Stimulating Preference Expression using Suggestions

Paolo Viappiani[1], Boi Faltings[1], Vincent Schickel Zuber[1] and Pearl Pu[2]
1) Artificial Intelligence Laboratory (LIA)
2) Human Computer Interaction Group(HCI)
Ecole Polytechnique Fédérale de Lausanne (EPFL)
1015 Lausanne, Switzerland
paolo.viappiani@epfl.ch|boi.faltings@epfl.ch|vincent.schickel-zuber@epfl.ch|pearl.pu@epfl.ch

Abstract

Users often have to search for a most preferred item but do not know how to state their preferences in the language allowed by the system. Example-Critiquing has been proposed as a mixed-initiative technique for allowing them to construct their preference model in an effective way.
In this technique, users volunteer their preferences as critiques on examples. It is thus important to stimulate their preference expression by the proper choice of examples, called suggestions. We analyze what suggestions should be and derive several new techniques for computing them. We prove their effectiveness using simulations and live user studies.

Introduction

To find products in online environments, people increasingly rely on computerized search tools. The performance of such tools depends crucially on an accurate model of their users' preferences. Obtaining such models requires an adequate interaction model and system guidance.

Utility theory provides a solid mathematical foundation for optimal decision support. However, it assumes complex preference models that cannot be obtained in e-commerce scenarios: people are not willing to go through lengthy preference elicitation processes. Furthermore, they are usually not very familiar with the available products and their characteristics. Thus, their preferences are not well established, but *constructed* while learning about the available products (Payne, Bettman, & Johnson 1993). To allow such construction to take place, we need to let users explore the space of possible options while building their preference model.

A good way to do this is through a mixed-initiative system based on *example critiquing* (see Figure 1). It shows examples of complete solutions and invites users to state their critique of this solution. This allows users to better understand their preferences. Example critiquing has been proposed by a number of authors. (Linden, Hanks, & Lesh 1997; Burke, Hammond, & Young 1997; Shimazu 2001; Pu & Faltings 2000)

It has been shown (Pu, Faltings, & Torrens 2003; Pu & Faltings 2004) that example critiquing enables users to perform rational decision tasks more efficiently with considerably fewer errors than a non-critiquing interface.

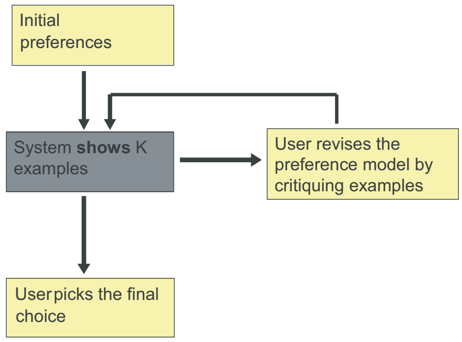

Figure 1: *Example critiquing interaction. The dark box is the computer's action, the other boxes show actions of the user.*

In an example-critiquing interaction, user's preferences are *volunteered*, not elicited: users are never forced to answer questions about preferences they might not be sure about. Thus, users will only state preferences that they actually have, so that a preference model with a higher number of preferences will also lead to more accurate decisions.

To encourage users to provide as complete a preference model as possible, the system should show them examples that will stimulate expression of their preferences as much as possible. Several authors have proposed that examples should include not only outcomes that are good with respect to the current preference model, but also outcomes that are designed to educate the user about possible other options and thus stimulate preference expression. Thus, the examples would include:

- **candidate** examples that are optimal for the current preference query, and
- **suggested** examples that are chosen to stimulate the expression of preferences.

While most earlier work, such as (Burke, Hammond, & Young 1997; Shearin & Lieberman 2001; Faltings, Torrens, & Pu 2004) has concentrated on rankings and filters for finding candidate examples, the main topic of this paper are methods for generating suggested examples.

