Acquiring User Preferences for Personal Agents

Paolo Viappiani, Pearl Pu, Boi Faltings

Artificial Intelligence Laboratory (I&C) - Swiss Federal Institute of Technology (EPFL)
IN-Ecublens, 1015 Lausanne, Switzerland
paolo.viappiani|pearl.pu|boi.faltings@epfl.ch

Abstract

Personal agents represent a novel paradigm that merges ideas from the agent based computing and the intelligent information system areas. A personal agent gathers and filters information on user's behalf, models user's needs and preferences in order to generate recommendations. To build an efficient user model, both explicitly stated and *hidden* (inferred from context or other users) preferences have to be considered because people are not good at describing their own decision criteria; moreover preferences can change over time.

In this paper different techniques for modeling and obtaining preferences are presented, with special emphasis on systems that interact with the user in form of dialogue, in a way that makes it possible to elicit preferences just in time. Several questions that require further investigations are raised.

Introduction

People have to face increasing amount of complex information. In the *information* era human cognitive capabilities are brought to extreme. Every day, we loose a significant amount of time searching for information, browsing the www, handling and answering to mails. In recent years, whilst our means of getting information increase (with mobile devices like mobile phones, PDA, etc.) the need for information filtering has dramatically emerged as an important issue.

A personal agent that will effectively address this problem will gather information and filter it to generate recommendations. Such systems are at the merge point of the agent based and intelligent information management research areas. Agents will build a user profile through interaction. A user profile could consist of various kinds of information about the user but usually will record preferences in some ways.

In reality users preferences depend heavily on the decision context and are often unknown in the beginning, ill specified, and incomplete. If asked, people often state some criteria to be very important, but prefer others when more decision criteria emerge later on.

Agent based computing

Agents are a paradigm for computation that, if not yet mature, has been now addressed by the research community for some years. Distributed and heterogeneous information systems are the framework in which agents are proliferating.

Agent attributes usually include autonomous behavior, social interaction (the ability of interacting through dialogue users or other agents), reactivity and goal-oriented behavior. Sometimes learning capabilities and code mobility are also present (Wooldridge 2002).

Personal agents will show these characteristics: they will run for long time and assist the user. Autonomous behavior means that the agent would eventually collect information independently, and act towards the goal of getting the recommendation.

Interests in having a tool for personalized information access are increasing, specifically through mobile devices. Siemens' new TravelAgent (www.mobile-travel.com), a personal travel assistant, developed jointly with IBM (now marketed by Siemens mobile travel services) involves a project of several millions of Euros. Siemens, Motorola and other companies have developed the LEAP agent platform for mobile telephones, which is used in projects such Agentcities (www.agentcities.org) to provide access into networks of information agents.

The Artificial Intelligence Laboratory, in cooperation with the Human Computer Interaction group (both at EPFL), has developed SmartClient for flight reservation. Whereas traditional on-line flight reservation systems impose a fixed decision making sequence on the user, SmartClient gives the opportunity to refine his query, modifying previously stated preferences (Pu and Faltings, 1999-2002).

In the case of eCommerce, conversational interfaces facilitate the information gathering need for getting orders. Deployed solutions are Microsoft Agent, Extempo Imps e NetSage agent.

However, the technology seems to be still immature to have a wide impact on the information technology world, due to the lack of understanding of the formulation of user preferences, user interaction, integration of information from different ontologies.

Information retrieval & recommender systems

A lot of effort in recent years has been addressing the problem of overwhelming information. Intelligent management systems have been proposed and prototyped. For example, soft-bots can perform autonomously tasks and wrappers can find information matching some well-defined rules. But totally independent systems that can find the information you want without interaction with the users do not exist yet and won't exist for some years to come.

A personal agent can consist of different modules, one that gathers information in a partially supervisioned way, and another module that, based on user preferences, gives recommendations to the users. For example, if the user would like to organize his vacation, the agent could suggest different plans based on the information available from the information sources (databases, websites of hotels and travel agencies) and the user profile (that the agent has learned over time). For the problem of linking different information sources with different users, several formalisms have been proposed (for example, Gonzalez and Faltings 2002).

At this point it is better to clarify the distinction between reputation and recommendation. Reputation is the accumulated scale of opinions of products or persons from a significant population of people, whereas a recommendation is computed information to help users to find similar and related items, based on the knowledge of user profile.

Preferences are the general concept that denotes all the component of the user profile. They can either be

- explicitly acquired, or
- obtained from other users/agents or inferred from the context

In the next sections we'll describe the possible formalisms for representing preferences, following by their acquisition techniques.

