions, we call them *browsing features*.

We have now shown that we can compute a simple set of features relevant to the question "How does the user signal her information need during a single browsing session?". At this point we can apply supervised machine learning to the problem. We collect many examples of browsing sessions and then split these sessions into session pages and terminating pages. In some cases, the session terminates in a useful IC-page. In other cases the session terminates in a non IC-page. For training purposes, we ask users to label each terminating page as an IC-page or a non IC-page. We represent the content of session pages by the session words they contain and the IC-page by the IC-words it contains. Finally, we calculate the browsing features of each session word. We then train a model that uses these browsing features to predict whether the session word is an IC-word or not. Intuitively, the trained model assigns weights to the different features which indicate how significant the feature is in signalling the user's information need. The trained model can then be applied to arbitrary browsing sessions to extract the information signalled by the user. This information need could then be used to recommend IC-pages to the user and hopefully shorten the user's search.

Consider the example suggested by Figure 1. Imagine the user needs information about marine animals. The user sees a page with links on "Dolphins" and "Whales" Clicking on the "Dolphins" link takes the user to a page about the NFL Football team. This page is not an IC-page for this user at this time, so the user "backs-up". We might conclude that the word "Football" which appears on the previous page, but not on the current page is not an IC-word.

The user then tries the "Whale" pointer, which links to a page whose title includes "whale", and which includes whales, oceans, and other marine terms in its content. The user then follows a link on that page to another with similar content terminating in a

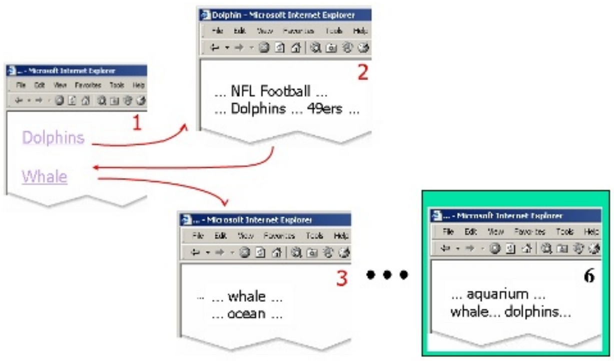


Fig. 1. Browsing Behavior to Locate IC-page

page about a local aquarium. We might conclude that words such as "Dolphin" and "Ocean" are IC-words.

Formally, we represent a browsing session (see [15] for an explanation of how sessions are identified from raw web logs) as a page-action sequence $\mathcal{S}$ = [(p_1, a_1),(p_2, a_2), ... , (p_n,a_n)] where p_i is page i in the session and a_i is the action applied to that page by the user. For each session word w appearing in the first $n-1$ pages, we define the role-action sequence $\mathcal{R}_w = [(R_1, a_1), \ldots, (R_{n-1}, a_{n-1})]$ where R_i is a vector of roles played by word w in page i (e.g., in the role-action sequence for "Dolphin", $R_{dolphin}$, the term $R_i = [title, plain]$ indicates that the word $Dolphins$ appears on page i in the title and once in the main text.). The browsing features of a word can be calculated directly from its role-action sequence. For instance we might want to define a feature that gives the number of times a word w appears in a user's session. Let $|R_i|$ represent the number of roles a word plays on page i and $R_{w,i,j}$ be the j^{th} role of word w on page i. Then the session level feature *appears* could be defined:

$$f_{\text{appear}}(\mathcal{R}_w) = \sum_{i<n} \sum_{j<|R_i|} \mathcal{R}_{w,i,j} .$$

In our experiments we define around twenty of these features (See [15] for list.). Additional examples include:

$f_1(\mathcal{R}_w) \equiv \{$number of appearance of w in title $\}$
$f_2(\mathcal{R}_w) \equiv \{$backed up from page with w $\}$
$f_3(\mathcal{R}_w) \equiv \{$appearances of w in a followed hyperlink $\}$
$f_4(\mathcal{R}_w) \equiv \{$appearances of w in plain text $\}$
...

We can think of the set of features as a vector function that produces the vector feature of a word for a given sequence:

$$F(\mathcal{R}_w) = [f_0(\mathcal{R}_w), f_1(\mathcal{R}_w), \ldots, f_k(\mathcal{R}_w)]$$

Let $\mathbb{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$ be the set of sessions collected for training. For each session, remove the last page p_n from the session to get the browsing session sequence

$\mathcal{S}'$. Let W be the session words in the browsing session pages (i.e., the union of all the words in the session pages minus the stop words). For each browsing session $\mathcal{S}'$ and word $w \in W$ we define the truncated role-action sequence $\mathcal{R}'_w$. Let $\mathbb{L}$ be the set of user labels for each of the sessions indicating whether each session terminated in an IC-page or not. Let the set of words appearing on terminating pages be $\mathbb{T}$. Each of the words appearing in a terminal page is then labelled as an IC-word or not IC-word. The training set consists of pairs generated as follows. For each word in the set of words on terminating pages $\mathbb{T}$, we create a training example consisting of the features $F(\mathcal{R}'_w)$ calculated on the role-action sequence for the corresponding session word and the label $IC - word$ or not $IC - word$ derived from $\mathbb{L}$. An illustrative example appears in Figure: 2.

Word	#title	#hyperlink ... #backup		IC-page
dolphin	0	2	1	Y
NFL	1	3	2	N
footbal	0	4	0	Y
⋮				

Fig. 2. Browsing Features of the IC-session

Given the features and an IC label for each word, we train one model to predict $\Pr(IC(w)|F(\mathcal{R}_w))$ for every word $w \in \mathbb{T}$.

4 Experiments

In this section we present an experiment performed in a laboratory setting to evaluate the feasibility of training an IC-word prediction model from labelled data.

4.1 User Studies

Web browsing sessions were collected from students at The School of Business at the University of Alberta. Each subject was asked to list three novel vacation destinations they had never visited before and plan a *detailed* vacation to each destination specifying travel dates, flight numbers, accomodation, activities, etc. The task was chosen to provide subjects with concrete information needs, a reason to consult a variety of web pages on various topics and and to favour the use of web sites with static content that our tools can easily parse. The subjects were instructed to use a specially augmented web-browser (AIE; see Section 5.1), to locate the information they needed on the world wide web. The browser logs the web pages the subject views and allows the user to explicitly mark terminating pages as IC-pages or not IC-pages. In all other respects, the browser performs the same as a standard web browser permitting unrestricted access to the complete web. Each participant was given about 45 minutes. Subjects were informed that two randomly selected participants would win $500 towards the specific vacation they had planned.

Table 1. Number of Requests vs Percentage of the URLs

Number of Request(s)	Percentage of the URLs
1	58.93%
2	23.46%
3	7.63%
4	4.08%
5	1.85%
6	1.16%
7	0.88%
8	0.40%
9	0.40%
10	0.18%
...	...

The study included 114 subjects. Subjects requested 15,105 pages of which 1,887 pages were labelled as IC-pages for an average of 14.63 IC-pages per participant. There were 5,995 distinct URLs, meaning each URL was requested 2.52 times on average. Of these, 3,039 pages were search pages (from 11 different search engines); if we ignore these, we find that each non-search-engine page was visited 2.02 times on average.

Table 1 shows how often each page was visited; notice 82.39% of the URLs were visited only one or two times. Clearly very few URLs had strong support in this dataset. Building a recommendation system based on correlations among users, association rule [1] or sequential pattern [2] would be difficult.

4.2 Empirical Results

According to the labelled log data, only 12.5% of the pages were IC-pages. Similarly, only 9.15% of the words (105,376 of 1,152,442) were IC-word.s To deal with this imbalanced dataset, we have tried both down-sampling [11] and over-sampling [8], and found that down-sampling produced more accurate classifiers than over-sampling.

We then down-sampled the Non-IC-word instances to obtain the balanced training data. The training data was then used with two different model learners: a NaïveBayes network [7] and a decision tree C4.5 (see [13]). The NaïveBayes network makes the assumption that the attributes are independent of one another, conditioned on the class label. The training examples were prepared from the raw data as described in Section 3.

Both of these models output a probability that a session word is an IC-wordgiven the session word's browsing features. For testing purposes we considered any prediction > 0.5 to be a classification of a word as an IC-word.

Our main results were generated using tenfold cross-validation. Instances of both models were repeatedly trained on %90 of the data and tested on the remaining %10. The process was repeated ten times with a different split of the data each time. The results were averaged over the ten trials.

In order to examine how well models learned from one subject generalize to other subjects, we performed a series of "leave-a-subject-out" tests in which both models were trained on data from all subjects but one and then tested on the data for the remaining subject. We repeated the process leaving out a different subject each time.

We evaluated the performance of the two models under both of the above conditions using three standard measures from the information retrieval literature: precision, recall and F-Measure [14]. We give the definitions below:

$$\begin{aligned} \text{ICprecision}(S,\ell) &= \frac{|WP(S_\ell)\bigcap W(s_N)|}{|WP(S_\ell)|} \\ \text{ICrecall}(S,\ell) &= \frac{|WP(S_\ell)\bigcap W(s_N)|}{|W(s_N)|} \\ \text{F}(S,\ell) &= \frac{2\times\text{ICprecision}(S,\ell)\times\text{ICrecall}(S,\ell)}{\text{ICprecision}(S,\ell)+\text{ICrecall}(S,\ell)} \end{aligned} \quad (1)$$

The results appear in Table 2. The first row gives precision, recall and F-measure for the cross validation tests across all subjects. The line is designated “10-fold”. Each measure is shown with its mean and standard deviation. In the first two rows of the table, the models are trained on all subjects. The high precision and recall of C4.5 under these conditions suggest that there is some commonality across users, which our algorithm is finding.

The accuracy of C4.5 is much better than that of NaïveBayes, which is about 80% versus 65%. We conjecture two possible reasons for C4.5’s superior performance. First, C4.5 uses *local* discretization of integer attributes, whereas NaïveBayes used global approach. Second, C4.5 does implicit selection relevant features through its tree splitting and pruning operations.

The second two rows of the table show the results of the two models trained with a leave-one-subject-out protocol. The results under these conditions are distinctly inferior to the model trained on all subjects. On the positive side, C 4.5 is still producing useful recommendations showing that significant generalization across users is possible. The distinct drop in performance, however, leads us to conclude that there is some variation in browsing behaviours between individuals. In a deployed application, the most cost-effective model might be to combine a prior model generated from a pool of generic subjects with a small amount of data learned directly from a specific user.

In the model we have presented, the predictor is given all of the prior session pages and asked to predict the words that will appear on the IC-page. In a deployed application, it would be useful to predict the IC-page very early in the search process from the first few pages. Early predictions could then be used by a web browser to locate pages for the user much earlier than the user would locate them and thereby save the user considerable effort. In a second experiment we evaluate the ability of the model to predict IC-words from a subset of the session pages.

Table 2. Precision/Recall of IC-word Prediction

		Precision	Recall
10-fold	C4.5	0.84 ±0.07	0.80 ±0.08
	NB	0.66 ±0.12	0.53 ±0.25
LSO	C4.5	0.74 ±0.05	0.66 ±0.17
	NB	0.55 ±0.12	0.67 ±0.30

Table 3. NaïveBayes Average F-Measure Measurement Matrix for Leave One Out Evaluation

	Prefix length ℓ				
h	1	2	3	4	≥ 5
1-10	26.50	24.51	21.21	20.06	15.15
11-15	18.99	20.21	19.09	20.94	16.66
16-20	26.57	26.27	26.08	21.79	14.57
21-25	23.33	25.13	19.99	20.21	11.82
>25	10.32	10.04	6.61	6.49	6.54

Table 4. C4.5 Average F-Measure Measurement Matrix for Leave One Out Evaluation

	Prefix length ℓ				
h	1	2	3	4	≥ 5
1-10	37.80	45.92	54.15	49.58	29.51
11-15	40.20	47.43	38.24	30.96	22.67
16-20	46.72	39.67	21.72	14.23	18.73
21-25	23.41	27.04	26.11	24.49	13.04
>25	11.22	11.22	11.22	10.18	6.98

Recall that a session can be represented as a page-action sequence S = [(p_1, a_1), (p_2,a_2), ..., (p_N,a_n)] of length N (where (p_N, a_N) is the IC-page). In the experiments above, the models were trained on the session pages [(p_1,a_1), (p_2,a_2), ..., (p_{N-1},a_{N-1})]. We wish to predict the IC-page from a prefix of the session pages. Let the prefix S^ℓ = [(p_1,a_1),(p_2,a_2), ..., (p_ℓ,a_ℓ)] be the first consecutive ℓ pages. We then train both of our models on data sets generated from these prefixes for $\ell = \{0, 1, \ldots, N-1\}$. The total length of user sessions varies considerably from session to session. Sessions range in length from under ten pages to well over twenty-five pages. In a session with ten pages, a model based on a prefix of length five will be predicting five pages ahead. In a sequence of length twenty-five a model based on a prefix of length five will be predicting twenty pages ahead. In order to get a better idea of how far ahead the model is predicting, we define h to be the *horizon* of prediction. We define h to be the number of pages from the prefix to the actual IC-page.

All results for our long-range prediction experiment are based on leave-one-subject-out testing. We present the results for naïve Bayes in Table 3 and C4.5 in Table 4.

When we compare the results for NaïveBayes in Table 3 to the results for C4.5 in Table 4 we see that C4.5 is again the clear winner. Within each table we see that the F-score decreases as ℓ increases — i.e., as more pages are used to define the context. The possible reason is that when more pages are been observed, it will also introduce much noise to the learner. When one considers that many of the IC-words are not even present in the limited prefix sessions, the performance is quite impressive.

As a final observation about these data: When we compared the training data with the associated testing data, we found that only 30.77% of IC-pages in the testing data appeared anywhere in the training data; and that the average support for these in-training IC-pages is only 0.269. This is why we cannot use standard recommendation systems that only use page frequency: about 70% of the IC-pages would never be recommended, and even for the remaining 30%, there is only a small chance that they would be selected as recommendations; see below.

5 Applications

5.1 AIE: Annotation Internet Explorer

To enable us to collect the IC information, we built an enhanced version of Microsoft Internet Explorer, called AIE (shown in Figure 3), which we installed on all computers in the lab we used for our study.

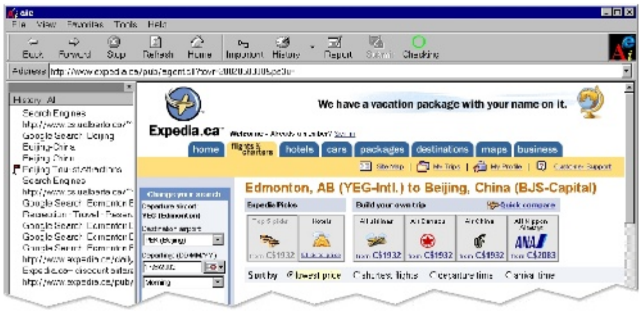
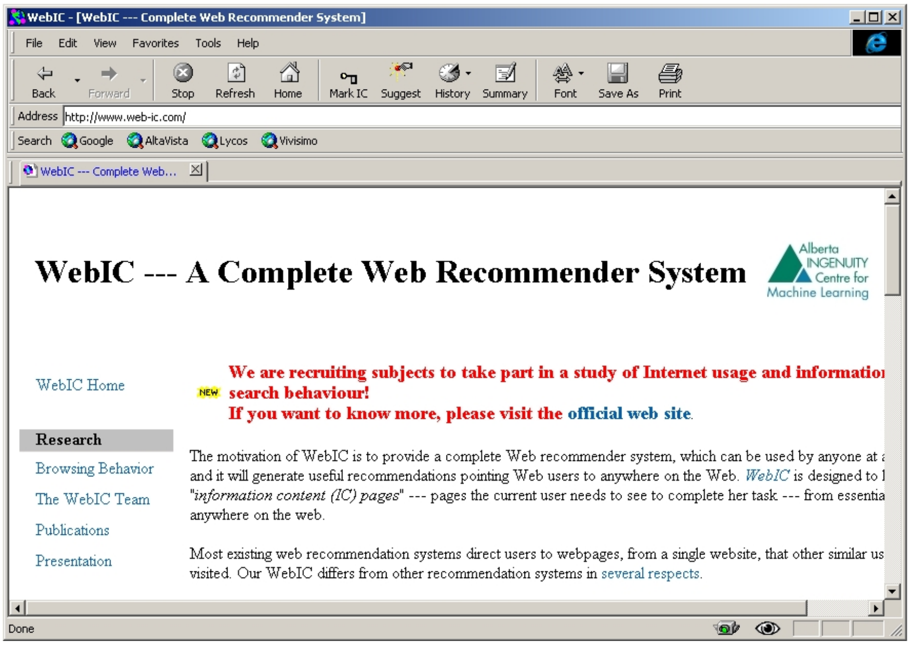

File View Favorites Tools Help
Back Forward Stop Refresh Home Important History Report Checking
Expedia.ca
We have a vacation package with your name on it.
home flights & charters hotels cars packages destinations maps business
Edmonton, AB (YEG-Intl.) to Beijing, China (BJS-Capital)

Fig. 3. The AIE Browser (top portion)

As with all browsers, the user can see the current web page. This tool incorporates several relevant extensions — see the toolbar across the top of Figure 3. The user can declare the current page to be "important" (i.e., an IC-page), by clicking the *Important* button on the top bar. Our paper [15] gives more details on AIE.

5.2 Overview of *W*ebIC

*W*ebIC is a client-side, browser based on the Microsoft internet explorer. The recommendation system is shown in Fig. 4. It uses determines IC-pages using a trained model of user browsing patterns from previously annotated web logs. *W*ebIC computes browsing properties for essentially all of the words that appear in any of the observed pages, and then use the model to predict the user's current information need: a list of word-probability pairs $\{w, p(w)\}$, where $p(w)$ estimates the probability that the word w will be an IC-word. There are two ways that *W*ebIC could use this information: First, it could "scout ahead": follow the outward links from the current page (recursively, in a breadth-first fashion) seeking pages that include many of these IC-words. It would then recommend such IC-word-rich pages to the user. Alternatively, *W*ebIC could send an appropriate query to a search engine (e.g., Google), then possibly scout forward from the pages returned.


WebIC - [WebIC --- Complete Web Recommender System]
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Mark IC Suggest History Summary Font Save As Print
Address http://www.web-ic.com/
Search Google AltaVista Lycos Vivisimo
WebIC --- Complete Web...
Alberta INGENUITY Centre for Machine Learning
WebIC Home
Research
Browsing Behavior
The WebIC Team
Publications
Presentation
NEW
Done

WebIC --- A Complete Web Recommender System

We are recruiting subjects to take part in a study of Internet usage and informatio
search behaviour!
If you want to know more, please visit the official web site.

The motivation of WebIC is to provide a complete Web recommender system, which can be used by anyone at
and it will generate useful recommendations pointing Web users to anywhere on the Web. *WebIC* is designed to
"*information content (IC) pages*" --- pages the current user needs to see to complete her task --- from essentia
anywhere on the web.

Most existing web recommendation systems direct users to webpages, from a single website, that other similar us
visited. Our WebIC differs from other recommendation systems in several respects.

Fig. 4. *W*ebIC — A Complete Web Recommender System

There are two phases involved in the whole process:

Modeling (Training). In this phase, we will build a “browsing behavior model” for the web user. The input may be the annotated web logs from this user, or from the user community that she belongs to, or the web logs we collected from any other people. In case of the training on her own web logs, we can obtain a learner that specified to her, thus we can provide more realistic personalized service. It is expected that the quality of service is down from the self-trained model to the community-based model, and the general model from any other people cannot compete with two other models.

Recommendation. For the recommendation generation, after watching the user’s click stream (*without annotation*): at first, extract the properties of the words involved in the click stream, then apply the extracted patterns to predict which words will be IC-words. After the IC-words prediction, *W*ebIC can start a web crawler to find the pages matching the predicted IC-words, or send a synthesized query to a search engine, such as Yahoo.com. Since the model is built on the “browsing property” of the involved words, and not the words themselves, it can be used on any web environment. *W*ebIC can predict the IC-words no matter where the user is or what she is working on.

6 Conclusion and Future Work

Our browsing behaviour model identifies relevant IC-words based on the browsing features, and the model is independent of any particular words or domain. Although we can train a personalized model, the patterns are largely user-independent. Our feature-based model uses a unique source of information, i.e., browsing behavior feature, to provide recommendations when other paradigms cannot. It can be implemented in a practical and useful application.

Correlation-based recommenders point users to pages other users visit - not necessarily to pages other users found useful or that will be useful to the current user. Content-based recommenders that learn content models of a specific web site or set of sites cannot recommend pages for other sites. Browsing pattern-based recommenders lack leverage provided by content or peer knowledge but tap into a new source of knowledge that works for any content and user.

We are currently investigating more effective ways to predict IC-words so that training patterns have a better target. We might incorporated timing information into the prediction, since it is commonly used to indicate the user's interest on a page. In *W*ebIC, using a crawler instead of search engine might preserve context, but we should figure out how to handle data accessed through forms. We could develop other features of the IC-session(e.g., other page content information, etc), combining content, peer and browsing knowledge.

We are also exploring the best way to connect our with a scouting system and/or multiple search engines, and perhaps yet other ways to provide specific page recommendations to the user. We plan to explore Natural Language processing systems to extend the range of our IC-words, and other machine learning algorithms to make better predictions, and help us cope better with our imbalanced dataset. We also would like to do further tests of *W*ebIC on other domains and larger user pools.

Acknowledgement

The authors gratefully acknowledges the generous support from Canada's Natural Science and Engineering Research Council, the Alberta Ingenuity Centre for Machine Learning (http://www.aicml.ca), and the Social Sciences and Humanities Research Council of Canada Initiative on the New Economy Research Alliances Program (SSHRC 538-2002-1013).

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th Int'l Conference on Very Large Databases (VLDB'94)*, Santiago, Chile, Sep 1994.
2. R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. of the Int'l Conference on Data Engineering (ICDE)*, Taipei, Taiwan, Mar 1995.
3. D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling (UM '99)*, Banff, Canada, 1999.

4. M. Blackmon, P. Polson, M. Kitajima, and C. Lewis. Cognitive walkthrough for the web. In *2002 ACM conference on human factors in computing systems (CHI'2002)*, pages 463–470, 2002.
5. Jay Budzik and Kristian Hammond. Watson: Anticipating and contextualizing information needs. In *Proceedings of 62nd Annual Meeting of the American Society for Information Science*, Medford, NJ, 1999.
6. Chun Wei Choo, Brian Detlor, and Don Turnbull. A behavioral model of information seeking on the web – preliminary results of a study of how managers and it specialists use the web. In Cecilia Preston, editor, *Proceedings of the 61st Annual Meeting of the American Society for Information Science*, pages 290–302, Pittsburgh, PA, Oct 1998.
7. Richard Duda and Peter Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
8. N. Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI '2000)*, 2000.
9. David Lewis and Kimberly Knowles. Threading electronic mail: A preliminary study. *Information Processing and Management*, 33(2):209–217, 1997.
10. H. Lieberman. Letizia: An agent that assists web browsing. In *International Joint Conference on Artificial Intelligence*, Montreal, Canada, Aug 1995.
11. C. Ling and C. Li. Data mining for direct marketing problems and solutions. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, New York, NY, 1998. AAAI Press.
12. P. Pirolli and W. Fu. Snif-act: A model of information foraging on the world wide web. In *Ninth International Conference on User Modeling*, Johnstown, PA, 2003.
13. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, 1992.
14. C. Rijsbergen. *Information Retrieval*. 2nd edition, London, Butterworths, 1979.
15. Tingshao Zhu, Russ Greiner, and Gerald Häubl. An effective complete-web recommender system. In *The Twelfth International World Wide Web Conference(WWW2003)*, Budapest, HUNGARY, May 2003.
16. Tingshao Zhu, Russ Greiner, and Gerald Häubl. Learning a model of a web user's interests. In *The 9th International Conference on User Modeling(UM2003)*, Johnstown, USA, June 2003.