Different strategies for suggestions have been proposed in the literature. Linden (Linden, Hanks, & Lesh 1997) used extreme examples, where some attributes take extreme values. Others use diverse examples as suggestions (Smyth & McGinty 2003; Shimazu 2001).

In this paper, we take a deeper look at how suggestions should be generated and derive a a family of new strategies, called Pareto-strategies. We show through simulations and experiments with live users that they strongly outperform suggestion strategies that have been proposed earlier.

A deeper look at suggestions

The problem faced by a user in a search tool is that he has to learn how to state his preferences so that the tool can find his most preferred option. We can assume that he is minimizing his own effort and will add preferences to the model only when he can expect them to have an impact on the solutions. This is the case when:

- he can see several options that differ in a possible preference, and
- these options are relevant, i.e. they could be reasonable choices.

In all other cases, stating an additional preference is irrelevant: when all options would evaluate the same way, or when the preference only has an effect on options that would not be eligible anyway, stating it would only be wasted effort. This leads us to the following principle as a basis for a new set of suggestion strategies, called Pareto-strategies:

> Suggestions should be options that could become optimal when an additional preference is stated.

As a simple example consider searching for a suitable flight between two cities A and B. Options are characterized by the attributes: `<price, arrival time, departure airport>`. For the departure airpot, there is a city airport (`CTY`) which is very close to where the user lives and a big international airport (`INT`) which takes several hours to reach. Assume that the user has three preferences in this order of importance:

- the lowest price
- arrive by 12:00
- depart from the city airport

and that he initially only states a preference on the price. The other two preferences remain hidden. Finally, assume that the choice is among the following options:

- f_1: `<200, 13, INT>`
- f_2: `<250, 14, INT>`
- f_3: `<300, 9, INT>`
- f_4: `<600, 8:30, INT>`
- f_5: `<400, 12, CTY>`
- f_6: `<400, 16:30, CTY>`
- f_7: `<900, 18, CTY>`
- f_8: `<280, 15, INT>`

According to the first stated preference (lowest price), the options are ordered $f_1 \succ f_2 \succ f_8 \succ f_3 \succ f_5 = f_6 \succ f_4 \succ f_7$.

Assume that the system shows the 2 most promising ones: f_1 and f_2, the two with lowest price. Here f_1 already dominates f_2 (f_1 is better in all respects) according to the users hidden preferences, so he is unlikely to state any additional preference based on these examples.

A strategy that generates suggestions according to diversity might pick f_7 as suggestion as it is most different from what is currently displayed. However, the user is likely to discard this option, because it is very expensive and arrives very late.

A strategy that chooses examples with extreme values would show one of f_4 or f_7. Neither of them is likely to be taken seriously by the user: f_4 is likely to leave at a very early and inconvenient hour, while f_7 arrives much too late to be useful.

What makes f_7 a bad suggestion to show? From the system point of view where only the preference about the price is known f_7 is not a great suggestion because for most of the possible hidden preferences, it is likely to be dominated by f_5 or f_6. If the hidden preference is for the city airport, then f_5 dominates because it is cheaper. If the hidden preference is on arrival time, then only if the user requires an arrival later than 16:30 there is a chance that it will not be dominated by f_6, which is otherwise significantly cheaper.

Without knowing the hidden preferences, good suggestions for this scenario would be f_3, which has a reasonable arrival time without a significantly higher price, f_5 or f_6. These examples differ from f_4 and f_7 in that they have a good chance of becoming optimal for a wide range of possible hidden preferences.

We are now going to formalize criteria for choosing such suggestions automatically based on the current preference model, and prove its usefulness through a set of experiments.

Implementing Pareto suggestion strategies

To further show how to implement Pareto suggestion strategies, we have to define preference models and some minimal assumptions about the shape that user preferences might take. We stress that these assumptions are only made for generating suggestions. The preference model used in the search tool could be more diverse or more specific as required by the application.

