

# NP-Completeness of Outcome Optimization for Partial CP-nets

Keith Purrington and Edmund H. Durfee

Computer Science and Engineering  
University of Michigan  
Ann Arbor, Michigan 48109 USA  
purringk@umich.edu, durfee@umich.edu

## Abstract

Partial CP-nets extend the standard CP-nets framework to allow users to express indifference about the values of some variables. Prior work has shown that linear-time *forward sweep* algorithms for outcome optimization in classic CP-nets can also be used with partial CP-nets when there is no evidence that fixes the values of one or more variables. In this paper we prove that, in the more general case where evidence is present, outcome optimization in partial CP-nets is, unfortunately, NP-complete. Fortunately, we are able to describe a polynomial-time *backward sweep* algorithm that finds optimal outcomes for a specialized class of partial CP-net structures (to be defined). Furthermore, by comparison to a GSAT-style random-walk algorithm and an exhaustive search, we empirically demonstrate that the backward sweep algorithm finds approximately optimal outcomes for general partial CP-nets and is faster by several orders of magnitude.

## Introduction

CP-nets (Boutilier, *et al.* 2004a) are a useful representation of users' preferences in part because they facilitate preference elicitation without excessively burdening users. However, for scheduling problems where a user's preferences might be "I prefer a morning meeting to an afternoon meeting," it can be cumbersome to require the user to express distinct preferences over all possible meeting times. Instead, it is more natural if a user can say "I am indifferent between meeting at 9, 10, or 11 AM."

Partial CP-nets (Rossi, *et al.* 2004) address this concern by allowing the presence of *unranked variables* (variables for which the user expresses no preference) in a CP-net. This addresses the representational issue, but complicates the optimization task. Rossi *et al.* (2004) demonstrate that the linear-time forward sweep algorithm for finding optimal outcomes in CP-nets translates directly to partial CP-nets when the agent is able to control the value of every variable. In general, however, external *evidence* might determine the value of one or more variables, so that those variables are not settable by the agent. (The term *evidence* is borrowed from the Bayes-net literature.)

---

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

When evidence is present, the problem becomes much

harder. We prove that for even a very restricted class of partial CP-networks, the problem of outcome optimization in the presence of evidence is NP-complete.

The remainder of the paper is structured as follows. We first provide a brief overview of the CP-nets formalism. For a fuller description, we encourage consulting Boutilier *et al.* (2004a). We then summarize the partial CP-nets extension introduced by Rossi *et al.* (2004), and demonstrate why the standard forward sweep is insufficient when some evidence is present, proving the NP-completeness of outcome optimization in this case. Although this result is somewhat limiting, we describe an algorithm based on the classic forward sweep that notably includes a *backward sweep* phase that uses evidence to determine the values for unranked variables. We describe the restricted class of partial CP-net structures for which this algorithm is provably optimal. Finally, we compare the performance on random CP-nets of this backward sweep algorithm with an exhaustive outcome-space search and with a random-walk algorithm that is similar to the well-known GSAT SAT-solver (Selman, *et al.* 1992), and find that even when partial CP-nets lack the special structure that is necessary to guarantee optimality, the backward sweep algorithm remains a good approximation.

## CP-nets Overview

Boutilier *et al.* (2004a) present a model (called a CP-net, short for *Conditional Preference Network*) that describes an agent's conditional *ceteris paribus* preferences. Briefly, a CP-net is a directed graph, in which each node is annotated with a conditional preference (CP) table that indicates the preferred value for that node's variable, conditioned on the values of its parent variables (the nodes that point to it). In this way, CP-nets are analogous to belief networks (Kim and Pearl, 1983), but express conditional preference instead of conditional probability.

Figure 1 shows a simple CP-net for Boutilier's example of a diner. We have two variables, and name them with capital letters: *D* for *DinnerIsFish* and *W* for *WineIsWhite*. Lowercase letters (and their negations) denote values that are assumed by variables. Thus, *d* and *d'* represent,

respectively, fish and non-fish (meat) dinner, while  $w$  and  $w'$  stand for white and non-white (red) wine. For simplicity, we assume throughout that variables are binary.