Mobile Portal Personalization: Tools and Techniques[*]

Barry Smyth, Kevin McCarthy, and James Reilly

Adaptive Information Cluster, Smart Media Institute,
Department of Computer Science,
University College Dublin
Belfield, Dublin 4, Ireland
barry.smyth@ucd.ie

Abstract. The usability of mobile portals has been a major stumbling block since the advent of the mobile Internet and WAP handsets. Indeed poor usability is cited as a major contributing factor to the poor take-up of mobile Internet services amongst consumers. A key problem relates to the amount of time that users spend navigating to content as they browse mobile portals. Recent advances in personalization technology have the potential to solve this problem, and today a number of leading operators already provide their users with access to intelligent portals that are automatically personalized based on subscriber usage patterns. In this chapter, we examine this so-called *personalized navigation* technology and describe how it has been used to significantly enhance the usability of leading mobile portals. In addition we consider ways in which this approach to personalization may be enhanced by combining structural properties of a mobile portal (such as the distance to content sites) with the access probabilities of users. We demonstrate that although such distance factors have proven successful in Web personalization, they are less beneficial when it comes to the personalization of mobile portals.

1 Introduction

The mobile Internet (MI) has failed to live up to end-user expectations. Limited bandwidth, unreliable handsets, patchy content and poor usability have all contributed to this state of affairs. And although recent developments have seen significant improvements in bandwidth, handsets and content, usability remains a problem, particularly in relation to the navigation effort faced by users when searching for content in a typical mobile portal. For example, recent studies have highlighted how content services are usually positioned to be more than 16 *clicks* from the portal home page. In other words, to access a typical content service a user can expect to have to make 16 clicks on their mobile phone as they navigate through the portal, scrolling through menus and selecting options enroute [1]. The result is low levels of satisfaction from end-users and lackluster usage levels for mobile operators.

[*] The support of the Science Foundation Ireland is gratefully acknowledged.

Recently, however, a compelling solution has emerged that has been proven to have a dramatic impact on portal usability by significantly reducing the above navigation problem. This so-called *personalized navigation* solution applies personalization techniques to the navigation task. That is, instead of recommending individual content items to users, menus and menu options are recommended in such a way that users require an average of 50% fewer clicks to locate content, leading to significant increases in mobile usage [2, 3]. Very briefly, this personalized navigation approach estimates $P_u(o|m)$, the probability that a given user, u, currently in menu m, is looking for menu, o. Menu options are promoted to the user based on their past access probabilities.

In this chapter we focus on this probabilistic personalized navigation technique in the context of mobile portal personalization. We review the basic approach taken and the impact that this has had on portal usability. In addition, we investigate whether or not performance improvements can be achieved by extending the basic probabilistic personalization model to take into account an options's distance from the current menu as well as its access probability. For example, consider two menu options, o_1 and o_2, both with the same access probabilities; that is, $P_u(o_1|m)$=$P_u(o_2|m)$. But suppose that the distance from m to o_1 is greater than the distance from m to o_2, that is $Distance(m, o_1) > Distance(m, o_2)$, then shouldn't o_1 be promoted ahead of o_2 because if correct a greater number of navigation clicks will have been saved? We call this *distance-biased promotion* and it clearly has the *potential* to improve the degree to which personalized navigation can save a user navigation effort. In fact evidence from Web personalization suggests that such an extension is likely to pay dividends [4]. However, we are conscious that traditional Web-based portals and current mobile portals are very different and what works on the Web does not always translate well to the mobile space. With this in mind, in this chapter we also examine the potential of distance-biased techniques in the personalization of mobile portals.

In the next section we discuss the background to this research focusing in particular on recent developments in the mobile Internet and outlining past research related to the issue of navigation effort. Section 3 provides a review of our core personalized navigation strategy as detailed in [2, 3]. Section 4 describes the click-distance model of navigation effort and explains how this can be used to bias personalized navigation with respect to portal distance. Finally, before concluding, in Section 5 we describe a recent evaluation to investigate the benefits of this distance-biased technique, based on a large-scale European portal and live user activity logs.

2 The Mobile Internet

The mobile Internet refers to the delivery of data services across wireless networks for Internet-enabled handsets as implemented through a group of related infrastructure, protocol and device technologies. It allows the end-user to access various types of data services from their mobile handsets, including Web-style information content, email services, games etc. Access devices range from limited, first-generation WAP (Wireless Application Protocol, see www.wapforum.org) phones to today's sophisticated PDAs (Personal Digital Assistants) and so-called SmartPhones (see www.microsoft.com/smartphone).

In the past the usability of mobile services has been compromised by limited device functionality, bandwidth, and content. Fortunately the new generation of mobile services (so-called 2.5G services) represents a significant improvement. The major bandwidth and content issues have largely been resolved, and the latest phones offer users significant interface and functionality improvements over earlier models. However, key portal usability problems remain, due to poor mobile portal design. Users find that they are spending too much of their time navigating to content because mobile portals are designed as fixed, complex hierarchies of menu options.

2.1 Mobile Internet Devices

One of the most important features of the mobile Internet relates to the degree to which existing consumer devices (WAP phones, for example) represent a significant step backwards in terms of their functionality, at least when compared to the traditional Internet device (the desktop PC or laptop). In particular, presentation and input capabilities tend to be extremely limited on most mobile devices. For instance, a typical desktop PC, with a screen size of 1024x768 pixels, offers more than 10 times the screen real-estate of a PDA, and more than 20 times the screen space of second-generation Internet phones (eg. I-mode and Vodafone Live! handsets or Microsoft's SmartPhone).

Mobile handsets are further limited in their ability to receive user input. The keyboard and mouse functionality of a modern PC are usually absent. Moreover,

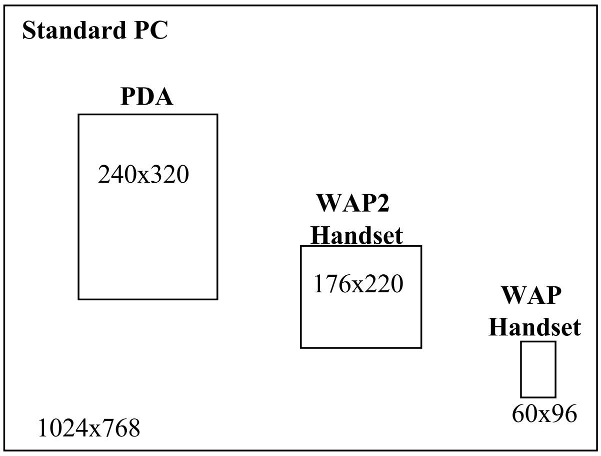


Fig. 1. A typical desktop PC, with a screen size of 1024x768 pixels, offers more than 10 times the screen real-estate of a PDA, and more than 20 times the screen space of second-generation Internet phones

the mobile phone numeric keypad makes it extremely difficult for user to input any quantity of information. The popularity of ‘texting’ aside mobile phones are not well adapted for text entry and are certainly not designed to make it easy for users to enter URLs, for example. From a mobile Internet viewpoint, these devices restrict selection features to simple scroll and select keys that allow the user to scroll through menu lists and perform selections. Some improvements are present in most PDAs, which tend to offer touch sensitive screens that are easier to manipulate. More recently, handset manufacturers have been adding QWERTY keyboards to some of the higher-end devices, although these keyboards are far from full-size and can only be used by the more dexterous users. In the main, data input remains difficult at best and this is likely to remain the case for some time to come.

2.2 Mobile Information Access

These differences (input and output capabilities) that exist between mobile handsets and more traditional Internet devices, such as PCs and laptops, directly influence the manner in which users access information using these devices. For example, on the Internet today search has largely become the primary mode of information access. It is relatively easy for users to input search queries and search engines have improved significantly in their ability to respond intelligently to user needs. In addition the large screen sizes make it feasible for users to efficiently parse the long lists of search results returned. In contrast, search is far more problematic on mobile devices. Entering queries is simply too time consuming and complex for the average user to tolerate and small screen sizes make it practically impossible for users to easily process the result lists returned. As a result, browsing is the primary mode of information access on the mobile Internet. Instead of searching for information, users attempt to navigate to information by using mobile portals. Today the vast majority of mobile Internet services are accessed via an operator portal with direct search constituting a small fraction (<10%) of mobile Internet activity.

This distinction between alternative modes of information access on the mobile and fixed Internet is an important one and it sets the scene for our own research. The bottom line is that to help users to locate information and services more effectively on the mobile Internet we must attempt to improve the efficiency of mobile portal browsing or navigation.

2.3 Mobile Portal Navigation

Mobile portals are examples of hierarchical menu systems (HMS), and long before the arrival of the mobile Internet different forms of hierarchical menu systems were studied extensively with respect to their general usability and navigation characteristics [5, 6, 7, 8, 9, 10, 11, 12, 13]. Very briefly, much of this research has focused on the structural properties of hierarchical menu systems, for example their depth and width, as they relate to the ability of a user to easily navigate through the HMS. For example, [9] discovered that for moderate sized menu systems, wide hierarchies are preferable to deep hierarchies due to the short-term

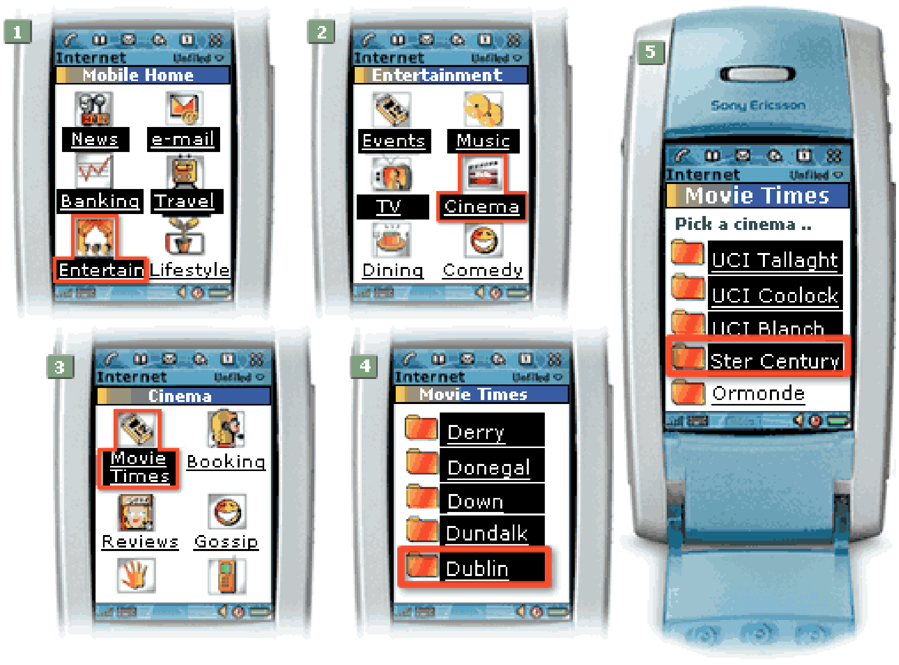


Fig. 2. In this sample portal the user must navigate through a series of menu pages to locate their local cinema

memory limitations of end users, which led to a greater number of navigation errors in deep hierarchies; the interested reader is also referred to [6, 10] for related work. Similar observations have been made with respect to the menu hierarchies found in the World-Wide Web [13]. Thus the evidence suggests that the complexity of a hierarchical menu system has a significant impact on its usability and the ability of users to navigate through menu levels. The type of menu hierarchies found on the mobile internet are likely to be subject to similar findings. Indeed the scale of the navigation problem associated with mobile portals today, and the mismatch between user expectations and realities, is highlighted by a number of recent studies. For example, one study claims that while the average user expects to be able to access content within 30 seconds, the reality is closer to 150 seconds [14]. For instance, Figure 2 presents a typical navigation scenario in which a mobile user must navigate through 4 levels of menus, and make 11 separate scrolls, in order to get from the portal home page to her local cinema listings. Of course the time that it takes a user to access a content item is a useful measure of navigation effort and we suggest that the navigation effort associated with an item of content depends critically on the location of that item within the portal structure, and specifically on the number of navigation steps (scrolls and selects) that are required in order to locate and access this item from a given starting position within the portal (typically the portal home page). We will return to this idea in the next section when we introduce the *click-distance* model of navigation effort. We show how it can be used to guide and evaluate the personalization of a portal.

3 A Probabilistic Model of Personalized Navigation

The basic idea behind personalized navigation is that instead of presenting each user with a fixed portal hierarchy, each user is presented with a hierarchy that has been adapted to his or her needs. By *adapted* we mean that individual menu options may be promoted within the portal so that they are more accessible to relevant users. For example, menu options may be reordered within a menu or they may even be promoted from lower levels of the portal to higher levels [4, 15, 16, 17, 18, 19, 20, 3, 2]. In other words, each time a user accesses a given menu, m, this menu is dynamically created given their short and long-term preferences and menu options or content items that the user is likely to be interested in are promoted.

In the following sections we describe a probabilistic model of personalized navigation that drives promotion by computing access probabilities for each user given their current menu. This serves as a review of our recent research and is the benchmark against which we propose to judge the usefulness of a modified approach to personalized navigation that also considers the distance of items from the current menu when selecting promotion candidates (see also [4]).

3.1 Profiling and Personalization

Tracking user accesses across a mobile portal provides the basis for an effective profiling mechanism. For example, individual menu accesses can be stored in a so-called hit-table, which provides a snapshot of a user's navigation activity over time. For example, Figure 3(a) indicates that a user has accessed option B from menu A 10 times and option C 90 times. Of course in reality other activity information including device, temporal and location information is normally stored as part of this evolving profile but a more detailed discussion is outside of the scope of this chapter.

In fact two types of hit table can be used: a global, *static* hit table that is initialized with respect to the default portal structure (Figure 3(b)); and a *user* hit table that records each user's individual history. The initial values for the static hit table are chosen such that the probabilities they produce deliver a default portal structure. For example, in Figure 3(b) we see that options B and C have default hit values of 20 and as such have the same access probability from A. By default B will be presented first and C second as options in the menu corresponding to A. The static table makes it possible to deliver a default menu structure early on that will be over-ridden by the personalized menu once a user's access probabilities build. Moreover, the hit values set in the static table make it possible to control personalization latency - low values mean that personalization takes effect very quickly. For example, by initialising the static hit table with very low values we can expect the user hit table to quickly dominate the access probability calculations and personalization effects will be seen sooner by the

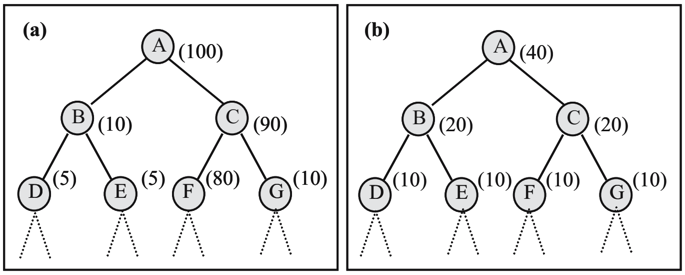


Fig. 3. User (a) and static (b) hit-table representations

user and on the basis of fewer accesses. If the static values are larger then more user accesses will be required before the user hit table values have a tangible effect. As such the static table can be tuned to impose a dampening effect on user accesses when it comes to the probability calculations.

P(B|A)		= (20+10)/(40+100)	.214
P(C|A)		= (20+90)/(40+100)	.786
P(D|A)	= P(B|A) P(D|B)	= (30/140)(10+5)/(20+10)	.107
P(E|A)	= P(B|A) P(E|B)	= (30/140)(10+5)/(20+10)	.107
P(F|A)	= P(C|A) P(F|C)	= (110/140)(10+80/20+90)	.642
P(G|A)	= P(C|A) P(G|C)	= (110/140)(10+10)/(20+90)	.142

Fig. 4. Sample access probabilities; note that the user subscript has been omitted for simplicity

To build a personalized menu m for user u we must identify the k most probable options for m (the k options with the highest $P_u(o|m)$ values) by combining the frequency information in the user and static hit tables. Consider the data in Figure 3 and the construction of menu A. The access probabilities can be determined as shown in Figure 4. In descending order of access probability we have C, F, B, G, D, and E. For $k = 3, C, F$, and B are selected, in order, for menu A.

The complexity of the proposed personalization method depends on the complexity of the process that identifies the k most probable options for the menu, m. As described this can mean examining not just the default options of m, but also all the options contained in menus that are descendents of m; essentially a breadth-first search from m to the content leaves of the menu tree is required. Fortunately, a more efficient algorithm is possible once we recognize that, by definition, $P_u(o|m)$ is always greater than or equal to $P_u(o'|m)$ where o' is an option of a menu, m', which is itself a descendent of m through o. This means that we can find the k most probable nodes for menu m by performing a depth-limited, breadth-first search over the menu tree rooted at m. We only need to expand the search through an option o' if $P_u(o'|m)$ is greater than the k^{th} best probability so far found. Once again, a detailed description of this issue is beyond the scope of the current chapter but the interested reader is referred to [3] for further information.

The approach just described supports two types of menu adaptations: (1) a menu option may be *reordered* within its parent menu by changing its position within its parent menu; or (2) a menu option may be *promoted* into an ancestral menu. Such adaptations are side-effects of the probability calculations. In the above example, option F is promoted to A's menu - options can even be promoted from deeper levels if appropriate. If F is subsequently selected from A, it is added to A's hit table entry for that user, so the next time that A is created, the computation of $P_u(F|A)$ must account for the new data on F. Specifically, assuming a single access to F as an option in A, we get:

$$P_u(F|A) = 1/101 + (110/141)(10 + 80)/(20 + 90) = 0.647.$$

3.2 Deployment Experiences

The above approach to personalized navigation has been fully developed and deployed in the field through the ClixSmart Navigator^{TM} product by Changing-

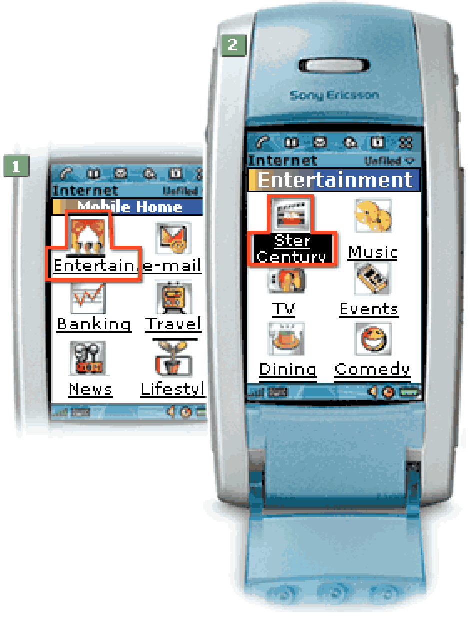


Fig. 5. Personalizing a mobile portal; the screenshots show how menus can be promoted to provide more direct access to local cinema listings. In this example the *Entertain* option on the home page has been promoted to be the top option and the *SterCentury* option has been promoted out of the *cinema* section of the portal and into the *Entertain* menu.

Worlds (www.changingworlds.com). Large-scale deployments by Vodafone and O_2 have attracted millions of mobile subscribers and carefully controlled evaluations prove that this approach to personalization can lead to dramatic improvements in the practical usability of a mobile portal. Figure 5 shows screenshots that illustrate the type of personalization offered by the ChangingWorlds solution based on an initial portal structure shown in Figure 2. In this case, over time the navigation path to the user's local cinema is modified as menu options are rearranged. For example, the "Entertain" menu in the homepage is promoted from the third row to be the first menu in the homepage; in this case the user regularly accesses a range of entertainment related services through this menu. In addition, the "Ster Century" menu option has been promoted out of its original position in the "Movie Times" menu (see Figure 2) to a new position in the "Entertain" menu.

This type of personalization can lead to significant increases in mobile portal usage. For example, airtime, user sessions, and page impressions have all be shown to increase as a direct result of ClixSmart Navigator's personalized navigation solution (see [2, 3]), and in general these significant increases have been observed across many live deployments involving millions of subscribers.

4 Distance-Biased Promotion

Deployments of ClixSmart Navigator demonstrate a compelling link between the practical usability of a mobile portal and the *distance* that a user must travel in order to access its content. Reducing this distance improves usability and drives usage. Therefore it is reasonable to seek out ways of reducing this navigation distance even further. For example, as it stands the above personalization technique is based on access probabilities alone and does not take navigation distance into account in any explicit way; although obviously navigation distance is reduced as a *side-effect* of the personalization process. By considering access probabilities *and* the likely reduction in navigation distance we can further reduce navigation distance as a side-effect of promotion and personalization (see also [4] for related work on the creation of *shortcut* links between Web pages).

4.1 Click-Distance

How can navigation distance be usefully measured? Anderson et al. [4] suggest a simple model of navigation distance that counts the number of links that must be followed to locate a page in a Web site. However, while this simple model is appropriate in the context of more traditional Web sites, it is not well suited to modern mobile portals. With the current generation of mobile phones, there are two basic types of navigation action. The first is the *menu select*: the user clicks to select a specific menu option. The second is a *menu scroll*: the user clicks to scroll up or down through a series of options. Scroll actions are less important in the context of traditional Web portals and traditional Web access devices such as PCs, laptops and PDAs, with their sophisticated point-and-click user input. In contrast, the input capabilities of most Internet-enabled mobile phones are far more limited and even simple scrolling actions correspond to a significant degree of user effort.

Thus, an item of content, i, within a mobile portal can be uniquely positioned by the sequence of selects and scrolls needed to access it, and the navigation effort associated with this item can be simply modelled as click-distance, the corresponding number of these selects and scrolls (see Equation 1).

$$ClickDistance(i) = Selects(i) + Scrolls(i) \quad (1)$$

Recent studies illustrate the extent of the click-distance problem. For example, a recent analysis of 20 European mobile portals reported an average click-distance in excess of 16 [1]; see Figure 6. In other words, a typical European mobile user can expect to have to make 16 or more clicks (scrolls and selects) to navigate from their portal home page to a typical content target. Moreover, on average European portals are organised such that less than 30% of content sites are within 10-12 clicks of the portal home page; 10-12 clicks corresponds to a navigation time of about 30 seconds, which is expected by mobile Internet users [14]. To put this another way, more than 70% of mobile portal content is essentially invisible to users because of its positioning within its parent portal. Finally it is worth highlighting that although the above click-distance model constitutes a fairly simple model of navigation effort it is nonetheless an effective one in practical terms. For example, a recent analysis of the activity of 3500 users of a major European portal, over the course of a 30-day period in 2002, found a correlation of -0.65 between the click-distance of content-sites (that exist as leaf nodes in the portal hierarchy) and their access frequencies. In other words, all other things being equal, sites with a low click-distance are accessed more frequently than sites with a high click-distance. In one sense of course this result is fairly intuitive, nevertheless the strength of the correlation is significant.

