POCKET RESTAURANTFINDER: A Situated Recommender System for Groups

Joseph F. McCarthy

Accenture Technology Labs
161 North Clark Street
Chicago, IL 60601 USA
mccarthy@techlabs.accenture.com
www.accenture.com/techlabs/mccarthy
+1.312.693.6761

Abstract. Most recommender systems are designed to suggest alternatives – such as a movie to see or a music CD to buy – to individuals as they browse the World Wide Web at their desktop. POCKET RESTAURANTFINDER is a new recommender system that suggests alternatives to a *group* of people, taking into account the different preferences of the members of that group. Furthermore, the system is a *situated computing* application: designed to run on a kiosk or a handheld computer, allowing it to be used in the physical contexts in which it may be most useful, such as when a group of conference attendees are trying to decide upon a restaurant for dinner.

Keywords. Situated computing, ubiquitous computing, handheld computing, CSCW, human-computer interaction, social issues.

1. Introduction

A recommender system is a computer program that suggests a course of action to a user based on information known or inferred about that user. Recommendations are usually based on some combination of three factors: a profile of the user's preferences or history, the profiles of other users who are somehow similar to the user, and/or an analysis of the content of the alternatives being recommended [Claypool, *et al*., 1999; Mooney & Roy, 1999]. Examples of application domains developed thus far include the recommendation of videos [Hill, *et al.*, 1995], music albums [Shardanand & Maes, 1995] and news [Konstan, *et al.*, 1997].

Most recommender systems are designed to suggest alternatives to a single *individual*. However, there are a number of domains for which a recommender system might make suggestions to help *groups* of people decide among alternatives. For example, while decisions regarding the purchase of a music CD primarily affects an individual, decisions regarding concerts or music clubs to attend often affect groups of people who all want to participate in the activity. Similarly, the purchase of a

videotape is often an individual decision, whereas the decision about which movie (or other performance) to attend at a theater is a decision often arrived at by group consensus. Unfortunately, most recommendation systems have no explicit mechanism to take into account the preferences of a group of people who all want to participate in an activity.[1]

Furthermore, the most common scenario of use for a recommender system has been for a user to seek suggestions while sitting at a desktop computer connected to the Internet. Although wireless Internet access may enable many of these systems to be used in more varied contexts, very few systems have taken advantage of this new mobility by using contextual features available in the physical world.[2] With the advent of Situated Computing [Gershman, *et al.*, 1999], one can imagine a *situated* recommender system that suggests one or more movies, plays or restaurants that are available in the user's immediate vicinity.

POCKET RESTAURANTFINDER is designed to address both of these issues, recommending restaurants to groups of people all desiring to dine together, based both on the location of these people as well as their culinary preferences. Prospective diners fill out a profile of their preferences regarding restaurants, including how far they are willing to travel, how much they are willing to spend, what types of cuisine they like (and don't like), and what types of restaurant amenities they like (and don't like). When a group of people is gathered together, POCKET RESTAURANTFINDER pools these preferences together and present a list of potential restaurants, sorted in order of expected desirability for the group.

One scenario of potential use for such a system is at a workshop or conference, where people who may not know much about area dining options nor the other attendees can use the system to find a post-event dining site. The inquiries and negotiations that take place in such scenarios can often be lengthy, and usually result in the selection of sub-optimal restaurants. We are looking to insert technology into those scenarios to allow for quicker and better selections for such groups.

POCKET RESTAURANTFINDER is an extension of many ideas first explored in MUSICFX [McCarthy & Anagnost, 1998; Nagendra Prasad & McCarthy, 1999; McCarthy, 2000], a system that selects music that will best please a group of people working out in a fitness center. MUSICFX contains a database of user preferences, a badge system for determining who is working out at any given time, and a group preference arbitration algorithm for determining the best music to play. We now seek to extend these ideas into a new domain and in varying contexts.

---

[1] The PolyLens system [O'Connor, et al., 2001] allows MovieLens users to join a group for which recommendations will be made for all members, but it does not explicitly support ad-hoc collections of people who spontaneously decide to jointly participate in an activity.

[2] Notable exceptions include comparison shopping agents that utilize bar code scanners [Brody & Gottsman, 1999; Barpoint.com], and location-based services that may use the Global Positioning System (GPS) data such as go2online.com.

The rest of this paper describes the POCKET RESTAURANTFINDER application, the social and technical issues that arise in the design of a *situated group preference arbitration system*, and future extensions envisioned for the specific application as well as the class of systems it represents.

