Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

An Empirical Investigation of Ceteris Paribus Learnability

Loizos Michael and **Elena Papageorgiou**
Open University of Cyprus
loizos@ouc.ac.cy and elena.papageorgiou@st.ouc.ac.cy

Abstract

Eliciting user preferences constitutes a major step towards developing recommender systems and decision support tools. Assuming that preferences are *ceteris paribus* allows for their concise representation as Conditional Preference Networks (CP-nets). This work presents the first empirical investigation of an algorithm for *reliably* and *efficiently* learning CP-nets in a manner that is *minimally intrusive*. At the same time, it introduces a novel process for *efficiently reasoning* with (the learned) preferences.

1 Introduction

In an era of online commercial transactions and massive online open courses, the ability to personalize products and services becomes more pertinent than ever before. Accommodating this personalization requires a means not only to represent the preferences of those involved, but also to elicit them.

Conditional Preference Networks (CP-nets) have been proposed as a formalism to represent and reason with such preferences [Boutilier *et al.*, 2004], and their learnability has been studied [Dimopoulos *et al.*, 2009; Lang and Mengin, 2009; Koriche and Zanuttini, 2010] in situations resembling the following: Assume that a user's preferences are expressible by a fixed unknown CP-net, henceforth called the target CP-net. A system observes (or queries) the user expressing preferences between choices (e.g., indicating which course to take, movie to rent, or web-link to follow, among pairs of such), and seeks to approximate the target CP-net. If such an approximation can be achieved reliably, efficiently, and non-intrusively, then the user may choose to employ such a system so that, after an initial training phase, the user's choices in analogous but novel future situations can be delegated to the system.

We take herein *(i)* reliability to mean the offering of *predictive guarantees* in novel situations, *(ii)* efficiency to mean the running of the *learning and reasoning* processes in time polynomial in the relevant problem parameters, and *(iii)* non-intrusiveness to mean only the *passive* observation of a user's stated preferences without the user's active involvement.

Although previous work identified formal conditions under which one can ensure such reliability, efficiency (albeit for the learning process only), and non-intrusiveness [Dimopoulos *et al.*, 2009], no systematic empirical investigation of the learning algorithm proposed therein has been performed. The present work seeks to push that empirical front. In the process of our investigation, we extend the known efficiency guarantees to the reasoning process also, by developing a novel reasoning algorithm for CP-nets. Our algorithm is shown to be sound and complete on all *transparently entailed* preferences, a natural subclass of preferences identified by previous work.

2 Background on CP-nets

A ***CP-net*** N is a graph with a vertex set $\mathcal{V} = \{X^1, \ldots, X^n\}$ corresponding to the variables over which preferences are expressed. The set $\mathrm{Pa}_N(X^t)$ of vertices with incoming edges to vertex X^t is the ***parent set*** of variable X^t according to N. Each variable X^t is associated with a ***domain*** $\mathrm{dom}(X^t) = \{x_1^t, x_2^t, \ldots\}$ determining the values the variable ranges over.

A subclass of CP-nets that has received special attention in the literature is that of acyclic CP-nets over binary attributes. We shall restrict our attention to this subclass henceforth.

For any subset $\mathcal{X} \subseteq \mathcal{V}$ of variables, $\mathrm{dom}(\mathcal{X})$ is the cross-product of the domains of variables in $\mathcal{X}$ in order of increasing index; that is, $\mathrm{dom}(\mathcal{X})$ is the set of all possible ***assignments*** of values to all variables in $\mathcal{X}$, so that each variable $X^t \in \mathcal{X}$ is mapped to a value in its domain $\mathrm{dom}(X^t)$. The assignments in $\mathrm{dom}(\mathcal{V})$ (i.e., when $\mathcal{X} = \mathcal{V}$) are also called ***outcomes***. We denote by $o_{[\mathcal{X}]} \in \mathrm{dom}(\mathcal{X})$ the assignment of values that outcome o associates with the variables $\mathcal{X} \subseteq \mathcal{V}$. For disjoint subsets of variables $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{V}$, $xy \in \mathrm{dom}(\mathcal{X} \cup \mathcal{Y})$ is the assignment obtained from the assignments $x \in \mathrm{dom}(\mathcal{X})$ and $y \in \mathrm{dom}(\mathcal{Y})$ when variables are appropriately reordered.

