Imprecise Soft Constraint Problems

Mirco Gelain[1], Maria Silvia Pini[1], Francesca Rossi[1], K. Brent Venable[1], and Nic Wilson[2]

[1] Dipartimento di Matematica Pura ed Applicata, University of Padova, Italy
E-mail: {mgelain,mpini,frossi,kvenable}@math.unipd.it
[2] Cork Constraint Computation Centre, University College Cork, Ireland,
Email: n.wilson@4c.ucc.ie

Abstract. We define interval-valued soft constraints, where users can associate an interval of preference values, rather than a single value, to each instantiation of the variables of the constraints. This allows us to model a form of uncertainty and imprecision that is often found in real-life problems. We then define several notions of optimal solutions for such problems, providing algorithms to find optimals and also to test whether a solution is optimal. Besides the usefulness of the algorithms, that can be the base for an environment where to reason with uncertainty in soft constraints problems, it is important to notice that most of the times these algorithms require the solution of a soft constraint problem. This means that it is possible to handle uncertainty in soft constraints without increasing the computational effort to reason with such problems. We also show that the same results hold if users are allowed to use multiple disjoint intervals rather than a single one.

1 Introduction

Constraints [3, 8] are useful to model real-life problems when it is clear what should be accepted and what should be forbidden. Soft constraints [1, 7] extend the constraint notion by allowing several level of acceptance. This allows to express preferences rather than (and besides) strict requirements.

However, in soft constraints, each instantiation of the variables of a constraint must be associated to a precise preference value. Sometimes it is not possible for a user of a soft constraint system to know exactly all these values. For example, a user may have a vague idea of the preference value. Also, a user may not be willing to reveal his preference, for example for privacy reasons. In this paper we consider these forms of imprecision, and we provide a formalism and reasoning tools to handle them.

In particular, we extend soft constraints by allowing users to state an interval of preference values for each instantiation of the variables of a constraint. This interval can contain a single element (in this case we have usual soft constraints), or the whole range of preference values (when there is complete ignorance about the preference value), or it may contain more than one element but a strict subset of the set of preference values. In an elicitation procedure there will typically be some degree of imprecision, so attributing an interval rather than a precise preference degree can be a more reliable model of the information elicited. In particular, linguistic descriptions of degrees of preference (such as ”quite high” or ”low” or ”undesirable”) may be more naturally mapped to preference intervals, especially if the preferences are being elicited from different experts, since they may mean somewhat different things by these terms. We call such problems *interval-valued* CSPs (or also IVCSPs).

Two examples of real world application domains where preference intervals can be useful or necessary are energy trading and network traffic analysis [10], where the data information is usually incomplete or erroneous. In energy trading costs may be imprecise since they may evolve due to market changes; in network traffic analysis the overwhelming amount of information and measurement difficulties force the use of partial or imprecise information.

In an IVCSP, a *scenario* is a soft constraint problem (SCSP) obtained by selecting a specific preference value from each interval. Also, given any solution s (that is, a complete assignment of the variables) of an IVCSP, we can associate to it two preference values $L(s)$ and $U(s)$, obtained by combining all the lower bounds (resp., upper bounds) in the intervals that s selects in the constraints.

Given an IVCSP, we consider several notions of optimal solutions. We first start with interval-based notions. For example, we define *lower-optimal* solutions, which are complete variable assignments s such that there is no other assignment s' with $L(s')$ better than $L(s)$. Such solutions are optimal in the worst scenario, that is, in the scenario obtained by selecting the lower bound in every interval. They are therefore useful in a pessimistic approach to uncertainty, since they outperform the other assignments in the worst scenario. We also define several other notions of optimal solutions, which may be interesting in other approaches to uncertainty.

We then provide algorithms to find such optimal solutions, and also to test whether a given solution is optimal. In most of the cases, finding or testing an optimal solution amounts at solving a soft constraint problem. Thus, even if our formalism significantly extends soft constraints, and gives users much more power in modelling his knowledge of the real world, in the end the work needed to find an optimal solution (or to test it is optimal) is not more than that needed to find an optimal solution in a soft constraint problem.

We then pass to more general notions of optimality, which do not refer to intervals but to more general ideas that apply whenever we have several scenarios to consider. For example, as in [5], we consider *necessarily optimal* solutions, which are optimal in all scenarios, or *possibly optimal* solutions, which are optimal in at least one scenario. We also consider solutions that guarantee a certain level of preference in all (resp., some) scenarios, and we aim to find those that guaranteee the highest level.

