Learning Conditional Preference Networks with Queries

Frédéric Koriche, Bruno Zanuttini

Learning Conditional Preference Networks with Queries

Frédéric Koriche
LIRMM, CNRS UMR 5506
Université Montpellier II, France
frederic.koriche@lirmm.fr

Bruno Zanuttini
GREYC, CNRS UMR 6072
Université de Caen Basse-Normandie, France
bruno.zanuttini@info.unicaen.fr

Abstract

We investigate the problem of eliciting CP-nets in the well-known model of exact learning with equivalence and membership queries. The goal is to identify a preference ordering with a binary-valued CP-net by guiding the user through a sequence of queries. Each example is a dominance test on some pair of outcomes. In this setting, we show that acyclic CP-nets are not learnable with equivalence queries alone, while they are learnable with the help of membership queries if the supplied examples are restricted to swaps. A similar property holds for tree CP-nets with arbitrary examples. In fact, membership queries allow us to provide *attribute-efficient* algorithms for which the query complexity is only logarithmic in the number of attributes. Such results highlight the utility of this model for eliciting CP-nets in large multi-attribute domains.

1 Introduction

The spectrum of AI applications that resort on the ability to reason about preferences is extremely wide, ranging from configuration softwares and recommender systems to autonomous agents and group decision-making. As many, if not most, of these applications are defined over large, multiattribute domains, a key challenge in preference research is to develop representation languages and elicitation techniques that cope with the exponential size of the outcome space.

Conditional preference networks (*CP-nets*), have emerged as an expressive language capable of representing ordinal preferences relations in a compact and structured manner [Boutilier *et al.*, 2004]. Briefly, a CP-net is a graph where each node is labelled with a table describing the user's preference over alternative values of this node given different values of the parent nodes. For example, the entry $J_b \wedge P_b : S_r \succ S_b$ might state that, all other things being equal, I prefer a red shirt than a black one if the color for both the jacket and the pants is black. The semantics of a CP-net is defined by a dominance ordering on the outcome space, derived from such reading of entries in the tables. Based on this relation, a key reasoning task is *dominance testing*: given a CP-net N and a pair of outcomes (o, o'), determine whether o dominates o', according to the dominance ordering induced by N.

Ideally, in preference elicitation with CP-nets, the decision-maker should simply "fill the tables" by asking the user how her preference over the values of one node depends on the values of its parents. Yet, in practice, eliciting preferences is far from easy because the dependency graph is generally not known in advance: the decision-maker must therefore seek the interdependencies between attributes and identify a minimal set of parents for each target node. The problem is exacerbated still further by the fact that real-world applications typically involve many irrelevant attributes. For instance, it is not uncommon in recommender systems to describe products using thousands of variables, with the expectation that only a small fraction of these are crucial for describing preferences [Basilico and Hofmann, 2004; Ziegler *et al.*, 2008]. The decision-maker is thus required to select, within a large collection of attributes, a relatively small subset over which the network will be specified.

Such considerations bring into sharp focus the need for *query learning algorithms* that aim at extracting CP-nets by guiding the user through an appropriate sequence of queries. A widely adopted framework for studying this issue is the model of exact learning with equivalence and membership queries [Angluin, 1988]. In essence, equivalence queries simulate a form of *passive* learning in which the decision-maker observes the user's behavior until she finds a counterexample to her hypothesis. By contrast, membership queries capture a form of *active* learning by allowing the decision-maker to ask about examples of her own choice. The utility of this model lies in the fact that rich concept classes, including Horn theories, decision trees, and some description logics, have been shown to be learnable with both equivalence queries and membership queries, while in weaker versions one can prove superpolynomial lower bounds [Angluin *et al.*, 1992; Bshouty, 1995; Frazier and Pitt, 1996].

In the learning model suggested by this study, the target concept is a dominance ordering on the outcome space. Each example is a preference situation involving some pair of outcomes. For a *membership query*, the learner supplies an example (o, o') and is told whether o dominates o', or not. For an *equivalence query*, the learner presents a CP-net N, and either is told that N correctly identifies the target concept, or it is given a counterexample (o, o'). The goal is to identify the target concept using as few resources as possible, where resources refer both to the run time and the number of queries.

From a practical perspective, one must take into account the fact that outcomes are typically *not* comparable with an equivalent cost. As observed in [Green and Srinivasan, 1978], users can meaningfully compare outcomes if they differ only on very few attributes. Similarly, for the learner, this task can be arduous because dominance testing is generally NP-hard, even for acyclic CP-nets. Thus, our learnability results are defined in terms of a *concept class* in which the target concept is chosen, and an *instance class* that circumscribes the set of examples used by equivalence and membership queries.

The key message to be gleaned from this paper is that *active* learning is required to correctly and efficiently extract preference networks in binary-valued domains. On the one hand, acyclic CP-nets are *not* learnable with equivalence queries alone, while on the other, they are learnable with equivalence and membership queries, provided that the instance class is restricted to simple outcome pairs for which dominance testing takes linear time. Interestingly, a similar property holds for tree-structured CP-nets by extending the instance class to arbitrary examples. When membership queries are available, we provide *attribute-efficient* learning algorithms for which the query complexity is *linear* in the size of the minimal CP-net that identifies the target concept, and *logarithmic* in the total number of attributes. Such encouraging results pave the way for fast elicitation techniques capable of extracting "small" CP-nets in "large" domains.