Preference model

Given a fixed set of n *attributes* $A = \{A_1, .., A_n\}$, an option o is characterized by the values $a_1(o), ..., a_n(o)$ that have to belong to the fixed domains $D_1, .., D_n$, that can be explicitly *enumerated* or can be *intervals* of continuous or discrete elements.

The user's *preferences* are supposed to be independent and defined on individual attributes:

Definition 1 *A* preference r *is an order relation* $\preceq_r$ *of the values of an attribute* a*;* $\sim_r$ *expresses that two values are equally preferred. A* preference model *R is a set of preferences* $\{r_1, .., r_m\}$.

If there can be preferences over a combination of attributes, such as the total travel time in a journey, we assume that the model includes additional attributes that model these combinations. As a preference r always applies to the same attribute a_z, we simplify the notation and apply $\preceq_r$ and $\sim_r$ to the options directly: $o_1 \prec_r o_2$ iff $a_z(o_1) \prec_r a_z(o_2)$. We use $\prec_r$ to indicate that $\preceq_r$ holds but not $\sim_r$.

Suggestions

We consider 4 strategies of increasing complexity for selecting suggestions. All are based on the principle of selecting options that have the highest chance of becoming optimal, as explained earlier.

Depending on the formalism used for modeling preferences, there are different ways of combining the order relations given by the individual preferences r_i in the user's preference model R into a global order of the options. For example, each preference may be expressed by a number and the combination may be formed by summing the numbers corresponding to each preference, or by taking their minimum or maximum.

We can obtain suggestion strategies that are valid with most known preference modeling formalisms by using qualitative optimality criteria based on *dominance* and *Pareto-optimality*:

Definition 2 *An option o is* dominated by *an option o' with respect to R if and only if for all $r_i \in R$, $o \preceq_{r_i} o'$ and at least one $r_j \in R$, $o \prec_{r_j} o'$. We write $o \prec_R o'$ (equivalently we can say that o' dominates o and write $o' \succ_R o$)*

We also say that o is dominated *(without specifying o')*

Note that we use the same symbol $\prec$ for both individual preferences and sets of preferences.

Definition 3 *An option o is* Pareto-optimal *(PO) if and only if it is not dominated by any other option.*

Pareto-optimality is the strongest concept that would be applicable regardless of the preference modeling formalism used. Our techniques use the concept of *dominating set*:

Definition 4 *The dominating set of an option o is the set of all options that dominate o: $O_R^+(o) = \{o' \in O : o' \succ_R o\}$.*

We will write $O^+(o)$ if it is clear from the context which is the set R of preferences we are considering.

In our applications, users initially state only a subset R of their true preference model $\overline{R}$. When a preference is added, dominated options with respect to R can become Pareto-optimal. The following observation is the basis for evaluating the likelihood that a dominated option will become Pareto-optimal:

Proposition 1 *A dominated option o' with respect to R becomes Pareto-optimal with respect to $R \cup r_i$ (a new preference r_i is added), if and only if o' is strictly better with respect to r_i than all options that currently dominate it: $o' \succ_{r_i} o, \forall\, o \in O_R^+(o')$.*

In general, the Pareto-optimal set increases when stating more preferences, as the dominance relation becomes sparser.

Strategies

The *extreme* strategy, proposed initially by Linden et al. in ATA (Linden, Hanks, & Lesh 1997), selects options that have either the smallest or the largest value for an attribute on which the user did not state any preference yet. Because many possible choices often do not have a clear ranking, random selection may be required. This strategy is included for comparison purposes.

The 3 strategies we propose (that we call *Pareto* suggestion strategies) use Pareto-optimality to implement the principle stated in the introduction: suggestions should not be optimal yet but have a high likelihood of becoming optimal when an additional preference is added. An ideal suggestion is an option that is Pareto-optimal with respect to the full preference model $\overline{R}$ but is dominated in R, the partial preference model.