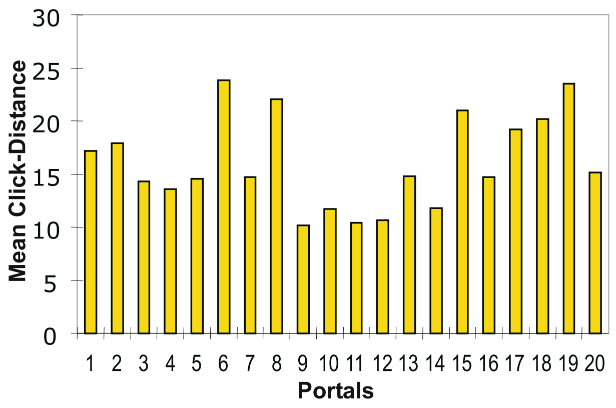


Fig. 6. Mean click-distances for 20 European mobile portals

4.2 Expected Click-Distance

At this stage it is possible to combine the click-distance of a portal item i and its access probability to calculated the *expected click-distance* (ECD) of i. Then, in order to promote those items that have the largest expected click-distances, because such promotions are likely to result in the greatest expected click-distance savings (see also [4]). As a result, during personalization, instead of simply computing the access probabilities for descendants of the current menu, m, we calculate the expected click-distances of these descendants according to Equation 2 (the expected click-distance of item i from menu m, for some user u).

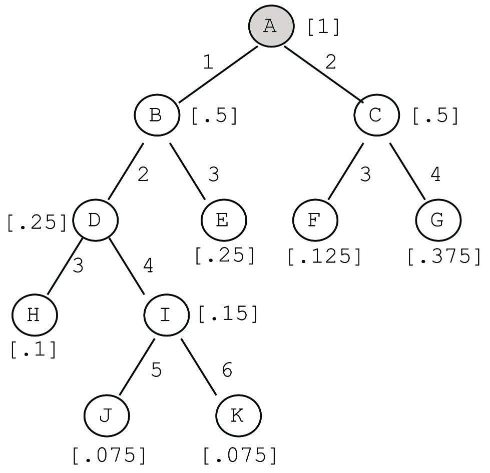
$$ECD_u(i,m) = NormalisedClickDistance(i) * P_u(i|m) \tag{2}$$

Figure 7 illustrates a simple example of this concept. A small portal, rooted at home page A, is presented and the expected click-distance values for all of the descendants of A are calculated. For example, menu G has the largest expected click-distance (1.5) based on an access probability of 0.375 and a click-distance from A of 4 (2 scrolls and 2 selects). Note that here we assume that the first option in a menu is available for selection by default. Therefore to navigate from menu A to option G requires 1 scroll to move from B to C, 1 selection of C to enter menu C, one scroll from C to G and finally a selection of G.

It turns out that G has the highest expected click-distance and so would be promoted into menu A ahead of even B or C, A's default descendants. Similarly E would be promoted ahead of B as it has an expected click-distance of 0.75 compared to B's expected click-distance of 0.5. As a result, if we assume that menu A has been configured to allow for 5 options then, in order of their expected click-distances, options G, C, E, I, B would be selected. In contrast, on the basis of access probabilities alone, options B, C, G, E, D would have been selected in order. Of course, it is also possible to impose tighter limits on the number of options that can be displayed in a menu but in certain circumstances this may limit promotions in favour of default options in order to ensure that all parts of the portal remain accessible. For example, if A had been limited to 3 options then G, C, B would have been selected ahead of E and I; that is, even though E and I have higher ECDs than B, B must be selected to preserve access to its part of the portal.

It is important to to realise here, in relation to our expected click-distance metric, that the form and objective of the metric is not simply to provide an arbitrary way of combining click-distance and access probabilities. After all, the scales of these two factors are very different. The point is that this metric uses the access probabilities to calculated an expected click-distance and as such the resulting value continues to be meaningful as a measure of distance.

Note that Equation 2 assumes that there is only one path from m to i. Of course in general, and regardless of whether we use expected click-distance or access probabilities on their own to drive promotion, there can be multiple paths from m to i each with their own access probabilities and click-distance from m. As such, in practice, the expected click-distance must be summed over these alternate paths in the obvious way.



Item, i	CD(i)	$P_u(i \mid A)$	$CD(i)*P_u(i \mid A)$
G	4	0.375	1.5
C	2	0.5	1
E	3	0.25	0.75
I	4	0.15	0.6
B	1	0.5	0.5
D	2	0.25	0.5
K	6	0.075	0.45
F	3	0.125	0.375
J	5	0.075	0.375
H	3	0.1	0.3

Fig. 7. A sample portal showing access probabilities (in square brackets) and item click-distances from the portal root. The table shows the expected click-distances for each of the descendants of A in descending order.

5 Experimental Evaluation

In previous work we have reported widely on the results of extensive live-user trials of our personalized navigation techniques [2, 3]. These trials prove a strong link between click-distance reduction and increased portal usability. In this chapter we are interested in evaluating the likely impact of biasing our personalization technique to include a distance factor as well as its core access probability factor. Ultimately we are interested in understanding if the above distance-biased promotion technique is likely to result in an increased click-distance reduction (compared to the pure probabilistic approach) in real mobile portals. If increased click-distance reductions are proven then this bodes well for the new distance-biased approach because these greater reductions are likely to result in further (or at least more rapid) usability improvements.

5.1 Setup

For the purpose of this experiment we used data from a leading European mobile portal. This included the portal structure containing over 450 portal nodes and 14-days worth of user access logs covering the activity of 3,500 individual users. We also made use of two versions of ClixSmart Navigator: the standard version that relies on a pure probabilistic approach to promotion and personalization; and an enhanced version that incorporates the above distance-biased approach. In each case we used the entire set of logs of user activity to drive the two promotion schemes.

Unfortunately in this evaluation it was not possible to present the resulting personalized portals back to live-users so we were not in a position to evaluate whether the promoted pages were actually being accessed more or less. Instead, as discussed below, we replayed the access logs to mimic user activity with a view to measuring the daily change in portal click-distance as a result of our two different promotion schemes.

5.2 Comparative Click-Distance Profiles

The key question to answer is whether there is any significant difference in the click-distance reduction obtained using the distance-biased method when compared to the reduction obtained using the pure probabilistic method. To test this we used the supplied user logs to *replay* the user activity over two versions of the portal: one that is personalized by the pure probabilistic strategy and one that is personalized according to the distance-biased strategy. This allowed us to evolve two different portals over the 14 day test period: a pure probabilistic portal and a distance-biased portal. At the end of each simulation "day" we calculated the mean click-distance of the two portals, averaged over all user sessions that occurred during that day; in other words the mean click-distance values calculated are based on the new click-distances of those pages that the users actually did access according to the access logs.

The results are presented in Figure 8 as a graph of mean click-distance against simulation day for each of the two portals corresponding to the two different personalisation strategies. The results clearly show the click-distance reduction capability of each strategy. At the end of the first day both portals have an average click-distance of between 8.4 (distance-biased) and 8.6 (pure probability) and by the end of the 14th day this has dropped to about 5.3. In other words, to begin with it takes users more than 8 clicks to get to a typical content site from their portal home page. Remember, these 8 clicks are made up of a combination of scrolls and selects and in this portal the ratio is over 3 to 1, so on average 2 of these 8 clicks will be menu selections and the remaining 6 will be scrolls.

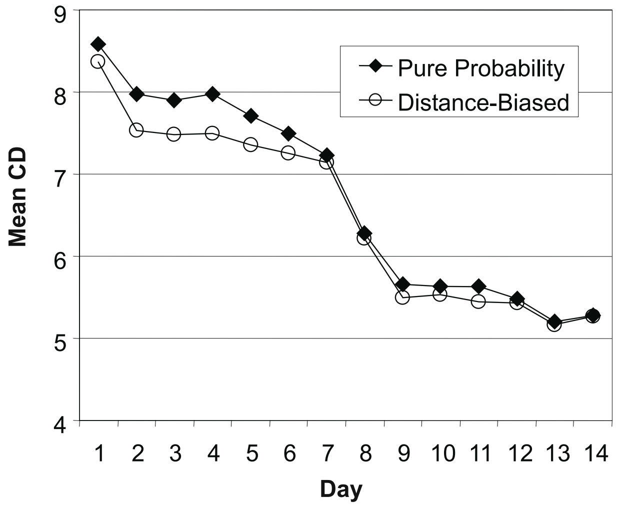


Fig. 8. Click-distance results

After only two weeks users are able to access content sites in about 5 clicks (approximately 1 menu select and 4 scrolls), an overall click-distance reduction of 40%.

Perhaps the most important thing to note is the lack of any significant difference between the click-distance profiles of the two portals, at least beyond day 6. In other words, although the distance-biased technique is capable of delivering improved click-distance reductions, this benefit is relatively short-lived and appears to disappear after day 6. Moreover, the extent of this improvement for the first 6 days is marginal at roughly 0.5 clicks. The average click-distance for the pure probabilistic portal is 8, over the first 6 days, compared to 7.5 for the distance-biased portal; that is, the pure probabilistic portal suffers from a 7% increase in click-distance when compared to the distance-biased portal.

5.3 Further Analysis

On the face of it then there appears to be relatively little advantage in biasing promotion using distance factors. Why should this be the case? One possibility is hinted at by the actual average click-distances reported in the above results. The evaluation portal contains more than 450 individual nodes and has an average click-distance in excess of 15 across all of its content sites; that is, 15 is the average click-distance from the portal home page to each of the content sites that exist at the leaf nodes of the portal. Nevertheless, the average click-distance reported above, which is based on actual user sessions rather than a static click-distance analysis of the entire portal, is no more than 8. In other words, although many sites within the portal exist at very large click-distances from the portal home page, the majority of users actually never wander very far from the home page in a typical session. In fact when we further analysed the user logs we found that more than 80% of content accesses were for content sites that were within a click-distance of 10 from the portal home page, sites that were linked to from level 1 or level 2 pages.

This observation has two important implications. First, it means that many of the content sites accessed by users have similar click-distance to begin with and this limits the impact of the distance factor during personalization. For example, the 80% of accesses referred to above are for content sites with an average click-distance of 5.2 and a standard deviation of 3.4. Secondly, and perhaps more importantly, because these sites were positioned on level 1 or level 2 pages the distance that they could be promoted was also limited from the start. For example, if a site is linked to from a level 1 menu then it can only be promoted to the portal home page (level 0) and similarly level 2 pages can only be promoted to level 1 or level 2 menus.

The essential point is that distance-biased promotion is only likely to have a large impact on click-distance when very distant sites are promoted because of the distance bias. The behaviour of mobile portal users is so limited that such distant sites, although they exist, are rarely accessed and thus rarely promoted. Therefore, although distance-biased promotion has the potential to improve our pure probabilistic personalization technique *in theory*, we find that in practice it is not well adapted to the needs of behaviour of real mobile portal users. Contrast this with the work of [4] where a form of distance-biased personalization is used to good effect in the generation of shortcut links between Web pages. Web usage is less limited than mobile portal usage and Web users are more likely to follow long chains of links to their destination content. Therefore, the benefits of distance-biased personalization are more pronounced.

6 Conclusions

In general, limited usability and poor value-for-money are major contributing factors to the low levels of interest in the mobile Internet currently shown by the general public. These problems are closely aligned with the difficulty that users have in locating content on mobile portals. This navigation problem is especially acute on the mobile Internet

In this chapter we have described how personalization techniques can be used as a potential solution [2, 3] by actively reducing portal click-distance. In particular we have focused on the ClixSmart Navigator product-suite developed by ChangingWorlds, which uniquely offers mobile operators the ability to develop and deploy fully personalized mobile portals. From a personalization perspective ClixSmart Navigator represents a significant commercial success story for personalization research. It is widely deployed by Europe's leading mobile operators and actively personalizes the mobile portals of many millions of mobile subscribers. Indeed ClixSmart Navigator has been directly responsible for helping to increase portal usage because of its ability to improve portal usability.

In this chapter we have also focused on ways to further improve our personalization approach by directly considering the navigation distance to portal items during personalization - to promote more distant items before nearby items on the assumption that distant items are likely to lead to even greater navigation savings. However, after evaluating this approach on 3,500 users of a large European portal, we have found that any improvements are marginal and short-term. However, this is not so much a failing of the distance-biasing concept, but rather a side-effect of the usage patterns of mobile users. The simple fact of the matter is that, compared to their Web cousins, mobile users are impatient and rarely tolerate long navigation times. The majority of accesses are to content sites that are within a limited distance of the portal home page and this fundamentally limits the impact of any distance-bias that is introduced into the personalization mechanism.

References

1. Smyth, B.: The Plight of the Mobile Navigator. MobileMetrix (2002)
2. Smyth, B., Cotter, C.: Personalized Adaptive Navigation for Mobile Portals. In: Proceedings of the 15th European Conference on Artificial Intelligence - Prestigious Applications of Artificial Intelligence, IOS Press (2002)
3. Smyth, B., Cotter, C.: The Plight of the Navigator: Solving the Navigation Problem for Wireless Portals. In: Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'02), Springer-Verlag (2002) 328–337
4. Anderson, C., Domingos, P., Weld, D.: Adaptive Web Navigation for Wireless Devices. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence. (2001) 879–884
5. Jacko, J., Salvendy, G.: Hierarchical Menu Design: Breadth, Depth and Task Complexity. Perceptual and Motor Skills **82** (1996) 1187–1201
6. Kiger, J.: The Depth/Breadth Tradeoff in the Design of Menu-Driven Interfaces. International Journal of Man/Machine Studies **20** (1984) 201–213
7. Larson, K., Czerwinski, M.: Web Page Design: Implications of Memory, Structure and Scent for Information Retrieval. In: Proceedings of the CHI'98 Human Factors in Computer Systems, ACM Press (1998) 25–32
8. Lee, E., MacGregor, J.: Minimizing User Search Time in Menu Retrieval Systems. Human Factors **27** (1985) 157–162
9. Miller, D.: The Depth/Breadth Tradeoff in Hierarchical Computer Menus. In: Proceedings of the 25th Annual Meeting of the Human Factors and Ergonomics Society. (1981) 296–300
10. Snowberry, K., Parkinson, S., Sisson, N.: Computer Display Menus. Ergonomics **26** (1983) 699–712
11. Wallace, D., Anderson, N., Shneiderman, B.: Time Stress Effects on Two Menu Selection Systems. In: Proceedings of the 31st Annual Meeting of the Human Factors and Ergonomics Society. (1987) 727–731
12. Webb, J., Kramer, A.: Maps or Analogies? A Comparison of Instructional Aids for Menu Navigation. Human Factors **32** (1990) 251–266
13. Zaphirs, P.: Depth vs. Breadth in the Arrangement of Web Links. In: Proceedings of 44th Annual Meeting of the Human Factors and Ergonomics Society. (2000) 139–144
14. Ramsey, M., Nielsen, J. In: The WAP Usability Report. Neilsen Norman Group (2000)

15. Billsus, D., Pazzani, M., Chen, J.: A learning agent for wireless news access. In: Proceedings of Conference on Intelligent User Interfaces. (2000) 33–36
16. Fu, X., Budzik, J., Hammond, K.: Mining Navigation History for Recommendation. In: Proceedings of Conference on Intelligent User Interfaces. (2000) 106–112
17. Perkowitz, M. In: Adaptive Web Sites: Cluster Mining and Conceptual Clustering for Index Page Synthesis. PhD Thesis, Department of Computer Science and Engineering. University of Washington (2001)
18. Perkowitz, M., Etzioni, O.: Towards adaptive web sites: Conceptual framework and case study. Journal of Artificial Intelligence **18(1-2)** (2000) 245–275
19. Reiken, D.: Special issue on personalization. Communications of the ACM **43(8)** (2000)
20. Smyth, B., Cotter, C.: Wapping the Web: A Case-Study in Content Personalization for WAP-enabled Devices. In: Proceedings of the 1st International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'00). (2000) 98–108

IKUM: An Integrated Web Personalization Platform Based on Content Structures and User Behavior

Magdalini Eirinaki[1], Joannis Vlachakis[2], and Sarabjot Singh Anand[3]

[1] Athens University of Economics and Business,
Department of Informatics,
Patision 76, Athens, 10434, Greece
eirinaki@aueb.gr
[2] Fraunhofer IAO,
Nobelstr. 12, 70569 Stuttgart, Germany
joannis.vlachakis@iao.fhg.de
[3] Department of Computer Science, University of Warwick,
Coventry CV4 7AL, UK
s.s.anand@warwick.ac.uk

Abstract. Web personalization is the process of customizing a web site to the needs of each specific user or set of users, taking advantage of the knowledge acquired through the analysis of the user's navigational behavior. The objective of the I-KnowUMine project (IKUM) is to develop an integrated platform (referred to in the paper as the "IKUM system") that uses state of the art technology and research results from different application domains in order to provide the basis for the development of online services in a wide range of application areas, presenting personalized content, services and applications to users in a structure more suited to their needs. The benefits provided by the IKUM system result mainly from the combination and integration of technology advances in areas such as Web Mining, Content Management, Personalization and Portals. As a result of this novel combination of these technologies, users of the IKUM system will benefit from the optimal logical structure of information/content provided by the system, allowing them to efficiently execute their processes and to reach their information targets.

1 Introduction

The continuous growth of the World Wide Web in terms of content and usage has heightened the need for new methods in design and development of online services to the end-user. The need for predicting the users' needs in order to improve the usability and user retention of a web site is more than evident and can be addressed by personalizing it. *Web personalization* is defined as any action that adapts the information or services provided by a web site to the needs of a user or a set of users, taking advantage of the knowledge gained from the users' navigational behavior and individual interests, in combination with the content and the structure of the web site [12]. As mentioned in [16], *"The objective of a web personalization system is to provide users with the information they want or need, without expecting from them to ask for it explicitly"*.

The objective of the I-KnowUMine project is the development of a novel and innovative content delivery platform based on Content, Knowledge and Behavioral data to present personalized content to users in a structure more suited to their needs. Users of the IKUM system shall benefit from the optimal logical structure of information/content provided by the system, allowing them to efficiently execute their processes/tasks and to reach their information targets. The advantage of this approach arises mainly from the combination and integration of technology advances in areas such as Web Mining, Content Management, Personalization and Portals. The system combines knowledge of typical user behaviors with rules and conditions of the underlying content structure and semantics in order to provide the optimal flow of information in content provision applications and services assuring contextual content integrity.

The rest of the paper is organized as follows: In Section 2 we define Content Contextualisation Servers (CCS) in the context of the application areas adjacent to it, technologies from which influence the ability to deliver a system within the CCS area. In Section 3 we present related research efforts to IKUM, focusing on those that present an integrated web personalization system, using usage and content data. The IKUM architecture is described in more detail in Section 4. We present the modules that comprise the system and detail its most innovative features. In Section 5 we present some preliminary experimental evaluation, and finally conclude in Section 6.

2 Content Contextualization Servers

As the IKUM objectives span across a number of application areas, there is a need to define the area within which systems like the IKUM system exist. We refer to this area as the **Content Contextualization Server** area. The Content Contextualization Server is defined as *"Software that delivers content deemed relevant to a users' context, taking into account the users' behavior and semantic content preferences as defined by the users' previous and current access of content."*

As the users' context is dynamic, so must be the strategies for structuring the content delivered to the user. Hence, Content Contextualization Servers typically require strong predictive analytics, multiple recommendation strategies, content and knowledge management capabilities and flexible content delivery functions. Thus, content contextualization servers depend on:

- Traditional *content management servers* to provide components for content authoring and publishing.
- *Web Mining components* for data collection, pre-processing, analysis and generation of knowledge for use in personalization.
- Multiple *personalization components* for using the knowledge generated and recommendation of content based on current behavior.
- *Portals* to provide the deployment platform for content contextualization.

Given the definition of the Content Contextualization Server area, the following four areas are deemed to be adjacent to it: Personalization, Web Mining, Content Management and Portals. These represent market areas that provide part solutions to the objectives of the I-KnowUMine platform, however, they fall short on delivering the complete content contextualization server vision. The following sub-sections very briefly define these areas.

2.1 Personalization

Personalization tools aim to enable enterprises to adapt their interactions with their customers based on their individual needs. This includes targeting advertising, promoting products, personalizing content presentation on web channels, recommending documents, giving appropriate advice, target e-mailing and custom pricing. One of the most prominent technical and organizational challenges in this context is the provision of dynamic, personalized, collaborative interaction between user (employee, customer etc.) and supplier across all possible interaction channels.

Data mining is the technology used to discover non-obvious, potentially useful and previously unknown information from data sources. The potential of web mining is in the application of existing and new data mining algorithms to web data, which include server logs, as well as external data on customer, sales, and products etc. The business benefits that web mining affords to e-Business providers include personalization, collaborative filtering, enhanced customer support, product and service strategy definition, product marketing, online usability improvement and fraud detection.

2.2 Content Management

Content management encompasses a set of processes and technologies, enabling the creation and packaging of content (documents, web services, complex media, applets, components, etc.) as part of a dynamic and integrated web-centric environment. It also comprises of any action performed on online (web) content in order to extract usefule information, such as keywords, metadata, semantics, etc. A content management system may also contain a content delivery system, which uses and compiles that information to update the web site. In general, the features of a content management system vary, but most include web-based publishing, format management, revision control, and indexing, search, and retrieval.

2.3 Portals

A portal can be defined as a personalizable, browser-based user interface to all appropriate corporate resources from any Internet-capable device. Being holistic business solutions, portals are platforms offering users efficient performance of various business processes across enterprise boundaries. To support efficient process performance, a portal must exhibit comprehensive knowledge concerning customer-specific business processes and provide all contextually relevant information and services to users (for administrators as well as end users) in a customizable manner.

3 Related Work

The use of web usage mining for supporting web personalization has recently attracted a lot of interest [1, 16]. In most of the cases, data mining techniques are used in order to extract useful patterns and rules concerning the users' navigational behavior. Site modifications are then made either by humans, or by a recommendation engine which helps the user navigate through a site. Some of the more integrated systems provide greater functionality, introducing the notion of adaptive web sites and providing means for dynamically changing a site's structure. Few research projects integrate semantic web site structure knowledge with usage knowledge within the personalization process. An extensive overview of the most representative systems can be found in [11]. In this paper we focus on the integration of semantic, structure and usage knowledge to dynamically modify a web site.

Coenen et al. [9] proposed a framework for self-adaptive web sites, taking into account the site structure but not the site usage. They underline the distinction between strategic changes, referring to the adaptations that have important influence on the original site structure, and tactical changes, referring to the adaptations that leave the site structure unaffected. The proposed approach is based on the fact that the methods used in web usage mining produce recommendations including links that don't exist in the original site structure, resulting in the violation of the beliefs of the site designer and the possibility of the visitor getting lost following conceptual but not active links. Therefore, they suggest that any strategic adaptations based on the discovery of frequent item sets, sequences and clusters, should be made offline and the site structure should be revised. On the other hand, as far as the tactical adaptations are concerned, an algorithm for making online recommendations leaving the site structure unaffected is proposed.

