2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology

Adapting Recommendations Organization to User Preferences

Li Chen
Department of Computer Science
Hong Kong Baptist University, Hong Kong
lichen@comp.hkbu.edu.hk

***Abstract* — Given the overwhelming information appearing in the current web environment, recommendations have been increasingly applied to assist users in handling with the information overload and locating items that interest them. As a different way to generate and display recommendations, the organization interface has been found being more effective in building users' trust. In this paper, we propose a novel approach to generating the organization of recommendations, with the goal of making it dynamically adaptive to different conditions of user preferences (i.e. "incomplete preferences" and/or "preference conflicts"), so as to optimally support users' preference construction process and accurate decisions.**

Keywords - web recommendations, organization interface, preference elicitation

I. Introduction

The recommender system has emerged as an important research area in the web environment over the last decade [1]. It is a software application that aims to support users in efficiently locating their desired items when interacting with large information spaces. Most of related systems have been oriented to products with low-risks such as music, movies and books, for which users' preferences can be built according to their prior usage experiences. However, as for products with complex structures and high financial risks (e.g. digital cameras, computers, and cars), it is unlikely to infer the user's needs up front given that few people would have experienced them before they search for a new one. Essentially, the user's preferences in such product domains have been often defined as multi-objective preferential decision problems [4,8], since they inherently consist of multiple criteria to be satisfied (e.g. the criteria on the computer's price, processor speed, memory, etc.).

A more intelligent and personalized preference elicitation tool is hence necessarily required to help users build accurate preference models and maximally improve their decision accuracy. According to adaptive decision theory [7], human decision process is in nature highly constructive and adaptive to the current decision task and decision environment, especially when s/he is confronted with an unfamiliar product domain or overwhelming information. Most of traditional preference elicitation tools nevertheless neglect that users' initial preferences can be uncertain and erroneous. They ask users to answer a list of fixed need or preference assessment questions to which users may lack the knowledge and motivation to respond correctly. On the other hand, when a user has established a certain set of criteria while they have conflicting values (e.g. higher processor speed and cheaper price), a "nothing found" message is usually returned as in most of existing e-commerce websites because the system simply retrieves products that exactly match all of the user's criteria.

In fact, the two phenomena, "incomplete preferences" and "conflicting preferences", commonly appear at different stages of the user's preference construction process [10]. It hence poses the question of how to effectively guide users to establish accurate preferences via appropriate and informative recommendations. Some approaches, such as case-based conversational recommenders and example-critiquing systems [6,8,11], unfortunately, are limited in adapting the generation of their recommendations to the variety of user preferences in nature.

In this paper, we propose a novel approach, called *adaptive recommendations organization*, with three principal objectives: 1) *personalization and adaptability*. The recommendation computation is personalized to treat different preference conditions separately, considering individual requirements and being adaptive within the single user's whole decision session. For instance, when the user's preferences are incomplete, preference suggestions will be presented to stimulate users to reveal hidden needs. In another condition that no available item is satisfactory with all of the user's current criteria, a partial satisfaction set will be returned with suggested tradeoff directions so as for the user to adjust her preferences' weights; 2) *organization*. Recommended items are organized into categories in terms of their similar and shared properties, given that the structured and organized view has been found to more effectively enhance users' subjective trust constructs and have higher potential to increase their decision performance [3,9], compared to the list view where items are purely listed one by one; 3) *explanation*. We use explanations not only to provide system transparency, but more importantly to educate users about the incompleteness and/or conflicts appearing in their specified preferences.

II. Adaptive Recommendations Organization

Owing to the constructive nature of user preference establishment, our algorithm is developed to dynamically adjust to the user's current preferences and accordingly evoke the appropriate component. More concretely, during each recommendation cycle, it first analyzes the user's stated preferences by far and then determines whether to produce recommendations to stimulate users to uncover hidden needs, or guide them to revise preferences (if there are conflicts), or act with both purposes. The user's reactions to those recommendations will be used by the system to automatically refine the user's preference model and compute a new set of recommendations in the next cycle. This incremental preference elicitation and recommendation process can continue till the user's preferences are maximally complete and precise, at which point the best matching product would be the user's ideally targeted choice. In the following, we in detail describe how recommendations are computed and organized respectively in "incomplete preferences" and "conflicting preferences" conditions.