Preferences modeling

Soft Constraints

Constraint satisfaction problem is a highly successful framework for formalizing problem solving (Bartak 1999); it has been proven to be expressive enough for a wide range of practical problems, while at the same time being restricted enough to allow efficient general techniques for solving them.

It is possible to take advantages of the power of CSP approach for representing preferences using the notion of multi criteria CSP (Meseguer Rossi Scheix 2002).

A *multi-criteria constraint optimization problem* is defined by a tuple *P=(X,D,C)*, where X is a finite set of **variables**, each associated with a **domain** of discrete D values and a set of **constraint** C. This time a constraint *c* is a function that asserts whether the tuple is *allowed* (0) or *fully disallowed* (K), being the value in the interval between 0 and K (also possible to assign conatraints values in "the higher the better" fashion). Constraints that can have not only crisp value are said to be *soft*.

If it is not possible to find a solution (an assignment of the variables to domain values in CSP terminology) that fully satisfies all the constraints then we would like to select as optimal solutions the ones that better satisfy the constraints (that can be seen as criteria).

Given the fact that there's no way to optimize many criteria without any compromises, three different strategies are more commonly used in literature (Meseguer, Rossi, Scheix 2001).

- **Weighted-CSP** we sum values returned by each-criterion with a weight and select the solution with the lowest (greatest) sum, solving a traditional optimization problem (frequently with branch&bound). Therefore the method is also known as MinCSP (MaxCSP);
- **Fuzzy-CSP** each solution is associated with the worst criteria evaluation;
- **Lexicographic-CSP** criteria have different importance and we order solutions in a lexicographic order.

We will adopt the first solution. It is then possible to define the **user model** as a set of couples $\{(c_1,w_1),..,(c_n,w_n)\}$, where for each i, c_i is a soft constraint and w_i the associated weight. Refining the user model means to add/remove constraints, and change the associated weights (as consequences of users' critiques in the dialogue model).

Given a user model to find a solution means to solve the associated CSP. The **error E of a solution s**, given a user model, is defined as the weighted sum of the constraint satisfaction degrees *c(s)*, which we want to minimize.

$$E(s) = \sum_{i=1}^{n} c_i(s) \times w_i$$

However, since the user model can be wrong, there's no guarantee that the optimal solution of the MinCSP will be the best solution to the problem. Once again, the tricky part seems to be the user model identification.

When the relative importance of criteria is unknown, it is not possible to identify a single best solution. Instead, we can discard a lot of solutions that are recognized to be worse than *(dominated by)* others for all the criteria. A **Pareto set** groups all such solutions that are not discarded.

The system could propose solutions in the Pareto set as candidates, based on the user reaction (acceptance/critiques), we can then estimate the relative importance of constraints, and propose another solution belonging to the Pareto set until a fairly good estimation of the user model is acquired.

Unfortunately, computing the Pareto set is not trivial, although there exists algorithms for approximation (Torrens, Faltings 2002):

1. First, solve the associated MinCSP with a chosen weight vector and save the best K solution in vector V
2. Then, extract from V the solutions that are not dominated by other solutions in V (they can be proven to be Pareto optimal)
3. Go back to step 1 and extract other solution from the associated weighted problems (with different weights)

Logical Approach

The treatment of preferences as soft constraints involves the use of numerical weights. Research has been addressing qualitative preferences too. One of the proposals is the use of a logical framework to formalize and reason about preferences.

It is possible to extend ordinary logics with the introduction of new operators, as the **ordered disjunction X.** *(a X b)* means *(a OR b)* where *a is preferred to b*. An example includes formal non monotonic logic clauses *(a X b)<-c*, meaning *a is preferred to b if c is the case*, called **LPOD** rules (Brewka, 2002).

Examples:

- cinema X beach <- not hot
- beach X cinema <- hot

Logic programming with ordered disjunction is an extension of logic programming with two kinds of negation (default and strong negation). We don't want to go into details here (semantics, etc.), but just have a glimpse of the main ideas underlying this approach. Semantics cannot be reduced to standard logic programs with two kind of negations and involves the answer set notion (Geolfond & Lifschitz).

In the examples above, we state that beach is preferred to the cinema if it is known that is hot, whereas beach is preferred if it is not known that is hot (that's what it means to have two negations: "not hot" is different from "¬hot").

The user model, in the logical approach, will consist of a set of clauses as before, which will state the user preferences in the different conditions. All the facts known to the agent will be also recorded in the form of logical clauses.

There will be then a decision making engine that, given the actual knowledge, will evaluate the preferences clauses and select the best options.