Perkowitz et al. [20] were the first to refer to the notion of adaptive web sites, defining them to be sites that semi-automatically improve their organization and presentation by learning from visitor access patterns [19]. The system, they proposed, semi-automatically modifies a web site allowing only non-destructive transformations. Therefore, nothing is deleted or altered, instead new index pages containing collections of links to related but currently unlinked pages are added to the web site. They proposed PageGather, an algorithm that uses a clustering methodology to discover web pages visited together and to place them in the same group. More recently [21], they proposed IndexFinder, which fuses statistical and logical information to synthesize index pages. In this latter work, they formalize the problem of index page synthesis as a conceptual clustering problem and try to discover coherent and cohesive link sets which can be presented to a human Webmaster as candidate index pages. In the case of IndexFinder information is derived from the site's structure and the page content. Therefore, IndexFinder combines the statistical patterns gleaned form the log file with logical descriptions of the contents of each web page in order to create index pages.

The WebPersonalizer system proposed by Mobasher et al. [14] provides a framework for mining web log files to discover knowledge for the provision of recommendations to current users based on their browsing similarities with previous users. It relies solely on anonymous usage data provided by logs and the hypertext structure of a site. After data gathering and pre-processing (converting the usage, content and structure information contained in the various data sources into various data abstractions) [8], data mining techniques such as association rules, sequential pattern discovery, clustering and classification are applied in order to discover interesting usage patterns. The results are aggregated usage profiles. The recommendation engine matches each user's activity against these profiles and provides him/her with a list of recommended hypertext links. This framework was extended in a more recent work [15] to incorporate content profiles into the recommendation process as a way to enhance the effectiveness of personalization actions.

Berendt et al. introduced "service based" concept hierarchies in [3], for analysing the search behaviour of visitors, i.e. "how they navigate rather than what they retrieve". This idea is further analysed in [2], where concept hierarchies as the basic method of grouping web pages together. STRATDYN, is the add-on module that extends WUM's ([22]) capabilities by identifying the differences between navigation patterns, and exploiting the site's semantics in the visualization of the results. The accessed pages or paths are abstracted, since web pages are treated as instances of a higher-level concept, based on page content, or by the kind of service requested.

Dai et al. [10] propose a web personalization framework that incorporates usage profiles and domain ontologies. The usage profiles can be transformed to "domain-level" aggregate profiles by representing each pageview with a set of related ontology objects. Recommendations can then be generated by matching the current user's profile with them. The paper proposes a general framework for a system which is divided into two modules, the offline, which is comprised of data preparation and specific web mining tasks, and the online component, which is a real-time recommendation engine.

The idea of enhancing usage mining by registering the user behavior in terms of an ontology is described by Oberle et.al. [17]. This framework is based on a (semantic) web site built on an underlying ontology. The web logs are semantically enriched with ontology concepts. Data mining may then be performed on these semantic web logs to extract knowledge about groups of users, users' preferences, and rules. Since this process is based on a semantic web knowledge portal, the web content is semantically annotated exploiting the portal's inherent RDF annotations, and no further automation is provided. The proposed framework focuses mainly on web mining and does not perform any further processing in order to support web personalization.

Finally, Eirinaki et. al. [12] present SEWeP, a web personalization system that incorporates usage and content knowledge in the web mining and personalization process. More specifically, the web usage logs are augmented with semantics derived from the content of the web site's pages, which are then used to broaden the final set of recommendations to the user with semantically similar content. The web site's pages are characterized using terms that belong to a domain-specific taxonomy. This abstraction favors the system's functionality in several ways; the document clustering, web log mining and recommendation processes are no more based on exact keyword matching, but on semantic similarity (i.e. similarity between terms of a concept hierarchy) instead.

The system presented in the paper is most similar in nature to the work carried out in the SEWeP system. In fact, SEWeP system is partly based on the IKUM framework described in this paper, in terms of the content management and knowledge extraction processes. However, compared to any other approach, the IKUM system architecture provides a more generic framework, providing novel methods for generating the semantics used to characterize the web pages. It also differentiates from all previous approaches in terms of the use of multiple recommendation systems, the tighter integration with a content management system and the conformance to standards such as PMML and SOAP.

4 System Architecture

Based on the observations in Section 2 the system architecture developed, benefits from the combination of features derived from the following areas: Personalization, Web Mining, Content Management and Portals. Accordingly, the system modules are classified into four main layers: the Content Management Layer, Web Mining Layer, Knowledge Management Layer, and Interaction Layer.

The *Content Management Layer* incorporates the Content Management Module, the Taxonomy Management Module and the Content Classification Module. The main functionality implemented in this layer are the support for consistent authoring and storage of the content of the web site, its enrichment with semantic information, produced automatically or corrected/provided by a domain expert, support for creating/importing Taxonomies and support for administrative functions such as workflow and user management.

The *Web Mining Layer* consists of modules for enhancing web log files with semantic information extracted from the web site's content, leading to the creation of the C-logs (concept-logs), which are used as input to the Web Mining Module, loading the data into a Webhouse and the mining modules. The knowledge generated by the mining modules is represented in PMML [11] and stored within a knowledge base.

The *Knowledge Management Layer* is responsible for managing the knowledge generated by the Web Mining layer and includes its deployment through various recommendation engines (Recommendation Module).

Apart from these three general layers there is also an *Interaction Layer*, which includes the Publishing Module and the web server, which will present the corresponding personalized page to every user, by combining possibly "fixed" parts of the web page with parts where the personalized information should be presented.

The layers and modules described are depicted in Figure 1. The system architecture may be divided into an on-line and an off-line part. Functions performed off-line include those supported by the Content Management and the Web Mining Layer, whereas the module which creates the recommendations and the mechanism that publishes the personalized page for the visitors to the web site belong to the on-line part of the IKUM system.

The layers and modules are described in more detail in the following sections. Here, we illustrate the fundamental functionality of the system as well as the interaction of its different modules, by means of a scenario that describes a typical user visit to a web site based on the IKUM system: The user requests a page from the web site that uses the IKUM system. This request is received by the web server and passed on to the Content Management Layer. At the same time the web server stores information about the request within the web server log file. The Content Management Module sends to the Publishing Mechanism the "fixed" part of the requested content, i.e. those sections of the page that do not contain personalized recommendations and should appear the same for all visitors. At the same time, through the combination of the visitors' current clickstream and the knowledge stored in the knowledge base, either using a user profile for the user or using group behavioral knowledge, generated by the web mining modules, the Recommendations Module generates recommendations and sends them to the Publishing Mechanism. The Publishing Module combines the two inputs ("fixed" content + recommendations) and renders the personalized web page via the web server to the end-user. This process is performed online.

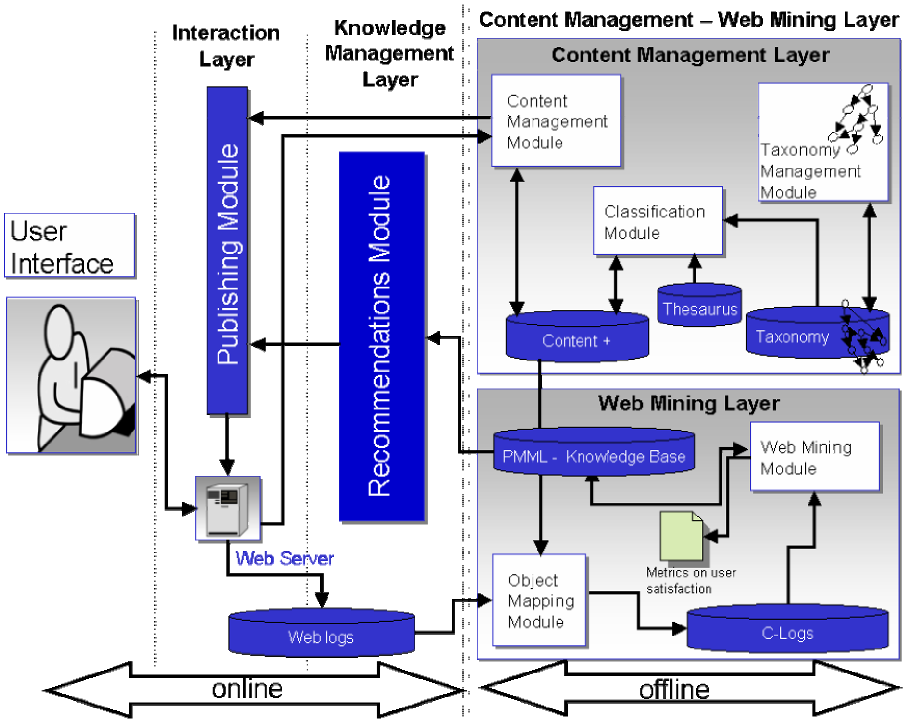


Fig. 1. The IKUM system architecture

The knowledge base that provides input to the Knowledge Management Layer in order to produce recommendations, is created according to the following offline procedure: The content of the web site is stored in the Content Repository (Content+) in the form of objects containing the content and related/associated metadata. The metadata may be added manually by a domain expert or by the use of (semi-) automated classification techniques provided by the Classification Module. Through the use of a domain-specific taxonomy and a thesaurus, the Classification module extracts keywords using text mining techniques and identifies taxonomy categories that characterize every content object that is stored in the Content+ repository. The domain-specific taxonomy is created and administered through the Taxonomy Management Module. The content classification is performed once for every content object and should be repeated only if new content is added, or old content is modified.

Another factor that should be considered is the need for a mapping between the content URIs stored in the Content+ database and the corresponding portal URIs, since usually a portal web page (portal URI) consists of several different content objects (content URIs). This mapping is needed in order to perform web usage mining and enhance the usage data with content knowledge stored as metadata properties of the several content objects. It is performed by the Object Mapping Module, that takes input from the Content+ repository and the web logs and creates an extended version of the web logs, called *C-Logs*. A C-Log record is the same as the web log's record, except for an extra field including the semantic characterization of the corresponding URI. This characterization consists of the taxonomy categories extracted in the previous phase, by the Classification module. Therefore, in this phase every record of the web log is updated with the corresponding categories (taxonomy terms) to create the C-Logs. The C-Logs are then used by the Web Mining Module on the Web Mining Layer and are processed in order to extract patterns, which are stored in the Knowledge Base, and metrics on user satisfaction.

4.1 Content Classification

When a personalization system relies solely on usage-based patters, information conceptually related to what is finally recommended will not be considered. Moreover, in dynamic web environments (such as portals or database-based web pages), specific URIs may not be valid for a long period of time but accesses to them still provide useful information at the level of the content category of interest to a user. Using semantic annotation of the content, the final set of links that are proposed to the end user is expanded containing URIs and general content categories that would not have been recommended to the user otherwise. This is our motivation in designing and including in the general architecture a special module for the classification of the web site's content into categories belonging to a domain-specific taxonomy. After this procedure, the content is enhanced with semantic information, which, as already mentioned, is then stored along with behavioral data (user clickstreams) in the C-Logs. The process of classifying the web site content is as follows: The web content is parsed to remove all presentation/structure-related tags (Content Parsing), and subsequently information retrieval methods are applied in order to convert it into a more structured representation, by assigning a set of keywords to every web document (Keyword Extraction). These keywords are then used to classify every document into the various content categories (Keyword Category Mapping) as defined in a domain specific taxonomy. Each of these stages in the process is described in more detail in the following subsections.

4.1.1 Content Parsing

In order to extract keywords characterizing each web page, the web content should be isolated from structure data (HTML and XML tags, meta-tags etc.). Therefore, in the first stage each object representing content (content URI) to be published in a web page (portal URI) of the web site is parsed in order to separate its content for further processing. In order to accomplish this, a parser processes all the content URIs that constitutes the web site and contained in the Content Repository.

4.1.2 Keyword Extraction

In the second stage, a text-mining algorithm is employed in order to extract keywords that characterize each web page. This procedure can be divided in two sub-stages: (a) document indexing, where content-bearing terms are extracted from the web page text, and (b) term weighting, where the indexed terms are given weights in order to choose the most important terms for characterizing the text.

4.1.2.1 Document Indexing

Many of the words in a document may not specifically describe its content. We refer to these words as non-significant words and remove them from consideration when generating the document indexing. The decision of which terms are significant and which ones are not may be defined using term frequency, by using a stop-words list or by some metric that is a combination of both these or indeed other measures of term importance. In the first case, the assumption that words with very high or low frequency are functional words is made, and those words are removed. In the second case a list of words that are commonly used, referred to as stop words, is provided to remove these words from consideration during document indexing. The stop-words list includes very common words, such as pronouns, articles, adjectives, adverbs, and prepositions. In our architecture, we use a stop list in order to remove the non-content bearing words.

4.1.2.2 Term Weighting

Term weighting, extensively used in the vector space model (also referred to as bag-of-words) for document clustering, may be done using several methods, such as raw term frequency, or algorithms belonging to the Tf*Idf family [23]. Raw term frequency is based on the term statistics within a document and is the simpler way of assigning weights to terms. Tf*Idf is a method used for collections of documents, i.e. documents that have similar content. In the case of a web site however, this assumption is not always true since a web site may contain documents that refer to different thematic categories (especially in the case of web portals).

4.1.2.3 Keyword Selection

The most straightforward approach for extracting keywords from a document is to perform text mining in the document itself, following standard IR techniques. However, this approach proves insufficient for the web content, since it relies solely on the information included in the document ignoring semantics arising from the connectivity features of the web [4, 6]. It is difficult to extract keywords from web documents that contain images, programs etc. Additionally, many web pages do not include words that are the most descriptive ones for their content. Therefore, in many approaches information contained in the links that point to the document and the text near them is used for characterizing a web document. Chakrabarti et al. define this as the “anchor-window” [7]. The assumption made is that the text around the link to a page is descriptive of its contents. This approach overcomes the problems of the content-based approach, since it takes into consideration the way others characterize a specific web page. A related effort capitalizing on link semantics appears in [13].

In the IKUM system, the most representative words that describe a web document are selected using a combination of the aforementioned methods and a method based on page links. More specifically, the keywords are extracted using:

1. raw term frequency of the web page
2. raw term frequency of a selected fraction of the reference to web pages that are pointed to by this page (outlinks)
3. raw term frequency of a selected fraction of the most important web pages that point to this page (inlinks)

The first method is straightforward, since the most frequent words extracted from the text are included in the proposed keyword set. In the second method, all the links that are contained in the content object under consideration are visited and parsed in order to extract frequent terms. As for the third method, a web crawler or explicit site structure data is used in order to find web pages that have links to the content object under consideration. When the link to this page is found, it is parsed along with 100 characters before and after the link, in order to extract keywords. The second and third method enhance the keyword extraction process, since they are based on the assumption that the characterization of the content that other people (content authors) give to the content object may provide more valuable information than the content in the object itself.

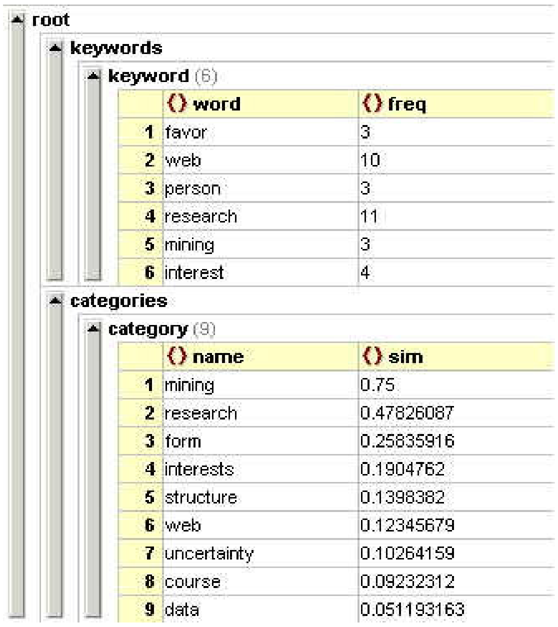
4.1.3 Keyword-Category Mapping

After the aforementioned process, the most highly ranked words that are extracted using the three methods are selected as representatives of its content. However, since all web documents should be uniformly characterized by a limited, domain specific vocabulary, the keywords that were extracted in the previous stage should be mapped to the concepts defined in the taxonomy. This mapping is performed by using a thesaurus and a domain-specific taxonomy that has been defined in a machine-readable format (RDF) using the Taxonomy Management Module. If the keyword belongs to the taxonomy, then it is included as it is. Otherwise, the system finds the "closest" category word to the keyword through the mechanisms provided by the thesaurus. In our system implementation, we used WordNet[1] [24] as a thesaurus and Wu & Palmer similarity measure [WP94] for calculating the distance between different terms. An example of the output of the keyword-category mapping process can be found in Figure 2.

For more details on mapping between keywords and categories, and the calculation of the respective similarities, the user may refer to [13]. It should be noted that in the case that the extracted keywords are in a language other than English, these words are translated to English before further processing using a dictionary. At the end of this stage, each content URI is characterized by a set of categories that are part of this taxonomy.

At the end of this process, each record in the web logs is updated to include the related taxonomy categories, leading to the creation of C-Logs. As already mentioned, C-Logs are in turn used by the Web Mining Module and are processed in order to extract navigational patterns, which are stored in the Knowledge Base, and metrics on user satisfaction.

[1] WordNet is a lexical database containing English nouns, verbs, adjectives and adverbs organized into synonym sets, each representing one underlying lexical concept. It provides mechanisms for finding synonyms, hypernyms, hyponyms, etc. for a given word.



root

keywords

keyword (6)

	word	freq
1	favor	3
2	web	10
3	person	3
4	research	11
5	mining	3
6	interest	4

categories

category (9)

	name	sim
1	mining	0.75
2	research	0.47826087
3	form	0.25835916
4	interests	0.1904762
5	structure	0.1398382
6	web	0.12345679
7	uncertainty	0.10264159
8	course	0.09232312
9	data	0.051193163

Fig. 2. XML file including extracted keywords and categories for the web page http://www.db-net.aueb.gr/magda/research.htm

4.2 Sequence Tree Based Recommendation

Sequence patterns generated by Sequence pattern discovery algorithms such as Capri [5] provide insights into navigational behavior of visitors to a web site. Matching these patterns to current behavior provide the basis for recommendation generation with the confidence measure associated with the matching sequence rule providing the measure of confidence in the recommendations. Capri generates PMML as well as a more compact, tree-based representation of the discovered sequences. As both these knowledge representations have associated XML representations, one representation can be easily transformed into the other using transformation languages such as XSLT. The sequence based recommendation engine used by the IKUM system takes as input sequence patterns represented in the tree format. However, it is worth stressing on the fact that as a PMML document representing sequences generated by a different sequence discovery algorithm can be easily transformed into this representation, thus this input format by no means restricts the use of Capri for generating the knowledge.

The recommendation engine takes as input the current user clickstream and matches all sub-sequences contained within the clickstream to the sequences contained in the sequence tree, recommending the ‘n’ content objects that have the highest confidence values associated with them, given the set of matching sub-sequences.

In addition to the sequence tree, two user-defined parameters bias the recommendation generation by associating a weighting with each sub-sequence matched, based on the recency of the matching sub-sequence within the clickstream and the length of the sub-sequence matched. These parameters take the form of discrete functions, *f(s.l)* and *f(s.r),* that map to the interval [0,1], the length and recency of the sub-sequences respectively. A function, *f(s.r),* that assigns weights to a sub-sequence that is directly proportional to the recency of the sub-sequence, would bias recommendations to those that are generated by sub-sequences that were more recent within the clickstream, allowing the recommendations to reflect the changing context of a user within the visit. A function, *f(s.l),* that assigns a weight to a sub-sequence that is inversely proportional to its length, biases recommendation to those generated by shorter sub-sequences.

The recommendations generated are represented in an XML document referred to as a recommendation pack. An example recommendation pack is shown in Figure 3.

```
<?xml version= "1.0" encoding= "UTF-8"?>
        <recommendation-pack process-time= "551">
           <recommendation description= "description" score= "0.83"
                          title= "Vice President Sales"
                          value= "/people/management/vps.htm">
                <engine name= "Binary Segmentation Advisor">
                <engine name= "Precompiled Sequence Advisor">
           </recommendation>
           <recommendation description= "description" score= "0.74"
                          title= "Chief Executive Officer"
                          value= "/people/management/ceo.htm">
                <engine name= "Precompiled Sequence Advisor">
           </recommendation>
           <recommendation description= "description" score= "0.73"
                          title= "Chief Technology Officer"
                          value= "/people/management/cto.htm">
                <engine name= "Frequency Segmentation Advisor">
                <engine name= "Precompiled Sequence Advisor">
           </recommendation>
        </recommendation-pack>
```

Fig. 3. Recommendation pack example

4.3 Using Multiple Recommenders

The architecture of the IKUM system uses a message driven approach to engaging multiple recommendation engines. Each recommendation engine is implemented as a message driven bean as defined within the EJB2.0 specification. Each of the recommendation engines deployed generates a recommendation pack that consisting of recommendations and an associated perceived value (attribute "score" within the recommendation element) of the recommendations. The knowledge used in the recommendation pack generation depends on the underlying approach used by the recommendation engine and can range from content filtering that use the semantic tagging generated by the IKUM system as defined in Section 4.1, through to behaviour based group profiles (based on Segmentation knowledge) or sequence tree based recommendation engines as outlined in Section 4.2. The IKUM system administrator may currently provide relative weightings to the recommendation engines to create a single consolidated recommendation pack for a web site visitor. More complex mediation strategies are an interesting, open research question.

4.4 Integration with a Content Management System

The Content Management Layer consists of several components such as the Content Management Module, the Classification Module and the Taxonomy Management Module and plays a central role in the IKUM architecture. As the main component of this layer one of the key requirements of the Content Management Module is the provision of tools for managing and administrating web content. Further key requirements consist of the provision and support of functionalities and interfaces for seamless integration between the Content Management Module and the other components within the Content Management Layer as well as the Publishing Module, which belongs to the Interaction Layer. As described in section 4.1, the Content Classification Module uses a thesaurus and a taxonomy to classify the content into semantic categories. Therefore the Taxonomy Management Module provides tools for creating and managing taxonomies. Furthermore it offers the possibility of importing/exporting taxonomies in a standard format allowing the exchange of already available taxonomies, which could be used for certain domains. Within the IKUM system this standard is RDF (Resource Description Framework). The domain expert is able to modify the imported taxonomy in the same way as he would modify a taxonomy created from scratch using the GUI provided by the Taxonomy Management Module.

4.5 Conformance to Standards

The IKUM system is architected with ease-of-integration in mind. As a result, the recommendation engines use XML/PMML input formats. The conformance to PMML enables the use of any data mining vendors' tools that support the PMML standard, as the knowledge generation engine.

Furthermore, the taxonomy used can be imported from RDF and hence the user does not have to use the Taxonomy Management module of the Content Management layer, specifically in order to develop the taxonomy used by the content classification module. Instead, he may use a pre-defined one (e.g. a fraction of the DMOZ [18] taxonomy).

Moreover, the Content Management and Recommendation modules provide a SOAP interface, enabling these modules to be used by any content management or alternative service that controls the interaction with the user.

5 Experimental Evaluation

We presented so far the general architecture framework of the IKUM system, which provides a content delivery platform based on Content, Knowledge and Behavioral data to present personalized content to users in a structure more suited to their needs. The integration of several innovative methods for generating content semantics and sequence-pattern based recommendations, the use of multiple recommendation systems, as well as the conformance to standards such as PMML and SOAP make this system architecture very promising. In this section, we present a set of preliminary user-based evaluation of the content management, the data mining and the knowledge management layers of the system. More on experimental evaluation of several modules of this architecture (namely the Object Mapping, Web Mining, Content Management, and Classification modules) embedded in different system prototypes can be found in [13, 12].

