Constrained CP-nets

Steve Prestwich[1], Francesca Rossi[2], Kristen Brent Venable[2], Toby Walsh[1]

1: Cork Constraint Computation Centre, University College Cork, Ireland. Email: s.prestwich@cs.ucc.ie, tw@4c.ucc.ie
2: Department of Pure and Applied Mathematics, University of Padova, Italy. Email: {frossi,kvenable}@math.unipd.it.

Abstract. We present a novel approach to deal with preferences expressed as a mixture of hard constraints, soft constraints, and CP-nets. We construct a set of hard constraints whose solutions are the optimal solutions of the set of preferences, where optimal is defined differently w.r.t. other approaches [2, 7]. The new definition of optimality introduced in this paper, allows us to avoid dominance testing (is one outcome better than another?) which is a very expensive operation often used when finding optimal solutions or testing optimality, while being reasonable and intuitive. We also show how hard constraints can sometimes eliminate cycles in the preference ordering. Finally, we extend this approach to deal with the preferences of multiple agents. This simple and elegant technique permits conventional constraint and SAT solvers to solve problems involving both preferences and constraints.

1 Introduction

Preferences and constraints are ubiquitous in real-life scenarios. We often have hard constraints (as “I must be at the office before 9am”) as well as some preferences (as “I would prefer to be at the office around 8:30am” or “I would prefer to go to work by bicycle rather than by car”). Whilst formalisms to represent and reason about hard constraints are relatively stable, having been studied for over 20 years [5], preferences have not received as much attention until more recent times. Among the many existing approaches to represent preferences, we will consider CP-nets [6, 3], which is a qualitative approach where preferences are given by ordering outcomes (as in “I like meat over fish”) and soft constraints [1], which is a quantitative approach where preferences are given to each statement in absolute terms (as in “My preference for fish is 0.5 and for meat is 0.9”).

It is easy to reason with hard and soft constraints at the same time, since hard constraints are just a special case of soft constraints. Much less is understood about reasoning with CP-nets and (hard or soft) constraints. One of our aims is to tackle this problem. We will define a structure called a constrained CP-net. This is just a CP-net plus a set of hard constraints. We will give a semantics for this structure (based on the original flipping semantics of CP-nets) which gives priority to the hard constraints. We will show how to obtain the optimal solutions of such a constrained CP-net by compiling the preferences into a set of hard constraints whose solutions are exactly the optimal solutions of the constrained CP-net. This allows us to test optimality in linear time, even if the CP-net is not acyclic. Finding an optimal solution of a constrained CP net is NP-hard [1] (as it is in CP-nets and in hard constraints).

Prior to this work, to test optimality of a CP-net plus a set of constraints, we had to find all solutions of the constraints (which is NP-hard) and then test if any of them dominate the solution in question [2]. Unfortunately dominance testing is not known to be in NP even for acyclic CP-nets, as we may have to explore chains of worsening flips that are exponentially long. By comparison, we do not need to perform dominance testing in our approach. Our semantics is also useful when the CP-net defines a preference ordering that contains cycles, since the hard constraints can eliminate these cycles. Lastly, since we compile preferences down into hard constraints, we can use standard constraint solving algorithms (or SAT algorithms if the variables have just two values) to reason about preferences and constraints, rather than develop special purpose algorithms for constrained CP-nets (as in [2]). We also consider when a CP-net is paired with a set of soft constraints, and when there are several CP-nets, and sets of hard or soft constraints. In all these cases, optimal solutions can be found by solving a set of hard or soft constraints, avoiding dominance testing.

2 Background

2.1 CP-nets

In many applications, it is natural to express preferences via generic qualitative (usually partial) preference relations over variable assignments. For example, it is often more intuitive to say "I prefer red wine to white wine", rather than "Red wine has preference 0.7 and white wine has preference 0.4". The former statement provides less information, but does not require careful selection of preference values. Moreover, we often wish to represent conditional preferences, as in "If it is meat, then I prefer red wine to white". Qualitative and conditional preference statements are thus useful components of many applications.

CP-nets [6, 3] are a graphical model for compactly representing conditional and qualitative preference relations. They exploit conditional preferential independence by structuring an agent's preferences under the *ceteris paribus* assumption. Informally, CP-nets are sets of *conditional ceteris paribus (CP)* preference 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. Many users' preferences appear to be of this type.

CP-nets bear some similarity to Bayesian networks. Both utilize directed graphs where each node stands for a domain variable, and assume a set of features $F = \{X_1, \ldots, X_n\}$ with finite domains $\mathcal{D}(X_1), \ldots, \mathcal{D}(X_n)$. For each feature X_i, each user specifies a set of *parent* features $Pa(X_i)$ that can affect her preferences over the values of X_i. This defines a dependency graph in which each node X_i has $Pa(X_i)$ as

[1] More precisely it is in FNP-hard, since it is not a decision problem. In the rest of the paper we will write NP meaning FNP when not related to decision problems.

its immediate predecessors. Given this structural information, the user explicitly specifies her preference over the values of X_i for *each complete outcome* on $Pa(X_i)$. This preference is assumed to take the form of total or partial order over $\mathcal{D}(X)$ [6, 3].

For example, consider a CP-net whose features are A, B, C, and D, with binary domains containing f and $\overline{f}$ if F is the name of the feature, and with the preference statements as follows: $a \succ \overline{a}$, $b \succ \overline{b}$, $(a \wedge b) \vee (\overline{a} \wedge \overline{b}) : c \succ \overline{c}$, $(a \wedge \overline{b}) \vee (\overline{a} \wedge b) : \overline{c} \succ c$, $c : d \succ \overline{d}$, $\overline{c} : \overline{d} \succ d$. Here, statement $a \succ \overline{a}$ represents the unconditional preference for $A = a$ over $A = \overline{a}$, while statement $c : d \succ \overline{d}$ states that $D = d$ is preferred to $D = \overline{d}$, given that $C = c$.

