Preferences and Domination

Judy Goldsmith[1]

University of Kentucky, Dept. of Computer Science
773 Anderson Hall, Lexington, KY, 40506-0046, USA
goldsmit@cs.uky.edu

Dagstuhl Seminar Proceedings 04421
Algebraic Methods in Computational Complexity
http://drops.dagstuhl.de/opus/volltexte/2005/103

Preferences and Domination

Judy Goldsmith

University of Kentucky, Dept. of Computer Science
773 Anderson Hall, Lexington, KY, 40506-0046, USA
goldsmit@cs.uky.edu

Abstract. We show that the dominance problem for CP-nets is P-hard, and that the dominance problem for the more general case of *cyclic* CP-nets is PSPACE complete.

Keywords. Qualitative preferences, CP-nets, complexity, PSPACE completeness

1 Introduction

The problems of representing and eliciting user preferences over a multi-attribute domain arise in e-commerce, planning, and a host of other fields. Preferences can be represented explicitly by a ranked ordering of domain instances, but this list will be of length exponential in the number of attributes. Preferences can sometimes be represented logarithmically more succinctly by describing preferences on individual attributes, or on small sets of attributes.

CP-nets [1] provide a qualitative representation of preferences. They allow a user to order values of an attribute, *all other things being equal (ceteris paribus)*, or to specify the attributes on which preference for a particular attribute's values depends. CP-nets (formal definitions given in Section 3) are attractive because they allow for succinct representations, and because they suggest a simple and easily-understood elicitation process. Acyclic CP-nets are seemingly computationally easy to reason with.

Unfortunately, many preferences about which we would like to reason can only be expressed in a way that creates a cycle of attribute dependencies. It is possible to have a cyclic CP-net that is *consistent* (i.e., no instance is preferable to itself), but it seems that consistency is computationally difficult to check.

Another computational question of interest in CP-nets is, given two instances α and β, which one is preferred. This is called the *dominance problem.* It is possible to show that α is preferred to β ($\alpha \succ \beta$) by exhibiting a sequence of instances α_i with $\beta = \alpha_0$ and $\alpha_T = \alpha$, and each $\alpha_i \succ \alpha_{i+1}$ explicitly from the CP-net. Then each pair α_i, α_{i+1} differs in exactly one attribute. The sequence $\langle \alpha_i \rangle_{i \leq T}$ is called an *improving flipping sequence.*

If it can be shown that all improving flipping sequences have length polynomial in the number of attributes, then the dominance problem is in NP. Boutilier, et al. [1] showed that the dominance problem for acyclic CP-nets is in P for some very restricted classes of CP-nets, and is NP-complete for other restricted classes. However, the complexity of the general problem, either for acyclic or cyclic CP-nets, was only conjectured to be PSPACE-complete.

We show here that the dominance problem for cyclic CP-nets with incomplete tables is indeed PSPACE-complete. This work was done independently and simultaneously by Jérôme Lang [2]. Lang's original proof used inconsistent preferences; Nic Wilson showed that there was a reduction to CP-nets with consistent preferences [3]. The problem of incomplete tables was resolved by Truszczynski [4]: For any cyclic CP-net with incomplete tables, there is an equivalent one with complete tables. In particular, there is a dominance-preserving reduction from the dominance problem for CP-nets with incomplete tables to the dominance problem for CP-nets with complete tables.

2 Example Networks

Consider the possibility that there are many saunas available, and we wish to decide to which sauna we should bring a guest named Bill[1]. We wish to represent Bill's preferences with respect to the attributes of RESTRICTIONS (single-sex vs. mixed), TYPE (steam vs. dry), BUSYNESS (crowded vs. not crowded).

Bill tells us that he prefers uncrowded saunas to crowded ones; that he prefers dry to steam, and that his preferences with respect to restrictions depend on crowdedness: If the sauna is crowded or steamy, he prefers mixed-sex saunas, otherwise he prefers single-sex saunas. Figure 1 shows the underlying dependency graph for Bill's sauna preferences. A complete specification of the network would also include the (conditional) preference tables.