Experimental setup: We chose the web logs of the DB-NET web site[2] (http://www.db-net.aueb.gr), including 300 web pages of different types (html, php, pdf, ppt, doc etc.), collected over a 5 months period (1/9/02-28/2/03). After preprocessing, the total web logs' size was 10^5 hits (order of magnitude) including a set of over 14,000 distinct user sessions. We applied the processes of the Content Management and the Web Mining Layers to the web logs as described in previous sections extracting 500 rules and for each page we stored 1-15 categories relevant to the page's content. We then used the representative ones and grouped the web site documents into 29 clusters. For this purpose, we used a domain-specific taxonomy containing 130 categories.

Table 1. Recommendation sets evaluation based on users' blind testing (1: indifferent, 2: useful, 3: very useful)

Recommendation Set	**Original recommendations usefulness**	**Semantic recommendations usefulness**
A	2.36	1.54
B	1.27	2.09
C	2.09	2.36
Total Average Usefulness:	**1.9**	**2**

We chose the most popular paths followed by the web site visitors. We analyzed the paths and found the best recommendations using the initial usage data (*original* recommendations). We created another set of recommendations taking into consideration the semantic similarity between documents as it is expressed through

[2] We selected this Web site, because the creation/maintenance of a domain-specific taxonomy would be easier and more precise. Moreover, this site contains big amount of information not only in html pages, but also in pages such as php, pdf, ppt etc.

the associated categories included in the Knowledge Base (*semantic* recommendations). We presented each path along with the two sets of recommendations to 12 blind testers and asked them to evaluate them. The recommendation sets were ranked by the users according to their usefulness in the range 1-3 (1: indifferent, 2: good, 3: very good). Therefore, the recommendation sets with higher sums are considered better. The comparative results for the rankings of the original and the semantic recommendations are presented in Table 1.

In the first set, A, the average ranking of the original recommendations is better than the semantic one. This happened because the path included visits to the most important top-level pages of the site. The users preferred the original recommendations including other top-level pages than the semantic ones that concentrated on one subject. In the second set, B, there is a significant difference in favor of the semantic recommendations. This occurred because this set included links to pages that were new, therefore not included in the original recommendations' set, but very relevant to the users' interests. In the third set, C, there is a slight advantage of the semantic recommendations too. The overall recommendation relevance indicates that the blind testers found the semantic recommendations as useful as the original ones.

We should stress, at this point, that for the recommendations' production we did not fully exploit the system's capabilities by utilizing the multiple recommenders' option. Instead, we produced the recommendations using only one recommendation engine (based on Segmentation knowledge). Moreover, we didn't use the online functionality provided by the Interaction Layer. Even in that case, it is evident that users consider the semantic recommendations provided by the system as useful (or even more useful) as the ones that would be recommended initially (original recommendations). Using a combination of both, the end user receives a more cohesive and precise set of recommendations. It is expected that by fully utilizing the system's functionality, the results will be further improved in favor of the recommended approach.

6 Conclusions and Future Work

In this paper we presented a system being developed as part of the IKnowUMine project. The objective is to build a system that incorporates semantics as well as the navigational behavior of users of portals to provide a personalized service. The observation that the context of the user, when using a portal, can vary from one visit to the next, or indeed within the same visit, requires the development of flexible mechanisms for personalization that can adapt to the changing context of the user. Keeping this in mind, an architecture for a system that closely integrates web mining, personalization, content management and portal functionality to deliver personalized content is presented.

The innovations of the presented system with respect to automated content tagging, use of sequence knowledge for recommendation generation, support for multiple recommendation engines, close integration with a content management system and the conformance to industry standards such as EJB2.0, PMML and SOAP are presented in greater detail. Initial tests on a Greek academic web site show promise for the system while highlighting difficulties related to the availability of robust multi-lingual methods for classifying content.

Future work will focus on the evaluation of the system in controlled tests based on commercial portals, to measure the lift in user experience using typical database marketing techniques. Additionally new research areas highlighted by the work, to date, include the lack of robust evaluation methodologies for recommendation engines and mediation strategies when using multiple recommendation engines. The authors are actively researching these research topics.

Acknowledgements

This research work was partially supported by the IST-2000-31077/I-KnowUMine R&D project funded by the European Union.

We would like to thank our testers Foteini, Nikos, Alkis, Vassilis, Maria, Iraklis, Harris, Lily, Stratos, Michalis, and Stratis for their help.

References

[1] S. S. Anand, B. Mobasher (ed.). Working Noted of the IJCAI-01 Workshop on Intelligent Techniques in Web Personlisation, August, 2001.

[2] B. Berendt, Understanding Web usage at different levels of abstraction: coarsening and visualizing sequences, In Proceedings of the Workshop "WEBKDD 2001 - Mining Log Data Across All Customer TouchPoints", 7th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, August 2001, San Francisco, CA

[3] B. Berednt, M. Spiliopoulou, Analysis of navigation behaviour in web sites integrating multiple information systems, The VLDB Journal (2000) 9, 56-75

[4] S. Brin, L. Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, in Proceedings of WWW7, 1998

[5] A.G. Büchner, M. Baumgarten, S.S. Anand, M.D. Mulvenna, J.G. Hughes. Navigation Pattern Discovery from Internet Data, B. Masand, M. Spiliopoulou (eds.) Advances in Web Usage Analysis and User Profiling, Lecturer Notes in Computer Science, Springer-Verlag, 2000.

[6] S. Chakrabarti, B. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, J. Kleinberg, Automatic Resource Compilation by Analyzing Hyperlink Structure and Associated Text, in Proceedings of WWW7, 1998

[7] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, Mining the Link Structure of the World Wide Web, IEEE Computer (1999) Vol.32 No.6

[8] R. Cooley, B. Mobasher, J. Srivastava, Data preparation for mining world wide Web browsing patterns, Knowledge and Information Systems, February 1999/Vol.1, No. 1

[9] F. Coenen, G. Swinnen, K. Vanhoof, G. Wets, A Framework for Self Adaptive Websites: Tactical versus Strategic Changes, in proceedings of WEBKDD'2000 "Web Mining for E-Commerce – Challenges and Opportunities", 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2000, Boston, MA

[10] H. Dai and B. Mobasher, Using Ontologies to Discover Domain-Level Web Usage Profiles, In proceedings of the Second Workshop on Semantic Web Mining, at the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'02), Helsinki, Finland, August 2002.

[11] The PMML2.1 Specification, http://www.dmg.org/pmml-v2-1.html

[12] M. Eirinaki, M. Vazirgiannis, Web Mining for Web Personalization, ACM Transactions on Internet Technology (TOIT), Feb. 2003/vol.3, no.1, 1-27

[13] M. Eirinaki, M. Vazirgiannis, I. Varlamis, SEWeP: Using site semantics and a taxonomy to enhance the Web personalization process, in Proceedings of the 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'03), August, 2003

[14] M. Halkidi, B. Nguyen, I. Varlamis, M. Vazirgiannis, THESUS: Organizing Web Documents into Thematic Subsets using an Ontology, VLDB journal, Nov. 2003/vol.12, No.4, 320-332

[15] B. Mobasher, R. Cooley, J. Srivastava, Automatic Personalization Based on Web Usage Mining, Communications of the ACM, August 2000/Vol. 43, No. 8, pp. 142-151

[16] B. Mobasher, H. Dai, T. Luo, M. Nakagawa. Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. In Data Mining and Knowledge Discovery, Kluwer Publishing, Vol. 6, No. 1, pp. 61-82, January 2002.

[17] M. D. Mulvenna, S. S. Anand and A. Büchner (ed.). Special Issue on Personalization, Communications of the ACM, August, 2000

[18] D. Oberle, B. Berendt, A. Hotho, J. Gonzalez, Conceptual User Tracking, in Proceedings of the 1st AWIC Conference, 2003

[19] The Open Directory Project, http://www.dmoz.org

[20] M. Perkowitz, O. Etzioni, Adaptive Web Sites: An AI Challenge, in Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, Nagoya, Japan, 1997

[21] M. Perkowitz, O. Etzioni, Towards Adaptive Web Sites: Conceptual Framework and Case Study, in Proceedings of WWW8, 1999

[22] M. Perkowitz, O. Etzioni, Adaptive Web Sites, Communications of the ACM, August 2000/Vol. 43, No. 8, pp. 152-158

[23] M. Spiliopoulou, Web Usage Mining for Web Site Evaluation, Communications of the ACM, August 2000/Vol. 43, No. 8, 127-134

[24] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information Processing and Management (1998), Vol. 24, 513-523

[WN] WordNet, A lexical database for the English language, http://www.cogsci princeton.edu/~wn/

[WP94] Z. Wu, M. Palmer: Verb Semantics and Lexical Selection, 32nd Annual Meetings of the Assoc. for Computational Linguistics, 1994

A Semantic-Based User Privacy Protection Framework for Web Services*

Arif Tumer[2], Asuman Dogac[1], and I. Hakki Toroslu[1]

[1] Software Research and Development Center & Dept. of Computer Eng., Middle East Technical University (METU), 06531 Ankara Türkiye
asuman@srdc.metu.edu.tr, toroslu@ceng.metu.edu.tr

[2] Intro Solutions, Ankara Türkiye
arif.tumer@introsolutions.com

Abstract. Web service technology is an Internet-based distributed computing paradigm to address interoperability in heterogeneous distributed systems. In this paper, we present a privacy framework for Web services which allows user agents to automatically negotiate with Web services on the amount of personal information to be disclosed on behalf of the user. In developing this framework the following key privacy considerations are taken into account: revealing only the minimal pertinent information about the user, not to overwhelm the users while declaring their privacy preferences and requiring only limited user interaction.

In the framework proposed, the Web services declare their input parameters as *Mandatory* or *Optional* and allow users to declare how much of their personal information can be made available to the services. The users specify their privacy preferences in different permission levels on the basis of a *domain specific service ontology* based on DAML-S. The major components of the system are a globally accessible context server which stores user preferences and a service registry where the services advertised and the service semantics are available.

1 Introduction

Currently Web services have become a main thrust of the IT industry: Many packaged application vendors, such as SAP, Siebel, and PeopleSoft are moving towards providing Web service APIs; others are building tools that export legacy mainframe applications in Web services form, while still others are making important functionality for business partners available through Web services [4] (e.g., the Amazon.com Web Service [1] and Google Web service API [14]).

Considerable progress has been made in the area of Web service description and invocation: There are two almost universally accepted standards for these purposes: SOAP (Simple Object Access Protocol) [23] for invoking services and

* This work is supported by the European Commission through IST-1-002104-STP Satine project and in part by the Scientific and Technical Research Council of Turkey, Project No: EEEAG 102E035.

WSDL (Web Services Description Language) [31] for describing the technical specifications of the services. There are also two well-known service registries, UDDI [24] by Microsoft and IBM and ebXML [12] by UN/CEFACT.

Well accepted standards like WSDL and SOAP make it possible only to "dynamically access" to Web services in an application. That is, when the service to be used is known, its WSDL description can be accessed by a program which uses the information in the WSDL description like the interface, binding and operations to access the service at run time.

In order to exploit the Web services to their full potential, their semantics must also be available. There is an important initiative in this respect, namely, DAML-S [7] which defines an upper ontology for describing the semantics of Web services. There are also efforts to complement this upper ontology with domain specific service ontologies such as the tourism ontology given in [21] and the bioinformatics ontology given in [29].

Describing the semantics of Web services improves the Web service technology in several respects such as being able to define the properties of services like their real life counterparts and facilitating automated service discovery and composition.

Another area that can benefit from the semantics of Web services is protecting user's privacy when accessing the Web services. There are some important considerations in developing privacy mechanisms:

– Only the minimal pertinent information should be provided to the Web service to prevent disclosing unnecessary personal information. As an example, a user may have to provide her credit card number when invoking a "purchasing" service but may prefer not to do so for example for a "reservation" service.
– Another critical issue is not to overwhelm the users while declaring their privacy preferences. Indeed declaring privacy prefences on the basis of service instances may be quite cumbersome and sometimes even not possible. A user may not in advance know which service she will need.
– Determining whether the data requested by a Web service violates user's privacy preferences should be automatic. Current privacy management mechanisms like P3P [6] are oriented towards Web browsing and thus require user interaction.

In this paper we address protecting the users' privacy when using Web services by addressing the issues mentioned above. We allow Web services to declare their input parameters related with personal user information as *Mandatory* or *Optional.* We show how DAML-S Service Profile input parameter specification [7] can easily accommodate the changes to differentiate between input parameters that are essential for the service to execute from those which are optional. Optional parameters are those a service provider is requesting for its own use.

The services also declare alternate data element requests in case that a user does not want to provide some mandatory input parameters. For example, if a contact address is necessary for the service and the user is not giving her mobile phone number, her email address may be requested instead.

We developed a framework to allow users to declare how much of their personal information can be made available to the services. The users declare their privacy preferences as *Free*, *Limited*, or *NotGiven* on the basis of a domain specific service ontology. As an example, assume a service ontology in the "travel" domain (Figure 5). A "Hotel Reservation" service (a node in the ontology) may require the user's name, email address, and date of reservation as *Mandatory* and a credit card number as *Optional.* A user on the other hand, may declare that for any hotel reservation service, she will provide all the requested information mentioned as *Free* except her credit card number (*NotGiven*). Furthermore, a user may declare her e-mail address as *Limited* in which case it is given to the service only if it is mandatory for the execution of the service.

The approach presented has the following advantages:

– The user preferences are defined not for individual service instances but for a node in the service ontology and thus applies to all instances unless the user overrides this explicitly. Furthermore, a user may declare the same policy for several different service classes. The effort required by the user is further minimized since the privacy preferences at the upper level classes of the ontology are inherited by lower level service classes. Note that a user can override a privacy preference at any level she likes.
– The presented framework allows Web services to declare alternate data requests if a mandatory input is not given by the user. This provides flexibility and creates room for reaching an agreement through negotiation.
– Finally, we believe that declaring the user preferences based on a standard service ontology like DAML-S helps with the interoperability problem.

The work accomplished in this paper is realized within the scope of the IST-1-002104-STP Satine Project [21],[11]. Satine aims to develop a semantic Web service based interoperability framework for tourism industry. In order to semantically annotate Web services, travel domain ontologies like Harmonise [15] are used. The project also proposes Web service ontologies based on the recent travel industry message specifications produced by Open Travel Alliance (OTA) [22].

The paper is organized as follows: In Section 2, a privacy framework for Web services based on domain specific service ontologies is presented. Section 3 describes the negotiation process between the user agent and the Web service. In Section 4, an example scenario is given to illustrate the concepts. Section 5 describes the related work. Finally, Section 6 concludes the paper.

2 A Privacy Framework for Web Services

We propose the following basic elements for the privacy model of the Web services:

– *Policy Statement:* Web services describe their business practices regarding the use of personal user information in *Policy Statements* - or *Service Policies.*

- *Input Requests:* The data set requested from the user as the input parameters of a Web service.
- *Privacy Preferences:* The user is responsible for declaring her *Privacy Preferences* regarding the policy statements and input parameters of Web services. The declaration of privacy preferences is based on domain specific service ontologies developed for the Web service messages. A user declares her privacy preferences by annotating the messages of the Web service through ontology nodes as explained in Section 2.3. Note that the privacy preferences declared for a node is inherited by its subclasses in the class hierarchy.
- *Service Request Analyzer:* This component is responsible for parsing and analyzing data request files and the policy statements given by the service provider. The relevant rules in these files are converted into internal representations appropriate for the applications that manages the negotiation process.
- *Rule Extractor:* This component determines the user's privacy rules regarding a Web service, to be utilized during negotiation between the user agent and the service. Rule extraction is explained in Section 3.1.
- *Negotiation Component:* Based on the rules declaring the service's request and rules describing the privacy preferences of the user, the negotiation component tries to find a ground for agreement between the user agent and the Web service. Comparing the necessity rules and permission levels and employing the alternatives, this component generates an *agreed-upon element set* that describes the elements that will be sufficient for the enactment of the service and allowed by the user. Negotiation mechanism is presented in Section 3.2.

2.1 Declaration of Privacy Policy

Policy statements provide a way to describe the data use practices. Web services should also declare their policies regarding their input parameters such as their purpose in requesting the data, with whom they may share the data and whether and how they will retain data. For this purpose, we adapted the P3P policy mechanism to Web services as follows:

- *Purpose* section describes the basis for collecting user's data,
- *Recipient* section declares the entities with whom the data will be shared,
- *Retention* section defines the activity scope during which the data will be retained.

A possible mechanism for Web services to declare such policies is to exploit DAML-S Service Profile input parameter. DAML-S service profile describes "what the service does"; that is, it gives the type of information needed by a service-seeking agent to determine whether the service meets its needs, typically such things as input and output types, preconditions and postconditions, and binding patterns [7].

We propose to specify the Web service privacy policies in DAML-S as shown in Figure 1.

```
<rdf:Property rdf:ID="input">
  <rdfs:subPropertyOf rdf:resource="&process;#parameter"/>
  <rdfs:domain rdf:resource="&service;#ServiceProfile"/>
  <rdfs:range rdf:resource="InputSpecs"/>
</rdf:Property>

<rdf:Property rdf:ID="purpose">
  <rdfs:subPropertyOf rdf:resource="#input"/>
  <rdfs:domain rdf:resource="InputSpecs"/>
  <rdfs:range rdf:resource="&daml;#Thing"/>
</rdf:Property>

<rdf:Property rdf:ID="recipient">
  <rdfs:subPropertyOf rdf:resource="#input"/>
  <rdfs:domain rdf:resource="InputSpecs"/>
  <rdfs:range rdf:resource="&daml;#Thing"/>
</rdf:Property>

<rdf:Property rdf:ID="retention">
  <rdfs:subPropertyOf rdf:resource="#input"/>
  <rdfs:domain rdf:resource="InputSpecs"/>
  <rdfs:range rdf:resource="&daml;#Thing"/>
</rdf:Property>
```

Fig. 1. Web Service Privacy Policies in DAML-S

Disputes and remedies are not incorporated into this work, as they are high level statements, accompanied by textual descriptions, hence not generally machine-processable. A further point to note is the following: our privacy framework allows for *Purpose* and *Recipient* policies to be defined within multi level taxonomies [16] since subtypes are necessary for expressing more fine-grained policies as explained in Section 2.3.

There are two basic elements in the privacy model: the data requested by Web services and the privacy preferences of the users. These issues are discussed in the following subsections.

2.2 Specifying Data Requests of Web Services

We define the data requested by a Web service to be composed of three parts: the first one is the set of elements requested by the Web service (that is, the input parameters of the service). The second one is the declaration of how essential the data is for the service to execute. Finally, a Web service may also provide rules stating alternate data elements if a mandatory piece of information is not provided by the user. For example a rule may state that if a user is not willing to disclose her email address, she should provide her postal address. Alternatives may help to reach an agreement during negotiation.

We use DAML-S service profile input parameter definition to specify whether the input parameter is essential for the service to execute (i.e., mandatory) or it is requested for some other purpose (i.e., optional) as shown in Figure 2.

We define alternative data requests through *Conditional Request* and express them in RDF. A conditional request is an "if-then" rule describing what alternate data elements may be of use if some mandatory data elements are not given by

```
<rdf:Property rdf:ID="mandatory">
<rdfs:subPropertyOf rdf:resource="&process;
                                  #inputParameter"/>
</rdf:Property>
<rdf:Property rdf:ID="optional">
<rdfs:subPropertyOf rdf:resource="&process;
                                  #inputParameter"/>
</rdf:Property>
```

Fig. 2. Defining the types of Data Element Requests through DAML-S

```
<pri:IfRule>
  <pri:If rdf:parseType="Resource">
    <pri:NotGiven rdf:resource="...#emailAddress"/>
  </pri:If>
  <pri:Then rdf:parseType="Resource">
    <pri:Mandatory rdf:resource="...#postalAddress"/>
  </pri:Then>
</pri:IfRule>
<pri:IfRule>
  <pri:If rdf:parseType="Resource">
    <pri:NotGiven rdf:resource="...#creditCardNo"/>
  </pri:If>
  <pri:Then rdf:parseType="Resource">
    <pri:Mandatory rdf:resource="...#bankAccountNo"/>
  </pri:Then>
</pri:IfRule>
```

Fig. 3. Example Conditional Statements

the user for a specific service class. Example conditional statements are presented in Figure 3. These rules state that if the user is not providing an "EmailAddress" and a "CreditCardNo" which are essential for the service to execute then, the user should provide a postal address and a bank account number, respectively.

2.3 Describing User Privacy Preferences

The users define their privacy preferences for Web services through a rule-based mechanism with a reference to a domain specific service ontology.

There are three permission level rules that can be imposed on a data element for a given service class:

– *Free*: The data element is given freely by the user.
– *Limited*: The data element is provided by the user only if it is mandatory for the service enactment.
– *NotGiven*: The given data element is not provided by the user.

As an example consider the rule segment given in the Figure 4 and an example travel ontology given in Figure 5. The first rule is associated with the top level service class defined in this ontology (Figure 5). For the top level class, the user releases her home phone number freely but her age information is limited (that is it will be provided only if the service declares it to be mandatory). Through the

```
<pri:Rule>
 <pri:Role rdf:resource=".../TravelService"/>
 <pri:Data rdf:parseType="Resource">
    <pri:Limited rdf:resource="...#age"/>
    <pri:Free rdf:resource="...#home.Phone"/>
 </pri:Data>
</pri:Rule>
<pri:Rule>
 <pri:Role rdf:resource=".../TransportationService"/>
 <pri:Data rdf:parseType="Resource">
   <pri:NotGiven rdf:resource="...#creditCardNo"/>
   <pri:NotGiven rdf:resource="...#mobile.Phone"/>
   <pri:Limited rdf:resource="...#emailAddress"/>
   <pri:Free rdf:resource="...#name"/>
 </pri:Data>
</pri:Rule>
<pri:Rule>
 <pri:Role rdf:resource=".../BuyTicket"/>
 <pri:Data rdf:parseType="Resource">
  <pri:Free rdf:resource="...#creditCardNo"/>
 </pri:Data>
</pri:Rule>
```

Fig. 4. User Context Privacy Rules

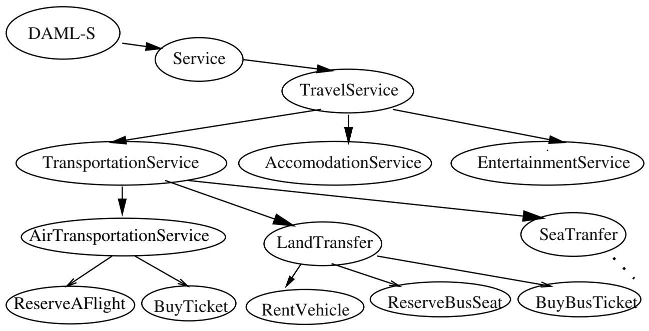


Fig. 5. An Example Class Hierarchy for Travel Domain

second rule associated with the “TransportationService”, the user does not give her credit card number and mobile phone number; her email address is limited but her name is freely accessible. Finally, through the third rule, she overrides the rule related with her credit card number and provides this information freely to the *BuyTicket* (Figure 5) class of services.

