Solution generation with qualitative models of preferences

Boi Faltings
Swiss Federal Institute of Technology in Lausanne (EPFL)
Artificial Intelligence Laboratory (LIA)
CH-1015, Lausanne, Switzerland

Marc Torrens
i:FAO's Future Lab. and LIA, EPFL
PSE-C, CH-1015, Lausanne, Switzerland

Pearl Pu
Swiss Federal Institute of Technology in Lausanne (EPFL)
Human Computer Interaction Group
CH-1015, Lausanne, Switzerland

Abstract

When delegating complex tasks to a computer tool, eliciting and representing users' decision preferences is a crucial task. It is usually too complex to get a complete and accurate model of their preferences, especially regarding the tradeoffs between different criteria.

We consider decision aid systems where users specify their preferences qualitatively as the combination of criteria they consider, but the relative weights are uniform for all users. To compensate for the imprecision of this qualitative model, we let the user choose among a *displayed set* of possibilities rather than a single optimal solution.

We then consider how large this set of possibilities has to be to ensure that the target solution can be found, and how the probability of finding the correct solution varies with the number of preferences stated. We present a probabilistic analysis, randomly generated configuration problems and a commercial application. The results show that the selection mechanism must be carefully designed if the user should be guaranteed to find the target solution.

1 Introduction

Many real-world applications require people to select a most preferred item from a set of choices. For example, in e-commerce an electronic catalog system might provide access to millions of products, and the user has to navigate the catalog to find the most preferred one, which we call the *target*. Decision theory provides algorithms which guarantee to find this target solution given the options and an accurate preference model.

In the classical decision theory approach ([1]), solutions are characterized by a set of criteria and individual users characterize their preferences by the relative weights they give to criteria or combinations of criteria. Such approaches are described for example in [2], methods for eliciting the weights are discussed in more detail in [3, 4]. An example of its application in a decision aid system is the PC selection tool of IBM described in [5], where users can adjust the weights of different criteria and interactively see how different PC models rank according to these weights.

However, there are many practical situations where the space of possible criteria is too large to expect each user to state a numerical preference model for all of them. This is the case for example in travel, where there is a huge number of possible criteria involving times, means of transportation, locations, that vary from one user to another. In such a case, it is more feasible to characterize a user's preference *qualitatively* as the specific combination of criteria that apply, without also eliciting their exact weights. While the weights can be set to reflect those of average users, their values will not be exact and so the result is only an approximate model that will in general not correctly identify the most preferred solution. Such qualitative decision theory has recently become the subject of increased interest in the research community ([6, 7]).

We consider compensating for this shortcoming by letting the user pick the most preferred solution from a larger *displayed set* D of k solutions. Such an approach has been taken in numerous practical systems, for example in [8, 9, 10, 11, 13]. The process will be sound, i.e. allow the user to find the target solution, only if the displayed set actually contains the solution. It turns out that this heavily depends on the model used for selecting displayed solutions as well as on the number of preferences that have been stated.

In this paper, we analyze three different methods for selecting displayed solutions:

- *dominance filters* that display k solutions that are not dominated by any other one.
- *sum filters* that keep the k solutions with the lowest sum of preference violations, and
- *fuzzy filters* that keep the k solutions with the smallest maximum preference violation.

Using both a theoretical, probabilistic model and empirical examples, we show that none of the filters can guarantee soundness in all cases, and that all have to be carefully tuned to the application at hand.

1.1 Assumptions and Definitions

We assume that solutions are characterized by a fixed (but possible very large) set of *attributes*, and that users formulate qualitative *preferences* in the form of penalty functions formulated on the attributes, defined as follows:

Definition 1. *Every configuration solution is characterized by a finite set of* n *attributes* $A = \{a_1, ..., a_n\}$*. Every attribute can take values from a fixed domain* $\{d_1, ..., d_n\}$*. Attributes can be either attributes of the components or derived from their composition.*

A preference $p_i(a_k)$ *is a mapping* $d_k \rightarrow \Re$ *from an attribute* a_k *to a number that gives the penalty of that attribute value to the user. We assume that the smallest values are the most preferred ones. A* conditional preference $p_i(a_k|a_j = v)$ *is a preference that only applies to solutions where attribute* a_j *has value* v*.*

Note that each unconditional preference constructs a total order *of the domain of the attribute* a_k*, while conditional preferences only construct partial orders.*

To simplify, we write $p_i(S)$ *for* $p_i(a_j(S))$.

Example 1: Solution attributes and penalties.
The following attributes and preferences can for example be stated:

- Attributes:
  $a_1 =$ `arrival_time` (f_2)
  $a_2 =$ `departure_time` (f_2) - `arrival_time` (f_1) : transit time
  $a_3 =$ `transfer_airport` (f_1, f_2)
  $a_4 =$ `arrival_time` (f_2) - `departure_time` (f_1) : travel time
- Penalties:
  $p_1(a_1) = max(0, a_1 - 18:00)$
  $p_2(a_2|a_3 = \texttt{LHR}) = max(0, 2\,hours - a_2)$
  $p_3(a_4) = a_4/5$

We assume that the decision support system allows users to choose preferences from a predefined, but possibly unboundedly large set.

We assume that the preferences of user u are expressed by a set $P^*(u) = \{p_1, .., p_k\}$ as well as a set of weights $\{w_1, .., w_k\}$ associated with these preferences such that the weighted sum $W(S, u) = \sum_{p_i \in P^*(u)} w_i \cdot p_i(S)$ reflects the preference order of solutions, i.e. if $W(S_1, u) < W(S_2, u)$ then the user prefers solution S_1 over solution S_2.