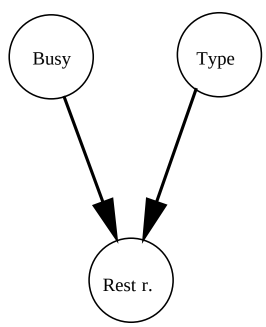

Fig. 1. Sauna preferences

Next, consider Judy's preferences about music. The attributes include GENRE, VOCALS, MOOD, and VOLUME. We learn that she prefers to be in a good mood,

[1] The preferences expressed in this section are purely hypothetical, and do not represent the preferences of particular people.

and then she prefers folk music to rock. But when she is in a bad mood, she prefers rock to folk. Whether she prefers vocals depends on her mood and the genre. Volume preferences depend on genre as well. Folk music is preferred soft and rock is preferred loud.

What distinguishes this network from the sauna network is that there is a cycle in the dependencies: The preference on volume depends on mood, yet loud music improves a bad mood. We represent this by an arc from volume to mood (a dotted line in Figure 2).

Note that this introduces two new features to the network: cycles in the underlying directed graph, and an incomplete preference table. However, it can be shown that this particular network is consistent. In other words, no instance is preferred to itself; There are no cycles in the implicitly represented preference graph.

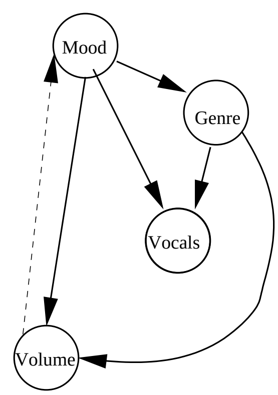

Fig. 2. Music preferences

3 Definitions

Definition 1 *A CP-net consists of a directed graph $G = \langle V, E\rangle$, where V is the set of preference attributes and an edge between attributes indicates dependence, and a set of conditional preference tables. Each $v \in V$ has a domain D_v of possible values, and a corresponding preference table. Each row of the preference table for v is labeled by values of v's parent attributes. Each row specifies a preference order on the values of D_v.*

If each preference table has a row for each set of values of the parent nodes, and if each row specifies a complete linear order on the values D_v, then we say that the CP-net has complete tables.

An instance of a CP-net is a setting of values for each node in the CP graph.

Consider two instances α and β which differ only on attribute v. We say that there is an *improving flip* from α to β if the value of v in α is preferred to the value of v in β, given the values of the parents of v in both α and β. A sequence $\alpha_0 \dots \alpha_T$ is an *improving flipping sequence* if for each $i < T$, there is an improving flip from α_i to α_{i+1}.

We say that α is preferred to β ($\alpha \succ \beta$) if there is an improving flipping sequence $\beta = \alpha_0 \dots \alpha_T = \alpha$.

Note that a self loop on a node in the CP graph makes the definition of an improving flip unclear. If v is parent to itself, then two instances that differ only on v do not agree on the parents of v. Therefore, we disallow self loops in our definition of CP-nets.

The CP-net specifies an ordering on instances. This can be expanded to the *preference graph*. The nodes of the preference graph are instances, and there is a directed edge from β to α if and only if there is an improving flip from β to α. Note that the number of nodes in the preference graph is exponential in the number of attributes (with base depending on the size of the attribute domains). Thus, it is conceivable that there could be exponentially long paths in the graph. In particular, it is conceivable that there may be exponentially long minimum-length paths between nodes. The ramifications of this observation are considered in Section 4.2.

4 Complexity Results

We begin with a proof that the dominance problem is P-hard for CP-nets. This holds even for the resticted class of acyclic CP-nets. The result is not surprising, but the proof sets up the proof that the dominance problem for cyclic CP-nets is PSPACE hard.

4.1 P-Hardness

Theorem 1. *The dominance problem for CP-nets is P-hard.*

Note that the construction that follows describes a CP-net with incomplete tables. By Truszczynski's result [4], the CP-net described in the proof can be transformed into a CP-net with complete tables, without loss of generality.

The table incompleteness arises because we specify, for multi-valued attributes, a single value that is preferred to all others. We do not specify an order on the less-preferred values.