Different rules may impose conflicting permissions on data elements. When a conflict arises among rules associated with a given service, the final rule for the data element is determined based on the rule priorities. *NotGiven* rule dominates over other rules. *Free* rule has the least priority and *Limited* rule’s priority is between these two. Among the rules associated with a data element, the one with the highest priority, i.e. the most restricted permission level, is chosen to be the rule for that element.

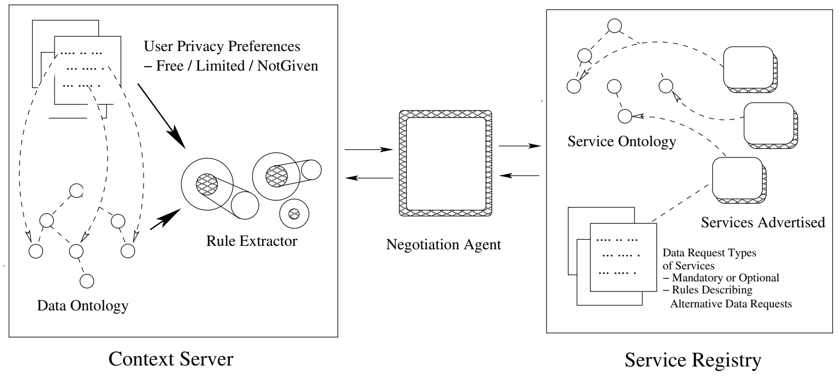


Fig. 6. General Architecture of the System

2.4 Architecture of the System

The general architecture of the system is shown in Figure 6. A context server, which is a trusted authority, stores the privacy preferences of a user based on domain specific service ontologies. The service registry, on the other hand, stores the advertised services, their semantic descriptions and the rules for alternate data requests by the Web Service. We propose the service semantics to be stored by conforming to the DAML-S upper ontology.

3 Negotiation Component

Negotiation is the set of activities where the user's data privacy preferences are compared with service's data request in order to reach an agreement. For this purpose, first the data requests of the Web service along with the rules defining alternatives are obtained from the service registry. Then user's rules are compared with Web service's data requests. If an agreement cannot be reached, the alternative data requests expressed through conditional statements provided by the service is used.

Rule extraction facility is provided by the context server, while the negotiation component is used by the user agent.

3.1 Extraction of Preference Rules

The initial activity of the rule extraction process is to obtain the Web service's data requests from the service registry. In order to find the rules governing the privacy of the data elements requested by the Web service, a temporary service graph is created. This graph is used for determining the privacy rules for those data elements that do not have any rule associated with themselves and need to inherit them from their parent nodes in the ontology.

As an example assume that the user wishes to invoke a *BuyTicket* service and the data requests for this service are a "name", and a "credit card number".

Assume further that the user privacy rules at this level provide for "credit card number" but no rule is given for "name" as shown in Figure 4. The privacy rule for this data element should be obtained from the rules given at the higher levels of the temporary service graph.

The rule extraction process has two phases:

- 1st Phase: Upward Traversal
  - At each node, extract rules related with the needed data elements.
  - Request the rules from parents for undetermined data elements.
- 2nd Phase: Downward traversal
  - For each data element that is needed, receive the rule from the parents.
  - Based on the priorities determine the final rule.
  - Push rules downwards in the temporary service graph. Output the rules applicable to the data elements requested by the service.

If more than one rule is applicable to a data element, the final rule is determined from the rule priorities as mentioned in Section 2.3. The conflict resolution basically declares that the most restricted rule should always be chosen, e.g. *NotGiven* over *Limited* rule.

During the second phase, where the temporary service graph is traversed top-to-bottom, the rules extracted at each node are pushed to the children, and incorporated into the rules of child nodes. In this way the final rule set is collected at the node of the requestor service. When there are more than one parent service nodes, the final rule associated with an element is determined by the resolution process mentioned above, i.e. the most restricted permission level is chosen.

3.2 Negotiation of Data Elements

When the data provided by the user does not match with the data requested by the service, that is, when there is a mandatory data element requested by the Web service that is not given by the user, the alternative rules provided by the Web service (if any) are used to reach an agreement.

Negotiation process basically tries to find out if another data element can be used in place of a "not given" but "mandatory" data element. The aim is to determine the set of elements that can be exchanged between the parties, without violating user's privacy.

4 An Example Scenario

In this section, we provide a scenario to better illustrate the concepts presented. Assume a domain specific service ontology for travel domain as given in Figure 5 and assume that the user wishes to invoke a service which is a member of *BuyTicket* class. Assume further that *BuyTicket* class of services require user's name, mobile phone number and credit card as mandatory data elements. User's age and email address are optional information for the service as shown in Figure 7.

The *BuyTicket* class of services further declare that if user's mobile phone number is not available then her email address is mandatory and her home phone is an optional data element. Recall that all this information is stored with the service registry. We process this information to obtain the rules given in Figure 8.

```
<daml:Class rdf:ID="BuyTicket">
  <rdfs:subClassOf rdf:resource=
                "#AirTransportationService"/>
  </rdfs:subClassOf>
</daml:Class>
<rdf:Property rdf:ID="name">
   <rdfs:subPropertyOf rdf:resource="#mandatory"/>
   <rdfs:domain rdf:resorce="#BuyTicket"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
<rdf:Property rdf:ID="mobile.Phone">
   <rdfs:subPropertyOf rdf:resource="#mandatory"/>
   <rdfs:domain rdf:resorce="#BuyTicket"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
<rdf:Property rdf:ID="creditCardNo">
   <rdfs:subPropertyOf rdf:resource="#mandatory"/>
   <rdfs:domain rdf:resorce="#BuyTicket"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
<rdf:Property rdf:ID="age">
   <rdfs:subPropertyOf rdf:resource="#optional"/>
   <rdfs:domain rdf:resorce="#BuyTicket"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
<rdf:Property rdf:ID="emailAddress">
   <rdfs:subPropertyOf rdf:resource="#optional"/>
   <rdfs:domain rdf:resorce="#BuyTicket"/>
   <rdfs:range rdf:resource="&xsd;string"/>
</rdf:Property>
```

Fig. 7. Input Parameters of *BuyTicket* Class

Note that the actual service instance may request other *interactive* parameters that are not directly related with the privacy of the user's context information. Such data may be received either directly from the user or through the user's agent. For example, for a ticket buying service, this information may include the flight destination and seat class.

4.1 Rule Extraction

The initial activity of rule extraction is to generate a temporary service graph that contains the service node in question (*BuyTicket*) and all of its ancestors. Figure 9 presents temporary service graph for *BuyTicket* service class. The corresponding data request of the service is given in Figure 8.

In the first phase of the rule extraction process, the service ontology is queried to extract the input parameters and the alternate data request rules

```
<pri:Data rdf:ID="Data">
    <pri:Mandatory rdf:resource=
        ".../UserContextTaxonomy#Name"/>
    <pri:Optional rdf:resource=
        ".../UserContextTaxonomy#Age"/>
    <pri:Mandatory rdf:resource=
        ".../UserContextTaxonomy#Mobile.Phone"/>
    <pri:Optional rdf:resource=
        ".../UserContextTaxonomy#EmailAddress"/>
    <pri:Mandatory rdf:resource=
        ".../UserContextTaxonomy#CreditCardNo"/>
</pri:Data> <pri:IfRule>
    <pri:If rdf:parseType="Resource">
        <pri:NotGiven rdf:resource=
            ".../UserContextTaxonomy#Mobile.Phone"/>
    </pri:If>
    <pri:Then rdf:parseType="Resource">
        <pri:Mandatory rdf:resource=
            ".../UserContextTaxonomy#EmailAddress"/>
        <pri:Optional rdf:resource=
            ".../UserContextTaxonomy#Home.Phone"/>
    </pri:Then>
</pri:IfRule>
```

Fig. 8. *BuyTicket* Web service's Data Request

of *BuyTicket* service class. The user context rule segment associated with *BuyTicket*, declares that the user releases `CreditCardNo` freely to the services of this class as shown in Figure 4. The service is in need of further data elements and the upper levels of the temporary service graph is processed to obtain these data elements.

Figure 10 shows which higher level service classes provide the data elements requested by *BuyTicket* class as *Free* or *Limited* elements after completing the first phase of the process. The data elements, for which no rule association has been found, constitute the *Needs* set of that node.

During the second phase of rule extraction process, the temporary service graph is traversed downwards starting from the *TravelService* node, while each service receives rules for the elements it needs from its parent service node.

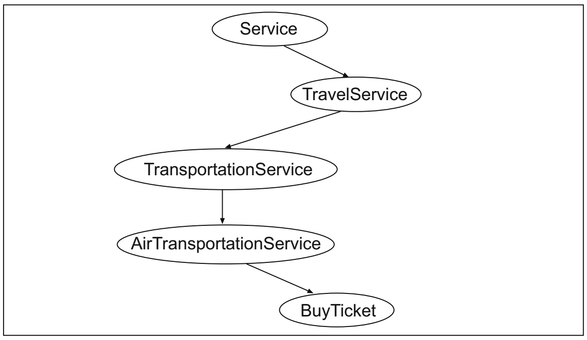


Fig. 9. Temporary service graph generated for BuyTicket service

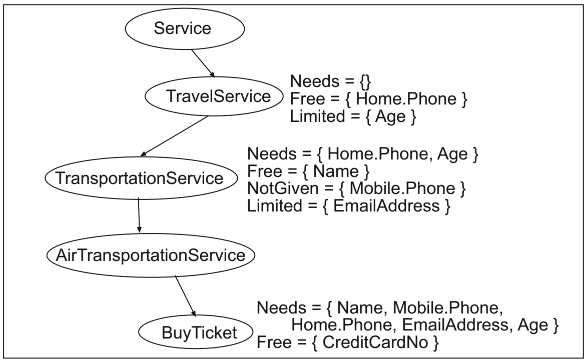


Fig. 10. State of temporary service graph, after Phase 1

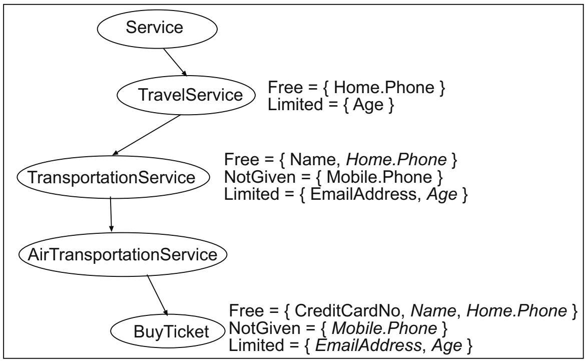

Service
TravelService
Free = { Home.Phone }
Limited = { Age }
TransportationService
Free = { Name, *Home.Phone* }
NotGiven = { Mobile.Phone }
Limited = { EmailAddress, *Age* }
AirTransportationService
BuyTicket
Free = { CreditCardNo, *Name*, *Home.Phone* }
NotGiven = { *Mobile.Phone* }
Limited = { *EmailAddress*, *Age* }

Fig. 11. State of temporary service graph, at the end of rule extraction process

Figure 11 presents the temporary service graph at the end of the second phase of rule extraction. The data elements shown in italics are the ones inherited from parent services in the graph. The permission rules collected at $BuyTicket$ service node define the rules associated with each element referred in the data request.

The negotiation process may start when the user's privacy preferences regarding the data elements, i.e. the permission levels, requested by the service are determined.

4.2 Negotiation

Following the running example, note that the mandatory `Mobile.Phone` is not given to the service. However, through the alternative rules (Figure 8), the service states that it accepts email address in place of mobile phone number. In addition, home phone number is also requested as an optional data element. Therefore, the conditional request is triggered, and the new alternative requests are added into the original input set. Note that, initially, user's email address was an optional data element for the service. When the conditional statement is triggered, this item becomes mandatory in the input set.

The resulting input set is as follows:

Mandatory	= { `Name`, `CreditCardNo`, `EmailAddress` }
Optional	= { `Age`, `Home.Phone` }

The permission levels of data elements obtained from the rule extraction process is as follows:

Free	= { `Name`, `Home.Phone`, `CreditCardNo` }
Limited	= { `Age`, `EmailAddress` }
NotGiven	= { `Mobile.Phone` }

In the final phase of the negotiation activities, the necessity levels for the requested data elements are compared with the permission levels extracted from user's data privacy preferences. The mandatory data elements `Name` and `CreditCardNo` are provided with *Free* rule, hence the user agrees to provide these data elements. The mandatory data element `EmailAddress` is associated with *Limited* rule in the privacy preferences of the user. As this element is crucial for the service enactment, it is also released by the user.

The home phone number of the user is provided freely, independent of the necessity level. Hence, it is included in the agreement set. However, the age of the user is not presented to the service, because the privacy preferences states that this element is provided in a limited fashion. Recall that, elements that are associated with *Limited* rules in the privacy preferences are released only if they are requested with *Mandatory* necessity level.

Even if the data element `Age`, is not released to the service, it is not a mandatory element for the service, hence does not hinder the service enactment. Therefore, there are no conflicts between service's input request and user's privacy preferences. An agreement is settled between the parties. The agreed-upon data set, which determines the elements that may be passed to the service, is presented in the following:

Mandatory	= { `Name`, `CreditCardNo`, `EmailAddress` }
Optional	= { `Home.Phone` }

5 Related Work

Authentication services like Microsoft Passport [19] and AOL Screen Name [2], store and manage personal user data and provide single sing-in identity at different sites and pass personal information more easily. The stored personal data is generally limited to user identification and user contact information that can be used in basic e-commerce sessions.

[13] describes the ePerson project developed at the HP Labs. An ePerson is a personal representative on the net that is trusted by a user to store and

make personal information available under appropriate controls. Such personal information includes user profiles, shared content and shared meta-data (such as annotations, comments, ratings and categorisations). However how privacy issues are handled in ePerson is not available in the literature.

[18] defines a vocabulary for composing policies to allow or deny access to the personal information that a policy governs. The work also describes how policies can be merged using negotiation rules and how Semantic Web logic processors reason through policies.

None of the work described above addresses how to describe the privacy preferences for Web services.

Among several approaches for privacy management using service policies and privacy preferences, the most mature one is the Platform for Privacy Preferences Project (P3P) [6] developed by the World Wide Web Consortium (W3C). P3P provides a simple, automated way for users to gain more control over the use of personal information on Web sites they visit. At its most basic level, P3P is a standardized set of multiple-choice questions, covering all the major aspects of a Web site's privacy policies. Taken together, they present a clear snapshot of how a site handles personal information about its users. P3P-enabled Web sites make this information available in a standard, machine-readable format. P3P enabled browsers can "read" this snapshot automatically and compare it to the consumer's own set of privacy preferences. P3P enhances user control by putting privacy policies where users can find them, in a form users can understand, and, most importantly, enables users to act on what they see.

The P3P Specification 1.0 [6] includes the definition of the syntax and semantics of a vocabulary to describe data uses, data recipients, data retention policy and other privacy disclosures in P3P privacy policy files. A base data schema defines a standard set of data elements that will be referenced from these policies, as well as a mechanism for associating policies with Web resources.

P3P policies are written in machine readable XML [27] format, facilitating XML Namespaces [28] to refer to the P3P policy vocabulary elements. The privacy vocabulary elements and the base data schema are defined using XML Schema [25, 26]. P3P policies declare the data elements that will be collected by the Web site and explain how the data will be used. They also identify the recipients of the collected data, and make a variety of other disclosures including information about dispute resolution procedures and remedy processes. Textual description about the company and related contact information, may also be included in the policy files.

APPEL (A P3P Preference Exchange Language) [5] provides a standard way of defining the user privacy preferences in a set of preference rules, which can be used by the user agent to make automated and semi-automated decisions regarding the acceptance of privacy policies from P3P enabled Web sites. While the user agent may present the user preferences in some internal format, APPEL provides a standard way to do this.

Recently, the Web Services Architecture (WSA) Working Group at W3C which is tasked with producing a Web service reference architecture, has produced the critical success factors and requirements for the privacy of Web services [30]:

– The WSA must enable privacy policy statements to be expressed about Web services.
– Advertised Web service privacy policies must be expressed in P3P.
– The WSA must enable a consumer to access a Web service's advertised privacy policy statement.
– The WSA must enable delegation and propagation of privacy policy.
– Web Services must not be precluded from supporting interactions where one or more parties of the interaction are anonymous.

Although WSA encouraged the use of P3P, it is later stated that P3P cannot be used in situations where a request is not directed to a URI, for example, some applications of Web Services and SOAP [20].

WSA described the very basic requirements to enable privacy protection for the consumer of a Web service across multiple domains and services. We extend this basic architecture by exploiting the semantics of Web services. It should also be noted that P3P does not intend to exploit Web semantics.

Infact only a few recent work address semantic issues for privacy management: [17] points out that a standard method of exchanging privacy policies, that is a privacy ontology, is needed for the Semantic Web.

6 Conclusions and Future Work

Privacy preferences of a user define the rules that control the read access for personal information. In related specifications like P3P, privacy preferences are based on URLs' of Web sites, as these technologies are mostly intended for Web browsing applications and interactive e-commerce sessions.

In this work, a privacy framework for Web services is proposed. Declaring privacy preferences on the basis of a service ontology prevents the user from repetitive specifications since the privacy preferences at the upper classes of the ontology are inherited by lower classes. Furthermore the framework presented allows Web services to declare alternate data requests if a mandatory input is not given by the user. In this way it becomes possible to automate the negotiation process with a Web service to reach an agreement.

The features provided within the proposed privacy scheme, i.e. service-class-based preferences, multi-level element types, consent-based policies and negotiation activities, are utilized to enhance the process of decision making in agent programs. What distinguishes our work is that in this privacy framework, domain specific Web service ontologies are used. How service ontologies can be stored into service registries and how service semantics can be related with the services advertised are available from our accompanying work [8, 9, 10].

There are a number of issues left as a future work:

– Privacy preferences should also include user's choice of accepted data-use practices, such as the data retention policies of the service.

– What Web services need to know is not only user preferences but a “user context” that includes any information that can be used to characterize the user and her situation. Hence user context should include user’s local data obtained through sensors as well as any data stored about the user such as those stored in customer relationship management (CRM) systems to make effective use of Web services.
– This context information should be available to any authorized agent at any time, anywhere in a secure manner: This necessitates developing secure “context servers”, that is, the user context information should be available anywhere, any time to a variety of devices from desktops to mobile devices. Since these devices accept input in different mark up languages; the context servers need to recognize the device and provide the information in the format that can be accepted by the device. Note that if the user permits, information on user activities should be collected to further improve user context.
– More importantly, user context should be available in a format that is machine processable and interoperable. In this respect developing a user context ontology is essential.
– Yet all this will make privacy a graver concern for users. There is a need for trusted authorities for delivering user context to authorized requestors in a secure manner.

References

1. Amazon.com Web services, http://www.amazon.com/gp/browse.html/002--8640824-9000064?node=3435361
2. AOL Screen Name, http://my.screenname.aol.com
3. Bargh, M. S., van Eijk, Ebben, P., Salden, A. H., “Agent-based Privacy Enforcement of Mobile sevices”, in Proc. of SSGRR Conference, Italy, January 2003.
4. Carey, M., Blevins, M., Takacsi-Nagy, P., “Integration, Web Services Style”, IEEE Data Engineering Bulletin, Vol. 25, No. 4, December 2002.
5. Cranor L., Langheinrich M., and Marchiori M., *A P3P Preference Exchange Language 1.0 (APPEL 1.0)*, W3C Working Draft, `www.w3.org/TR/P3P-preferences`, April 15, 2002
6. Cranor L., Langheinrich M., Marchiori M., Presler-Marshall M., Reagle J., *The Platform for Privacy Preferences 1.0 (P3P1.0) Specification*, W3C Recommendation, `www.w3.org/TR/P3P`, April 16, 2002.
7. DAML Services Coalition (A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, S. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, H. Zeng), DAML-S: Semantic Markup for Web Services, in Proceedings of the International Semantic Web Working Symposium (SWWS), July 2001.
8. Dogac, A., Laleci, G., Kabak, Y., Cingil, I., “Exploiting Web Service Semantics: Taxonomies vs. Ontologies”, IEEE Data Engineering Bulletin, Vol. 25, No. 4, December 2002, http://www.research.microsoft.com/research/db/debull/issues-list.-htm.
9. Dogac, A., Cingil, I., Laleci, G. B., Kabak, Y., “Improving the Functionality of UDDI Registries through Web Service Semantics”, 3rd VLDB Workshop on Technologies for E-Services (TES-02), Hong Kong, China, August 23-24, 2002.

10. Dogac, A., Kabak, Y., Laleci, G., "Enriching ebXML Registries with OWL Ontologies for Efficient Service Discovery", 14th Intl. Workshop on Research Issues on Data Engineering, Boston, USA, March 2004.
11. Dogac, A., Kabak Y., Laleci, G. , Sinir S., Yildiz A., Kirbas S., Gurcan Y., "Semantically Enriched Web Services for Travel Industry", ACM Sigmod Record, Vol. 33, No. 3, September 2004.
12. ebXML, http://www.ebxml.org/
13. e-person: Personal Information Infrastructure, http://www.hpl.hp.com/semweb/-e-person.htm
14. Google Web Service API, http://www.google.com/apis/
15. Harmonise Project, IST-2000-29329, Tourism Harmonisation Network, http://www.harmonise.org/
16. Karjoth, G., Schunter, M., *A Privacy Model for Enterprises*, 15th IEEE Computer Security Foundations Workshop, June 24-26, 2002.
17. Kim, A., Hoffman, L. J., Martin, C. D., "Building Privacy into the Semantic Web: An Ontology Needed Now", in Proc. of Semantic Web Workshop, Hawaii, USA, 2002.
18. Lee, R., "Personal Data Protection in the Semantic Web", ME Thesis, MIT, USA, 2002, http://www.w3.org/2002/01/pedal/thesis.html
19. Microsoft Passport, http://www.microsoft.com/myservices/passport.
20. The Platform for Privacy Preferences 1.1 (P3P1.1) Specification, W3C Working Draft 4 January 2005, http://www.w3.org/TR/2005/WD-P3P11-20050104/
21. IST-1-002104-STP Satine Project, http://www.srdc.metu.edu.tr/webpage/-projects/satine
22. Open Travel Alliance (OTA), http://www.opentravel.org/
23. Simple Object Access Protocol (SOAP), http://www.w3.org/TR/SOAP/
24. Universal Description, Discovery and Integration (UDDI), www.uddi.org
25. Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn, N., *XML Schema Part 1: Structures*, W3C Recommendation, `www.w3.org/TR/xmlschema-1`, May 2, 2001.
26. Biron, P., Malhotra, A., *XML Schema Part 2: Datatypes*, W3C Recommendation, `www.w3.org/TR/xmlschema-2`, May 2, 2001.
27. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., *Extensible Markup Language (XML) 1.0 (Second Edition)*, W3C Recommendation, `www.w3.org/TR/REC-xml`, October 6, 2002.
28. Bray, T., Hollander, D., Layman, A., *Namespaces in XML*, W3C Recommendation, `www.w3.org/TR/REC-xml-names`, January 14, 1999.
29. Wroe, C., Stevens, R., Goble, C., Roberts. A., Greenwood, M., "A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data", Intl. Journal of Cooperative Information Systems, to appear.
30. Web Services Architecture Requirements, http://www.w3.org/TR/2004/NOTE--wsa-reqs-20040211/
31. Web Service Description Language (WSDL), http://www.w3.org/TR/wsdl

Web Personalisation for Users Protection: A Multi-agent Method

Samir Aknine, Aurélien Slodzian, and Ghislain Quenum

Laboratoire d'Informatique de Paris 6
{samir.aknine, aurelien.slodzian, jose.quenum}@lip6.fr

Abstract. "*Interracial Breeding Destroys The White Race...*", "*Jews and Arabs should both be kicked out of the White West.*" are examples of sentences that are easy to find on the Internet. What can be done to help those who want to protect themselves from such discourse? Put another way, how can we personalise web browsers to protect the users who desire so? This is the difficult subject raised in this article and to which we propose a multi-agent solution. This solution involves a multi-dimensional linguistic analysis of the content of the documents and the role of the multi-agent system is to combine these dimensions. The multi-agent model we propose combines a pyramidal coordination structure with agent associations, two notions which we introduce herein and of which we demonstrate the relevance.