The semantics of CP-nets depends on the notion of a worsening flip. A worsening flip is a change in the value of a variable to a value which is less preferred by the CP statement for that variable. For example, in the CP-net above, passing from $abcd$ to $ab\overline{c}d$ is a worsening flip since c is better than $\overline{c}$ given a and b. We say that one outcome α is better than another outcome β (written $\alpha \succ \beta$) iff there is a chain of worsening flips from α to β. This definition induces a strict partial order over the outcomes. In general, there may be many optimal outcomes. However, in acyclic CP-nets (that is, CP-nets with an acyclic dependency graph), there is only one.

Several types of queries can be asked about CP-nets. First, given a CP-net, what are the optimal outcomes? For acyclic CP-nets, such a query is answerable in linear time [6, 3]: we forward sweep through the CP-net, starting with the unconditional variables, following the arrows in the dependency graph and assigning at each step the most preferred value in the preference table. For instance, in the CP-net above, we would choose $A = a$ and $B = b$, then $C = c$ and then $D = d$. The optimal outcome is therefore $abcd$. The same complexity also holds for testing whether an outcome is optimal since an acyclic CP-net has only one optimal outcome. We can find this optimal outcome (in linear time) and then compare it to the given one (again in linear time). On the other hand, for cyclic CP-nets, both finding and testing optimal outcomes is NP-hard.

The second type of query is a dominance query. Given two outcomes, is one better than the other? Unfortunately, this query is NP-hard even for acyclic CP-nets. Whilst tractable special cases exist, there are also acyclic CP-nets in which there are exponentially long chains of worsening flips between two outcomes. In the CP-net of the example, $\overline{a}b\overline{c}\overline{d}$ is worse than $abcd$.

2.2 Soft and hard constraints

There are several formalisms for describing *soft constraints*. We use the c-semi-ring formalism [1] as this generalizes most of the others. In brief, a soft constraint associates each instantiation of its variables with a value from a partially ordered set. We also supply operations for combining ($\times$) and comparing (+) values. A semi-ring is a tuple $\langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$ such that: A is a set and $\mathbf{0}, \mathbf{1} \in A$; $+$ is commutative, associative and $\mathbf{0}$ is its unit element; $\times$ is associative, distributes over $+$, $\mathbf{1}$ is its unit element and $\mathbf{0}$ is its absorbing element. A *c-semi-ring* is a semi-ring $\langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$ in which $+$ is idempotent, $\mathbf{1}$ is its absorbing element and $\times$ is commutative.

Let us consider the relation $\leq$ over A such that $a \leq b$ iff $a + b = b$. Then $\leq$ is a partial order, $+$ and $\times$ are monotone on $\leq$, $\mathbf{0}$ is its minimum and $\mathbf{1}$ its maximum, $\langle A, \leq \rangle$ is a complete lattice and, for all $a, b \in A$, $a + b = lub(a, b)$. Moreover, if $\times$ is idempotent: $+$ distributes over $\times$; $\langle A, \leq \rangle$ is a complete distributive lattice and $\times$ its glb. Informally, the relation $\leq$ compares semi-ring values and constraints. When $a \leq b$, we say that *b is better than a*. Given a semi-ring $S = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$, a finite set D (variable domains) and an ordered set of variables V, a *soft constraint* is a pair $\langle def, con \rangle$ where $con \subseteq V$ and $def : D^{|con|} \rightarrow A$. A constraint specifies a set of variables, and assigns to each tuple of values of these variables an element of the semi-ring.

A *soft constraint satisfaction problem* (SCSP) is given by a set of soft constraints. A solution to an SCSP is a complete assignment to its variables, and the preference value associated with a solution is obtained by multiplying the preference values of the projections of the solution to each constraint. A solution is better than another if its preference value is higher in the partial order of the semi-ring. Finding an optimal solution for an SCSP is NP-hard. On the other hand, given two solutions, checking whether one is preferable to another is straightforward: compute the semi-ring values of the two solutions and compare the resulting two values.

Each semiring identifies a class of soft constraints. For example, fuzzy CSPs are SCSPs over the semiring $S_{FCSP} = \langle [0, 1], max, min, 0, 1 \rangle$. This means that preferences are over [0,1], and that we want to maximize the minimum preference over all the constraints. Another example is given by weighted CSPs, which are just SCSPs over the semiring $S_{weight} = \langle \mathcal{R}, min, +, 0, +\infty \rangle$, which means that preferences (better called costs here) are real numbers, and that we want to minimize their sum.

Note that hard constraints are just a special class of soft constraints: those over the semiring $S_{CSP} = \langle \{false, true\}, \vee, \wedge, false, true \rangle$, which means that there are just two preferences ($false$ and $true$), that the preference of a solution is the logical *and* of the preferences of their subtuples in the constraints, and that true is better than false (ordering induced by the logical *or* operation $\vee$).

3 Constrained CP-nets

We now define a structure which is a CP-net plus a set of hard constraints. In later sections we will relax this concept by allowing soft constraints rather than hard constraints.

Definition 1 (constrained CP-net). *A Constrained CP-net is a CP-net plus some constraints on subsets of its variables. We will thus write a constrained CP-net as a pair $\langle N, C \rangle$, where N is a set of conditional preference statements defining a CP-net and C is a set of constraints.*

The hard constraints can be expressed by generic relations on partial assignments or, in the case of binary features, by a set of Boolean clauses. As with CP-nets, the basis of the semantics of constrained CP-nets is the preference ordering, $\succ$, which is defined by means of the notion of a worsening flip. A worsening flip is defined very similarly to how it is defined in a regular (unconstrained) CP-net.