Proof. Let M be a polynomial time bounded Turing machine with time bound $p(n)$. Without loss of generality, we assume that M a single-tape, single-head Turing machine that starts in state s_0, with the read/write head at the left end of the tape. We further assume that there is a unique accepting state, and that M accepts with an empty tape and the read-write head back in the first square. In other words, there is a unique accepting configuration that depends only on the length of the input.

Let $T(x)$ be the tableau of the computation $M(x)$. In other words, $T(x)$ is a $(p(|x|)+1) \times (p(|x|)+1)$ table, where the i^{th} row represents the configuration of M at step i of the computation. Cell j of row i represents either

- the state M at step i, written immediately to the left of the tape square being scanned by $M(x)$ at step i, or
- the contents of tape square j of $M(x)$ at step i, if the read/write head is to the right of tape square j in the configuration, or
- the contents of tape square $j-1$ of $M(x)$ at step i, if the read/write head is to the left of or at tape square j in the configuration.

Most of the preference attributes in the CP-net that we construct represent the cells in the tableau $T(x)$. There are $\mathcal{O}(p(|x|)^2)$ many such attributes. In addition, there are "gate-keeper" attributes g_0 to $g_{p(|x|)}$. The tableau attributes take on values from the tape alphabet of M, plus blank (B), and the set of states. The gate-keepers are binary.

The initial row of tableau attributes, $t(0,j)$, have unconditionally preferred values that reflect the initial configuration of $M(x)$. The initial value (in the β of our reduction output) of g_0 is 1.

The eventual dominance question, whose answer is equivalent to $M(x)$ accepting, is whether $\alpha \succ \beta$, where β reflects the input configuration of $M(x)$ and the rest of the tableau attributes set to B. In β, all of the g_i have value 0 except $g_0 = 1$. The instance α has B for all tableau attribute values except $t(p(|x|),0) = q_{accept}$, and all $g_i = 1$.

Each tableau attribute $t(i+1,j)$ depends on four tableau attributes in the previous row (from $t(i,j-1)$ to $t(i,j+2)$), to guarantee that it "sees" any movement of the read-write head into its place in the step $i+1$ configuration, or any local change from the previous configuration due to the proximity of the read-write head.

In addition, the tableau attribute $t(i+1,j)$ depends on $g_{|p(x)|}$, g_i and g_{i+1}: If $g_{|p(x)|} = 0$, $g_i = 1$ and $g_{i+1} = 0$ then a preference is specified for $t(i+1,j)$ that guarantees that the preferred value accurately reflects step i of the computation $M(x)$.

If $i < p(|x|)$ and $g_{p(|x|)} = 1$, then the preferred value for $t(i,j)$ is B.

Each g_{i+1} depends on g_i:

- If $g_i = 0$ then $g_{i+1} = 0 \succ g_{i+1} = 1$;
- If $g_i = 1$ then $g_{i+1} = 1 \succ g_{i+1} = 0$.

Suppose that M has the following transition: $\delta(s,q) = (s',R,q')$. Consider the following rows of the preference tables for $t(i+1,j)$ and $t(i+1,j+1)$. (Also see Figure 4.1.)

Note that a transition of the form $\delta(s,q) = (s',L,q')$ would require that we consider attributes $t(i+1,j+1) = q$ and $t(i+1,j+2) = s$, in order for $t(i+1,j)$ to take the value q'.

- If $g_i = 1$ and $g_{i+1} = 0$, $t(i,j) = q$ and $t(i,j+1) = s$ then the preferred value for $t(i+1,j)$ is s'.

– If $g_i = 1$ and $g_{i+1} = 0$, $t(i,j) = q$ and $t(i,j+1) = s$ then the preferred value for $t(i+1,j+1)$ is q'.

	j	$j+1$	
$\cdots$	q	s	$\cdots$
$\cdots$	s'	q'	$\cdots$

Fig. 3. A Turing machine transition

Let $C_{M,x}$ be the CP-net constructed thusly, with attributes $t(i,j)$ and g_i, $0 \leq i,j \leq p(|x|)$. Let β be the instance of $C_{M,x}$ that reflects the initial configuration of $M(x)$ and has $t(i+1,j) = B$ for all $i,j < p(|x|)$, $g_0 = 1$ and $g_{i+1} = 0$. Let α be the instance where $t(p(|x|),0) = q_{accept}$ and all the other $t(i,j) = B$, and all $g_i = 1$.

