Preferences, Constraints, Uncertainty, and Multi-Agent Scenarios [*]

Francesca Rossi
Department of Pure and Applied Mathematics
University of Padova, Italy
frossi@math.unipd.it

Abstract

Preferences occurr in many everyday tasks. Whether we look for a house, a car, a computer, a digital camera, or a vacation package, we usually state our own preferences and we expect to find the item of our dreams. It is therefore natural that modelling and solving sets of preferences is a central issue in AI, if we want to understand human intelligence and use computing devices to replicate some of functions of the human brain.

This paper will discuss different kinds of preferences, will describe and compare some of the AI formalisms to model preferences, and will hint at existing preference solvers. Uncertainty will also be considered, in the form of a possibly incomplete set of preferences, because of privacy issues or missing data. We will also discuss multi-agent settings where possibly incomplete preferences need to be aggregated, and will present results related to both normative and computational properties of such systems.

While the results on single-agent preference solving are mostly related to AI sub-areas such as constraint programming and knowledge representation, those on multi-agent preference aggregation are multi-disciplinary, since preference aggregation and its properties have been extensively studied also in in decision theory, economy, and political sciences.

1 Preference formalisms in AI

Constraints are requirements that should be met. For example, when choosing a camera, we may be interested only in those with a telephoto lens. On the other hand, preferences model the fact that some situations are more desirable than others. When problems are over-constrained, as they often are, we can use preferences to choose which constraints are not satisfied.

Preferences can be quantitative or qualitative (e.g., ”I like skiing very much” versus ”I like swimming more than skiing”). Preferences can also be conditional (e.g., ”If the main dish is fish, I prefer white wine to red wine”) or bipolar (e.g., ”I like swimming very much and I slightly dislike running”).

Preferences and constraints may also co-exist. For example, in a product (such as a car) configuration problem, there may be production constraints (for example, a limited number of convertible cars can be built each month), marketing preferences (for example, that it would be better to sell the standard paint types), as well as user preferences of various kind (for example, that if it is a sport car, she prefers red).

Because constraints and preferences often occur together in real life, representing and reasoning about preferences is an area of increasing interest within Artificial Intelligence. Unfortunately, there is no single formalism which allows the different kinds of preferences to be specified efficiently and reasoned with effectively. For example, soft constraints (Bistarelli, Montanari, & Rossi 1997) are most suited for reasoning about constraints and quantitative preferences, while CP-nets (Boutilier *et al.* 2004) are most suited for representing qualitative and possibly conditional preferences.

In the following two subsections, we will describe these two formalisms, that present very different advantages and drawbacks.

1.1 Soft constraints

Soft constraints (Bistarelli, Montanari, & Rossi 1997; Meseguer, Rossi, & Schiex 2006) model quantitative preferences by generalizing the traditional formalism of hard constraints (Rossi, Beek, & Walsh 2006). In a soft constraint, each assignment to the variables of a constraint is annotated with a level of its desirability, and the desirability of a complete assignment is computed by a combination operator applied to the local preference values. By choosing a specific combination operator and an ordered set of levels of desirability, we can select a specific class of soft constraints.

For example, in fuzzy constraints (Dubois, Fargier, & Prade 1993), preferences are between 0 and 1 (with 1 better than 0), and min is the combination operator. In other words, fuzzy constraints employ a very pessimistic approach, in which only the worst event is taken into account to evaluate a whole scenario.

Weighted CSPs instead have preferences over the integers, with higher integers denoting a lower preference, and with sum as the combination operator. This setting is what is usually used in cost-based scenarios, where the goal is to minimize the sum of the costs.

Classical constraints (that is, hard requirements) can be seen as soft constraints where there are only two levels of

[*]This paper summarizes joint work with S. Bistarelli, C. Domshlak, M. Gelain, M. S. Pini, K. B. Venable, T. Walsh.

preference (true and false), where true is better than false, and where the combination operator is logical and. Thus a scenario is evaluated positively only when all its constraints are satisfied, otherwise it is rejected.

Given a set of soft constraints, an ordering is induced over the assignments of the variables of the problem, which can be partial or total, and can also have ties.

Given two solutions of a soft constraint problem, checking whether one is preferable to the other one is easy: we compute the preference values of the two solutions and compare them in the preference order. However, finding an optimal solution for a soft constraint problem is a combinatorially difficult problem. Many search techniques have been developed to solve specific classes of soft constraints, like fuzzy or weighted. However, all have an exponential worst case complexity. Systematic approaches like backtracking search and constraint propagation, can be adapted to soft constraints. For example, backtracking search becomes branch and bound where the bounds are given by the preference levels in the constraints. Constraint propagation, which is very successful in pruning parts of the search tree in constraint solving, can also be generalised to certain classes of soft constraints.