We call the best solutions for the user's true preference model $P^*(u)$ the user's *target solution* $S_t(u)$. It is given as the solution that minimizes $W(S, u) = \sum_{p_i \in P^*(u)} w_i \cdot p_i(S)$.

We assume furthermore that for any penalty function, the average weight over all users that have chosen it to characterize their preference is equal to 1.

This amounts to a normalization of the penalty functions to fit average users. We furthermore assume that for a particular user, each weight differs by no more than ϵ from 1. Users are thus distinguished qualitatively by the choice of preferences rather than quantitatively by the importance of each preference in a predefined set.

2 Decision support with qualitative preference models

As the qualitative preference model is imprecise, we cannot be sure that it will select the target solution. To compensate for this shortcoming, we propose an approach where we generate a *displayed set* D of k target solutions and let the user pick himself from this set.

For this approach to be practical and successful, the following conditions must be satisfied:

1. it must be possible to limit the computed set D to a size of at most k displayed solutions so that it can be displayed in a consistent manner.
2. solutions that are pareto-optimal within the set D must also be pareto-optimal with respect to the set of all feasible solutions. This is important to keep the user from unknowingly picking a dominated solution as the final choice.
3. when the user has specified his preferences, the target (user's most preferred) solution must be included in the displayed solutions.

We consider three candidate techniques for computing the displayed solutions:

- a *dominance* filter that retains k dominant or pareto-optimal solutions.
- a *weight* filter that retains the k solutions such that the sum of their penalties is lowest.
- a *fuzzy* filter that retains k solutions such that their maximum penalty is lowest.

We now examine the three candidate techniques in detail. In order to allow a theoretical comparison, we assume that preferences p_i are independent, with real-numbered values in the interval [0..1], and that m solutions are distributed uniformly in the $|P^*|$-dimensional space of preference combinations. While in reality both assumptions are not likely to hold perfectly, comparing the theoretical results to measurements on real-world configuration problems shows quite a good match with reality.

3 Dominance filter

In the dominance filter, we choose as the displayed set a set of k *pareto-optimal* solutions, defined as follows:

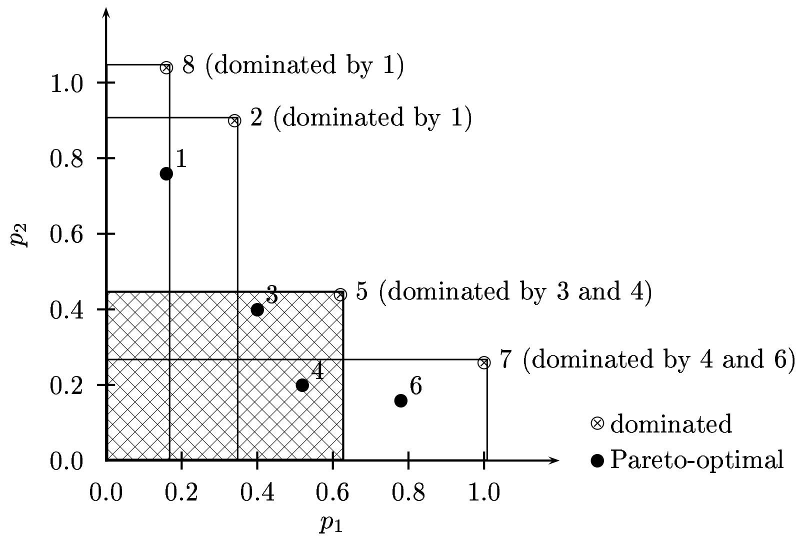


Fig. 1: Example of solutions with two penalties. The two coordinates show the values of preferences p_1 (horizontal) and p_2 (vertical). Rectangles show where dominating solutions must fall. For example, solutions dominating solution 5 must fall into the hatched rectangle.

Definition 2. *A solution S_d is* dominated *with respect to $P^*(u)$ iff there is another solution S' such that for all $p_i \in P^*(u)$, $p_i(S_d) > p_i(S')$.*

A solution S_p is pareto-optimal *iff it is not dominated.*

In Figure 1, the Pareto-optimal set is $\{1, 3, 4, 6\}$, as solution 7 is dominated by 4 and 6, 5 is dominated by 3 and 4, and 2 and 8 are dominated by 1.

From the point of view of decision theory, this is the cleanest, since this does not require any assumption about the weights that are not precisely known.

As shown in Figure 1, a solution S_j with preference values $p_1 = c_1, ..., p_d = c_d$ is dominated by any solution that falls within the subspace $p_1 \in [0..c_1], ..., p_d \in [0..c_d]$, which is a hypercube. Given a uniform distribution of solutions, the average probability that a solution S_j is dominated by another solution S_k is thus equal to the probability that S_k falls into that subspace. This probability is given by the proportion of the subspace with respect to the entire space. Since we assume all attributes to vary between 0 and 1, the size of the entire space is $1^d = 1$, so that the probability is just the size of the subspace, i.e.:

$$Pr(S_j \prec S_k) = \prod_{p_i \in P^*(u)} p_i(S_j)$$

and the probability that a solution S_j is pareto-optimal is the probability that in the m solutions, not a single one dominates S_j:

