

Preference-based search with Adaptive Recommendations

Paolo Viappiani^a, Pearl Pu^b and
Boi Faltings^a

^a *Artificial Intelligence Laboratory (LIA)
Ecole Polytechnique Fédérale de Lausanne
(EPFL)*

1015 Lausanne, Switzerland

*E-mail: {paolo.viappiani@gmail.com,
boi.faltings@epfl.ch}*

^b *Human Computer Interaction Group (HCI)*

*Ecole Polytechnique Fédérale de Lausanne
(EPFL)*

1015 Lausanne, Switzerland

E-mail: pearl.pu@epfl.ch

Conversational recommenders can help users find their most preferred item among a large range of options, a task that we call *preference-based search*.

Motivated by studies in the field of behavioral decision theory, we take a user centric design perspective, focusing on the trade-off between decision accuracy and user effort. We consider example-critiquing, a methodology based on showing examples to the user and acquiring preferences in the form of critiques. In our approach critiques are volunteered in a mixed-initiative interaction. Some recommendations are suggestions specifically aimed at stimulating preference expression to acquire an accurate preference model.

We propose a method to adapt the suggestions according to observations of the user's behavior. We evaluate the decision accuracy of our approach with both simulations exploiting logs of previous users of the system (in order to see how adaptive suggestions improve the process of preference elicitation) and surveys with real users where we compare our approach of example critiquing with an interface based on question-answering.

Keywords: recommender systems, preference-based search, preference elicitation, example-critiquing, user studies

1. Introduction

People often face the difficult task of having to select the best option from a large set of alternatives, such as choosing an apartment to rent, a notebook computer to buy, or financial products in which to invest. We call this task *preference-based search*.

Many e-commerce sites offer a possibility to search for structured items. Usually they ask the user to fill in a form with questions about the requirements that the desired item must satisfy. This process is used, for example, when searching for flights on the most popular travel web sites^{1,2}, or when searching for apartments, used cars or health insurance³.

All results that match with the preferences provided in the form are retrieved. The user can go back to the form page to modify the requirements and obtain different search results. This method is very popular because it is simple to implement in a product database.

This kind of interaction is used by virtually all existing travel web sites. A corporate study of electronic commerce with emphasis on air travel [1] identified the poor level of interaction design as one of the major limitations of the use of Internet tools. They analyzed different interfaces and reported that only 18% of the users are able to find the required information. Surprisingly, user interfaces for searching a flight or an apartment have not changed substantially since the study was made.

The reason for the poor performance is that users do not usually know how to correctly translate their preferences into the requirements that the form allows them to specify. Thus they are unlikely to provide answers that reflect their true needs.

¹<http://www.travelocity.com/>

²<http://www.expedia.com>

³<http://www.comparis.ch/>

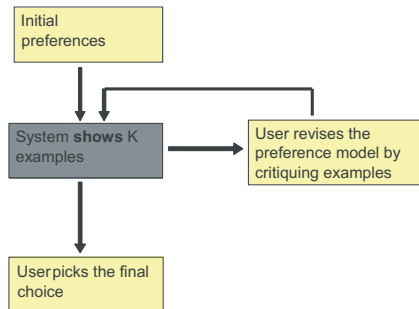


Fig. 1. The generic system-user interaction model of a conversational recommender system. In our work, some recommendations are suggestions specifically aimed at stimulating preference expression.

For example, if a travel site asks for an airline preference when the user does not actually have one, she may fill in an airline which she believes offers the best price or connection. If this belief turns out to be incorrect, it precludes the user from reaching the most preferred option. In section 2, we will analyze these kinds of biases in detail.

In our opinion, search tools should interact with the user in a way similar to that of a shop assistant, who guides the customer by showing available items and makes her discover her preferences.

As people often express preferences as reactions to the example solutions shown to them, such as “I like this digital camera, but find something less expensive”, many researchers have proposed to elicit user preferences as critiques on examples, and use this feedback to direct the next search cycle, as shown in Figure 1. This approach mimics the interaction between human buyers and sellers and is called example or candidate critiquing or conversational recommender systems [4,14,21,26,3,24,15]. These systems show examples of available options and invite users critique these examples. This process allows users to better understand their preferences and incrementally construct their preference model.

According to behavioral decision theory [18,27,33] many of the preferences are *constructed* when considering specific examples. To best support this process, example-critiquing should show examples that educate the user about the diversity of the available options, called *suggestions*. In this paper we present example-critiquing with *adaptive model-based suggestions* (a kind of conversational recommender system) and show that it greatly outperforms existing web interfaces.

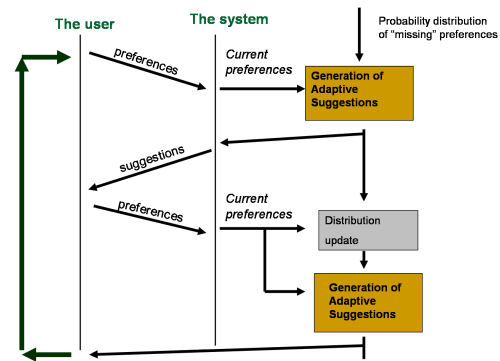


Fig. 2. With adaptive suggestions, the tool learns from the user by observing her reactions (whether or not the user critiqued a given example). If the user has no reaction after seeing an example showing a possible interesting feature (for instance access to subway in the rental domain), in the next interaction cycle the system is likely to choose suggestions that show other opportunities.

1.1. Contributions of this paper

This paper describes a new technique for eliciting preferences in conversational recommender systems based on iterative critiquing. We consider a mixed initiative tool for preference-based search in which preferences are stated as critiques to shown examples (Figure 1).

In our approach we consider additional recommendations, suggestions, that are aimed at stimulating preference expression in order to acquire an accurate model and avoid biases typical of human decision-making.

The possibility to adapt the suggestions according to the user’s reactions to previously shown examples is the main contribution of this paper. Our novel approach introduces Bayesian reasoning into a conversational recommender: the system maintains a set of beliefs modeled as probability distributions (possibly incorporating prior knowledge) about the preferences the user might not have expressed yet. The interaction cycle of a conversational recommender system with adaptive suggestions is shown in Figure 2.

We evaluate our framework with both simulations and user studies. The application domain (student housing) was chosen because of its ability to attract subjects motivated in doing the search seriously; the database size (around 200 items) is big enough to make the use of an interactive interface appealing and at the same time it makes post-hoc assessment viable (manual identification

of the real target item in order to measure decision accuracy).

We compare the critiquing approach with a standard elicitation technique based on question-answering (form-filling). We show that due to *means objectives* and other kind of biases presented in Section 2, form-filling often fails to elicit a correct preference model, while example-critiquing with adaptive suggestions provides significantly better results.

1.2. Objectives for preference-based search tools

Several recent papers have evaluated example critiquing interfaces. Pu and Kumar [22] showed that example-critiquing interfaces enable users to perform decision tradeoff tasks more efficiently with considerably less errors than non-critiquing interfaces. More recently, Pu and Chen [20] showed that the implementation of tradeoff support can increase users' decision accuracy by up to 57%.

Tools for preference-based search can be evaluated according to two conflicting design goals:

- **decision accuracy**, the fraction of users that find their most desired option using the tool, and
- **user effort**, either the number of interaction cycles or task time that the user takes to find the option that she believes to be the target using the tool.

In this framework, we favor systems that offer maximum accuracy while requiring the same or less user effort. This framework was first proposed by [18] while studying user behaviors in high-stakes decision making settings and later adapted to online user behaviors in medium-stakes decision making environments by Pu and Chen [20] and Zhang and Pu [37].

2. Human biases in preference elicitation

Behavioral Decision Theory studies how people make choices. According to Payne et al. [18], people show a contingent decisional behavior, and adapt their strategies for solving decision processes according to the situation. However, this also means that human decisions are sensible to a variety of context factors.

Figure 3 illustrates the contexts of human decision making. Decisions are influenced by

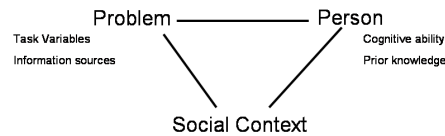


Fig. 3. According to Payne [18] and several other researchers, there are three major types of factors in human decision making: those related to the problem (as the way information is displayed), those related to the decision maker (his cognitive ability and prior experience) and those related to the social context (as accountability and group membership).

- the characteristics of the decision tasks,
- the person's prior knowledge and expertise in the problem domain: experts tend to follow some pre-established patterns that might involve quantitative reasoning, while novices often use simplifying strategies that avoid complex reasoning,
- the social context: the need to justify a decision to others may cause the choice to be more sensitive to certain aspects.

These aspects are also correlated, as the amount of information will have influence on the cognitive ability of the person.

Given the complexity of the human decision behavior, it is easy to understand that as a consequence of the multiple interdependencies of these factors, people sometimes make substantial decision errors [31].

We consider some of the common biases in human decision making: means objectives, the prominence effect and the anchoring effect.

