Combining Collaborative Filtering and Knowledge-Based Approaches for Better Recommendation Systems

Thomas Tran
School of Information Technology and Engineering
University of Ottawa
Ottawa, Ontario, Canada K1N 6N5
ttran@site.uottawa.ca

Abstract

In electronic commerce applications, prospective buyers may be interested in receiving recommendations to assist with their purchasing decisions. Previous research has described two main models for automated recommender systems: collaborative filtering and knowledge-based approaches. In this paper, we present an architecture for designing a hybrid recommender system that combines these two approaches. We then discuss how such a recommender system can switch between the two methods, depending on the current support for providing good recommendations from the behavior of other users, required for the collaborative filtering option. We also comment on how the overall design is useful to support recommendations for a variety of product areas and present some directions for future work.

1 Overview

One important task in the application area of business-to-consumer e-commerce is that of providing recommendations to potential shoppers. In an environment where there is a wide choice for the prospective buyers, an automated system which serves to present a more narrow selection for the buyer would be desirable.

Recommender systems are systems which provide recommendations to potential buyers. Two widely used techniques for building recommender systems to date are collaborative filtering and knowledge-based approaches. Collaborative filtering is a real-time personalization technique that leverages similarities between people to make recommendations. In other words, a collaborative filtering recommender system assumes that human preferences are correlated; thus, it predicts preferences and makes recommendations to one user based on the preferences of a group of users. In contrast, a knowledge-based recommender system exploits its knowledge base of the product domain to generate recommendations to a user, by reasoning about what products meet the user's requirements.

In this paper, we analyze the advantages and shortcomings of both techniques and present an architecture for a hybrid recommender system that integrates the two approaches. Such a system will inherit all the strengths from a collaborative filtering recommender system, but will be able to avoid its weaknesses.

Although there have been some proposals for designing systems which make use of both the knowledge-based approach and collaborative filtering [4], collaborative filtering is used more in a post-processing stage, so that the knowledge-based design predominates. In this paper, we outline some specifications for changing between the collaborative filtering and the knowledge-based styles of recommendation, within a single system.

This design strategy will be useful for electronic commerce applications where the number of buyers and the make-up of the community of buyers dictates whether collaborative filtering will be effective or not.

2 Background

In this section, we introduce the collaborative filtering and knowledge-based approaches in building recommender systems. We discuss the strengths and weaknesses of each approach as the motivation for the design of a hybrid architecture that combines the two approaches.

2.1 Collaborative Filtering Approach

The key idea of the collaborative filtering approach is that a user will prefer those items that like-minded people prefer. A collaborative filtering recommender system (CFRS), therefore, makes prediction for a user based on the similarity between the interest profile of that user and those of other users [3, 7, 14].

Suppose we have a database of user ratings for items, where users indicate their interest in an item using a numerical scale. Then, it is possible to define similarity scores between two user profiles. A variety of similarity metrics has been proposed. Items with high-predicted ratings will be recommended to the user. The following formula is used by [14]:

$$U_x = \bar{U} + \frac{\sum_{J \in \text{ Raters of } x} (J_x - \bar{J}) R_{UJ}}{\sum_{J \in \text{ Raters of } x} |R_{UJ}|} \quad (1)$$

where U_x is the rating to be predicted for user U on item x, $\bar{U}$ is the mean of the ratings of user U, J_x is the rating of some user J on item x, $\bar{J}$ is the mean of the ratings of user J, R_{UJ} is the correlation (similarity score) between user U and user J, and the two summations are performed on the set of all users J, who have rated item x.

An example of a CFRS is the Instant Recommendations at Amazon.com. This system can recommend books, CDs, and other products. It works by first building an interest profile for a user based on the user's ratings on items and the items that the user has purchased in the past. The system then generates recommendations to a user based on the similarity of the user's interest profile and the other users' interest profiles.

It should be obvious that CFRSs have the following advantages and shortcomings:

Advantages:

- They can make personalized recommendations.
- They are able to identify appropriate items to users.
- Their prediction quality improves over time as their databases of user preferences get larger and larger.