Definition 2 ($O1 \succ O_2$). *Given a constrained CP-net $\langle N, C \rangle$, outcome O_1 is* **better** *than outcome O_2 (written $O_1 \succ O_2$) iff there is a chain of flips from O_1 to O_2, where each flip is worsening for N and each outcome in the chain satisfies C.*

The only difference with the semantics of (unconstrained) CP-nets is that we now restrict ourselves to chains of *feasible* outcomes. As we show shortly, this simple change has some very beneficial effects. First, we observe that the $\succ$ relation remains a strict partial ordering as it was for CP-nets [6, 3]. Second, it is easy to see that checking if an outcome is optimal is linear (we merely need to check it is feasible and any flip is worsening). Third, if a set of hard constraints are satisfiable and a CP-net is acyclic, then the constrained CP-net formed from putting the hard constraints and the CP-net together must have at least one feasible and undominated outcome. In other words, adding constraints to an acyclic CP-net does not eliminate all the optimal outcomes (unless it eliminates all outcomes). Compare this to [2] where adding constraints to a CP-net may make all the undominated outcomes infeasible while not allowing any new outcomes to be optimal. For example, if we have $O_1 \succ O_2 \succ O_3$ in a CP-net, and the hard constraints make O_1 infeasible, then according to our semantics O_2 is optimal, while according to the semantics in [2] no feasible outcome is optimal.

Theorem 1. *A constrained and acyclic CP-net either has no feasible outcomes or has at least one feasible and undominated outcome.*

Proof. Take an acyclic constrained CP-net $\langle N, C\rangle$. N induces a preference ordering that contains no cycles and has exactly one most preferred outcome, say O. If O is feasible, it is optimal for $\langle N, C\rangle$. If O is infeasible, we move down the preference ordering until at some point we hit the first feasible outcome. This is optimal for $\langle N, C\rangle$. $\square$

4 An example

We will illustrate constrained CP-nets by means of a simple example. This example illustrates that adding constraints can eliminate cycles in the preference ordering defined by the CP-net. This is not true for the semantics of [6], where adding hard constraints cannot break cycles.

Suppose I want to fly to Australia. I can fly with British Airways (BA) or Singapore Airlines, and I can choose between business or economy. If I fly Singapore, then I prefer to save money and fly economy rather than business as there is good leg room even in economy. However, if I fly BA, I prefer business to economy as there is insufficient leg room in their economy cabin. If I fly business, then I prefer Singapore to BA as Singapore's inflight service is much better. Finally, if I have to fly economy, then I prefer BA to Singapore as I collect BA's airmiles. If we use a for British Airways, $\overline{a}$ for Singapore Airlines, b for business, and $\overline{b}$ for economy then we have: $a : b \succ \overline{b}$, $\overline{a} : \overline{b} \succ b$, $b : \overline{a} \succ a$, and $\overline{b} : a \succ \overline{a}$.

This CP-net has chains of worsening flips which contain cycles. For instance, $ab \succ a\overline{b} \succ \overline{a}\overline{b} \succ \overline{a}b \succ ab$. That is, I prefer to fly BA in business (ab) than BA in economy ($a\overline{b}$) for the leg room, which I prefer to Singapore in economy ($\overline{a}\overline{b}$) for the airmiles, which I prefer to Singapore in business ($\overline{a}b$) to save money, which I prefer to BA in business (ab) for the inflight service. According to the semantics of CP-nets, none of the outcomes in the cycle is optimal, since there is always another outcome which is better.

Suppose now that my travel budget is limited, and that whilst Singapore offers no discounts on their business fares, I have enough airmiles with BA to upgrade from economy. I therefore add the constraint that, whilst BA in business is feasible, Singapore in business is not. That is, $\overline{a}b$ is not feasible. In this constrained CP-net, according to our new semantics, there is no cycle of worsening flips as the hard constraints break the chain by making $\overline{a}b$ infeasible. There is one feasible outcome that is undominated, that is, ab. I fly BA in business using my airmiles to get the upgrade. I am certainly happy with this outcome.

Notice that the notion of optimality introduced in this paper gives priority to the constraints with respect to the CP-net. In fact, an outcome is optimal if it is feasible and it is undominated in the constrained CP-net ordering. Therefore, while it is not possible for an infeasible outcome to be optimal, it is possible for an outcome which is dominated in the CP-net ordering to be optimal in the constrained CP-net.

5 Finding optimal outcomes

We now show how to map any constrained CP-net onto an equivalent constraint satisfaction problem containing just hard constraints, such that the solutions of these hard constraints corresponds to the optimal outcomes of the constrained CP-net. The basic idea is that each conditional preference statement of the given CP-net maps onto a conditional hard constraint. For simplicity, we will first describe the construction for Boolean variables. In the next section, we will pass to the more general case of variables with more than two elements in their domain.

Consider a constrained CP-net $\langle N, C\rangle$. Since we are dealing with Boolean variables, the constraints in C can be seen as a set of Boolean clauses, which we will assume are in a conjunctive normal form. We now define the **optimality constraints** for $\langle N, C\rangle$, written as $N \oplus_b C$ where the subscript b stands for Boolean variables, as $C \cup \{opt_C(p) \mid p \in N\}$. The function *opt* maps the conditional preference statement $\varphi : a \succ \overline{a}$ onto the hard constraint:

$$(\varphi \wedge \bigwedge_{\psi \in C, \overline{a} \in \psi} \psi|_{a=true}) \rightarrow a$$

where $\psi|_{a=true}$ is the clause ψ where we have deleted $\overline{a}$. The purpose of $\psi|_{a=true}$ is to model what has to be true so that we can safely assign a to true, its more preferred value.

To return to our flying example, the hard constraints forbid b and $\overline{a}$ to be simultaneously true. This can be written as the clause $a \vee \overline{b}$. Hence, we have the constrained CP-net $\langle N, C\rangle$ where $N = \{a : b \succ \overline{b}, \overline{a} : \overline{b} \succ b, \overline{b} : a \succ \overline{a}, b : \overline{a} \succ a\}$ and $C = \{a \vee \overline{b}\}$. The optimality constraints corresponding to the given constrained CP-net are therefore $a \vee \overline{b}$ plus the following clauses:

$$(a \wedge a) \rightarrow b \qquad (b \wedge \overline{b}) \rightarrow \overline{a}$$
$$\overline{a} \rightarrow \overline{b} \qquad \overline{b} \rightarrow a$$

The only satisfying assignment for these constraints is ab. This is also the only optimal outcome in the constrained CP-net. In general, the satisfying assignments of the optimality constraints are exactly the feasible and undominated outcomes of the constrained CP-net.