$$Pr(S_j \text{ is pareto-optimal}) = (1 - \prod_{p_i \in P^*(u)} p_i(S_j))^m$$

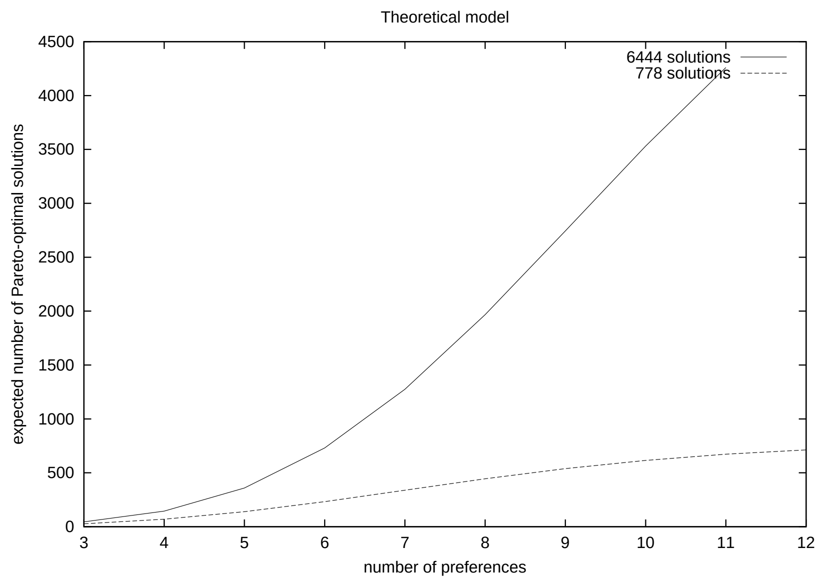


Fig. 2: Number of Pareto-optimal solutions expected from the probabilistic analysis for 2 example scenarios.

The expected number of pareto-optimal solutions is given by integrating this probability over the possible value combinations of the preferences:

$$E[|\{S_j|S_j \text{ is pareto-optimal}\}|] = m \int_0^1 \cdots \int_0^1 (1 - \prod_{p_i \in P^*(u)} p_i)^m dp_1 \cdots dp_d$$

where $d = |P|$ is the number of preferences in the set $P^*(u)$.

Figure 2 shows graphs of this number for two example scenarios that match experiments we made with randomly generated configuration examples[1] shown in Figure 3. The scenarios involve either $m = 778$ and $m = 6,444$ and the number of preferences d ranging from 3 to 12. We can observe that in the experiments the randomly generated preferences are not completely independent of one another, so that the number for 12 preferences in the experiments are already reached with 8 preferences in the theoretical analysis, and similarly 8 preferences in the experiments correspond to about 5 preferences in the theory. However, besides this effect, there is a very good match, showing that the model is quite realistic.

A consequence of this rapid rise in the number of pareto-optimal solutions is that when there are few preferences, there may not be enough solutions to fill the displayed set, while with too many preferences, there will be too many and random sampling will be required. Thus, in order to satisfy condition (1), the

[1] The generation of the randomly generated configuration problems is described in Section 4.

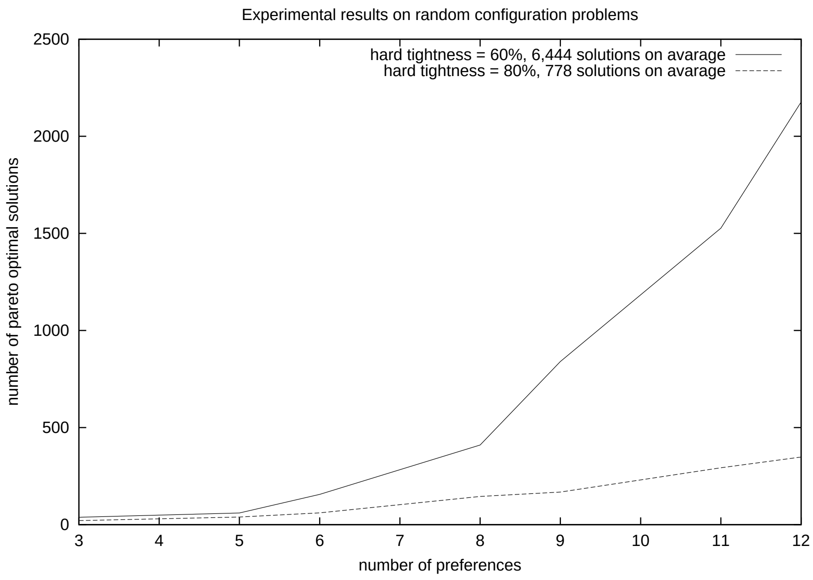


Fig. 3: Average number of Pareto-optimal solutions actually observed on randomly generated configuration examples in the two example scenarios.

filter has to perform random sampling when the number of preferences gets too high.

Condition (2) is trivially satisfied as only solutions that are pareto-optimal in the entire set of feasible solutions are shown.

For verifying condition (3), we assume that the target solution is pareto-optimal. However, as already mentioned, it will often be necessary to make a random selection of solutions to actually display. Thus, the probability of it being included in the k displayed solutions can be estimated as:

$$\begin{aligned} Pr(S_t \text{ is included}) &= \frac{k}{max(k, E[|\{S_j | S_j \text{ is pareto-optimal}\}|])} \\ &= \frac{k}{max(k, m \int_0^1 \cdots \int_0^1 (1 - \prod_{p_i \in P^*(u)} p_i)^m dp_1 \cdots dp_{|P|})} \end{aligned}$$

Figure 4 shows a plot of this probability vs. $|P| = 3, \ldots, 12$ for $m = 778, k = 30$ and $m = 6,444, k = 60$. We can see that the probability of including the target solution rapidly decreases as the overall set of pareto-optimal solutions becomes too large and the target often is no longer selected, even for a relatively small number of preferences. Thus, the method does not satisfy condition (3) very well.

