Reasoning about soft constraints and conditional preferences: complexity results and approximation techniques*

C. Domshlak
Dept. of Computer Science
Cornell University
Ithaca, NY, USA
dcarmel@cs.cornell.edu

F. Rossi and **K. B. Venable**
Dept. of Mathematics
University of Padova
Padova, Italy
{frossi,kvenable}@math.unipd.it

T. Walsh
Cork Constraint Computation Centre
University College Cork
Cork, Ireland
tw@4c.ucc.ie

Abstract

Many real life optimization problems contain both hard and soft constraints, as well as qualitative conditional preferences. However, there is no single formalism to specify all three kinds of information. We therefore propose a framework, based on both CP-nets and soft constraints, that handles both hard and soft constraints as well as conditional preferences efficiently and uniformly. We study the complexity of testing the consistency of preference statements, and show how soft constraints can faithfully approximate the semantics of conditional preference statements whilst improving the computational complexity.

1 Introduction and Motivation

Representing and reasoning about preferences is an area of increasing interest in theoretical and applied AI. In many real life problems, we have both hard and soft constraints, as well as qualitative conditional preferences. For example, in a product configuration problem, the producer may have hard and soft constraints, while the user has a set of conditional preferences. Until now, there has been no single formalism which allows all these different kinds of information to be specified efficiently and reasoned with effectively. For example, soft constraint solvers [Bistarelli *et al.*, 1997; Schiex *et al.*, 1995] are most suited for reasoning about the hard and soft constraints, while CP-nets [Boutilier *et al.*, 1999] are most suited for representing qualitative conditional preference statements. In this paper, we exploit a connection between these two approaches, and define a framework based on both CP-nets and soft constraints which can efficiently handle both constraints and preferences.

Soft constraints [Bistarelli *et al.*, 1997; Schiex *et al.*, 1995] are one of the main methods for dealing with preferences in constraint optimization. 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. Whilst soft constraints are very expressive, and have a powerful computational machinery, they are not good at modeling and solving the sort of conditional preference statements that occur in the real world. Moreover, soft constraints are based on *quantitative* measures of preference, which tends to make preference elicitation more difficult.

Qualitative user preferences have been widely studied in decision-theoretic AI [Doyle and Thomason, 1999]. Of particular interest are CP-nets [Boutilier *et al.*, 1999]. These model statements of qualitative and conditional preference such as "I prefer a red dress to a yellow dress", or "If the car is convertible, I prefer a soft top to a hard top". These are interpreted under the *ceteris paribus* (that is, "all else being equal") assumption. Preference elicitation in such a framework is intuitive, independent of the problem constraints, and suitable for naive users. However, the Achilles heel of CP-nets and other sophisticated qualitative preference models [Lang, 2002] is the complexity of reasoning with them [Domshlak and Brafman, 2002; Boutilier *et al.*, 2002].

Motivated by a product configuration application [Sabin and Weigel, 1998], we have developed a framework to reason simultaneously about qualitative conditional preference statements and hard and soft constraints. In product configuration, the producer has hard (e.g., component compatibility) and soft (e.g., supply time) constraints, while the customer has preferences over the product features. We first investigate the complexity of reasoning about qualitative preference statements, addressing in particular preferential consistency. To tackle the complexity of preference reasoning, we then introduce two approximation schemes based on soft constraints.

To the best of our knowledge, this work provides the first connection between the CP-nets and soft constraints machinery. In addition, for product configuration problems or any problem with both hard and soft quantitative constraints as well as qualitative conditional preferences, this framework lets us treat the three kinds of information in a unifying environment. Finally, we compare the two approximations in terms of both expressivity and complexity.

*This research was partially funded by AFOSR, grant F49620-01-1-0076 (Intelligent Information Systems Institute, Cornell University) and F49620-01-1-0361 (MURI grant on Cooperative Control of Distributed Autonomous Vehicles in Adversarial Environments), DARPA, F30602-00-2-0530 (Controlling Computational Cost: Structure, Phase Transitions and Randomization) and F30602-00-2-0558 (Configuring Wireless Transmission and Decentralized Data Processing for Generic Sensor Networks), the Italian MIUR projects NAPOLI and COVER, the ASI (Italian Space Agency) project ARISCOM and Science Foundation Ireland.