Theorem 2. *Given a constrained CP-net $\langle N, C\rangle$ over Boolean variables, an outcome is optimal for $\langle N, C\rangle$ iff it is a satisfying assignment of the optimality constraints $N \oplus_b C$.*

Proof. ($\Rightarrow$) Consider any outcome O that is optimal. Suppose that O does not satisfy $N \oplus_b C$. Clearly O satisfies C, since to be optimal it must be feasible (and undominated). Therefore O must not satisfy some $opt_C(p)$ where $p \in N$. The only way an implication is not satisfied is when the hypothesis is *true* and the conclusion is *false*. That is, $O \vdash \varphi$, $O \vdash \psi|_{a=true}$ and $O \vdash \overline{a}$ where $p = \varphi : a \succ \overline{a}$. In this situation, flipping from $\overline{a}$ to a would give us a new outcome O' such that $O' \vdash a$ and this would be an improvement according to p. However, by doing so, we have to make sure that the clauses in C containing $\overline{a}$ may now not be satisfied, since now $\overline{a}$ is false. However, we also have that $O \vdash \psi|_{a=true}$, meaning that if $\overline{a}$ is false these clauses are satisfied. Hence, there is an improving flip to another feasible outcome O'. But O was supposed to be undominated. Thus it is not possible that O does not satisfy $N \oplus_b C$. Therefore O satisfies all $opt_C(p)$ where $p \in N$. Since it is also feasible, O is a satisfying assignment of $N \oplus_b C$.

($\Leftarrow$) Consider any assignment O which satisfies $N \oplus_b C$. Clearly it is feasible as $N \oplus_b C$ includes C. Suppose we perform an improving flip in O. Without loss of generality, consider the improving flip from $\overline{a}$ to a. There are two cases. Suppose that this new outcome is not feasible. Then this new outcome does not dominate the old one in our semantics. Thus O is optimal. Suppose, on the other hand, that this new outcome is feasible. If this is an improving flip, there must exist a statement $\varphi : a \succ \overline{a}$ in N such that $O \vdash \varphi$. By assumption, O is a satisfying assignment of $N \oplus_b C$. Therefore $O \vdash opt(\varphi : a \succ \overline{a})$. Since $O \vdash \varphi$ and $O \vdash \overline{a}$, and *true* is not allowed to imply *false*, at least one $\psi|_{a=true}$ is not implied by O where $\psi \in C$ and $\overline{a} \in \psi$. However, as the new outcome is feasible, ψ has to be satisfied independent of how we set a. Hence, $O \vdash \psi|_{a=true}$. As this is a contradiction, this cannot be an improving flip. The satisfying assignment is therefore feasible and undominated. □

It immediately follows that we can test for feasible and undominated outcomes in linear time in the size of $\langle N, C\rangle$: we just need to test the satisfiability of the optimality constraints, which are as many as the constraints in C and the conditional statements in N. Notice that this construction works also for regular CP-nets without any hard constraints. In this case, the optimality constraints are of the form $\varphi \rightarrow a$ for each conditional preference statement $\varphi : a \succ \overline{a}$.

It was already known that optimality testing in acyclic CP-nets is linear [6]. However, our construction also works with cyclic CP-nets. Therefore optimality testing for cyclic CP-nets has now become an easy problem, even if the CP-nets are not constrained. On the other hand, determining if a constrained CP-net has any feasible and undominated outcomes is NP-complete (to show completeness, we map any SAT problem directly onto a constrained CP-net with no preferences). Notice that this holds also for acyclic CP-nets, and finding an optimal outcome in an acyclic constrained CP-net is NP-hard.

6 Non-Boolean variables

The construction in the previous section can be extended to handle variables whose domain contains more than 2 values. Notice that in this case the constraints are no longer clauses but regular hard constraints over a set of variables with a certain domain. Given a constrained CP-net $\langle N, C\rangle$, consider any conditional preference statement p for feature x in N of the form $\varphi : a_1 \succ a_2 \succ a_3$. For simplicity, we consider just 3 values. However, all the constructions and arguments extend easily to more values. The optimality constraints corresponding to this preference statement (let us call them $opt_C(p)$) are:

$$\varphi \wedge (C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}} \rightarrow x = a_1$$

$$\varphi \wedge (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}} \rightarrow x = a_1 \vee x = a_2$$

where C_x is the subset of constraints in C which involve variable x [2], and $\downarrow X$ projects onto the variables in X. The optimality constraints corresponding to $\langle N, C\rangle$ are again $N \oplus C = C \cup \{opt_C(p) \mid p \in N\}$. We can again show that this construction gives a new problem whose solutions are all the optimal outcomes of the constrained CP-net.

Theorem 3. *Given a constrained CP-net $\langle N, C\rangle$, an outcome is optimal for $\langle N, C\rangle$ iff it is a satisfying assignment of the optimality constraints $N \oplus C$.*

Proof. ($\Rightarrow$) Consider any outcome O that is optimal. Suppose that O does not satisfy $N \oplus C$. Clearly O satisfies C, since to be optimal it must be feasible (and undominated). Therefore O must not satisfy some $opt_C(p)$ where p preference statement in N. Without loss of generality, let us consider the optimality constraints $\varphi \wedge (C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}} \rightarrow x = a_1$ and $\varphi \wedge (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}} \rightarrow x = a_1 \vee x = a_2$ corresponding to the preference statement $\varphi : a_1 \succ a_2 \succ a_3$. The only way an implication is not satisfied is when the hypothesis is *true* and the conclusion is *false*. Let us take the first implication: $O \vdash \varphi$, $O \vdash (C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}}$ and $O \vdash (x = a_2 \vee x = a_3)$. In this situation, flipping from $(x = a_2 \vee x = a_3)$ to $x = a_1$ would give us a new outcome O' such that $O' \vdash x = a_1$ and this would be an improvement according to p. However, by doing so, we have to make sure that the constraints in C containing $x = a_2$ or $x = a_3$ may now not be satisfied, since now $(x = a_2 \vee x = a_3)$ is false. However, we also have that $O \vdash (C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}}$, meaning that if $x = a_1$ these constraints are satisfied. Hence, there is an improving flip to another feasible outcome O'. But O was supposed to be undominated. Therefore O satisfies the first of the two implications above.