By relating these general notions of optimal solutions to the specific ones based on intervals, we are then able to provide algorithms to find or test optimal solutions according to these notions. Again, it is very important to notice that solving a soft constraint problem is almost always enough, thus confirming that it is possible to handle uncertainty in soft constraints without increasing the computational effort to reason with such problems.

The optimality notions considered in this paper would not produce different results if users were allowed to use multiple disjoint intervals rather than a single one. This means that a level of precision greater than a single interval does not add any useful information when looking for an optimal solution.

Previous approaches to uncertainty in soft constraint problems assumed either a complete knowledge of the preference value, or a complete ignorance. In other words, a preference value in a domain or a constraint was either present or not [4–6, 9]. Then, the solver was trying to find optimal solutions with the information given by the user or via some form of elicitation of additional preference values. Here instead we consider a more general setting where the user may specify preference intervals. Also, we assume that the user has given us all the information he has about the problem, so we do not resort to preference elicitation.

The most related work is the one presented in [5]. However, while [5] considers the same interval for all preference values, which is the entire range of preferences, here we consider a possibly different interval for each preference value. This makes most of the results in [5] inapplicable to our general case. Moreover, the formalism in [5] is unable to represent linguistic descriptions of degrees of preference (such as ”quite high” or ”low” or ”undesirable”) that may be mapped to preference intervals. Also, [5] looks only for necessarily optimal solutions, and uses preference elicitation, if needed, to find them. Here instead we consider many other notions of optimal solutions, with the aim of returning interesting solutions without resorting to preference elicitation.

Other papers consider preference intervals, such as the work in [2]. However, these lines of work focus on specific preference aggregation mechanisms (such as the Choquet integral) and of modelling issues without addressing the algorithmic questions related to finding optimal solutions according to different risk attitudes. We are instead interested in providing efficient algorithms to find optimal solutions according to different risk attitudes (called pessimistic and optimistic in the paper), besides the modelling concerns. For this reason, we model imprecise problems within an extension of soft constraints that allows us to exploit the existing machinery to solve soft constraint problems to obtain optimal solutions in the presence of preference intervals.

2 Background: soft constraints

A soft constraint [1] is just a classical constraint [3] where each instantiation of its variables has an associated value from a (totally or partially ordered) set. This set has two operations, which makes it similar to a semiring, and is called a c-semiring. More precisely, a c-semiring is a tuple $\langle A, +, \times, \mathbf{0}, \mathbf{1}\rangle$ such that: A is a set, called the carrier of the c-semiring, and $\mathbf{0}, \mathbf{1} \in A$; $+$ is commutative, associative, idempotent, $\mathbf{0}$ is its unit element, and $\mathbf{1}$ is its absorbing element; $\times$ is associative, commutative, distributes over $+$, $\mathbf{1}$ is its unit element and $\mathbf{0}$ is its absorbing element.

The relation $\leq_S$ over A such that $a \leq_S b$ iff $a + b = b$ is a partial order, over which $+$ and $\times$ are monotone, and where $\mathbf{0}$ is the minimum and $\mathbf{1}$ the maximum. Moreover, $\langle A, \leq_S\rangle$ is a lattice and, for all $a, b \in A$, $a + b = lub(a, b)$. If $\times$ is idempotent, then $\langle A, \leq_S\rangle$ is a distributive lattice and $\times$ is its glb. Informally, the relation $\leq_S$ gives us a way to preference values: when $a \leq_S b$, we say that *b is better than a*. Thus, $\mathbf{0}$ is the worst value and $\mathbf{1}$ is the best one.

A c-semiring $\langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$ is said to be *idempotent* iff the combination operator $\times$ is idempotent, i.e., for every $a \in A$, $a \times a = a$, while it is said to be *strictly monotonic* iff the combination operator $\times$ is strictly monotonic, i.e., for every $a, b \in A$, if $a < b$ then, for every $c \in A$, $a \times c < b \times c$. If a c-semiring is totally ordered, i.e., if $\leq_S$ is a total order, then the $+$ operation is just max with respect to $\leq_S$. If the c-semiring is also idempotent, then $\times$ is equal to min, and the c-semiring is of the kind used for fuzzy constraints (see below).

Given a c-semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$, a finite set D (the domain of the variables), and an ordered set of variables V, a soft constraint is a pair $\langle def, con \rangle$ where $con \subseteq V$ and $def : D^{|con|} \rightarrow A$. Therefore, a soft constraint specifies a set of variables (the ones in con), and assigns to each tuple of values of D of these variables an element of the c-semiring set A, which will be seen as its *preference*. A soft constraint satisfaction problem (SCSP) is just a set of soft constraints over a set of variables.