2 Conditional Preference Networks

The learning problems under consideration in this study are defined over a set of Boolean variables $X = \{x_1, \cdots, x_n\}$. As usual, we refer to x_i and $\overline{x}_i$ as literals. Given a literal p, we denote by $\overline{p}$ the opposite of p; for example, if p_i is $\overline{x}_i$, then $\overline{p}_i$ is x_i. A term t is a conjunction of literals. By $var(t)$ we denote the set of variables occurring in t. A term t is *maximal* for a subset of variables $Y \subseteq X$ if $var(t) = Y$.

A *conditional preference rule* (*CP-rule*) on a variable x is an expression of the form $t : p \succ \overline{p}$, where p is a literal of x and t is a term such that $x \notin var(t)$. Such a rule captures the statement "given that t holds, the value p is preferred to the value $\overline{p}$ for the variable x, all other things being equal".

A *conditional preference table* (*CP-table*) on a variable x with respect to a set $Y \subseteq X \setminus \{x\}$ is a set of CP-rules on x that associates *at most* one rule $t : p \succ \overline{p}$ to each maximal term t for Y. The CP-table is *complete* if *exactly* one rule $t : p \succ \overline{p}$ is associated to each maximal term t for Y.

A *conditional preference net* (*CP-net*) is a labelled digraph N over a subset $var(N)$ of X, where each node $x \in var(N)$ is annotated with a CP-table $cpt(x)$ on x with respect to the set $Pa(x)$ of parents of x in the graph. The CP-net is *complete* if any node $x \in var(N)$ is annotated with a complete CP-table. A CP-net is *acyclic* if its digraph is acyclic, and *tree-structured* if its digraph forms a forest, that is, a set of trees.

These notions are illustrated in the left part of Figure 1, where an acyclic complete CP-net is specified for the popular evening dress scenario. I unconditionally prefer black (j, p) to white $(\overline{j}, \overline{p})$ as the color of both the jacket and the pants, while my preference for a red shirt (s) versus a white one $(\overline{s})$ depends on the combination of jacket and pants.

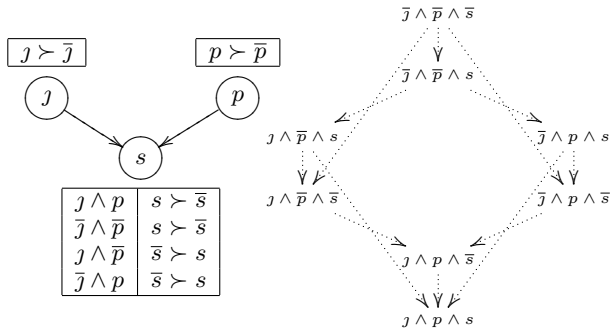

Figure 1: A CP-net and its preference graph for "evening dress"

An *outcome* is a maximal term o for X. Given a term t, we write $o[t]$ for the outcome obtained by making o agree with t, i.e. $o[t] = \{p : p \in t \text{ or } p \in o - t\}$. For example, if $o = x_1x_2x_3$ and $t = x_1\overline{x}_2$, then $o[t] = x_1\overline{x}_2x_3$. An outcome o *satisfies* a term t if $o = o[t]$. We write $\mathbf{0}$ (resp. $\mathbf{1}$) for the outcome that assigns 0 (resp. 1) to every variable in X.

The ceteris paribus semantics of a CP-rule $t : p \succ \overline{p}$ on a variable x can be described as follows: if an outcome o satisfies the condition t and assigns the value p to x, then it is preferred to an outcome o' which differ from o only in that is assigns the value $\overline{p}$ to x. In formal terms, we say that o is *preferred* to o' for $t : p \succ \overline{p}$ if $o = o[t \wedge p]$ and $o' = o[\overline{p}]$. In this case, (o, o') is called a *model* of the rule $t : p \succ \overline{p}$.

Given a CP-net N and two outcomes o and o', we say that o *dominates* o' for N if there is a sequence $(o_1, \cdots, o_m)$ such that $o_1 = o'$, $o_m = o$ and for each $i : 1 \leq i < m$, (o_{i+1}, o_i) is a model of some CP-rule of N. In this case, (o, o') is called a *model* of N, and $(o_1, \cdots, o_m)$ an *improving sequence* from o' to o. The *dominance ordering* of N, denoted $\succ_N$, is the set of all models of N. For example, the dominance ordering of the CP-net for evening dress is the transitive closure of the digraph displayed in the right part of Figure 1.

A CP-net N is *consistent* if there is no outcome o which dominates itself, i.e. $o \succ_N o$. If N is consistent, then $\succ_N$ is a strict partial order over the outcome space. As reported in [Boutilier *et al.*, 2004], any acyclic CP-net is consistent.

