Exploiting a Map-based Interface in Conversational Recommender Systems for Mobile Travelers

Francesco Ricci
Free University of Bolzano, Italy

Quang Nhat Nguyen
Hanoi University of Technology, Vietnam

Olga Averjanova
Image Data Systems Ltd, UK

ABSTRACT

Nowadays travel and tourism Web sites store and offer a large volume of travel related information and services. Furthermore, this huge amount of information can be easily accessed using mobile devices, such as a phone with mobile Internet connection capability. However, this information can easily overwhelm a user because of the large number of information items to be shown and the limited screen size in the mobile device. Recommender systems (RSs) are often used in conjunction with Web tools to effectively help users in accessing this overwhelming amount of information. These recommender systems can support the user in making a decision even when specific knowledge necessary to autonomously evaluate the offerings is not available. Recommender systems cope with the information overload problem by providing a user with personalized recommendations (i.e., a well chosen selection of the items contained in the repository), adapting this selection to the user's needs and preferences in a particular usage context. In this chapter, the authors present a recommendation approach integrating a conversational preference acquisition technology based on "critiquing" with map visualization technologies to build a new map-based conversational mobile RS that can effectively and intuitively support travelers in finding their desired products and services. The results of the authors' real-user study show that integrating map-based visualization and critiquing-based interaction in mobile RSs improves the system's recommendation effectiveness, and increases the user satisfaction.

Keywords: recommender systems, location-based services, conversational systems, travel planning

INTRODUCTION

The users of travel and tourism web sites often find it difficult in choosing their desired travel products and services due to an overwhelming number of options to consider, and the lack of the system's help in making product selection decisions. The problem becomes even harder for users of the mobile Internet, who browse travel sites and their product repositories using a mobile device. This additional difficulty is

due to the intrinsic obstacles of the mobile usage environment, i.e., mobile devices have small screens and the data transfer rates of wireless networks are typically lower than those of wired ones.

Recommender Systems RSs are decision support tools aimed at addressing the information overload problem, providing product and service recommendations personalized to the user's needs and preferences at a particular request context (Resnick & Varian, 1997; Adomavicius & Tuzhilin, 2005). However, existing recommendation technologies have not been developed specifically for mobile users; and this chapter shows that recommendation techniques developed for the wired web must be adapted to the mobile environment in order to better exploit the available information, and provide software tools usable on mobile devices.

The evolution of mobile devices (e.g., PDAs and mobile phones), wireless communication technologies (e.g., wireless LAN and UMTS), and position detection techniques (e.g., RFID beacon-based and GPS), have created favorable conditions for the development and commercialization of a large number of location-based mobile services (Mohapatra & Suma, 2005; Steinfield, 2004), i.e., information services accessible by mobile devices through the mobile network, and utilizing the geographical position of the mobile device. As a consequence, many location-based mobile services have been introduced in the recent years, including emergency services, information services, navigation support services, etc.

For example, mobile travelers can access local tourist information services providing information about nearby points of interests (Pospischil et al., 2002), such as pubs and restaurants (Dunlop et al., 2004), or get routing guidance from their position to a target location (Pospischil et al., 2002). In many of these systems, maps and map-based interfaces are used to visualize points of interests (e.g., restaurants, museums, or hotels), their spatial relations, and various kinds of information related to these points (e.g., menus, opening hours, or in-room services).

However, map-based interfaces do not solve all the information access problems. In fact, a major problem in map-based visualization is the need to keep the display readable and free from irrelevant information. This is particularly true in the mobile usage environment. Because of the limitations of mobile devices, especially small screens and limited computing power, displaying on an electronic map a large number of objects and their related information is computationally expensive and usually not effective. Hence, systems providing mobile travel services should employ filtering mechanisms to reduce the amount of data (information objects) that is displayed on an electronic map.

Though the specific benefits of map-based interfaces and recommendation technologies have been demonstrated in a number of previous research projects, the integration of these two technologies and their empirical evaluation, in terms of usability and effectiveness, in mobile travel support systems have not been studied yet. In this chapter, we present an approach for integrating recommendation technologies and electronic map visualization technologies to build a map-supported travel RS that can effectively and intuitively provide personalized recommendations on mobile devices.

Our recommendation methodology integrates long-term and session-specific user preferences, uses a composite query representation, employs a case-based model of the recommendation problem and its solution, furthermore, it exploits a critique-based conversational approach (Ricci & Nguyen, 2007). The integration of long-term and session-specific user preferences enables the exploitation of multiple knowledge sources of user’s data. The long-term user preferences are derived (inferred) from past recommendation sessions, and on the other hand, the session-specific user preferences are collected through the user’s explicit input including the critiques made in the current session.

The composite query representation consists of a logical query and a favorite pattern. This allows using both strict logical constraints and weak similarity-based conditions. The logical query component helps

the system to precisely focus on the most relevant subsets of the product space, whereas the favorite pattern component enables the system to correctly sort the relevant products, ranking higher the products that are most suitable for their needs and wants.

In our approach, a travel product recommendation session is modeled as a case, and the Case-Based Reasoning (CBR) problem solving strategy is used (Aamodt & Plaza, 1994). CBR is based on learning from previous experiences, and case-based RSs exploit (reuse) the knowledge contained in a set of past recommendation cases. In our recommendation methodology, CBR is used for building a personalized user-query augmenting (by specializing) the original query derived from the conditions explicitly entered by the user. This personalized query is adapted in such a way that, taking into account the knowledge contained in the case base, the set of recommended products better match the user's preferences, even if these preferences are not expressed in the original query entered by the user.

Furthermore, our recommendation approach is based on the interactive elicitation of the user's preferences through critiques. The user, instead of being required to formulate a precise search query at the beginning of the interaction, is involved in a dialogue where the system's product suggestions interleave with the user's critiques to the recommended products. A user's critique is a comment (judgment) on a displayed product, which points out an unsatisfied preference (e.g., "I would like a cheaper restaurant") or confirms the importance of a product feature for the user ("I'd like to have dinner in a restaurant with a garden terrace"). This critique-based user preferences elicitation procedure results in the system building a better understanding the user's needs and preferences, and hence, in constantly improving the recommendations.

Our recommendation methodology has been firstly implemented in MobyRek (Ricci & Nguyen, 2007), a mobile RS that supports travelers in finding their desired travel products (restaurants). The results of some previous empirical evaluations of MobyRek, consisting of a live-user test (Ricci & Nguyen, 2007) and a number of simulations (Nguyen & Ricci, 2007; 2008a), showed that our critique-based recommendation methodology is effective in supporting mobile travelers in product selection decisions. However, MobyRek employs a text-based interface for recommendation visualization and system-user interaction, where the recommendations are presented to the user in a ranked list, as in a standard search engines like Google. We conjectured that this visualization and interaction approach may not be optimal for mobile devices and for mobile recommendation problems. We believed that this type of "standard" interface used for web applications can, in fact, cause some difficulties and inconveniences for mobile users in their interactions with the system.

Hence, in this chapter, we analyze the usability limitations of MobyRek, which we believe are typically found in all list-styled RSs, and we illustrate how this analysis leads to the design of an extended and improved version of the system called MapMobyRek. MapMobyRek implements the same core recommendation techniques used in MobyRek, i.e., the computation of the recommended items and their ranking are done exactly as in MobyRek. However, MapMobyRek uses maps as the main user interface for information display and access; furthermore, it provides new decision-support functions based on the map. The design of the map-based interface of MapMobyRek focused on the ability to offer the following user functions:

- to enter the search query by specifying preferences for item features,
- to see the system's recommendations on the map,
- to recognize immediately the differences between good and weak recommendations,
- to compare two selected recommendations,
- to input critiques to the recommended items,
- to see on the map how the expressed critique influences the system's recommendations, and
- to select the best item(s).

MobyRek and MapMobyRek are then compared, through a real-user test, with respect to functionality, efficiency, and convenience. The objective measures taken during the test and the subjective evaluations of the real-users show that the map-based interface is more effective than the list-based one. We also find that the integration of a map-based interface in a RS results in increased user satisfaction.