A classical CSP is just an SCSP where the chosen c-semiring is $S_{CSP} = \langle \{false, true\}, \vee, \wedge, false, true \rangle$. Fuzzy CSPs [7] are instead modeled by choosing the idempotent c-semiring $S_{FCSP} = \langle [0,1], max, min, 0, 1 \rangle$: we want to maximize the minimum preference. For weighted CSPs, the strictly monotonic c-semiring is $S_{WCSP} = \langle \Re^+, min, +, +\infty, 0 \rangle$: preferences are interpreted as costs from 0 to $+\infty$, and we want to minimize the sum of costs.

Given an assignment s to all the variables of an SCSP P, its preference, written $pref(P, s)$, is obtained by combining the preferences associated by each constraint to the subtuples of s referring to the variables of the constraint: $pref(P, s) = \Pi_{\langle idef, con \rangle \in C} def(s_{\downarrow con})$, where Π refers to the $\times$ operation of the c-semiring and $s_{\downarrow con}$ is the projection of tuple s on the variables in con. For example, in fuzzy CSPs, the preference of a complete assignment is the minimum preference given by the constraints. In weighted constraints, it is instead the sum of the costs given by the constraints. An optimal solution of an SCSP P is then a complete assignment s such that there is no other complete assignment s'' with $pref(P, s) <_S pref(P, s'')$.

3 Interval-valued constraint problems

Soft constraint problems require users to specify a preference value for each tuple in each constraint. Sometimes this is not reasonable, since a user may have a vague idea of what preferences to associate to some tuples. In [5] a first generalization allowed users to specify either a fixed preference (as in usual soft constraints) or the complete $[\mathbf{0}, \mathbf{1}]$ interval. Thus an assumption of complete ignorance was made when the user was not able to specify a fixed preference. Here we generalize further by allowing users to state any interval over the preference set.

Given a set of variables V with finite domain D, and a totally-ordered c-semiring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$, an **interval-valued constraint** is a pair $\langle int, con \rangle$ where $con \subseteq V$ is the scope of the constraint and int: $D^{|con|} \longrightarrow A \times A$ specifies an interval over A by giving its lower and upper bound. If $int(x) = (a, b)$, it must be $a \leq_S b$. In the following we will denote with $l(int(x))$ (resp., $u(int(x))$) the first (resp., second) component of $int(x)$, representing the lower and the upper bound of the preference interval. An

interval-valued constraint problem (IVCSP) is a 4-tuple $\langle V, D, C, S\rangle$, where C is a set of interval-valued constraints over S defined on the variables in V with domain D.

Figure 1 shows an IVCSP P defined over the fuzzy c-semiring $\langle [0,1], max, min, 0, 1\rangle$, that contains three variables X_1, X_2, and X_3, with domain $\{a, b\}$, and five constraints: a unary constraint on each variable, and two binary constraints on (x_1, x_2) and (x_2, x_3). The figure shows the definition of each constraint, giving an interval for each variable assignment.

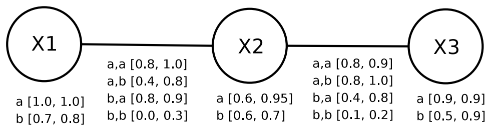

Fig. 1. An IVCSP.

In an IVCSP, a complete assignment of values to all the variables can be associated to an interval as well, by combining all the intervals of the relevant tuples in the constraints.

Given an IVCSP $P = \langle V, D, C, S\rangle$ and an assignment s to all its variables over D the **preference interval** of s in P is $[L(s), U(s)]$, where $L(s) = \Pi_{<int,con>\in C} l(int(s_{\downarrow con}))$ and $U(s) = \Pi_{<int,con>\in C} u(int(s_{\downarrow con}))$, and Π is the combination operator of the c-semiring S.

Figure 2 shows all the complete assignments of the IVCSP in Figure 1, together with their preference interval. The details of the computation of the preference intervals are shown for s_1.

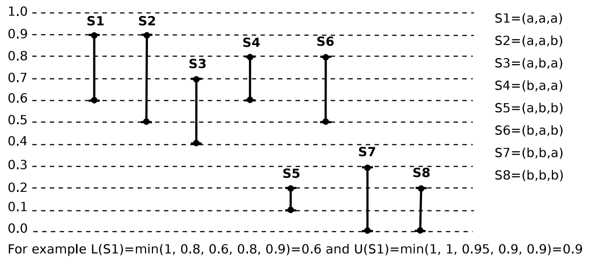

Fig. 2. Solutions of the IVCSP shown in Figure 1.