2. System Overview

POCKET RESTAURANTFINDER consists of several components: a database of restaurant information, a database of users' preferences, an interface for accessing those preferences and specifying the group of people to be considered as recommendees, and an algorithm to decide which alternative[s] to suggest. Each of these will be described below.

2.1. Restaurant Database

The Restaurant Database contains a collection of records, where each record corresponds to a specific restaurant, and each field represents a different attribute for that restaurant. The fields can be partitioned into four general categories of information for each restaurant record:

- The *location* of the restaurant, represented as the street address and city.
- The *average cost* of a meal at the restaurant, rounded to the nearest dollar.
- The primary *cuisine(s)* or style of food in which the restaurant specializes, e.g., French, pizza or delicatessen. There are currently 15 types of cuisine in our database, represented as a binary-valued feature vector; each type of cuisine offered by the restaurant is represented by a one in the corresponding position of the vector, with all other elements being zero.
- A set of features or *amenities* offered by the restaurant, e.g., non-smoking seating, vegetarian selections or outside dining. We again use a binary-valued feature vector for amenities, with 17 elements; each element corresponding to an amenity offered by the restaurant is set to one, all other elements are set to zero.

Ideally, the restaurant database would encompass a rich set of features as contained in a guide such as Zagat [zagat.com], including ratings of restaurant features such as the food, service and decor. However, our initial prototype is based on a much simpler list of restaurants near our former office in Northbrook, IL (USA) that was compiled by the local Accenture concierge service.

2.2. User Database

The fields in the User Database correspond to the information stored in the Restaurant Database. However, since these fields represent preferences, they are all on a 5-point

common scale indicating how important each restaurant feature is to the user.[3] The values on this scale are interpreted as shown in Table 1.

Value	Interpretation
2	I definitely want this feature
1	I want this feature
0	I don't care about this feature
-1	I don't want this feature
-2	I definitely don't want this feature

Table 1: Preference Values and Interpretations

These preference ratings are applied to each of the potential restaurant features, so for each user, we store the following information:

- *Distance*: The distance the user is willing to travel to get to a restaurant, measured in minutes. Rather than specifying a single value, e.g., an absolute maximum distance, we have broken this down into three categories: less than 10 minutes, 11 to 20 minutes, more than 20 minutes. The user assigns a preference value to each of these categories, e.g., if eating nearby is very important, the user may specify values of 5, 1 and 1 for these categories, whereas a weaker preference for something close by might be represented by values of 4, 3 and 2.
- *Cost*: The amount the user is willing to pay for a meal, measured in US dollars. Again, rather than specifying a single value, we have broken this down into three categories: less than $10, $11 to $20, and more than $20. The user assigns a preference value to each of these categories.
- *Cuisine*: A preference value for each of the types of cuisine listed in the Restaurant Database.
- *Amenities*: A preference value for each of the types of amenities listed in the Restaurant Database.

In addition to these low-level preferences, we have inserted a higher layer that allows each user to specify the *relative importance* of each of the four categories of preferences. The four values must sum to one. So, for example, to represent priorities – in decreasing order of importance – of distance, then cost, then cuisine, then amenities, one might specify values of 0.4, 0.3, 0.2 and 0.1.

2.3. Group Preference Arbitration Algorithm

When a group of people expresses a desire to dine out together, a group preference arbitration algorithm is invoked to create a list of prospective restaurants, sorted in order of expected desirability (for that group). The inputs to the algorithm are the

[3] We don't claim that this scale is optimal, but we believe it corresponds well enough to people's intuitive notion of preference to enable us to test our prototype. The same scale has worked with great success in MUSICFX [McCarthy & Anagnost, 1998], which selects music, rather than restaurants, for a group of people.

Restaurant Database, the indices into the records in the User Database for each member of the group, and the location of that group.

The algorithm first computes each person's *individual preference* for each restaurant, then takes the average of these values to represent the *group preference* for each restaurant, and uses that single value to sort the restaurant list.

The algorithm for computing the individual preference of person *i* for restaurant *j* can further be broken down according to the four categories of preferences currently encoded within the system. For simplicity of exposition, assume that there are three *location* values in the Restaurant Database, corresponding to the three location fields in the User Database (< 10 minutes, 10-20 minutes, > 20 minutes), exactly one of which is one, all others being zero. Similarly, assume that there are three *average cost* fields in the Restaurant Database that correspond to the three cost fields in the User Database (< \$10, \$10-20, > \$20), and that exactly one of these fields is one and all others are zero.