1.2 CP-nets

CP-nets (Boutilier *et al.* 2004) (Conditional Preference networks) are a graphical model for compactly representing conditional and qualitative preference relations. They exploit conditional preferential independence by structuring a user's possibly complex preference ordering with the ceteris paribus assumption. CP-nets are sets of conditional ceteris paribus preference statements (cp-statements). For instance, the statement ”I prefer red wine to white wine if meat is served” asserts that, given two meals that differ only in the kind of wine served and both containing meat, the meal with a red wine is preferable to the meal with a white wine.

CP-nets bear some similarity to Bayesian networks, as both utilize directed acyclic graphs where each node stands for a domain variable, and assume a set of features with finite, discrete domains (these play the same role as variables in soft constraints). Given a CP-net, an ordering is induced over the set of assignments of its features. This ordering is, in the most general case, a preorder (that is, reflexive and transitive).

The Achille's heel of CP-nets and other sophisticated qualitative preference models is the complexity of reasoning with them. Given an acyclic CP-net, finding an optimal assignment to its features can be done in linear time. However, for cyclic CP-nets, it becomes NP-hard. Even worse, comparing two outcomes is NP-hard, even when the CP-net is acyclic.

1.3 Comparing their expressive power

We could say that a formalism B is at least as expressive than a formalism A iff from a problem expressed using A it is possible to build in polynomial time a problem expressed using B such that the optimal solutions are the same. If we apply this to the comparison of some classes of soft constraints, we see for example that fuzzy CSPs and weighted CSPs are at least as expressive as classical constraints. If instead we use it compare CP-nets and soft constraints, we see that classical constraints are at least as expressive as CP-nets. In fact, it is possible to show that, given any CP-net, we can obtain in polynomial time a set of classical constraints whose solutions are the optimal outcomes of the CP-net. On the contrary, there are some classical constraint problems for which it is not possible to find in polynomial time a CP-net with the same set of optimals.

However, we could be more fine-grained in the comparison, and say that a formalism B is at least as expressive than a formalism A iff from a problem expressed using A it is possible to build in polynomial time a problem expressed using B such that the orderings over solutions are the same. Here not only we must maintain the set of optimals, but also the rest of the ordering over the solutions. In this case, CP-nets and soft constraints are incomparable.

Summarizing, CP-nets and soft constraints have complementary advantages and drawbacks. CP-nets allow one to represent conditional and qualitative preferences, but dominance testing is expensive. On the other hand, soft constraints allow to represent both hard constraints and quantitative preferences, and have a cheap dominance testing.

2 Uncertainty

Traditionally, tasks such as scheduling, planning, and resource allocation have been tackled using several techniques, among which constraint reasoning is one of the winning ones: the task is represented by a set of variables, their domains, and a set of constraints, and a solution of the problem is an assignment to all the variables in their domains such that all constraints are satisfied. Preferences or objective functions have been used to extend such scenario and allow for the modelling of constraint optimization, rather than satisfaction, problems. However, what is common to all these approaches is that the data (variables, domains, constraints) is completely known before the solving process starts.

On the contrary, the increasing use of web services and in general of multi-agent applications demands for the formalization and handling of data that is only partially known when the solving process works, and that can be added later, for example via elicitation. In many web applications, data may come from different sources, which may provide their piece of information at different times. Also, in multi-agent settings, data provided by some agents may be voluntarily hidden due to privacy reasons, and only released if needed to find a solution to the problem.

2.1 Missing preferences

These issues can be considered in the context of constraint optimization problems. In particular, we can consider problems with soft constraints, where variables, domains, and constraint topology is given at the beginning, while the preferences can be partially specified and possibly added during the solving process.

There are several application domains where such setting is useful. One regards the fact that quantitative preferences, as needed in soft constraints, may be difficult and tedius to provide for a user. Another one concerns multi-agent settings, where agents agree on the structures of the problem by they may provide their preferences on different parts of the problem at different times. Finally, some preferences can be initially hidden because of privacy reasons.

Although some of the preferences can be missing, it could still be feasible to find a solution which is optimal independently on what the missing preferences will be. If not, we can ask the user for help and we start again from the new problem with some added preferences.