2 Formalisms for Describing Preferences

2.1 Soft constraints

There are many formalisms for describing *soft constraints*. We use the c-semi-ring formalism [Bistarelli *et al.*, 1997], which is equivalent to the valued-CSP formalism when total orders are used [Bistarelli *et al.*, 1996], as this generalizes many 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 *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. For example, a classical CSP is an SCSP with the c-semi-ring $S_{CSP} = \langle\{false, true\}, \vee, \wedge, false, true\rangle$, a fuzzy CSP [Schiex, 1992] is an SCSP with the c-semi-ring $S_{FCSP} = \langle[0, 1], max, min, 0, 1\rangle$, and probabilistic and weighted CSPs are SCSPs with the c-semi-rings $S_{prob} = \langle[0, 1], max, \times, 0, 1\rangle$ and $S_{weight} = \langle\mathcal{R}, min, +, 0, +\infty\rangle$, respectively. A solution to an SCSP is a complete assignment to its variables. The preference value associated with a solution is obtained by multiplying the preference values of the projections of the solution to each constraint. One solution is better than another if its preference value is higher in the partial order. Finding an optimal solution for an SCSP is an NP-complete problem. On the other hand, given two solutions, checking whether one is preferable is easy: we compute the semi-ring values of the two solutions and compare the resulting values.

2.2 CP-nets

Soft constraints are the main tool for representing and reasoning about preferences in constraint satisfaction problems. However, they require the choice of a semi-ring value for each variable assignment in each constraint. They are therefore a *quantitative* method for expressing preferences. In many applications, it is more natural for users to express preferences via generic qualitative (usually partial) preference relations over variable assignments. For example, it is often more intuitive for the user to say "I prefer red wine to white wine", rather than "Red wine has preference 0.7 and white wine has preference 0.4". Of course, the former statement provides less information, but it does not require careful selection of preference values to maintain consistency. Moreover, soft constraints do not naturally represent conditional preferences, as in "If they serve meat, then I prefer red wine to white wine". It is easy to see that both qualitative statements and conditions are essential ingredients in many applications.

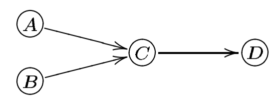

Figure 1: The CP-net graph for the example.

CP-nets [Boutilier *et al.*, 1999] are a graphical model for compactly representing conditional and qualitative preference relations. They exploit conditional preferential independence by structuring a user'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 philosophers (see [Hansson, 2001] for an overview) and AI researchers [Doyle and Wellman, 1994], have argued that most of our preferences are of this type.

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 $\mathbf{F} = \{X_1, \ldots, X_n\}$ with finite, discrete domains $\mathcal{D}(X_1), \ldots, \mathcal{D}(X_n)$ (these play the same role as variables in soft constraints). During preference elicitation, for each feature X_i, the user is asked to specify a set of *parent* features $Pa(X_i)$ that can affect her preferences over the values of X_i. This information is used to create the graph of the CP-net in which each node X_i has $Pa(X_i)$ as its immediate predecessors. Given this structural information, the user is asked to explicitly specify her preference over the values of X_i for *each complete assignment* on $Pa(X_i)$, and this preference is assumed to take the form of total [Boutilier *et al.*, 1999] or partial [Boutilier *et al.*, 2002] order over $\mathcal{D}(X)$. These conditional preferences over the values of X_i are annotated with the node X_i in the CP-net. For example, consider a CP-net with the graph given in Figure 1, 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 of the user for $A = a$ over $A = \overline{a}$, while statement $c : d \succ \overline{d}$ represents that the user prefers $D = d$ to $D = \overline{d}$, given that $C = c$.

Several types of queries can be asked about CP-nets. First, given a CP-net N, one might be interested in finding an optimal assignment to the features of N. For acyclic CP-nets, such a query is answerable in linear time [Boutilier *et al.*, 1999]. Second, given a CP-net N and a pair of complete assignments α and β, one might be interested in determining whether $\alpha \succ \beta$, i.e. α is preferred to β. Unfortunately, this query is NP-hard even for acyclic CP-nets [Domshlak and Brafman, 2002] though some tractable special cases do exist.

3 Consistency and Satisfiability

Given a set of preference statements Ω extracted from a user, we might be interested in testing *consistency* of the induced preference relation. In general, there is no single notion of preferential consistency [Hansson, 2001]. In [Boutilier *et al.*, 1999], a CP-net N was considered consistent iff the partial ordering $\succ$ induced by N is *asymmetric*, i.e. there exist at least one total ordering of the outcomes consistent with $\succ$. However, in many situations, we can ignore cycles in the preference relation, as long as these do not prevent a rational choice, i.e. there exist an outcome that is not dominated by any other outcome. In what follows, we refer to this as *satisfiability*[1]. It is easy to see that satisfiability is strictly weaker than asymmetry, and that asymmetry implies satisfiability. We will consider two cases: When the set Ω of preference statements induces a CP-net and, more generally, when preferences can take any form (and may not induce a CP-net).