We compute the *LocationScore* for person *i* and restaurant *j* as:

$$LocationScore_{i,j} = \arg\max\left\{LocationPref_{i,k} \times LocationVal_{i,k} \middle| k = 1..3\right\}$$

Similarly, we compute the *CostScore* as

$$CostScore_{i,j} = \arg\max\left\{CostPref_{i,k} \times CostVal_{i,k} \middle| k = 1..3\right\}$$

In both cases, we take the maximum amount, since only one of the three elements in the *LocationVal* and *CostVal* vectors will have a non-zero value.

The *CuisineScore* is calculated by taking the average of person *i*'s preference for each of the types of cuisine offered by restaurant *j*:

$$CuisineScore_{i,j} = \frac{\sum_{k=1}^{MAXCUISINES} CuisinePref_{i,k} \times CuisineVal_{j,k}}{\sum_{k=1}^{MAXCUISINES} CuisineVal_{j,k}}$$

The *AmenityScore* is calculated similarly:

$$AmenityScore_{i,j} = \frac{\sum_{k=1}^{MAXAMENITIES} AmenityPref_{i,k} \times AmenityVal_{j,k}}{\sum_{k=1}^{MAXAMENITIES} AmenityVal_{j,k}}$$

Once we have these four scores, we adjust them according to the relative weights specified by the user:

$$\begin{aligned} IndividualPref_{i,j} = \; & LocationScore_{i,j} \times LocationPriority_i + \\ & CostScore_{i,j} \times CostPriority_i + \\ & CuisineScore_{i,j} \times CuisinePriority_i + \end{aligned}$$

$$AmenityScore_{i,j} \times AmenityPriority_i$$

Once the individual preference for each person and each restaurant is computed, we can calculate the group preference for each restaurant *j* by simply summing the individual preferences:

$$GroupPref_j = \sum_{i=1}^{NumDiners} IndividualPref_{i,j}$$

The restaurant list is then sorted according to this value, and presented to the users for inspection.

2.4. User Interfaces

Two versions of POCKET RESTAURANTFINDER have been developed; one that runs on a kiosk another that runs on a handheld device. Each of these will be described below.

2.4.1. Kiosk Interface

There are two primary activities in which a kiosk user of POCKET RESTAURANTFINDER might engage: entering or updating his or her preferences, and seeking a restaurant recommendation. Although these two tasks might eventually be supported within the same interface, our current kiosk prototype uses two different applications for each of the tasks.

For accessing preferences, the user visits a web page in a standard browser such as Internet Explorer or Netscape Communicator, and logs in to the system by entering his or her name.[4] The user is then presented with a series of web pages that display preferences in each of the four categories (distance, cost, cuisine and amenities). These pages are generated by Active Server Page (ASP) scripts that format the data contained in the User Database. Each preference can be modified by adjusting the value in a listbox. The User Database is then updated to reflect any modifications made by the user.

The interface for recommending restaurants in the current prototype is a new button on our ACTIVEMAP application [McCarthy & Meidel, 1999], running at a kiosk. Our workspace contains a network of infrared sensors – mounted in the ceiling above each office, meeting room, hallway and open area – and most of our colleagues wear infrared badges that emit identification signals every 2 seconds. ACTIVEMAP displays pictures of people on a blueprint-style map of the workspace over the locations they

[4] Since the initial prototype is being used in a small group in a closed environment – researchers in our lab – we have conveniently ignored security issues. However, adding password protection and other security features will likely be required for more widespread distribution and use.

were last seen. We modified the application to include location information for the computer upon which ACTIVEMAP is running, and added a "Lunch" button. When this button is pressed, the preferences for the people gathered at the kiosk (or wherever else the application might be running) are used for determining the list of candidate restaurants.

2.4.2. Handheld Interface

The handheld interface has been developed for a Palm computer. The initial screen for the application allows the user to update his or her preferences, update the group of people whose preferences should be considered, and view a sorted list of restaurants based on the preferences of that group.

The interface for preference updates starts of with a menu specifying the four categories (distance, cost, cuisine, amenities). Within each main category, the different subcategories are listed, across from which a pull-down menu is provided in order to allow the user to specify a preference value for each subcategory.

The group management interface allows a user to send his or her preferences to another user (using the Palm infrared beaming capability), receive preferences from another user, delete [the preferences of] any person on the current group list, or clear the entire list.