The i^{th} row of a consistent tableau represents the i^{th} configuration of the computation of $M(x)$. The construction of $C_{M,x}$ guarantees that there is an improving flipping sequence that sets the attributes of the CP-net according to the values of a consistent tableau, and then sets all but the last row of $t(i,j)$ attributes to B. And the only way to set $t(p(|x|),0) = q_{accept}$, is to simulate the computation of $M(x)$ for $p(|x|)$ steps. Thus, we get the following claim.

Claim. $M(x)$ accepts if and only if $\alpha \succ \beta$ in $C_{M,x}$.

If the deterministic computation $M(x)$ accepts, then, by our assumptions on M, there is a unique final configuration of $M(x)$, represented by the last row of attributes in α. Once the final row of attributes has been evaluated, $g_{p(|x|)}$ will be 1, and by preference, all other $t(i,j)$ will be set to B.

There may be many other possible improving flipping sequences, but no others will prove that $\alpha \succ \beta$.

To finish the proof of the theorem, we observe that $C_{M,x}$ can be computed in time polynomial in the representation of M and the length of x.

4.2 Flipping Sequence Lengths and Membership in PSPACE

If we could show that all improving flipping sequences have length polynomial in the number of attributes of the CP-net, then the dominance problem would at least be in NP: Guess a polynomial-length flipping sequence, and verify that each flip is an improving flip.

However, without a polynomial-length guarantee, we can only show that the dominance problem is in PSPACE. The following is a sketch of a nondeterministic linear space algorithm for dominance.

Given α and β on tape 1 and 2, respectively, repeat until the string on tape 1 is the same as that on tape 2: Perform an improving flip on the string on tape 1.

If there is an improving flipping sequence from α to β, some nondeterministic computation will find it. Since NPSPACE = PSPACE, this is sufficient to show that the dominance problem for CP-nets is in PSPACE.

It is not known whether there are exponentially long improving flipping sequences for acyclic CP-nets. However, it is possible to build a cyclic CP-net with an exponential-length improving flipping sequence [5].

To show that the dominance problem for cyclic CP-nets is PSPACE hard, it would be nice to use a reduction like that given in the proof of Theorem 1. However, a generic PSPACE Turing machine must be assumed to run in exponential time. Thus, enumerating attributes for each time step would extend the reduction beyond polynomial time. The modification to that construction is to use (and reuse) only two rows of the tableau: "now" and "next step". Once "next step" is updated, the values of that row are copied to the "now" row, and "next step" is rewritten with blanks. The two phases, UPDATE and COPY, are governed by gate-keeper variables g_0 through $g_{p(|x|)}$, where p is now the space bound of the Turing machine.

Note that the reuse of attributes over time implies an essential cyclicity in this construction.

4.3 PSPACE-Hardness

Theorem 2. *The dominance problem for cyclic CP-nets is PSPACE-complete.*

Proof. We have argued in the previous subsection that the dominance problem for cyclic CP-nets is in PSPACE. We now sketch a polynomial-time computable reduction from a PSPACE Turing machine M and input x to a CP-net and two instances, α and β, such that $\alpha \succ \beta$ if and only if $M(x)$ accepts.

We make the same assumptions about M as in the proof of Theorem 1, including that it has a unique accepting configuration for each input length.

Given M and x and polynomial bound $p(n)$, we construct a CP-net $C_{M,x}$ with $3p(|x|) + 3$ attributes and construct instances α and β.

The attributes of $C_{M,x}$ represent two configurations, "now" (attributes c_0 through c_p) and "next" (attributes d_0 through d_p), and gate-keepers g_0 through g_p. Here "p" is short for $p(|x|)$. The gate-keepers are used to force each cell of the configuration to be updated in each phase.

The two phases, UPDATE and COPY, are regulated by the g_p gate-keeper: When $g_p = 0$, we are in an update phase, and when $g_p = 1$ we are in a copy phase. Each configuration attribute d_j depends on c_{j-1}, c_j, c_{j+1} and c_{j+2}, and on g_{j-1}, g_j, and g_p.