Given two CP-nets N and N', we say that N *subsumes* N' if for any CP-rule $t' : p \succ \overline{p}$ in N', there is a CP-rule $t : p \succ \overline{p}$ in N such that $t' \subseteq t$. A CP-net N is *minimal* if there is no distinct net N' subsumed by N and for which $\succ_N = \succ_{N'}$. For example, the CP-net in Figure 1 is minimal.

Finally, the *size* $|r|$ of a CP-rule r will be the number of occurrences of literals in its definition. Specifically, if r is of the form $t : p \succ \overline{p}$, then $|r| = |t| + 2$. The *size* $|N|$ of a CP-net N will be the sum of $|r|$ over all rules r in N. For example, the size of the CP-net in Figure 1 is $2 + 2 + 4 \times (2 + 2) = 20$.

3 Exact Learning with Queries

The learning criterion expected in this study is that of *exact identification*, which is achieved if the learner can infer a CP-net that correctly represents the target concept. A *concept* is a strict partial ordering $\succ$ on the outcome space. A *representation class* is a set $\mathcal{N}$ of CP-nets. A concept $\succ$ is *representable* by $\mathcal{N}$ if there is a CP-net $N \in \mathcal{N}$ such that $\succ_N = \succ$. The

concept class $\mathcal{C}_\mathcal{N}$ over $\mathcal{N}$ is the set of all concepts that are representable by $\mathcal{N}$. The *description size* of a concept $\succ$ in $\mathcal{C}_\mathcal{N}$ will be the minimum of $|N|$ over all representations N of $\succ$ in $\mathcal{N}$. Finally, an instance or *example* is a pair (o, o') of outcomes, and an *instance class* is a set $\mathcal{O}$ of examples.

Let $\succ$ be a target concept of some concept class $\mathcal{C}_\mathcal{N}$, and $\mathcal{O}$ be an instance class. The learner may extract information about $\succ$ using two types of queries. A *membership query* MQ over $\mathcal{O}$ takes an example $(o, o') \in \mathcal{O}$ and returns `Yes` if $o \succ o'$, and `No` otherwise. An *equivalence query* EQ over $\mathcal{O}$ takes a CP-net $N \in \mathcal{N}$, and returns `Yes` if N is a representation of $\succ$, or returns a counterexample $(o, o') \in \mathcal{O}$ otherwise. The counterexample (o, o') is *positive* if $o \succ o'$ and $o \not\succ_N o'$, and *negative* if $o \not\succ o'$ and $o \succ_N o'$.

Definition 1. An algorithm A is a *query learning algorithm* for a concept class $\mathcal{C}_\mathcal{N}$ with respect to an instance class $\mathcal{O}$ if there are two polynomials p and q such that, for any target concept $\succ$ in $\mathcal{C}_\mathcal{N}$, after $p(n)$ membership and equivalence queries over $\mathcal{O}$, and total running time in $q(n)$, A outputs a representation $N \in \mathcal{N}$ of $\succ$. It is *attribute-efficient* if the number of queries $p(n)$ is polynomial in the description size of $\succ$, but only polylogarithmic in the number of variables n.

Given an instance class $\mathcal{O}$, we say that a concept class $\mathcal{C}_\mathcal{N}$ is *attribute-efficiently learnable* if there is an attribute-efficient query learning algorithm for $\mathcal{C}_\mathcal{N}$ with respect to $\mathcal{O}$.

Clearly, the strength of query-directed learning lies in membership queries, which model not only the interaction with a user, but also the careful crafting of experiments by a decision-maker in order to observe the response of the user. In order to demonstrate that a class of CP-nets is *not* learnable with equivalence queries alone, we shall use the technique of *approximate fingerprints* introduced in [Angluin, 1990].

Intuitively, a concept class $\mathcal{C}_\mathcal{N}$ has approximate fingerprints if it includes a set $\mathcal{C}^*_\mathcal{N}$ such that for each hypothesis N in $\mathcal{C}_\mathcal{N}$ supplied by the learner, the user can choose a counterexample for N that eliminates only a superpolynomially small fraction of candidate concepts in $\mathcal{C}^*_\mathcal{N}$. By repeating this process, the learner cannot be certain of the target concept in $\mathcal{C}^*_\mathcal{N}$ after only polynomially many equivalence queries.

A pair (o, o') of outcomes is called an α-*fingerprint* of a concept class $\mathcal{C}^*_\mathcal{N}$ according to some concept $\succ_1$ if

$$\frac{|\{\succ_2 \in \mathcal{C}^*_\mathcal{N} : o \succ_2 o' \text{ iff } o \succ_1 o'\}|}{|\mathcal{C}^*_\mathcal{N}|} < \alpha$$

Formally, a concept class $\mathcal{C}_\mathcal{N}$ has *approximate fingerprints* with respect to an instance class $\mathcal{O}$ if for any polynomial $p(n)$, $\mathcal{C}_\mathcal{N}$ includes a subclass $\mathcal{C}^*_\mathcal{N}$ such that for any sufficiently large n, $\mathcal{C}^*_\mathcal{N}$ contains at least two concepts, and for all concepts $\succ$ in $\mathcal{C}_\mathcal{N}$ of description size bounded by $p(n)$, there is an example in $\mathcal{O}$ which is a $\frac{1}{p(n)}$-fingerprint of $\mathcal{C}^*_\mathcal{N}$ according to $\succ$.