2.1. Means objectives

While fundamental objectives are the eventual goal for taking an action, a means objective is an objective whose importance stems from its contributions to achieving another objective (e.g. “make my spouse happy” is a fundamental objective, while “arrive home from work early” is a means objective). Whether an objective is a means or a fundamental objective depends on the decision context. Only fundamental objectives should be used to evaluate and compare alternatives; means objectives can be used to create alternatives [11,12]. By mistakenly making decisions based on means-objectives people can make serious errors. Value-focused thinking [11] is a success-

Itinerary

Depart
From: Geneva To: Dublin [DUB] Airport list

Arrival
From: Dublin [DUB] To: Geneva Airport list

Travel dates
Depart: 10 July 2006 noon
Return: 12 July 2006 Morning
 Search only direct or nonstop connections

Flight type and class
Flight type
 Return One Way
Flight class
 Economy Business

Number of passengers
Maximum 9 passengers (adults + children) possible per online booking.
Adults: 1
Children: 0 (from 2nd to completed 11th year)
Infants: 0 (from 0 to 24 months)

Preferred airlines
Select up to 3 airlines
Swiss

[Search](#)

Fig. 4. We suppose that the user prefers to arrive before 5pm in the outbound flight, to arrive before 3pm in the inbound flight and to pay as little as possible. The user has difficulty in translating his preferences into a set of preconstructed possibilities.

SWISS + Lufthansa		Fare per person: 2351 CHF (excl. taxes and fees) Total for all passengers: 2610 CHF (incl. taxes and fees)	
Depart	Arrival	Duration	
Geneva [GVA] 10 Jul 13:35	Dublin [DUB] 10 Jul 21:05	08h 30m / 2 Stops / via Frankfurt Main [FRA] London [LHR] Economy	
Dublin [DUB] 12 Jul 06:45	Geneva [GVA] 12 Jul 12:45	05h 00m / 1 Stop / via Frankfurt Main [FRA] Economy	

[Up to 8 more flights for this airline](#) [Compare Fares](#) [Book this flight](#)

Fig. 5. The result is a very expensive connection provided by Swiss that requires changing in London and arrives at 21:05, thus not satisfying the user preferences.

BRITISH AIRWAYS		Fare per person: 1030 CHF (excl. taxes and fees) Total for all passengers: 1217 CHF (incl. taxes and fees)	
Depart	Arrival	Duration	
Geneva [GVA] 10 Jul 11:55	Dublin [DUB] 10 Jul 18:05	07h 10m / 1 Stop / via London [LHR] Economy	
Dublin [DUB] 12 Jul 08:50	Geneva [GVA] 12 Jul 16:30	06h 40m / 1 Stop / via London [LHR] Economy	

[Up to 32 more flights for this airline](#) [Compare Fares](#) [Book this flight](#)

Fig. 6. By making the same search without the airline preference, the result, although cheaper than in the first case, is still quite expensive and arrives later than the preferred arrival time.




Aer Lingus 			Fare per person: 635 CHF (excl. taxes and fees)
			Total for all passengers: 704 CHF (incl. taxes and fees)
Depart	Arrival	Duration	
 Geneva [GVA] 10 Jul 15:30	Dublin [DUB] 10 Jul 16:40	02h 10m / non stop Economy	
 Dublin [DUB] 12 Jul 11:45	Geneva [GVA] 12 Jul 14:50	02h 05m / non stop Economy	
			Book this flight

Fig. 7. If the user did not state any preference about the arrival time nor about the airline, the tool would have provided a far better solution: a direct flight that is reasonably cheap, arrive in Dublin at 16:40 and returns to Geneva at 14:50.

ful decision-making strategy that concentrates on objectives.

Means-objectives often arise when users are asked about preferences on an aspect of the problem that is not a fundamental objective. They are likely to answer and state a preference that influences the decision process based on a means-objective.

We illustrate how means-objectives can lead to bad decisions with an example considering current search tools based on questions. We anticipate that in general the form-filling approach is an ineffective elicitation strategy, as we will show in our user studies.

The example is based on an actual scenario with one of the largest travel web sites in Europe (as of July 5th, 2006). Our hypothetical user is going from Geneva to Dublin and wants to book a return flight. She has the following preferences:

- for the outbound flight, arrive by 5pm
- for the inbound flight, arrive by 3pm
- she prefers the cheapest option

The website presents a form to the user asking information about travel, departure times, flight type and class, number of passengers and preferred airline.

The user then does the following reasoning:

- “I want to be there by 5 pm” \wedge “leaving around noon, I will probably arrive by 5pm”
→ “I prefer to leave around noon”
- “I want to be back by 3 pm” \wedge “leaving in the morning, I will probably arrive by 3pm” → “I prefer to leave in the morning”
- “I want a cheap flight” \wedge “SWISS is usually cheap” → “I prefer SWISS”

According to her true preferences, the user fills in the form as shown in Figure 4. The system shows a very expensive connection involving Swiss code-

share flights (Figure 5) that requires two changes (in Frankfurt and in London) and arrives at 21:05.

Considering again the same search problem, if the user had not specified the airline preference for Swiss, the displayed option would have been cheaper (Figure 6) but still would not satisfy the preferences about arrival time.

Finally, if the user had not stated any preference regarding airline or departure time, the user would have retrieved the option satisfying all of her desires, shown in Figure 7.

From this example we can see that users that are asked about their preferences may state wrong preferences based on *incorrect means objectives*. In other words, they formulate the real goal (a cheap flight that arrives early) by a “substitute” goal (Swiss flight leaving at noon and returning in the morning) believed to lead to the desired outcome.

From the perspective of logical reasoning, means objectives arise because people perform an abductive reasoning. The inference pattern is “A [usually] implies B” and “I want B”, therefore it seems a good idea to make A true. However, the user’s beliefs are often not accurate and lead to wrong means objectives.

In general, users often state more preferences than necessary when prompted (the preference model might be complete, but not accurate!). We believe that form-filling is ineffective because users do not know how to translate their needs to a set of consistent answers to the questions posed by the interface.

2.2. The prominence effect

The prominence effect is that options that are superior in the most prominent attribute (most important or salient) are preferred more often in choice (asking which of two alternatives is preferred) than in matching (asking a tradeoff value

for a particular value); suggesting that the prominent attribute is weighted more heavily in choice than in matching.

The prominence effect shows that logically equivalent questions can systematically elicit different preferences, thus violating standard models of rational choice. The task of measuring preferences is therefore complicated. Tversky et al. [32] generalized this effect: prominent attributes are weighted more by qualitative tasks (such as choosing) than quantitative ones (such as matching).

2.3. The anchoring effect

In psychology and behavioral decision theory, anchoring is an experimental result in which an uninformative element influences the judgments. Tversky and Kahneman (1974) proposed to evaluate this heuristic with a paradigm in which participants are given an irrelevant number and asked if the answer to a question is greater or less than the value. Anchors have been found to influence many judgment tasks, including answers to factual knowledge questions, estimation of risk and uncertainty, statistical inferences, evaluation of monetary lotteries, partner selection, and predilections of future performance; several studies have found that anchoring occurs even when the anchors are extreme or are an implausible response to the question [6]. Anchoring plays a slightly different role in some situations: when asked questions about information that they do not know, people may spontaneously anchor on information that comes to mind and adjust their responses in a direction that seems appropriate, using the “Anchoring and adjustment procedure” [8]. This heuristic is often helpful, but the final estimate might be biased toward the initial anchor value [9].

Example-based tools have the great advantage of showing instances of real options. The user can quickly increase familiarity and awareness with the subject by considering concrete examples.

In example-critiquing, we expect the search to become more and more accurate as long as the critiques supplement the preference model. Unfortunately, the examples shown can also have the anchoring effect. If a user is looking for a notebook computer, he might start looking for a low price. Assume that all cheap models weigh about 3 kg. If only those are presented, they will all have about the same weight, and it may not occur to her to

look for lighter models. The influence of the current examples prevents the user from refocusing the search in another direction.

3. Effective Interaction Principles

Preference-based search occurs in a tradeoff of decision accuracy, decision confidence, and the effort expended by the user. The goal is to make the decision accuracy and the confidence of the user as high as possible given the amount of effort the user is willing to exert. At the same time, the design must take the characteristics of human decision-making into account. This means ensuring that means objectives, the prominence and anchoring effects do not drive the decision process towards an incorrect solution.

We propose that user interaction should incorporate the following principles to achieve this. Later in the paper, we will then show how they can be satisfied in tools based on example-critiquing with suggestions.

User in control The user should be free to state preferences on any attribute of the problem at any time [19]. Users should not be forced or induced to answer questions about preferences they do not possess yet; on the other hand, the tool should support the process of preference *construction*. It should also be possible for the user to revise her stated preferences. This is important to avoid means objectives. For example, Chen and Pu [7] compared user-motivated critiques with system-generated dynamic critiquing, showing that users are more confident with tools that allows critiques of the first kind.

Allow partial satisfaction By considering the possibility that preferences can be partially satisfied, the system can avoid the situations where no result is returned at all, and thus reduces user effort. Moreover, it can provide the user with the possibility of reasoning about trade-offs.

Immediate Feedback By giving an immediate feedback to the user’s preferences, the system can increase the familiarity of the user with the decision context. The user can understand the implication of her preferences on the retrieval. In case she has stated a preference that she

does not actually have, the new set of displayed examples can induce her to revise the previously stated wrong preferences.