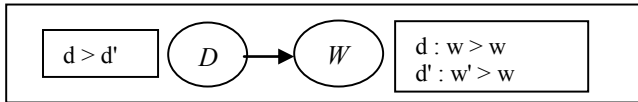


Figure 1. The CP-net for the diner example

### Outcome Optimization

Boutilier *et al.* (2004a) describe how, given some (possibly empty) set of evidence, the most-preferred outcome that is consistent with that evidence can be found in time that is proportional to the number of variables in the CP-net. The algorithm for doing this, called a *forward sweep*, is very straightforward: proceeding in (any) topological order, each variable is assigned the value that is most preferable given the existing assignment(s) to its parent(s). The semantics of the CP-net representation guarantee that the forward sweep constructs an outcome that is preferable to any other outcome that is consistent with the evidence.

### The Induced Preference Graph

CP-nets induce associated preference graphs that partially order outcomes. In the induced preference graph for a CP-net, there is a directed edge between every pair of outcomes that differ on the value of only a single variable. For any pair of outcomes  $O_1$  and  $O_2$  that differ only on the value of a single variable  $V$ , the CP-table for  $V$  indicates which of the competing values of  $V$  is preferred. Under the *ceteris paribus* semantics of the CP-net, the outcome in which  $V$  assumes a preferred value is the preferred outcome, all else being equal. By convention, we draw edges from more preferable outcomes to less preferable.

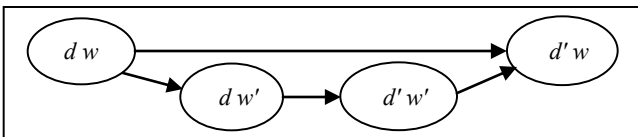


Figure 2. Induced preference graph for diner example

In Figure 2, the induced preference graph for the diner example, we observe that  $(d w) > (d w') > (d' w') > (d' w)$ . We rank  $(d, w')$  over  $(d, w)$  since  $w' > w$  when  $D$  has the value  $d$ . Similarly, we rank  $(d, w)$  over  $(d', w)$  because when  $W$  has the value  $w$ , then  $d$  is preferable to  $d'$ . Finally,  $(d', w) > (d', w')$ , because  $w > w'$  when  $D$  takes the value  $d'$ . The final relationship, between  $(d, w')$  and  $(d', w')$ , can be directly established based on the differing values of  $D$ , and also can be inferred as a transitive relationship.

One consequence of the definition of the induced preference graph is that every CP-net has a single best outcome that corresponds to the node in the graph for which changing any variable value worsens the outcome. This makes certain natural preference relationships inexpressible, as the following example shows.

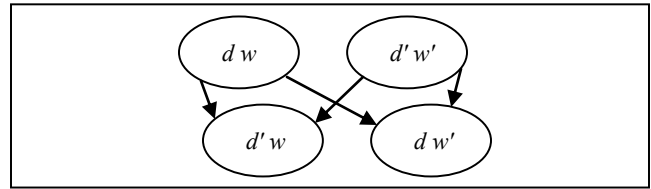


Figure 3. Preference graph with two "best" outcomes

Assume that the diner has no particular preference for either the choice of food or of wine, but instead cares only that they go well together. The preference graph in Figure 3 shows this preference: the outcomes in which the food matches the wine are better than those in which there is no match, but there is no preference between the outcomes in which there is a match. This outcome graph has two different "best" outcomes, and therefore, cannot be induced by a standard (acyclic) CP-net. One way to express this preference is to explicitly model the diner's indifference over the particular choice of food and of wine.