4 Learning Acyclic CP-nets with Queries

Acyclic CP-nets take a central part in preference research by providing the right level of expressivity for many real-world applications, while remaining tractable for certain reasoning tasks such as outcome optimization [Domshlak *et al.*, 2001; Boutilier *et al.*, 2004]. We begin with some useful properties.

Two outcomes form a *swap* if they differ in the value of only one variable. Such examples correspond to simple situations of the form "for this car I prefer it red than white", where the color is one of the multiple attributes of the car. The next property states that, in acyclic CP-nets, swaps can be retrieved in linear time by simple rule matching.

Lemma 1. Let N be an acyclic CP-net and (o, o') be a swap. Then $o \succ_N o'$ if and only if there is a CP-rule r in N such that (o, o') is a model of r.

Proof. The *if* direction is immediate. Conversely, if $o \succ_N o'$, then there is an improving sequence from o' to o in N. Assume w.l.o.g. that o satisfies x_1 and $o' = o[\overline{x}_1]$. Using the suffix fixing rule [Boutilier *et al.*, 2004, Section 5.1], we can also assume that the sequence affects only x_1 and its ascendants in N. By acyclicity, one of those ascendants x_k is modified but has none of its parents modified in the sequence. Thus, x_k cannot be modified back to its value in o', which entails $x_k = x_1$. So only x_1 is modified, which concludes. □

A CP-net N is a *minimal representation* of a concept $\succ$ if N represents $\succ$ and $|N|$ is minimal.

Lemma 2. For the class $\mathcal{C}_{\text{ACY}}$ of acyclic CP-nets, the minimal representation of any concept is unique.

Proof. Let $\succ$ be a concept in $\mathcal{C}_{\text{ACY}}$ and N a representation of $\succ$ satisfying the following: for any variables x and y, and any rule $r \in N$ on x, if y is a parent of x in N, then there is a model (o, o') of r such that exactly one of $(o[y], o'[y])$ and $(o[\overline{y}], o'[\overline{y}])$ is not a model of r. This amounts to say that y is a *relevant* parent of x. Clearly, such a representation exists: take any representation and remove all irrelevant parents. Now let N' be any representation of $\succ$; we show that N' subsumes N. Let $r \in N$ be a rule of the form $t : p \succ \overline{p}$. Any swap (o, o') for which $o = o[t \wedge p]$ and $o' = o[\overline{p}]$ is a model of r. Since $\succ_N = \succ_{N'}$, by Lemma 1, (o, o') must be a model of some rule r' in N' of the form $t' : p \succ \overline{p}$. If $t \not\subseteq t'$, then there is a variable $y \in var(t) \setminus var(t')$ such that $(o[y], o'[y])$ and $(o[\overline{y}], o'[\overline{y}])$ are both models of r', but exactly one of these pairs is not a model of r, contradicting the fact that $\succ_N = \succ_{N'}$. So $t \subseteq t'$, and hence, N' subsumes N. □

We now show that acyclic and complete CP-nets have approximate fingerprints, even if examples are restricted to swaps. Thus, by applying Theorem 1 in [Angluin, 1990], they are *not* learnable with equivalence queries alone.

Theorem 1. The class $\mathcal{C}_{\text{ACC}}$ of acyclic complete CP-nets has approximate fingerprints with respect to swap examples.

Proof. Let $\mathcal{C}^*_{\text{ACC}}$ be the class of all concepts represented by a CP-net N^* with $\log n$ root nodes x_j pointing to the same fixed child node x_1. Each table $cpt(x_j)$ has the rule $x_j \succ \overline{x}_j$. The table $cpt(x_1)$ includes one rule $s : x_1 \succ \overline{x}_1$, where s is the conjunction of all positive literals in $Pa(x_1)$, and $n - 1$ rules $s' : \overline{x}_1 \succ x_1$, where s' is any maximal term of $Pa(x_1)$ with at least one negative literal. Clearly N^* is acyclic and complete. Furthermore, $|\mathcal{C}^*_{\text{ACC}}| = \binom{n-1}{\log n}$. Now, let $p(n) = n^k$ for some constant k, and let $N \in \mathcal{C}_{\text{ACC}}$ with $|N| \leq p(n)$.

The fingerprint (o, o') is defined as follows. If N does not include any rule of the form $t : x_1 \succ \overline{x}_1$, then we let $o = \mathbf{1}$ and $o' = o[\overline{x}_1]$. Then $o \not\succ_N o'$ but $o \succ o'$ for any concept $\succ$ in $\mathcal{C}^*_{\text{ACC}}$. So (o, o') is an α-fingerprint of $\mathcal{C}^*_{\text{ACC}}$ for any $\alpha > 0$.