A. *Modeling User Preferences*

We first model the user preferences over all products according to the Multi-Attribute Utility Theory (MAUT) under the additive independence assumption [4]. This MAUT-based user model is inherently in accordance with the most normal and compensatory decision strategy, the Weighted Additive Rule (WADD) that explicitly resolves conflicting value preferences by considering tradeoffs [7]. Each user's preference model is formally defined as a set of pairs $\{(V_1, w_1), (V_2, w_2), \ldots, (V_n, w_n)\}$, where V_i is the value function for each participating attribute A_i (normalized within the range of [0..1]), and w_i is the importance (i.e., weight) of A_i relative to other attributes. A utility score of each product ($\langle a_1, a_2, \ldots, a_n \rangle$) (where a_i is the product's value on A_i) can be hence calculated with the formula (1), which represents its weighted satisfying degree:

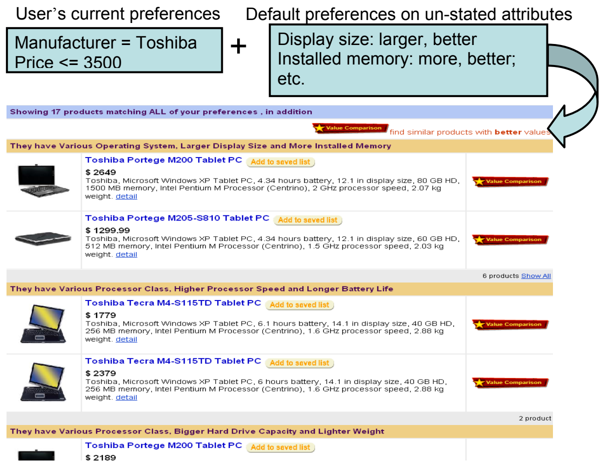
$$U(\langle a_1, a_2, \ldots, a_n \rangle) = \sum_{i=1}^{n} w_i V_i(a_i) \qquad (1)$$

B. *Stimulating Hidden Preferences*

According to adaptive decision theory as well as observations from our previous user studies [7,8], the user's preferences are often incomplete and uncertain, especially during her initial interaction cycles when she is unfamiliar with the product catalog or has not formed strong objectives for what she is truly interested in. Our algorithm constantly captures such incompleteness through both of the preferences the user already stated and the reaction she did to the displayed items. Specifically, during one interaction cycle, assuming that the user has already stated criteria over attributes $A^* = (A_{i1}, A_{i2}, \ldots, A_{im})$, while not on other attributes $A' = \{A_j, A_j \in A$ and $A_j \notin A^*\}$ (A is the set of all attributes), default preferences on the attributes A' will be incorporated for the recommendation computation. For example, if the user stated a price range initially but did not specify any requirement on the processor speed of PCs (either due to unfamiliarity with such feature or lack of knowledge about its association with other attributes), the system will integrate the default preference on the processor speed (e.g. the higher, the better) and then suggest to the user for her consideration.

As indicated by [5], adding default preferences saves the user's effort by allowing her to provide fewer preferences initially. We believe that except for this benefit, it could also help users become more knowledgeable about the product domain and stimulate them to reveal hidden needs when concrete example products with such suggestions are shown. Furthermore, the cold-start problem, a typical phenomenon encountered by related recommender systems [1], can be avoided in our approach since even none of criteria was specified by the user in the beginning, a set of recommendations will be still returned with suggested preferences on participating attributes. These suggestions are additionally organized by discovering frequently associated attributes among the retrieval set and presented in the form as "these products have higher processor speed and bigger memory that you may like".

Formally, we discover association rules between un-stated attributes, based on which suggestions are made on them. The Apiori algorithm (a typical data mining tool to discover association rules) has been applied to fulfill this task [2]. Each product in the retrieval set (that matches the user's currently stated preferences) is converted into a vector, indicating its properties on the un-stated attributes ("*improved*" denoted as ↑ or "*compromised*" denoted as ↓) by comparing each of the attribute values with its average across all retrieved products. As an example, one product can be formalized as {(processor speed, ↑), (weight, ↑), (memory, ↓)} (where the three attributes are with un-stated criteria), meaning that this product has higher processor speed, lighter weight but smaller memory relative to their average values in the retrieval set. These product vectors are then inputted to the Apriori algorithm so as to identify how different attribute properties are associated between one another. The discovery of associative properties can hence help the user realize what additional benefits she could obtain in the displayed products (see the sample interface in Figure 1).