Suggestions (alternative recommendations) To avoid the anchoring effect and teach the user about the possible options, alternative recommendations or suggestions should be shown that can stimulate preference expression. If certain preferences are missing from the current model of the user, the system may provide solutions that do not satisfy those unknown preferences. If the user is aware of all of her preferences, she can realize the necessity to state them to the system. However, this is not usually the case, because the user might not know all the available options.

Elicit explicit preferences on attributes Preferences can be elicited either implicitly by asking for comparisons between options, or explicitly by asking for general preferences on attributes. Comparisons imply a heavier cognitive load on users, since they involve complex trade-offs between different attributes, and this is part of the cause of the prominence effect. Instead, preferences should be stated on attributes. This counters the prominence effect, and increases decision accuracy and confidence while reducing effort.

4. Design of preference-based search

Example-critiquing has been proposed by various researchers, starting with the ATA system of Linden et al. [14] and SmartClient [21], and more recently with incremental critiquing [15]. Lately, with the emergence of ubiquitous computing, mobile recommender systems have been implemented [25].

In another form of example-critiquing, search proceeds as *navigation* from one example to another, where the current example is considered to be the most preferred at that point. Critiques of the current example then lead to another example that improves on the aspect that the user has critiqued while being as similar as possible to the current example. Performance can be improved by providing users with compound critiques [24]. This form of example-critiquing was first proposed in the FindMe systems [4,3], and more recently was

also used in the ExpertClerk system [26] and in dynamic critiquing [16].

In some works [36] the navigation allows users to look for products similar to one they like (*show me more like this*) posing the challenge of defining a suitable metric to identify similar products as many real catalogs contain unstructured information.

In our approach we consider an example-critiquing framework with an explicit preference model where

1. Preferences are stated as reactions to displayed options (as “The price should be less than 600”). Such critiques are user-motivated.
2. Preferences are internally represented as *soft constraints*, a formalism that allows partial satisfaction.
3. These critiques are used as feedback in the next interaction cycle to generate a new set of displayed items.

These design choices are consistent to the principle expressed before in Section 3. Later in this section we introduce *suggestions*, in order to implement the last principle.

4.1. Example

To illustrate the importance of implementing preference-based search according to the previously stated principles (Section 3), consider the problem of selecting a flight among the following set of options:

	fare	arrival	airport	airline
o_1	250	14:00	INT	B
o_2	300	9:00	INT	A
o_3	350	17:30	CITY	B
o_4	400	12:30	CITY	B
o_5	550	18:30	CITY	B
o_6	600	8:30	CITY	A

For simplicity, assume that options are modelled by just 4 attributes: fare, arrival time, departure airport, and the airline.

Initially, the user starts with a preference for the lowest price. Assume that the user also has two other, hidden preferences:

- arrive by 12:00;
- leave from the CITY airport, which is much closer than the INTernational airport.

The best choice for this user will be o_4 which gives a reasonable tradeoff among the objectives.

In a form-filling or wizard approach, the tool might also ask the user for a preference on the airline. Even though she does not have any preference on the airline, she might believe that airline A operates more flights from the CITY airport and thus formulate a *means objective* for airline A. When this preference is added to the others, it now makes options o_2 or even o_6 the most attractive, thus keeping the user from finding her best choice. We believe that this effect is responsible for the poor performance of form-filling in the user studies we report later in this paper.

Now consider an example-critiquing approach. Given the initial preference on price, the tool will start by showing only option o_1 .

In a navigation without an explicit preference model, the tool uses the current best example as the starting point and finds the one that is closest to it while also satisfying the user's critique. In this case, the user might first critique the fact that the arrival time is too late, leading to o_2 , and then the fact that the departure is from the INTERNATIONAL airport, leading to o_6 as the most similar option leaving from the CITY airport. Now the user might critique the fact that this option is too expensive, and get back to o_2 . In this cycle, the best option, o_4 , will never be discovered, since it requires a tradeoff among the different preferences. The importance of such tradeoffs has been pointed out for example in [20].

In a tool with an explicit preference model, o_4 will be determined as the best option once both hidden preferences have been added.

However, the user might need to be motivated to state the preferences about airport and arrival time by seeing suggestions that show the availability of these opportunities. This problem of stimulating preference expression with suggestions is discussed in the next section; we will return to this example later.

4.2. Introducing suggestions

If certain preferences are missing from the current model of the user, the system may provide solutions that do not satisfy those unknown preferences. If the user is aware of all of her preferences, she realizes the necessity to state them to the system. However this is not usually the case,

because the user might not know all the available options. Moreover, stating a preference costs some user effort and she would make that effort only if she perceives this as beneficial.

To enable the user to refocus the search in another direction (avoiding the *anchoring effect*), many researchers have suggested displaying alternatives or diverse examples, in addition to the best options (candidates). In one user study it was observed that the majority of critiques (79%) were a reaction to seeing an additional opportunity rather than seeing unsatisfactory examples [34].

Different strategies for suggestions have been proposed in the literature. Linden [14] used extreme examples, where some attribute takes an extreme value. Others use diverse examples as suggestions [28,29,26].

An extreme or diverse example might often be an unreasonable choice: it could be a cheap flight that leaves in the early morning, a student accommodation where the student has to work for the family, or an apartment extremely far from the city. Moreover, in problems with many attributes, there will be too many extreme or diverse examples to choose from, while we have to limit the display of examples to few of them.

The user should be motivated to state new preferences by options that are reasonable choices (given the previously stated preferences) and have a potential of optimality (a new preference is required to make them optimal).

This is expressed by the *lookahead principle* [23]:

Suggestions should not be optimal under the current preference model, but should provide a high likelihood of optimality when an additional preference is added.

Model-based suggestions are calculated based on that principle. Results show that such suggestions are highly attractive to users and can stimulate them to express more preferences to improve the chance of identifying their most preferred item by up to 78% [35].

We propose to implement the lookahead principle using Pareto-optimality as the optimality concept. An option is Pareto-optimal if there is no other option that is better or equally preferred with respect to all the preferences and strictly better for at least one preference. The advantage of this concept is that it is qualitative, since it does not rely on any particular parameterization. To

Flat Finder - Example Critiquing

Preferences

Price ? - (importance)+ ○ ○ ● ○ ○ Remove Bathroom ? - (importance)+ ○ ○ ● ○ ○ Remove

Search according to these preferences:

Add preferences

Results

There are a total of **187** options, of which **8** fully match your preferences.

These are the best solutions that match your query.

ID	Type	Price	Rooms	Furnished	Smoking	Bathroom	Kitchen	Transportation	Distance to Uni	Distance to Centre	Choose
8142	room in a house	300	1.0	true	non smoking	private	shared	bus	10	12	<input checked="" type="radio"/>
7849	room in a house	400	1.0	true	either	private	shared	none	18	16	<input type="radio"/>
8080	room in a house	410	1.0	true	either	private	not available	bus	19	19	<input type="radio"/>

In the dataset you can also find..

ID	Type	Price	Rooms	Furnished	Smoking	Bathroom	Kitchen	Transportation	Distance to Uni	Distance to Centre
8122	studio	420	1.0	false	either	private	private	bus	7	5
8046	shared apartment	450	1.0	true	either	shared	shared	bus	9	8
8027	shared apartment	400	1.0	true	either	shared	shared	metro	12	6

Look at the solutions displayed. If you realize that you did not stated some of your preferences you can do it now.

State an additional preference

My Basket

Here you can store entries for comparison. When you choose one of them, you can proceed to checkout

No element in that set

Copyright Artificial Intelligence Laboratory -EPFL. Version 1.2 Last update 29/05/2005.

Fig. 8. The FlatFinder example-critiquing interface.

make an option Pareto-optimal, a new preference has to make this option escape the dominance with better solutions (the “dominators”).

An example of a conversational recommendation system is FlatFinder, a tool for finding student accommodation through example critiquing, using data available from the student housing office (containing around 200 items). Figure 8 shows an example of an interaction with FlatFinder. The user has posted preferences for a cheap accommodation (price less than 450) with a private bathroom. The tool shows 3 candidate options (satisfying the preference for price and bathroom) which are all rooms in a family house, with poor public transit service. It also displays 3 suggestions, which show that by paying a bit more, the user can have a studio or a small private apartment which are both closer to the university and the centre of town; the second also is close to a subway (metro) station. The last option is still a room but it is closer to the center than the options currently showed as best options. Analyzing the suggestions, the user could realize that she cares about these features,

and then state a preference on the transportation, for the accommodation type or on distance from centre and university.

4.3. Example (continued)

Consider again the problem of choosing a flight. We compare how different strategies would select options as suggestions and their effectiveness in stimulating the user to state her other preferences.

Consider first the strategy of suggesting options with extreme attribute values proposed by Linden et al. [14]. For the departure time, o_5 is the earliest and o_6 the latest departure. Are these good suggestions to make the user understand the opportunities offered by the available options? Let’s consider that:

- unless the arrival has to be after 17:30, o_3 is a much cheaper, thus more attractive option than o_5 .
- unless the arrival has to be before 9:00, o_2 is much more attractive than o_6 .