Application of LPODs and answer set programming can be the efficient automatic configuration task for workstation. We can have the different profile as condition on the right part of the rules (such as Developer, Non Developer User, etc.) and ordered alternatives of libraries and modules to be loaded by the system on startup.

Obtaining user preferences

A recommender system takes information as input and outputs recommendations, often in the form of predictions of user preferences. Information sources can be acquired through explicit questions to the users, or it can be elicited implicitly by observing user's behavior (for instance tracing his actions, i.e. observing which links he clicks at).

Our goal is to find the right user profile. This will be the definition of his preferences in some ways. However, this profile is hidden and often not known explicitly to the users himself! The user, in fact, knows only qualitative preference relations.

We classify the different kinds of methods available for getting preferences.

The **questions&answers** method is the easiest and the most used. The user is simply asked to fill a form or answer a set of questions. The answers are then saved and usually the user will need a username and a password to access to the system again, so that the system could find the user profile (in this context just the information that the user submitted).

In this basic method, there's no way for the system to learn user preferences beside the initial data submission; unless the user could edit his profile (but what will encourage him to do so?). Many users also, with the increasing pervasive presence of "register first" (yet free) service, cannot stand anymore to fill very long forms for each different site and are likely to submit false data.

In the **mixed-iniative** method (also known as *candidate/critique* mode), preferences are elicited through a dialogue with the user that critiques those solutions that don't fit him as long as an acceptable solution is not found (Linden 1997, Torrens and Faltings 1999 and Pu and Faltings 2002).

Execution loop:

1. Define a user model
2. Display optimal solution given the current model
3. Elicit and update the U's model on the basis of the user's critiques, if there are no critiques then exit
4. Explain the dataset range to get more feedback from the user
5. Go back to step 2

Mixed initiative features are:

- Any-criteria, any order search methods
- Preferences elicitation based on example critiquing
- Conflict resolution and performance valuation through visualization techniques

Flexibility and usability provide important advantages. Having no fixed ways for the user to state his requirements is much closer to the human way of thinking.

A **Content based** recommender learns a profile of the user's interests (with neural networks, decision trees, or other methods) and evaluates a new object on the basis of which features have been considered positively by the user in the past (item-to-item correlation).

Utility based and **knowledge based** are similar in the sense that they need some domain knowledge to work (where this is not true with collaborative filters).

Collaborative filtering is a method for estimating missing values from a data set. The current user is matched against the other users by the system to find the *K-nearest neighbors*, on the basis of a similarity function (usually Pearson's correlation). Then, it is assumed that the user will be likely to have similar preferences to his neighbors, so that unknown preference values (usually in the form of numerical ratings) are elicited from them (weighted sum of the nearest neighbor ratings).

Demographic filters cluster users into classes depending on users' characteristics, and assume that people of the same cluster have similar behaviors. The ways these clusters are made, and the deductions associated to the clusters, can be elicited in some other ways (with collaborative methods for instance).

This classification is not orthogonal: a system can have a candidate/critique behavior, while at the same time having some content based and utility based mechanism for generating new candidate solutions.

The greatest quality of collaborative recommenders is its ability of discovering "hidden" links between different genre of products (objects). It may be possible that people who likes Star Trek movies also enjoys Woody Allen, but there's no way a content based method can learn the user profile only on science fiction movies. Cross-genre suggestions can be made by having some inter-user preference sharing.

We can see in the following table a summary of the hitherto cited techniques, ranked in the order of accuracy. In fact, interactive methods like mixed initiative can infer a very trustable user profile through interaction. On the other hand, it is a really costly method.

"Question&answers" are at second place because if the user enters correct information, he will get correct recommendations. But they have very strong limits: they cannot *hidden* preferences and fix the way users can states requirements.

Content based are the next best thing: depending on their learning methods they can have very good performance, but since interaction is not present, can get a wrong model. For example, if I often fly very early in the morning for business, I probably won't appreciate to be suggested a 5 a.m. flight for my vacation in Greece!

Utility and knowledge based techniques can have good accuracy, but they largely depend on the knowledge base.

Collaborative methods rank only at the fourth position, but they can be very impressive in certain circumstances, as stated before. Moreover, they work very well with simple preferences such as numeric ratings in domain where it is the *personal taste* that counts. Have a look at MovieLens project website, www.movielens.org.

Demographic methods are least accurate because they rely on stereotypes. However they can be useful when only few information is known (i.e. after a new registration, we have not more than registry information). The more data the system gathers, the more accurate the clusters can be made.