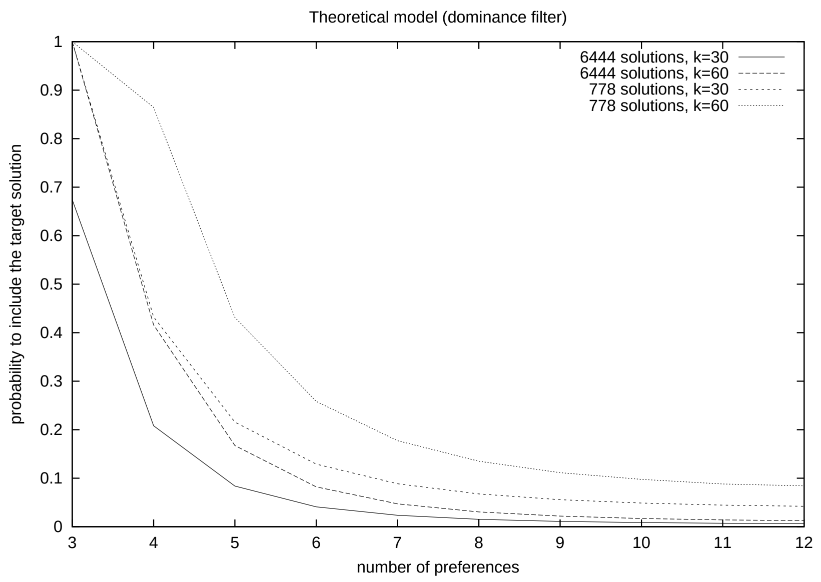


Fig. 4: Probability that the target solution is actually in the displayed set for different numbers of preferences.

3.1 Penalty sum filter

In this model, we choose as displayed solutions the k best solutions according to the unweighted sum of the penalties:

$$C(S) = \sum_{p_i \in P^*(u)} p_i(S)$$

This set can be generated efficiently using branch-and-bound algorithms; in fact it can often be integrated with the generation of the feasible set itself.

The method obviously satisfies condition (1).

The following Theorem shows that it also satisfies condition (2):

Theorem 1
Given a set of m solutions $\mathcal{S} = \{S_1, \ldots, S_m\}$ and a set of d preference penalties $\{p_1, \ldots, p_d\}$. Let be $\mathcal{S}' = \{S_{i_1}, \ldots, S_{i_k}\} \subseteq \mathcal{S}$ the best k solutions:$\forall S' \in \mathcal{S}', \forall S \notin \mathcal{S}' : C(S') \leq C(S)$.

If $S_x \in \mathcal{S}'$ and S_x is not dominated by any other solution $S_y \in \mathcal{S}'$, then S_x is pareto-optimal in $\mathcal{S}$.

Proof. Assume that S_x is not pareto-optimal in $\mathcal{S}$. Then, there is a solution $S_z \notin \mathcal{S}'$ which dominates solution S_x, and by definition:

$$\forall p_i,\, p_i(S_z) \quad \leq \quad p_i(S_x) \text{ and}$$

$$\exists p_j,\, p_j(S_z) \quad < \quad p_j(S_x)$$

As a consequence, we also have:

$$\sum_{i=1}^{d} p_i(S_z) < \sum_{i=1}^{d} p_i(S_x)$$

and therefore $C(S_z) < C(S_x)$. But this contradicts the fact that $S_x \in \mathcal{S}'$ and $S_z \notin \mathcal{S}'$. □

Thus, the method also satisfies condition (2).

To understand how far the method satisfies condition (3), consider the relation between the sums of penalties for the best solutions and the number of preferences.

Let $S_j(t)$ be the j-th best solution according to the sum of the penalties $C(S) = \sum_{p_i \in P^*(u)} p_i(S)$, and let S_t be the target solution. Since the target solution is optimal for the weighted sum, we have:

$$W(S_t) = \sum_{p_i \in P^*(u)} w_i \cdot p_i(S_t) \leq \sum_{p_i \in P^*(u)} w_i \cdot p_i(S_j) \;\; ; \;\; t \neq j$$

As we assume that the weights are close to 1:

$$1 - \epsilon \leq w_i \leq 1 + \epsilon$$

We can rewrite this as:

$$\begin{array}{rcl} \sum_{p_i \in P^*(u)} (1-\epsilon) \cdot p_i(S_t) & \leq & \sum_{p_i \in P^*(u)} (1+\epsilon) \cdot p_i(S_j) \;\; ; \;\; t \neq j \\ (1-\epsilon) \cdot C_t & \leq & (1+\epsilon) \cdot C_j \\ C_t & \leq & \frac{1+\epsilon}{1-\epsilon} C_1 \end{array}$$

where C_1 is the sum of the penalties of the best solution according to the penalty sum filter.

In order to ensure that C_t is among the k best solutions with probability 1 once all preferences are specified, we need to choose k sufficient large so that:

$$C_k / C_1 \geq \frac{1+\epsilon}{1-\epsilon}$$

For example, if we assume a maximum error in the weights of $\epsilon = 0.2$, then $C_k < 1.5 C_1$.

We now consider how the ratio C_k/C_1 depends on the number of stated preferences. The k best solutions are characterized by $C(S) \geq C_k$. Assuming a uniform distribution of solutions, C_k is the height of the volume between the hyperplane $C = C_k$ and the point $p_i = 0$ such that the volume is equal to k/m.