Shortcomings:

- A CFRS must be initialized with a large database of users' preferences in order to make useful recommendations [17]. Moreover, the prediction accuracy of the system is very sensitive to the number of items that have been rated [16].
- As a new user starts off with nothing in her profile, a training period is required to train the profile before it can accurately reflects the user's preferences. This problem is known as the "cold start problem" [15].
- When a new item appears in the database, there is no way that it can be recommended to a user until more information is obtained (via another user rating it or specifying which other items it is similar to). This problem is referred to as the "early-rater problem" [11].
- A CFRS may not behave properly when a user's interests change, since it makes recommendations based on the past interests of that user. For example, a book shopper, who usually purchased computer books in the past, may find a CFRS recommendations not very helpful when she is seeking out books for her children.

2.2 Knowledge-Based Approach

As the name suggests, a knowledge-based recommender system (KBRS) generates recommendations to a user by consulting its knowledge base of the product domain, and then reasoning what items will best satisfy the user's requirements [4].

An example of a KBRS is the PersonalLogic system [1]. This system helps users make decisions on variety of products, ranging from cars to computers, from pets to family activities, and from careers to graduate schools. It works by gathering the users' requirements on a particular product, and consulting its knowledge base to find the items that best meet the users' requirements. Thus, a car-shopper may need to provide answers to such questions as what type, size, and features she prefers, what price she can afford, whether luxury or economy (lower cost, better fuel etc.) is more important to her. The system then searches its knowledge base for cars that best satisfy these requirements.

KBRSs have several advantages:

- A KBRS does not need to be initialized with a database of user preferences since its recommendations do not depend on the user ratings of items.
- Since a KBRS does not make recommendations based on users' interest profiles (with their ratings on items), it does not suffer from the "cold start" and the "early-rater" problems.
- A KBRS can adjust its recommendations quickly as a user's interests change because its recommendations are independent of the user's preferences in the past.
- A KBRS can make good recommendations for those products that lend themselves to the knowledge-based approach by nature. Examples of such products are cars, houses, careers, grad schools etc., whose features (characteristics) are obviously of great importance to users. Since a KBRS can suggest to a user those items that best meet the user's requirements, it is an ideal recommender system in this case.

KBRSs, nevertheless, have a main disadvantage:

- To make good recommendations, a KBRS must understand the product domain well. It must have knowledge of important features of the product, and be able to access the knowledge base where these important features are stored in an inferable way. Thus, a KBRS requires knowledge engineering with all of its attendant difficulties.

In the next section we introduce an architecture of a hybrid recommender system that integrates the CFRS and the KBRS approaches. We believe that the proposed architecture will inherit all the advantages of a CFRS, but will not suffer from its shortcomings. Moreover, the combination of the two techniques will make our architecture suitable to products that are collaborative-filtering oriented (e.g., books, CDs, movies etc., as empirically showed by the above-mentioned existing systems), as well as those products that lend themselves to the knowledge-based approach (e.g., houses, cars etc.).

3 The Proposed Architecture

Figure 1 below shows an architecture for a hybrid recommender system that combines the collaborative filtering and knowledge-based approaches.

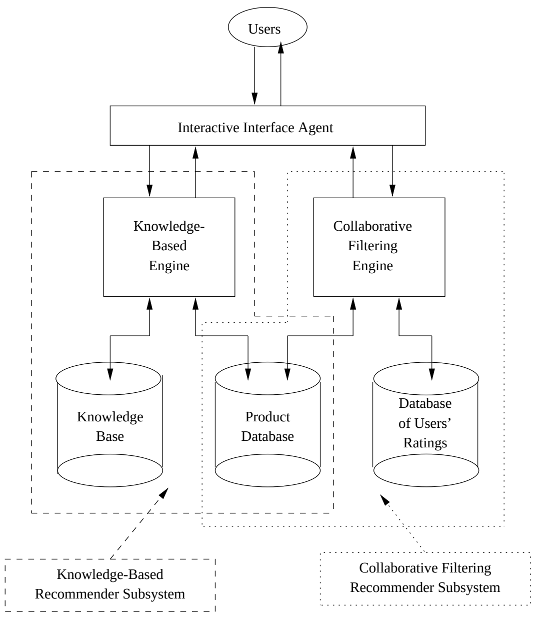

Users
Interactive Interface Agent
Knowledge-Based Engine
Collaborative Filtering Engine
Knowledge Base
Product Database
Database of Users' Ratings
Knowledge-Based Recommender Subsystem
Collaborative Filtering Recommender Subsystem