N associates each variable X^t with a ***conditional preference table*** $\mathrm{CPT}(X^t)$ specifying an order $\succ^t$ over $\mathrm{dom}(X^t)$ for each assignment $u \in \mathrm{dom}(\mathrm{Pa}_N(X^t))$ to X^t's parents. Although more generality is possible [Boutilier *et al.*, 2004], for our purposes it suffices to assume $\succ^t$ is total, so that either $u : x_i^t \succ^t x_j^t$ or $u : x_j^t \succ^t x_i^t$ for every $x_i^t, x_j^t \in \mathrm{dom}(X^t)$.

Example 1. *Figure 1 shows CP-net* N^**:* $\mathcal{V} = \{A, B, C, D\}$, $\mathrm{dom}(A) = \{a_1, a_2\}$ *and* $\mathrm{Pa}_N(A) = \{\}$, $\mathrm{dom}(B) = \{b_1, b_2\}$ *and* $\mathrm{Pa}_N(B) = \{A\}$, $\mathrm{dom}(C) = \{c_1, c_2\}$ *and* $\mathrm{Pa}_N(C) = \{A, B\}$, $\mathrm{dom}(D) = \{d_1, d_2\}$ *and* $\mathrm{Pa}_N(D) = \{B, C\}$.

The semantics of a CP-net can be given in terms of satisfaction of preference orderings that one defines over all possible outcomes. Instead of replicating here the relevant definitions

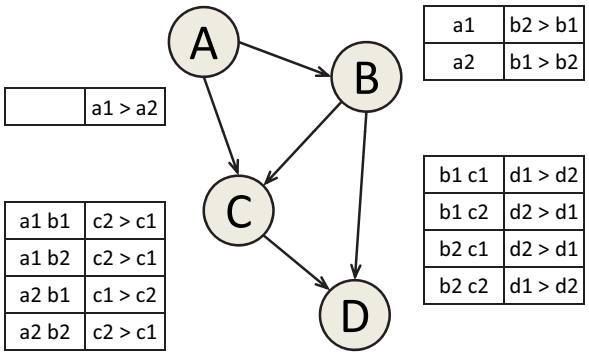

Figure 1: Graphical representation of a CP-net N^*.

[Boutilier *et al.*, 2004, Definitions 2–3], we present the semantics of a CP-net through an equivalent (for acyclic CP-nets [Boutilier *et al.*, 2004, Theorems 7–8]) procedural lens, which we shall find easier to refer to in the sequel.

A ***flipping sequence*** from o_1 to o_m with respect to a CP-net N is a sequence of outcomes $o_1, o_2, \ldots, o_m \in \mathrm{dom}(\mathcal{V})$ such that for every pair of consecutive outcomes o_s, o_{s+1} in the sequence there exists a unique variable $X^t \in \mathcal{V}$ such that $o_{s[X^t]} \neq o_{s+1[X^t]}$, and, furthermore, for this variable it holds that $o_{s[\mathrm{Pa}_N(X^t)]} : o_{s[X^t]} \succ^t o_{s+1[X^t]}$ is in $\mathrm{CPT}(X^t)$.

We can now use such flipping sequences to define when a CP-net prefers an outcome o_i over another outcome o_j, thus: A CP-net N ***entails*** a comparison $o_i \succ o_j$ if and only if there exists a flipping sequence from o_i to o_j with respect to N.

Example 2. *The CP-net N^* in Figure 1 entails the comparison $a_1b_2c_1d_1 \succ a_2b_1c_2d_2$, by the existence of the flipping sequence $a_1b_2c_1d_1 \succ a_1\underline{b_1}c_1d_1 \succ a_1b_1c_1\underline{d_2} \succ \underline{a_2}b_1c_1d_2 \succ a_2b_1\underline{c_2}d_2$; underlined is the one value that changes each time.*