According to Proposition 1 the probability of a dominated option o of becoming Pareto-optimal is equal to:

$$p(o) = \prod_{o_+ \in O^+(o)} p_d(o, o_+) \tag{1}$$

where p_d is the probability that a new preference makes o escape the domination relation with a dominating option o_+, i.e. if o is preferred over o_+ according to the new preference. As evaluating this probability exactly requires the probability distribution of the possible preferences, in general difficult to evaluate, we propose several strategies based on increasingly detailed assumptions about these distributions.

The simplest strategy, the *counting strategy*, is based on the assumption that p_d is constant for all dominance relations. Thus, we assume:

$$p(o) = \prod_{o_+ \in O^+(o)} p_d = p_d^{|O^+(o)|} \tag{2}$$

Since $p_d \leq 1$, this probability is the largest for the smallest set $O^+(o)$. Consequently, the best suggestions are those with the lowest value of the following counting metric:

$$F_C(o) = |O^+(o)| \tag{3}$$

The *attribute strategy* considers the fact that for breaking the dominance relation with all options in the dominating set, there has to be one attribute where all dominating options have different values. To express this concept, we define the predicate *Diff*:

Definition 5 *For an attribute a_i and a given option o_1 with dominating set O^+, $Diff(o_1, a_i, O^+)$ holds if:*

- *interval domains: $a_i(o_1)$ should be either greater than or smaller than the attribute values for a_i of all options in O^+*
- *enumerated domains: $a_i(o_1)$ should be different than the attribute values for a_i for all options in O^+*

The operator $diff(o_1, a_i, O^+)$ has value 1 if the predicate Diff holds, 0 otherwise.

The reasoning here is the following. For interval domains, we assume that preferences are continuous, i.e. the user is likely to prefer values to be larger or smaller than a certain threshold, or as large or as small as possible. This applies to attributes like price or travel time and fits well with the majority of users. For enumerated domains, a new preference may break the dominance relation whenever the attribute has a different value. Then we count the number of attributes for which there are no preferences yet and where all dominating options have a different value:

$$F_A(o) = \sum_{a_i \in A_u} P_{a_i} \text{diff}(a_i, o, O^+(o)) \qquad (4)$$

where A_u is the set of attributes on which no preference has been expressed yet; P_{a_i} is the probability that the user has an unstated preference on attribute a_i. It chooses as suggestions those options with the largest value of this metric.

The *probabilistic strategy* finds the best possible estimation of the probability that a particular solution will become Pareto-optimal. p_d (Equation 1) can be written as:

$$p_d(o, o_+) \approx \sum_{a_i \in A_u} P_{a_i} \delta_i(o, o_+) \qquad (5)$$

where $o_+ \in O^+(o)$, the set of dominators of o, and δ_i is an heuristic estimation of the probability that an hidden preference on attribute a_i make o better than o^+ according to that preference, hence escaping the dominance relation. As heuristic we use a normalized difference for interval domains; for discrete we simply look if the attribute values are the different.

Such a heuristic (the more two attribute values are different, the more a preference is likely to discriminate them) implement the intuition that suggestion should be different from the candidates. It is a correct estimation of the probability (when the attributes are not correlated) under some circumstances: preference are of the form $LessThan(v)$, $GreaterThan(v)$ and values on the same side with respect to v are indifferent between themselves.

The example continued In the example before, f_1 and f_2 are shown as candidate optimal examples. We will now consider which options will be chosen by the filters as suggestions, omitting the calculations.

The dominance relation directly map to the counting filter rank: the first suggestion will be f_8 (that is not very interesting because very similar to the candidates), if two are shown, the other will be f_3. The attribute filter selects f_6 as best suggestion: its dominators for price (f_1, f_2, f_8, f_3) all depart from a different airport and leave before (external interval): the *diff* is equal to 1 on both attributes. The attribute filter cannot choose a second suggestion: all other options have the same values for *diff* on both attributes. The probabilistic filter chooses f_6 and f_5: they are both dominated by four options (f_1,f_2,f_8 and f_3) but have high chance of breaking this domination because they significantly differ on the other attributes (they leave from the other airport; f_6 lands few hours after, f_5 before).