Once we have an IVCSP, it is useful to consider specific scenarios arising from choosing a preference value from each interval.

Given an IVCSP P, a **scenario** of P is an SCSP P' obtained from P as follows: given any constraint $c = \langle int, con\rangle$ of P, we insert in P' the constraint $c' = \langle def, con\rangle$, where $def(t) \in [l(int(t)), u(int(t))]$ for every tuple $t \in D^{|con|}$. We will denote with $S(P)$ the set of all possible scenarios of P. The **best scenario** ($BS(P)$) (resp., **worst scenario** ($WS(P)$)) of an IVCSP is obtained by replacing every interval with its upper (resp., lower) bound.

The preference interval of a complete assignment s of an IVCSP P contains all the preference values associated to s by the SCSPs in $S(P)$. The inverse does not necessarily hold. That is, there may be values in the preference interval of s which cannot be obtained in any scenario. However, if the c-semiring is idempotent, then there is a one-to-one correspondence. In the general case, we can however prove that, if the preference interval of s is $[a, b]$, there exist at least a scenario where s has preference a and there exist at least a scenario where s has preference b.

4 Interval-based optimality notions

Given an IVCSP, several notions of optimality can be given. Since an IVCSP presents a form of uncertainty, specified by the intervals, there are several ways to approach such an uncertainty. For example, one could be pessimistic or optimistic about the possible scenarios.

More precisely, given an IVCSP $P = \langle V, D, C, S \rangle$, an assignment s to the variables in V is said:

- **lower-optimal** iff, for every other complete assignment s', $L(s) \geq L(s')$.
  Thus, a lower-optimal assignment is better than or equal to all other assignments in the worst scenario. Therefore, lower-optimal assignments are useful in pessimistic approaches to uncertainty, since they outperform the other assignments in the worst case. We denote with $LO(P)$ the set of the lower optimal assignments of P. The IVCSP P of Figure 1 has $LO(P) = \{s_1, s_4\}$.
- **upper-optimal** iff, for every other complete assignemnt s', $U(s) \geq U(s')$.
  Thus, an upper-optimal assignment is better than or equal to all other assignments in the best scenario. Therefore, it is useful for optimistic approaches to uncertainty. We denote with $UO(P)$ the set of the upper optimal assignments of P. The IVCSP P of Figure 1 has $UO(P) = \{s_1, s_2\}$.
- **interval-optimal** iff, for every other complete assignment s', $L(s) \geq L(s')$ or $U(s) \geq U(s')$.
  In words, an interval-optimal assignment is an assignment with either a higher or equal lower bound, or a higher or equal upper bound, w.r.t. all other assignments. This means that it must be better than, or equal to, all other assignments in either the worst or the best scenario. We denote with $IO(P)$ the set of the interval optimal assignments of P. The IVCSP P of Figure 1 has $IO(P) = \{s_1, s_2, s_4\}$.
- **lower (resp., upper) lexicographically-optimal** iff, for every other assignment s', either $L(s) > L(s')$ (resp., $U(s) > U(s')$), or $L(s) = L(s')$ and $U(s) \geq U(s')$ (resp., $U(s) = U(s')$ and $L(s) \geq L(s')$).
  Thus, lower (resp., upper) lexicographically-optimal assignments are the best assignments for a pessimistic (resp., optimistic) approach, where ties are broken optimistically (resp., pessimistically). We denote with $LLO(P)$ (resp., $ULO(P)$) the set of the lower (resp., upper) lexicographically-optimal assignments of P. The IVCSP P of Figure 1 has $LLO(P) = ULO(P) = \{s_1\}$.
- **weakly-interval-dominant** iff, for every other assignment s', $L(s) \geq L(s')$ and $U(s) \geq U(s')$.

Thus, weakly-interval-dominant assignments are better than or equal to all others in both the worst and the best scenario. We denote with $WID(P)$ the set of the weakly interval dominant assignments of P. The IVCSP P of Figure 1 has $WID(P) = \{s_1\}$.
– **interval-dominant** iff, for every other assignment s', $L(s) \geq U(s')$.
Thus, interval-dominant assignments are better than or equal to all others in all scenarios. They are therefore very robust w.r.t. uncertainty. We denote with $ID(P)$ the set of the interval dominant assignments of P. The IVCSP P of Figure 1 has that $ID(P) = \emptyset$.