The remainder of this chapter is organized as follows. In the next section, we survey some related work. Then, we present our recommendation methodology, we discuss the limitations of MobyRek's user interface, and we present MapMobyRek – the improved system. Next, we state our research hypotheses, describe the test procedure, and present and discuss the test results. Finally, we give our conclusions and indicate some avenues for future research. We note that this chapter extends (Averjanova et al., 2008) that more briefly describes MapMobyRek and its validation.

RELATED WORK

There have been several research works focusing on map-based mobile services (Meng et al., 2008), map-based mobile travel guides (Baus et al., 2005), recommender systems (Adomavicius & Tuzhilin, 2005; Burke, 2007), and travel RSs (Ricci, 2002; Fesenmaier et al., 2006). In this section, we first discuss traditional recommendation techniques, and then we focus on related works that aim to integrate mobile computing, map visualization and recommendation technologies in providing personalized travel products and services to tourists.

Recommendation Techniques

Many recommendation methods have been introduced in the RSs literature. Nevertheless, they are often classified into four well-known categories: collaborative, content-based, knowledge-based, and hybrid (Adomavicius & Tuzhilin, 2005; Burke, 2007).

Collaborative RSs generate recommendations using the information of the ratings given by users on items. In the collaborative recommendation approach, the system recommends to a given user those items that have been highly rated by the other users who have similar taste (i.e., similar ratings for co-rated items). The collaborative recommendation approach is inspired by the daily habit of people, i.e., when finding information or choosing between options, people often consult friends who have similar likes and tastes.

Content-based RSs generate recommendations for a given user exploiting feature-based descriptions of items and the ratings that the user has given on some items. In the content-based recommendation approach, a user is modeled by a profile that represents the user's needs and preferences with respect to item's features; and the system recommends to the user those items whose features (highly) match the user's profile.

The *knowledge-based* recommendation approach uses specific domain knowledge to reason on the relationship (i.e., suitability/appropriateness) between a user's needs and preferences and a particular item. In knowledge-based RSs, the system acquires the user's requirements on items (e.g., the user's query), and then consults the knowledge base to determine the best fitting items.

Each of the three recommendation approaches (i.e., collaborative, content-based, and knowledge-based) has its own limitations and disadvantages (Adomavicius & Tuzhilin, 2005; Burke, 2007). Therefore, some RSs take a *hybrid* recommendation approach that combines two (or more) recommendation techniques in order to take full advantages of each individual technique, or to overcome some of their disadvantages

(Burke, 2007). With respect to this categorization of recommendation methods, our recommendation approach can be considered a knowledge-based approach.

Many RSs, such as those based on collaborative filtering, follow the *single-shot* recommendation strategy, i.e., for a given user's request the system computes and shows to the user the recommendation list, and the session ends. If the user is not satisfied she could enter a new query, if possible, but this process is up to the user and the system does not provide any support through this sequence of requests.

Conversely, in *conversational RSs* a recommendation session does not terminate immediately after the first set of recommendations are shown to the user, but it evolves in a dialogue where the system tries to elicit step-by-step the user's needs and preferences to produce better recommendations (Bridge et al., 2005). In our approach, the system-user conversation is supported through critiquing, where the system's recommendations are interleaved with the user's critiques to the recommended items (Burke, 2002; McGinty & Smyth, 2006).

Mobile Recommender Systems

The systems and techniques described above have been mostly applied in the Web usage context, without any special concern with the context of the user, i.e., in particular, if she is on the go and accesses the information service using a mobile device. In the rest of this section we will review some personalized travel and tourism support systems that have been specifically designed for the mobile usage context.

CityGuide (Dunlop et al., 2004) is designed for PalmOS devices and helps tourists in finding attractions (such as restaurants) around a city. This recommender system uses the constraint-based filtering approach to control which attractions are shown on the map. In particular, the user, through the system's map interface, is asked to specify constraints on attraction type, restaurant cuisine and price. The system retrieves from the database only those attractions that satisfy the user's indicated constraints, and then ranks these retrieved attractions according to their match to the preferences stored in the user's profile. The system builds and updates the user's profile, which maintains her long-term preferences, by mining and interpreting the user's actions (such as writing a restaurant review, reading a review, viewing a restaurant's details, etc.) and collecting the user's ratings to the restaurants. Though the system ranks the recommended attractions and shows them on the map, it does not visualize how close each recommended attraction is to the user's preferences (i.e., the recommendation level).

Burigat et al. (2005) illustrate a system running on PDA devices that supports tourists in searching for travel products (i.e., hotels or restaurants) in a geographic area that best satisfy their needs and preferences. The system builds the user-query used to search in the services repository by asking the user to indicate her constraints on the service attributes, e.g. the facilities offered by the hotel or the restaurant. However, the system does not employ the constraint-filtering approach or a multi-attribute utility function. Instead, the system constructs the recommendation list by ranking the services according to their satisfaction score. A service's satisfaction score is measured by the number of constraints (indicated in the user's query) that are satisfied by the service. Hence, each recommended service is visualized by an icon superimposed on the map of the geographic area, augmented by a "filled-in" vertical bar representing how much the service satisfies the user's query. We observe that this system does not reuse the knowledge derived from past user interactions to provide better recommendations.

Dynamic Tour Guide (*DTG*) (ten Hagen et al., 2005) is a mobile tour guide system that helps travelers in discovering a destination. Given a user's request, the system computes a personalized tour in the city by asking the user her interests and the time she would like to spend for the visit. In *DTG* a tour is composed of a set of points of interest (i.e., restaurants, attractions and events). The system then presents the recommended tour on the map and provides some audio guide information. This audio visual presentation lets the user to visualize the tour itinerary. The system computes and presents just one tour per user

request, instead of providing a list of recommended tours. *DTG* also utilizes the content-based approach for building the recommendations (Burke, 2007), just as the previous one.

The *George Square* system (Brown et al., 2005) runs on handheld tablet PCs. It is designed for supporting visitors to explore a city as well as share their visit experiences. As the user moves in the city and visits the attractions, her positions and activities are automatically recorded (logged) by the system. The user's activities include the attractions she encounters, the web pages (i.e., weblogs created by previous visitors) she browses, and the photographs she takes. To build the recommendation list, the system makes use of patterns of co-occurrence of positions and activities, and employs the collaborative filtering recommendation technology (Adomavicius & Tuzhilin, 2005). In particular, the system uses the user's recent activities (i.e., in the last few minutes) to define the user's current context, then finds in the previous visitors' log data those periods of time with similar contexts, and finally includes in the recommendation list the activities done in those periods of time by these visitors. In addition, when showing the recommended items (i.e., attractions, web pages, photographs) on the map, the system displays the positions of nearby visitors and supports their leisure collaboration. For instance, the system let the user to collaboratively producing pictures of the attractions, or to have (voice-over-IP) discussions on co-interested attractions and objects.

Mobile Piste Map (Dunlop et al., 2007) is designed to support mobile users in selecting ski routes, i.e., individual or combined runs appropriate to their level of ability, preferences, and current mood and track conditions. The system first asks the user to indicate her ski-run preferences, for instance the route difficulty level, and then uses the indicated preferences to compute a list of recommended routes. The system presents on the map the recommended routes and their suitability for the user.

Park et al. (2007) presents a restaurant RS for mobile users. The system computes personalized recommendations using a Bayesian Network (Pourret et al., 2008) that models the probabilistic influences of the input parameters, i.e., the user's personal and contextual information, in relation to the restaurant attributes. This approach represents a restaurant by three discrete-valued attributes: class (e.g., Korean or Italian restaurant), price (e.g., low or medium), and mood (e.g., romantic or tidy). The Bayesian Network is defined by an expert, and is trained using a training dataset. To begin with, each user is required to provide personal information (e.g., age, gender, income, preferred restaurant class, etc.). The user's contextual information is automatically collected (detected) by the system, including season (e.g., spring), time in day (e.g., breakfast), position, weather (e.g., sunny), and temperature (e.g., warm). When a user requests for a restaurant recommendation, the system computes the recommendation score for all the restaurants in the database. A restaurant's recommendation score is a weighted sum of the conditional probabilities of the restaurant's attribute values, where these conditional probabilities are derived (inferred) from the learned Bayesian network. The system presents on the map a small set of the restaurants, i.e., those achieving the highest recommendation scores.