User's current preferences
Manufacturer = Toshiba
Price <= 3500
+
Default preferences on un-stated attributes
Display size: larger, better
Installed memory: more, better; etc.
Value Comparison
find similar products with better values
Add to saved list
Value Comparison

Showing 17 products matching ALL of your preferences , in addition

They have Various Operating System, Larger Display Size and More Installed Memory

Toshiba Portege M200 Tablet PC
$ 2649
Toshiba, Microsoft Windows XP Tablet PC, 4.34 hours battery, 12.1 in display size, 80 GB HD, 1500 MB memory, Intel Pentium M Processor (Centrino), 2 GHz processor speed, 2.07 kg weight. detail

Toshiba Portege M205-S810 Tablet PC
$ 1299.99
Toshiba, Microsoft Windows XP Tablet PC, 4.34 hours battery, 12.1 in display size, 60 GB HD, 512 MB memory, Intel Pentium M Processor (Centrino), 1.5 GHz processor speed, 2.03 kg weight. detail

6 products Show All

They have Various Processor Class, Higher Processor Speed and Longer Battery Life

Toshiba Tecra M4-S115TD Tablet PC
$ 1779
Toshiba, Microsoft Windows XP Tablet PC, 6.1 hours battery, 14.1 in display size, 40 GB HD, 256 MB memory, Intel Pentium M Processor (Centrino), 1.6 GHz processor speed, 2.88 kg weight. detail

Toshiba Tecra M4-S115TD Tablet PC
$ 2379
Toshiba, Microsoft Windows XP Tablet PC, 6 hours battery, 14.1 in display size, 40 GB HD, 256 MB memory, Intel Pentium M Processor (Centrino), 1.6 GHz processor speed, 2.88 kg weight. detail

2 product

They have Various Processor Class, Bigger Hard Drive Capacity and Lighter Weight

Toshiba Portege M200 Tablet PC
$ 2189

Figure 1. Organization of recommendations to stimulate hidden preferences.

C. *Solving Preference Conflicts*

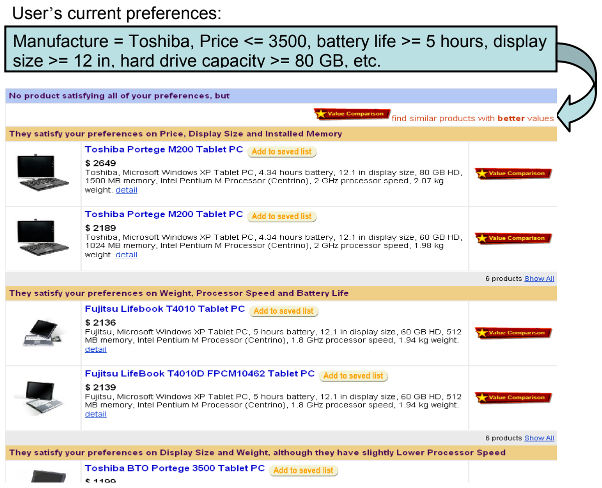
Compete preferences do not mean that they are accurate enough. It sometimes happens that there are conflicting attribute preferences, so no available item exactly satisfies all of them, which indeed also occurs in the condition of incomplete preferences where conflicts exist in the user's stated criteria. At this point, the system should be able to help the user revise the relative importance (i.e. the weight w_i in the formula 1), which is in nature a tradeoff process. Thus, it can be seen that during this step, the main purpose is to adjust weight values, while in section *B* it is to elicit value functions (i.e. $V_i(a_i)$ in the formula (1)) on un-stated attributes.

Tradeoff-making involves increasing weights on attributes that are more important for the user, while accepting compromises with the decreased weights on less important ones. We propose different tradeoff directions for the user to consider in order to help them decide which attribute constraints they would be willing to relax in return for ideal matching on more important ones.

Specifically, a partial satisfaction set is first retrieved (which are best nearly satisfying the user's current preferences) and tradeoff relations between conflicting attributes among these retrieved products are to be discovered. The Apriori algorithm is again employed here to determine the association rules, but the inputs to it is different from the ones in section *B*. That is, each product is formally converted into a tradeoff vector containing the information of which attribute preferences it satisfies and which it does not. For example, one product is formalized like {(display size, ↑), (weight, ↑), (processor speed, ↓)} (here ↑ means "satisfactory" and ↓ "unsatisfactory"), representing that "this product satisfies your preferences on display size and weight, but not on processor speed". All of the tradeoff vectors are then transferred into Apriori so as to mine the recurring and representative association rules with the form of X => Y. Each rule infers that X (with satisfying attributes) is frequently associated with Y (dissatisfying ones) in the retrieved products. With the presence of these rules (see Figure 2), the user could decide the tradeoff she would like to accept, e.g. decreasing weights of less important attributes (i.e. Y in one rule) and emphasizing on more important ones (i.e. increasing weights of attributes in X).

Therefore, such association rules reveal to the user the conflicting relations being in her stated preferences and indicate different tradeoff directions that she may choose, which will result in the refinement of her preference model in terms of weight adjustment. Our method can be hence regarded as an improvement on purely presenting partially satisfied products, through showing the association knowledge and revision suggestions.