Let's assume now that the user has stated the preference about the price and about the time. The candidates will be now f_1 and f_3. The suggestions: the counting strategy will propose f_2 and f_5 (dominated respectively only by f_1 and f_3), the attribute f_5 (different airport than its dominator, f_3) and the probabilistic f_5 and f_6. All suggestions technique show an example with the city airport and the user is stimulated to state that preference.

	interface C	interface C+S
number of critiquing cycles	3.48	3.90
initial preferences	2.19	2.10
final preferences	2.81	4.19
increment	0.62	2.09

Table 1: *Results of the between groups experiment.*

Empirical Evaluation

We conducted two experiments to evaluate the suggestion strategies on FlatFinder, a web application for finding student housing that uses real offers from a university database. User preferences are elicited interactively and incrementally. There are 10 attributes; four are interval domains (e.g. number of rooms), and six are enumerated domains (e.g. type of available transportation). The users state their initial preferences and perform a query pressing the *search* button; based on the results displayed, the users then refine their preference model through the example critiquing process.

We define the *interaction cycle* as the period between two consecutive times that the search button is pressed. During each interaction cycle, the user can state additional preferences, change the reference value of existing preferences or even remove one or more of the preferences.

The application was made available in two versions: **C**, showing only the candidate set, and **C+S**, also showing the suggestions produced by the probabilistic strategy. The subjects included doctoral and undergraduate students, as well as university staff. The experiment was supervised. We divided the 54 subjects in two groups, A and B, of 27 subjects each.

We conducted two experiments. In the first one, we compared the two different interfaces showing one version to each group. In the second experiment, we further showed interface C+S to each subject in the group that was shown interface C before and compared their preferences and choices. Note that we could not ask the group that had worked with the full interface to use the simpler version in the second experiment, as they would have already constructed a more complete preference model based on the suggestions.

Between groups experiment The first group (group A), was shown interface C while the second group (group B) was shown interface C+S. We measured the number of critiquing cycles and the number of expressed preferences, defined as the number of preferences at the end of the interaction minus the number of initial preferences. Table 1 shows the results we obtained.

Note first that both groups state about the same number of initial preferences r, showing that there is no bias in their

	interface C	interface C+S
initial preferences	2.19	2.19*
final preferences	2.81	3.62
increment	0.62	1.43
% that changed their choice		77%

Table 2: *Result of the within group experiment. *= the initial preferences are considered the same as in the first use of the application.*

distribution. Second, the number of critiquing cycles for both groups is not significantly different (t-test: tStat=-0.72, p=0.23), showing that they both spent a comparable amount of effort. However, there is a big difference in the number of elicited preferences; subjects who saw the suggestions in the C+S interface on average stated 2.1 additional preferences during critiquing, whereas those who did not see the suggestions added only 0.62. This difference is statistically very significant (t-test: tStat=-4.07, p=0.01). As all preferences are volunteered by users, it is likely that they reflect true preferences. Thus, a model with more preferences is more accurate. The suggestions almost double the number of preferences, and thus very significantly increase the accuracy of the preference model.

Within group experiment In the second experiment, which was conducted simultaneously with the first, we additionally asked the subjects who had used only interface C to subsequently complete the same task with interface C+S. 15 of the 27 subjects in the group agreed to continue for this experiment. We measured how many additional preferences they stated with interface C+S, and how often the option they finally chose was different from their initial choice.

Table 2 shows the results of this experiment. They show that subjects who used interface C+S after interface C stated on average 0.93 additional preferences (t-test: tStat=-2.3, p=0.02). The average total number of 3.62 preferences is slightly lower than the 4.19 in the first experiment, which can be attributed to the fact that subjects are likely to be tired when performing the second of two experiments. Thus, the suggestions again seem to significantly improve the quality of the preference model.