1	Mixed initiative (Candidate/critique)	Conversation between system and users, preferences in the form of constraints inferred by the dialogue	-I want to fly to Paris but I can't leave before 3pm and I don't like AirFrance -what do think of flight AZ187? -I don't like Alitalia neither
2	Question & answers	Submit data with forms	Enter date, time, departure & destination
3	Content based	Learns a profile of the user's interests on the objects' associated features	You want a laptop with weight < 2 kg and prefer Dell -> Latitude 2200
4	Utility and knowledge based	Evaluate an utility function over the available options, and/or infer from functional knowledge	
5	Collaborative	Aggregate recommendations of objects and generate new ones by inter-users comparisons	People like you enjoyed 'Minority Report'
6	Demographic	Divide users into demographic classes	You're male, 18-22, go to see 'Men in black II'

The accuracy order in acquiring *hidden* preferences is different (the ones not listed are not capable of hidden preferences discovery):

1. Collaborative methods
2. Content based methods
3. Utility and knowledge based methods
4. Demographic methods

Mixed-initiative cannot find hidden preferences but can guide the user in a particular direction through the interaction.

In the second table, it is possible to compare the different approaches in terms of pros and cons other than accuracy. The table is taken from (Burke02), with some additions (he didn't treat mixed initiative and question&answer) and changes (for instance it is not clear how a pure Demographic system could identify cross-genre niches as stated there – cross genre niches generally belongs to different demographic cluster!).

Techniques	Virtues	Problems
Mixed initiative	Adaptive, Feedback	Interaction can be long
Questions& answers	Correct if the information enter is correct	People don't like to enter many data
Content based	Adaptive	New user ramp-up, dependent on large dataset
Utility based	Sensitive to changes	Must learn utility function
Knowledge based	As utility based	Knowledge engineering required
Collaborative	Identify cross-genre niches, Domain knowledge not needed, Adaptive	New user/new item ramp up problems, bad for the "gray sheep", dependent on large dataset
Demographic	Adaptive, domain knowledge not needed	Must gather demographic information

Mixing techniques is a possible way to overcome the shortcomings. For instance, a recommender could generate knowledge based recommendations for a new user and switch to collaborative method when a large enough dataset is available. For a new item, it is enough to use a content based.

All these points force the developer to focus not only on technical side, but also consider some sociological aspects.

Privacy is perhaps the trickiest point. Every vendor can choose different strategies. For example, Microsoft's Passport support collaborative filtering; privacy is guaranteed, giving the possibility to the user to edit his own profiles and grant the share with other business vendors.

Dynamic preferences are not easy to address. A steak eater that becomes vegetarian will continue to get steak-house recommendations! We believe that only a dialog approach could address this problem, giving the possibility to drop earlier stated constraints.

Just in time preference elicitation through a mixed-initiative system, where humans and agents collaborate in building an accurate and context-aware model of the user, can stimulate people better to recognize and state their preferences by critiquing possible solutions in their task context.

If, given a user model, no solution is found (i.e. the model is inconsistent), this approach will alert the user to modify some of the preferences without having to restart over the query. Dynamic preferences can be treated in the same way: if an agent has gotten some clues about the possibility that the user changed some of his preferences, he can interrogate the user. However, how this will be implemented in real case need some more investigations.

Therefore we believe that interesting possibilities remain open by integrating in a mixed-initiative system contributions from collaborative, demographic and other techniques. The work on hybrid systems (Burke 02), in fact, did not address mixed-initiative systems, even though he presented very good solutions for integrating the other techniques.

Agent-User interaction.

People are not always good at numbers. It is not easy at all to valuate criteria and say, for instance "price is important to me 0.75, time is worth 0.5 and I would prefer AirSanMarino with a weight of 0.3." The mixed-initiative approach lies on user-agent interaction and CSP formalism needs numerical values. How is then possible to define a realistic way to obtain preferences in the form of weighted constraint?

Different ways to infer the importance of user's criteria can be:

- Ask the user to rank two different solutions (products)
- Ask the user to list minimum requirements
- Ask the users to explicitly rank criteria

Some other interesting suggestions come from the important application domain of automatic recommendations for marketing purpose in the electronic market. Suppose there is an electronic catalogue accessible via WWW. The user will have in mind the desired optimal product and he wants to find the one that is as close as possible to what he is looking for.

When asked about the most important matter in a purchasing decision, people are likely to assert price as the most important one, because it is what they think at first.

Morris and Maglio (2001) have wondered whether price is really the most important thing, since people can become more flexible with price when they realize that different features related to the project are offered. They found out that the price can be seen as a *border* between what is a requirement and what is just a preference: criteria stated more important than price are requirements otherwise just preferences.