When Ω defines an acyclic CP-net, the partial order induced by Ω is asymmetric [Boutilier *et al.*, 1999]. However, for cyclic CP-nets, asymmetry is not guaranteed. In the more general case, we are given a set Ω of conditional preference statements without any guarantee that they define a CP-net. Let the *dependence graph* of Ω be defined similarly to the graphs of CP-nets: the nodes stand for problem features, and a directed arc goes from X_i to X_j iff Ω contains a statement expressing preference on the values of X_j conditioned on the value of X_i. For example, the set $\Omega = \{a : b \succ \overline{b}, a \wedge c : \overline{b} \succ b\}$ does not induce a CP-net (the two conditionals are not mutually exclusive), and the preference relation induced by Ω is not asymmetric, despite the fact that the dependence graph of Ω is acyclic.

Note that while asymmetry implies satisfiability, the reverse does not hold in general. For example, the set Ω above is not asymmetric, but it is satisfiable (the assignment $a\overline{c}b$ is undominated). Given such a satisfiable set of statements, we can prompt the user with one of the undominated assignments without further refinement of its preference relation. Theorem 1 shows that, in general, determining satisfiability of a set of statements is NP-complete. On the other hand, even for CP-nets, determining asymmetry is not known to be in NP [Domshlak and Brafman, 2002].

Theorem 1 SATISFIABILITY *of a set of conditional preference statements* Ω *is* NP*-complete.*

Proof: Membership in NP is straightforward, as an assignment is a polynomial-size witness that can be checked for non-dominance in time linear in the size of Ω. To show hardness, we reduce 3-SAT to our problem: Given a 3-cnf formula F, for each clause $(x \vee y \vee z) \in F$ we construct the conditional preference statement: $\overline{x} \wedge \overline{y} : z \succ \overline{z}$. This set of conditional preferences is satisfiable iff the original original formula F is satisfiable. $\square$

[1] In preference logic [Hansson, 2001], these notions of "consistency as satisfi ability" and "consistency as asymmetry" correspond to the notions of *eligibility* and *restrictable eligibility*, respectively. However, we will use the former terms as they seem more intuitive.

While testing satisfiability is hard in general, Theorem 2 presents a wide class of statement sets that can be tested for satisfiability in polynomial time.

Theorem 2 *A set of conditional preference statements* Ω*, whose dependency graph is acyclic and has bounded node in-degree can be tested for satisfiability in polynomial time.*

Proof: The proof is constructive, and the algorithm is as follows: First, for each feature $X \in \mathbf{V}$, we construct a table T_X with an entry for each assignment $\pi \in \mathcal{D}(Pa(X))$, where each entry $T_X[\pi]$ contains all the values of X that are not dominated given Ω and π. Subsequently, we remove all the empty entries. For example, let A, B and C be a set of boolean problem features, and let $\Omega = \{c \succ \overline{c}, a : b \succ \overline{b}, a \wedge c : \overline{b} \succ b\}$. The corresponding table will be as follows:

Feature	π	Values
T_A	$\emptyset$	$\{a, \overline{a}\}$
T_C	$\emptyset$	$\{c\}$
T_B	$a \wedge \overline{c}$	$\{b\}$
	$\overline{a} \wedge \overline{c}$	$\{b, \overline{b}\}$
	$\overline{a} \wedge c$	$\{b, \overline{b}\}$

Observe that the entry $T_B[a \wedge c]$ has been removed, since, given $a \wedge c$, b and $\overline{b}$ are dominated according to the statements $a \wedge c : \overline{b} > b$ and $a : b > \overline{b}$, respectively. Since the in-degree of each node X in the dependence graph of Ω is bounded by a constant k (i.e. $|Pa(X)| \leq k$), these tables take space and can be constructed in time $O(n2^k)$. Given such tables for all the features in $\mathbf{V}$, we traverse the dependence graph of Ω in a topological order of its nodes, and for each node X being processed we remove all the entries in T_X that are not "supported" by (already processed) $Pa(X)$: An entry $T_X[\pi]$ is not supported by $Pa(X)$ if there exists a feature $Y \in Pa(X)$ such that the value provided by π to Y appears in no entry of T_Y. For instance, in our example, the rows corresponding to $a \wedge \overline{c}$ and $\overline{a} \wedge \overline{c}$ will be removed, since $\overline{c}$ does not appear in the (already processed) table of C. Now, if the processing of a feature X results in $T_X = \emptyset$, then Ω is not satisfiable. Otherwise, any assignment to $\mathbf{V}$ consistent with the processed tables will be non-dominated with respect to Ω. $\square$