The list of restaurants is shown in decreasing order of overall group preferences. Tapping on a restaurant shows the individual category scores that contributed to the group preference value for that restaurant (the current version is not very user-friendly, but useful for debugging).

3. Social Issues

The selection of a destination or activity that will please a group of people is an iterative, sociopolitical process that can prove quite challenging, depending on the destination, activity and people involved. Two of the most vexing challenges have to do with a lack of awareness: not knowing everyone's preferences and not knowing what the local alternatives are. Some people are more apt to express their preferences than others, potentially leading to dominance by a vocal minority. This can be especially problematic when there are different power relationships among the members in the group. When the people don't know each other, other problems arise because no one knows how much to rely on others' recommendations.

Another problem arises when there is little awareness of the alternatives available in a location. The members of the group can examine various guides and brochures, but most are incomplete and inconsistent with respect to the kinds of information they provide about the alternatives. A good concierge can be of great assistance in this regard, but one is not always available.

POCKET RESTAURANTFINDER seeks to solve these issues by providing a means for everyone to have a voice in the matter – through the electronic specification of their preferences – and by providing a complete and consistent list of alternatives, sorted according to the alternatives that are most likely to satisfy the specified preferences.

Privacy is another social issue we need concern ourselves with, since the preferences of a group of people must somehow be combined in order for the system to work, but people may not want to share all their preferences with others. The handheld application currently does not reveal individual preferences of other users; only the group preference values are accessible. The kiosk application has previously noted security flaws, for which straightforward solutions exist; thus far, people in our group have not expressed any privacy concerns.

4. Future Work

We currently have two different implementations of the POCKET RESTAURANTFINDER, neither of which can communicate with the other. We would like to introduce a link between these systems to accommodate those situations in which some people have handheld computers and others do not. For example, people without handheld computers planning to attend a conference could enter their preferences via a [more secure] browser interface. Once at the conference, a kiosk computer with infrared capability (or docking station) could be used so that people with handheld computers could either retrieve preferences of those without handhelds, or send their preferences to the kiosk computer, so that all preferences of an ad-hoc group could be gathered in one repository. Another possible solution might incorporate a handheld computer with a wireless Internet connection and a barcode scanner; if conference attendees had badges with barcodes, these could be scanned in order to retrieve their preferences (obviously, some security issues would have to be resolved in this scenario).

Our Restaurant Database currently contains static distance information – the distance between our research lab and any of the restaurants listed. Ideally, this information would be dynamically computed, either by manually entering information about the user's current (or intended) location, or by using an automatic location sensing system such as the Global Positioning System (GPS). This would be especially useful if the group of prospective diners arrives at a restaurant only to find it closed or full, and wants to find another restaurant near that location (vs. their starting location).

One important area of future work is to conduct usability studies to see how people use the system. We have released the system internally and held some informal discussions with initial users of the system, and the pilot group has given us generally positive feedback. However, we need to do more extensive studies – including the construction of an evaluation scheme – to determine how useful people find POCKET RESTAURANTFINDER, and the whole concept of a group preference arbitration system for restaurant selection.

Another area of future work is to enrich our Restaurant Database, e.g., by using data compiled by the Zagat restaurant guide. In addition to its extensive set of cuisine types (over 70) and amenities (over 60), the food, décor and service ratings would add entirely new dimensions to the kinds of preferences people could express. For example, one could specify a preference for restaurants with a food rating above 20 (the Zagat scale for each of these three dimensions ranges from 0 to 30). Real-time seating availability information for each restaurant would be another positive enhancement.

We would also like to enhance the User Database with new features. One area we'd like to enhance is the specification of restricted diets. For example, a vegetarian might have a variety of preferences, but having meatless dishes available would be a requirement for any restaurant to be acceptable. Such restrictions would also necessitate enhancements to our Restaurant Database, and collecting data on all restaurants for a collection of potential dietary restrictions would be a significant undertaking.

Introducing a history mechanism would also be useful, enabling us to use content-based or collaborative filtering techniques to supplement the user-specified preferences. For example, if the user has specified high ratings for a set of restaurants that he or she has dined in previously, any restaurants in the current location that are similar to those restaurants should also be appealing to that user. Likewise, if people with similar profiles to the user have rated a set of local restaurants highly, those restaurants are also likely to be appealing to that user. Of course, incorporating features such as these would require a reworking of the group preference algorithm, which is based on preferences for features rather than specific restaurants.