Research Goals

The main goal of our research work is similar to that addressed by the papers presented in the previous section, i.e., to provide personalized suggestions for travel products and services to mobile travelers, and to use the map as primary tool for visualizing these suggestions. However, our approach differs in the following aspects.

- In our approach, the recommendation process begins with the user specifying explicit preferences, however, this user input is optional in our system, i.e., the user can obtain some recommendations even without explicitly providing any initial input. The main rationale for this design solution is that the user's preferences can be also acquired during the product search interaction phase, using the critiquing function (discussed in details in the next section) and exploiting the knowledge derived from the previous recommendations. We note that in all the referenced approaches either it is mandatory for the user to input initial preferences (Dunlop et

al., 2004; Burigat et al., 2005; ten Hagen et al., 2005; Dunlop et al., 2007) or the user is not even given the option to specify initial preferences (Brown et al., 2005; Park et al., 2007). In this respect, our approach is more flexible, since it give the option to the user to specify preferences if she wants.

- In most of the systems described earlier, the user is not supported in providing feedback to the system's recommendations; or if supported, as in *CityGuide* (Dunlop et al., 2004), such a feedback is exploited only in future recommendation sessions, but not immediately for personalizing the current one. In our approach, various types of feedback are supported, i.e., critiquing the recommended items and rating the selected product(s). Moreover, critiquing is used immediately to refine the current user query.
- Our approach allows the users to indicate their preferences at different levels, in particular as a wish or a must satisfy criteria, and it utilizes these preferences in computing the system's recommendation(s). None of the earlier mentioned approaches support such a real-time use of user's preferences and critiques.
- Furthermore, none of the systems described earlier support the users in comparing two recommended items. In fact, given the system's recommended items, a user may first focus her interest on a few items, and then she may compare pairs of items by browsing their characteristics (see Figure 5, presented later on). Hence, our approach supports an item-to-item comparison, highlighting the differences between the two compared items, and hence provides additional support to users in selecting the best item.
- In our approach, case-based reasoning and user-interaction mining methodologies are used to exploit the knowledge contained in the past recommendation cases for learning how to rank the recommendations. Of the systems discussed earlier, only *CityGuide* (Dunlop et al., 2004) and *George Square* (Brown et al., 2005) learn from past recommendation interactions; whereas, *CityGuide* exploits the users' ratings and implicit feedback, and *George Square* exploits the users' past activities.

RECOMMENDATION METHODOLOGY

In this section we summarize our proposed methodology for travel product recommendation in a mobile context, providing details about the product representation and the user preferences model. Moreover, we illustrate how the user can critique a product recommendation and how this action is interpreted and exploited by the system. A more detailed description of the recommendation methodology is provided by Ricci & Nguyen (2007).

We observe that both MobyRek -- the system using text-based display of recommended items, and MapMobyRek -- the system using a map-based interface, identify the recommendation set, i.e., the restaurants to suggest to the user, in the same way. The two systems differ only in the way they present the results to the user, i.e., in their user interface.

Product Representation and User Preferences Model

In our recommendation technology, a travel product is represented as a feature values vector $x = (x_1, x_2, \ldots, x_n)$, where a feature value x_i may be numeric, nominal, or a set of nominal values. For instance, the restaurant x= ("Dolomiti", 1713, {pizza}, 10, {air-conditioned, smoking-room}, {Saturday, Sunday}, {credit-card}) has: name x_1="Dolomiti", distance from the user's position x_2=1713 meters, type x_3={pizza}, average cost x_4=10 Euros, characteristics x_5={air-conditioned, smoking-room}, opening days x_6={Saturday, Sunday}, and payment method x_7={credit-card}.

Any recommender system, for generating personalized recommendations, requires a representation of the user's preferences. Preferences vary from user to user, and even the same user in different situations (contexts) may have different preferences. In our approach, the user preferences model, which is encoded in a product search query, includes both contextual preferences (such as the space-time constraints) and preferences on product features. Contextual preferences characterize a user's current context of request, whereas product preferences express a user's taste and like. In the restaurant recommendation problem, for example, space-time constraints guarantee that the recommended restaurants are open on the day of the request and not too far from the user's position. Preferences on product features may state that the user, for instance, is interested in cheap restaurants, or prefers to go for pizza.

The user preferences model incorporates both long-term and session-specific user preferences (Nguyen & Ricci, 2008a; Ricci & Nguyen, 2007). Long-term (stable) user preferences, such as a preference for non-smoking room, are inferred from the user's recent interaction sessions with the system, and remain true throughout these sessions. In contrast, session-specific user preferences, such as a wish to eat pizza, are transient and specific to a session. Acquiring session-specific preferences helps the system to effectively capture the transient needs of the user, whereas exploiting long-term preferences allows the system to rely on more stable preferences, or default choices, and also avoids interrupting the user or requiring her effort in specifying preferences when these are not needed.

Session-specific and long-term preferences must be combined and used in the most appropriate way. In our approach, the system acquires session-specific user preferences in two ways: by user's initial specifications and by user's critiques. The user's initial preferences explicitly specified at the beginning of the recommendation session help orient the system's initial search. Meanwhile, the user's critiques to the recommended products help the system refine its understanding of the user's needs and preferences. As noted earlier the initial preference specification is optional in our system.

In a recommendation session, the user query has the function of encoding the user's preferences (both session-specific and long-term), and is used by the system to compute the recommendation list (more on this subject will be provided later). In our model, the user-query q consists of three components, $q = (q_l, p, w)$.

- The *logical query* (q_l) models the conditions that must be satisfied by the recommended products. The logical query is a conjunction of constraints: $q_l = (c_1 \wedge c_2 \wedge \ldots \wedge c_m)$, where c_j is a constraint on j-th feature. A constraint deals with only one feature, and a feature appears in only one constraint.
- The *favorite pattern* (p) models the conditions that the recommended products should match as much as possible. The wish conditions allow the system to make trade-offs. The favorite pattern is represented in the same vector space of the product representation: $p = (p_1, p_2, \ldots, p_n)$, where x_i and p_i belong to the same feature type, $\forall i = 1..n$.
- The *feature importance weights vector* (w) models how much each feature is important for the user with respect to the others: $w = (w_1, w_2, \ldots, w_n)$, where w_i ($\in [0,1]$) is the importance weight of i-th feature. The system refers to the feature importance weights when it needs to rank the products or make trade-offs or to find relaxation solutions for over-constrained logical queries.

For instance, the query $q = (q_l = (x_2 \leq 2000) \wedge (x_6 \supseteq \{\text{Saturday, Sunday}\}))$, $p = (?, ?, \{\text{pizza}\}, ?, ?, ?, ?)$, $w = (0, 0.4, 0.2, 0, 0, 0.4, 0))$ models a user interested only in those restaurants within 2 km from her position, open on Saturday and Sunday, and preferring pizza restaurants to the others. The user considers the distance and the opening day as the most important features and then the restaurant type, and she is indifferent to the other features.

Recommendation Process

In our approach, the overall model of a recommendation session is logically divided into three phases: 1) initialization, 2) interaction and adaptation, and 3) retaining, as shown in Figure 1. Given a user's request

for a product recommendation, the system builds the initial representation of the user's query. Then, the system finds the products most suitable to the initial query, and recommends them to the user. Given these recommendations the user can browse and evaluate them. Then the user may critique the recommended products, e.g., "This product is too expensive for me". This user's evaluation helps the system to adapt (update) the current understanding of the user's needs and preferences encoded in the user-query representation. These two steps, i.e., the system's recommendation and the user's critiquing, iterate until when the user is satisfied with one of the recommended products or when she terminates the session with no product selection. At the end of the session, the system records the current recommendation session for future exploitations.