Figure 1. Architecture for Integrating Knowledge-Based and Collaborative Filtering Approaches

Our architecture consists of the following major components:

- The Interactive Interface Agent.
- The Knowledge-Based Engine.
- The Knowledge Base of the product domain.
- The Collaborative Filtering Engine.
- The Database of Users' Ratings for Items.
- The Product Database.

3.1 The Interactive Interface Agent (IIA)

The IIA plays the role of the control unit in the system. It acts as an intermediary between the user and the two recommender subsystems. The main job of the IIA is to select the appropriate subsystem for service, to coordinate the operations of the two subsystems, and to interact with users in uncertain situations.

3.1.1 Selection of the Subsystems

As we mentioned in the discussion of CFRSs above, a CFRS may not be helpful to a particular user until a large number of users, whose interest profiles are known, and a sufficient number of rated items have been collected into the database. In our architecture, the IIA keeps track of these two variables. When the system is executed, the IIA compares these variables with their corresponding thresholds. The threshold values may be defined through experiments, depending on the type of products and businesses. If either of the two variables is less than its corresponding threshold, the IIA selects the knowledge-based recommender subsystem (KBRSS) for service; otherwise, the collaborative filtering recommender subsystem (CFRSS) will be used.

When the KBRSS is selected, the IIA first guides the user through the log-in process. If this is an existing user, the IIA simply sends the user's name and password to the Collaborative Filtering Engine, which opens the user's interest profile in the Database of Users' Ratings for modification. If this is a new user, the IIA helps the user to create the user name and password, and requests the user to rate some representative items. It then transfers the user's information to the Collaborative Filtering Engine, which creates an interest profile for the user[1].

Since the KBRSS is being used, the IIA solicits product requirement information from the user and transfers it to the Knowledge-Based Engine. The Knowledge-Based Engine consults its Knowledge Base to generate recommendations for the user. The IIA receives these recommendations, formats them, and presents them to the user. To facilitate the

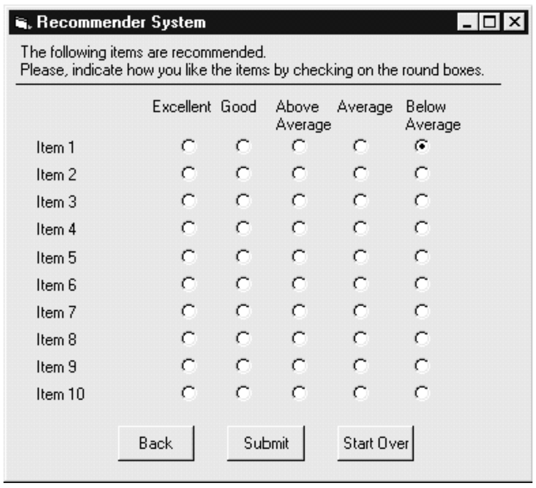
[1]The system wants to take the opportunity of interacting with the user (during the operation of the KBRSS) to update the user's profile of interests, or to create a profile of interests for a new user. The user's profile of interests will be used later on if the CFRSS is selected.

interaction with the user, the system should use a graphical user interface, in which the outputs presented to the user can take the forms of listings, tables, or charts, depending on the nature and the information conveyed in the outputs.

During the operation of the KBRSS, the IIA collects the user's ratings on the items via interaction with the user. These ratings are sent by the IIA to the Collaborative Filtering Engine, which then stores them in the user's interest profile (in the Database of Users' Ratings). The process of getting users' feedback can be done in a number of ways. One way is to request the user's ratings on the recommended items using a rating set such as

Rating Set = {Excellent, Good, Above Average, Average, Below Average}.