Now, if N includes a rule of the form $t : x_1 \succ \overline{x}_1$, then o is any outcome satisfying $t \wedge x_1$ and containing $k \log n$ positive literals excluding x_1, which can be constructed as follows. Because N is complete, the size of its table on x_1 is more than $2^{|t|}$. Since $|N| \leq n^k$ we get $|t| \leq k \log n$. Thus, o can be constructed by satisfying t first and filling the rest as necessary. Again, $o' = o[\overline{x}_1]$. So $o \succ_N o'$ and o has $k \log n$ positive literals (excluding x_1). Hence, the number of concepts $\succ$ in $\mathcal{C}^*_{\text{ACC}}$ for which $o \succ o'$ is $\binom{k \log n}{\log n}$. Using $\frac{a-i}{b-i} \leq \frac{a}{b}$,

$$\frac{|\{\succ \in \mathcal{C}^*_{\text{ACC}} : o \succ o'\}|}{|\mathcal{C}^*_{\text{ACC}}|} = \prod_{i=0}^{\log n - 1} \frac{k \log(n-i)}{n-1-i} \leq \frac{(k \log n)^{\log n}}{(n-1)^{\log n}}$$

Taking logarithms, this proportion is less than $\frac{1}{n^k}$ if and only if $\frac{n-1}{\log n} > k 2^k$, which is true for sufficiently large n. □

When membership queries are available, we can provide an attribute-efficient algorithm for learning acyclic, and possibly *incomplete*, CP-nets, provided that the supplied examples are restricted to swaps. As specified in Algorithm 1, the learner iteratively updates her hypothesis N by asking equivalence queries. On seeing a counterexample (o, o'), the learner checks whether N includes a rule that covers either (o', o) or (o, o'). If this is indeed the case, she asks membership queries in order to refine the condition of that rule. Otherwise, she expands her net with a new rule. Here, each rule r of the form $t : p_i \succ \overline{p}_i$ is associated with an outcome o_r, called the *support* of r, and such that $(o_r[p_i], o_r[\overline{p_i}])$ is a model of r.

The key routine SEARCHPARENT finds a new parent of some misclassifying rule r, using only a logarithmic number of membership queries. Using the support o_r of r and the last counterexample (o, o'), it operates a binary search on the sequence $(o_1, \cdots, o_n)$ where o_j is formed by the first j literals occurring in o_r and the last $n - j$ literals occurring in o. As shown in the lemma below, SEARCHPARENT is guaranteed to find a new parent in the rule r, by maintaining the invariant that for each explored subsequence $(o_a, \cdots, o_b)$, we have both $o_a[p_i] \not\succ o_a[\overline{p}_i]$ and $o_b[p_i] \succ o_b[\overline{p}_i]$ in the target concept.

Lemma 3. Let $\succ$ be a concept of $\mathcal{C}_{\text{ACY}}$, o_a, o_b be two outcomes, and $p_i, \overline{p}_i$ be a pair of opposite literals for some variable x_i. If we have $o_a[p_i] \not\succ o_a[\overline{p}_i]$ and $o_b[p_i] \succ o_b[\overline{p}_i]$, then there is a parent x_j of x_i in the minimal representation N^* of $\succ$ whose value is different in o_a and o_b.

Proof. Consider the outcome sequence $(o_0, \cdots, o_n)$ where $o_j = o_a[t_j]$ with t_j the conjunction of the first j literals in o_b. Since $o_0 = o_a$ and $o_n = o_b$, there is some $j > 0$ such that $o_{j-1}[p_i] \not\succ o_{j-1}[\overline{p}_i]$ and $o_j[p_i] \succ o_j[\overline{p}_i]$. So there is a rule $t : p_i \succ \overline{p}_i$ in N^* such that o_j satisfies t but o_{j-1} does not. Since they differ only on x_j, we get $x_j \in var(t)$. □

Lemma 4. Let $\succ$ be a target concept in the class $\mathcal{C}_{\text{ACY}}$. Then Algorithm 1 maintains an acyclic CP-net N which is always subsumed by the minimal representation N^* of $\succ$.

Algorithm 1: Learning Acyclic CP-nets

```
N ← ∅
while (o, o') ← [EQ(N) ≠ Yes] do
    let x_i be the variable s.t. p_i ∈ o and p̄_i ∈ o'
    if (o', o) is a model of some rule ⟨r, o_r⟩ in N then
        (o, o') ← (o', o)
    if (o, o') is a model of some rule ⟨r, o_r⟩ in N then
        /* The counterexample is negative */
        x_j ← SEARCHPARENT(x_i, p_i, o, o_r, 0, n)
        foreach rule ⟨r', o_r'⟩ in the table of x_i expand the
        condition of r' with the literal of x_j in o_r'
    else
        /* The counterexample is positive */
        add the rule ⟨t : p_i ≻ p̄_i, o⟩ to N where t is the
        projection of o onto the set Pa(x_i) in N
return N
```

Procedure SEARCHPARENT(x_i, p_i, o, o', a, b)

```
if a = b + 1 then return x_b
j ← ⌊(a + b)/2⌋
o_j ← o[t_j] where t_j are the first j literals occurring in o'
if MQ(o_j[p_i], o_j[p̄_i]) = Yes then
    return SEARCHPARENT(x_i, p_i, o, o', a, j)
else
    return SEARCHPARENT(x_i, p_i, o, o', j, b)
```

Proof. Initially, $N = \varnothing$, so the property holds. Now, consider an iteration of the main loop and suppose by induction hypothesis (I.H.) that N is subsumed by N^* before calling EQ. If EQ returns a positive counterexample (o, o'), then by Lemma 1 N^* includes a rule $t^* : p_i \succ \overline{p}_i$ for which $o = o[t^* \wedge p_i]$ and $o' = o[\overline{p}_i]$. Then N is expanded with $t : p_i \succ \overline{p}_i$ where t is the projection of o onto the parent set $Pa(x_i)$ of x_i in N. Since (by I.H.) N is subsumed by N^*, we have $Pa(x_i) \subseteq Pa^*(x_i)$, where Pa^* is the parent set in N^*. Therefore $t \subseteq t^*$, and hence, the invariant is preserved.

