Integrating Travel Planning and On-Tour Support in a Case-Based Recommender System

Francesco Ricci, Dario Cavada and Quang Nhat Nguyen

eCommerce and Tourism Research Laboratory
ITC-irst
via Sommarive 18
38050 Povo, Italy
{ricci,dacavada,quang}@itc.it

1 Introduction

Nowadays a large number of web sites support a traveller in the selection of a travel destination or a travel service (e.g., flight or hotel). Typically, the user is required to input product constraints or preferences, that are matched by the system in an electronic catalogue (see for instance Expedia.com, Priceline.com, TISCover.com, etc,). Actually, querying such catalogues is not straightforward, the user may not have enough knowledge to formulate the query or the number of results could overflow the capability of the user to compare the offers and finally select the most appropriate one.

Therefore major eCommerce web sites dedicated to travel and tourism have recently started to better cope with leisure travel planning by incorporating recommender systems, i.e., applications that provide advice to users about products that might interest them in [1]. Recommender systems for travel planning try to mimic the interactivity observed in traditional counselling sessions with travel agents [2]. None of the existing recommender systems for travel planning can support a multi-stages interaction (conversation) or help the user in building a "user-defined" travel, made of one or more locations to visit, an accommodation and additional attractions (museum, theater, etc.).

The ITR (Intelligent Travel Recommender) system is aimed at supporting the user in building such a coherent (from the user point of view) bundling of products [5]. The system exploits a case base of travel bags built by a community of users as well as catalogues provided by a Destination Management Organization (APT Trentino). ITR integrates case-based reasoning with interactive query management [3]. ITR tries first to cope with user needs satisfying the logical conditions expressed in the user's query and, if these are not satisfiable, it suggests query changes (relaxation and tightening) that will produce acceptable results. Then, case similarity is exploited first, to retrieve relevant old recommendation sessions, and second to rank the items in the result set of the user's given logical query. The rank is given by computing the similarity of the items in the result set with those contained in past similar sessions.

ITR is very successful in managing the man/machine interaction, reducing the number of interaction steps needed for the user to finally select the chosen item, but still it is not directly usable on a mobile device. The mobile context imposes quite a different interaction management and decision model.[1] For instance, ITR assumes that the user : a) is planning with some time in advance, b) he is far from the destination, c) he wants to compare alternative options and optimize his "utility" function, d) he should be allowed to browse information and multimedia data describing the option, e) he would like to locate the destination on a map and assess the spatial relationships with some potential secondary destinations.

When the traveller is on-tour the perspective is totally different. He/she is rarely interested in a complete travel offer but rather on some complementary service, e.g. additional attractions or restaurants or events that fit well into the plan. Moreover the interaction must be faster since context information and pre-travel choices, which are already included in the travel plan, must narrow down the available options. In this abstract we briefly describe a methodology for reducing at minimal the input required to select a tourism service. Instead of asking the user to input explicitly his needs, the mobile component (ITR-mobile) exploits the knowledge contained in the pre-travel plan to guess an initial set of candidate solutions. The user is then required to select one of the proposed items or to provide critics to some of the items, such as: "I like this feature of this item", "I want something less expensive", "I do not like this feature of this item". This enables the system to update the query and re-run the selection and ranking tools implemented in ITR but reducing at minimal the user efforts. Our approach is inspired by previous approaches [7, 4] that have introduced the idea of "recommendation by proposing" and similarity based query revision. The originality of our approach resides in the combination of logical and similarity-based queries and in the method used to identify the initial set of candidate products that exploits the case base of travel plans contained in the ITR systems. Hence, this paper shows a synergic integration of pre-travel services and knowledge with on-tour context-aware support.

2 ITR Description

This section describes a typical user/system interaction and shows how ITR can be used as a pre-travel planning tool. Let's assume that a traveller wants to have a summer vacation with his family in Trentino. The main page, as depicted in Figure 1, allows the traveller to input his general travel wishes and constraints. In this scenario, the traveller specifies that he will travel with his family, that his budget is between 20 and 40 euros and he will use his car; he wants to stay in a hotel for 2 weeks in July, and he comes from Italy. He and his wife have never been in Trentino, and they wish to relax and to practice some sport activities.

Then the system allows the user to search locations, accommodations, attractions and activities that can be added to the tourist's "travel bag", a cart where the user stores the items that he considers interesting and he wants to

[1] For lack of space we do not address here additional ITR limitations such as those related to the GUI or to the bandwidth requirements.