Showing o_5 and o_6 as suggestions would give the user an impression that a different arrival time implies significant compromises on her preference for low cost, which in reality is not true. Thus, she might never state her preferences on time given these suggestions.

The strategy of showing maximally diverse options gives the same result: as o_1 is the best option, the options that are most different are again o_5 and o_6 . We have already seen that these are not the best to motivate preferences on arrival time, but they are not motivating for other attributes either:

- if the user prefers the CITY over the INTERNATIONAL airport, o_3 is a much cheaper option that already allows this.
- if the user prefers airline A, o_2 allows this at a much lower cost.

The model-based suggestion strategy (whose technical implementation will be discussed later in Section 5) takes this observation into account by evaluating options with respect to others that might dominate them.

4.4. Adaptive elicitation

The acquisition of the user model might be facilitated if the system could anticipate the behavior of the user given observations of their past behavior. Recently, several researchers followed this intuition by considering an adaptive strategy for utility-based elicitation and question-answering interface.

Chajewska et al. [5] proposed an elicitation procedure that models the uncertainty of the utility function and selects assessment questions to maximize the expected value of information. Boutilier [2] has extended this work by taking into account values of future questions to further optimize decision quality while minimizing user effort. The elicitation procedure itself is optimized based on a partially observable Markov decision process (POMDP) where the reward of a given choice is discounted by the cognitive cost of the elicitation questions.

Stolze considers how to optimize an online question/answer interface, by adapting the questions taking into account the distribution of possible answers and solving a decision tree optimization problem [30].

Motivated by these works, we improve the suggestion generation component of our example critiquing tool by taking into account the reactions of the user to the shown example, i.e. whether or not the user was stimulated to state a preference. The belief distribution over the preferences is updated with a Bayesian approach.

In our implementation, the preference statements acquired by the user’s critiques are translated into soft constraint represented by parametric cost functions (see Section 5). Given this factorization, the update of the beliefs can be implemented efficiently for many realistic situations, and the computational complexity does not depend on the number of available options (in contrast with [5,2]).

5. Example-critiquing with Adaptive Suggestions

In this section we present example-critiquing with adaptive suggestions, an implementation of preference-based search that complies with the principles presented in Section 3 following the discussions outlined in Section 4.

5.1. Basics

We assume that O is the set of options o_1, \dots, o_n defined over A , set of attributes $\{a_1, \dots, a_n\}$ with domains D_1, \dots, D_n ; $a_i(o)$ is the value that o takes on attribute a_i .

Domains can be qualitative or numeric. A **qualitative domain** (such as colors or names) consists of an enumerated set of possibilities; a **numeric domain** has numerical values (as price, distance to center), either discrete or continuous. For numeric domains, we consider a function $range(Att)$ that gives the range on which the attribute domain is defined. We define qualitative (respectively numeric) attributes those with qualitative (numeric) domains.

Preferences are stated on individual attributes. A preference r_i applies to a particular attribute a_i and results in a total or partial order on the values in the domain D_i of a_i .

A preference model R consists of a set of preferences. We assume that a preference r_i is expressed by a cost function c_i . Since a preference always ap-

plies to the same attribute we can use $c_i(o)$ instead of $c_i(a_i(o))$.

We assume that the cost functions correctly express the user's *ceteris paribus* preferences, i.e. that for any pair of options o_1 and o_2 that are identical in all preferences except preference r_i , the user prefers o_1 over o_2 if and only if $c_i(s_1) < c_i(s_2)$.

The individual costs obtained by each preference are merged with a particular combination function, for example a weighted sum.

$$C(o) = \sum_i w_i * c_i(o) \quad (1)$$

The *candidate* best options can be found by sorting the database items according to their cost. This is known as the *top-k query* [10]. The set of options retrieved $\{o_1, \dots, o_k\}$ is such that $C(o_1) \leq C(o_2) \leq \dots \leq C(o_k)$ and for any other option \bar{o} in the database $C(\bar{o}) \geq C(o_k)$.

5.2. Pareto-dominance and optimality

We model preferences by standardized functions that correctly reflect the preference order of individual attribute values but may be numerically inaccurate.

Pareto-optimality is the strongest concept that can be applied without knowledge of the numerical details of the penalty functions.

An option o is **(Pareto) dominated** by another option \bar{o} (equivalently we say that \bar{o} dominates o) if

- \bar{o} is not worse than o according to all preferences in the preference model: $\forall c_i \in R : c_i(\bar{o}) \leq c_i(o)$
- \bar{o} is strictly better than o for at least one preference: $\exists c_j \in R : c_j(\bar{o}) < c_j(o)$

An option o is **Pareto-optimal** if it is not Pareto-dominated by any other option.

The **dominating set** $O^+(o)$ is the set of options that dominates the option o . The **equal set** $O^=(o)$ is the set of options that are equally preferred with o .

Proposition 1 *A dominated option o with respect to R becomes Pareto-optimal with respect to $R \cup r_i$ if and only if o is*

- *strictly better with respect to r_i than all options that dominate it with respect to R and*

- *not worse with respect to r_i than all options that are equally preferred with respect to R .*

Thus, the dominating set $O^>$ and the equal set $O^=$ of a given option are the potential dominators when a new preference is considered.

5.3. Assumptions about the cost functions

We suppose that the preference value function can be written in a parameterized form, known to the system. The uncertainty is on the actual value taken by the parameters. Here we assume a single parameter θ , but the method can be generalized to handle cases of multiple parameters.

$$c_i = c_i(\theta, a_i(o)) = c_i(\theta, o) \quad (2)$$

We use $r_{i,\bar{\theta}}$ to denote the preference on attribute i with parameter $\bar{\theta}$, corresponding to the cost function $c_i(\bar{\theta}, o)$.

For simplicity, we let the parameter θ represent the *reference point* of the preference stated by the user.

Consider a preference statement like “I prefer black cars”. It can be represented by the following cost function c_i considering the attribute a_i being the color and $\theta = \text{black}$.

$$c_i(\theta, x) \equiv \mathbf{if } a_i(x) = \theta \mathbf{ then } 0 \mathbf{ else } 1. \quad (3)$$

For numeric domains we consider that the user can choose between a fixed set of qualitative statements like:

- **LowerThan**(θ): the value should be lower than θ
- **Around**(θ): the value should be around θ
- **GreaterThan**(θ): the value should be greater than θ

We suppose that for a given implementation we will choose a particular form in order to represent these statements. The designers of the application will consider the kind of preference statements that the user can state through the user interface and the way to translate the statements into quantitative cost functions. A similar approach is taken by Kiessling in the design of **PREFERENCE SQL** [13], a database system for processing queries with preferences.

A reasonable possibility is to represent the cost functions for numeric domains as graded step func-

tions, with the penalty value increasing from 0 to 1. In case where the degree of violation can worsen indefinitely, a ramp function can be used. For example, in a system for flight reservations where the user states that she prefers to arrive before a certain hour (θ), a possible function is the following:

$$c_i(\theta, x) = \begin{cases} (x - \theta) & \text{if } a_i(x) > \theta \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

5.4. Prior distribution

Suppose we have a prior probability distribution over the possible preference models, learnt from previous interactions with the system. We consider

- p_{a_i} the probability that the user has a preference over an attribute,
- $p_i(\theta)$ the distribution of probability over the parametric value of the preference representation for the cost function c_i ,
- $p(r_{i,\theta})$ the probability that the user has a preference on attribute i and its parameter is θ . We assume it to be $p_{a_i} * p_i(\theta)$.

Such distribution will be updated by the user during the interaction based on the user's action.

5.5. Suggestions

According to our look-ahead principle, we choose suggestions that have the highest likelihood of becoming optimal when a new preference is added. Since we would like to avoid sensitivity to the numerical error of the cost functions, we use the concept of Pareto-optimality.

In this section we show how to compute the probability that a given option becomes Pareto optimal (breaks all dominance relations with options in its dominating set). These formulas depend on the probability distributions p_{a_i} and $p_i(\theta)$ that we have introduced before. At the beginning of the interaction, they are initialized by prior knowledge, considering the preference models of previous users. During the interaction, they are updated according to the observations of the user's actions.

5.5.1. Probability of escaping attribute dominance

We first consider the probability of breaking one dominance relation alone. To escape dominance, the option has to be better than its dominator with respect to the new preference.

The probability $\delta_i(o_1, o_2)$ that a preference on attribute i makes o_1 be preferred to o_2 can be computed integrating over the values of θ for which the cost of o_1 is less than o_2 .

$$\delta_i(o_1, o_2) = \int_{\theta} H(c_i(\theta, o_2) - c_i(\theta, o_1))p(\theta)d\theta$$

where H is the function $H(x) \equiv \mathbf{if}(x > 0) \mathbf{then} 1 \mathbf{else} 0$.

For a qualitative domain, we iterate over θ and sum up the probability contribution of the cases in which the value of θ makes o_1 preferred over o_2 .