Dually, if EQ returns a negative counterexample (o, o'), then by Lemma 1 N includes a rule r of the form $t : p_i \succ \overline{p}_i$ for which $o = o[t \wedge p_i]$ and $o' = o[\overline{p}_i]$. By construction, for the support o_r of r we also have $o_r = o_r[t \wedge p_i]$, and $o_r \succ o_r[\overline{p}_i]$. So by Lemma 3, SEARCHPARENT returns a parent x_j of x_i in N^*. Now, let r' be any rule of the form $t' : p'_i \succ \overline{p}'_i$. Since the support $o_{r'}$ of r' satisfies $o_{r'}[p'_i] \succ o_{r'}[\overline{p}'_i]$, by I.H., there is a rule $t^* : p'_i \succ \overline{p}'_i$ in N^* such that $t' \subseteq t^*$ and $o_{r'}$ satisfies t^*. This together with the fact that x_j is a parent of x_i in N^* ensures $t' \cup \{p_j\} \subseteq t^*$, so the invariant is preserved. □

Now let us examine the complexity of Algorithm 1. For equivalence queries, each counterexample allows us to find a new rule $t : p_i \succ \overline{p}_i$ or a new literal p_j of some rule in the minimal representation N^* of the target concept. Because the hypothesis N is always subsumed by N^*, this can happen at most $|N^*|$ times. For membership queries, at most $\log n$ of these are used in each call of SEARCHPARENT, which always uncovers a new parent of some variable. So the number of these queries is at most $e \log n$, where $e = \sum_i Pa(x_i)$.

Theorem 2. The class $\mathcal{C}_{\text{ACY}}$ is attribute-efficiently learnable from equivalence and membership queries over swaps: any concept $\succ$ in $\mathcal{C}_{\text{ACY}}$ can be identified in polynomial time, using at most $|N^*| + 1$ equivalence queries and $e\lceil \log n \rceil$ membership queries, where N^* is the minimal representation of $\succ$, and e is the number of edges in N^*.

5 Learning Tree CP-nets with Queries

Binary-valued tree-structured CP-nets constitute a restricted, yet important, class of preference networks for which dominance testing on *arbitrary* pairs of outcomes is solvable in quadratic time using a backtrack-free search technique [Boutilier *et al.*, 2004]. It is therefore legitimate to study the learnability issues of this class in the general setting where the examples supplied to the learner are arbitrary preference situations. In this context, we first show that tree-structured CP-nets are not learnable with equivalence queries alone.

Theorem 3. The class $\mathcal{C}_{\text{TREE}}$ of tree CP-nets has approximate fingerprints with respect to arbitrary outcome pairs.

Proof. We assume w.l.o.g. that n is even to avoid floors and ceilings. To each permutation π of $(x_1, \cdots, x_n)$ we associate the smallest set of rules N_π such that $x_{\pi(1)} \succ \overline{x}_{\pi(1)}$ is in N_π, and for each $i > 1$, N_π includes $x_{\pi(i-1)} : x_{\pi(i)} \succ \overline{x}_{\pi(i)}$. Let $\mathcal{C}^*_{\text{TREE}}$ be the class of all concepts represented by some N_π specified as above. Clearly, $|\mathcal{C}^*_{\text{TREE}}| = n!$. Now, let $p(n) = n^k$ for some constant k and let $N \in \mathcal{C}_{\text{TREE}}$ with $|N| \leq p(n)$.[1]

The fingerprint (o, o') is defined as follows. Assume first that there is an outcome o_1 containing at least $n/2$ ones and such that $(o_1, \mathbf{0})$ is a model of N. Then there is an improving sequence from $\mathbf{0}$ to o_1 in N, and since variables are flipped one by one, it must contain an outcome o with exactly $n/2$ ones. Moreover, by construction $o \succ_N \mathbf{0}$ holds. Let $o' = \mathbf{0}$. We claim that (o, o') is an $\frac{1}{n^k}$-fingerprint of $\mathcal{C}^*_{\text{TREE}}$ w.r.t. $\succ_N$. Indeed, a concept N_π in $\mathcal{C}^*_{\text{TREE}}$ has (o, o') as a model if and only if the first $n/2$ variables according to π are exactly those assigned 1 by o. Otherwise, any improving sequence in N_π should flip at least one variable assigned 0 by both o and o', with no way back, a contradiction. It follows that there are $(n/2)!(n/2)!$ concepts in $\mathcal{C}^*_{\text{TREE}}$ with (o, o') as a model, hence