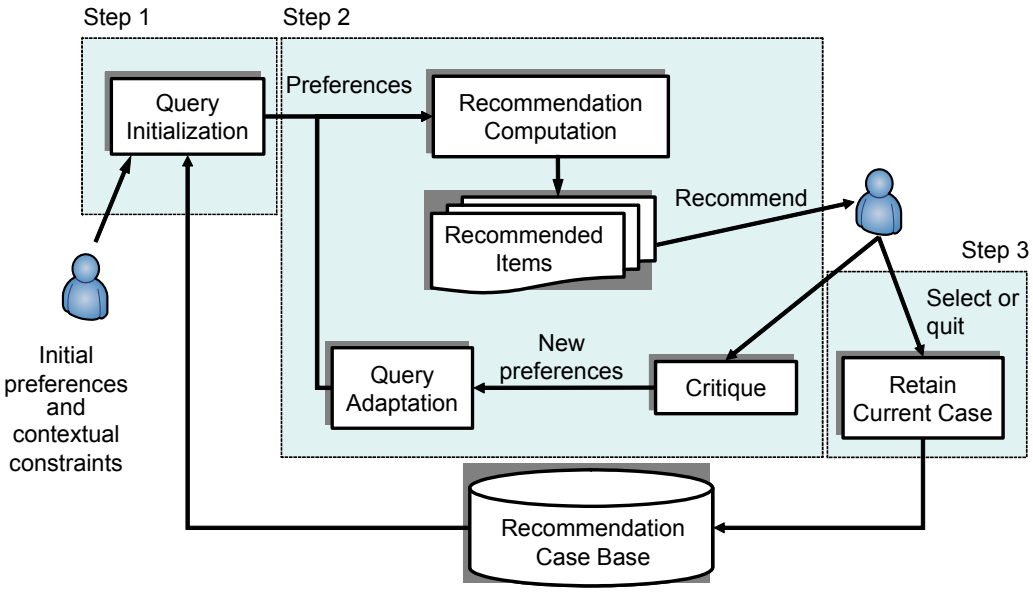
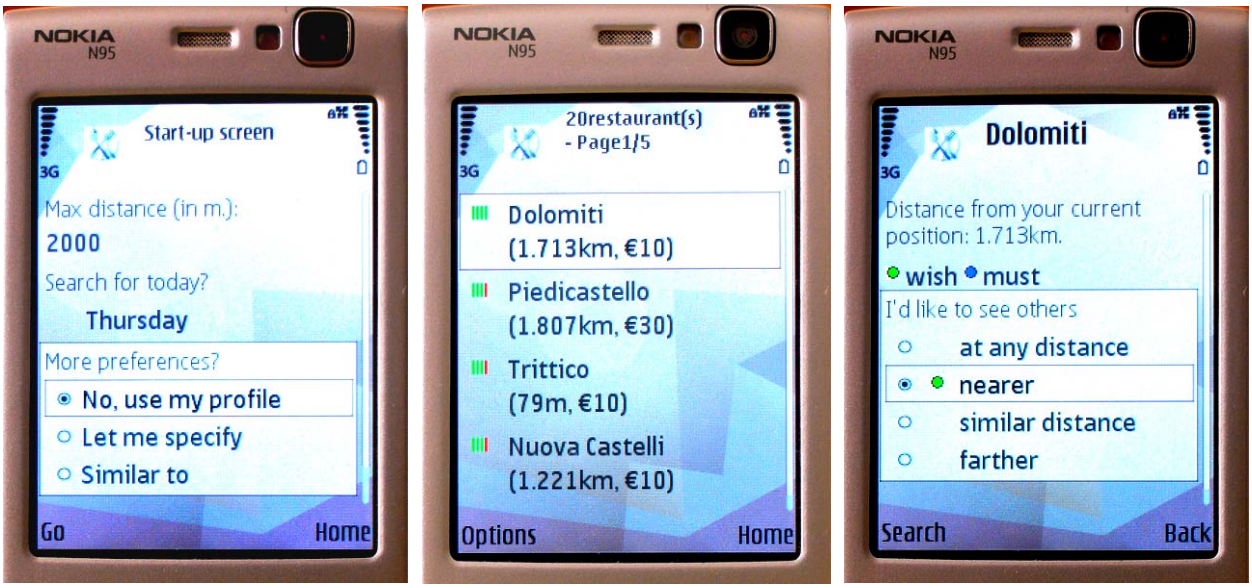


Figure 1. The recommendation process.

Hence, a recommendation session starts when a user asks for a product suggestion (Figure 2a) and ends when the user selects a product or when she quits the system. A recommendation session evolves in cycles. In a recommendation cycle, MobyRek shows a ranked list of recommended products (Figure 2b) that the user can browse and criticize (Figure 2c), and the cycle ends when a new recommendation list is requested and shown.

At the beginning of the recommendation session, MobyRek and MapMobyRek offer the user three options for search initialization (Figure 2a).

- "*No, use my profile*". To let the system automatically construct the initial search query.
- "*Let me specify*". To let the user explicitly specify some initial preferences.
- "*Similar to*". To let the user specify a known product, which the user has consumed previously, as the starting point of the system's search.



(a) *(b)* *(c)*

Figure 2. MobyRek user interface.
(a)Search initialization options; (b) Recommendation list; (c) User critique on a numeric feature.

Once the user selects an initialization option, the system builds the initial query (q^0) exploiting multiple knowledge sources of users-related data, including a) the current user's space-time information, b) the current user's previous products' selections, c) the past selections of other users of the system, and d) the current user's initial preferences, explicitly specified at the beginning of the interaction with the system. Basically, the initial logical query (q_l^0) is built based on the user's space-time constraints and initial preferences stated as a must-have option. The initial favorite pattern (p^0) is built by integrating the user's long-term preferences pattern, the characteristics of the product that was selected in a past case that is most similar to the current case, and the initial preferences of the current user stated as a wish-have conditions. The initial feature weights vector (w^0) is built by exploiting data collected in previous user's interactions with the system. The details of the query representation initialization are discussed in (Nguyen & Ricci, 2004; Ricci & Nguyen, 2007).

The initial query (q^0) is then used by the system to compute the first recommendation list. In the computation of the recommendation list, the system first filters out those products that do not satisfy the logical query (q_l), and then ranks the remaining products according to their similarity to the (p, w), i.e., the favorite pattern and the feature weights vector, respectively. The ranking is done so that the more similar to (p, w) a product is the higher that product appears in the ranked list of MobyRek. A product's similarity score is computed as described in Equation (1). It is 1, i.e., the maximum value for similarity, minus the generalized weighted Euclidean distance, $d(x,p,w)$, between the product x and the favorite pattern p, using the weights in the vector w:

$$sim(x, p, w) = 1 - d(x, p, w) = 1 - \frac{\sum_{i=1}^{n} w_i d(x_i, p_i)}{\sum_{i=1}^{n} w_i}; \quad (1)$$

where $d(x_i, p_i)$ is the distance (dissimilarity) between product x and the favorite pattern p with respect to feature f_i, and w_i is the importance weight of feature f_i.

In the computation of the recommendation list, if no products in the catalogue satisfy the logical query (q_l), then the system attempts to discard the minimum number of constraints for making the query satisfiable, i.e., such that the new relaxed query returns a non-empty set of items. In this procedure, which is a kind of constraint relaxation, a constraint involving a less important feature is eliminated before

another involving a more important one. The relaxed constraints are then converted to wish conditions and incorporated in the favorite pattern p.

When MobyRek shows the recommendation list to the user (Figure 2b), for each recommended product the system also displays the product's basic information (i.e., name, average cost, and distance) together with an icon (on the left of the restaurant's name) that provides a hint about how close the product matches the user's preferences. Looking at the recommendation list, the user could quickly understand the "appropriateness" of the recommended product. Moreover, to inform the user of the number of available options matching her preferences, the system displays this number on top of the screen . In MobyRek the recommendation list is divided into "pages" so that the number of products displayed on one page fits the mobile device's screen size.