For breaking the dominance relation with all the options in the dominating set when a new preference is set on attribute a_i , all dominating options must have a less preferred value for a_i than that of the considered option. For numeric domains, we have to integrate over all possible values of θ , check whether the given option o has lower cost, and weigh the probability of that particular value of θ . Thus we have the probability $\delta_i(o, O^{\geq})$ of simultaneously breaking all dominance relations

$$\delta_i(o, O^{\geq}) = \int [\prod_{o' \in O^>} H(c_i(\theta, o') - c_i(\theta, o)) \prod_{o'' \in O^=} H^*(c_i(\theta, o'') - c_i(\theta, o))] p(\theta) d\theta$$

where H^* is a modified Heaviside function that assigns value 1 whenever the difference of the two costs is 0 or greater. ($H^*(x) \equiv \mathbf{if}(x \geq 0) \mathbf{then} 1 \mathbf{else} 0$).

For qualitative domains, simply replace the integral with a summation over θ .

In general the cost functions in use for a particular system allow substantial simplification of the calculations of the integrals. Considering the cost function

$$c_i(\theta, x) = \mathbf{if} a_i(x) = \theta \mathbf{then} 0 \mathbf{else} 1 \quad (6)$$

the probability of breaking a dominance relation between option o_1 and o_2 simplifies to the probability that the value of option o_1 for attribute i is

the preferred value, when it differs from the value of o_2 .

$$\delta_i(o_1, o_2) = \begin{cases} p(\theta = a_i(o_1)) & \text{if } a_i(o_1) \neq a_i(o_2) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

5.5.2. Probability of becoming Pareto-optimal

To make an interesting suggestion, it is necessary that an option has a chance of becoming Pareto-optimal, i.e. escape dominance in at least one of the attributes in the set A of all attributes modeling the options. This is given by the combination of the events that the user has a preference on a particular attribute a_i , and the event that this preference will break all attribute dominance in that attribute:

$$p^{opt}(o) = 1 - \prod_{a_i \in A} (1 - p_{a_i} \delta_i(o, O^{\geq}(o))) \quad (8)$$

where p_{a_i} is the probability that the user has a preference over attribute a_i .

To generate suggestions that are adapted to the user, we update the value p_{a_i} according to the user's actions. The computation of δ_i depends on $p_i(\theta)$, that is also updated according to the user's behavior.

Following the lookahead principle, the options for which p^{opt} is greatest are chosen as the best suggestions

5.6. Example (continued)

We return to our running example about flights. Here we show how the model-based suggestion strategy takes into account the information about available flights, the current preference model and the uncertainty over possible *hidden* preferences that the user has not stated yet. The following table shows the individual probability δ_i of escaping dominance with respect to each individual attribute, as well as the overall probability p of breaking dominance, assuming that the user has a probability P_{a_i} of 0.5 of having a preference on each of the three attributes that do not yet have a preference. The preference for arrival time is assumed to be a *step* function with a reference value that varies in a range of 10 hours; the reference value is assumed to be uniformly distributed.

	fare (a_1)	arr (a_2)	δ_2	airport (a_3)	δ_3	airline (a_4)	δ_4	p
o_1	250	14:00	-	INT	-	B	-	-
o_2	300	9:00	0.5	INT	0	A	0.5	0.437
o_3	350	17:30	0.35	CITY	0.5	B	0	0.381
o_4	400	12:30	0	CITY	0	B	0	0
o_5	550	18:30	0.1	CITY	0	B	0	0.05
o_6	600	8:30	0.05	CITY	0	A	0	0.025

We can see that now, options o_2 and o_3 are considered to be the best suggestions. This is due to the fact that they are quite competitive given the known preference on price, and show the availability of other attribute values without unduly compromising the already known objectives. Thus, they are most likely to stimulate the additional preference expression that we require.

Note that the best option - o_4 - is not among the suggestions. However, it will become the highest-ranked option once the hidden preferences have been stated. This illustrates that the function of suggestions is to motivate additional preferences, but not necessarily to provide the best choices by themselves.

5.7. Adaptive Suggestions

Model-based suggestions can become very effective if they can adapt to the user, responding to her actions. In particular, after the user has made a query and some example options have been shown (*candidates* and *suggestions*), the user might or might not state an additional preference. Observing the reaction of the user to the examples, the system can refine the uncertainty over the preference model. Suggestions stimulate the expression of those preferences on the values that are shown; therefore, if the user does not state any preference, the likelihood that there is a preference on those values will decrease.

After each interaction cycle, we need to update the p_{a_i} and $p(\theta_i)$ for all attributes i , on the basis of the user's reaction to the shown example.

Suppose we know a model of the user's reaction behavior, composed of the following probabilities (that refer to the situation in which at least one relevant option is displayed).

- $p(st|r_{i,\theta})$ the probability that the user **states** a preference given that the preference **is** in the user's preference model
- $p(st|\neg r_{i,\theta})$ the probability that the user **states** a preference given that the preference **is not** in the user's model

We will expect the second to be relatively small: from our experience in the user tests, users of example-based tools state preferences only when these are relevant (in contrast to the form-filling approach, where users are more likely to state preferences they do not have).

We use the Bayesian rule to update the probability of a particular preference being present in the user model, in the condition that a relevant option is presented to the user in the set of displayed examples. The probability that a preference on attribute i with parameter θ is present when the user has stated or not stated any critique is the following:

$$\begin{aligned} p(r_{i,\theta}|st) &= \frac{p(st|r_{i,\theta})p(r_{i,\theta})}{p(st|r_{i,\theta})p(r_{i,\theta}) + p(st|\neg r_{i,\theta})p(\neg r_{i,\theta})} \\ p(r_{i,\theta}|\neg st) &= \frac{p(\neg st|r_{i,\theta})p(r_{i,\theta})}{p(\neg st|r_{i,\theta})p(r_{i,\theta}) + p(\neg st|\neg r_{i,\theta})p(\neg r_{i,\theta})} \end{aligned} \quad (9)$$

where $p(\neg r) = 1 - p(r)$.

Once we know the new value for $p(r_{i,\theta})$, we can compute new values for p_{a_i} and $p_i(\theta)$ that will be used to generate suggestions at the next interaction cycle.

$$p_{a_i} = \int_{\theta} p(r_{i,\theta}) d\theta \quad (10)$$

$$p_i(\theta) = p(r_{i,\theta}) * (1/p_{a_i}) \quad (11)$$

The new value for p_{a_i} , the probability that there is a preference over a particular attribute, is obtained by the integrating the joint probability $p(r_{i,\theta})$ over θ . The the parametric distribution $p_i(\theta)$ is obtained dividing the joint probability $p(r_{i,\theta})$ by the attribute probability p_{a_i} .

We now need to define $p(st|r_{i,\theta})$. In a user study it was observed that the majority of critiques (79%) were a reaction to seeing an additional opportunity rather than seeing unsatisfactory examples [34], providing evidence for the correctness of the lookahead principle. Therefore, we assume a probabilistic model of the user coherent with the lookahead principle

$$p(st|r_{i,\theta}) = p^{opt}(o) \quad (12)$$

where o is the option displayed and $p^{opt}(o)$ the estimated probability of optimality. We think that this is a cautious approach: in many real situations the user will state a preference when a new opportunity is shown, even if optimality is not achieved.

Given these assumptions, any time that an option is shown and the user does not critique a given preference, the new value for the probability that the preference is present $p^{new}(r_{i,\theta}) = p(r_{i,\theta}|\neg st)$ can be written as:

$$\begin{aligned} p^{new}(r_{i,\theta}) &= p(r_{i,\theta}) \frac{1 - p^{opt}(o)}{[1 - p^{opt}(o)]p(r_{i,\theta}) + [1 - p(r_{i,\theta})]} \\ &= \frac{p(r_{i,\theta}) - p^{opt}(o)p(r_{i,\theta})}{1 - p^{opt}(o)p(r_{i,\theta})} \end{aligned}$$

The belief update depends on $p^{opt}(o)$, that is an estimation of the quality of the suggestion. This means that if the system shows an option that has no chance of becoming optimal ($p^{opt}(o) = 0$) then $p(r)$ will remain unchanged. Instead if $p^{opt}(o) = 1$ (an ideal suggestion is shown), $p(r)$ will go to 0 if no reaction is observed.

5.8. Implementing the belief update

At each cycle of the conversational recommender system, the probability distribution for possible hidden preferences is updated.

Given the assumptions about the type of cost functions, the algorithm for updating the probability distribution is straightforward: as the parameter θ represents the “reference value” of the preference (the most preferred value for qualitative domains; the extreme of acceptable values for numeric domains) we need to consider the values that the shown examples take on each of the attributes and update the probability of a preference with such value as reference.

For each of the displayed options o and for each of the attributes a_i , we update the probability $p(r_{i,a_i(o)})$, where $\theta = a_i(o)$, conditioned on whether the user has stated an additional preference on i or not, using the Bayes formula as we have shown in the previous paragraph.

The computation time of the belief update does not depend on the number of available options in the database (this was the case in some Bayesian utility-based recommenders [5,2]), but only on the (constant) number of options shown at each cycle, so the belief update can scale to large databases.