Example 3. *The CP-net N^* in Figure 1 does not entail the comparison $a_2b_2c_1d_2 \succ a_2b_2c_2d_1$, by a reductio ad absurdum argument: Assume there exists a flipping sequence from $a_2b_2c_1d_2$ to $a_2b_2c_2d_1$. Then, c_1 changes to c_2 between some two outcomes in the sequence. This is possible only if a_2b_1 holds in the earlier of the two outcomes. Since a_2b_2 holds in the initial outcome, b_2 changes to b_1 between some two outcomes in the sequence. This is possible only if a_1 holds in the earlier of the two outcomes. Since a_2 holds in the initial outcome, a_2 changes to a_1 between some two outcomes in the sequence. This is, however, not possible; a contradiction.*

3 Transparent Entailment

It is known that *reasoning* with, or checking for entailment of, general comparisons in CP-nets (even if acyclic over binary attributes) is intractable [Boutilier *et al.*, 2004, Section 4.4]. The same is true for *learning* CP-nets (even if acyclic over binary attributes) from general comparisons [Dimopoulos *et al.*, 2009, Theorem 1]. The latter work has investigated a natural subclass of comparisons for which learnability is tractable.

Definition 1 (Transparent Entailment (as given in [Dimopoulos *et al.*, 2009])). *Consider a CP-net N over a set $\mathcal{V}$ of variables that entails every comparison in a set $\mathcal{P}$ of pairwise comparisons between outcomes over $\mathcal{V}$. N entails $\mathcal{P}$ **transparently** if for every $\mathbf{o_i}o_{i[\mathrm{Pa}_N(X)]}o_{i[X]} \succ \mathbf{o_j}o_{j[\mathrm{Pa}_N(X)]}o_{j[X]} \in \mathcal{P}$ with $o_{i[X]} \neq o_{j[X]}$, N entails either $\mathbf{o}o_{i[\mathrm{Pa}_N(X)]}o_{i[X]} \succ \mathbf{o}o_{i[\mathrm{Pa}_N(X)]}o_{j[X]}$ or $\mathbf{o}o_{j[\mathrm{Pa}_N(X)]}o_{i[X]} \succ \mathbf{o}o_{j[\mathrm{Pa}_N(X)]}o_{j[X]}$ for some $\mathbf{o} \in \mathrm{dom}(\mathcal{V} \setminus (\mathrm{Pa}_N(X) \cup \{X\}))$.*

In other words, a comparison $o_i \succ o_j$ is transparently entailed by a CP-net N if and only if: *(i)* $o_i \succ o_j$ is entailed by N; and *(ii)* for any variable $X^t \in \mathcal{V}$ that is assigned different values $x_i^t, x_j^t \in \mathrm{dom}(X^t)$ in o_i and o_j respectively, it is the case that $\mathrm{CPT}(X^t)$ includes the preference $u : x_i^t \succ^t x_j^t$ for an assignment $u \in \mathrm{dom}(\mathrm{Pa}_N(X^t))$ to the parents of X^t as determined by either o_i or o_j. Thus, the reason for preferring o_i over o_j is not hidden, but it is made transparent in $o_i \succ o_j$.

Example 4. *Recall from Example 2 that the CP-net N^* in Figure 1 entails the comparison $a_1b_2c_1d_1 \succ a_2b_1c_2d_2$. Looking at the flipping sequence used to establish this, one may observe that d_1 changes to d_2 at a point where its parents have values b_1c_1. Furthermore, this is true for all flipping sequences, except possibly that the parents have values b_2c_2 in them. These values are present in neither the initial nor the final outcome, but only in an intermediate (or hidden) outcome. In this sense, the comparison is not transparent to the reasons that support the change in the value of variable D, and the comparison is not transparently entailed by N^*. This is so independently of how the rest of the variables change.*

Example 5. *The CP-net N^* in Figure 1 entails the comparison $a_1b_2c_2d_1 \succ a_2b_1c_1d_1$, by the existence of the flipping sequence $a_1b_2c_2d_1 \succ a_1\underline{b_1}c_2d_1 \succ a_1b_1\underline{c_1}d_1 \succ \underline{a_2}b_1c_1d_1$; underlined is the one value that changes each time. One may observe that c_2 changes to c_1 at a point where its parents have values a_1b_1, which are present in neither the initial nor the final outcome. However, the entailment of the comparison $a_1b_2c_2d_1 \succ a_2b_1c_1d_1$ by N^* can be established through another flipping sequence $a_1b_2c_2d_1 \succ a_1b_2\underline{c_1}d_1 \succ a_1\underline{b_1}c_1d_1 \succ \underline{a_2}b_1c_1d_1$ also, in which this hiding is avoided (for all variables). Thus, the comparison is transparently entailed by N^*.*