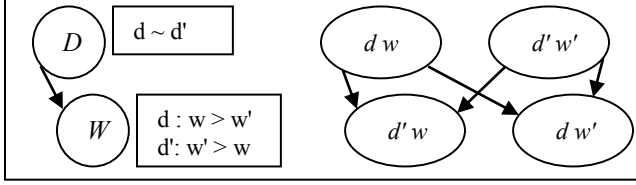
### Partial CP-nets

Partial CP-nets were originally devised by Rossi *et al.* (2004) as a way to allow the preferences that one agent has to depend on the value of a variable that is important only to some other agent. The introduction of *unranked variables* has the main consequence of expanding the ways in which two outcomes that differ by just one variable value can be related. As in traditional CP-nets, when two outcomes differ on the value of exactly one variable, then (since all else is equal) the outcome where that variable has its preferred value is preferred to the other outcome. If however, the differing variable is unranked, then preference over the outcomes is not so easily determined: even though neither value is directly preferred, one value might match better with the values of the variable's children. For an unranked node  $U$ , Rossi *et al.* (2004) call a change to its value is *worsening* if it makes the values for all variables that depend on  $U$  less preferred. Similarly, a change called *indifferent* if it results in the value for each dependent variable being just as good. Finally, *incomparable* changes are those that are neither worsening nor indifferent.

In addition to defining the semantics of partial CP-nets, Rossi *et al.* (2004) show that, in cases when no evidence is present, outcome optimization can still be done with a linear forward sweep by choosing values (arbitrarily) for unranked nodes as they are encountered, and then assigning the downstream variables with a standard forward sweep. In the more general case, when evidence fixes the values of some variables, such a forward sweep is no longer guaranteed to be successful.

As an example of this, consider Figure 4, which shows a very simple partial CP-net and its induced outcome graph (which is reproduced from Figure 3). This network

describes someone's preferences for a food-wine pairing. For this individual, the particular choice of food is unimportant; her preference is rather that the food and wine go well together. This can be represented in several ways; here, the diner is indifferent as to choice of food, but expresses a preference for the type of wine that depends on what entrée is chosen.



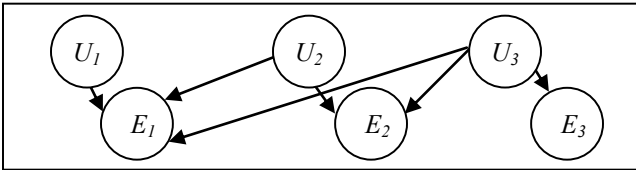
**Figure 4. Partial CP-net for indifferent diner**

Provided that the diner can choose all of the variable values, the procedure described by Rossi *et al.* (2004) finds one of the most preferred meals. It can simply choose a value for  $D$ , and no matter the choice, the CP-table for  $W$  provides a value that ensures a successful pairing. If, however, the restaurant is out of white wine (fixing the value of  $W$  to be  $w'$ ), the forward sweep is no longer guaranteed to find the optimal outcome. When the value of  $w'$  is fixed, the diner can still be maximally satisfied, but to do so with a forward sweep requires having made the right "guess" and choosing red meat before finding out that the restaurant is out of white wine. To avoid the need for omniscience, a general optimization algorithm must allow for backtracking.

This small example can be solved with a backward sweep that uses the evidence to determine the proper value for the unranked variable. However, as the CP-net structure becomes more complex, a backward sweep would encounter the same need for backtracking as needed by a forward sweep. In fact, we show that even for a very restricted class of problems, making assignments to the unranked nodes so as to ensure that the value of each evidence node is preferred is NP-complete.

### NP-Completeness of Outcome Optimization

In this section, we demonstrate that outcome optimization when some evidence is present is NP-complete for a very specialized class of partial CP-nets.



**Figure 5. Example CP-net showing bi-partite structure**

We consider partial CP-nets, as in Figure 5, that have a bipartite graph structure  $\langle U, E, D \rangle$ , where  $U$  is a set of unranked nodes,  $E$  is a set of evidence nodes, and  $D$  is a set of directed edges from nodes in  $U$  to nodes in  $E$ . (We use

"node" and "variable" interchangeably.) Finally, we restrict the variables in these specialized networks to be binary.

**Theorem 1.** *Outcome optimization in partial CP-nets in the presence of evidence is NP-complete.*

*Proof:* As is standard, we establish NP-completeness by separately demonstrating that outcome optimization is both NP-hard and contained in NP, using the following two lemmas.