The ratings can be mapped into numerical values (by the IIA) to facilitate the calculation of the user's predicted ratings as illustrated by equation (1) in the Background section. These values are then stored in that user's interest profile in the Database of Users' Ratings. A sample system output for getting users' feedback may look like Figure 2 below.



The following items are recommended.
Please, indicate how you like the items by checking on the round boxes.

	Excellent	Good	Above Average	Average	Below Average
Item 1	○	○	○	○	●
Item 2	○	○	○	○	○
Item 3	○	○	○	○	○
Item 4	○	○	○	○	○
Item 5	○	○	○	○	○
Item 6	○	○	○	○	○
Item 7	○	○	○	○	○
Item 8	○	○	○	○	○
Item 9	○	○	○	○	○
Item 10	○	○	○	○	○

Figure 2. Sample Form for Getting Users' Feedback

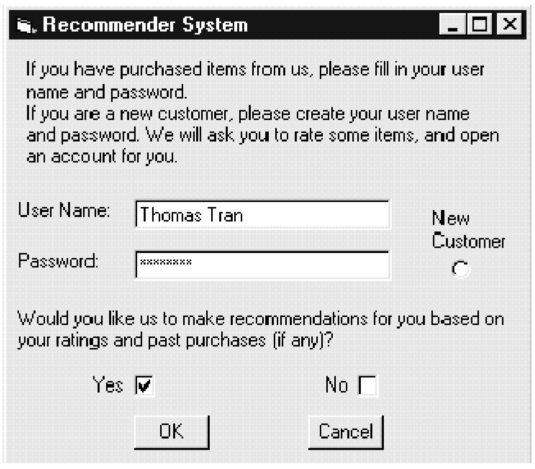
When the number of users, whose interest profiles are known, and the number of rated items are both greater than or equal to their corresponding thresholds, the CFRSS is selected for service on behalf of the system.

As with the case where the KBRSS is selected, the IIA first performs the log-in process. If a user has already had an account with the system, the IIA simply sends the user's name and password to the Collaborative Filtering Engine. The Collaborative Filtering Engine then generates recommendations for the user based on the similarity between the user's interest profile (that the system has already had in its Database of Users' Ratings) and the other users' interest profiles. Items recommended by the Collaborative Filtering Engine will be formatted and presented to the user by the IIA.

If the user is a new user, the IIA guides the user through the process of creating user's name and password, and requests the user to rate some representative items. It then sends the user's information to the Collaborative Filtering Engine, which creates an interest profile for the user. From this point, the operation of the system is the same as the case where the user has already had an account with the system.

During the CFRSS operation, the IIA still collects the user's ratings on items via interaction with the user. These ratings are entered into the user's interest profile as a routine of refining the user's model viewed by the system. This accounts for the fact that the prediction quality of a CFRS improves over time.

There are times when a user does not want the system to make recommendations based on her interest profile. A book-shopper, who usually purchased computer books in the past, certainly does not like the system to make suggestions using her purchasing history when she is seeking out fairy books for her children. To address this situation, the IIA may simply ask if a user likes the system to make recommendations based on her interest profile. This may be done during the log-in process via a graphical form similar to figure 3 below.



If you have purchased items from us, please fill in your user name and password.
If you are a new customer, please create your user name and password. We will ask you to rate some items, and open an account for you.

User Name: Thomas Tran

Password: ********

New Customer ○

Would you like us to make recommendations for you based on your ratings and past purchases (if any)?

[✓ Yes] No

Figure 3. Sample Log-in Form

If the user checks "Yes" as in Figure 3, and if the CFRSS is eligible for service, it will be selected. If the user checks "No", the KBRSS will be selected; however, the IIA does not need to modify the user's interest profile in this case (since the user is not purchasing items for herself).

3.1.2 Coordination of the Subsystems

We can tune our architecture such that the system will always make the best recommendations it can to users. We recall formula (1) that an item x will be recommended to a user U if its predicted rating U_x is greater than the predicted ratings on other items. Hence, we can define that an item x is a good item for recommendation if

$$U_x \geq k\bar{U} \quad (2)$$

where k is a pre-defined satisfaction coefficient, and $\bar{U}$ is the average rating of user U.