Note that, for sets of preference statements with cyclic dependence graphs, SATISFIABILITY remains hard even if the in-degree of each node is bounded by $k \geq 6$, since 3-SAT remains hard even if each variable participates in at most three clauses of the formula the proof of Theorem 1). However, when at most one condition is allowed in each preference statement, and the features are boolean, then SATISFIABILITY can be reduced to 2-SAT, and thus tested in polynomial time. Further study of additional tractable cases is clearly of both theoretical and practical interest.

4 Approximating CP-nets with Soft Constraints

In addition to testing consistency and determining preferentially optimal outcomes, we can be interested in the *preferential comparison* of two outcomes. Unfortunately, determining dominance between a pair of outcomes with respect to a set of qualitative preferential statements under the *ceteris paribus* assumption is PSPACE-complete in general [Lang, 2002], and is NP-hard even for acyclic CP-nets [Domshlak and Brafman, 2002]. However, given a set Ω of preference statements, instead of using a preference relation $\succ$ induced by Ω, one can use an approximation $\gg$ of $\succ$, achieving tractability while sacrificing precision to some degree. Clearly, different approximations $\gg$ of $\succ$ are not equally good, as they can be characterized by the precision with respect to $\succ$, time complexity of generating $\gg$, and time complexity of comparing outcomes with respect to $\gg$. In addition, it is vital that $\gg$ faithfully extends $\succ$ (i.e. $\alpha \succ \beta$ should entail $\alpha \gg \beta$). We call this *information preserving*. Another desirable property of approximations is that of preserving the *ceteris paribus* property (we call this the *cp-condition* for short).

For acyclic CP-nets, two approximations that are information preserving have been introduced, both comparing outcomes in time linear in the number of features. The first is based on the relative position of the features in the CP-net graph [Boutilier *et al.*, 2002]. This approximation does not require any preprocessing of the CP-net. However, it is problematic when there are hard constraints. The second, based on UCP-nets [Boutilier *et al.*, 2001], can be used as a quantitative approximation of acyclic CP-nets. UCP-nets resemble weighted CSPs, and thus they can be used in constraint optimization using the soft constraints machinery. However, generating UCP-nets is exponential in the size of CP-net node's Markov family[2], and thus in the CP-net node out-degree. An additional related work is described in [McGeachie and Doyle, 2002], where a numerical value function is constructed using graph-theoretic techniques by examining the graph of the preference relation induced by a set of preference statements. Note that this framework is also computationally hard, except for some special cases.

Here we study approximating CP-nets via soft constraints (SCSPs). This allows us to use the rich machinery underlying SCSPs to answer comparison queries in linear time. Moreover, this provides us a uniform framework to combine user preferences with both hard and soft constraints. Given an acyclic CP-net, we construct a corresponding SCSP in two steps. First, we build a constraint graph, which we call *SC-net*. Second, we compute the preferences and weights for the constraints in the SC-net, and this computation depends on the actual semi-ring framework being used. Here we present and discuss two alternative semi-ring frameworks, based on *min+* and *SLO* (Soft constraint Lexicographic Ordering) semi-rings, respectively. In both cases, our computation of preferences and weights ensures information preserving and satisfies the cp-condition. We illustrate the construction of the SCSP using the example in Figure 2, which continues our running example from Figure 1.

Given a CP-net N, the corresponding SC-net N_c has two types of nodes: First, each feature $X \in N$ is represented in N_c by a node V_X that stands for a SCSP variable with $\mathcal{D}(V_X) = \mathcal{D}(X)$. Second, for each feature $X \in N$, such that $|Pa(X)| \geq 2$, we have a node $V_{Pa(X)} \in N_c$, with $\mathcal{D}(V_{Pa(X)}) = \Pi_{Y \in Pa(X)} \mathcal{D}(Y)$. Edges in N_c correspond to

[2] *Markov family* of a node X contains X, its parents and children, and the parents of its children.