Suppose that M has the following transition: $\delta(s, q) = (s', R, q')$. Consider the following rows of the preference table for d_j.

- If $g_p = 0$, $g_{j-1} = 1$, $g_j = 0$, and if $c_{j-1} = q$ and $c_j = s$, then the preferred value for d_j is q'.
- If $g_p = 0$, $g_{j-1} = 1$, $g_j = 0$, and if $c_j = q$ and $c_{j+1} = s$, then the preferred value for d_j is s'.

– If $g_p = 1$, and $g_{j-1} = 1 = g_j$ then the preferred value for d_j is B.

The COPY phase affects the c_js as follows: If $g_p = 1$, $g_{j-1} = 0$, and $g_j = 1$, then the preferred value for c_j is the value of d_j. (Note that there is a distinct row in the preference table for d_j for each possible value of c_j.)

Finally, and cyclically, we define the preferences for the g_js. Each g_j depends on g_{j-1}, c_{j-1} to c_{j+2}, d_j, and on g_p.

– If $g_p = 0$, $g_{j-1} = 1$, and d_j has been updated according to the transitions of M and the values of c_{j-1} to c_{j+2}, then the preferred value for d_j is 1.
– If $g_p = 1$, $g_{j-1} = 0$, and $c_j = d_j$, then the preferred value for d_j is 0.

For g_0, if $g_p = 1$ and $c_p = d_p$, then the preferred value for g_0 is 1. If $g_p = 0$ and d_p has been updated according to the transitions of M and the values of c_{p-1} and c_p, then the preferred value for g_0 is 0.

Thus, the “updating” of the g_js according to improving flips is interlaced with the updating of the c_js and d_js. In the UPDATE phase, when d_js are updated to reflect the next configuration after that represented by the c_js, the g_js are flipped to 1. In the COPY phase, they are flipped one by one to 0.

Claim. In this construction of $C_{M,x}$, any improving flipping sequence that starts with $g_0 = 1$ and the other g_js set to 0 will first alternate updates of the d_js and g_js, and then alternate updates of the c_js and g_js. This two-phase updating may be repeated as many times as there are steps in the computation of $M(x)$, and each two-phase set of updates, or improving flips, will correspond to one step of that computation. Note that the number of two-phase sets of updates may be exponential in the number of nodes in the CP-net $C_{M,x}$.

Using Claim 4.3, we can show the following.

Claim. Let $C_{M,x}$ be constructed as described in this proof. Let α be the instance for $C_{M,x}$ with $c_0 = q_{accept}$, all other c_j and d_j are B, and all the g_j are 0. Let β be the instance for $C_{M,x}$ where the c_js reflect the initial configuration of $M(x)$, all the d_js are B, $g_0 = 1$, and all $g_{j+1} = 0$. Then $\alpha \succ \beta$ if and only if $M(x)$ accepts.

We observe that the construction of $C_{M,x}$ and of α and β can be done in time polynomial in the description of M and in the length of x.

5 Conclusions

We have shown that the dominance problem for cyclic CP-nets is PSPACE-complete. While this does not prove that the complexity of acyclic CP-net dominance is high, it does indicate that the general model of CP-nets might be a bad choice for a computational model of preferences. We conjecture that many natural preferences are inherently cyclic.

Acknowledgments

The author acknowledges the input from the logic and AI seminar at the University of Kentucky and proofreading by Chris Lusena and Andy Klapper. Thanks are due to Andy Klapper for the observation about self loops and improving flips. This work partially supported by NSF grants CCR-0100040 and ITR-0325063.

The main result and others appear, with significantly different proofs, in [6].

References

1. Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., Poole, D.: CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. JAIR **21** (2004) 135–191
2. Lang, J.: Notes on the computation complexity of dominance and consistency in CP-nets (2004) Manuscript.
3. Wilson, N.: (2004) Personal communication from Jérome Lang.
4. Truszczyński, M.: PSPACE-completeness of dominance in CP-nets (2004) Manuscript.
5. Finkel, R.: CP-nets can produce exponentially long flipping sequences (2004) Manuscript.
6. Goldsmith, J., Lang, J., Truszczyński, M., Wilson, N.: The computatational complexity of dominance and consistency in CP-nets (2005) Manuscript.