More precisely, we consider two notions of optimal solution: *possibly optimal* solutions are assignments to all the variables that are optimal in *at least one way* currently unspecified preferences can be revealed, while *necessarily optimal* solutions are assignments to all the variables that are optimal in *all ways* in which currently unspecified preferences can be revealed.

Given an incomplete soft constraint problem (ISCSP), its set of possibly optimal solutions is never empty, while the set of necessarily optimal solutions can be empty. Of course what we would like to find is a necessarily optimal solution, to be on the safe side: such solutions are optimal regardless of how the missing preferences would be specified. However, since such set may be empty, in this case there are two choices: either to be satisfied with a possibly optimal solution, or to ask users to provide some of the missing preferences and try to find, if any, a necessarily optimal solution of the new ISCSP. If we follow this second approach, we can repeat the process until the current ISCSP has at least one necessarily optimal solution.

To achieve this, we can first check whether the given problem has a necessarily optimal solution (by just solving the completion of the problem where all unspecified preferences are replaced by the worst preference). If not, we can find the most promising among the possibly optimal solutions (where the promise is measured by its preference level), and asks the user to reveal the missing preferences related to such a solution. This second step is then repeated until the current problem has a necessarily optimal solution.

Experimental results with various versions of this basic algorithm on randomly generated problems show that a necessarily optimal solution can be found by eliciting a very small percentage of the missing preferences (Gelain *et al.* 2007).

2.2 Uncontrollable variables

Besides the scenarios where some preference values are missing, there are other ways in which uncertainty can occur in a problem with preferences. One of the most commnly considered is the one in which some variables are uncontrollable, that is, their value cannot be chosen by the agent. This can model situations where there are parts of the problem that are under the control of some other agent, or of Nature.

A typical example of an uncontrollable variable, in the context of satellite scheduling, or weather prediction, is a variable representing the time when clouds will disappear. A more general setting in which uncertainty occurs are scheduling problems, which constrain the order of execution of various activities, and where the durations of some activities are uncertain. In this case the goal is to define a schedule which is the most robust with respect to uncertainty.

While it may happen that no information is available on the uncontrollable variables, usually there is some information, possibly related to historic data. This usually provides a probability or possibility distribution over the values in the domain of the uncontrollable variable.

Possibilities (Zadeh 1978) are useful to model imprecise probabilities, since they provide an upper and a lower bound to probabilities. They have been shown to be useful in data analysis, structural learning, diagnosis, belief revision, argumentation, and case-base reasoning. Moreover, experimental results in cognitive psychology suggest that there are situations where people reason about uncertainty using the rules of possibility theory, rather than those of probability theory.

We model a real-life problem via a set of variables (some controllable and some not controllable) with finite domains and a set of soft constraints among subsets of the variables. In particular, we consider *fuzzy preferences*, that are useful in many applications, like for example the spatial applications and the medical ones, where it is required to have a pessimistic approach.

Fuzzy constraints allow for an easy integration with possibilities, since both possibilities and fuzzy preferences are values between 0 and 1 associated to events, and express the level of plausibility that the event will occur, or its preference.

For the solutions of this kind of problems, we can define the notion of preference and robustness, as well as several desirable properties that such notions should respect, also in relation to the solution preference ordering.

To find the optimal solutions of such problems, we can eliminate the uncontrollable variables (and all the constraints connecting them) and add new fuzzy constraints in the controllable part. Such new constraints contain, in a different form, some of the information that was present in the removed part. While the old constraints are used to compute the preference of a solution, these additional constraints are used to compute its robustness (Pini, Rossi, & Venable 2005).

Several semantics that use the notions of preference and robustness can be defined to order the solutions. For all such semantics, we can check whether the desired properties hold. The semantics that take into account both preference and robustness are a finer way to evaluate the solutions of these problems, since they allow us to distinguish between highly preferred solutions which are not robust, and robust but not preferred solutions. It is also possible to see that they respect all the desired properties. For example, they guarantee that, if there are two solutions with the same robustness (resp., the same preference), then the ordering is given by their preference (resp., robustness).

3 Multi-agent preference aggregation

In many situations, we need to represent and reason about the simultaneous preferences of several agents, and to aggregate such preferences. To deal with these situations, let us assume that we have a number of agents which represent their preferences via CP-nets. To aggregate the agents' preferences, we can query each CP-net in turn and collect together the results. We can see this as each agent "voting" whether an outcome dominates another. We can thus obtain different semantics by collecting these votes together in different ways (Rossi, Venable, & Walsh 2004).