The conflicting phenomenon can be easily captured by the system when few or no available product is matching to the user's stated preferences. It may also appear simultaneously with the observation of "incomplete preferences" as described in section *B,* at which point both of preference and tradeoff suggestions will be presented, e.g. "these products satisfy your preferences on price, weight, but not on processor speed. In addition, they have bigger hard capability". The user can then decide whether to take such tradeoff (e.g. lower processor speed for cheaper price and lighter weight) and indicate additional interests on other previously un-considered attributes (e.g. hard capability).


User's current preferences:

Manufacture = Toshiba, Price <= 3500, battery life >= 5 hours, display size >= 12 in, hard drive capacity >= 80 GB, etc.

No product satisfying all of your preferences, but

find similar products with **better** values

They satisfy your preferences on Price, Display Size and Installed Memory

Toshiba Portege M200 Tablet PC

$ 2649

Toshiba, Microsoft Windows XP Tablet PC, 4.34 hours battery, 12.1 in display size, 80 GB HD, 1500 MB memory, Intel Pentium M Processor (Centrino), 2 GHz processor speed, 2.07 kg weight. detail

Toshiba Portege M200 Tablet PC

$ 2189

Toshiba, Microsoft Windows XP Tablet PC, 4.34 hours battery, 12.1 in display size, 60 GB HD, 1024 MB memory, Intel Pentium M Processor (Centrino), 2 GHz processor speed, 1.98 kg weight. detail

6 products Show All

They satisfy your preferences on Weight, Processor Speed and Battery Life

Fujitsu Lifebook T4010 Tablet PC

$ 2136

Fujitsu, Microsoft Windows XP Tablet PC, 5 hours battery, 12.1 in display size, 60 GB HD, 512 MB memory, Intel Pentium M Processor (Centrino), 1.8 GHz processor speed, 1.94 kg weight. detail

Fujitsu LifeBook T4010D FPCM10462 Tablet PC

$ 2139

Fujitsu, Microsoft Windows XP Tablet PC, 5 hours battery, 12.1 in display size, 60 GB HD, 512 MB memory, Intel Pentium M Processor (Centrino), 1.8 GHz processor speed, 1.94 kg weight. detail

6 products Show All

They satisfy your preferences on Display Size and Weight, although they have slightly Lower Processor Speed

Toshiba BTO Portege 3500 Tablet PC

$ 1199

Figure 2. Organization of recommendations to solve preference conflicts.

D. *Organizing and Explaining*

Since a number of association rules will be likely returned by the Apriori, it then comes to the step of selecting the most prominent ones. Being different from standard ranking strategy that simply selects ones with lower supports (i.e. lower percentage of products that satisfy the rule) [11], our approach is to favor rules with higher relative gains (against losses) according to the user's preferences. The score is concretely determined by two parts (see the formula (2)): one is the weighted value of the rule, and another is the average utility of products satisfying it. Formally, each association rule is a set of (attribute, tradeoff) pairs, where the "improved"/"compromised" property assigned to attribute (in section *B*) or "satisfactory"/ "unsatisfactory" property (in section *C*) are all unified under the term "tradeoff". Thus, the score of each association rule is computed as:

$$RuleScore(C) = (\sum_{i=1}^{|C|} w(attribute_i) \times tradeoff_i) \times (\frac{1}{|SR(C)|} \sum_{r \in SR(C)}^{|SR(C)|} U(r)) \qquad (2)$$

where C denotes the rule (a set of (attribute, tradeoff) pairs) and SR(C) is the set of products that satisfy C.

$\sum_{i=1}^{|C|} w(attribute_i) \times tradeoff_i$ hence computes the weighted sum of tradeoff properties involved in C. $w(attribute_i)$ is the weight of *attribute$_i$* (default as 3, the middle point of the range from 1 to 5 for attributes without explicitly stated weights) and *tradeoff*$_i$ is default as 0.75 if "improved" or "satisfactory", or 0.25 if "compromised" or "unsatisfactory". $\frac{1}{|SR(C)|}\sum_{r \in SR(C)}^{|SR(C)|} U(r)$ is the average utility (from formula (1)) of all the products that satisfy the rule C.