$$\frac{|\{\succ \in \mathcal{C}^*_{\text{TREE}} : o \succ o'\}|}{|\mathcal{C}^*_{\text{TREE}}|} = \frac{\left(\frac{n}{2}!\right)^2}{n!} = \binom{n}{\frac{n}{2}}^{-1} \leq \frac{\sqrt{2n}}{2^n}$$

which is clearly less than $\frac{1}{n^k}$ for sufficiently large n. Now, assume that there is no o_1 as above. Let $o = \mathbf{1}$ and $o' = \mathbf{0}$. So $o \not\succ_N o'$, but $o \succ o'$ holds for every concept $\succ$ in $\mathcal{C}^*_{\text{TREE}}$. Hence, (o, o') is an α-fingerprint of $\mathcal{C}^*_{\text{TREE}}$ for any $\alpha > 0$. □

As further evidence for the utility of membership queries, we now give an attribute-efficient algorithm for eliciting tree CP-nets in presence of *arbitrary* examples. Let $\triangle(o, o')$ be the set of all variables whose value differ in two outcomes o and o'. Algorithm 2 uses the fact that considering only variables in $\Delta(o, o')$ and their ascendants is enough for searching an improving sequence from o' to o. Thus, on seeing a counterexample (o, o'), the learner computes the tables for each such variable. Because any variable has at most one parent, its table can be found using few membership queries.

From a practical perspective, note that the examples used in MQ's are restricted to swaps in order to minimize the cognitive effort spent by the user in comparing outcomes.[2]

Algorithm 2: Learning tree CP-nets

```
N ← ∅
while (o, o') ← [EQ(N) ≠ Yes] do
    /* The counterexample is necessarily positive */
    for each x ∈ △(o, o') do PROPAGATE(x)
return N
```

Procedure PROPAGATE(x)

```
if x ∈ var(N) then return
var(N) ← var(N) ∪ {x}
for o, p ∈ {0, 1} × {x, x̄} do pref_{o,p} ← MQ(o[p], o[p̄])
if pref_{0,p} = pref_{1,p} = Yes for some p ∈ {x, x̄} then
    parents ← ∅
else if pref_{0,p} = Yes for some p ∈ {x, x̄} then
    parents ← {SEARCHPARENT(x, p, 1, 0, 0, n)}
else if pref_{1,p} = Yes for some p ∈ {x, x̄} then
    parents ← {SEARCHPARENT(x, p, 0, 1, 0, n)}
else
    parents ← ∅
add to N a table for x with parents parents and the rules
satisfied by the pref_{o,p}'s
if parents = {y} then PROPAGATE(y)
```

Lemma 5. Let N^* be a minimal representation of the target concept in $\mathcal{C}_{\text{TREE}}$. Then PROPAGATE is called only on variables x in $var(N^*)$, and always extracts the table of x in N^*.

Proof. By construction, when PROPAGATE(x_i) is called by Algorithm 2, we must have $x \in \triangle(o, o')$ and $o \succ o'$. So $x \in var(N^*)$, because otherwise it would have no table in N^* and hence, its value could not change along any improving sequence from o' to o. Now given $x \in var(N^*)$, we show that PROPAGATE computes the right set of parents for x.

First, let $\text{MQ}(\mathbf{0}[p], \mathbf{0}[\overline{p}]) = \text{MQ}(\mathbf{1}[p], \mathbf{1}[\overline{p}]) = \texttt{Yes}$. Then by Lemma 1 there is a rule $t : p \succ \overline{p}$ in N^* such that both $\mathbf{0}$ and $\mathbf{1}$ satisfy t. Hence t is empty. Now to the second case. Here $\text{MQ}(\mathbf{0}[p], \mathbf{0}[\overline{p}]) = \texttt{Yes}$ and $\text{MQ}(\mathbf{1}[p], \mathbf{1}[\overline{p}]) = \texttt{No}$, so by Lemma 3 there is a parent y of x in N^*, which is found by SEARCHPARENT. The third case is symmetric. In the last case, all queries answer No, so there is no rule on x in N^*, and hence, x has no parent in N^*.

Consequently, in all cases PROPAGATE computes the right set of (at most one) parents. Because each possible rule is validated by one of the queries $\text{MQ}(o[p], o[\overline{p}])$, the table computed for x is the correct one. Furthermore, since each recursive call of PROPAGATE is on a variable y which is the parent of some variable in $var(N^*)$, we have $y \in var(N^*)$. □

[1] For a tree CP-net N, $|N| \leq 6|var(N)|$ (two rules of size 3 per variable), so if $p(n) \geq 6n$, N can be any tree CP-net.

[2] In addition, the outcomes $\mathbf{1}$ and $\mathbf{0}$ used in PROPAGATE can be replaced by any suitable pair $(o, \overline{o})$ for which $\overline{o} = \{\overline{p} : p \in o\}$.

By Lemma 5, it follows that all counterexamples supplied to the learner are positive. Moreover, from the structure of the algorithm it is easily seen that after treating (o, o'), the hypothesis N contains the correct tables for all ascendants of all variables in $\triangle(o, o')$. This together with the suffix fixing principle [Boutilier *et al.*, 2004, Section 5.1] ensures that N now agrees with $o \succ o'$, and so that the algorithm converges.