Let us now consider the second implication: $O \vdash \varphi$, $O \vdash (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}$ and $O \vdash x = a_3$. In this situation, flipping from $x = a_3$ to $x = a_2$ would give us a new outcome O' such that $O' \vdash x = a_2$ and this would be an improvement according to p. However, by doing so, we have to make sure that the constraints in C containing $x = a_3$ may now not be satisfied, since now $x = a_3$ is false. However, we also have that $O \vdash (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}$, meaning that if $x = a_2$ these constraints are satisfied.

[2] More precisely, $C_x = \{c \in C | x \in con_c\}$.

Hence, there is an improving flip to another feasible outcome O'. But O was supposed to be undominated. Therefore O satisfies the second implication above. Thus O must satisfy all constraints $opt_C(p)$ where $p \in N$. Since it is also feasible, O is a satisfying assignment of $N \oplus C$.

($\Leftarrow$) Consider any assignment O which satisfies $N \oplus C$. Clearly it is feasible as $N \oplus C$ includes C. Suppose we perform an improving flip in O. There are two cases. Suppose that the outcomes obtained by performing any improving flip are not feasible. Then such new outcomes do not dominate the old one in our semantics. Thus O is optimal.

Suppose, on the other hand, that there is at least one new outcome, obtained via an improving flip, which is feasible. Assume the flips passes from $x = a_3$ to $x = a_2$. If this is an improving flip, without loss of generality, there must exist a statement φ : $\ldots \succ x = a_2 \succ x = a_3 \succ \ldots$ in N such that $O \vdash \varphi$. By hypothesis, O is a satisfying assignment of $N \oplus C$. Therefore $O \vdash opt(\varphi : \ldots \succ x = a_2 \succ \ldots \succ x = a_3 \succ \ldots)$ = $\varphi \wedge (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}} \rightarrow \ldots \vee x = a_2$. Since $O \vdash \varphi$ and $O \vdash x = a_3$, and *true* is not allowed to imply *false*, O cannot satisfy $(C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}$. But, as the new outcome, which contains $x = a_2$, is feasible, such constraints have to be satisfied independent of how we set x. Hence, $O \vdash (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}$. As this is a contradiction, this cannot be an improving flip to a feasible outcome. The satisfying assignment is therefore feasible and undominated. □

Notice that the construction $N \oplus C$ for variables with more than two values in their domains is a generalization of the one for Boolean variables. That is, $N \oplus C = N \oplus_b C$ if N and C are over Boolean variables. Similar complexity results hold also now. However, while for Boolean variables one constraint is generated for each preference statement, now we generate as many constraints as the size of the domain minus 1. Therefore the optimality constraints corresponding to a constrained CP-net $\langle N, C \rangle$ are $|C| + \mid N \mid \times \mid D \mid$, where D is the domain of the variables. Testing optimality is still linear in the size of $\langle N, C \rangle$, if we assume D bounded. Finding an optimal outcome as usual requires us to find a solution of the constraints in $N \oplus C$, which is NP-hard in the size of $\langle N, C \rangle$.

7 CP-nets and soft constraints

It may be that we have soft and not hard constraints to add to our CP-net. For example, we may have soft constraints representing other quantitative preferences. In the rest of this section, a constrained CP-net will be a pair $\langle N, C \rangle$, where N is a CP-net and C is a set of soft constraints. Notice that this definition generalizes the one given in Section 6 since hard constraints can be seen as a special case of soft constraints (see Section 2).

The construction of the optimality constraints for constrained CP-nets can be adapted to work with soft constraints. To be as general as possible, variables can again have more than two values in their domains. The constraints we obtain are very similar to those of the previous sections, except that now we have to reason about optimization as soft constraints define an optimization problem rather than a satisfaction problem.

Consider any CP statement p of the form $\varphi : x = a_1 \succ x = a_2 \succ x = a_3$. For simplicity, we again consider just 3 values. However, all the constructions and arguments extend easily to more values. The optimality constraints corresponding to p, called $opt_{soft}(p)$, are the following hard constraints:

$$\varphi \wedge cut_{best(C)}((\varphi \wedge C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}}) \rightarrow x = a_1$$

$$\varphi \wedge cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}) \rightarrow x = a_1 or x = a_2$$

where C_x is the subset of soft constraints in C which involve variable x, $best(S)$ is the highest preference value for a complete assignment of the variables in the set of soft constraints S, and $cut_\alpha S$ is a hard constraint obtained from the soft constraint S by forbidding all tuples which have preference value less than α in S. The optimality constraints corresponding to $\langle N, C\rangle$ are $C_{opt}(\langle N, C\rangle) = \{opt_{soft}(p) \mid p \in N\}$.

Consider a CP-net with two features, X and Y, such that the domain of Y contains y_1 and y_2, while the domain of X contains x_1, x_2, and x_3. Moreover, we have the following CP-net preference statements: $y_1 \succ y_2$, $y_1 : x_1 \succ x_2 \succ x_3$, $y_2 : x_2 \succ x_1 \succ x_3$. We also have a soft (fuzzy) unary constraint over X, which gives the following preferences over the domain of X: 0.1 to x_1, 0.9 to x_2, and 0.5 to x_3. By looking at the CP-net alone, the ordering over the outcomes is given by $y_1x_1 \succ y_1x_2 \succ y_1x_3 \succ y_2x_3$ and $y_1x_2 \succ y_2x_2 \succ y_2x_1 \succ y_2x_3$. Thus y_1x_1 is the only optimal outcome of the CP-net. On the other hand, by taking the soft constraint alone, the optimal outcomes are all those with $X = x_2$ (thus y_1x_2 and y_2x_2).