Given the recommendation list shown to the user (Figure 2b), if she is satisfied with a recommended product, the user can select and save it for future reference, and the session ends successfully. However, if the user is somewhat interested in, but not completely satisfied with, a recommended product, then she can criticize that product to express her additional preferences on the unsatisfactory feature values (Figure 2c) or to indicate that some product feature is more important for her (e.g., "I like to have lunch in an air-conditioned room"). When making a critique, the user is supported by the GUI to express the strength of the preference implied in that critique, i.e., as a must or a wish condition. This helps the system to correctly exploit the user's critique, i.e., whether to focus on a certain part of the product repository (by using the must conditions) or to refine the products' ranking (by using the wish conditions). In particular, a critique stated as a must is incorporated in q_l, whereas a critique stated as a wish is incorporated in p. Hence, the user's critiques help the system to adapt the query representation, i.e., refining its previous guess about the user's preferences. Then, the adapted query representation is used by the system to produce a new recommendation list.

Eliciting user preferences through critiques has some advantages. Firstly, the preferences are explicitly stated by the user, and hence, are much more reliable than those implicitly collected, for instance, by mining the user's interaction behavior. Secondly, the user effort to make a critique is low, as compared to other methods using interviews or early ratings. In practice, a critique to a recommended product is done simply by a few button clicks (Figure 2c). Finally, compared to the explicit request to specify preferences, the request to criticize a real product is more convincing, because in the latter approach the system first provides some immediate benefit to the user showing some recommendations and motivating the user to further reveal her preferences, and only after that it requires the user to provide additional input (critique). All these considerations convinced us to conjecture that critiquing could be a viable and preferred approach to collect user preferences in a mobile application.

We note that our critiquing approach is different in several aspects from others that were developed primarily for web applications. For instance, in the notable approaches used by Burke(2002); and McGinty & Smyth (2006) the criticized item defines the new user-query, i.e., at the next interaction the items closer in similarity to the criticized item are retrieved, and the user's critique on a specific feature is used as a constraint to reduce the number of items retrieved. Conversely in our approach the query is modified incorporating the user critique and the selected item is just used to elicit the user preference encoded in a critique. The rationale of our choice, which departs from these more common approaches, is based on the peculiar characteristics of the mobile usage environment. In fact, in these approaches, by transferring all the characteristics of the criticized item into the user query, it is implicitly assumed, or required, that the user has looked at many of the recommended items, and all of the features of an item, before making a critique. This is very difficult to achieve when using a mobile device, where, because of the small screen, one can have only a partial view of the critiqued item, and can focus only on a small part of the item description, namely just the critiqued feature. Finally, in our approach, when making a critique, the user is supported to indicate the preference strength (i.e., wish or must) of the critique; and this functionality is not provided by other critique-based RSs.

When a recommendation session finishes, it is retained as a case in the system's case base for future references (Aamodt & Plaza, 1994). In this way, the past recommendation sessions can be exploited by the system in making new recommendations for the future users as described by Nguyen & Ricci (2008b).

VISUALIZATION AND INTERACTION ISSUES IN MOBYREK'S USER INTERFACE

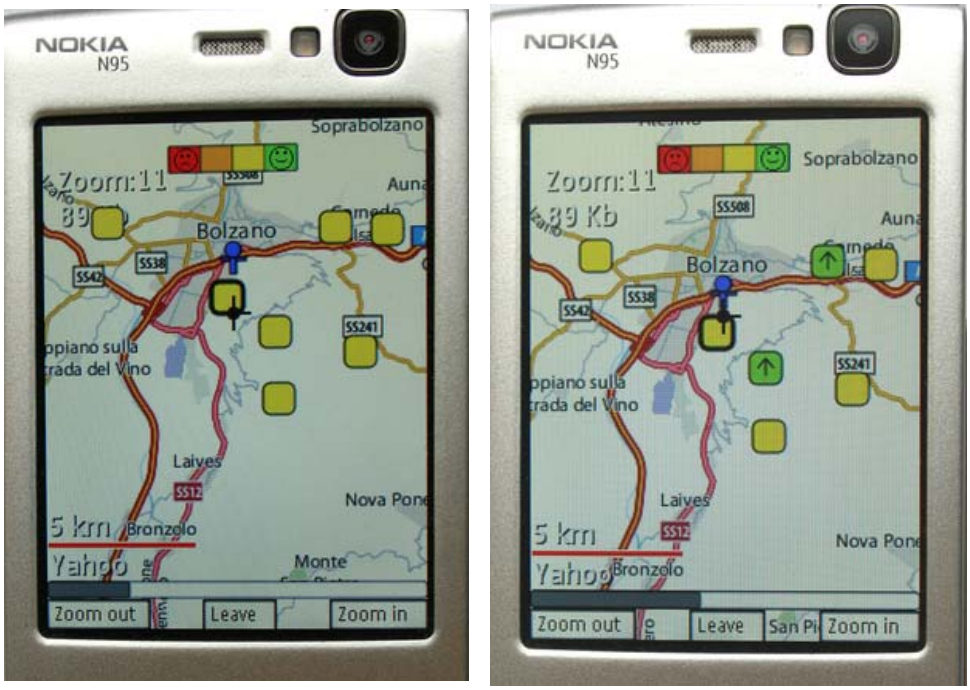
The proposed recommendation methodology has been implemented in MobyRek, a restaurant RS for mobile travelers. A live-user evaluation of MobyRek showed that it can effectively support mobile users in finding their desired travel products (Ricci & Nguyen, 2007).

However, that live-user evaluation, and some further analysis pointed out the following limitations of the MobyRek's user interface.

- *User perception of the influence of a critique on the new recommendation list*. After making a critique, it might be difficult for the user to get an immediate feedback of 1) how the expressed critique has influenced (changed) the ranking of the recommended items, or 2) the inclusion of new recommendations, or 3) the elimination of items that were previously recommended. To determine which items lose/gain higher/lower recommendation score, and consequently their rank, in MobyRek the user must remember the previous recommendation list (i.e., that produced in the previous cycle). This is clearly quite difficult, and the result is that users feel unsure about the changes produced by their input and the system looses transparency.
- *Clear perception of the restaurant's distance from the user's position*. When a user is on the go, it is very important to illustrate how far the recommended venues are from the user's current position. In MobyRek, it is difficult for the user to get an immediate understanding of the position of the recommended restaurants. In fact, the user, in order to evaluate this aspect, must scroll down the recommendation list and compare the recommended items' distances to her position, as it is displayed close to the name of the restaurant (Figure 2b).
- *Inconvenience and difficulties in accessing an item's description*. In MobyRek, a user may find it inconvenient to access an item appearing in a long recommendation list. For instance, if a user is interested in a recommended item and, after browsing other recommendations, decides to come back to see that item, she must remember its position in the list, or its name.
- *Items comparison*. Given a list of recommended items, a user may be uncertain of the choice between two or more items. In such a situation, a system function providing item-to-item comparison could help the user in identifying the most appropriate items. This useful function, however, is not supported by MobyRek.

To overcome these limitations and increase the usability of MobyRek, we developed an extended system called MapMobyRek that provides the following new features.

- *Map-based visualization of the recommendations*. In MapMobyRek, the recommended items are shown as objects placed on a map (Figures 3a, 4a). The map interface supports typical electronic map features, such as zooming in/out or panning up/down/left/right. On the map, a user can easily perceive the recommended items' relative distance from the user's position, and can easily access any of the recommended items.
- *Color-encoded representation of the item's recommendation level*. Instead of showing in a ranked list the recommended items, as in MobyRek, MapMobyRek uses a range of colors to show the predicted degree of suitability of the recommended items. The colors range is red-orange-yellow-green, where the best recommendations are shown in green and the worst in red. In other words, the ranked list produced by the recommendation algorithm is divided into four parts, where the highest ranking items are encoded in green, those ranked lower are encoded in yellow, and so on. Smiley icons, shown on the top of the mobile screen, help the user understand the meaning of these colors (Figures 3a, 4a). We believe that these visual clues of the recommendation score are easier to grasp, and provides better support for filtering items; the user can easily focus on the green-colored (and yellow-colored) items if she wants to rely on the filtering provided by the recommendation algorithm.



(a) *(b)*

Figure 3. MapMobyRek user interface.
(a) Visualization of the recommended items; (b) Recommendation-level change after a wish critique.