Although the learnability of CP-nets from transparently entailed preferences has been established [Dimopoulos *et al.*, 2009, Corollary 3] under the Probably Approximate Correct semantics [Valiant, 1984], the efficient *checking* for transparent entailment of comparisons remains open. On the one hand, the transparent entailment of a comparison implies the entailment of that comparison, and checking for entailment is known [Boutilier *et al.*, 2004] to be either intractable (cf. dominance queries), or incomplete in a rather strong sense (cf. ordering queries). On the other hand, the extra structure of transparently entailed preferences makes it plausible that a more efficient decision process can be devised. We propose and formally analyze such a decision process next.

Effectively, Algorithm 1 seeks to create a flipping sequence that would support the entailment of comparison $o_i \succ o_j$ by the CP-net N. For entailment in general, the order in which variables change values in such a flipping sequence could be very involved and intractable to identify. If transparent entailment is of interest, however, Algorithm 1 prescribes a short and universal order (the existence of which is hinted in Example 5) in which variables need to be checked: first in a reverse topological order, and then in a normal topological order, as determined by the graphical representation of the CP-net N.

Algorithm 1 *reason*(CP-net N, comparison $o_i \succ o_j$)

Set $\mathcal{V}$ to include the variables over which N is defined.
Set $\mathcal{V}^{\uparrow}$ to be a reverse topological ordering of $\mathcal{V}$ w.r.t. N.
Set $\mathcal{V}^{\downarrow}$ to be a normal topological ordering of $\mathcal{V}$ w.r.t. N.
Set $\mathcal{X}$ to be the concatenation of list $\mathcal{V}^{\uparrow}$ before list $\mathcal{V}^{\downarrow}$.
Set $o_* := o_i$. * current outcome in flipping sequence *\
while $\mathcal{X} \neq \emptyset$ and $o_* \neq o_j$ **do**
  Remove from $\mathcal{X}$ the variable X^t that is ordered first.
  if $o_{*[\mathrm{Pa}_N(X^t)]} : o_{*[X^t]} \succ^t o_{j[X^t]}$ is in $\mathrm{CPT}(X^t)$ **then**
    Update o_* by replacing $o_{*[X^t]}$ with $o_{j[X^t]}$ in o_*.
  end if
end while
if $o_* = o_j$ **then**
  return *true*. * $o_i \succ o_j$ transparently entailed *\
else $\{o_* \neq o_j\}$
  return *false*. * $o_i \succ o_j$ not transparently entailed *\
end if

Example 6. *Recall from Examples 2 and 4 that the CP-net N^* in Figure 1 entails — but not transparently — the comparison $a_1b_2c_1d_1 \succ a_2b_1c_2d_2$. Looking at the flipping sequence used to establish the entailment, one may observe that the parents of variable D have their values changed both before and after D has its value changed. Such an ordering of variables is not — and rightly so — accepted by Algorithm 1.*

Example 7. *Recall from Example 5 that the CP-net N^* in Figure 1 transparently entails the comparison $a_1b_2c_2d_1 \succ a_2b_1c_1d_1$. Looking at the flipping sequence (the second in Example 5) used to establish this, one may observe that each variable that changes value does so before its parents. Such an ordering of variables is accepted by Algorithm 1.*

We establish next that Algorithm 1 is sound, complete, and computationally efficient for deciding transparent entailment.