Given an IVCSP P, we have that:

– $(UO(P) \cup LO(P)) \subseteq IO(P)$;
– $UO(P) \cap LO(P) = WID(P)$;
– $ID(P) \subseteq WID(P)$;
– $LLO(P) \subseteq LO(P)$ and $ULO(P) \subseteq UO(P)$;
– $ID(P) \subseteq (LLO(P) \cap ULO(P)) = WID(P)$.
– $IO(P)$, $LO(P)$, $UO(P)$, $LLO(P)$, and $ULO(P)$ are never empty, while $WID(P)$ and $ID(P)$ may be empty.
– If $ID(P) \neq \emptyset$, either $ID(P)$ contains a single solution, or several solutions all with the lower bound equal to the upper bound and all equal to the same value.

5 Finding and testing optimal assignments

Lower and upper optimal assignments. To find a lower-optimal solution, it is enough to take the worst scenario and find an optimal solution. In fact, a lower-optimal solution is a complete assignment whose lower bound is greater than or equal to every other complete assignment, and thus it is a complete assignment that is better than or equal to all other assignments in the scenario obtained by replacing every interval with its lower bound, i.e., the worst scenario. Similarly, to find an upper-optimal solution, we can take the best scenario and find an optimal solution. Thus finding assignments in $LO(P)$ or $UO(P)$ is as complex as solving an SCSP.

To test if an assignment s in $LO(P)$ or in $UO(P)$, it is enough to solve the SCSP representing the worst or the best scenario and to check if the preference of the optimal solution coincides with $L(s)$ or $U(s)$.

Interval optimal assignments. To find an interval optimal assignment, it is sufficient to find a lower optimal solution or an upper optimal solution, since $(UO(P) \cup LO(P)) \subseteq IO(P)$, and both $UO(P)$ and $LO(P)$ cannot be empty. Thus finding assignments of $IO(P)$ is as complex as solving an SCSP.

To test if an assignment s is in $IO(P)$, if the combination operator is idempotent, we can find the solutions of the CSP obtained by putting together two CSPs obtained as follows: one is obtained by considering the worst scenario and by allowing only tuples with preference greater than $L(s)$, the other one is obtained by considering the best scenario and by allowing only tuples with preference greater than $U(s)$. Then, s is in $IO(P)$ if and only if this CSP has no solution.

If the combination operator is not idempotent, we can consider the SCSP with the same variables, domains, and constraint topology as P, and defined on the c-semiring $\langle (A \times A), (+', +'), (\times, \times), (\mathbf{0}, \mathbf{0}), (\mathbf{1}, \mathbf{1}) \rangle$, where $+'$ induces the strict ordering related to $+$. Then, s is optimal in this SCSP if and only if it is interval-optimal.

Lower and upper lexicographically optimal assignments. To find the lower-lexicographically optimal solutions of an IVCSP P defined on c-semiring S, let us consider the SCSP with the same variables, domains, and constraint topology as P, and with c-semiring $\langle A \times A, lex, (\times, \times), (\mathbf{0}, \mathbf{0}), (\mathbf{1}, \mathbf{1}) \rangle$, where lex induces the ordering $\succeq_{lex}$ defined as follows: for each $(a, a'), (b, b') \in (A \times A)$, $(a, a') \succeq_{lex} (b, b')$ iff $a > b$ or $a = b$ and $a' \geq_S b'$. In words, the first component is the most important, and the second one is used to break ties. To find the upper-lexicographically optimal solutions, it is sufficient to consider the same SCSP as defined above except for the ordering which considers the second component as the most important. Thus, finding assignments in $LLO(P)$ and $ULO(P)$ is as complex as solving an SCSP.

To test if a solution s is in $LLO(P)$, it is enough to find the preference pair, say $(p1, p2)$, of the optimal solution of the SCSP defined above and to check if $(L(s), U(s)) = (p1, p2)$. Similarly to test if a solution is in $ULO(P)$.

Weakly interval dominant solutions. We know that $WID(P) = LO(P) \cap UO(P)$. Thus a straightforward, but costly, way to find a solution in $WID(P)$ is to compute all the optimal solutions of the best and the worst scenario and then to check if there is a solution in the intersection of the two sets. However, if the c-semiring is idempotent, this is not necessary. In fact, it is sufficient to do the following:

- to find the optimal preference levels of the best and worst scenario, say l_{opt} and u_{opt};
- to consider the CSP obtained from the worst (resp., best) scenario by allowing in the constraints only tuples with preference greater than or equal to l_{opt} (resp., u_{opt}); we will denote such two CSPs by P_1 and P_2;
- to solve the CSP obtained obtained by putting together the constraints in P_1 and in P_2.

In this way, finding a weakly interval dominant solution amounts to solving two SCSPs and one CSP.