- *Visualization of the influence of a critique on the next recommendations*. As discussed in the previous section, in our recommendation methodology a critique stated as a must condition changes the recommendation list, i.e., some new items may be recommended whereas some previously recommended items may be removed. On the contrary, a critique stated as a wish condition only changes the ranking of the recommended items. In MapMobyRek, the influence of the user critique is visualized immediately and intuitively on the map.

  In fact, after the user has made a wish critique, MapMobyRek gradually changes the colors of the icons on the map (from red to green, or from green to red) and draws an arrow (upward or downward) on each recommended item to show the change (increase or decrease) of its recommendation level (Figure 3b). Moreover, the system displays a progress bar showing the progress of display update (above the wish critique).

  After the user has expressed a must critique, some previously recommended items may not satisfy the new condition, and hence these should be removed from the map. MapMobyRek shows this by using an animation that gradually reduces the size of the icons of the removed items until they disappear. Similarly, MapMobyRek attracts the attention of the user on the new recommended items by gradually increasing their icon size up to the final standard dimension. When updating the display upon a must critique, the system displays a progress bar showing the progress of the display update.

(a) *(b)*

Figure 4. MapMobyRek user interface.
(a) Visualization of the recommended items; (b) Recommendation-level change after a must critique.

- *Items comparison functionality*. During the interaction with MapMobyRek a user can compare two interesting items. After she has selected the two items, their characteristics are displayed side-by-side on the screen; so their advantages and disadvantages can be easily compared (Figure 5).

As described earlier, MobyRek and MapMobyRek implement the same recommendation methodology and ranking algorithm; however, they utilize different user interfaces. This is important because any measured difference in the users' evaluation must be tracked back to the changes in the graphical user interface (i.e., list-based vs. map-based).

To implement map visualization in MapMobyRek we used J2meMap (a freeware library for Java MIDP applications, available on http://j2memap.landspurg.net/) that allows the management and display of map-related content. For displaying graphical objects as an overlay on the map, we exploited the J2ME SVG (Scalable Vector Graphics) library for drawing various geometrical shapes and text objects.

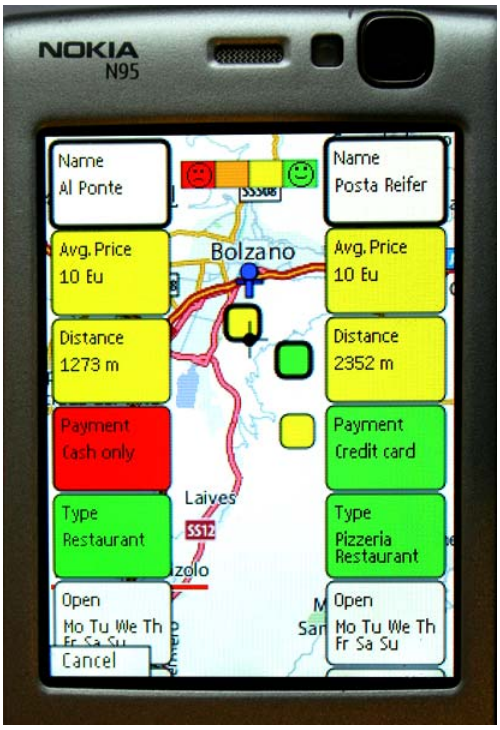


Figure 5. Items comparison functionality.

EMPIRICAL EVALUATION

In this section, we present the live-user evaluation of MapMobyRek that we conducted in February, 2008. This test was aimed at comparing MobyRek and MapMobyRek with respect to the recommendation effectiveness and the system usability. We first state the research hypotheses and the test procedure, and then we present and discuss the test results.

The Research Hypotheses and the Test Procedure

Based on the features incorporated in MapMobyRek we hypothesize that MapMobyRek, as compared to MobyRek, provides the following benefits:

- increased system usability and user satisfaction,
- reduced interaction length, i.e., the number of recommendation cycles produced by user critiques,
- reduced the time to complete the user task, i.e., finding a suitable product in a particular context of usage, and
- MapMobyRek users tend to rely more on the recommendation score computed by the system and choose the items with higher score, i.e., with a green color.These are the top ranked items as computed by the recommendation algorithm ((Ricci & Nguyen, 2007).

To validate our research hypotheses we ran a user study, which involved twenty testers, using a mobile phone (Nokia N95) with both MobyRek and MapMobyRek installed. 18 testers were male and 2 female; most of them were aged below 30 with only 3 testers above that age. In this experiment each tester was asked to use both systems, MobyRek and MapMobyRek, and to express her subjective opinions. Each tester used both systems, one after the other, to find her desired restaurant and add it to her travel notes. The travel notes are a container for the selected travel products, services, and information that the user

can easily access and consult. The order in which the two systems were used was assigned randomly to compensate any learning effect. In fact, it could be easily guessed that the users will perform the task better with the latter system, as they will get more familiar with the product domain and the task itself.

In the comparative evaluation of the two systems, we collected some objective and subjective measures. The objective measures consisted of:

- **Average interaction length**. This is the average number of recommendation cycles, where a cycle is defined by the application of a critique. This is an important metric to consider, since if the two systems give the same user satisfaction, one must prefer the system that enables to complete the task quicker.
- **Average time required to complete the task**. This is the elapsed time of the user-system interaction. The motivation for considering this metric is similar to that mentioned above for the interaction length metric.
- **Average position of the selected restaurant in the ranked list**. The color encoding the recommendation score of a recommended restaurant is computed by considering its position in the ranked list computed by the common recommendation algorithm. In order to check if the user actually based her item selection on the system's suggestions, we looked at the position of the selected item in the ranked list. The higher the position of that item, the more likely that the user will make a decision that is influenced by the recommender system.

In addition to these objective measures, we obtained the subjective user evaluations of the two systems. The tester, after she has used each system, was asked to complete a usability questionnaire articulated in Table 1). In particular, the tester was asked to judge the statements in the questionnaire using a 5-point Likert scale, where 1 means "strongly disagree" and 5 "strongly agree".

Table 1. The usability questionnaire.

Statement	Description
S1	It was simple to use this system.
S2	It was easy to find the information I needed.
S3	The information (such as on-screen messages, and other documentation) provided by this system was clear.
S4	It was easy to learn to use this system.
S5	The information is effective in helping me complete the task and scenario.
S6	The interface of this system is pleasant.
S7	I found it easy to understand what the best recommended items are.
S8	I found it useful to criticize a restaurant and get new offers.
S9	This system has all the functions and capabilities I expected.
S10	Overall, I am satisfied with this system.

Finally, after the users had tested both systems, they were asked to complete a final questionnaire to express their preference between MobyRek and MapMobyRek based on the questions listed in Table 2. We observe that a similar test approach has been used in the experiment discussed by Reilly et al. (2007). This questionnaire asks the user to compare the two systems along the following dimensions: informativeness (Q1), user interface (Q2), usefulness (Q3), and overall preference (Q4).

Table 2. The final preference questionnaire.

Question	Description
Q1	What system do you find more informative?
Q2	What system has the better interface?
Q3	What system do you find more useful?
Q4	What system do you prefer?

The Test Results

In addition to the quoted measure we logged all the testers' actions when interacting with both systems. By mining the log file, which recorded the testers' recommendation sessions data, we found that for both systems, the majority of the testers could find their desired restaurant within 2-3 recommendation cycles. Furthermore, the average interaction length, i.e., the average number of recommendation cycles in each recommendation session, for MapMobyRek was 17% shorter than that for MobyRek (Figure 6), which proves our hypothesis that MapMobyRek improves (reduces) the interaction time.