We can now make a rule that the IIA will present an item x recommended by the CFRSS to a user only if x is a good item. If no items recommended by the CFRSS are good items, then the IIA will switch the user to the KBRSS for service. Obviously, such a rule is in favor of the KBRSS at this moment, since it assumes that the KBRSS may be able to make "above average" recommendations.

Of course, the IIA can check if its assumption is correct by looking at the user's ratings on the items suggested by the KBRSS. If the user's average rating on the items suggested by the KBRSS is higher than that on the items suggested by the CFRSS, the IIA can go on using the KBRSS. Otherwise, the IIA will adjust the satisfaction coefficient to a lower value so that (2) holds for the items previously recommended by the CFRSS. The IIA then switches the user back to the CFRSS and presents these items to the user.

The following numeric example will illustrate the above idea. Suppose users' ratings are in the range $[0, 1]$, initially $k = 1.2$, $\bar{U} = 0.5$, and the CFRSS sends to the IIA the top five predicted ratings (for five different items), namely $0.59, 0.58, 0.57, 0.56$, and 0.55. Since $k\bar{U} = 0.6$, according to (2), none of the five items is eligible to be presented to the user. Thus, the IIA switches the user to the KBRSS for service. If the average rating of the user on the items suggested by KBRSS is greater than 0.57 (the average of the five predicted ratings by the CFRSS), the IIA will go on using the KBRSS to provide service for the user. However, if the average rating of the user on the items suggested by the KBRSS is less than or equal to 0.57, the IIA will lower k to 1.1 so that $k\bar{U} = 0.55$. Now, the five items previously recommended by the CFRSS can be presented to the user, and the IIA switches the user back to the CFRSS for service.

3.1.3 Initiation of Dialogs

Uncertainty is inevitable in recommender system environments. It is possible that some users do not make preferences clear. For example, a user may rate an item as both "Excellent" and "Below Average", or may want a Middle Ages history book on personal computers. Whether this is the user's mistake or an attempt to test the "intelligence" of the system, the IIA will generate confirmation dialogs in such uncertain situations. These dialogs are not through natural language, rather via text boxes, menu, buttons in a graphical user interface.

3.1.4 View and Change of Users' Interest Profiles

To cope with the fact that users' interests may change, the IIA provides a feature that allows users to view and modify their interest profiles. When a user clicks a button (or selects a menu item) for this feature, the IIA will instruct the Collaborative Filtering Engine to query for the user's profile. It then presents this profile to the user in the modifiable mode. After the user is finished with the modification, the IIA transfers the modified profile to the Collaborative Engine, which then saves this profile into the Database of Users' Ratings, overwriting the old profile.

3.2 The Collaborative Filtering Recommender Subsystem (CFRSS)

The goal of the CFRSS is to make recommendations to a user based on the similarity between the interest profile of that user and those of other users. The heart of the CFRSS is the Collaborative Filtering Engine, which has two major responsibilities:

1. The Collaborative Filtering Engine receives the user's information (user's name, password) from the IIA. It generates queries to the Database of Users' Ratings using the user's information as input. The data returned by the queries are the ratings on items of that user, the ratings on items of other users, and the similarity scores between that user and other users. The Collaborative Filtering Engine then performs the calculation of predicted ratings on items for the user based on the data returned by the queries (e.g., Formula (1)). It then selects n items (say, n = 10) that have the highest predicted calculated ratings. Finally, it queries the Product Database for the attributes (characteristics) of these n selected items, and returns these items together with their attributes to the IIA as output. These items and their attributes are presented to the user by the IIA.

2. The Collaborative Filtering Engine also creates and modifies the users' interest profiles. It creates a new user's interest profile in the Database of Users' Ratings using the user information (user's name, password, and the user's ratings on representative items) that it receives from the IIA. It modifies an existing user's interest profile based on the user's ratings for recommended items provided by the IIA.