Diversity is further incorporated in order to avoid returning categories of products too similar to each other. Concretely, each remaining un-selected category (i.e. one association rule with its satisfying products) is computed with a diversity degree which indicates its dissimilarity to the so-far selected categories. The one with the highest unified score (combined with the RuleScore as in the formula (3)) will be then selected:

$$F(C) = RuleScore(C) \times Diversity(C, SC) \qquad (3)$$

where SC denotes the set of categories selected thus far. Thus the first selected category should be the one with the highest RuleScore (since its SC is empty), and the subsequent category is selected if it has the highest value of *F(C)* in comparison with the current SC set. The selection process ends when the desired *k* categories have been determined. The diversity degree of C is calculated as the minimal local diversity of C with all categories in the SC set. The local diversity of two categories (C and C_i in SC) is further defined by two factors (see the formula (4)): the diversity between rules themselves and the diversity between their associated products (i.e. SR(C) and SR(C_i)).

$$Diversity(C, SC) = \min_{C_i \in SC}\left(\left(1 - \frac{|C \cap C_i|}{|C|}\right) \times \left(1 - \frac{|SR(C) \cap SR(C_i)|}{|SR(C)|}\right)\right) \qquad (4)$$

Once the desired *k* categories have been selected, each category containing a group of products sharing the same association rule (e.g. {(price, ↑), (weight, ↑), (memory, ↓)}), a pre-designed set of explanation templates is used to explain the rules in a conversational language so that the user may easily understand. Specifically, during this explanation step, the property (i.e. ↑ and ↓) assigned to each attribute is concretized in terms of its actual meaning. For instance, if it is "improved" on an attribute (e.g. processor speed), it means it is a suggested preference on this attribute and the explanation is hence generated like "higher processor speed" by correlating it with the appropriate phrase in the pre-designed explanation base. In another case, if it is "satisfactory" referring to an attribute value satisfying the user's stated criterion (e.g. on price), the explanation outcome will be like "satisfy your preference on price". The explanations of a whole association rule will be displayed as a category title to represent its associated products (see Figure 1&2).

III. CONCLUSION

In this paper, we proposed a new approach to generating and organizing recommendations taking into account of users' adaptive preference construction in nature. At different stages of the construction process, the preferences' completeness and certainty degrees may vary. They will be incomplete in terms of un-stated attributes that the user may lack of knowledge. The phenomenon of preference conflicts will also appear when no product satisfies all of the user's stated criteria. An adaptive recommender method that can dynamically respond to the variety of preference conditions should be hence meaningful to help users establish accurate preferences accordingly, especially when they are searching for less experienced and high involvement products (e.g. computers and cars) that standard collaborative filtering and content-based recommender systems can not ideally fit for.

Our algorithm is essentially based on the association rule mining technique to discover association rules dependent on the current condition of the user preferences. Preference suggestions on un-stated attributes are identified to stimulate hidden needs, or explanations of partial satisfaction set are returned to show tradeoff relations among users' conflicting attribute preferences. We descried the algorithm steps in these conditions and illustrated the interface design respecting its organization and explanation characteristics. For the future work, we will verify our method's practical benefits through both of empirical experiments and user evaluations, and will attempt to integrate it in real online websites in order to test its stability and scalability.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, Toward the Next Generation of Recommender Systems: a Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17 (6), 2005, pp. 734-749.

[2] R. Agrawal, T. Imielinski and A. Swami, Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD international Conference on Management of Data*, 1993, pp. 207–216.

[3] L. Chen and P. Pu, Preference-based Organization Interfaces: Aiding User Critiques in Recommender Systems. In *Proceedings of International Conference on User Modeling*, 2007, pp. 77-86.

[4] R. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge University Press, 1976.

[5] G. Linden, S. Hanks and N. Lesh. Interactive Assessment of User Preferences Models: the Automated Travel Assistant. In *Proceedings of the 5th International Conference on User Modeling*, 1997, pp. 67-78.

[6] L. McGinty and B. Smyth, Adaptive Selection: An Analysis of Critiquing and Preference-Based Feedback in Conversational Recommender Systems. *International Journal of Electronic Commerce* 11 (2), 2006, pp. 35-57.

[7] J.W. Payne, J.R. Bettman and E.J. Johnson, *The Adaptive Decision Maker*. Cambridge University Press, 1993.

[8] P. Pu and L. Chen, Integrating Tradeoff Support in Product Search Tools for e-Commerce Sites. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, 2005, pp. 269-278

[9] P. Pu and L. Chen, Trust Building with Explanation Interfaces. In *Proceedings of the 11th International Conference on Intelligent User Interfaces*, 2006, pp. 93-100.

[10] P. Pu and B. Faltings, Decision Tradeoff Using Example-Critiquing and Constraint Programming. *Constraints* 9 (4), 2004, pp. 289-310.

[11] J. Reilly, K. McCarthy, L. McGinty and B. Smyth, Dynamic Critiquing. In *Proceedings of the 7th European Conference on Case-Based Reasoning*, 2004, pp. 763-777.