To test whether a solution s is in $WID(P)$, it is sufficient to find the preference of the optimal solution of the worst and best scenarios, say l_{opt} and u_{opt}, and to check if $L(s) = l_{opt}$ and $U(s) = u_{opt}$.

Interval dominant assignments. To find an assignment in $ID(P)$, if the c-semiring is idempotent, we can

- compute the optimal preference of the worst scenario, say l_{opt};
- consider the CSP obtained from the best scenario by allowing in the constraints only tuples with preference greater than l_{opt};

– check the number of solutions of this CSP: if it has no solution, then $ID(P) = LO(P)$, thus it is enough to find an optimal solution of $WS(P)$; if it has one solution, say s, and $L(s) = l_{opt}$, then this solution is the only one in $ID(P)$; otherwise $ID(P) = \emptyset$.

Thus, we need to solve an SCSP and then one CSP.

To test if an assignment s is in $ID(P)$, we can consider two cases. If $L(s) \neq U(s)$, then, if the c-semiring is idempotent, we can take the best scenario and consider the CSP obtained by allowing only tuples with preference greater than $L(s)$. This CSP has only s as solution if and only if s is in $ID(P)$. If instead $L(s) = U(s)$, we can check if s is an optimal solution of the best scenario.

6 Necessary and possible optimality

We will now consider more general notions of optimality, that are applicable to any setting where the lack of precision gives rise to several possible scenarios. We will then show how to exploit the interval-based notions of optimality introduced above to characterize these general notions.

Necessarily optimal solutions

Given an IVCSP $P = \langle V, D, C, S \rangle$, an assignment s to the variables in V is **necessarily optimal** if it optimal in all scenarios. Given an IVCSP P, the set of its necessarily optimal solution will be denoted by $NO(P)$.

Necessarily optimal solutions are very attractive, since they are very robust: they are optimal independently of the uncertainty of the problem. Unfortunately, the set $NO(P)$ may be empty. More precisely, given an IVCSP P, we have that:

– $ID(P) \subseteq NO(P) \subseteq WID(P)$;
– if $ID(P) \neq \emptyset$, then $ID(P) = NO(P)$.

It is easy to see that an interval-dominant solution is a necessarily optimal solution. Moreover, if $ID(P) \neq \emptyset$, then the converse holds as well. In fact, every solution not in $ID(P)$ has at most preference equal to the lower bound of those in $ID(P)$ in all scenarios. If instead $ID(P) = \emptyset$, then it may be $NO(P) \neq \emptyset$. In our running example of Figure 1, we have $ID(P) = NO(P) = \emptyset$. However, consider the IVCSP P over the fuzzy c-semiring with three variables X_1, X_2, and X_3, with domain $\{a, b\}$ and with two constraints c_1 and c_2 such that $c_1 = \langle int_1, con_1 \rangle$ with $con_1 = \{X_1, X_2\}$, $int_1(a, a) = (0.4, 0.7)$, and $(0, 0)$ otherwise, while $c_2 = \langle int_2, con_2 \rangle$ with $con_1 = \{X_2, X_3\}$, $int_2(a, a) = (0.8, 1.0)$, $int_2(a, b) = (0.9, 1.0)$, and $(0, 0)$ otherwise. We have $ID(P) = \emptyset$ while $NO(P) = \{(a, a, a), (a, a, b)\}$.

Also, $NO(P) \subseteq LO(P)$, since, if s is not lower-optimal, then in some (for sure the worst) scenario it is not optimal. Similarly, a necessarily optimal solution must be optimal also in the best scenario and thus $NO(P) \subseteq UO(P)$. This allows us to conclude that $NO(P) \subseteq LO(P) \cap UO(P) = WID(P)$.

To find a necessarily optimal solution, we can start by trying to find an interval-dominant assignment, since, if $ID(P) \neq \emptyset$, then $ID(P) = NO(P)$. To this purpose, we can use the procedure described in Section 5. If instead $ID(P) = \emptyset$, then, since $NO(P) \subseteq WID(P)$, we may check if $WID(P)$ is empty, since in such a case also $NO(P)$ is empty. If the previous steps do not allow us to conclude, we can compute set $WID(P)$ and, for each solution in such a set, to test if it is necessarily optimal (see below).

To test if a solution s is necessarily optimal, we can check if s is an optimal solution of an SCSP with the same c-semiring, variables, domains, and constraint topology as P, where we replace the interval of every tuple associated with s with its lower bound and the interval of all the other tuples with their upper bound. If s is not an optimal solution of this SCSP, then s is not necessarily optimal. If the c-semiring is strictly monotonic, this is a necessary and sufficient condition. However, this is not so when the combination operator is idempotent.