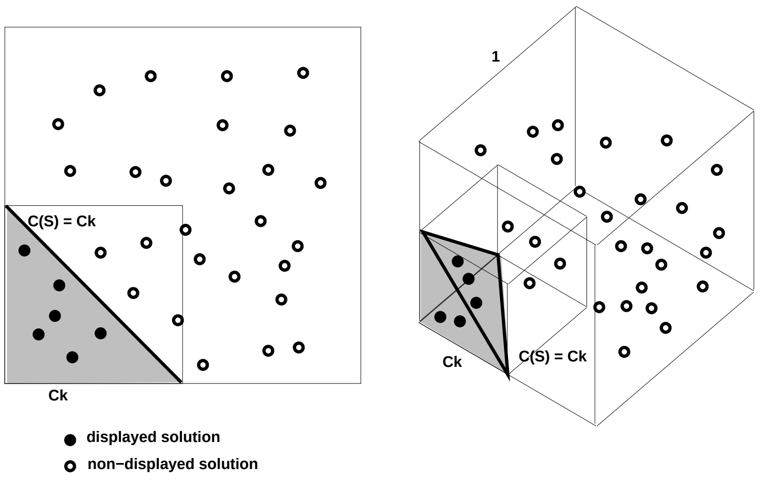


Fig. 5: The volume containing the k solutions with the smallest sum of penalties is bounded by the surface C(S) = Ck. It delimits a fraction $\beta(d)$ of the volume of a cube whose size is equal to Ck.

As shown in Figure 5, the hyperplane $C = C_k$ cuts out a portion $\beta(d)$ of a d-dimensional hypercube H of size C_k, where $\beta(d)$ is a constant depending on the dimension, for example $\beta(1) = 1$, $\beta(2) = 1/2$. H thus has a volume of $\beta(d)C_k^d$. As $\beta(d)$ of the cube should contain a fraction k/m of the solutions, the volume of the displayed set is equal to:

$$\beta(d)C_k^d = k/m$$

and thus

$$C_k = \left(\frac{k}{m\beta(d)}\right)^{1/d}$$

and

$$C_k/C_1 = k^{1/d}$$

which somewhat surprisingly is independent of the total number of solutions m.

We again compare the theoretical curve with the one observed on practical experiments. Figure 7 shows the curve obtained for the same random configuration problems mentioned earlier for $k = 30$ and $k = 60$ displayed solutions. Figure 6 shows the corresponding curves which were generated on random configuration problems. For the case of $m = 6444$ solutions, the curve match exactly, while for $m = 778$ solutions, we observe a significant discrepancy when the number of preferences is low. This is due to the fact that many solutions to the configuration problem have identical quality and thus often score the same when only a few preferences are stated. In general, we can again conclude that the theoretical model is a good match of reality.

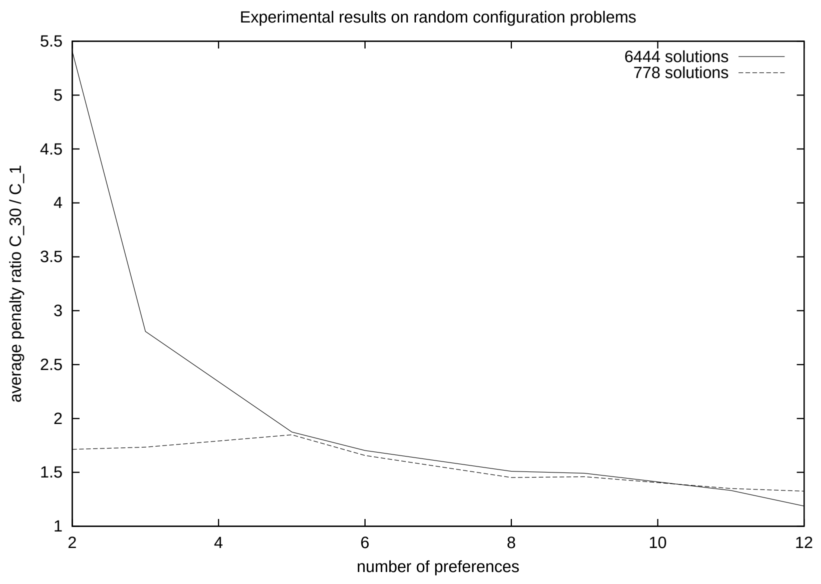


Fig. 6: Theoretically expected ratio of the summed penalty for the k-th best and the best solution as a function of the number of criteria.

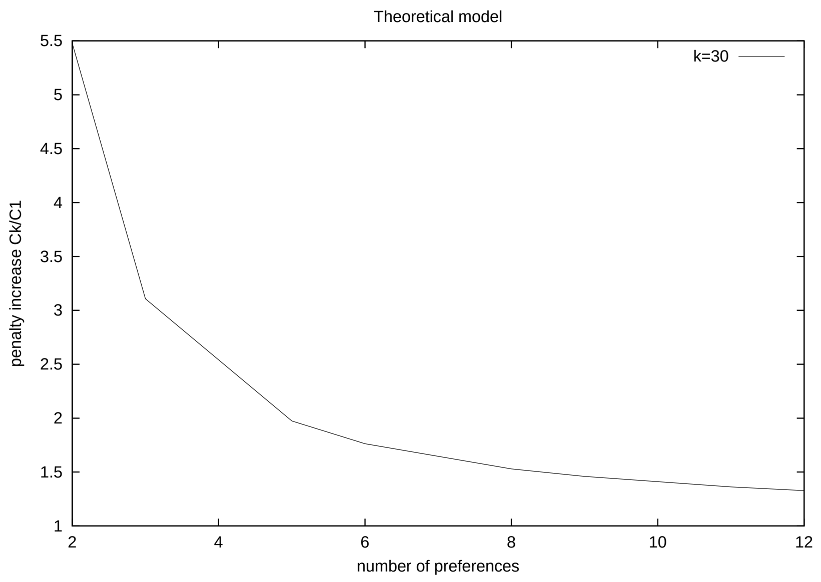


Fig. 7: Observed ratio of the summed penalty for the k-th best and the best solution as a function of the number of criteria.