In commerce in general (in electronic commerce in particular), satisfying buyer's need is a key competitive element. An advanced approach to user preferences elicitation can be of great advantage, since in most cases the approach is very traditional: there's a separation between the moment in which the potential buyer describes what he's looking for (needs identification) and the phase of product brokering (often only explicitly expressed preferences in the form of answers to questions are taking into account).

Open issues

To conclude this paper, we like to outline the context and the challenges of research in the Personal Agents framework, especially for generating automatic recommendations and adapting a better interaction with the user.

Different choices are possible on the representation and collection of preferences. We have presented the mixed-initiative method as the one that can promise better result since it's the one that directly interacts with the user. Still, a general optimal interaction model has to be found. Another challenge will be finding an accurate way to gather preference information from multiple participants, human or agents.

We summarize some issues that are worthy of further research:

Delegation/confidence. There's no point in developing agents if people are not using them! We have to identify which tasks the users are likely to delegate to an agent. On the other side, the agent has to be sure to fully understand the problem (confidence).

Privacy. Collaborative filtering techniques gather information from several users to give personalized recommendations. However, there should be a policy of managing the data acquired to guarantee user's privacy (confidentiality of user preferences). Users won't probably use agents if they would feel that their data are not handled in a safe way.

Conflicts. The use of multiple methods for gathering preferences can cause conflicts. Which is the best way to deal with that? We suggested to give a priority order, but in some cases that could not be the best solution.

Dynamics. Users preferences are inherently dynamic and change over time. A recommender system should take care of this (and eventually predicts future changes!).

Context dependencies. A good system has to distinguish between *stable* preferences and those only relative to particular context. I could like to eat pizza with my friends (since pizzerias are often friendly and cheap – especially in Italy…) but if I plan a business dinner than I will have other expectations! Which formalism can promise to address such situations?

Performance evaluation. It is not always possible to define a well-formed quantitative manner to evaluate recommender systems: the efficiency of a recommendation encapsulates psychological aspects. On the other hands, considering performance in terms of computation load is a second step that will have to be taken in real systems.

References

Barták, R. 1999. Constraint Programming: In Pursuit of the Holy Grail. Proceedings of the Week of Doctoral Students (WDS99), Part IV, MatFyzPress, Prague, pp. 555-564

Boutilir, C., Brafman, R.I., Hoos, H.H. and Poole,D. 1999, Reasoning with conditional ceteris paribus preferences statements, in *Proc. Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*, Stockholm, Sweden.

Brewka, G.2002. Logic Programming with Ordered Disjunction, Proc. AAAI-02, Edmonton, Canada, WS-02-13, AI Press, Menlo Park, California

Burke, R. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. In Press

Gelfond, M. and Lischitz, V. 1991. Classical negation in logic programs and disjunctive databases. New Generation Computing, 9:365-385

Gonzalez, S.M. and Faltings, B. 2002 Open Constraint Satisfaction, Cooperative Information Agents, Springer Verlag Lecture Notes in Artificial Intelligence 2446, pp.218-225

Lieberman, H. Shearing, S. 2000. Intelligent Profile by Example, Technical Report, MIT media Lab, MIT

Linden, G., Hanks, S. and Lesh, N. 1997 Interactive assessment of user preference models: The automated travel assistant. In Proceedings, User Modeling '97

Meseguer, P., Rossi, F. and Scheix, T. 2002. Soft constraints theory, algorithms and applications, International Conference on Constraint Logic & Programming, CP01, Cyprus, Greece, tutorial notes

Morris, J. and Maglio, P. 2001. When Buying On-line, Does Price Really Matter? In Proceedings of the Conference on Human Factors in Computing Systems (CHI 2001), Seattle, WA, April 3-5

Pu, P. and Faltings, B. 2002. *Enriching Buyers experiences: the SmartClient approach",* Proceedings of the CHI2000 conference on Human Factors and Computing Systems, April 2002, pp. 289-296, ACM

Pu, P. and Gupta, P. 2002. *Visualizing Reputation and Recommendation in Scientific Literature*, submitted to IUI'03

Torrens, M., and Faltings, B. 1999. *SmartClients: Constraint Satisfaction as a paradigm for scaleable intelligent information systems.* AAAI-99 Workshop for Electronic Commerce, AAAI Technical Report WS-(99-01, pp. 10-15

Torrens, M. and Faltings, B. 2002. Using Soft CSPs for Approximating Pareto-Optimal Solution Sets. AAAI Workshop on Preferences in Constraint Satisfaction.

Wooldridge, M. 2002. *Reasoning about Rational Agents*. The MIT Press.