A good, efficient design of the Collaborative Filtering Engine should include at least an Automatic Query Generator and a Math Processor as its components (see Figure 4).

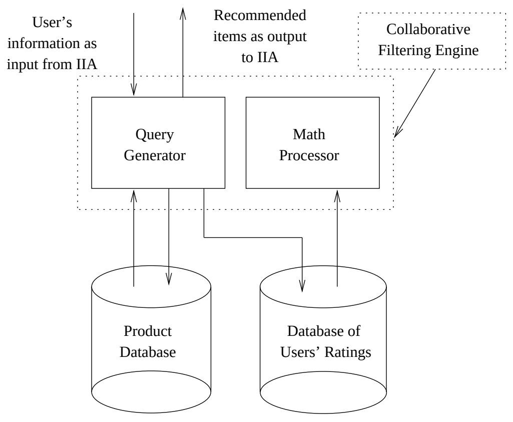

User's information as input from IIA
Recommended items as output to IIA
Collaborative Filtering Engine
Query Generator
Math Processor
Product Database
Database of Users' Ratings

Figure 4. The Collaborative Filtering Recommender Subsystem

The Automatic Query Generator generates queries to the Database of Users' Ratings and the Product Database (as described above), using the user information provided by the IIA as input. The process of query generation occurs dynamically at run time as soon as the Automatic Query Generator receives input from the IIA.

The Math Processor provides the capability to perform complex mathematical manipulations on the data returned from the queries generated by the Automatic Query Generator. These include the ability to apply various arithmetic expressions to the data (such as sums of rows, sums of columns, total etc.), and the ability to perform standard statistical analyses on the data (such as calculating the mean, the standard variation etc.).

3.3 The Knowledge-Based Recommender Subsystem (KBRSS)

The purpose of the KBRSS is to recommend to a user those items that best meet the user's requirements. The core component of the KBRSS is the Knowledge-Based Engine, which generates queries to the Knowledge Base of the product domain, based on the user's requirements it receives from the IIA as input. Upon the query completion, the Knowledge-Based Engine chooses n items that best satisfy the user's requirements. It then queries the Product Database for the attributes (characteristics) of these n chosen items, and returns these items together with their characteristics to the IIA as output. These chosen items and their characteristics are finally presented to the user by the IIA.

The Knowledge-Based Engine, thus, must contain at least an Automatic Query Generator, which dynamically generates queries to the Knowledge Base and the Product Database, using the user's requirements provided by the IIA as input (Figure 5).

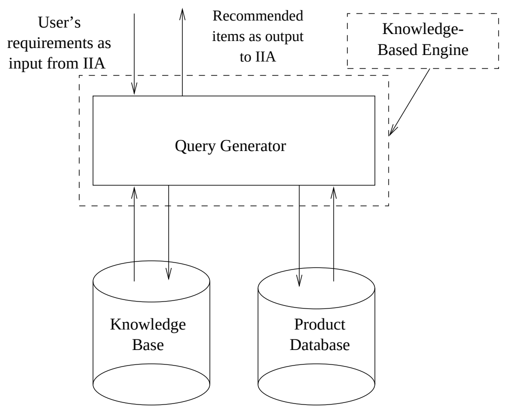

User's requirements as input from IIA
Recommended items as output to IIA
Knowledge-Based Engine
Query Generator
Knowledge Base
Product Database

Figure 5. The Knowledge-Based Recommender Subsystem

The Knowledge Base of the KBRSS contains information about the product domain. Building a knowledge base is not a trivial task. Designers must have a good understanding of the product domain, knowing which features of the product matter to consumers. These features need to be integrated into the entries of the knowledge base in such a way that best facilitates the query process.

4 An Example

To illustrate how our architecture works, we provide an example in which the IIA has to coordinate the operations of both the CFRSS and the KBRSS. The dialogs in the example represent the information exchanged between the user and the system, although in reality, the information exchange process is done through a graphical user interface using boxes, menu and button selection. Italicized sentences denote explanations, scenario descriptions, user's actions, or background actions of the system.