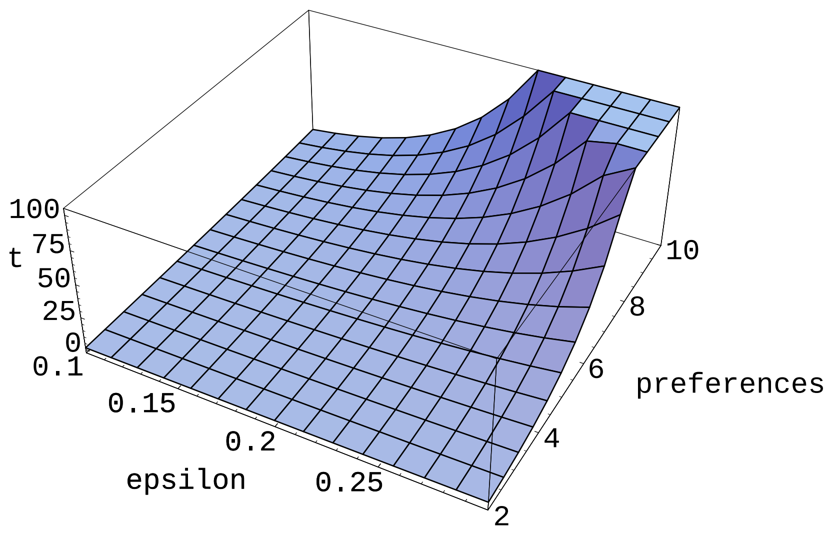


Fig. 8: Size of the displayed set required to ensure that the target solution is displayed, for different values of $|P|$ and ϵ.

Given the error margin ϵ as described above, we can compute the expected maximal rank t of the target solution in terms of the summed penalty metric by the fact that:

$$C_t = C_1 \frac{1+\epsilon}{1-\epsilon} = \left(\frac{t}{m\beta(d)}\right)^{1/d}$$

from which we obtain:

$$\begin{aligned} t &= m\beta(d)\left(\frac{C_1(1+\epsilon)}{1-\epsilon}\right)^d \\ &= \left(\frac{1+\epsilon}{1-\epsilon}\right)^d \end{aligned}$$

This function can be used to calculate the number of solutions that have to be displayed for a given number of preferences and a given tolerance ϵ for user diversity. This can be used in the design of a decision support system, or to adaptively adjust the number of displayed solutions during a decision process. Figure 8 shows a plot of this function.

With the weight errors at the maximum admissible limits, the probability

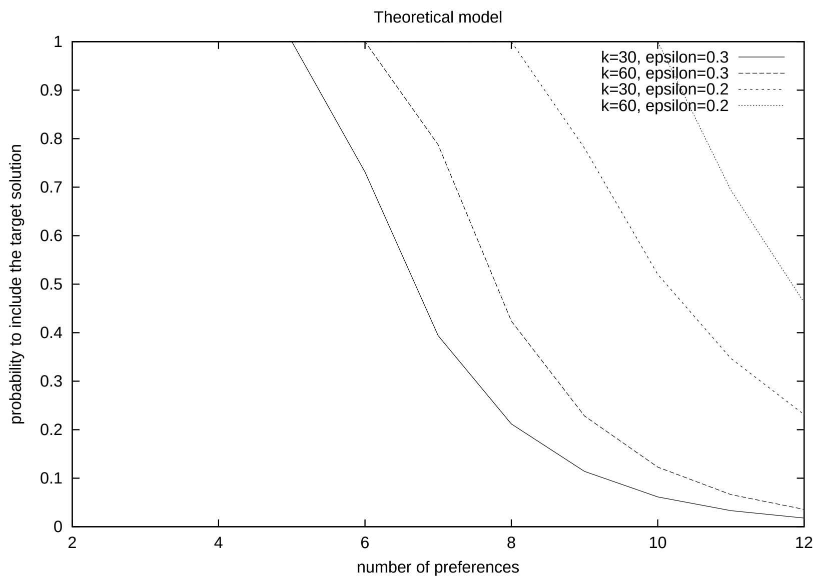


Fig. 9: Probability that the target solution is within the displayed set.

that the target solution lies within the displayed set is given by:

$$pr(S_t \text{ is included}) = \begin{cases} 1 & \text{if t} \leq \text{k} \\ k/t & \text{if t} > \text{k} \end{cases}$$

Figure 9 shows a plot of this probability for $|P| = 2, \ldots, 12$ and combinations of $\epsilon = 0.2, 0.3$ and $k = 30, 60$.

We can see that in comparison to the dominance filter, we can often accommodate significantly more preferences. However, this depends strongly on the weight diversity and the number of displayed solutions. While the approach in principle can be expected to perform better than dominance filtering, careful tuning is still required for each application.

3.2 Fuzzy filter

In this method, we choose as displayed solutions the k solutions that minimize the maximum of any penalty, i.e. that minimize:

$$max_{p_i \in P^*(u)} p_i(S)$$

where ties are broken randomly. The method corresponds to the decision criteria used in fuzzy logic, and has the advantage that it can be very efficiently implemented using constraint satisfaction techniques.

This method obviously satisfies condition (1), as it generates exactly k solutions.

However, it does not at all satisfy condition (2). In fact, on the random configuration problems mentioned earlier, we never found that among of the solutions that were pareto-optimal in the displayed set, there were never more than 2% that were also pareto-optimal in the full set of solutions!