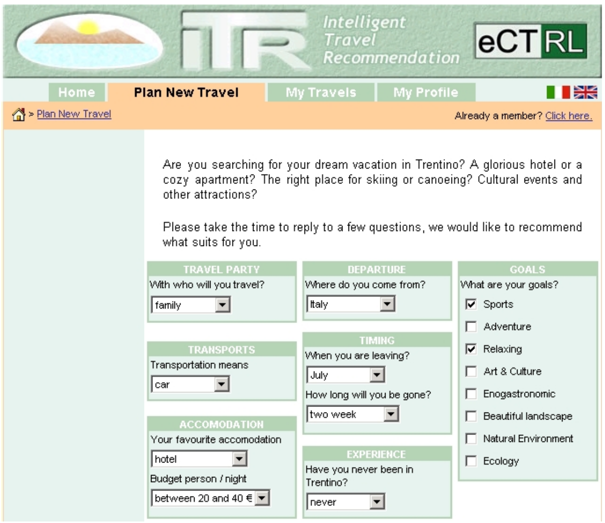

Fig. 1. ITR main page.

consume or visit during his vacation. Our traveller starts looking for a specific location. The system presents a page where the user can specify some logical constraints on the location. The system searches in the locations catalog, and suggests the three locations that match the best. The system shows the rank assigned to each item, and provides an explanation. The explanation arguments is that the suggested item is similar to another item contained in another travel bag, built by the traveller himself or by another user, having similar wishes and constraints. The traveller can add a location to his personal travel bag.

Then the traveller looks for an accommodation in the selected location expressing new constraints. If no accommodation meets all the constraints then the system helps the user by suggesting how he can modify the query to get some results.[2] If the user accepts one of the system proposed alternative queries then a new result list can be obtained. The user can browse the list of the suggested hotels, and adds one of them to his personal travel bag. With a similar interaction, the user can further add attractions or sports activities.

3 ITR-mobile

We now focus on ITR-mobile, a system, currently under development, that aims at providing additional services when the traveller is on-tour. Therefore, we assume that the user has already built some pre-travel bag, and during the travel he can access such data, e.g download it on the mobile.

[2] for lack of space we do not show here the corresponding GUI, neither explain how this is technically achieved [6, 5]

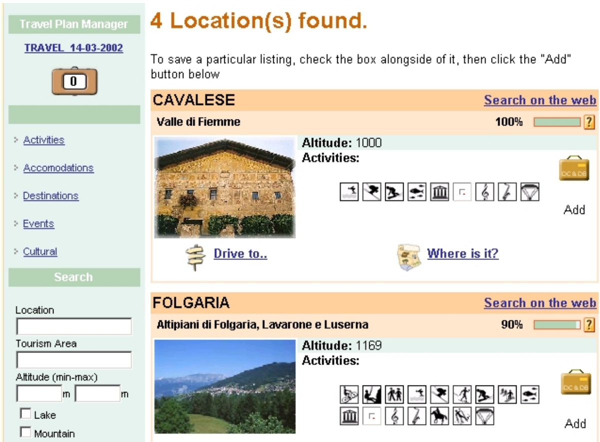

Fig. 2. Result of a location search (example).

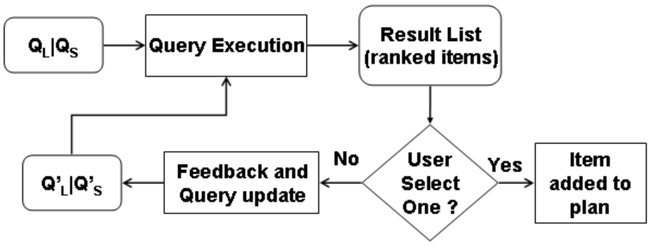

Fig. 3. Query refining cycle.

We concentrate here on the methodology designed to reduce the dialog length when the ITR-Mobile is queried to recommend something to the user. Let us consider the situation in which the user wants to select a restaurant that "fits" into a given travel bag. Therefor, if for instance the bag is adapted to a low-budget profile, the restaurant recommended should be of that type.

The basic idea is to initialize the restaurant selection process with a query, built using conditions determined by the bag and the context (expanded context). The query contains two components:

- the **logical component** of the query Q_L, contains some logical constraint such as location and date, that must be necessarily satisfied.
- the **similarity-based component** Q_S, contains a restaurant pattern (partially defined restaurant) that is computed using a restaurant found in other bags stored in ITR and similar to the current bag which is going to be extended (see [5] for a description of the similarity function).

Therefore the logical component contains strict constraints (mainly related to the context) whether the similarity-based component contains soft constraint, dealing with optional personalization issues. The initial query used for selecting an initial set of results (ranked list) is the combination of both, first the logical part is executed and then the similarity-based one sorts the items found. We denote this combination with $(Q_L|Q_S)$.