Concerning the complexity of our learning algorithm, the number of equivalence queries is at most $|var(N^*)| + 1$, because each counterexample allows the learner to treat at least one new variable in N^*. Likewise, the routine PROPAGATE treats each variable in $var(N^*)$ exactly once, using at most $\log n + 4$ membership queries for computing the $pref_{o,p}$'s plus searching for a parent. Finally, the hypothesis maintained by the learner is always a subtree of N^*, and hence, dominance testing can be evaluated in quadratic time.

Theorem 4. The class $\mathcal{C}_{\text{TREE}}$ is attribute-efficiently learnable from equivalence and membership queries over arbitrary outcome pairs: any concept $\succ$ in this class can be identified in polynomial time using at most $k + 1$ equivalence queries and $k\lceil \log n \rceil + 4k$ membership queries, where k is the number of nodes in the minimal representation of $\succ$.

6 Discussion

Along the lines of making query-directed learning applicable to preference elicitation, we have provided a model for learning preference networks from equivalence and membership queries, together with significant learnability results. Taking into account the cognitive effort required by human users to answer queries, our model is distinguished by the close way in which it integrates learning and dominance testing, and the insistence on having convergence bounds that are polynomial in the minimal description size of the target concept, but only polylogarithmic in the total number of attributes. In essence, our results reveal that membership queries are essential for extracting both acyclic CP-nets from restricted outcome pairs, and tree CP-nets from arbitrary outcome pairs. Importantly, the examples used by these queries can be limited to "swaps" in order to facilitate their comparison.

To the best of our knowledge, this work provides the first connection between *active* learning and graphical preference languages. Some authors, though, have recently focused on *passive* learning, where the goal is to extract a CP-net from a set of examples [Athienitou and Dimopoulos, 2007]. Yet, in this "offline" passive learning model, the problem of finding an acyclic CP-net consistent with a set of arbitrary outcome pairs is NP-hard, even if the dependence graph is known in advance [Lang and Mengin, 2009]. Note that in the "online" passive learning model, acyclic CP-nets are not predictable with a polynomial mistake bound, even if examples are restricted to swaps; this is a direct corollary of Theorem 1, that follows from a standard conversion between online learning and exact learning with EQ's alone [Littlestone, 1988].

Perhaps the closest framework to ours is due to Sachdev [2007], who investigates some preference logics in different learning models. Although encouraging, his results do not take into account the cost of dominance testing, and the query complexity grows exponentially with the size of rules.

A direction of research that naturally emerges from our study is to extend the learnability results to larger classes of preference networks. Algorithm 1 can provide a starting point for attacking multi-valued acyclic CP-nets. Similarly, Algorithm 2 might be extended to directed-path singly-connected CP-nets. Less obvious, however, is to efficiently learn *cyclic* CP-nets even with swap examples. Finally, the problem of *revising* CP-nets with queries looks challenging.

Acknowledgments: Thanks to the referees for helpful comments. This work was supported by the ANR project ANR-06-BLAN-083-02.

References

[Angluin *et al.*, 1992] D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9:147–164, 1992.

[Angluin, 1988] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

[Angluin, 1990] D. Angluin. Negative results for equivalence queries. *Machine Learning*, 5:121–150, 1990.

[Athienitou and Dimopoulos, 2007] F. Athienitou and Y. Dimopoulos. Learning CP-networks: A preliminary investigation. In *Adv. in Preference Handling (M-PREF)*, 2007.

[Basilico and Hofmann, 2004] J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In *ICML'04*, pages 9–17, 2004.

[Boutilier *et al.*, 2004] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning with conditional *ceteris paribus* preference statements. *J. Artif. Intell. Res.*, 21:135–191, 2004.

[Bshouty, 1995] N. Bshouty. Exact learning boolean functions via the monotone theory. *Information and Computation*, 123(1):146–153, 1995.

[Domshlak *et al.*, 2001] C. Domshlak, R. Brafman, and S. Shimony. Preference-based configuration of web page content. In *IJCAI'01*, pages 1451–1456, 2001.

[Frazier and Pitt, 1996] M. Frazier and L. Pitt. Classic learning. *Machine Learning*, 25(2-3):151–193, 1996.

[Green and Srinivasan, 1978] P. Green and V. Srinivasan. Conjoint analysis in consumer research: Issues and outlook. *J. Consumer Research*, 5(2):103–123, 1978.

[Lang and Mengin, 2009] J. Lang and J. Mengin. The complexity of learning separable ceteris paribus preferences. In *IJCAI'09 (these proceedings)*, 2009.

[Littlestone, 1988] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

[Sachdev, 2007] M. Sachdev. On learning of *ceteris paribus* preference theories. Master's thesis, Graduate Faculty of North Carolina State University, 2007.

[Ziegler *et al.*, 2008] C. Ziegler, G. Lausen, and J. Konstan. On exploiting classification taxonomies in recommender systems. *AI Communications*, 21(2-3):97–125, 2008.