As for condition (3), there is no guarantee that the target solution will be included in the displayed set, as there is no relation between fuzzy and weighted sum optimization criteria. Only if the target solution is dominant, i.e. better than any other solution with respect to all preferences, will it also be the best with respect to the fuzzy criterion.

We thus consider fuzzy filters fundamentally inadequate for this task.

4 Experimental results on randomly generated problems

In order to validate the theoretical results, we have analyzed the concepts described in this paper with randomly generated configuration problems. We model configuration problems with preferences as valued Constraint Satisfaction Problems (VCSP [14, 15]) as described in [12].

A random valued CSP is defined by: $< n, m, hc, ht, sc, st >$, where n is the number of variables in the problem and m the size of their domains. hc is the graph density in percentage for unary and binary hard constraints. ht is the tightness in percentage for disallowed tuples in unary and binary hard constraints. sc and st are the graph density and tightness in percentage for unary and binary soft constraints. Valuations for soft constraints can take values from 0 to 1. For simplicity, hard and soft constraints are separated and we are not considering mixed constraints, therefore $hc + sc \leq 100$.

For building random instances of valued CSPs, we choose the variables for each constraint following a uniform probabilistic distribution. In the same way we choose the tuples in constraints. Valuations for soft tuples are randomly generated between 0 and 1 and valuations for hard tuples are represented by a maximum valuation (∞).

The algorithms have been tested with different set of problems of valued CSPs with 5 variables and 10 values for each variable. Hard unary/binary constraint density hc has been varied from 20% to 80% in steps of 20, and the tightness for hard constraints ht varies also from 20% to 80% in steps of 20. Soft unary/binary constraint density sc has been varied from 20% to 80% in steps of 10, resulting in 3, 5, 8, 9, 11 and 12 soft constraints with tightness $st = 100$. In total there could be $5 + (5 * 4)/2 = 15$ constraints (5 unary constraints and 10 binary constraints).

For every different class of problems, 40 different instances were generated. For the different problem topologies the average of the results for each instance are evaluated.

For simplicity and easing the readability of the graphs presented in this paper, only the graphs for problems with the topology $< n = 5, m = 10, hc = 20, ht = \{80, 60\}, sc = \{20, \ldots, 80\}, st = 100 >$ are shown. The total number of solutions for these two problem topologies ($ht = 60$ and $ht = 80$) on average are 778 and $6,444$.

5 reality: a commercial application for decision support in business travels

reality is an application commercialized by i:FAO[2] which significantly extends the reach of electronic commerce in travel (for more details about this application, refer to [13]). In particular, *reality* addresses the challenge of modeling customers' personal preferences and providing solutions that are tailored to just those preferences. In contrast to existing technology, which allow to optimize only a small and predefined set of preferences, our tool allows a wide variety that can accurately model the preferences of different customers. *reality* thus allows customers to more quickly find solutions of better quality than existing tools.

The idea underlying *reality* is to replace the travel agent while keeping the same interaction model, *i.e.* keeping a kind of dialog between the customer and the system by means of a mixed-initiative system.

Following the analysis presented in this paper, the solution generation adopted for *reality* consists on the *penalty sum filter* presented in Section 3.1. We have chosen $k = 30$ (the number of solutions to be displayed) as the optimal number since we found that users typically state between 3 and 6 preferences for a trip and that $\epsilon = 0.3$ is a good upper bound on their diversity.

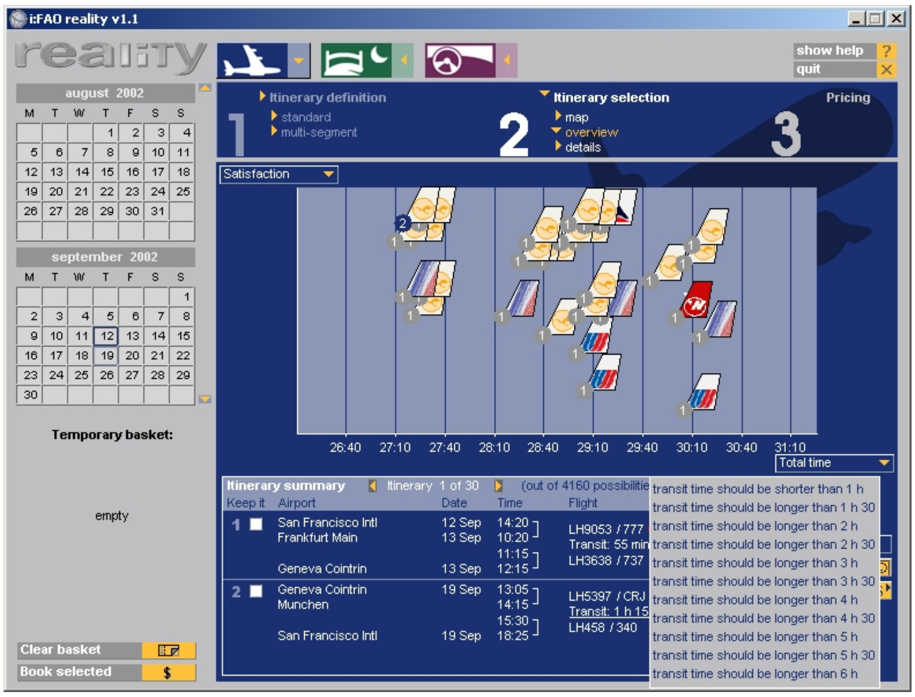
Figure 10 shows how *reality* displays the best 30 solutions out of the generation process to the user. A scatterplot (inspired by [16]) shows how solutions are positioned with respect to two criteria. The criteria to visualize the solutions are chosen by the user who can easily evaluate the characterization of the solution space. Below the scatterplot, the details of the current selected solution are displayed by a standard table of flight attributes. Preferences can be posted in any attribute of the shown solution. In the example of Figure 10, the user is posting a preference on the transit time for the itinerary segment from Geneva to San Francisco. The scatterplot shows the 30 best solutions with respect to criterion *satisfaction* and total travel time. The *satisfaction* criterion represents the penalty sum of all the preferences, *i.e.* the quality of the solutions.