Instead, the computation load of the belief representation and update is heavily influenced by the kind of preferences the user is allowed to state by means of critiquing. In the simple case where preferences are of the form “I prefer θ to anything else” the belief distribution can be compactly represented by an array expressing for each value the probability of being the most preferred and the

belief update is linear in the size of the domain. In the general case where the user can express complex preferences consisting of an arbitrary long chain of values with decreasing preference (e.g. “I prefer d_1 to d_2 , d_2 to d_3 , d_3 to d_4 , ..”, with values $d_1, d_2, d_3, d_4, .. \in D$, the attribute domain) or even partial orders as lattices, the complexity of the probability distribution becomes exponential in the cardinality of the domains. Many practical situations have attributes with discrete and limited values, or preferences that take some compact form, so the belief update can be performed reasonably fast. In the other cases the probability distribution can be approximated; for instance in continuous domains it is possible to approximate the probability distribution by partitioning the domain in several intervals.

In general given a particular application the designer will have to make a compromise between flexibility (with respect to the degree of choice of possible critiques) and precision (with respect to the computation of the value of the recommendations).

6. Evaluation

In this section we present evaluations made with simulation and user studies. We evaluate the performance of adaptive suggestions using prior knowledge and Bayesian reasoning.

We consider:

- **model-based suggestions (suggestions)** as the suggestions technique presented above with uniform probability distribution of preferences and no Bayesian update,
- **adaptive suggestions** as the model-based suggestions with prior knowledge (learned from previous users) and the Bayesian update performed at each cycle.

Simulations are used to compare adaptive suggestions with standard model-based suggestions and classic suggestion techniques. Since the suggestion strategies we presented are heuristic and user studies are expensive, simulations have the advantage of comparing different techniques in great detail under the assumptions underlying their design.

Two experiments with real users were carried out. In the first (supervised) we compared

example-critiquing without suggestions, example-critiquing with non-adaptive model-based suggestions (uniform distribution of the preferences is used) and a form-filling interface. In the second experiment, unsupervised, the example-critiquing application was made available on line; in this way we could compare different approaches (form-filling, example-critiquing with standard suggestions, example-critiquing with adaptive suggestions).

6.1. Evaluation of adaptive suggestions with simulations

We evaluated our adaptive strategy of generating suggestions with simulations. Assuming that our look-ahead strategy is correct, we look at the effectiveness of the suggestions in stimulating preference expression.

To obtain realistic prior distributions we considered logs from previous user studies using the FlatFinder tool. A total of 100 interaction logs, each constituting a complete search process, were considered.

For the adaptive suggestions we need an estimate for the probability that the user states a preference given that an option is shown (in the Bayesian update formula, Equation 9). We substitute this with the value of the estimated probability of becoming pareto optimal, according to our look-ahead principle.

6.2. The effect of prior knowledge

	hit-rate
prior knowledge	87%
model based suggestions	61%
diversity strategy	25%
extreme strategy	13%

Table 1

The average number of discovered preferences. We compare the model-based suggestions with prior knowledge, the standard model-based suggestion strategy assuming uniform distribution, the strategy of maximizing diversity and the extreme strategy.

We ran simulations in two settings. In the first, we used the logs to draw random preference models and measure the percentage of time that the look-ahead principle is satisfied (an option that is selected by the system as a suggestion becomes

Pareto optimal). We call this measure the *hit rate*. We compare different strategies of suggestions: model-based suggestions using prior knowledge, model-based suggestions assuming uniform distribution, the *extremes* strategy (suggesting options where attributes take extreme values, as proposed in [14]), the *diversity* strategy (computing the 20 best solutions according to the current model and then generating a maximally diverse set of 5 of them, following the proposal of [17]).

Considering the logs of previous interactions with the tool, we set up the value for $p(r_{i,\theta})$, the probability that the user has a preference on attribute i and its parameter is θ , proportional to the occurrence of the preference $r_{i,\theta}$.

The results (Table 1) show that prior knowledge of the distribution of the preferences can greatly help in making better suggestions (87% against 61%).

6.3. Simulating a real user

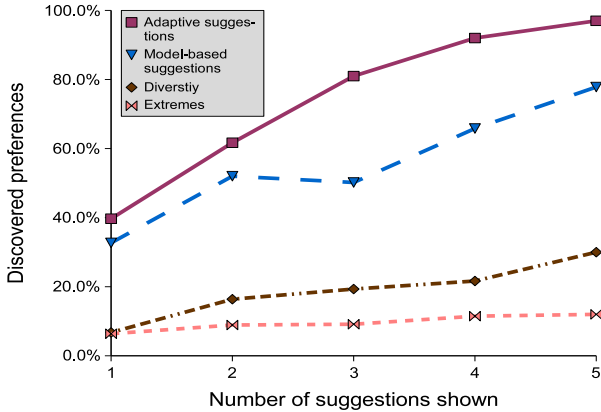


Fig. 9. Evaluation of adaptive suggestions, measuring the number of preferences discovered with simulations, with respect to basic model-based suggestions. Adaptive suggestions perform significantly better. Diversity and extreme strategies are also shown.

To better evaluate suggestions in a more realistic scenario, we considered the simulation of an interaction between a simulated user and the example-critiquing tool.

In the simulations, we set $p(st|\neg r_{i,\theta}) = 0$, meaning that the user states only preferences that she really has. This is reasonable because the user can only state preferences on her own initiative. Therefore, $p(\neg st|\neg r_{i,\theta}) = 1$.

	number of suggestions shown			
	1	2	3	4
<i>full discovery</i>				
adaptive suggestions	32%	53%	72%	88%
prior knowledge	32%	46%	58%	85%
basic suggestions	30%	44%	42%	53%

Table 2

The percentage of full discovery (interactions that acquire a complete preference model) in the simulation with probabilistic profile acquired from real logs. We compare the adaptive model-based suggestions, model-based suggestions with prior knowledge and the standard model-based suggestion strategy assuming uniform distribution. Adaptive suggestions manage to discover the complete set of preferences almost all the time.

To handle the probability distribution of continuous attributes, we discretized the domains in few intervals, because otherwise the probabilistic update would be untractable.

We split the data (100 logs of user interactions) in two sets.

1. A learning data-set, used to represent prior knowledge to feed the the adaptive tool.
2. A test data-set, used to generate preference models of simulated users.

We ran several simulations with a random split of samples between the learning and the test data-set.

In the simulations, users have a set of preferences generated according to a particular probabilistic distribution. The simulated user behaves according to an opportunistic model by stating one of its hidden preferences whenever the suggestions contain an option that would become optimal if that preference was added to the model with the proper weight.

Every step of the simulation represents an interaction with the recommender system. Based on the preferences that are known to the system a set of candidates and suggestions are retrieved at each step. When the suggestions retrieved by the strategy satisfies the look-ahead principle, a new preference is stated and considered in the user model. In this case, we say that a new preference is discovered and the suggestions were appropriately chosen.

According to our user studies, on average the users interact on average for 6.25 cycles, of which in 3.30 cycles the users did not either remove nor add preferences. We implemented the simulation

so that the interaction continues until either the user model is complete or the simulated user states no further preference twice in a row (choosing a prudent approach).

If all preferences are stated, the preference model is complete (a *full discovery* of the preferences). In this case, it is guaranteed that the user can find the target (his true best option) by a series of trade-off actions that modify the relative importance of the preferences.

The results are shown in Figure 9 and Table 2. We compare the basic formulation of model-based suggestions (assuming uniform distribution), model-based suggestions with prior knowledge and adaptive suggestions. We measure the fraction of preferences discovered.

Even with a simple probabilistic model of the user, the gains obtained by adaptive suggestions become considerable, especially when 3 or more suggestions are shown (72% of runs achieve full discovery for adaptive suggestions, 58% prior distribution, 42% normal suggestion). Adaptive suggestions are better than suggestions generated according to prior distribution. It is important to note that the performance of suggestions with prior knowledge degrades with respect to the first simulation setting.

It would be interesting to compare the effects of different choices for the value $p(st|r_{i,\theta})$ of the probabilistic behavior of the user.

6.4. Evaluation with user studies

We present the user studies we set up to test our approach and compare it to question-answering elicitation.

1. Supervised study comparing

- (a) a form-filling interface similar to that used in existing web sites, shown in Figure 12. The form contains all preferences that can be used when searching for housing, and the tool returns 6 options which best fit the preferences;
- (b) the example-critiquing interface (**EC**), where preferences are stated on the user's initiative. At each step the tool returns 6 best options according to the current preferences;

- (c) the same example-critiquing interface (**EC with suggestions**), but returning the 3 best options and 3 best suggestions at each cycle according to the standard model-based suggestion strategy (with standard probabilities and no bayesian update).

2. Unsupervised user study comparing

- (a) the same form-filling interface acquiring the preferences through questions
- (b) the example-critiquing with standard model-based suggestions (**EC with suggestions**) (3 best options and 3 best suggestions at each cycle)
- (c) example-critiquing showing adaptive model-based suggestions (**EC with adaptive suggestions**) using prior knowledge and performing a bayesian update at each cycle (3 best options and 3 adaptive model-based suggestions at each cycle)

For each tool considered we measured decision accuracy, defined as the percentage of users who actually found their most preferred option with the tool.

We stress that the form-filling interface allows partial satisfaction of the preferences, so that a set of results is always retrieved.