**Lemma 1.** *Outcome optimization is NP-hard.*

*Proof:* We show hardness by reducing 3SAT to the problem of finding the optimal outcome in a bipartite graph as we have defined. We observe that the evidence nodes are analogous to the clauses of a SAT instance: we say that an evidence variable is *satisfied* when its parents are chosen in such a way that the value of the evidence is preferred by the agent. By mapping 3SAT clauses to evidence nodes and carefully choosing the parent nodes and the CP-tables for the evidence nodes, we ensure that each evidence variable is satisfied IFF the 3SAT clause is satisfied.

Consider a particular 3SAT clause, say  $(U_1 \text{ OR } U_2 \text{ OR } U_3)$ . To translate this to our CP-net, we create an evidence node  $E$ , with three parents  $U_1$ ,  $U_2$ , and  $U_3$ . To construct the CP-table for  $E$ , say *w.l.o.g.* that a value of  $e$  (rather than  $e'$ ) satisfies the node. Then, because  $E$  has three binary-valued parents, there are eight entries in its CP-table. We fill in the table so that  $e$  is preferred to  $e'$  in exactly the rows that correspond to satisfying assignments to the SAT clause. In fact, for each evidence node, seven of the eight possible parent assignments will satisfy that evidence. In this case, only for  $(u_1', u_2', u_3)$  do we set  $e'$  preferable to  $e$ .

The general reduction, then, is a 3-step procedure. First, we create one unranked node for each variable in the SAT instance, and one evidence node for each clause. Second, for each clause  $C_i$  of the 3SAT instance, which contains variables  $X_j$ ,  $X_k$ , and  $X_m$ , we connect the corresponding unranked nodes  $U_j$ ,  $U_k$ , and  $U_m$  to the corresponding evidence node  $E_i$ . Finally, for each evidence node  $E_i$ , we construct the CP-table so that  $e_i$  is preferred to  $e_i'$  in the seven cases that would satisfy the corresponding SAT clause, as in the example above.

Then, for a network constructed in this way, an assignment to the unranked nodes that makes the value  $e_i$  conditionally preferred to  $e_i'$  for each evidence node is also a satisfying assignment for the 3SAT instance. Likewise, if no assignment can simultaneously satisfy each evidence node, then the original 3SAT instance is unsatisfiable as well. Thus, this partial CP-net outcome optimization in the presence of evidence can be no easier than 3SAT, and must be NP-hard.

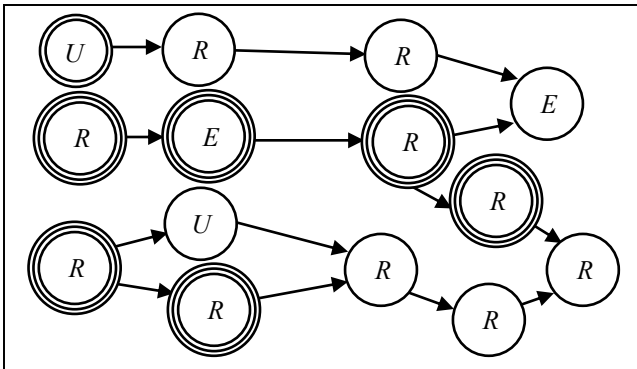
**Lemma 2.** *Outcome optimization is contained in NP.*

*Proof:* A solution certificate or "witness" can be produced by nondeterministically choosing an assignment to the unranked nodes. Then, this certificate can be checked to see if it simultaneously satisfies the evidence nodes in polynomial time (with respect to the number of evidence nodes) by inspecting the CP-table for each evidence node.

## The Backward Sweep Algorithm

While outcome optimization in partial CP-nets is hard to solve in full generality, it can be done efficiently in special cases. As we observed earlier with the simple diner example, a backward sweep that uses the evidence to decide on values for the unranked variables can sometimes be effective. In fact, we identify an entire class of network structures for which such a linear-time algorithm will find an optimal outcome.