Figure 3, describes the proposed interaction cycle. The ranked list of restaurants proposed by the ITR-mobile are browsed by the user and either an item is selected or feed-back (critic) is provided regarding one or more items displayed. Before describing the user feed-backs allowed by the system let us define a few terms:

- **Item** is represented as a vector of feature values $x = (x_1, \ldots, x_n)$. The feature value x_i may be numeric or nominal.
- **Logical query** is defined as a conjunctions of constraints c_i on a single feature $Q_L = c_1 \wedge \cdots \wedge c_n$. We assume that some c_i may be null (no constraint on i-th feature) and the form of the constraint depend on the feature type. Typically constraints on nominal features are equality constraints and those on numeric features are range constraints.
- **Similarity-based query** is determined by a probe item, i.e., a vector of feature values $Q_S = (p_1, \cdots, p_n)$, that is matched with the items contained in the catalogue with an etherogeneous Euclidean Overlapping distance metric (see [5]). Note that some of the p_i can be unknown.

For instance, let us assume that the restaurants can be described by the features: City (nominal), Cost (numeric), Cuisine (nominal). A Logical query could be $Q_L = (City = Trento)$ and a similarity-based query $Q_S = (?, 20, indian)$, where the special symbol ? means that the city is not specified. $(Q_L|Q_S)$ will retrieve the restaurant in Trento that are around 20 euro and possibly have cuisine type indian.

We now sketch how given a query $(Q_L|Q_S)$ and a user feed-back on an item belonging to the result list of the query $y = (y_1, \ldots, y_n)$, the query can be update to retrieve a new set of items. Here we simplify the discussion assuming that only one feed-back is provided. Moreover, we assume that the feed-back is explicitly given by the user and not guessed reasoning on the user behavior. We list here three kinds of feed-back and their corresponding query update rules.

- **F1** The user states that an item is good because he likes the value y_i of that item. Moreover he wants to see some more items similar to that. Then $p_i := y_i$, i.e. the i-th probe feature value p_i becomes equal to the feature value of the liked item.
- **F2** The user states that he generally likes an item but wants something "less" with respect to a feature, for instance, less expensive. In this case, assuming that the i-th feature is commented by the user, then the constraint $c_i = [x_i \leq y_i - \delta]$ is added Q_L. Here δ is a small positive number that is tuned to cause the retrieval of items rather different from that criticized.
- **F3** The user states that he likes the item $y = (y_1, \ldots, y_n)$ but he would accept even something "more ..." with respect to a feature y_i, e.g. "more distant" or "more expensive". In this case the constraint $c_i = [x_i \leq y_i + \delta]$ is added to Q_L.

It must be stressed that when a new query is issued the items presented in the previous stage are marked as "viewed" and the new items are shown to the user. Hence the user can always backtrack to a previous stage and accept one of the previous items or ask the system for additional alternatives.

4 Conclusions

In this short paper we have presented an integrated recommender system that provides support for pre-travel planning and on-tour travel plan update. The mobile component, which is under development, extends an Intelligent Travel Recommender system (ITR) aimed at supporting a user in information filtering and product bundling. In this paper we have shown how a travel plan built with ITR can be enriched searching for additional services exploiting an innovative approach to interaction management, query revision and personalization of recommendation on the mobile and how this compares to the approach used in ITR.

References

[1] 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.

[2] J. Delgado and R. Davidson. Knowledge bases and user profiling in travel and hospitality recommender systems. In *Proceedings of the ENTER 2002 Conference*, pages 1–16, Innsbruck, Austria, January 22-25 2002. Springer Verlag.

[3] T. Gaasterland, P. Godfrey, and J. Minker. An overview of cooperative answering. *Journal of Intelligent Information Systems*, 1(2):123–157, 1992.

[4] L. M. Ginty and B. Smyth. Deep dialogue vs casual conversation in recommender systems. In *Recommendation and Personalization in eCommerce, Proceedings of the AH'2002 Workshop*, pages 80–89, Malaga, Spain, May, 28th 2002. Universidad de Malaga.

[5] F. Ricci, B. Arslan, N. Mirzadeh, and A. Venturini. Itr: a case-based travel advisory system. In S. Craw, editor, *6th European Conference on Case Based Reasoning, ECCBR 2002*, Aberdeen, Scotland, 4 - 7 September 2002. Springer Verlag.

[6] F. Ricci, N. Mirzadeh, and A. Venturini. Intelligent query managment in a mediator architecture. In *IEEE International Symposium "Intelligent Systems'*, Sunny Day, Bulgaria, September 10-12 2002.

[7] H. Shimazu. Expertclerk: Navigating shoppers buying process with the combination of asking and proposing. In B. N. (Ed.), 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.