6.4.1. Evaluation of Example-critiquing and form-filling (supervised experiment)

The between-group experiment used 20 users in each of the three groups. They were students looking for new or better housing and thus were very motivated to carry out the experiment.

The subjects first selected the most preferred options using their respective version of the FlatFinder tool. For the first group we recorded both the best option selected in the first use, after iterated use of the form-filling. At the end of the experiment, we asked the subjects to carefully go through the entire list of available options (about 150) and select their real preferred choice, a process that took them on average about 30 minutes. We can thus measure decision accuracy as the fraction of times that the choice found using the tool agrees with that found during this detailed examination.

60 subjects (46 males, 14 females), mostly undergraduate students (47 undergraduates, 13 postgraduates and PhDs), of 10 different nationalities took part in the study. Most of them (44 out of 60) had searched for an apartment in the area before

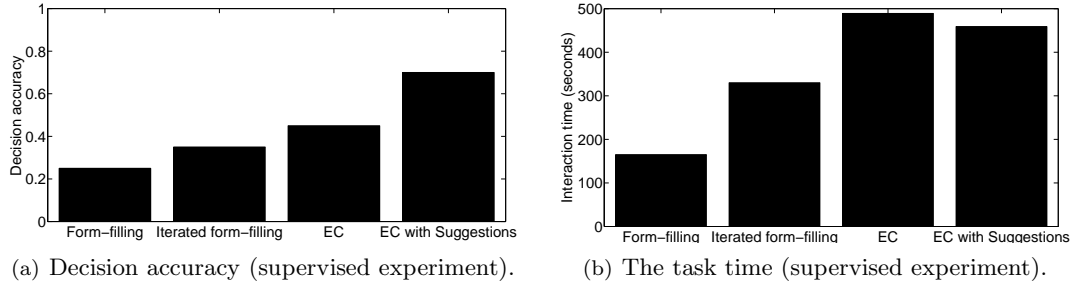


Fig. 10. Accuracy and task execution time for the different versions of the tool (supervised experiment). We compared the following interactions: form-filling (retrieval based on one shot question-answering), iterated form-filling (the same tool based on a question form but users were encouraged to modify the search until they were sufficiently satisfied) and our example-critiquing tool showing suggestions. The latter demonstrated a significantly higher decision accuracy.

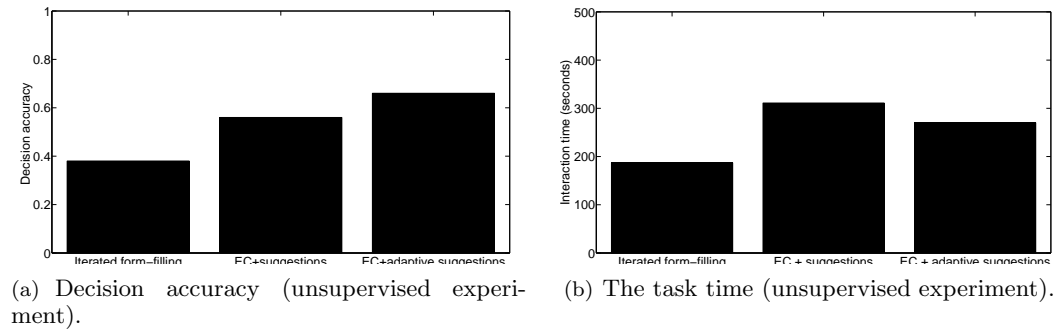


Fig. 11. Accuracy and task execution time for the different versions of the tool (unsupervised experiment). Example-critiquing with adaptive suggestions achieved the highest accuracy and it demands on average less user effort than example-critiquing with suggestions.

and had used online sites (42 out of 60) to look for accommodations.

The results of the experiment are shown in Fig 10a and 10b. They clearly show the variation in performance for the different versions of the tool. Using the traditional form-filling approach, only 25% of users found their most preferred solution. This fraction only increased to 35% when they could repeat the use of this interface as many times as they liked. On the other hand, example-critiquing reached a 45% accuracy. However, the improvement is not statistically significant: the student test gives values of $p=0.097$ for the difference between form-filling and simple example-critiquing, and $p=0.265$ for repeated form-filling and simple example-critiquing.

When using example-critiquing with suggestions we obtained 70%. This difference of accuracy is strongly statistically significant with $p=0.00176$ when example-critiquing with suggestions is compared to single form-filling and with $p=0.0133$

when compared to repeated form filling. This shows that suggestions are critical in improving decision accuracy for example-critiquing tools.

It is important to note that the higher performance of example-critiquing is obtained with users who are not bound to a particular dialogue, but are free to interact with the system on their own initiative.

The increased accuracy comes at the expense of a longer interaction time. Partly this is due to the fact that users were not familiar with the example-critiquing interface and thus took some time to get used to it. Interestingly, example-critiquing with suggestions required less interaction time than without suggestions, but achieved much higher accuracy. This is another indication of the importance of suggestions.

We attribute the poor performance of form-filling to the fact that people were driven to state many preferences before having considered any of the available options. Subjects that used the form-

FlatFinder - Web Search

This site allows you to perform a query over possible student accommodations in the region of **Lausanne**.

text size
Medium
Change

MySearch

Search for Accommodation

1 What type of accommodation do you need ?(What does it mean?)
 Apartment Room Studio Shared apartment
 Don't Care

2 How do you care about price ? (What does it mean?)
 Prefer accommodations cheaper than Swiss Francs.
 (blank = not consider price)

3 How many rooms do you need? (What does it mean?)
 Prefer accommodations with more than rooms.

4 Do you want a Furnished apartment? (What does it mean?)
 Furnished Not furnished Don't care

5 Rooms type (What does it mean?)
 Preferences for kitchen and bathroom.
 Kitchen: Bathroom:

7 Do you smoke? (What does it mean?)
 Yes No Don't care

8 Which mean of transportation do you prefer? (What does it mean?)
 Metro Bus Train Don't care

9 How far should it be from the center ?(What does it mean?)
 Maximum minutes by car

10 How far should it be from the university ?(What does it mean?)
 Maximum minutes by car

Search Reset

Powered by LIA - Last modified on Tuesday, November 1, 2005

Fig. 12. The form-filling interface used in our user study comparing example-critiquing (with and without suggestions) with the form-filling interface.

filling approach started with an average of 7.5 preferences, and only 5 of 20 ever removed any of them. In contrast, subjects that used example-critiquing with suggestions started with an average of only 2.7 preferences, but added an average of 2.6 to reach 5.3 preferences at the end of the interaction. Here, half of the preferences were constructed while considering examples, and the results suggest that they were much more accurate.

It appears that when users are asked to state their preferences without considering the options, many of these preferences are simply inaccurate. Since users are reluctant to retract them, this leads to inaccurate search results. On the other hand, significant gain in accuracy can be achieved by the use of active suggestion strategies to stimulate preference expression and increase its fluency.

We also found a correlation between the number of preference revisions and accuracy: people

who found their target item made 6.93 preference revisions on average, whereas those who did not find their item made an average of only 4.51 revisions. The difference is statistically significant with $p=0.0439$.

6.5. Evaluation of adaptive suggestions with unsupervised user studies

In order to assess the effectiveness of the adaptive strategy for suggestions, we carried out a user study with an unsupervised setting. Users (students in computer science department) were asked to participate using the mailing list of the department. The students were rewarded with the participation in a lottery that assign a prize to a user among all those who performed the study.

In the experiment users are assigned to different interfaces. For each tool, the users need to make

a choice. They are finally asked to go through the list of the options to select their target item.

The results are shown in Figure 11 and confirm our previous findings that questions-answering interface can lead to wrong decisions: only a minority of users can find their target with a form-filling interface based on questions-answering (form-filling has an accuracy of 38% that is lower than example-critiquing with suggestions, statistically significant with $p < 0.05$).

Adaptive suggestions gives an accuracy of 66% while example critiquing with standard suggestions gives only 56%. It is interesting to note that the average session for example critiquing when adaptive suggestions are present is shorter: only around 4.5 minutes opposed to more than 5 minutes for the sessions in which standard.

We stress the fact that unsupervised experiments are more difficult because we cannot monitor the interactions as close as in a supervised setting; users might not understand some technical aspects of the interface and therefore fail to use the tool correctly. It is not surprising that example-critiquing with standard suggestions gives a value that is lower than the value obtained in the supervised experiment.

At the end of the experiment, we asked what the user thought of the suggestions (on a 1 to 5 scale). The average value for adaptive suggestions (3.55) was higher than the value for standard model-based suggestions (3.05). Users were also asked questions about possible improvements of the example-critiquing tool. Overall, the majority of them demanded to see more examples at each cycle (53%), and a strong majority (65%) would like to directly select the suggestions without constructing their complete preference model. A sizeable number of users would be interested in the use of colors to highlight preferences that cannot be matched (45%).

While only a fraction of the users stated that they would certainly use the example-critiquing interface for ecommerce activities (35%), a vast majority (68%) stated that they would find example-critiquing useful for decision problems in which many users are involved, as for example in collective vacation search.