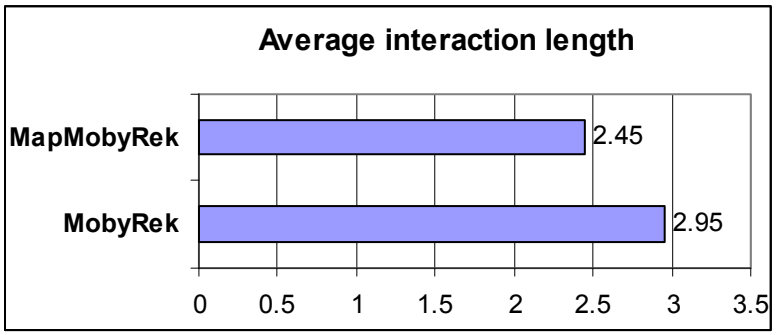


Figure 6. Comparison on the average interaction length in recommendation cycles

Regarding the average time needed to accomplish the task, the user interaction with MapMobyRek was 1.43 minutes longer than with MobyRek (Figure 7). This result contradicts our hypothesis that users can accomplish the task faster with MapMobyRek than with MobyRek. However, we observed that with MapMobyRek the time required to download the maps on the phone caused significant delays. Hence, as the previous result on the interaction length illustrates, this increased task completion time was not due to additional interactions, or cognitive load on the user; therefore, it could be reduced by a better implementation of the map management functionality. In fact, in the future we plan to use caching techniques to avoid repeated downloading of the same map data.

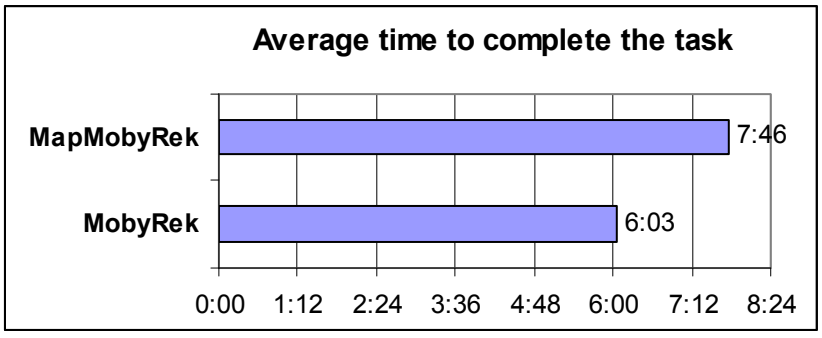


Figure 7. Comparison on the average time to complete the task in minutes and seconds (m:ss).

The average position of the selected item in the final recommendation list is an indication of the goodness of the ranking algorithm. In other words, if the user on average selects higher ranking items, i.e., those mapped to the green color, then either the user agrees with the recommendation, or she is influenced by the system and follows its suggestion. The average position of the selected restaurant was high, 2.65 for MobyRek and 2.3 for MapMobyRek (Figure 8). This is quite surprising, given the fact that MapMobyRek does not show precisely the items' positions in the recommendation list, and just uses a rough scale (4 levels corresponding to the four colors). We explain this by conjecturing that MapMobyRek users were strongly influenced by the color encoding and that MapMobyRek users selected the "green" restaurants on the maps more often than MobyRek users selected the top restaurants in the ranked lists. This result is quite interesting and suggests that the user interface for a mobile application should be different from those used, and even validated for the corresponding web-based information service.

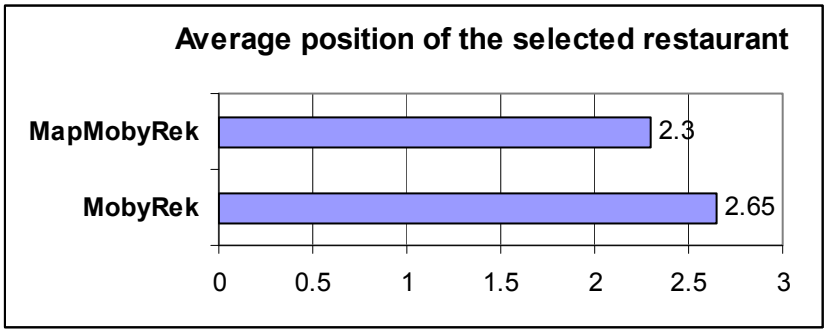


Figure 8. Comparison on the average position of the selected restaurant.

Figure 9 shows a comparison of the testers' average rating for the statements contained in the usability questionnaire. As shown in Figure 9, both systems received a positive evaluation. However, overall MapMobyRek received higher evaluations than MobyRek on all the inspected dimensions. We performed a two-tailed, paired t-test, of statistical significance on each statement's result. We found that the significant differences between the two systems relate to the statements S2, S3, S6, S7, S9, and S10. This implies that MapMobyRek provides an easier information search tool, it has a clearer information display and a more pleasant interface, it is more transparent for the users, it has a richer set of system functionalities, and it is generally preferred by the users. Nonetheless, with respect to some statements, such as S1 and S8, there is no significant difference between the two systems. For instance, the result for statement S8 ("it is useful to criticize a restaurant and get new offers") shows that the map-based interface does not change the user evaluation on this aspect. This is a reasonable conclusion and provides a counter check, since the same critiquing functionality (and the same user interface for critiquing) is implemented in both systems.

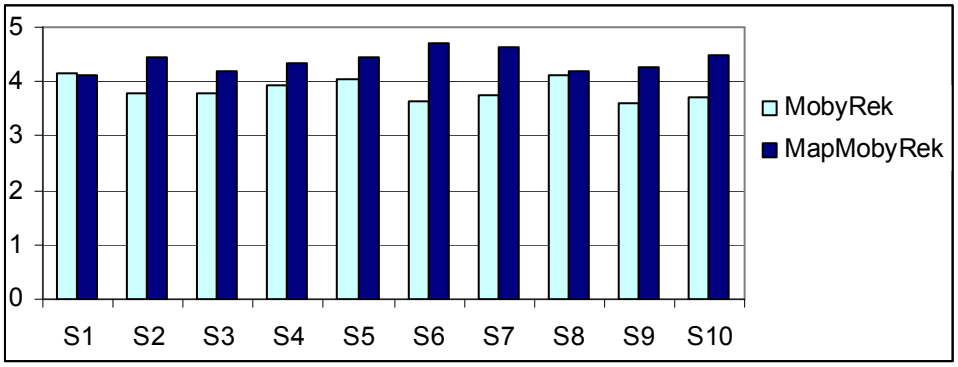


Figure 9. Comparison on the statements in the usability questionnaire.

In the final questionnaire we asked each tester to vote for the preferred system. The comparative results show that almost all the testers stated that the map-based system is better than the list-based one in terms of usefulness, interface and informativeness (Figure 10). Hence, overall we can conclude that MapMobyRek represents a step ahead for mobile RSs, and its main characteristic, i.e., of being based on a map-based display, should be used as the basis for future developments.

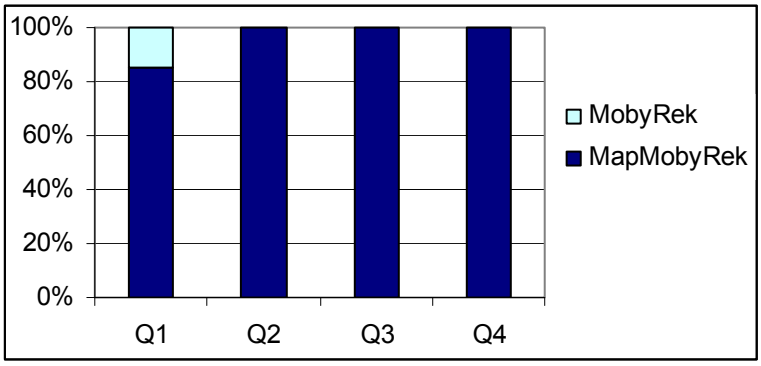


Figure 10. Comparison on the questions in the final preference questionnaire.

FUTURE WORK

Notwithstanding the success of this research, there are several open issues that can be further investigated for applying the proposed solutions to other systems and in other application contexts. Firstly, we did not investigate how different mappings of rank values to colors could influence the user decision. It would be interesting to analyze to what extent we can push the extension of "green" items (i.e., strongly recommended), and how an overabundance of "green" items can influence the user decision. This touches a fundamental issue of RSs, i.e., whether this technology really offers to the user a tool for making better decisions or the effects of recommendations must be considered as manipulations of the user preference, i.e., the recommender system just persuades the user that the recommended items are suitable for her (Gretzel & Fesenmaier, 2005).