Let us now consider the CP-net and the soft constraints together. To generate the optimality constraints, we first compute best(C), which is 0.9. Then, we have:

- for statement $y_1 \succ y_2$: $Y = y_1$;
- for statement $y_1 : x_1 \succ x_2 \succ x_3$: we generate the constraints $Y = y_1 \wedge false \rightarrow X = x_1$ and $Y = y_1 \wedge Y = y_1 \rightarrow X = x_1 \vee X = x_2$. Notice that we have false in the condition of the first implication because $cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_1) \downarrow_Y = false$. On the other hand, in the condition of the second implication we have $cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_2) \downarrow_Y = (Y = y_1)$. Thus, by removing false, we have just one constraint: $Y = y_1 \rightarrow X = x_1 \vee X = x_2$;
- for statement $y_2 : x_2 \succ x_1 \succ x_3$: similarly to above, we have the constraint $Y = y_2 \rightarrow X = x_2$.

Let us now compute the optimal solutions of the soft constraint over X which are also feasible for the following set of constraints: $Y = y_1, Y = y_1 \rightarrow X = x_1 \vee X = x_2$, $Y = y_2 \rightarrow X = x_2$. The only solution which is optimal for the soft constraints and feasible for the optimality constraints is y_1x_2. Thus this solution is optimal for the constrained CP-net.

Notice that the optimal outcome for the constrained CP-net of the above example is not optimal for the CP-net alone. In general, an optimal outcome for a constrained CP-net has to be optimal for the soft constraints, and such that there is no other outcome which can be reached from it in the ordering of the CP-net with an improving chain of optimal outcomes. Thus, in the case of CP-nets constrained by soft constraints, Definition 2 is replaced by the following one:

Definition 3 **($O1 \succ_{soft} O_2$).** *Given a constrained CP-net $\langle N, C\rangle$, where C is a set of soft constraints, outcome O_1 is* **better** *than outcome O_2 (written $O_1 \succ_{soft} O_2$) iff there* *is a chain of flips from O_1 to O_2, where each flip is worsening for N and each outcome in the chain is optimal for C.*

Notice that this definition is just a generalization of Def. 2, since optimality in hard constraints is simply feasibility. Thus $\succ = \succ_{soft}$ when C is a set of hard constraints.

Consider the same CP-net as in the previous example, and a binary fuzzy constraint over X and Y which gives preference 0.9 to x_2y_1 and x_1y_2, and preference 0.1 to all other pairs. According to the above definition, both x_2y_1 and x_1y_2 are optimal for the constrained CP-net, since they are optimal for the soft constraints and there are no improving path of optimal outcomes between them in the CP-net ordering. Let us check that the construction of the optimality constraints obtains the same result:

- for $y_1 \succ y_2$ we get $cut_{0.9}(C_y \wedge Y = y_1) \downarrow_X \rightarrow Y = y_1$. Since $cut_{0.9}(C_y \wedge Y = y_1) \downarrow_X = (X = x_2)$, we get $X = x_2 \rightarrow Y = y_1$.
- for statement $y_1 : x_1 \succ x_2 \succ x_3$: $Y = y_1 \wedge cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_1) \downarrow_Y \rightarrow X = x_1$. Since $cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_1) \downarrow_Y = false$, we get a constraint which is always true. Also, we have the constraint $Y = y_1 \wedge cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_2) \downarrow_Y \rightarrow X = x_1 \vee X = x_2$. Since $cut_{0.9}(Y = y_1 \wedge C_x \wedge X = x_2) \downarrow_Y = (Y - y_1)$, we get $Y = y_1 \wedge Y = y_1 \rightarrow X = x_1 \vee X = x_2$.
- for statement $y_2 : x_2 \succ x_1 \succ x_3$: similarly to above, we have the constraint $Y = y_2 \rightarrow X = x_2 \vee X = x_1$.

Thus the set of optimality constraints is the following one: $X = x_2 \rightarrow Y = y_1$, $Y = y_1 \rightarrow X = x_1 \vee X = x_2$, and $Y = y_2 \rightarrow X = x_2 \vee X = x_1$. The feasible solutions of this set of constraints are x_2y_1, x_1y_1, and x_1y_2. Of these constraints, the optimal outcomes for the soft constraint are x_2y_1 and x_1y_2. Notice that, in the ordering induced by the CP-net over the outcomes, these two outcomes are not linked by a path of improving flips through optimal outcomes for the soft constraints. Thus they are both optimal for the constrained CP-net.

Theorem 4. *Given a constrained CP-net $\langle N, C\rangle$, where C is a set of soft constraints, an outcome is optimal for $\langle N, C\rangle$ iff it is an optimal assignment for C and if it satisfies $C_{opt}(\langle N, C\rangle)$.*

Proof. ($\Rightarrow$) Consider an outcome O that is optimal for $\langle N, C\rangle$. Then by definition it must be optimal for C. Suppose the outcome does not satisfy C_{opt}. Therefore O must not satisfy some constraint $opt_C(p)$ where p preference statement in N. Without loss of generality, let us consider the optimality constraints

$$\varphi \wedge cut_{best(C)}((\varphi \wedge C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}}) \rightarrow x = a_1$$

$$\varphi \wedge cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}) \rightarrow x = a_1 or x = a_2$$

corresponding to the preference statement $\varphi : a_1 \succ a_2 \succ a_3$.