7. Conclusions

The internet allows access to an unprecedented variety of information, but forces people to see the

world through the restricted lens of a computer. Currently, only a small minority of users manage to reliably find the items they are looking for, and better tools for preference-based search are badly needed.

We believe that the mixed-initiative approach for preference-based search presented in this paper is a major step in this direction. Our conversational recommender system keeps a model of the user's preferences and show a set of options generated on the basis of this model. The user reacts by either picking one as her choice or modifying her preference model. To increase the quality of the final decisions, the system provides suggestions (special recommendations) to stimulate preference expression.

We reviewed our research contribution in the domain of critiquing recommender systems and summarized some of the previous results. In particular, the distinguishing trait of our approach lies in the display of suggestions that are recommendations aimed at stimulating the expression of preferences.

Then we described how we can further increase the recommendation accuracy by refining the uncertainty over users' preferences and making suggestions that are adaptive to the users reactions.

Our approach can be applied to any domain where the options are modeled as multi-attribute items; however large databases or configuration problems would require some tradeoff on the preference expressiveness (the type of statements allowed by critiques) in order to keep the generation of recommendations fast enough.

We evaluated the effectiveness of adaptive suggestion strategies with both simulation and unsupervised user studies.

In the simulations, we used previously collected logs of interactions and showed that our approach works significantly better than standard model-based suggestions. While a considerable part of the improvement is due to prior knowledge, the adaptation of suggestions according to the reactions gives an important improvement, especially in cycles in which the user does not state additional preferences. In the simulations, we manage to discover the complete set of preferences almost all the time.

In the user studies, we showed that example-critiquing with suggestions with fixed probabilities already increase decision accuracy from 25% (form-filling) to 65% in a supervised setting.

In form-filling, because the interaction is based on question-answering, users fell prey to means-objectives, and state preferences that they do not really have.

We found a correlation between the number of preference revisions and accuracy, meaning that more the user is engaged in the interaction the more likely she will find her target item.

We evaluated adaptive suggestions in an unsupervised setting where accuracy tends to be lower because users might not understand some technical aspect of the interface. Example-critiquing with adaptive suggestions achieved 66% of accuracy, while example-critiquing with standard model-based suggestions achieved 56%. The users of example-critiquing with adaptive suggestions interact with the system for less time, therefore less effort is required. The users also find adaptive suggestions more useful on average.

The unsupervised experiment shows that the techniques can be used in real e-commerce applications and it is expected to perform well.

References

- [1] Martin Borghetto, Ralph Kent, Matthew Owen, and Deborah Cornell. Transportation e-commerce and the task of fulfilment. *Morgan Stanley and Dean Witter. Equity Research Europe*, March 2000.
- [2] Craig Boutilier. A pomdp formulation of preference elicitation problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI'02)*, pages 239–246, 2002.
- [3] Robin Burke. Interactive critiquing for catalog navigation in e-commerce. *Artificial Intelligence Review*, 18(3-4):245–267, 2002.
- [4] Robin D. Burke, Kristian J. Hammond, and Benjamin C. Young. The FindMe approach to assisted browsing. *IEEE Expert*, 12(4):32–40, 1997.
- [5] Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI'00)*, pages 363–369. AAAI Press / The MIT Press, 2000.
- [6] Gretchen B. Chapman and E. Johnson. Incorporating the irrelevant: Anchors in judgments of belief and value. In T. Gilovich and Kahneman [31], pages 120–138.
- [7] Li Chen and Pearl Pu. Evaluating critiquing-based recommender agents. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, pages 157–162, Boston, MA, USA, July 2006. AAAI press.
- [8] Amos Tversky Daniel Kahneman, Paul Slovic. Judgment under uncertainty: Heuristics and biases. *Science*, 185:1124–1131, 1974.
- [9] Nicholas Epley and Thomas Gilovich. Putting adjustment back in the anchoring and adjustment heuristic: Differential processing of self-generated and experimenter-provided anchors. *Psychological Science*, 12(5):391–396, 2001.
- [10] Ronald Fagin. Fuzzy queries in multimedia database systems. In *PODS '98: Principles of database systems*, pages 1–10, New York, NY, USA, 1998. ACM Press.
- [11] Ralph L. Keeney. *Value-Focused Thinking. A Path to Creative Decision Making*. Cambridge: Harvard University Press, 1992.
- [12] R.L. Keeney, J.S Hammond, and H. Raiffa. *Smart choices: A guide to making better decisions*. Harvard University Press, Boston, 1999.
- [13] Werner Kiesling. Foundations of preferences in database systems. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02)*, pages 311–322, 2002.
- [14] Greg Linden, Steve Hanks, and Neal Lesh. Interactive assessment of user preference models: The automated travel assistant. In *Proceedings of the Fifth International Conference on User Modeling (UM'97)*, pages 67–78, 1997.
- [15] Kevin McCarthy, Lorraine McGinty, Barry Smyth, and James Reilly. A live-user evaluation of incremental dynamic critiquing. In *Proceedings of the 6th International Conference on Case-Based Reasoning (IC-CBR'05)*, volume 3620, pages 339–352. Springer LNAI, 2005.
- [16] Kevin McCarthy, Lorraine McGinty, Barry Smyth, and James Reilly. On the evaluation of dynamic critiquing: A large-scale user study. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI'05)*, pages 535–540, 2005.
- [17] David McSherry. Diversity-conscious retrieval. In *Proceedings of 6th European Conference on Advances in Case-Based Reasoning (ECCBR'02)*, pages 219–233, 2002.
- [18] J.W. Payne, J.R. Bettman, and E.J. Johnson. *The Adaptive Decision Maker*. Cambridge University Press, 1993.
- [19] P. Pu, B. Faltings, and M. Torrens. Effective interaction principles for online product search environments. In *Proceedings of the 3rd ACM/IEEE International Conference on Web Intelligence*, pages 724–727. IEEE Press, September 2004.
- [20] Pearl Pu and Li Chen. Integrating tradeoff support in product search tools for e-commerce sites. In *Proceedings of ACM Conference on Electronic Commerce (EC'05)*, pages 269–278, 2005.
- [21] Pearl Pu and Boi Faltings. Enriching buyers' experiences: the smartclient approach. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI'00)*, pages 289–296. ACM Press New York, NY, USA, 2000.

- [22] Pearl Pu and Pratyush Kumar. Evaluating example-based search tools. In *Proceedings of the ACM Conference on Electronic Commerce (EC'04)*, 2004.
- [23] Pearl Pu, Paolo Viappiani, and Boi Faltings. Increasing user decision accuracy using suggestions. In *ACM Conference on Human factors in computing systems (CHI06)*, pages 121–130, Montreal, Canada, April 2006.
- [24] James Reilly, Kevin McCarthy, Lorraine McGinty, and Barry Smyth. Dynamic critiquing. In *Proceedings of the 7th European Conference on Advances in Case-Based Reasoning (ECCBR'04)*, pages 763–777, 2004.
- [25] Francesco Ricci and Quang Nhat Nguyen. Acquiring and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent Systems*, 22(3):22–29, 2007.
- [26] Hideo Shimazu. Expertclerk: Navigating shoppers buying process with the combination of asking and proposing. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI'01)*, volume 2, pages 1443–1448, 2001.
- [27] Paul Slovic. The construction of preference. *American Psychologist*, 50(5):364–371, 1995.
- [28] Barry Smyth and Paul McClave. Similarity vs. diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR'01)*, pages 347–361, 2001.
- [29] Barry Smyth and Lorraine McGinty. The power of suggestion. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI 2003)*, Acapulco, Mexico, pages 127–132, 2003.
- [30] Markus Stolze and Michael Ströbel. Utility-based decision tree optimization: A framework for adaptive interviewing. In Mathias Bauer, Piotr J. Gmytrasiewicz, and Julita Vassileva, editors, *User Modeling*, volume 2109 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2001.
- [31] D.Griffen T. Gilovich and D. Kahneman, editors. *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge University Press, 2002.
- [32] A. Tversky, S. Sattath, and P. Slovic. Contingent weighting in judgment and choice. *Psychological Review*, 95:371–384, 1988.
- [33] Amos Tversky. Contrasting rational and psychological principles in choice. *Wise Choices: Decisions, Games and Negotiations*, pages 719–736, 1996.
- [34] Paolo Viappiani, Boi Faltings, and Pearl Pu. The lookahead principle for preference elicitation: Experimental results. In *Seventh International Conference on Flexible Query Answering Systems (FQAS)*, 2006.
- [35] Paolo Viappiani, Boi Faltings, and Pearl Pu. Preference-based search using example-critiquing with suggestions. *Journal of Artificial Intelligence Research (JAIR)*, 27:465–503, 2006.
- [36] Markus Zanker, Sergiu Gordea, Markus Jessenitschnig, and Michael Schnabl. A hybrid similarity concept for browsing semi-structured product items. In Kurt Bauknecht, Birgit Pröll, and Hannes Werthner, editors, *EC-Web*, volume 4082 of *Lecture Notes in Computer Science*, pages 21–30. Springer, 2006.
- [37] Jiyong Zhang and Pearl Pu. Performance evaluation of consumer decision support systems. *International Journal of E-Business Research*, 2:28–45, 2006.