Our backward sweep algorithm is actually a three-phase algorithm that consists of two (partial) forward sweeps surrounding a backward sweep phase that moves from evidence nodes to unranked nodes. In contrast to the bipartite networks from the previous section, we now consider completely arbitrary CP-nets in which ranked, unranked, and evidence nodes can all be interconnected. We continue to assume that all variables are binary.



**Figure 6.** Backward sweep first assigns triply-circled nodes, then doubly-circled, then singly-circled

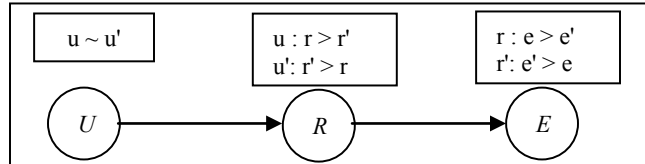
The first phase of the algorithm is a classic CP-nets forward sweep, beginning with all the unconditional (parentless) variables. This phase only assigns values to ranked variables whose ancestors are all also ranked and/or evidence variables. The algorithm does not yet assign values to unranked variables, and thus, by definition cannot assign a value to any descendants of those variables. Finally, each evidence node encountered in this pass (those with only ranked ancestors) becomes marked. Since marked evidence nodes aren't influenced by unranked variables, they can safely be ignored in the next phase of the algorithm.

After this first phase of the algorithm, the nodes of the CP-

net that remain unassigned are unranked nodes and nodes descended from unranked nodes, terminating at either evidence nodes or graph leaves. Figure 6 shows an example network after this phase. The node labels distinguish unranked variables (U), ranked variables (R), and evidence variables (E). The triply-circled nodes are those that were assigned a value (or marked as seen, in the case of evidence nodes) in the first phase.

The second phase of the algorithm departs from the algorithm proposed by Rossi *et al.* (2004), and is the operation that provides the name backward sweep. In this phase, the algorithm begins at the unmarked evidence nodes and works backwards to their ancestral unranked nodes. The CP-table for each evidence node describes the combinations of assignments to its parents that result in its being satisfied. Then, this process can be repeated for those parent variables, until eventually encountering an unranked node. In this way, values can be chosen for unranked nodes that will satisfy their descendent evidence nodes.

Figure 7 shows a simple example of this procedure. Assume evidence specifies that the value of  $E$  is  $e$ . Then, this value is preferred when  $R$  has the value of  $r$ . Similarly  $r$  is preferred to  $r'$  when  $U$  has the value of  $u$ . Thus, we find that the outcome  $(u r e)$  satisfies both the intermediate variable and the evidence and is an optimal assignment. Most importantly, we have identified that  $u$  is the proper setting for  $U$  to ensure optimality. We will shortly define the conditions that must hold for a partial CP-net to guarantee that the procedure described here will be effective.



**Figure 7.** Partial CP-net solvable by a backward sweep

In Figure 6, the second phase finds a value for the doubly-circled node in the upper left. After the second phase, unranked nodes without any descendent evidence nodes remain unassigned, as do the ranked nodes that have unranked ancestors. (These are the singly-circled nodes in Figure 6.) Since all the evidence variables have been considered, the algorithm can finally assign arbitrary values to any remaining unranked variables just as described by Rossi *et al.* (2004). A final forward sweep phase then assigns the ranked descendants of the unranked nodes, completing the algorithm.

## Properties Sufficient for Correctness

As illustrated by the proof of NP-completeness, a backward-sweep is not in general guaranteed to make the right assignments, but as shown in the Figure 6 example, in some cases it can. Each evidence node determines the values that all of its parents must take if it is to be satisfied. In turn, for those parent variables to also be satisfied, the

algorithm must be able to independently assign *their* parents' values. And so on. In order to guarantee that each node's parents can be assigned values that will result in its being satisfied, the algorithm requires that two properties hold. First, it must be the case that each unranked node begins at most one sub-path that ends in an evidence node. This ensures that in the backward sweep phase of the algorithm, no ancestor of an evidence node has multiple influences on its "correct" value that might end up in conflict.