Guaranteeing a level of preference in all scenarios

An assignment s is **necessarily of at least preference** α if, for all scenarios, its preference is at least α. The set of all solutions of an IVCSP P with this feature will be denoted by $Nec(P, \alpha)$. In our running example, if we consider $\alpha = 0.5$, we have $Nec(P, 0.5) = \{s_1, s_2, s_4, s_6\}$.

If we are happy with a preference level of α, elements in $Nec(P, \alpha)$ are what we want, since they guarantee such a preference level independently of the uncertainty of the problem.

We can observe that $s \in Nec(P, \alpha)$ if and only if $\alpha \leq L(s)$. Thus, testing whether a solution s is in $Nec(P, \alpha)$ amounts at checking this condition, which can be done in linear time.

If the c-semiring is idempotent, the elements of $Nec(P, \alpha)$ are the solutions of the CSP obtained from $WS(P)$ by allowing only the tuples with preference at least α. Therefore, to find a solution in $Nec(P, \alpha)$, it is sufficient to solve a CSP.

If the combination operator is not idempotent, we can solve the worst scenario and compute the preference level of an optimal solution, say l_{opt}. Then, $s \in Nec(P, \alpha)$ if and only if $\alpha \leq l_{opt}$. Thus, in the general case, we must solve an SCSP.

Let α_* be the maximum α such that there exists a solution in $Nec(P, \alpha)$. In our running example, we have $\alpha_* = 0.6$, and $Nec(P, 0.6) = \{s_1, s_4\}$.

It is possible to show that the elements in $Nec(P, \alpha_*)$ are the solutions of $LO(P)$. This implies also that $NO(P) \subseteq Nec(P, \alpha_*)$. Thus, to find a solution in $Nec(P, \alpha_*)$, it is sufficient to find an optimal solution of the worst scenario of P.

Possibly Optimal Solutions

An assignment s is **possibly optimal** if it optimal in some scenario. The set of possibly optimal solutions of P will be denoted by $PO(P)$. In our running example, we have $PO(P) = \{s_1, s_2, s_3, s_4, s_6\}$.

To find a solution in $PO(P)$, we can observe that $LO(P)$, $UO(P)$, $LLO(P)$, or $ULO(P)$ are all contained in $PO(P)$, and are never empty. Thus we may find an element in any of such sets.

To test if a solution s is in $PO(P)$, if the combination operator is strictly monotonic, s is in $PO(P)$ if and only if s wins in the scenario where all its unknowns are set to the upper bound and the other unknowns to the lower bound. If instead the combination operator is idempotent, we have to consider the worst scenario and compute the preference level of its optimal solutions, say l_{opt}. Then s is in $PO(P)$ if and only if s wins in the worst scenario or in the scenario obtained by the worst one by raising all the unknowns of s to the level l_{opt} (if this is not possible, s is not in $PO(P)$). Thus, finding or testing possible optimality requires solving an SCSP.

Guaranteeing a level of preference in at least one scenario

An assignment s is **possibly of at least preference** α if there exists a scenario such that the preference of s in that scenario is at least α. The set of all solutions of an IVCSP P with this feature will be denoted by $Pos(P, \alpha)$. In our running example, if we take $\alpha = 0.3$, we have $Pos(P, 0.3) = \{s_1, s_2, s_3, s_4, s_6, s_7\}$.

An assignment s is in $Pos(P, \alpha)$ if and only if $\alpha \leq U(s)$. Thus, to test whether a solution is in $Pos(P, \alpha)$, it is enough to check this condition, that takes linear time.

To find an assignment in $Pos(P, \alpha)$, if the c-semiring is idempotent, we can consider the CSP obtained from the best scenario by allowing only the tuples with preference at least α. Therefore, it is sufficient to solve a particular CSP.

Let α^* be the maximum α such that $Pos(P, \alpha)$ is not empty. In our running example, we have $\alpha^* = 0.9$ and $Pos(P, 0.9) = \{s_1, s_2\}$.

It is possible to show that $Pos(P, \alpha^*) = UO(P)$. Thus $NO(P) \subseteq Pos(P, \alpha^*) \subseteq PO(P)$. Thus, to find a solution in $Pos(P, \alpha^*)$, it is sufficient to find an optimal solution of the best scenario of P, and thus to solve an SCSP.

7 Necessary and possible dominance

Besides finding or testing for optimality, it may sometimes be useful to know if a solution dominates another one.