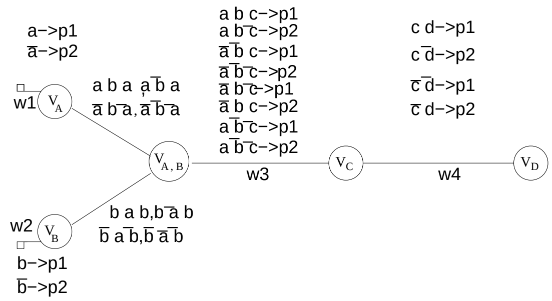

Figure 2: An SC-net.

hard and soft constraints, where the latter are annotated with weights. Each node V_X corresponding to an "independent feature" $X \in N$ has an incoming (source-less) soft constraint edge (e.g., see V_A and V_B). For each node V_X corresponding to a "single-parent" feature $X \in N$ with $Pa(X) = \{Y\}$, we have a soft constraint edge between X and Y (e.g., see V_D). Finally, for each node V_X such that $|Pa(X)| \geq 2$, we have (i) hard constraint edges between $V_{Pa(X)}$ and each $Y \in Pa(X)$ to ensure consistency (e.g., the edges between $V_{A,B}$ and both V_A and V_B), and (ii) a soft constraint edge between $V_{Pa(X)}$ and V_X (e.g., the edge between $V_{A,B}$ and V_C).

To assign preferences to variable assignments in each soft constraint, each soft constraint c (between $V_{Pa(X)}$ and V_X) is associated with two items: w_c, a real number which can be interpreted as a weight (will be defined in the next section), and $P_c = \{p_1, ..., p_{|\mathcal{D}(V_X)|}\}$, a set of reals which can be interepreted as "quantitative levels of preference". We will see in the next section how to generate the preference for each assignment to the variables of c, depending on the chosen semiring. In any case, each preference will be obtained by combining (via multiplication over naturals) the weight of the constraint w_c and one of the elements of P_c.

4.1 Weighted soft constraints

The weighted SCSP is based on the *min+* semi-ring $S_{WCSP} = \langle R_+, min, +, +\infty, 0 \rangle$. We assign preferences using real positive numbers (or penalties) and prefer assignments with smaller total penalty (i.e. the sum of all local penalties). In a soft constraint c on $V_{Pa(X)}$ and V_X, there are $|\mathcal{D}(V_X)|$ penalties. Without loss of generality, we assume they range between 0 and $|\mathcal{D}(V_X)| - 1$, that is, $p_1 = 0, ..., p_{\mathcal{D}(V_X)|} = |\mathcal{D}(V_X)| - 1$. In our example, since all variables are binary, there are only two penalties i.e., $p1 = 0$ and $p2 = 1$, in all the constraints.

To ensure the cp-condition, similar to [Boutilier *et al.*, 2001], we need to ensure that each variable dominates its children. We therefore set the minimum penalty on a variable to be greater than the sum of the maximum penalties of the children. In Figure 3 we show the pseudocode for the algorithm to compute the weights. In this code, $w(V_X)$ represents the weight of the soft constraint c between $V_{Pa(X)}$ and V_X.

Considering our example, let $\{D, C, B, A\}$ be the reverse topological ordering obtained in line 2. Therefore, the first soft constraint to be processed is the one between V_C and V_D. Since D has no children in N, in line 5 we assign $w(V_D)$ to

1. Next, we process the soft constraint between $V_{A,B}$ and V_C: V_D is the only child of V_C, hence $w(V_C) = w(V_D) \times \mathcal{D}(V_D) = 1 \times 2 = 2$. Subsequently, since V_C is the only child of both V_A and V_B, we assign $w(V_A) = w(V_B) = w(V_C) \times |\mathcal{D}(V_C)| = 2 \times 2 = 4$.

Now, consider two outcomes $o_1 = abcd$ and $o_2 = a\bar{b}cd$. The total penalty of o_1 is $(w(V_A) \times p_1) + (w(V_B) \times p_1) + (w(V_C) \times p_1) + (w(V_D) \times p_1) = 0$, since $p_1 = 0$, while the total penalty of o_2 is $(w(V_A) \times p_1) + (w(V_B) \times p_2) + (w(V_C) \times p_2) + (w(V_D) \times p_1) = (4 \times 1) + (2 \times 1) = 6$ since $p_2 = 1$. Therefore, we can conclude that o_1 is better than o_2 since $min(0, 6) = 0$.