Theorem 1 (Soundness of *reason*$(\cdot,\cdot)$). *If the call reason(N, $o_i \succ o_j$) returns true, then N transparently entails $o_i \succ o_j$.*

Proof (sketch). Entailment follows since the algorithm constructs a flipping sequence from o_i to o_j. Transparency follows since for every variable that changes value in the constructed flipping sequence, the parents of that variable change values either only before or only after the variable. □

Theorem 2 (Completeness of *reason*$(\cdot,\cdot)$). *If N transparently entails $o_i \succ o_j$, then the call reason(N, $o_i \succ o_j$) returns true.*

Proof (sketch). A change in the value of a variable that is conditioned on its parents having values as specified by o_i will be identified when traversing $\mathcal{V}^{\uparrow}$. A change in the value of any remaining variable will be identified when traversing $\mathcal{V}^{\downarrow}$. □

Theorem 3 (Efficiency of *reason*$(\cdot,\cdot)$). *The call reason(N, $o_i \succ o_j$) returns true or false in time polynomial in $size(N)$.*

Proof (sketch). Topological ordering can be carried out efficiently. The algorithm loops at most twice for each variable, and the test it performs can be carried out efficiently. □

4 Empirical Investigation

Our empirical investigation seeks to quantitatively evaluate the performance of an algorithm for learning CP-nets, known to formally guarantee that learning happens in a reliable, efficient, and non-intrusive manner [Dimopoulos *et al.*, 2009].

The algorithm first receives a training set $\mathcal{P}$ of comparisons between outcomes, that are entailed by an arbitrary target and hidden CP-net N. For each variable X, the algorithm seeks to find a set $\mathcal{U}$ of other variables on which the preferences over the values of X can be conditioned. For each candidate set $\mathcal{U}$, the algorithm translates the set $\mathcal{P}$ of comparisons into an appropriate 2-SAT formula, a solution to which prescribes the entries of $\mathrm{CPT}(X)$, with $\mathrm{Pa}_h(X) = \mathcal{U}$. Variables X are not considered in an arbitrary order, but all are considered first with $|\mathcal{U}| = 0$ and those for which a CPT can be constructed are kept; the remaining ones are considered with $|\mathcal{U}| = 1$, and so on. Once CPTs for all variables are constructed, the algorithm returns the resulting CP-net h. Pseudocode for the algorithm appears in [Dimopoulos *et al.*, 2009]. The algorithm is shown to be a PAC leaner [Valiant, 1984], in that it runs efficiently, and offers provable guarantees on the performance of the learned CP-net h in terms of its accuracy in entailing previously unseen comparisons (with N being the gold standard for evaluation). The accuracy guarantee is known to hold assuming $\mathcal{P}$ contains only transparently entailed preferences.

4.1 Methodology Used

The methodology followed in our experiments comprises the following steps: *(i)* construct a target CP-net N; *(ii)* construct a set $\mathcal{P}_{all}$ of some comparisons entailed by N; *(iii)* keep the subset $\mathcal{P}$ of comparisons in $\mathcal{P}_{all}$ that are transparently entailed by N; *(iv)* learn a CP-net h on a training subset of $\mathcal{P}$; *(v)* evaluate the performance of h on a testing subset of $\mathcal{P}$.

Constructing a Target CP-net N

This step takes as input a number n of variables, and a maximum number k of parents. It first selects each variable independently at random (with prob. 0.3) as one of the roots. If no root is chosen, the selection is repeated with successively increasing probability (+ 0.1 prob.) until it succeeds. For each remaining variable X, it selects each variable independently at random (with prob. 0.3) to become the parent of X, as long as this choice does not create cycles, nor results in X having more than k parents. Once the parents of X are chosen, its CPT is completed with uniformly at random assigned preferences. The steps returns the constructed CP-net N.

Constructing Comparisons $\mathcal{P}_{all}$

This step takes as input a number m of comparisons, a length ℓ for flipping sequences, and an "equality" bit. It constructs a comparison $o_i \succ o_j$ entailed by N by first selecting uniformly at random either i or j. It then chooses the value of o_i (resp., o_j) uniformly at random among all possible outcomes, sets $o_* := o_i$ (resp., $o_* := o_j$), and repeats the following cycle: It chooses a variable X uniformly at random, and checks whether the value of X can be changed to a less (resp., more) preferred value, given its CPT and the values that its parents have in o_*. If so, it changes the value of X in o_*, and marks the entry of the CPT that was used to support this change.