The second aspect that requires a better analysis relates to the fact that in MapMobyRek the user cannot review the recommendation lists produced in previous cycles. In fact, it could be helpful if the system records the recommendation states and supports the user (with an "undo" or "review" button) to view a previous recommendation state.

The third aspect relates to the system's user interface. MapMobyRek, by presenting the recommended items on the map, is very useful in helping users perceive the geographical relations between the items. Notwithstanding that MapMobyRek supports users to understand the items' recommendation level using a color encoding, it cannot visualize fine-grained differences of the match of the items to the user's preferences. For example, in Figures 3a and 4a the user cannot know how well the yellow-colored items match with her preferences. Moreover, the ranked-list style has been often used in RSs, and it is still liked by some users. Hence, it would be beneficial for the users if a RS can support both types of interface (i.e., map- and ranked list- based) and provides a menu function for users to switch between these two types of interface during their interaction with the system.

Finally, an important topic for further investigations is how to integrate recommendations for different product types (e.g., restaurant, hotel, activities, etc.) so that the system can suggest them as a single travel package. This would be very important for a real commercial implementation of the system, and would open the road to dynamic packaging of mobile commerce.

CONCLUSIONS

In this chapter we have presented an approach for integrating recommendation and electronic map technologies to build a map-based mobile recommender system that can effectively and intuitively provide personalized travel suggestions to mobile users. Our system cope with the information overload problem, e.g., which is produced by the large number of restaurants available in a city, by providing a user with personalized restaurant recommendations, i.e., a well chosen selection of restaurants. The information about the restaurants is contained in a Web repository, and the system recommendations are adapted to the user's needs and preferences in the particular mobile usage context. Our recommendation approach integrates a conversational preference acquisition technology based on "critiquing" with map visualization technologies to build a new map-based conversational mobile RS that can effectively and intuitively support travelers in finding their desired products and services.

Our real-user study showed that the map-based interface is more effective than the traditional list-based interface that was typically used in RSs so far. We also found that the integration of a map-based interface in an RS increases user satisfaction, with the only additional cost of a slightly longer recommendation session.

This work provides a clear example of the fact that mobile recommender systems deserve specific methods and technologies. They cannot be simply engineered with techniques that have been proved to be successful in the wired Web applications. The user mobility, the device mobility and the wireless networking technology have a profound impact on the user task and on the system usage and they all deserve special and careful attention in order to develop effective solutions for the user information needs.

REFERENCES

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*(1), 39-59.

Adomavicius, G., & Tuzhilin, A. (2005). 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), 734-749.

Averjanova, O., Ricci, F., & Nguyen, Q. N. (2008). Map-based interaction with a conversational mobile recommender system. In *Proceedings of the 2nd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*.

Baus, J., Cheverst, K., & Kray, C. (2005). A survey of map-based mobile guides. In L. Meng, A. Zipf, & T. Reichenbacher (Eds.), *Map-based mobile services: Theories, methods and implementations* (pp. 193-209). Berlin, Germany: Springer.

Bridge, D., Göker, M., McGinty, L., & Smyth, B. (2005). Case-based recommender systems. *Knowledge Engineering Review*, *20*(3), 315-320.

Brown, B., Chalmers, M., Bell, M., Hall, M., MacColl, I., & Rudman, P. (2005). Sharing the square: Collaborative leisure in the city streets. In *Proceedings of the 9th European Conference on Computer Supported Cooperative Work* (pp. 427-447).

Burigat, S., Chittaro, L., & De Marco, L. (2005). Bringing dynamic queries to mobile devices: A visual preference-based search tool for tourist decision support. In *Proceedings of the 10th IFIP TC13 International Conference on Human-Computer Interaction* (*INTERACT 2005*) (pp. 213-226).

Burke, R. (2002). Interactive critiquing for catalog navigation in e-commerce. *Artificial Intelligence Review*, *18*(3-4), 245-267.

Burke, R. (2007). Hybrid Web recommender systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The adaptive Web: Methods and strategies of Web personalization* (pp. 377-408). Heidelberg, Germany: Springer.

Dunlop, M. D., Morrison, A., McCallum, S., Ptaskinski, P., Risbey, C., & Stewart, F. (2004). Focussed Palmtop information access combining Starfield displays with profile-based recommendations. In *Proceedings of Workshop on Mobile and Ubiquitous Information Access* (pp. 79-89).

Dunlop, M. D., Elsey, B., & Montgomery Masters, M. (2007). Dynamic visualisation of ski data: A context aware mobile piste map. In *Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services* (*Mobile HCI 2007*) (pp. 211-214).

Fesenmaier, D. R., Werthner, H., & Wöber, K. (Eds.). (2006). *Destination recommendation systems: Behavioral foundations and applications*. London: CAB International.

Gretzel, U., & Fesenmaier, D. (2005). Persuasiveness of preference elicitation processes in destination recommendation systems. In *Proceedings of the 12th International Conference on Information and Communication Technologies in Travel and Tourism* (*ENTER 2005*) (pp. 194-204).

McGinty, L., & Smyth, B. (2006). Adaptive selection: An analysis of critiquing and preference-based feedback in conversational recommender systems. *International Journal of Electronic Commerce*, *11*(2), 35-57.

Meng, L., Zipf, A., & Winter, S. (2008). *Map-based mobile services: Design, interaction and usability*. Berlin, Germany: Springer.

Mohapatra, D., & Suma, S. B. (2005). Survey of location based wireless services. In *Proceedings of the 2005 IEEE International Conference on Personal Wireless Communications* (pp. 358-362).

Nguyen, Q. N. & Ricci, F. (2004). User preferences initialization and integration in critique-based mobile recommender systems. *Proceedings of the 5th International Workshop on Artificial Intelligence in Mobile Systems*, 71-78.

Nguyen, Q. N., & Ricci, F. (2007). Replaying live-user interactions in the off-line evaluation of critique-based mobile recommendations. In *Proceedings of the 2007 ACM Conference on Recommender Systems* (pp. 81-88).

Nguyen, Q. N., & Ricci, F. (2008a). Long-term and session-specific user preferences in a mobile recommender system. In *Proceedings of the 2008 International Conference on Intelligent User Interfaces* (pp. 381-384).

Nguyen, Q. N., & Ricci, F. (2008b). Conversational case-based recommendations exploiting a structured case model. In *Proceedings of the 9th European Conference on Case-Based Reasoning*.

Park, M. H., Hong, J. H., & Cho, S. B. (2007). Location-based recommendation system using Bayesian user's preference model in mobile devices. In *Proceedings of the 4th International Conference on Ubiquitous Intelligence and Computing* (pp. 1130-1139).

Pospischil, G., Umlauft, M., & Michlmayr, E. (2002). Designing LoL@, a mobile tourist guide for UMTS. In *Proceedings of the 4th International Conference on Human Computer Interaction with Mobile Devices and Services* (*Mobile HCI 2002*) (pp. 140-154).

Pourret, O., Naïm, P., & Marcot, B. (2008). *Bayesian networks: A practical guide to applications*. New York: Wiley.

Reilly, J., Zhang, J., McGinty, L., Pu, P., & Smyth, B. (2007). Evaluating compound critiquing recommenders: A real-user study. In *Proceedings of the 2007 ACM Conference on Electronic Commerce* (pp. 114-123).

Resnick, P., & Varian, H. R. (1997). Recommender systems. *Communications of the ACM*, *40*(3), 56-58.

Ricci, F. (2002). Travel recommender systems. *IEEE Intelligent Systems*, *17*(6), 55-57.

Ricci, F., & Nguyen, Q. N. (2007). Acquiring and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent Systems*, *22*(3), 22-29.

Steinfield, C. (2004). The development of location based services in mobile commerce. In B. Preissl, H. Bouwman, & C. Steinfield (Eds.), *E-Life after the dot com bust* (pp. 177-197). New York: Springer.

ten Hagen, K., Kramer, R., Hermkes, M., Schumann, B., & Mueller, P. (2005). Semantic matching and heuristic search for a dynamic tour guide. In *Proceedings of the 12th International Conference on Information and Communication Technologies in Travel and Tourism* (*ENTER 2005*) (pp. 149-159).