The only way an implication is not satisfied is when the hypothesis is *true* and the conclusion is *false*. Let us take the first implication: $O \vdash \varphi$, $O \vdash cut_{best(C)}((\varphi \wedge C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}})$ and $O \vdash (x = a_2 \vee x = a_3)$. In this situation, flipping from $(x = a_2 \vee x = a_3)$ to $x = a_1$ would give us a new outcome O' such that

$O' \vdash x = a_1$ and this would be an improvement according to p. However, by doing so, we have to make sure that the soft constraints in C containing $x = a_2$ or $x = a_3$ may now still be satisfied optimally, since now $(x = a_2 \vee x = a_3)$ is false. We also have that $O \vdash cut_{best(C)}((\varphi \wedge C_x \wedge x = a_1) \downarrow_{var(C_x)-\{x\}})$, meaning that if $x = a_1$ these constraints are satisfied optimally. Hence, there is an improving flip to another outcome O' which is optimal for C and which satisfies C_{opt}. But O was supposed to be undominated. Therefore O satisfies the first of the two implications above.

Let us now consider the second implication: $O \vdash \varphi$, $O \vdash cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}})$, and $O \vdash x = a_3$. In this situation, flipping from $x = a_3$ to $x = a_2$ would give us a new outcome O' such that $O' \vdash x = a_2$ and this would be an improvement according to p. However, by doing so, we have to make sure that the constraints in C containing $x = a_3$ may now still be satisfied optimally, since now $x = a_3$ is false. However, we also have that $O \vdash cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}})$, meaning that if $x = a_2$ these constraints are satisfied optimally. Hence, there is an improving flip to another feasible outcome O'. But O was supposed to be undominated. Therefore O satisfies the second implication above. Thus O must satisfy all the optimality constraints $opt_C(p)$ where $p \in N$.

($\Leftarrow$) Consider any assignment O which is optimal for C and satisfies C_{opt}. Suppose we perform a flip on O. There are two cases. Suppose that the new outcome is not optimal for C. Then the new outcome does not dominate the old one in our semantics. Thus O is optimal. Suppose, on the other hand, that there is at least one new outcome, obtained via an improving flip, which is optimal for C and satisfies C_{opt}. Assume the flip passes from $x = a_3$ to $x = a_2$. If this is an improving flip, without loss of generality, there must exist a statement $\varphi : \ldots \succ x = a_2 \succ x = a_3 \succ \ldots$ in N such that $O \vdash \varphi$. By hypothesis, O is an optimal assignment of C and satisfies C_{opt}. Therefore $O \vdash opt(\varphi : \ldots \succ x = a_2 \succ \ldots \succ x = a_3 \succ \ldots) = \varphi \wedge cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}) \rightarrow \ldots \vee x = a_2$.

Since $O \vdash \varphi$ and $O \vdash x = a_3$, and *true* is not allowed to imply *false*, O cannot satisfy $cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}})$. But O', which contains $x = a_2$, is assumed to be optimal for C, so $cut_{best(C)}((\varphi \wedge C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}})$ has to be satisfied independently of how we set x. Hence, $O \vdash (C_x \wedge x = a_2) \downarrow_{var(C_x)-\{x\}}$. As this is a contradiction, this cannot be an improving flip to an outcome which is optimal for C and satisfies C_{opt}. Thus O is optimal for the constrained CP-net. □

It is easy to see how the construction of this section can be used when a CP-net is constrained by a set of both hard and soft constraints, or by several sets of hard and soft constraints, since they can all be seen as just one set of soft constraints.

Let us now consider the complexity of constructing the optimality constraints and of testing or finding optimal outcomes, in the case of CP-nets constrained by soft constraints. First, as with hard constraints, the number of optimality constraints we generate is $|N|| \times (|D| - 1)$, where $|N|$ is the number of preference statements in N and D is the domain of the variables. Thus we have $|C_{opt}(\langle N, C \rangle)| = |N| \times (|D| - 1)$. To test if an outcome O is optimal, we need to check if O satisfies C_{opt} and if it is optimal for C. Checking feasibility for C_{opt} takes linear time in $|N| \times (|D| - 1)$. Then, we need to check if O is optimal for C. This is NP-hard the first time we do it, otherwise (if the optimal preference value for C is known) is linear in the size of C. To find an optimal outcome, we need to find the optimals for C which are also feasible for C_{opt}. Finding optimals for C needs exponential time in the size of C, and checking feasibility in C_{opt} is linear in the size of C_{opt}. Thus, with respect to the corresponding results for hard constraints, we only need to do more work the first time we want to test an outcome for optimality.

8 Multiple constrained CP-nets

There are situations when we need to represent the preferences of multiple agents. For example, when we are scheduling workers, each will have a set of preferences concerning the shifts. These ideas generalize to such a situation. Consider several CP-nets N_1, $\ldots$, N_k, and a set of hard or soft constraints C. We will assume for now that all the CP nets have the same features. To begin, we will say an outcome is optimal iff it is optimal for each constrained CP net $\langle N_i, C\rangle$. This is a specific choice but we will see later that other choices can be considered as well. We will call this notion of optimality, *All-optimal*.

Definition 4 (All-optimal). *Given a multiple constrained CP net $M = \langle (N_1,\ldots, N_k), C\rangle$, an outcome O is All-optimal for M if O if it is optimal for each constrained CP net $\langle N_i, C\rangle$.*

This definition, together with Theorem 4, implies that to find the all-optimal outcomes for M we just need to generate the optimality constraints for each constrained CP net $\langle N_i, C\rangle$, and then take the outcomes which are optimal for C and satisfy all optimality constraints.

Theorem 5. *Given a multiple constrained CP net $M = \langle (N_1, \ldots, N_k), C\rangle$, an outcome O is All-optimal for M iff O is optimal for C and it satisfies the optimality constraints in $\bigcup_i C_{opt}(\langle N_i, C\rangle)$.*

This semantics is one of consensus: all constrained CP nets must agree that an outcome is optimal to declare it optimal for the multiple constrained CP net. Choosing this semantics obviously satisfies all CP nets. However, there could be no outcome which is optimal. In [8] a similar consensus semantics (although for multiple CP nets, with no additional constraints) is called Pareto optimality, and it is one among several alternative to aggregate preferences expressed via several CP nets. This semantics, adapted to our context, would be defined as follows:

Definition 5 (Pareto). *Given a multiple constrained CP net $M = \langle (N_1, \ldots, N_k), C\rangle$, an outcome O is Pareto-better than an outcome O' iff it is better for each constrained CP net. It is Pareto-optimal for M iff there is no other outcome which is Pareto-better.*

If an outcome is all-optimal, it is also Pareto-optimal. However, the converse is not true in general. These two semantics may seem equally reasonable. However, while all-optimality can be computed via the approach of this paper, which avoids dominance testing, Pareto optimality needs such tests, and therefore it is in general much more expensive to compute. In particular, whilst optimality testing for Pareto optimality requires exponential time, for All-optimality it just needs linear time (in the sum of the sizes of the CP nets).