The second property concerns the CP-tables of every evidence node and every ranked node that is on the path backward from an evidence node to an unranked node. For each such node  $X$ , divide its parents  $Pa(X)$  into two sets,  $PR(X)$  and  $PU(X)$ ,  $Pa(X) = PR(X) \cup PU(X)$ , where  $PR(X)$  are those parents with no unranked ancestor, and  $PU(X)$  are those parents that have an unranked ancestor. Notably, the variables in  $PR(X)$  are assigned values in the initial forward sweep phase of the algorithm. Then, the desired property is that for any assignment to  $PR(X)$ , there must be some assignment to  $PU(X)$  that makes  $x$  preferred to  $x'$  and some assignment to  $PU(X)$  that makes  $x'$  preferred to  $x$ . This property insures that no matter how the first forward sweep phase assigns variables, the backward sweep phase will be able to find a satisfying assignment to the parents at each backward step.

When these properties hold, our three-phase backward sweep algorithm provably finds an optimal outcome. (The proof follows directly from the specified properties, but space limits preclude a full formalization of the proof here.) Furthermore, the time required for the algorithm remains linear in the number of variables. In addition, even when specified properties do not hold, the algorithm continues to give useful, approximately optimal outcomes, as the next section shows.

### Empirical Performance

The close correspondence between our problem of outcome optimization and Boolean satisfiability provides us with a large number of off-the shelf tools against which to compare the performance of the backward sweep algorithm. GSAT is one such well-known algorithm that finds SAT solutions through greedy local search combined with random restarting.

We implemented a GSAT-style random walk that performs local search over the possible configurations of the unranked nodes. Each step constitutes the flip of one such node's value; these flips are chosen greedily to maximize the number of satisfied evidence nodes. The initial configuration of the unranked nodes was chosen at random. We allowed the algorithm to run for 100 steps before restarting, and performed 20 random restarts per trial.

We generated sample problems using the Bayes-net generator from the University of Sao Paulo Decision Making Lab (available at <http://www.pmr.poli.usp.br/ltd/>). We generated five random 50-node networks, and then for 20 trials, we chose nodes at random to be unranked and to be set as evidence, resulting in a total of 100 experiments for each ratio of unranked to evidence nodes. We then compared the performance of the backward sweep with GSAT on such problems. In addition, in the case with 15 unranked nodes, we compared our results to a complete exhaustive search. With 25 unranked nodes, time limitations became an issue, but we did run seven iterations of the complete search to verify that the approximation algorithms continue to be nearly optimal. The results are summarized below.

	Average Satisfied Evidence Nodes	Average Time (sec)
Backward Sweep	10.27	0.005
GSAT	9.43	37.65
Complete Search	10.62	6.5

**15 evidence nodes, 15 unranked, 20 ranked**

	Average Satisfied Evidence Nodes	Average Time (sec)
Backward Sweep	9.55	0.003
GSAT	8.4	56.85
Complete Search	10.04	5775

**12 evidence nodes, 25 unranked, 13 ranked**

	Average Satisfied Evidence Nodes	Average Time (sec)
Backward Sweep	6.25	0.003
GSAT	6.15	69.5

**8 evidence nodes, 32 unranked, 10 ranked**

The quality of the solutions found by the competing approximation algorithms turns out to be quite similar, with the backward sweep holding a small advantage. Although both algorithms are incomplete, for the problems with 15 unranked nodes, their performance is nearly identical to that of an exhaustive search. Similarly, the few experiments with 25 unranked nodes show that the approximations remain good ones, although this is less conclusive because of the small number of data point available. Finally, as expected from a linear algorithm, the backward sweep enjoys a substantial advantage in running time that makes it appear to be a good choice as an approximation algorithm.

### Related and Future Work

Expressing preferences in a way that permits indifference is certainly not a new idea. Formal treatment of indifference in axiomatic utility theory dates back at least to Auman (1962). Our work retains the usual distinctions that separate CP-nets from classical utility theory,

including naturalness of expression, ease of elicitation, and the ability to leverage the graphical structure to perform certain reasoning tasks efficiently.