Given a scenario S, we say that a solution s strictly dominates (resp., dominates) a solution s' if and only if $pref(S, s) > pref(S, s')$ (resp., $pref(S, s) \geq pref(S, s')$) in the ordering of the considered c-semiring. Also, a solution s **possibly dominates** a solution s' if and only if there is at least one scenario where s strictly dominates s'. Instead, a solution s **necessarily dominates** a solution s' if and only if, in all scenarios, s dominates s', and there is at least one scenario where s strictly dominates s'.

In our running example, s_1 necessarily dominates s_8. In the best scenario, s_2 strictly dominated s_4, while in the worst scenario s_4 strictly dominates s_2. Thus s_2 possibly dominates s_4, and viceversa.

The maximal elements in the partial ordering given by the necessary dominance are the possibly optimal solutions. Also, the "possibly dominates" ordering may have cycles (see the cycle between s_2 and s_4 in our example), thus it may have no undominated elements. However, if it has undominated elements, they are the necessarily optimal solutions.

To test if s possibly dominates s' we can set each interval associated with s but not with s' to its upper bound; let λ be the combination of these values. Then we set each interval associated with s' but not with s to its lower bound; let μ be the combination of these values. Finally, we compare the preference values of s and s', by testing if the condition $\lambda \times u_1 \times \cdots \times u_k > \mu \times u_1 \times \cdots \times u_k$ holds for any selections of values $u_1, \ldots, u_k$ in the intervals of both s and s'.

If we have strict monotonicity, testing this condition amounts to testing if $\lambda > \mu$. If we have idempotence, we can replace each u_i with its upper bound, and then test the condition.

To test if s necessarily dominates s', we first check if s' possibly dominates s. If so, we can conclude negatively. Otherwise, we check if s possibly dominates s'. If so, we conclude positively, otherwise negatively.

8 Final considerations and future work

Given an IVCSP P, the solutions in NO(P) are certainly the most attractive, since they are the best one in every scenario. However, if there is none, we can pass to consider the solutions in $Nec(P, \alpha_*)$: they may be suboptimal, but they guarantee a preference level α_* in all scenarios. If α_* is too low, and we feel optimistic, we can consider the solutions in $Pos(P, \alpha^*)$: they guarantee it is possible to reach a higher level of preference, but not in all scenarios.

If we allowed users to associate to each partial assignment in the constraints not just a single interval, but a disjoint set of intervals, this would reduce the uncertainty of the problem. However, all the optimality notions would give the same set of optimals. The main reason for this result is the assumption of working with monotonic combination operators. This means that a level of precision greater than a single interval does not add any useful information when looking for an optimal solution.

This paper considers only totally ordered preferences. IVCSPs can be defined also for a partially ordered setting. We plan to extend the analysis of the optimality notions also to this more general setting.

We plan also to test empirically our algorithms over classes of IVSCPs over the fuzzy c-semiring. We are currently performing these experiments and we plan to add empirical results in the camera-ready version of the paper.

Acknowledgements

This work has been partially supported by Italian MIUR PRIN project “Constraints and Preferences” (n. 2005015491).

References

1. S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint solving and optimization. *JACM*, 44(2):201–236, mar 1997.

2. M. Ceberio and F. Modave. An interval-valued, 2-additive choquet integral for multi-criteria decision making. In *IPMU'04*, 2004.
3. R. Dechter. *Constraint processing*. Morgan Kaufmann, 2003.
4. B. Faltings and S. Macho-Gonzalez. Open constraint programming. *AI Journal*, 161(1-2):181–208, 2005.
5. M. Gelain, M. S. Pini, F. Rossi, and K. B. Venable. Dealing with incomplete preferences in soft constraint problems. In *Proc. CP'07*, volume 4741 of *LNCS*, pages 286–300. Springer, 2007.
6. S. Macho González, C. Ansótegui, and P. Meseguer. On the relation among open, interactive and dynamic CSP. In *The Fifth Workshop on Modelling and Solving Problems with Constraints (IJCAI'05)*, 2005.
7. P. Meseguer, F. Rossi, and T. Schiex. Soft constraints. In F. Rossi, P Van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*. Elsevier, 2006.
8. F. Rossi, P Van Beek, and T. Walsh, editors. *Handbook of Constraint Programming*. Elsevier, 2006.
9. N. Wilson, D. Grimes, and E. C. Freuder. A cost-based model and algorithms for interleaving solving and elicitation of csps. In *Proc. CP'07*, volume 4741 of *LNCS*, pages 666–680. Springer, 2007.
10. N. Yorke-Smith and C. Gervet. Certainty closure: A framework for reliable constraint reasoning with uncertainty. In *CP'03*, volume 2833 of *LNCS*, pages 769–783. Springer, 2003.