Other possibilities proposed in [8] require optimals to be the best outcomes for a majority of CP nets (this is called Majority), or for the highest number of CP nets (called Max). Other semantics like Lex, associate each CP net with a priority, and then declare optimal those outcomes which are optimal for the CP nets with highest priority, in a lexicographical fashion. In principle, all these semantics can be adapted to work with multiple constrained CP nets. However, as for Pareto optimality, whilst their definition is possible, reasoning with them would require more than just satisfying a combination of the optimality constraints, and would involve dominance testing.

Thus the main gain from our semantics (all-optimal and others that can be computed via this approach) is that dominance testing is not required. This makes optimality testing (after the first test) linear rather than exponential, although finding optimals remains difficult (as it is when we find the optimals of the soft constraints and check the feasibility of the optimality constraints).

9 Related work

The closest work is [2], where acyclic CP nets are constrained via hard constraints, and an algorithm is proposed to find one or all the optimal outcomes of the constrained CP net. However, there are several differences. First, the notion of optimality in this previous approach is different from the one used here: in [2], an outcome O is optimal if satisfies the constraints and there is no other feasible outcome which is better than it in the CP net ordering. Therefore, if two outcomes are both feasible and there is an improving path from one to the other one in the CP net, but they are not linked by a path of feasible outcomes, then in this previous approach only the highest one is optimal, while in ours they are both optimal. For example, assume we have a CP net with two Boolean features, A and B, and the following CP statements: $a \succ \overline{a}$, $a : b \succ \overline{b}$, $\overline{a} : \overline{b} \succ b$, and the constraint $\overline{a} \vee b$ which rules out $a\overline{b}$. Then, the CP net ordering on outcomes is $ab \succ a\overline{b} \succ \overline{a}\overline{b} \succ \overline{a}b$. In our approach, both ab and $\overline{a}\overline{b}$ are optimal, whilst in in the previous approach only ab is optimal. Thus we obtain a superset of the optimals computed in the previous approach.

Reasoning about this superset is, however, computationally more attractive. To find the first optimal outcome, the algorithm in [2] uses branch and bound and thus has a complexity that is comparable to solving the set of constraints. Then, to find other optimal outcomes, they need to perform dominance tests (as many as the number of optimal outcomes already computed), which are very expensive. In our approach, to find one optimal outcome we just need to solve a set of optimality constraints, which is NP-hard.

Two issues that are not addressed in [2] are testing optimality efficiently and reasoning with cyclic CP nets. To test optimality, we must run the branch and bound algorithm to find all optimals, and stop when the given outcome is generated or when all optimals are found. In our approach, we check the feasibility of the given outcome with respect to the optimality constraints. Thus it takes linear time. Our approach is based on the CP statements and not on the topology of the dependency graph. Thus it works just as well with cyclic CP nets.

Another related work is [7], where CP nets orderings are approximated via a set of soft constraints. The approximation here is not needed, since we are not trying to model the entire ordering over outcomes, but only the set of optimals.

Finally, our construction can be seen as a generalization of that given in Section 4 of [4], where they treat the case of mapping a CP net on Boolean features, without any constraints, onto a SAT problem.

10 Conclusions

We have presented a novel approach to deal with preferences expressed as a mixture of hard constraints, soft constraints, and CP nets. The main idea is to generate a set of hard constraints whose solutions are optimal for the preferences. Our approach focuses on finding and testing optimal solutions. It avoids the costly dominance tests previously used to reason about CP nets. To represent the preferences of multiple agents, we have also considered multiple CP nets. We have shown that it is possible to define semantics for preference aggregation for multiple CP nets which also avoid dominance testing. One of the main advantages of this simple and elegant technique is that it permits conventional constraint and SAT solvers to solve problems involving both preferences and constraints.

Acknowledgements. This work is partially supported by ASI (Italian Space Agency) under project ARISCOM (Contract I/R/215/02).

References

1. S. Bistarelli, U. Montanari, F. Rossi. Semiring-based Constraint Solving and Optimization. Journal of ACM, vol. 44, n. 2, pp. 201-236, March 1997.
2. C. Boutilier, R. Brafman, C. Domshlak, H. Hoos and D. Poole. Preference-based constraint optimization with CP-nets. Computational Intelligence, vol. 20, n. 2, pp. 137-157, May 2004.
3. C. Boutilier, R. Brafman, H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *Proceedings of 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pp. 71-80, Stockholm, Sweden, 1999.
4. R. Brafman and Y. Dimopoulos. Extended Semantics and Optimization Algorithms for CP-Networks. Computational Intelligence, vol. 20, n. 2, pp. 218-245, May 2004.
5. R. Dechter. Constraint processing. Morgan Kaufmann, 2003.
6. C. Domshlak, R. I. Brafman. CP-nets - reasoning and consistency testing. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR-02)*, pages 121–132. Morgan Kaufmann, Toulose, France, 2002.
7. C. Domshlak, F. Rossi, K. B. Venable, T. Walsh. Reasoning about soft constraints and conditional preferences: complexity results and approximation techniques. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, Morgan Kaufmann, Acapulco, Mexico, August 2003.
8. F. Rossi, K. B. Venable, T. Walsh. mCP nets: representing and reasoning with preferences of multiple agents. In *Proceeding of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, San Jose, CA, USA, July 2004.