5. Conclusion

POCKET RESTAURANTFINDER is a situated group preference arbitration system for helping a group of people select a restaurant that would best meet their dining preferences. Such a system is particularly useful in contexts in which people don't know each other very well, and in locations where people don't know the restaurants of the area very well, such as a gathering of researchers at a conference or workshop.

Initial prototypes of the system have been developed and tested for use at a kiosk within our research lab and on handheld computers used in the vicinity of our lab. While initial feedback has been promising, we have plans for a number of extensions and more extensive studies to assess its usability and utility. To this end, we hope to field test POCKET RESTAURANTFINDER at a conference or workshop in the near future.

We believe that situated group arbitration has broad applicability in a variety of contexts. In addition to restaurant selection, we expect similar systems could be used to arbitrate the selection of other group activities such as movies, plays and music performances, democratizing the selection process across a broad spectrum of activities.

6. Acknowledgements

The author wishes to thank Adam Brody, Tony Costa and Quan Tran for their help in developing different implementations of the POCKET RESTAURANTFINDER.

7. References

1. Adam B. Brody and Edward J. Gottsman. 1999. POCKET BARGAINFINDER: A Handheld Device for Augmented Commerce. In *Proceedings of the International Symposium on Handheld and Ubiquitous Computing (HUC '99).* Karlsruhe, Germany.
2. Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes and Matthew Sartin. 1999. Combining Content-Based and Collaborative Filters in an Online Newspaper. In *Proceedings of the ACM SIGIR Workshop on Recommender Systems*, Berkeley, CA, August, 1999.
3. Anatole V. Gershman, Joseph F. McCarthy and Andrew E. Fano. 1999. Situated Computing: Bridging the Gap between Intention and Action. In *Proceedings of the Third International Symposium on Wearable Computing (ISWC '99)*, San Francisco, CA.
4. Will Hill, Larry Stead, Mark Rosenstein and George Furnas. Recommending and Evaluating Choices in a Virtual Community of Use. 1995. In *Proceedings of the 1995 ACM Conference on Human Factors in Computing Systems (CHI '95)*, Denver, CO, pp. 194-201.
5. Joseph A. Konstan, Bradley N. Miller, David Maltz, Jon Herlocker, Lee R. Gordon and John Riedl. GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM* 40,3 (1997), pp. 77-87.
6. Joseph F. McCarthy. 2000. Active Environments: Sensing and Responding to Groups of People. In *Proceedings of the 2000 Conference on Human Factors in Computer Systems (CHI 2000)*, Extended Abstracts, The Hague, pp. 51-53.
7. Joseph F. McCarthy and Theodore D. Anagnost. 1998. MUSICFX: A Group Preference Arbitration System for Computer-Supported Collaborative Workouts. In *Proceedings of the ACM 1998 Conference on Computer Supported Cooperative Work (CSCW '98)*, Seattle, WA, pp. 363-372.
8. Joseph F. McCarthy and Eric S. Meidel. 1999. ACTIVEMAP: A Visualization Tool for Location Awareness to Support Informal Interactions. In *Proceedings of the First International Symposium on Handheld and Ubiquitous Computing (HUC '99)*, Karlsruhe, Germany. Published in Hans-W. Gellerson (ed), Lecture Notes in Computer Science 1707, Springer, pp. 158-170.
9. Raymond J. Mooney and Loriene Roy. 1999. Content-Based Book Recommending Using Learning for Text Categorization. In *Proceedings of the ACM SIGIR Workshop on Recommender Systems*, Berkeley, CA, August, 1999.
10. Mark O'Connor, Dan Cosley, Joseph A. Konstan and John Riedl. 2001. PolyLens: A Recommender for Groups of Users. In *Proceedings of the Seventh European Conference on Computer Supported Cooperative Work (ECSCW 2001)*, Bonn.
11. M. V. Nagendra Prasad and Joseph F. McCarthy. 1999. A Multi-Agent System for Meting out Influence in an Intelligent Environment. In *Proceedings of the Eleventh Conference on Innovative Applications of Artifical Intelligence (IAAI-99)*, Orlando, FL, pp. 884-890.
12. Upendra Shardanand and Pattie Maes. 1995. Social Information Filtering: Algorithms for Automating Word of Mouth. In *Proceedings of the 1995 ACM Conference on Human Factors in Computing Systems (CHI '95)*, Denver, CO, pp. 210-217.