6 Conclusions

In this paper, we have considered the problems posed by the use of a qualitative preference model in decision aid systems. We have considered compensating for the inaccuracy of a qualitative model by letting the user choose among a displayed set of alternatives. A key issue in such a system is then how to select the solutions to be displayed to the user to ensure that the target solution can indeed be found.

We have shown that of three candidate methods, only the penalty sum filter is sufficiently robust against incompleteness and approximativeness of the model

[2] `http://www.ifao.net`



Itinerary summary ◀ Itinerary 1 of 30 ▶ (out of 4160 possibilitie

Keep it		Airport	Date	Time	Flight
1	☐	San Francisco Intl	12 Sep	14:20	LH9053 / 777
		Frankfurt Main	13 Sep	10:20	Transit: 55 min
				11:15	LH3638 / 737
		Geneva Cointrin	13 Sep	12:15	
2	☐	Geneva Cointrin	19 Sep	13:05	LH5397 / CRJ
		Munchen		14:15	Transit: 1 h 15
				15:30	LH458 / 340
		San Francisco Intl	19 Sep	18:25	

transit time should be shorter than 1 h
transit time should be longer than 1 h 30
transit time should be longer than 2 h
transit time should be longer than 2 h 30
transit time should be longer than 3 h
transit time should be longer than 3 h 30
transit time should be longer than 4 h
transit time should be longer than 4 h 30
transit time should be longer than 5 h
transit time should be longer than 5 h 30
transit time should be longer than 6 h

Fig. 10: The user analyzes the solutions shown in a starfield where two criteria are compared (in this case satisfaction vs. total flying time).

to guarantee (at least with the proper size of the displayed set) that the target solution can be found by the user.

More precisely, we can show that the decision aid system using the penalty sum filter is sound, i.e. finds the target solution, if the size of the displayed set k is sufficiently large for the ϵ and the number of preferences that the users poses, and if the user correctly and completely states his preferences. While the first condition can be satisfied by experiments during the development of the system, the second condition would deserve further study.

Acknowledgments

We would like to sincerely thank all the people that contributes, or contributed, to the development of *reality*.

References

[1] Keeney, R. and Raiffa, H., Decision with multiple objectives: Preferences and value tradeoffs. 1976: Cambridge University Press.

[2] Keeney, R., Value-Focused Thinking: A Path to Creative Decision Making. Harvard University Press, 1992.

[3] V. Ha and P. Haddawy. Problem-focused incremental elicitation of multiattribute utility models. Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, pp. 215–222, August 1997.

[4] V. Ha and P. Haddawy. Toward Case-Based Preference Elicitation: Similarity Measures on Preference Structures, in Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence. 1998.

[5] Stolze, M. (2000). Soft Navigation in electronic product catalogs. International Journal on Digital Libraries 3(1): 60-66.

[6] J. Doyle and R. Thomason. Background to qualitative decision theory. AI Magazine **20**(2), Summer 1999.

[7] C. Boutilier, R. Brafman, C. Geib, and D. Poole. A Constraint-Based Approach to Preference Elicitation and Decision Making. AAAI Spring Symposium on Qualitative Decision Theory, Stanford, 1997.

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

[9] Pu, P. and Faltings, B. Enriching Buyers' experiences: the SmartClient Approach, in Proceedings of The ACM SIGCHI Conference on Human Factors in Computing Systems, 2000.

[10] R. Burke, K. Hammond, and B. Young. The findme approach to assisted browsing. In IEEE Expert, volume 12(4), pages 32–40, 1997.

[11] S. Shearin and H. Lieberman. Intelligent Profiling by Example. Proceedings of the International Conference on Intelligent User Interfaces (IUI 2001), pp. 145-152. Santa Fe, NM, January 14-17, 2001.

[12] Marc Torrens and Boi Faltings. Multi-criteria Optimization in Constraint Satisfaction Problems. AAAI-02 workshop on preferences in AI and CP: symbolic approaches, 2002, Edmonton, Alberta, Canada.

[13] Marc Torrens and Patrick Hertzog and Loic Samson and Boi Faltings. *reality*: a Scalable Intelligent Travel Planer, Decision Support for the Business Traveler. Proceedings of the Eighteenth Annual ACM Symposium on Applied Computing, 2003, Melbourne, Florida, USA.

[14] Thomas Schiex, Hélène Fargier and Gérard Verfaillie. Valued Constraint Satisfaction Problems: Hard and Easy Problems, IJCAI, 1995, pp. 631-637.

[15] Stefano Bistarelli, Hélène Fargier, Ugo Montanari, Francesca Rossi, Thomas Schiex and Gérard Verfaillie. Semiring-based CSPs and Valued CSPs: Basic Properties and Comparison. CONSTRAINTS: an international journal, 1999, Kluwer Academic Publishers, 4(3).

[16] Christopher Ahlberg and Ben Shneiderman. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. Human Factors in Computer Systems: Proceedings of the CHI'94 Conference, 1994, New York, pp. 313-317.