We now prove that our algorithm for weight computation ensures the cp-condition on the resulting set of soft constraints, and this also implies preserving the ordering information with respect to the original CP-net.

Theorem 3 *The SC-net based weighted SCSP N_c, generated from an acyclic CP-net N, is an information preserving approximation of N, i.e. for each pair of outcomes α, β we have $\alpha \succ \beta \Rightarrow \alpha >_{min+} \beta$.*

Proof: Due to the CP-net semantics, it is enough to show that, for each variable $X \in N$, each assignment $\mathbf{u}$ on $Pa(X)$, and each pair of values $x_1, x_2 \in \mathcal{D}(X)$, if CP-net specifies that $u : x_1 \succ x_2$, then we have $x_1\mathbf{uy} >_{min+} x_2\mathbf{uy}$, for all assignments $\mathbf{y}$ on $\mathbf{Y} = \mathbf{V} - \{\{X\} \cup Pa(X)\}$. By definition, $x_1\mathbf{uy} >_{min+} x_2\mathbf{uy}$ iff $\sum_{s \in S} p'((x_1uy)_{|s}) < \sum_{s \in S} p'((x_2uy)_{|s})$, where S is the set of soft constraints of N_c and notation $(x_1uy)_{|s}$ stands for the projection on the outcome on constraint s. The constraints on which x_1uy differs from x_2uy are: constraint c on $V_{Pa(X)}$ and V_X, and all the constraints $t_i \in T$ on $V_{Pa(B_i)}$ and V_{B_i} such that $X \in Pa(B_i)$ (in what follows, we denote the children of X by $\mathcal{B} = \{V_{B_1}, \cdots, V_{B_h}\}$). Thus, we can rewrite the above inequality as $p'((x_1uy)_{|c}) + \sum_{t_i \in T} p'((x_1uy)_{|t_i}) < p'((x_2uy)_{|c}) + \sum_{t_i \in T} p'((x_2uy)_{|t_i})$ By construction of N_c we have $p'(\pi_c(x_1uy)_{|c}) = w_c \times p(x_1u) < p'((x_2uy)_{|c}) = w_c \times p(x_2u)$ and thus $x_1\mathbf{uy} >_{min+} x_2\mathbf{uy}$ iff $w_c p(x_2u) - w_c p(x_1u) > \sum_{t_i \in T} p'((x_1uy)_{|t_i}) - \sum_{t_i \in T} p'((x_2uy)_{|t_i})$ In particular, this will hold if $w_c(min_{x,x' \in \mathcal{D}(X)}|p(xu) - p(x'u)|) > \sum_{t_i \in T} w_{t_i}(max_{x,x',z,b}|p(x'zb) - p(xzb)|)$ where z is the assignment to all parents of $\mathcal{B}$ other than X. Observe , that the maximum in the right term is obtained when $p(x'zb) = |\mathcal{D}(\mathcal{B})| - 1$ and $p(xzb) = 0$. On the other hand, $min_{x,x' \in \mathcal{D}(X)}|p(x'u) - p(xu)| = 1$. In other words: $w_c > \sum_{t_i \in T} w_{t_i}(|\mathcal{D}(B_i)| - 1)$ must hold. But this is ensured by the algorithm, setting (in line 7) $w_c = \sum_{t_i \in T} w_{t_i}(|\mathcal{D}(B_i)|$. □

```
Input : Acyclic CP-net N
1. Construct the SC-net N_c without weights.
2. Order variables of N in a reverse topological ordering.
3. foreach X ∈ N do
4.     if X has no successors in N then
5.         w(V_X) = 1
6.     else
7.         w(V_X) = Σ_{Y s.t. X ∈ Pa(Y)} w(V_Y) · |D(V_Y)|
8. return N_c
```

Figure 3: Algorithm for weight computation.

Theorem 4 (Complexity) *Given an acyclic CP-net N with the node in-degree bounded by a constant, the construction of the corresponding SC-net based weighted SCSP N_c is polynomial in the size of N.*

Proof: If the CP-net has n nodes then the number of vertices V of the derived SC-net is at most $2n$. In fact, in the SC-net a node representing a feature appears at most once and there is at most one node representing its parents. If the number of edges of the CP-net is e, then the number of edges E in the SC-net (including hard and soft edges) is at most $e + n$, since each edge in the CP-net corresponds to at most one constraint, and each feature in the CP-net generates at most one new soft constraints. Topological sort can be performed in $O(V + E)$, that is, $O(2n + e + n) = O(e + n)$. Then, for each node, that is, $O(V)$ times, at most V children must be checked to compute the new weight value, leading to a number of checks which is $O(V^2) = O(n^2)$. Each check involves checking a number of assignments which is exponential in the number of parents of a node. Since we assume that the number of parents of a node is limited by a constant, this exponential is still a constant. Thus the total time complexity is $O(V^2)$ (or $O(n^2)$ if we consider the size of the CP-net). □