Keywords: Multi-agent systems, information retrieval, applications.

1 Introduction

Web personalisation is a recent topic introduced in artificial intelligence (AI) community. Through this research topic, AI methods can be applied in order to ease web users information search or protect them against undesirable information. The Princip project[1] was initiated as an attempt to provide protection against racist and hate speech on the Internet. Most racist authors try to publicise their point of view while hiding the nature of their discourse behind innocent words, in an attempt to circumvent legal or web provider regulations. Hence usual keyword based approaches simply do not suit.

The practical goal of this project is twofold: firstly set up a web crawler that will repeatedly look for racist documents and secondly provide the list of identified racist sites to self-protection programs, either individual or collective. In this paper we focus on the first issue: how the collection of racist web sites can be constructed before one can use them to personalise web browsers. We aim to develop both theoretical and practical tools which should be embedded inside web browsers and prevent users who wish so from accessing racists and revisionists web sites. Such tools are of course for a more general purpose since one can configure them and use in other contexts, for example religious contents web sites. In particular, we motivate the use of a multi-agent system for combining efficiently different textual analysis techniques. These techniques

[1] This project is funded by the Safer Internet Action Plan of the European Commission, www.princip.net.

are used to analyse and filter the output of classical Internet search engines, to which wide spectrum lists of keywords are submitted at regular time intervals. The involved techniques include computational linguistic techniques like terminological databases, derivational morphology, part-of-speech tagging, etc. The usage of such deep analysis techniques is mandatory since the racist nature of a document may not be guessed simply from the presence of various keywords. This approach imposed a preliminary analysis of the racist documents, which resulted in the collection of a corpus of sample racist and anti-racist documents[2]. From this corpus, a number of more or less reliable criteria of racism have been identified.

This raised a first issue: all criteria are weak, in the sense that they capture *differences* between racist and non-racist documents but these differences do not *characterize* the racist content by itself. We elaborate on this issue in Section 2 where these criteria are briefly presented and discussed. As a second issue, we were confronted with the lack of formal models for combining these numerous criteria. Thus, a multi-agent approach was appealing. By associating one criterion with one agent, criteria combination effectively comes down to agent coordination. Furthermore, the absence of a static algorithm imposes that the candidate coordination models are dynamic ones.

This article presents an original multi-agent coordination scenario, which has the main property of allowing for an efficient organisation of the computer resources without relying on central control nor on plan sharing. It is presented in Section 4. More precisely, we propose a coordination model based on a dynamic pyramidal coordination structure combined with agent associations. Both notions are described in section 4.2 and the latter will be compared with the notions of coalition [9, 8], team [10] and congregation [3]. Shortly stated, the main coordination structure imposes constraints on resource usage and places agents in concurrence, forcing them to choose optimal solutions for document analysis. The associations between agents allow them to set up temporary interest groups for combining their computation power.

We formalise the content analysis problem in Section 2 and describe some criteria. Then we discuss the related work in Section 3. Afterwards, we go into the depth of our coordination model in Section 4. Finally, we discuss the implementation aspects in Section 5.

2 The Problem

2.1 Issues with the Racist Web

The tracking of racist documents on the Internet involves a number of obstacles which make it difficult to rely on the classical keyword-based approaches or on neural network techniques."

1. The racist discourse spans from hate speech to more subtle insinuations.
   - Different themes: racist, revisionist (denies the existence of the holocaust), anti-Semitic, etc.

[2] The size of the corpus is one million words per type of documents and per language (English, french and German).

- Different kinds of discourses : political, historical, religious, etc. Some are related to organisations or churches, to quote: " The National Alliance, World Church of the Creator, Eastern Hammerskins, and other racially conscious White groups... "
- Different genres: pseudo-scientific articles, pamphlets, essays, for example, the "History of the Jewish Assault on the World".

2. Racist people tend to hide the racist nature of their documents and avoid using straightforward statements. Hence, there are no keywords that allow us to identify the racist discourse, since the same keywords may be found in anti-racist discourses.
   - Understatements hide strong meaning behind usual terms. Sentences like "Kill Jews" are seldom found but rather mentions of "The Jewish war against civilization".
   - The meaning of words is inverted (e.g. "How is Genocide being perpetrated on White Americans?");
   - Apparent social discourse associates social problems to ethnic groups, often without even mentioning them (e.g. "Work as tax slaves to support people who love to make Americans pay for their children.");
   - Pseudo-scientific discourse is used to give a rational appearance to the hate speech (e.g. about superiority of the white race, as in "my motivations are not of insult or hatred, but of the deepest love for mankind", followed a few paragraphs later by "Throughout 6,000 years of recorded history, the Black African Negro has invented nothing.").
3. Organized racist people tend to use their own vocabulary or coded languages, which evolves rapidly ("Holocau$t", "Holoco$st", 88 for "Heil Hitler", etc.). Their web sites migrate quite often (several tens of identified pages have disappeared during the first year of the project).
4. There exists a number of anti-racist web sites which tend to share mostly the same words as racist documents. They often quote racist texts to prove their falsity. This may mislead automated detection systems.

Hence the challenge is of a double nature: (1) find out documents the content of which is related to racism and (2) separate racist content from anti-racist content.

Nevertheless, as it was expected from the beginning of the project, the analysis conducted so far have shown that the racist authors are betrayed by their linguistic habits. Of course, the identified linguistic features, which we call *criteria*, are not simply related to the use of certain words, but are rather a set of concordant features involving multiple word combinations in certain distance ranges, frequencies of certain common or less common words, etc.

2.2 Examples of Criteria

From the analysis of large sets of racist, anti-racist and non-racist documents, a number of candidate criteria for identifying racist content have been exhibited. Some of them are listed below and, as one can see, none of them may be used as a "proof" of racism.

Unique racial expressions created or used only by racist people are a strong clue (cannot be used to separate from anti-racist documents), for example "Repulingcunts" instead of "Republicans" or "Rahowa" standing for "Racial Holy War".

Average frequencies of certain or categories of words are not the same in racist documents. These words are not necessarily racist ones but rather:

- common words (like "their" or "white");
- thematic words (for example words that denote fear of the multiplicity of the ethnic out-group like "multiply", "takeover", "teeming", etc.);
- truth claims (words like "certain" or "fact" - as in "it is a fact that");
- hedging words ("almost", "maybe", etc.).

Adjectives are more frequently used in racist discourse which resorts a lot to compound adjectivisation or systematic adjectivisation.

Combined frequencies of certain word pairs are relevant, for example, the combination of "our" with words like "civilisation", "race" or "religion".

Suffixes like "al", "ence", "ism" are good indicators for separating racist and anti-racist documents.

Fonts like gothic fonts, or some images, are typical of racist pages, while they never make a proof of racism.

There are many other features like these ones and most of them have been discovered by a comparative statistical study of several aspects of documents (words, word combinations, word constituents, word categories, etc.).

2.3 Relying on Weak Criteria

As the previous examples show, there are no clear indicators of racism on which one might rely to build a detection system. This is a consequence that there are no word or any other linguistic feature that *only* racist people use. Hence we have to fall back on statistical analysis but, although it did exhibit differences between racist documents and non racist ones, the weakness of the statistical approach is that it does not allow to make assertions about one single document, only about groups or classes of documents.

Two factors influence the global complexity of the system. Firstly, only the convergence of several factors may be a good indication of racism, provided that there are no concomitant indications of anti-racism. Hence the number of criteria (several tens), their individual complexity, their correlations and relative relevance have an influence on the overall complexity. Secondly, the empirical factor has an important role: some criteria that seem conceptually close may have very different results, discriminating power, efficiency or computation speed. Each criterion may be used either for selecting, comparing or eliminating pages, with diverse quality. Some have side effects, like computing some information about the documents that might be useful for the computation of other criteria. Finally, the multiple possible combination of criteria may be more or less precise and efficient. But we do not possess any reliable theory or model to determine in advance the precision or the efficiency of such or such combination with respect to a given information retrieval goal.

Here are the obstacles to the formalisation of rules that might allow for a deterministic way to conduct the global filtering process.

3 Related Work

Kalgren J. and Cutting D. have addressed the issue of textual genre determination. In [5], they laid the groundwork in discriminant statistical analysis within the framework of information retrieval and textual mapping. Their method consists in defining text databases that have already been labelled and in automatically developing discriminating features using such databases so as to sort out new text in its relevant slot. This approach has been refined and implemented into a software program (Easify) that allows sorting out web documents in accordance with a series of parameters, including document genre [6]. Genres in this work are akin to the notion of stylistic variation and are opposed depending on their content. This approach, though appealing, appears to be incomplete. On the one hand, the approach sets out a mapping out of genres on the web, which is well suited to this medium and which uses all of the Web's diversity to carry out discriminant analysis that goes beyond mere text analysis. In this sense, the genres that are listed are uniquely specific to the Web publishing practices that have given rise to them. On the other hand, it leaves out all other generic dimensions of the text that can be published throughout the Web. More specifically, the approach used combines textual and Web analysis by defining heterogeneous categories: the distinction between ”personal pages” and ”commercial” ones is caused by an economic aspect and by who the site provider is; conversely, the difference between ”journalistic material” and ”reports” is rooted in page-content as well as text-genre analysis, while ”other text” appears to be a convenient slot to include a vast majority of pages. As for ”interactive pages”, it seems to label a property that is inherent in any Web publication. More broadly, the very question of discourse is being raised here: are all Web documents to be related to a similar discourse type and the same enunciative situation? Just as printed material can convey different types of discourse, so electronic publishing can be used for different purposes. Hence, the proposed categories though useful and relevant when it comes to labelling a Web page, do not seem to us to be about genres.

B. Kessler, G. Nunberg and H. Schutze have come up with a more text-oriented approach [2]. In order to meet the needs of sorting out heterogeneous Web-related documents, they proposed a theory of genres as bundles of facets. They argued that genres are defined as an a-priori text-independent principle for sorting out text. Thus they refer to genre as any widely recognised class of texts defined by some common communicative purpose or other functional traits, provided the function is connected to some formal cues or commonalities and that the class is extensible. Starting from such a premise, genres will be considered as a bundle of generic facets, a facet being a property that sets a class of texts apart depending on a specific criterion and which can be computationally processed. Their approach brings genuine flexibility in the way such features are defined and identified. The novelty of the approach, indeed, lies in the ability to easily define such features and to combine them in order to define genres. The method has been tested on the Brown Corpus and has shown its quality regarding surface items with a 70% global rate of recognition of genre. While Kessler et al. do not disclose the inner characteristics of each genre identified, they show that an automatic recognition system can pinpoint genres by analysing the combination of different elements. They also show that genres are not irreducible atoms that cannot be analysed, but bundles of features that can be separated and upgraded. However, the method has not so much dealt with genre as with generic fields, which are closer to fields of activity. This calls for two remarks: first, it can thus be better understood why lexical units have played a positive part in analysis, as they prove closely connected to changes in generic fields. Second, the question of the reference corpus within which genres must be compared and opposed must be raised.

By analysing genre, analysis of lexical units and of key words, which is less differentiating in relation to generic field change will be limited and other types of available data will be used. The work of Kessler et al. shows us that such variables can stand for generic categories, which is borne out by the failure of analysis based only on lexical units.

There is little overlap between the work we intend to do and past projects in the field of computer terminology (like WordNet). Indeed, the state of the art in terminological acquisition is based on a domain oriented approach rather than on a text oriented one. Existing projects like Euro Word Net (EWN) make the assumption of a close relationship between the words of the terminological base and the word occurrences in the texts. Experience shows that this assumption does not hold in the sense that text production is not ruled only by domain knowledge but also by discursive criteria that have an effect on the lexicon - including the specialised lexicon. Real vocabulary is not homogeneous. Hence the necessity to have a more text oriented approach, given the unsuitability of the resources provided by domain oriented projects (dictionaries, ontologies). Racist documents have a strong ”moral” dimension, glorifying or deprecating groups of people, which is most often expressed by means of adjectives, of specific relationships between nouns, adjectives and verbs, of particular usage of punctuation, etc. But existing linguistic projects focus almost completely on nouns. Therefore, none of the existing projects' results may be applied directly to solve the problems raised by this project: the necessarily fine-grained analysis of document contents requires a more textual approach, while most existing projects in the field focus on the linguistic and conceptual dimensions [4]. This is well suited to their goals but such an approach fails to engage with the reality of texts.

4 The Multi-agent Model

In this section, we present the multi-agent system and the coordination model that we propose for the agents and we will give the algorithmic behaviours of the agents during the coordination process. Then we will define more formally the concept of association in multi-agent systems and we will compare it with the concepts of coalitions, teams and aggregations.

The complexity and lack of algorithmic model account for the need to use a multi-agent system. In this system the agents will be requested to cooperate in order to combine filtering criteria. The key to this comparison resides in the pairing of a criterion and an agent, the combination of criteria being then solved in a cooperation between agents. But it must be as of now clear that it is not enough to encapsulate the criteria in agents to solve the problem, since the absence of algorithm implies also the inapplicability of those protocols which define the roles of the agents in a static way. One needs a dynamic coordination of the agents.

4.1 The Multi-agent Architecture

We consider herein that an agent is an autonomous entity which interacts with others through protocols. With regard to the filtering of the documents, three kinds of agents were defined.

- The *criteria-agents* C_j encapsulate each a different criterion of evaluation C_j, like those presented in Section 2.2. The service that they are likely to provide consists in evaluating and grading the documents which are presented to them.
- The *document-agents* A_j are associated to the documents brought back by the search engines. Such an agent is associated to each document and its role is to find criteria to evaluate it. The document-agents are thus, a priori, identical between them and their lifespan is limited.
- The *query-agents* Q_i are associated with the queries submitted to the search engines. They are those to which document-agents and criteria-agents must give their final evaluation and, as such, represent the "clients" of the agent-system. In practice, such agents will appear at regular time interval with the goal to update the list of racist sites and will send wide queries to search engines and have their results evaluated by the other agents.

The founding principle of the so-called "pyramidal" cooperation model is to carry out the application of the criteria in several passes (Figure 1). Thus, at a moment t, the evaluation of a document d with respect to a query q is defined as:

$$C(d,t) = \frac{1}{n}\sum_{i=1}^{n} \lambda_i(q,t)C_i(q,d)$$

The parameter $\lambda_i(q,t)$ measures the weight of the i^{th} criterion and will be equal to zero if it is not activated yet. The activation of the criteria is done according to a negotiation model between criteria-agents and document-agents, of which a detailed description is the subject of Section 4.2.

4.2 Dynamic Pyramidal Coordination

4.2.1 Principle

The goal of the document-agents is to select criteria-agents so as to maximise the evaluation of their documents while respecting some constraints related to the usage of computer resources. To a document-agent A_i and a criterion-agent C_j one may associate a partial utility function $U_{ij}(t)$ which measures the interest of A_i to require an evaluation from C_j at time t. A strategy of A_i may hence be evaluated as a global utility function $U_i(t) = \sum_j U_{ij}(t)$.

This process can be performed using a structure called a "dynamic pyramidal coordination structure" of agents that we use both to impose constraints and to enforce synergy between agents. This pyramidal coordination structure is built up of associations of agents whose utility functions match, at least partially, with each other.

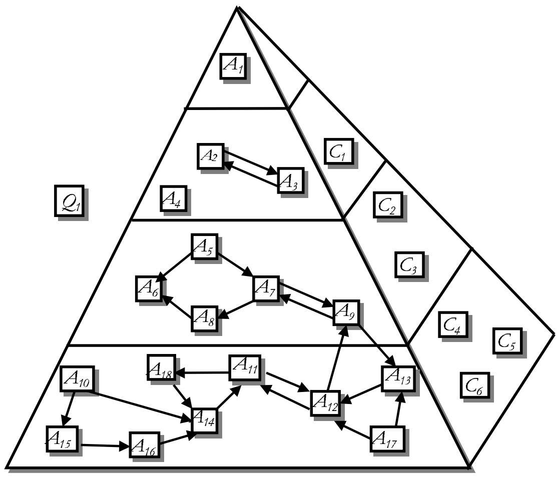


Fig. 1. Dynamic pyramidal coordination structure

Definition 1. *A dynamic pyramidal coordination structure for a set of agents with respective utilities $U_1 \dots U_m$ is a levelled partition whose components $\{L_1 \dots L_p\}$ contain each a set of document-agents $\{A_{k1} \dots A_{kx}\}$, a set $\{C_{k1} \dots C_{kz}\}$ of criteria-agents and a set of associations $\{S_{k1} \dots S_{ky}\}$ of document-agents.*

Each component of the partition is called a level. *Some resources R_k (for example documents analysis execution delay) are allocated at each level of the pyramidal structure. These resources are used by the agents which reached this level. Before reaching a higher level, an agent must collects a set positive evaluations which allow to use the resources allocated to this level. This set of positive evaluations increases with the level.*

An example of a pyramidal coordination structure of four levels with seventeen document agents, six criteria agents and seven associations is shown in Figure 1. On this figure, agents belonging to a same association are connected with arrows.

When a new document is to be evaluated, a new document-agent is created and placed at the lowest level of the pyramid. It then starts collecting evaluations from criteria-agents. If at some moment its utility function has reached a certain threshold, then it is entitled to raise to the next level. At each level, the amount of computing power available to a single agent increases but remains limited so as to limit its possible choice of a criterion. This principle enforces resource optimisation. However, this optimisation should not be inflexible, and this is where the organisation of the agents in associations comes into play.

A first informal definition of associations might be: an *association* is a set of agents whose utilities interact with each other due to some shared features.

In our application, these shared features are the features of the documents handled by the document-agents. Such features might be as simple as the author of the document, or the web site they come from, or some linguistic feature. The document-agent has some knowledge of the features of its document, and this knowledge may increase as a result of the application of some criteria-agents. On entering in a level of the pyramidal structure, a document-agent announces itself as well as the features of its document it already knows about. On this basis, it starts forming associations with already existing agents at that level. It also collects from criteria-agents information about their possibilities, their usage constraints and their requirements in terms of computing resources.

The utility U_i of each agent A_i is now depending only on the actions that A_i chooses. To make this choice the document-agent exploits the established associations by using the knowledge collected by the other agents belonging to the same associations. The document-agent can then decide its own strategy taking all the implications into consideration. The choice of optimal criteria and the insurance they will "pay back" will, of course, depend on the past decisions of the agents of the same associations, whose strategies have already provided feedback. Afterwards, the agent will communicates the results of the application of its strategy to other agents of the associations it belongs to, so that they improve their own strategies. If, for example, some agents are associated because their respective documents have the same author and if the *unique racial expression* test (Section 2.2) fails for one agent, then the other ones will avoid using it.

So, this pyramidal coordination structure limits the use of computer resources by document-agents, preventing them to apply resource intensive criteria while they have not proved that the document they carry is worthwhile. At the same time, this structure helps document-agents to exploit their relations through the associations so as to increase their utility functions. The query-agent takes this decision according to the result of the utility functions that document-agents obtained during their processing.

The algorithm of each document-agent proceeds repeatedly:

1. A_i broadcasts a query for identifying criteria-agents and receives from them the information $\{C_k, t_k : v_k\}$ where v_k is the range of the utility value returned by the criterion-agent and t_k is an estimate of the computing time needed.
2. A_i broadcasts an announce of its presence and receives in exchange proposals for joining existing associations in the form $\{S_x, A_y\}$ where A_y is the document-agent that A_i can contact to join the association S_x. After this step, A_i can join in one or more associations.
3. A_i builds its own maximisation strategy with respect to the criteria-agents which it will contact to analyse its document.
4. A_i performs the maximisation of its utility function U_i according to its strategy which it updates according to the messages that it receives from the documents-agents sharing common associations.
5. A_i distributes the information on the results obtained from the chosen criteria-agents to the document-agents of the associations in which it participates.

Once this procedure is performed by the document-agent in the lowest level of the pyramidal structure, it may or not be entitled to reach a higher level, where this process repeats. Each level L_k of the pyramidal coordination structure has a minimal utility threshold U_k^{min}. To reach this level, a document-agent A_i needs to have $U_i \geq U_k^{min}$.

The messages exchanged in step 4 are the base information needed by agents from the same associations to build their own strategies. Let's consider an association S_x, constituted by documents sharing the same author. If criterion C_k has given the best results for one agent in S_x then the other agents of S_x will tend to use the same criterion C_k to increase their utility.

In a different situation, the agents may coordinate their actions differently. Let's imagine two document-agents $A_{i'}$ and A_i in a same association because they have obtained good results with the criterion C_k. These agents need not to remain in the same association for ever. Indeed, relationships that hold at a certain moment can disappear as agents apply other strategies by choosing another criteria. For instance, if at a later time t', $U_{i'm}(t') = 10$ and $U_{im}(t') = 0$, then an optimal strategy for document-agent A_i would be to leave the association with $A_{i'}$. In this example, the previous confidence between A_i and $A_{i'}$ disappears due to differences in their respective strategies and therefore A_i and $A_{i'}$ do not need to communicate anymore their results.

4.2.2 Advantages of the Coordination Structure

This pyramidal coordination structure exhibits several important properties.

1. It allows to dispose at anytime of a temporary result of the filtering of the documents simply by examining how the corresponding agents are distributed in the pyramid, which reflects the current value of their utility functions.
2. Thanks to associations, it enforces cooperation between agents sharing common properties.
3. Associations reduce the time lost in unnecessary computations since document-agents avoid choosing criteria having proved to be inefficient on similar documents.
4. The coordination of the document-agents is managed thanks to the organisation of the dynamic pyramidal structure in several levels. These levels set dynamically the priority of the document-agents and make them evolve with their utility U_i.
5. This structure is flexible. It can be adapted incrementally and easily since new criteria may be introduced in the system simply by adding new criteria-agents.
6. There is no implicit central algorithm, no implicitly shared behaviour, no central ruling agent and no central planning: the agent behaviour and usage of the resources is explicitly constrained by the coordination model.

4.2.3 What is an Association?

In this section, we introduce a more precise definition of what we mean by an "association" and present the different features of the agents in an association.

An association of agents is a group of agents, but not characterised by a group rationality. The agents have of course their individual rationality but do not receive any direct payoff as a result of the group's performance. This is particularly true since there is no collective task. Such characteristics are more relevant for coalitions [1, 8, 9] and teams [10].