This is also confirmed considering that 77% of the subjects discovered an option that they considered better than their previous choice based on the generated suggestions. This is another strong indication that the resulting preference model is indeed significantly more accurate.

Experiments with a simulated user

To further compare the different strategies without using costly user studies, we simulated the interaction of a computer generated user behavior with an example critiquing system with randomly generated preferences. The simulated interaction starts with the initial preference (randomly chosen) and k solutions are selected as suggestions according to the strategy in use. The simulated user states a new preference in the case he/she is shown a solution that would become Pareto-optimal if such a preference were stated. If

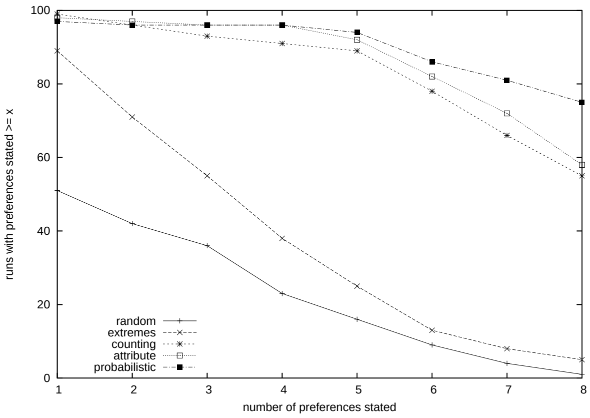

Figure 2: For each strategy, we compare the fraction of simulation runs that discover at least x preferences. 100 runs, data set with 8 attributes and preferences.

among these k options, there is one that triggers this rule, a new preference will be stated. The interaction continues until either the user model is complete or the simulated user states no further preference. Note that when the complete preference model is discovered the user finds its most wanted option.

The results of the simulation are summarized in Figure 2 for a catalog with 50 options, 9 attributes and complete user model. It is shown the percentage of runs (out of 100) that discover at least x out of the 8 preferences to discover. We see that the suggestion strategies provide a marked increase in the quality of the preference model. Furthermore, the increase closely mirrors the results of the study with human users, providing additional confirmation that the reasoning behind our strategies is correct.

We investigated the impact of the number of preferences, the number of attributes and the size of the data set.

Surprisingly we discovered that the number of attributes only slightly changes the results. Keeping the number of preferences constant at 6 (one being the initial preference), we made simulations with the number of attributes equal to 6,9,and 12. The fraction of runs (with 100 total runs) that discovered all the preferences varied for each strategy and simulation scenario by no more than 5%.

We were surprised by the fact that the strategy of generating extreme examples, as originally proposed by Linden (Linden, Hanks, & Lesh 1997), performed so poorly and only beats randomly selected suggestions by a narrow margin. This shows the importance of considering the already known preferences in the selection of suggestions.

Increasing the size of the data-set makes the performance gap bigger. This can be explained by the fact that the Pareto suggestion strategies can profit from the increased number of options from which to pick out the best options.

The simulations show that the simulated user is much more likely to state new preferences using the probabilistic strategy (statistically significant). Moreover, in the simulations the complete preference model was discovered up to 25 times more often with the probabilistic strategy than with

#P	random	extr.	c.	att.	prob.
2	0.40	0.55	0.75	0.79	0.73
5	0.29	0.44	0.78	0.81	0.82
8	0.28	0.39	0.80	0.83	0.86

Table 3: The impact of the variation in the number of preferences to be discovered. (% of correctly preferences discovered, on average. Legend: extr. = extreme strategy, c. = counting strategy, att.=attribute strategy, prob= probabilistic strategy.)

data size	random choice	extr.	c.	att.	prob.
50	0.28	0.39	0.80	0.83	0.86
75	0.23	0.29	0.84	0.81	0.91
100	0.20	0.24	0.81	0.83	0.89
200	0.09	0.16	0.80	0.84	0.92

Table 4: The impact of the size on the performance. For each strategy, we show the average fraction of preferences discovered (in %). We ran 100 Simulations with 8 attribute and 8 preferences. Legend: extr. = extreme strategy, c. = counting strategy, att.=attribute strategy, prob= probabilistic strategy.

randomly picked suggestions, up to 10 times more than using the extreme strategy, and 1.5 times more than the counting strategy. The probabilistic strategy has an average behavior better than the attribute strategy.