Our backward sweep algorithm is provably optimal for certain restricted network topologies. This design was inspired by similar ideas that are common to belief networks. In particular, the Kim and Pearl (1983) work describes an inference algorithm that achieves efficiency by being restricted to operate only on polytrees, even though such reasoning is intractable for arbitrary networks.

External evidence can be viewed as a hard constraint placed on the preferential optimization problem. In this respect, our work bears some resemblance to the problem of CP-nets-based constrained optimization addressed in Boutilier *et al.*, (2004b). However, the constraints considered in that work are richer than simply fixing the values of some variables, and the resulting optimization problem becomes quite different. With the addition of richer constraints, the problem naturally takes on the character (and complexity) of traditional constraint processing, and solution techniques involve some variety of pruned backtracking search.

The CP-nets considered in this paper are all assumed to be acyclic. Permitting cycles is another way to provide greater representational power; for example, the outcome graph in Figure 4 can be induced by a cyclic CP-net instead of a partial one. However, Domshlak and Brafman (2002) show that even consistency checking is NP-hard when a CP-net may have cycles. To our knowledge, categorizing the representational power of either cyclic or acyclic CP-networks remains an open problem. The space of representable preference functions has yet to be fully understood.

Despite the theoretical NP-completeness of outcome optimization in the face of indifference, the backward-sweep algorithm offers performance that is more than adequate for our eventual problem domain. We are looking at modeling hybrid (finite-domain and temporal) preferences using CP-nets. While indifference can be a useful tool in representing preferences for finite-domain variables, it is especially valuable when describing temporal preferences. In this context, the use of indifference empowers users to choose the proper level of granularity at which to specify their temporal preferences.

As the model evolves, the modeling of indifference will need to become more sophisticated. In this paper, we follow Rossi *et al.* (2004) in restricting variables to be either totally ranked or totally unranked. In practice, users will often have preferences for, say, a scheduling task, that violate this assumption. For example, the user who prefers an afternoon meeting to one in the morning (with no further preference over the time of the meeting) cannot be modeled in the current formalism.

## Conclusion

In this paper, we have shown that the combination of indifference and evidence in CP-nets makes outcome optimization intractable in general. Because we are faced with these kinds of problems in our application domain, we have developed approaches for handling such cases. In particular, we have defined a polynomial-time algorithm for networks with a particular restricted structure. In addition, we have empirically evaluated our backward sweep algorithm in comparison to both efficient SAT-solving tools and complete exhaustive search and established that algorithm quickly finds approximately optimal solutions even in the unrestricted case.

## Acknowledgments

The work reported in this paper was supported, in part, by the National Science Foundation under grant IIS-0534280. We would like to thank Jim Boerkoel and the anonymous reviewers for valuable feedback on this work.

## References

- Auman, R. J. 1962. "Utility Theory without the Completeness Axiom." *Econometrica*, Vol. 30, No. 3 (Jul., 1962), 445-462
- Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., and Poole, D. 2004. "CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements." *Journal of Artificial Intelligence Research (JAIR)* 21 (2004), 135-191
- Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., and Poole, D. 2004. "Preference-based Constrained Optimization with CP-nets." *Computational Intelligence* 20(2), 137-157
- Domshlak, C., and Brafman, R. I. 2002. "CP-Nets - Reasoning and Consistency Testing." *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning*, 121-132
- Kim, J. H. and Pearl, J. 1983. "A Computational Model for Combined Causal and Diagnostic Reasoning in Inference Systems." *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83)*, 190-193
- Rossi, F., Venable, K.B., and Walsh, T. 2004. "mCP Nets: Representing and Reasoning with Preferences of Multiple Agents." *Proceedings of the 19th Conference on Artificial Intelligence (AAAI-04)*, LNCS 749, 729-734
- Selman, B., Levesque, H. J., and Mitchell, D. 1992. "A New Method for Solving Hard Satisfiability Problems." *Procs. of the 10<sup>th</sup> National Conf. on Artificial Intelligence (AAAI-92)*, 440-447