For example, to obtain a Pareto-like semantics, we can say that an outcome A is better than another one, B, iff every agent says that A is better than B or that they are indifferent. An alternative criterion, that we may call majority, is that A is better than B iff a majority of the agents who are not indifferent vote in favor. A weaker criterion, that we may call Max, is that more agents vote in favor than against or for incomparability. Sometimes it is reasonable to assume that the agents are ordered in importance. If the first agent orders two outcomes then this is reflected in the final outcome. However, if they are indifferent between two outcomes, we consult the second agent, and so on. We say that A is lexicographically better than B iff there exists some distinguished agent such that all agents higher in the order are indifferent between A and B, and the distinguished agent votes for A.

3.1 Some normative properties: fairness and non-manipulability

Having cast our preference aggregation semantics in terms of voting, it is appropriate to ask if there is a way to aggregate the preferences that is "reasonable" according to some criteria. This is question asked by Arrow, who answered it with his famous impossibility theorem (Kelly 1978). In short, Arrow's theorem states that no voting system with two or more agents and which totally orders three or more candidates can be fair. More precisely, no voting system can be independent to irrelevant alternatives, unanimous, and non-dictatorial.

Observe that our context is different from Arrow' one, since the agents' preference orderings are not necessarily total orders, but can present incomparability between pairs of candidates. In fact, CP nets do not necessarily provide a total order. The same would hold if the agents used soft constraints rather than CP-nets.

The following properties of preference aggregation are related to fairness:

- Unanimity: if all agents agree that A is preferable to B, then the resulting order must agree as well.
- Independence to irrelevant alternatives: the ordering between A and B in the result depends only on the relation between A and B given by the agents.
- Absence of a dictator: Here we may have three definitions of dictator:
  - Strong dictator: an agent such that, no matter what the other agents say, its ordering is the resulting ordering;
  - Dictator: an agent such that, no matter what the others say, if it prefers A to B, then the resulting ordering must say the same;
  - Weak dictator: an agent such that, no matter what the others say, if it prefers A to B, then the resulting ordering cannot prefer B to A.

A preference aggregation system is strongly fair, fair or weakly fair if its is unanimous, independent to irrelevant alternatives, and does not have a strong dictator, dictator or weak dictator respectively. Arrow's impossibility theorem shows that, if preference aggregation is on total orders, and a preference aggregation system is unanimous, independent to irrelevant alternatives and there are at least two voters and three outcomes, then there must be at least one dictator.

In our partially ordered context, it is possible to be fair. In fact, for example, the Pareto semantics is fair. Since fairness implies strong fairness, preference aggregation can be strongly fair as well. Actually strong fairness is a very weak property to demand. Even voting rules which appears very unfair may not have a strong dictator. For example, the Lex preference aggregation semantics is not fair, but it is strongly fair.

However, as in Arrow's result for totally ordered preferences, it is impossible to be weakly fair (Pini *et al.* 2005). Thus the possible presence of incomparable candidates, both in the agents' preference ordering and in the social ordering, is not helpful in this respect.

Fairness is just one of the desirable properties for preference aggregations (Arrow, Sen, & Suzumura 2002). Other interesting properties are related to the non-manipulability of a preference aggregation system: if an agent can vote tactically and reach its goal, then the system is manipulable. Results for totally ordered preferences show that non-manipulability implies the existence of a dictator (Gibbard 1973). Unfortunately, this continues to hold also for partially ordered preferences (Rossi *et al.* 2006).

3.2 Uncertainty and computational properties

Some parts of the agents' preference ordering may be missing. Thus, a pair of outcomes can be ordered, incomparable, in a tie, or the relationship between them may be unspecified.

Notice that incomparability and incompleteness represent very different concepts. Outcomes may be incomparable because the agent does not wish very dissimilar outcomes to be compared. For example, we might not want to compare a biography with a novel as the criteria along which we judge them are just too different. Outcomes can also be incomparable because the agent has multiple criteria to optimize. For example, we might not wish to compare a faster but more expensive laptop with a slower and cheaper one. Incompleteness, on the other hand, represents simply an absence of knowledge about the relationship between certain pairs of outcomes. Incompleteness arises naturally when we have not fully elicited an agent's preferences or when agents have privacy concerns which prevent them revealing their complete preference ordering.