Among the three Pareto strategies, the probabilistic strategy gives the best results. However, it also makes the most assumptions about the preferences the user is likely to state. When these assumptions are not satisfied, the performance is likely to degrade. On the other hand, the counting strategy is the most robust among Pareto strategies as it makes no assumptions whatsoever about the form of the user's preferences, while still achieving a large gain over simpler strategies. In practice, it may often be a better choice.

Conclusions

Obtaining quality user preferences is essential for increasing the accuracy of search and recommender tools. Mixed-initiative systems such as example critiquing are a promising technology for efficiently eliciting accurate user preference models. Determining how to stimulate the user to state preferences on as many attributes as she or he may have is a key issue concerning such systems. We have developed a model for computing what examples would be the best for stimulating preference expression and designed several suggestion strategies based on this model. The main principle is that suggestions should be options that are dominated under the current preference model but would no longer be dominated with the inclusion of additional preferences. To implement this principle with a minimum of assumptions about the user's preference model, we defined different strategies based on the concept of Pareto-optimality, generally called *Pareto strategies*.

With a user study, we showed that this model is indeed very effective for eliciting user preferences; we followed this user study with a more precise investigation based on simulations. The results show the strong performance of Pareto strategies in comparison with other techniques that have been proposed earlier.

Overall, the suggestion strategies significantly contribute to making the most accurate decision by almost doubling the number of preferences stated by average users. As they are based on the very general notion of Pareto-optimality, they can be applied to a broad range of preference modeling formalisms, including utility functions, soft constraints (Bistarelli, Montanari, & Rossi 1997), and CP-networks (Boutilier *et al.* 2004). This should greatly strengthen the performance of example critiquing systems in applications ranging from decision support to e-commerce.

References

Bistarelli, S.; Montanari, U.; and Rossi, F. 1997. Semiring-based constraint satisfaction and optimization. *Journal of ACM* 44(2):201–236.

Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research* 21:135–191.

Burke, R. D.; Hammond, K. J.; and Young, B. C. 1997. The FindMe approach to assisted browsing. *IEEE Expert* 12(4):32–40.

Faltings, B.; Torrens, M.; and Pu, P. 2004. Solution generation with qualitative models of preferences. In *Computational Intelligence*, 246–263(18). ACM.

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

Payne, J.; Bettman, J.; and Johnson, E. 1993. *The Adaptive Decision Maker*. Cambridge University Press.

Pu, P., and Faltings, B. 2000. Enriching buyers' experiences: the smartclient approach. In *SIGCHI conference on Human factors in computing systems*, 289–296. ACM Press New York, NY, USA.

Pu, P., and Faltings, B. 2004. Decision tradeoff using example-critiquing and constraint programming. *Constraints: An International Journal* 9(4).

Pu, P.; Faltings, B.; and Torrens, M. 2003. User-involved preference elicitation. In *Working Notes of the Workshop on Configuration. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003)*.

Shearin, S., and Lieberman, H. 2001. Intelligent profiling by example. In *Intelligent User Interfaces*, 145–151.

Shimazu, H. 2001. Expertclerk: Navigating shoppers buying process with the combination of asking and proposing. In *Proceedings of the 17 International Joint Conference on Artificial Intelligence (IJCAI'01)*, volume 2, 1443–1448.

Smyth, B., and McGinty, L. 2003. The power of suggestion. In *IJCAI*, 127–132.