As in congregations, each agent has its own utility function that it maximises. To do so, it takes in consideration the benefits it has of joining associations. Agents join only associations with which they share features. They are free to leave these associations if during their processing they find that their utility does not evolve within these associations. They join associations in order to satisfy their needs in better conditions.

Even if the concept of association presents similarities with that of congregation defined in [3] such as those described above, nevertheless they remain two different concepts.

1. [3] defines the concept of congregation as:

   > "A group of agents that has come together for some mutually beneficial purpose, exchange of goods, services or informal accomplishing of tasks or an aggregation in order to accomplish goals which could not be met separately."

   As an example they propose that of "clubs" in human society. Thus they associate to the concept of congregations a cost for joining them. Agents should pay a fee to join the congregation and to have access to its services. Such a fee can be monetary for some congregations like clubs. On the contrary, in associations, agents do not need to pay any fee.
2. There is no substantial investment to create an association, while this is needed in congregations.
3. Contrary to a congregation, an agent joining an association does not seek a particular partner with whom it will interact directly but rather a group of agents providing knowledge.
4. An association is initially characterised by the agent that took the initiative to create it. Its characteristics do not change with time. For example, in our application, an association might be created to group agents that operate on documents written by one and a same author.
5. The integration of an agent into an association is done only on the basis of the objective features associated to the association and not on the basis of the agents already belonging to it as it would be the case in congregations [3].
6. An agent does not need to know all members of an association it belongs to. One "entry point" is enough to share its results with other agents and, of course, to take benefit of other agent's "know-how".
7. The belonging of an agent to an association is not necessarily a long-term contract. In the case of our application, it might be just the duration of a user's query.
8. Associations do not have any kind of central control.

Formally, we define an association as follows.

Definition 2. *An association S_i is represented by each agent A_j as a triple $\{A_i, F_i, K_i\}$, where:*

- *A_i is the "entry point" of agent A_j in the association the agent which introduced A_j to the association.*
- *F_i is the list of the features the documents represented by the document-agents share in the association. This list is initially incomplete as it serves as a basis for integrating new agent in this association.*
- *K_i is the knowledge acquired by agent A_j from the association S_i. This knowledge is in a form of rules that the association gathered from its different participants.*

The knowledge pertaining to an association is updated as members communicate to each other their observations on the operations they perform. In the case of our application, this knowledge is related to the results of document evaluation.

Definition 3. *A rule R_k of an association S_i is represented as a triple $\{C_j, Dom_k, p_k\}$, where:*

- *C_j is the criterion concerned by the rule R_k.*
- *Dom_k is the application domain of the rule R_k.*
- *p_k is the probability that R_k has a positive result.*

For instance, analysing the pattern "our race" on a document from a given author following the vectorial analysis criterion produces 80% of positive results.

We introduced association in the theoretical model in order to strengthen the criteria selection by document-agents. As there is no competition between these associations, their construction and evolution remain simple. In order to create a new association, a document-agent first analyses the set of associations already created in the same pyramidal structure. The information needed to perform this analysis is gathered from the query-agent which is in charge of initially linking agents in the structure. The document-agent then represents each association as a triple $\{A_i, F_i, K_i\}$. The following step consists, for the document-agent, in looking for common properties between these associations and its document. The analysis process leads to an array holding, for each association, the common properties, those which are different and finally the rest of properties upon which it can make no decision yet (un-deterministic). Using the vectors and following a given strategy, the document-agent may select some associations which it will join. Belonging to an association is of low cost for document-agent since the only common processing are messages exchanges and handling. However, when the document-agent gets busy, it does not have to handle messages from other agents of the association since these messages are most of the time informative. In addition joining several associations doesn't downgrade the value of its actions since an agent is not required to send messages. In case no existing association fits the document-agent and the information it hands on its own document doesn't allow it to make a decision, the document-agent resorts to criteria-agents in order to raise up more information about the document. While discovering more properties for its document, if it cannot still join any association, the document-agent will initiate the creation of a new association.

Each association is made of agents some of which interact with others. Document-agents individually interact with criteria-agents which apply linguistic criteria to the concerned documents. Applying criteria to documents produces more knowledge about the documents and the applied criteria. This knowledge may help a document-agent to advice another document-agent of the same association to apply a given criterion for its document. Concretely, any knowledge a document-agent deduces from a criterion application is sent to agents of all the associations this document-agent belongs to. Hence, the global knowledge K_i about the association S_i is extended with some local information about the association members as well as the mechanism to manage this information. As soon as a new knowledge is provided it is merged in K_i and new deductions are forwarded (by a document-agent) to other agents.

An association may also alter its features if some document-agents decide to drop focus from some properties while gaining focus on others. Thus some new features may be added to improve the association definition, provided these features are demanded by some of the members or have been extracted from most of the documents represented by the association members. The new features are incrementally raised up by the criteria-agents applications to documents. Conversely, an old feature is dropped if it is no more representative of all the association members or it becomes weakly representative of these agents.

5 Implementation

The implementation of the system is based on the three-tier architecture presented in Figure 2 and which was designed so as to facilitate the integration of the heterogeneous components involved in the system: linguistic modules, multi-agent system, terminological and document database.

1. The first tier is composed of client systems, connected through the HTTP protocol. Typical clients include protection software or browsers which use the PICS *label bureau* [7] protocol to filter out undesired pages.
2. The second tier is the *virtual server* represented as the main central rectangle of Figure 2. It is composed of a number of abstract services aggregated around a CORBA bus. They include, in particular, the software modules that implement the various criteria like word statistics, collocation statistics, etc.
3. The third tier contains lower level services like databases, linguistic software and search engines.

At the agent level, the coordination protocol is realised as per-role finite-state machines implemented in java and that agents may reuse at will, provided they implement the necessary methods for negotiation, etc. From the interaction point of view, KQML has been selected as communication language. We introduced the notion of a *class-agent*, a kind of specialised facilitator the role of which is to instantiate new agents

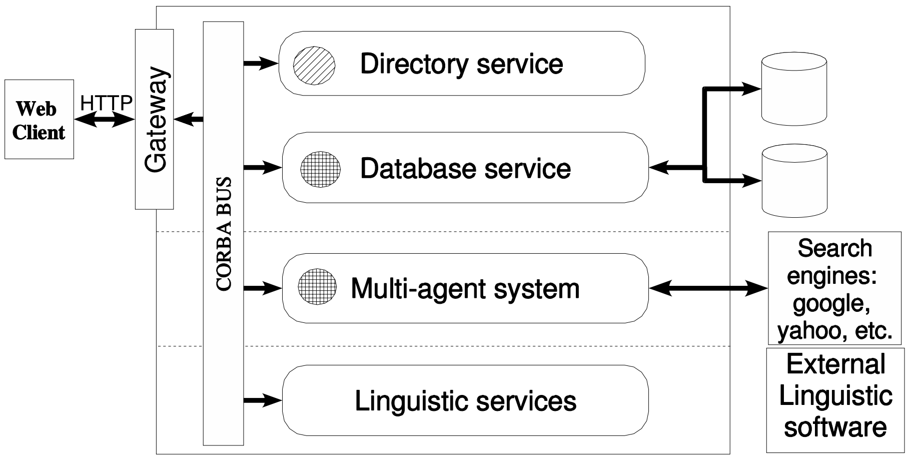


Fig. 2. Software architecture

of a particular class, to keep track of their existence and to broadcast them relevant messages.

We applied the system to the corpus and some results are consigned in Table 3, Table 2 and Table 1.

For each language and for each of the three subsets of the evaluation corpus (racist, anti-racist, neutral), the results of the evaluation are listed. The columns of the tables are:

category the category of documents being evaluated;
Size the number of documents in that category;
Racist the number of documents classified as racist;
Anti the number if document classified as anti-racist;
Neutral the number of documents classified as neutral (i.e. considered as neither racist nor anti-racist);
Recall the number of documents correctly classified.

For the French language (Table 1), 74% of racist documents submitted to the system were correctly classified as racist. Four (4) non-racist documents were wrongly considered as racist. For the English language (Table 2), 66% of racist documents were correctly classified as such. Only three (3) documents were wrongly considered as racist. And finally, for the German language (Table 3), 62% of racist documents were correctly classified as such. Only four (4) documents were wrongly considered as racist.

The evaluation of this system was performed using the Web interface of the system. It is the same interface as the one that was used by the linguists to conduct their exper-

Table 1. Figures for the French language

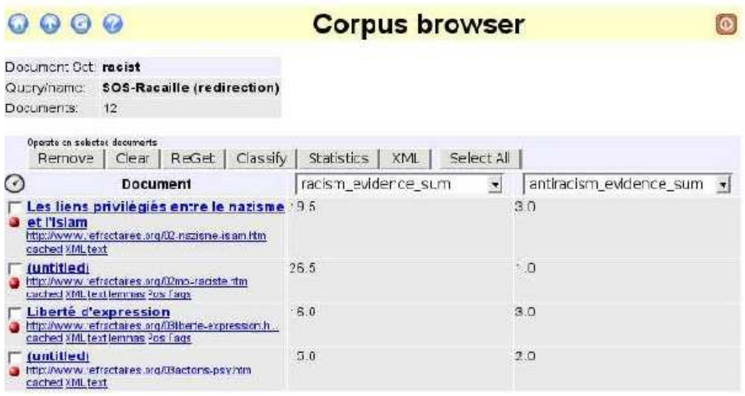
Category	Size	Racist	Anti	Neutral	Recall(%)
Racist	154	114	1	39	74
Anti-Racist	70	2	14	54	20
Neutral	42	2	2	38	90

Table 2. Figures for the English language

Category	Size	Racist	Anti	Neutral	Recall(%)
Racist	146	96	0	50	66
Anti-Racist	62	2	5	55	8
Neutral	85	1	5	79	93

Table 3. Figures for the German language

Category	Size	Racist	Anti	Neutral	Recall(%)
Racist	122	76	2	44	62
Anti-Racist	65	1	6	57	9
Neutral	66	3	1	62	94



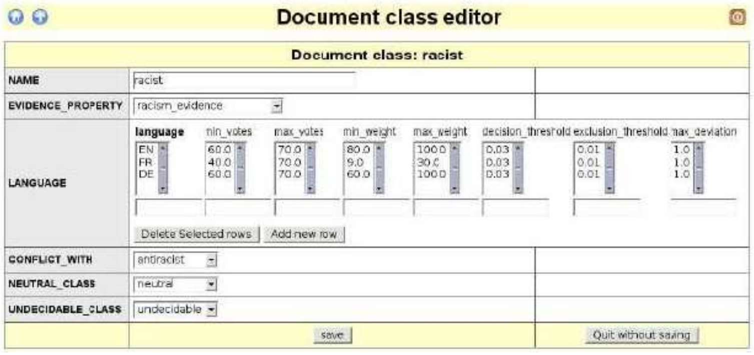
Corpus browser

Document Set: **racist**
Query/name: **SOS-Racaille (redirection)**
Documents: 12

Operate on selected documents
Remove | Clear | ReGet | Classify | Statistics | XML | Select All

Document	racism_evidence_sum	antiracism_evidence_sum
Les liens privilégiés entre le nazisme et l'Islam http://www.lefractares.org/02-nazisme-islam.htm cached XML text	19.5	3.0
(untitled) http://www.lefractares.org/02mo-raciste.htm cached XML text lemmas Pos Tags	26.5	1.0
Liberté d'expression http://www.lefractares.org/03liberte-expression.h... cached XML text lemmas Pos Tags	16.0	3.0
(untitled) http://www.lefractares.org/03actons-psy.htm cached XML text	5.0	2.0

(a) Snapshot of the tuning panel



Document class editor

Document class: racist

NAME	racist
EVIDENCE_PROPERTY	racism_evidence
LANGUAGE	see below
CONFLICT_WITH	antiracist
NEUTRAL_CLASS	neutral
UNDECIDABLE_CLASS	undecidable

language	min_votes	max_votes	min_weight	max_weight	decision_threshold	exclusion_threshold	max_deviation
EN	60.0	70.0	80.0	100.0	0.03	0.01	1.0
FR	40.0	70.0	9.0	30.0	0.03	0.01	1.0
DE	60.0	70.0	60.0	100.0	0.03	0.01	1.0

Delete Selected rows | Add new row

save | Quit without saving

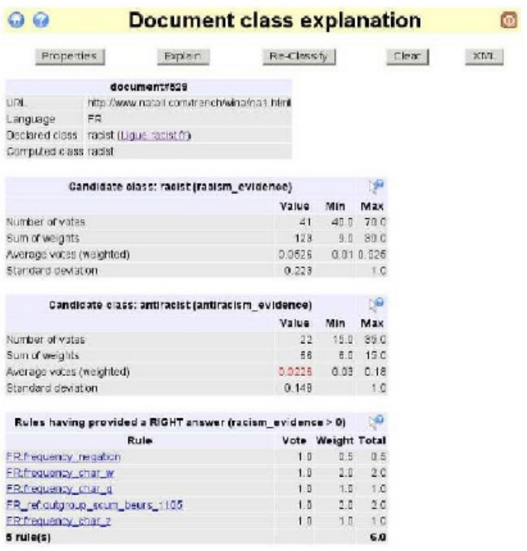
(b) Snapshot of the validation interface

Fig. 3. Evaluation and Tuning interfaces

iments. The Web interface allows to upload complete lists of URLs (in XML format) and to let the underlying system try to classify the documents.

Figure 3(b) shows the validation interface. In this snapshot, the result of the evaluation of a set of documents is displayed. The bullet on the left indicates the category in which the document has been placed. These categories are: *racist*, *anti-racist*, *neutral* and *not analysed yet*. The two rightmost columns allow to display some attributes of the document as they were computed by the system during the analysis phase. In the figure, the sums of the votes in favour of racism and anti-racism are displayed. All document properties may be inspected, including linguistic properties. For the tuning of the reactivity of the system, the users may use the control panel displayed in Figure 3(a). There is one such panel for each category: racist and anti-racist. It allows mainly to modify variables which decide decision making during documents analysis.

After the tuning phase (performed by linguists on their own corpus), it was possible to apply the system to the evaluation corpus. The interface allowed to enter lists of URLs and then to investigate why a given page was classified in some given category.



document#529	
URL	http://www.natall.com/french/wina/na1.html
Language	FR
Declared class	racist (Ligue racist fr)
Computed class	racist

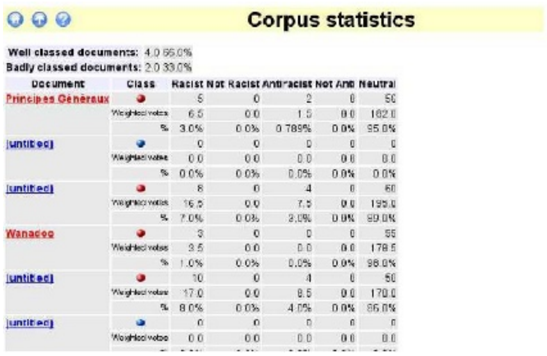
Candidate class: racist (racism_evidence)	Value	Min	Max
Number of votes	41	40.0	70.0
Sum of weights	128	9.0	80.0
Average votes (weighted)	0.0526	0.01	0.026
Standard deviation	0.223		1.0

Candidate class: antiracist (antiracism_evidence)	Value	Min	Max
Number of votes	22	15.0	35.0
Sum of weights	56	6.0	15.0
Average votes (weighted)	0.0226	0.03	0.18
Standard deviation	0.148		1.0

Rules having provided a RIGHT answer (racism_evidence > 0)			
Rule	Vote	Weight	Total
FR:frequency_negation	1.0	0.5	0.5
FR:frequency_char_w	1.0	2.0	2.0
FR:frequency_char_q	1.0	1.0	1.0
FR_ref:outgroup_scum_beurs_1105	1.0	2.0	2.0
FR:frequency_char_z	1.0	1.0	1.0
5 rule(s)			6.0

Fig. 4. Snapshot of the classification explanation window

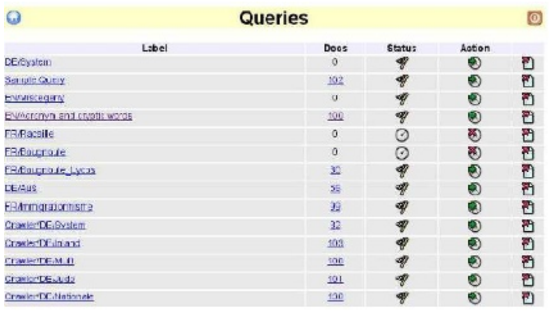
Figure 4 shows the explanation window which provides details about why one single document was placed in a given category. It first displays a summary of the votes related to the `racism_evidence` and `antiracism_evidence` properties of a document. For example, the document $www.natall.com/french/wina/na1.html$ obtained 41 votes for `racism_evidence` and 22 votes for `antiracism_evidence`. Note that each of these properties has a min votes and max votes values. Then it lists all rules that have provided an opinion about the document grouping them in correct rules and wrong rules. The five rules listed at the end of the figure are the ones which gave a good opinion.



Well classed documents: 4.0 66.0%
Badly classed documents: 2.0 33.0%

Document	Class	Racist	Not Racist	Antiracist	Not Anti	Neutral
Principes Généraux	🔴	5	0	2	0	50
	Weighted votes	6.5	0.0	1.5	0.0	182.0
	%	3.0%	0.0%	0.789%	0.0%	95.0%
[untitled]	🔵	0	0	0	0	0
	Weighted votes	0.0	0.0	0.0	0.0	0.0
	%	0.0%	0.0%	0.0%	0.0%	0.0%
[untitled]	🔴	8	0	4	0	60
	Weighted votes	16.5	0.0	7.5	0.0	195.0
	%	7.0%	0.0%	3.0%	0.0%	89.0%
Wanadoo	🔴	3	0	0	0	55
	Weighted votes	3.5	0.0	0.0	0.0	178.5
	%	1.0%	0.0%	0.0%	0.0%	98.0%
[untitled]	🔴	10	0	4	0	50
	Weighted votes	17.0	0.0	8.5	0.0	170.0
	%	8.0%	0.0%	4.0%	0.0%	86.0%
[untitled]	🔵	0	0	0	0	0
	Weighted votes	0.0	0.0	0.0	0.0	0.0

Fig. 5. Corpus statistics



Queries

Label	Docs	Status	Action	
DE/System	0	▶	●	▢
Sample Query	102	▶	●	▢
EN/Miscegeny	0	▶	●	▢
EN/Acronym and cryptic words	100	▶	●	▢
FR/Racaille	0	⏲	●	▢
FR/Bougnoule	0	⏲	●	▢
FR/Bougnoule_Lycos	30	▶	●	▢
DE/Aus	56	▶	●	▢
FR/Immigrationnisme	39	▶	●	▢
Crawler/DE/System	32	▶	●	▢
Crawler/DE/Inland	103	▶	●	▢
Crawler/DE/Multi	100	▶	●	▢
Crawler/DE/Jude	101	▶	●	▢
Crawler/DE/Nationale	100	▶	●	▢

Fig. 6. The list of running queries

Figure 5 shows the statistics window that provides a statistical view on the classification of a set of documents. This is particularly useful when the set gathers documents of a same category.

Figure 6 shows the list of queries with their current execution status. Apart testing specific queries, this interface is rather meant to monitor the real system operations (control queries executions, suspend and resume a query execution). One may view how many documents were retrieved by a query and inspect them.

6 Conclusion

In this article, we have tackled the issue of detecting the racist nature of documents. To our knowledge, this issue has never been addressed before, because (1) of its inherent complexity, (2) of the involvement of several scientific domains (computer science, linguistics, mathematics) and (3) of the "political" implications of the subject. Another step forward consists in putting together two important and complementary approaches, namely computer linguistics and multi-agent systems. From the latter point of view, we proposed a new coordination model, based on a pyramidal structure and agent associations – a new concept that we presented and compared to similar notions.

The results we gave proved that our system does not make ethically un- acceptable mistakes but it leaves out too much of racist pages. The precision of the system is very high (97%) in all languages: very few documents are wrongly classified as racist. This was the key figure that the evaluators monitored. The system has a bias towards "under reaction", which is preferable than misclassification. Indeed, the goal of the system is not scientific but applied to the protection of Internet users. Hence, the different categories are not equivalent with respect to the importance of precision and recall. In particular, the rule tuning phase focused on lowering the weight of those rules that tend to overreact to racist (for classifying them as anti-racist) documents, while raising the weight of the rules that overreact on anti-racist documents (for classifying them as anti-racist). Rules that wrongly classify documents as anti-racist are less an issue than rules that do the opposite, since it is not acceptable that anti-racist documents are blocked by filtering software. The drawback of this choice is that the recall is rather low (average

67%). This evaluation has also important consequences with respect to the applicability of the system in an industrial context.

References

[1] S. Aknine and P. Caillou. Agreements without disagreements: A coalition formation method. In *ECAI, European Conference on Artifitial Intelligence*, pages 3–7, 2004.

[2] Kessler B., Nunberg G., and Schtze H. Automatic detection of text genre. Technical report, Palo Alto Research Center, 1997.

[3] C.H. Brooks and E.H. Durfee. Congregation formation in multiagent systems. *Journal of Autonomous Agents and Multi-agent Systems*, 2001.

[4] Princip Consorsium. Final report of princip project. Technical report, LIP6, 2004.

[5] J. Kalgren and D. Cutting. Recognizing text genres with simple metrics using discriminant analysis. In *COLING94*, 1994.

[6] J. Kalgren and D. Cutting. Genres defined for a purpose, fast clustering, and an iterative information retrieval interface. In *Eighth DELOS Workshop on User Interfaces in Digital Libraries*, 1998.

[7] Jim Miller. Pics label distribution, label syntax and communication protocols. W3C Recommendation REC-PICS-labels-961031, 1996.

[8] T.W. Sandholm and V.R. Lesser. Coalitions among computationally bounded agents. *AI*, 94:99–137, 1997.

[9] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, May 1998.

[10] M. Tambe. Towards flexible teamwork. *Journal Of AI Research*, 7:83–124, 1997.

Author Index

Aknine, Samir, 306
Anand, Sarabjot Singh, 1, 272

Berendt, Bettina, 69
Burke, Robin, 133

Chan, Keith C.C., 169

Dogac, Asuman, 289

Eirinaki, Magdalini, 272

Greiner, Russ, 241

Häubl, Gerald, 241
Hay, Birgit, 187

Keenoy, Kevin, 201
Kritikopoulos, Apostolos, 229

Lam, Chuck P., 153
Levene, Mark, 201
Lorenzi, Fabiana, 89

McCarthy, Kevin, 255
McGinty, Lorraine, 114
Miller, Craig S., 37
Mobasher, Bamshad, 1

Price, Bob, 241

Quenum, Ghislain, 306

Ramakrishnan, Naren, 53
Reilly, James, 255
Ricci, Francesco, 89

Sideri, Martha, 229
Slodzian, Aurélien, 306
Smyth, Barry, 114, 255

Tang, Tiffany Ya, 169
Teltzrow, Maximilian, 69
Toroslu, I. Hakki, 289
Tumer, Arif, 289

Vanhoof, Koen, 187
Vlachakis, Joannis, 272

Wets, Geert, 187
Winoto, Pinata, 169

Zhu, Tingshao, 241