User: *Logs into the system as an existing user.*

System: *Verifies that it is OK to use the CFRSS. However, after calculating the predicted ratings on items, it finds that none of the items are "good items" (ref. Section 3.1.2) for recommendation. So, it switches the user to the KBRSS for service.* Please, choose a preferred book category from the following: Science, Sociology, History etc.

User:*Chooses Science.*

System: Please, selects one of the following channels: Mathematics, Physics, Computer Science etc.

User: *Chooses Computer Science.*

System: Which of the following computer topics do you

prefer: Software Engineering, Database, Operating Systems, Artificial Intelligence etc.?
User: *Indicates her preference in Database.*
System: The following books are recommended:

System shows list of recommended books with links to extra information such as table of contents, author's biography, reader reviews etc. for the user's evaluation.

1. Database Processing by Kroenke
2. Access Database by Steven Roman
3. *The list continues...*

Please, indicate how you like the books by checking on the appropriate boxes.
User: *Checks on boxes indicating her satisfaction.*
System: *Records these ratings into the user's interest profile.*

5 Discussion

One of the most challenging tasks that an online customer is facing today is how to sort through huge catalog of products to find items of most value to her. Even when the customer has an idea of some product to purchase, she will probably be confused by so many available products of the same type, produced by different manufacturers and perhaps slightly dissimilar in prices, usage, colors, features etc. It is therefore very hard for the customer to make a purchase decision from such a wide choice of availability.

Today's online customers are also known to be more impatient and demanding than ever before. They would like to make good purchases, yet in as little time as possible. In other words, customers want to receive satisfactory products, but they are not willing to spend much effort for their purchases.

For the above reasons, automated product recommendation presently becomes a very important issue in business-to-consumer e-commerce. In order to maximize the revenue generated by online customers, e-commerce businesses must be able to offer relevant recommendations to their online visitors as quickly as possible and with minimal customer burden. The architecture for an integrated recommender system presented in this paper is an attempt to address the issue. Developing such a recommender system would allow online businesses to satisfy both their own need to direct customer purchases, and their customers' desire to move quickly through large product catalogs to find the most appropriate items.

6 Related Work

Knowledge-base recommender systems solicit information about users' preferences and generate recommendations by reasoning about what products meet the users' requirements. For instance, the PersonalLogic recommender system creates a dialog to walk the user down a tree of product features [1]. Other systems have employed quantitative decision support or case-based reasoning tools for making recommendations [2, 5].

Collaborative filtering recommender systems have been developed for many applications. For examples, Grouplens assists users in finding usenet news articles [10], Memoir helps people find other people (rather than documents) with similar interests [6], and Ringo recommends music albums and artists based on word-of-mouth recommendations [16].

A number of hybrid recommender systems that make use of different approaches have also been advocated [4, 8, 9, 12, 13]. The FindMe system, as outlined in [4], attempts to combine the knowledge-based and collaborative filtering techniques; however, it takes a different approach compared to our architecture. In this system, the collaborative filter is only used as a post filter after the knowledge-based system has done its work. As admitted in [4], this approach does not capitalize on the full power of collaborative filtering, which, in its pure form, permits the discovery of niche groups of customers, who share similar interests.

The Decision-Theoretic Interactive Video Advisor (DIVA) is an application of decision theory to collaborative filtering [12]. The DIVA distinguishes the users' short-term and long-term preferences. A user can indicates her short-term preferences by specifying some constraints, such as actors and actresses, directors, genres, release years etc. These constraints act as an initial filter on the entries in the movie database. The movies that satisfy the constraints are then ranked and presented to the user according to the user's preference structure. The DIVA and our architecture differ in a number of ways. First, the DIVA with its "short-term preference" feature allows the user's interest to drift a little bit from the user's interest profile. Our architecture, by selecting the KBRSS, can make recommendations totally independent of the user's interest profile. Second, the DIVA was designed to make recommendations on movies, while our architecture can apply to a wide range of diverse products. Finally, the DIVA makes recommendations based on decision theory, in which user's preferences are represented using pair-wise comparisons among items. In contrast, our architecture integrates the knowledge-based and collaborative filtering techniques.