Let us compare in more details the original preference relation induced by the CP-net and this induced by its min+ semi-ring based SC-net. The comparison is summarized in the following table, where $\sim$ denotes incomparability. Notice that Theorem 3 shows that ordering information is preserved by the approximation.

CP-nets ⇒ min+		**min+** ⇒ CP-nets	
$\prec$	$<$	$<$	$\prec, \sim$
$\succ$	$>$	$>$	$\succ, \sim$
$\sim$	$<, >, =$	$=$	$=, \sim$

Since the min+ approximation is a total ordering, it is a linearization of the original partial ordering. In compensation, however, preferential comparison is now linear time.

4.2 SLO soft constraints

We also consider a different semi-ring to approximate CP-nets via soft constraints. The SLO c-semi-ring is defined as follows: $S_{SLO} = \langle A, max_s, min_s, \mathbf{MAX}, \mathbf{0} \rangle$, where A is the set of sequences of n integers from 0 to MAX, **MAX** is the sequence of n elements all equal to MAX, and **0** is the sequence of n elements all equal to 0. The additive operator, max_s and the multiplicative operator, min_s are defined as follows: given $s = s_1 \cdots s_n$ and $t = t_1 \cdots t_n$, $s_i = t_i, i = 1 \leq k$ and $s_{k+1} \neq t_{k+1}$, then $max_s(s, t) = s$ if $s_{k+1} \succ t_{k+1}$ else $max_s(s, t) = t$; on the contrary, $min_s(s, t) = s$ if $s_{k+1} \prec t_{k+1}$ else $min_s(s, t) = t$.

It is easy to show that S_{SLO} is a c-semi-ring and that the ordering induced by max_s on A is lexicographic ordering [Fargier *et al.*, 1993]. To model a CP-net as a soft constraint problem based on S_{SLO}, we set MAX equal to the cardinality of the largest domain - 1, and n equal to the number of soft constraints of the SC net. All the weights of the edges are set to 1. Considering the binary soft constraint on $Pa(X) =$

$\{U_1 \dots U_h\}$ and X, a tuple of assignments $(u_1, \dots, u_h, x)$ will be assigned, as preference, the sequence of n integers: $(MAX, MAX, \dots, MAX - i + 1, \dots, MAX)$. In this sequence, each element corresponds to a soft constraint. The element corresponding to the constraint on $Pa(X)$ and X is $MAX - i + 1$, where i is the distance from the top of the total order of the value x (i.e. we have a preference statement of the form $u : x_1 \succ x_2 \succ \dots x_i = x \succ x_{|D(X)|}$). In the example shown in Figure 2, all the preferences will be lists of four integers (0 and 1), where position i corresponds to constraint with weight w_i. For example, in constraint weighted w_3, $p_1 = (1,1,1,1)$ and $p_2 = (1,\underline{1},0,1)$. Given the pair of outcomes $o_1 = abcd$ and $o_2 = a\bar{b}cd$, the global preference associated with o_1 is $(1,1,1,1)$, since it does not violate any constraint, while the preference associated with o_2 is $min_S\{(1,1,1,1),(1,0,1,1),(1,1,0,1),(1,1,1,1)\} = (1,0,1,1)$. We can conclude that o_1 is better than o_2.

Similar to the comparison performed for min+ semi-ring, the following table compares the preference relation induced by the SLO semiring and that induced by the CP-net.

CP-nets $\Rightarrow$ SLO		**SLO** $\Rightarrow$ CP-nets	
$\prec$	$<$	$<$	$\prec, \sim$
$\succ$	$>$	$>$	$\succ, \sim$
$=$	$=$	$=$	$=$
$\sim$	$<, >$		

Note that the SLO model both preserves information and ensures the cp-condition. The proof of this is straightforward and is omitted due to lack of space. The SLO model, like the weighted model, is very useful to answer dominance queries as it inherits the linear complexity from its semi-ring structure. In addition, the sequences of integers show directly the "goodness" of an assignment, i.e. where it actually satisfies the preference and where it violates it.