We now must aggregate preferences taking into account the incompleteness of the agents' preference orderings. We can do this by considering all possible ways in which the incomplete preference orders can be consistently completed.In each possible completion, preference aggregation may give different optimal elements (or *winners*). This leads to the idea of the *possible winners* (those outcomes which are winners in at least one possible completion) and the *necessary winners* (those outcomes which are winners in all possible completions)Possible and necessary winners are useful in many scenarios including preference elicitation. In fact, elicitation is over when the set of possible winners coincides with that of the necessary winners. In addition, preference elicitation can focus just on the incompleteness concerning those outcomes which are possible and necessary winners. We can ignore completely all other outcomes.

Computing the set of possible and necessary winners is in general a difficult problem. However, there are sufficient conditions that assure tractability (Pini *et al.* 2007). Such conditions concern properties of the preference aggregation function, such as monotonicity and independence to irrelevant alternatives (Arrow, Sen, & Suzumura 2002), which are desirable and natural properties to require.

Some of the impossibility results cited above for normative properties of voting rules can be partially circumvented via the use of computational complexity results. For example, while non-manipulability is impossible for non-dictatorial voting rules, it may be computationally hard to manipulate a rule. A rule which is manipulable by hard to manipulate can still be considered reasonable to be used. Thus, while computational hardness is usually bad, it can indeed be useful in this respect.

4 Conclusions

Preferences are of various kinds and occur frequently in real-life problems. There are many preference modelling formalism, each of which succeeds in modelling some aspects of preferences while negletting others.

From a constraint perspective, preferences can be seen as a way to model a form of uncertainty about the statements of the problem. Other forms of uncertainty can involve missing preferential data, as well as uncontrollable variables.

In multi-agent settings, preferences have to be aggregated. Both normative and computational properties of preference aggregation are essential to provide a good multi-agent preference reasoning scenario. In fact, impossibility results on normative properties can sometimes be partially circumvented by means of computational complexity results.

The ideal scenario would be one where a single formalism would be able to accomodate for several different kinds of preferences and uncertainty. This would make life easier for a user and would allow for a more systematic study of the properties of a certain scenario.

We believe that Artificial Intelligence and Mathematics can do a lot if they work together towards this goal. AI can provide vision and motivation, together with modelling and complexity developments, while Mathematics can provide the formal machinery, especially for uncertainty and multi-agent preference aggregation.

References

Arrow, K. J.; Sen, A. K.; and Suzumura, K. 2002. *Handbook of social choice and welfare*. Elsevier.

Bistarelli, S.; Montanari, U.; and Rossi, F. 1997. Semiring-based Constraint Solving and Optimization. *Journal of the 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.

Dubois, D.; Fargier, H.; and Prade, H. 1993. The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In *Proc. IEEE International Conference on Fuzzy Systems*, 1131–1136. IEEE.

Gelain, M.; Pini, M. S.; Rossi, F.; and Venable, K. B. 2007. Dealing with incomplete preferences in soft constraint problems. In *Proc. CP 2007*. Springer LNCS 4741.

Gibbard, A. 1973. Manipulation of voting schemes: a general result. *Econometrica* 41(4).

Kelly, J. S. 1978. *Arrow Impossibility Theorems*. Academic Press.

Meseguer, P.; Rossi, F.; and Schiex, T. 2006. Soft constraints. In Rossi, F.; Beek, P. V.; and Walsh, T., eds., *Handbook of Constraint Programming*. Elsevier.

Pini, M. S.; Rossi, F.; Venable, K. B.; and Walsh, T. 2005. Aggregating partially ordered preferences: possibility and impossibility results. In *Proc. TARK X*. ACM Digital Library.

Pini, M.; Rossi, F.; Venable, K.; and T.Walsh. 2007. Incompleteness and incomparability in preference aggregation. In *Proc. IJCAI 2007*.

Pini, M. S.; Rossi, F.; and Venable, K. B. 2005. Possibility theory for reasoning about uncertain soft constraints. In *Proc. ECSQARU 2005*. Springer.

Rossi, F.; Beek, P. V.; and Walsh, T. 2006. *Handbook of Constraint Programming*. Elsevier.

Rossi, F.; Pini, M. S.; Venable, K. B.; and Walsh, T. 2006. Strategic voting when aggregating partially ordered preferences. In *Proc. AAMAS 2006*.

Rossi, F.; Venable, K. B.; and Walsh, T. 2004. mcp nets: Representing and reasoning with preferences of multiple agents. In *Proc. AAAI 2004*. AAAI Press.

Zadeh, L. 1978. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems* 1:3–28.