7 Conclusion

In this paper we have presented an architecture for a hybrid recommender system, which integrates knowledge-based and collaborative filtering recommender systems as its subsystems. Our architecture uses an interactive interface agent that flexibly selects either the knowledge-based or the collaborative filtering recommender subsystem for service. The interactive interface agent can even coordinate the operations of the two subsystems to make the best possible recommendations to users. Such a hybrid system will have all the advantages of a collaborative filtering recommender system, namely the ability to make personalized recommendations, the capability to identify niches precisely, and the prediction quality being improved over time. Additionally, with the presence of the knowledge-based subsystem, the proposed system can avoid the major disadvantages of a collaborative filtering recommender system, such as the requirement to be initialized with a large database of users' preferences, and the possibility to generate invalid recommendations when a user's interests change.

Furthermore, since our architecture combines both knowledge-based and collaborative filtering approaches, it can generate good recommendations for a wide range of diverse products. In other words, it can be used to recommend to users those products that lend themselves to the knowledge-based approach, such as cars, houses etc., as well as the products that are more suitable to the collaborative filtering approach, e.g., books, movies, CDs etc.

As with any knowledge-based systems, our approach requires knowledge engineering of the product domain. The necessary investment in knowledge engineering should be justified by the many advantages of the system mentioned above. As with any collaborative filtering systems, our proposed architecture faces two challenges: the computational cost and the search cost in a database of users' preferences that tends to get larger and larger. A possible solution to the former challenge is multi-threading. That is, we can use a lightweight process (or thread) to perform computation in the background, while the user is interacting with the main process in the foreground. A possible solution to the latter challenge is data partition. That is, the database can be partitioned into several sub-tables, using some category such as alphabetical ordering. Thus, a search can be done in an appropriate sub-table, rather than in the entire database. We would like to implement our approach in the next research step, to experimentally evaluate the proposed architecture and design ideas.

References

[1] Personallogic recommender system: http://www.personallogic.com.

[2] H. K. Bhargava, S. Sridhar, and C. Herrick. Beyond spreadsheets: Tools for building decision support systems. *IEEE Computer*, 32(3):31–39, 1997.

[3] D. Billsus and M. J. Pazzani. Learning collaborative information filters. Papers from The AAAI Workshop on Recommender Systems WS-98-08, AAAI Press, Menlo Park, California, 1998.

[4] R. Burke. Integrating knowledge-based and collaborative-filtering recommender systems. Papers from The AAAI Workshop on Artificial Intelligence for Electronic Commerce WS-99-01, AAAI Press, Menlo Park, California, 1999.

[5] R. Burke, K. Hammond, and E. Cooper. Knowledge-based navigation of complex information spaces. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 462–468, 1996.

[6] D. C. DeRoure, W. Hall, S. Reich, A. Pikrakis, G. J. Hill, and M. Stairmand. Memoir an open framework for enhanced navigation of distributed information. *Information Processing and Management*, 37:53–74, 2001.

[7] D. R. Greening. Collaborative filtering for web marketing efforts. Papers from The AAAI Workshop on Recommender Systems WS-98-08, AAAI Press, Menlo Park, California, 1998.

[8] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 241–250, 2000.

[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Information Systems*, 22(1):5–53, 2004.

[10] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[11] M. Montaner, B. Lopez, and J. L. Dela. A taxonomy of recommender agents on the internet. *Artificial Intelligence Review*, 19:285–330, 2003.

[12] H. Nguyen and P. Haddawy. Diva: Applying decision theory to collaborative filtering. Papers from The AAAI Workshop on Artificial Intelligence for Electronic Commerce WS-99-01, AAAI Press, Menlo Park, California, 1999.

[13] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI-2001)*, pages 437–444, 2001.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.

[15] P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[16] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating word of mouth. In *Proceedings of Computer Human Interaction*, pages 210–217, 1995.

[17] L. Terveen and W. Hill. Beyond recommender systems: Helping people help each other. In *HCI in the New Millennium, J. Carroll, Ed.* Addison Wesley, 2001.