4.3 Comparing the two approximations

Given an acyclic CP-net N, let N_c^{min+} and N_c^{SLO} stand for the corresponding min+ and SLO based SC-nets respectively. From the results in the previous section, we can see that pairs of outcomes ordered by N remain ordered the same way by both N_c^{min+} and N_c^{SLO}. On the other hand, pairs of outcomes incomparable in N are distributed among the three possibilities (equal or ordered in one the two ways) in N_c^{min+}, while being strictly ordered by N_c^{SLO}. Therefore, the (total) preference relation induced by N_c^{min+} is a less brutal linearization of the partial preference relation induced by N, compared to that induced by N_c^{SLO}. Mapping incomparability onto equality might seem more reasonable than mapping it onto an arbitrary strict ordering, since the choice is still left to the user. We might conclude that the min+ model is to be preferred to the SLO model, as far as approximation is concerned. However, maximizing the minimum reward, as in any fuzzy framework [Schiex, 1992], has proved its usefulness in problem representation. The user may therefore need to balance the linearization of the order and the suitability of the representation provided.

5 Future Work

We plan to use our approach in a preference elicitation system in which we guarantee the consistency of the user preferences, and guide the user to a consistent scenario. Morover, we also plan to exploit the use of partially ordered preferences, as allowed in soft constraints, to better approximate CP nets. Finally, we intend to use machine learning techniques to learn conditional preferences from comparisons of complete assignments.

References

[Bistarelli *et al.*, 1996] S. Bistarelli, H. Fargier, U. Montanari, F. Ros si, T. Schiex, and G. Verfaillie. Semiring-based CSPs and valued CSPs: Basic properties and comparison. In *Over-Constrained Systems*. 1996.

[Bistarelli *et al.*, 1997] S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based Constraint Solving and Optimization. *Journal of the ACM*, 44(2):201–236, 1997.

[Boutilier *et al.*, 1999] C. Boutilier, R. Brafman, H. Hoos, and D. Poole. Reasoning with Conditional Ceteris Paribus Preference Statements. In *Proc. of UAI-99*, 1999.

[Boutilier *et al.*, 2001] C. Boutilier, F. Bacchus, and R. I. Brafman. UCP-Networks: A Directed Graphical Representation of Conditional Utilities. In *Proc. of UAI-01*, 2001.

[Boutilier *et al.*, 2002] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: A Tool for Representing and Reasoning about Conditional *Ceteris Paribus* Preference Statements. *submitted for publication*, 2002.

[Domshlak and Brafman, 2002] C. Domshlak and R. Brafman. CP-nets - Reasoning and Consistency Testing. In *Proc. of KR-02*, pages 121–132, 2002.

[Doyle and Thomason, 1999] J. Doyle and R. H. Thomason. Background to Qualitative Decision Theory. *AI Magazine*, 20(2):55–68, 1999.

[Doyle and Wellman, 1994] J. Doyle and M. Wellman. Representing Preferences as Ceteris Paribus Comparatives. In *Proc. AAAI Spring Symposium on Decision-Making Planning*, pages 69–75, 1994.

[Fargier *et al.*, 1993] H. Fargier, J. Lang, and T. Schiex. Selecting preferred solutions in fuzzy constraint satisfaction problems. In *Proc. 1st European Congress on Fuzzy and Intelligent Technologies (EUFIT)*, 1993.

[Hansson, 2001] S. O. Hansson. Preference Logic. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume 4, pages 319–394. Kluwer, 2001.

[Lang, 2002] J. Lang. From Preference Representation to Combinatorial Vote. In *Proc. of KR-02*, 2002.

[McGeachie and Doyle, 2002] M. McGeachie and J. Doyle. Efficient utility functions for ceteris paribus preferences. In *Proc. AAAI 2002*, pages 279–284, 2002.

[Sabin and Weigel, 1998] D. Sabin and R. Weigel. Product Configuration Frameworks - A Survey. *IEEE Intelligent Systems and their Applications*, 13(4):42–49, 1998.

[Schiex *et al.*, 1995] T. Schiex, H. Fargier, and G. Verfaille. Valued Constraint Satisfaction Problems: Hard and Easy Problems. In *Proc. of IJCAI*, pages 631–637, 1995.

[Schiex, 1992] T. Schiex. Possibilistic constraint satisfaction problems, or "How to handle soft constraints?". In *Proc. of UAI*, pages 269–275, 1992.