The cycle is repeated until f changes / flips have been made in o_*, at which point the step sets $o_j := o_*$ (resp., $o_i := o_*$). Thus, $N \models o_i \succ o_j$ is established by a flipping sequence of length f, and is added in $\mathcal{P}_{all}$. The value of f is such that: either it is always equal to ℓ, if the "equality" bit is set; or it is chosen uniformly at random between 1 and ℓ at the beginning of the construction of each comparison, otherwise.

If n attempts are made without managing to get f changes, then the outcome o_i (resp., o_j) is discarded and chosen again; the change "direction" (i or j) is retained as initially chosen. The step records the average number of discarded outcomes per comparison that is constructed, and the average number of flip attempts per successful flip that was not later discarded.

Once $\mathcal{P}_{all}$ is of size m, the step records the utilization (or "involvement") of N during the construction in two ways: *(i)* the percentage U_1 of marked entries over the total number of entries in N; *(ii)* the average U_2 of the percentage of marked entries in a CPT over the total number of entries in that CPT.

Keeping a Transparent Subset $\mathcal{P}$

For each comparison $o_i \succ o_j \in \mathcal{P}_{all}$, this step invokes directly the test given below Definition 1, and keeps the comparison in $\mathcal{P}$ if and only if it passes the test. Since the comparison is already known to be entailed, it remains to check the second part of the test, which can be done efficiently. Alternatively, Algorithm 1 could have been used to the same effect. When this step concludes, it records the percentage $|\mathcal{P}|/|\mathcal{P}_{all}|$ of comparisons in $\mathcal{P}_{all}$ that are transparently entailed.

Learning a Hypothesis CP-net h

Set $\mathcal{P}$ is partitioned into a training subset $\mathcal{P}_r$ and a testing subset $\mathcal{P}_e$ in a $70\% - 30\%$ random split. The training set is further randomly partitioned into subsets $\mathcal{P}_r^s$. The following is then repeated for $s = 0, 1, 2, \ldots$, in ascending order: The learning algorithm is given $\mathcal{P}_r^1 \cup \mathcal{P}_r^2 \cup \ldots \cup \mathcal{P}_r^s$ as a training set, and produces a hypothesis CP-net h^s. The resulting CP-net is evaluated on the testing set $\mathcal{P}_e$, and its performance is recorded. Thus, the learning algorithm faces larger training sets during successive training epochs, which always expand the ones given in preceding epochs. Note that the training set size across epochs need not increase linearly with s. Indeed, we have chosen to increase it much more slowly for smaller values of s (i.e., $|\mathcal{P}_r^s|$ is small), to observe in more detail the performance improvement from the extra training data. For larger values of s the performance is nearly stable, so the size increase can be more rapid without major loss of information.

Evaluating the Performance of h

Each learned CP-net h^s is evaluated to quantify the extent to which it entails comparisons also entailed by N. Since testing for entailment is generally intractable (cf. dominance queries [Boutilier *et al.*, 2004]), we are left with the choice of either *(i)* checking for entailment by the tractable but incomplete use of ordering queries, or *(ii)* checking for *transparent* entailment by the tractable and complete use of Algorithm 1.

In fact, it would suffice to do the latter, without any loss of generality. Indeed, one can observe that the result establishing the PAC learnability of CP-nets (cf. [Dimopoulos *et al.*, 2009, Corollary 3]) can be easily extended to show that the performance guarantees of the learned CP-net are not only in terms of entailment but also transparent entailment. Thus, our empirical evaluation would accurately measure what the learning algorithm is claiming to do. Nonetheless, we chose to evaluate the performance of h^s using both approaches, as a way to evaluate the approaches against each other as well.

To evaluate h^s we first present it with an *unordered* pair of outcomes o_i, o_j from the testing set. We then invoke the first approach to obtain one of the following [Boutilier *et al.*, 2004, Theorem 5] mutually exclusive predictions: $h^s \not\models o_i \succ o_j$ but no decision on $o_j \succ o_i$; $h^s \not\models o_j \succ o_i$ but no decision on $o_i \succ o_j$; $h^s \not\models o_i \succ o_j$ and $h^s \not\models o_j \succ o_i$. We also invoke the second approach to obtain one of the following mutually exclusive predictions: $h^s \models o_i \succ o_j$; $h^s \models o_j \succ o_i$; $h^s \not\models o_i \succ o_j$ and $h^s \not\models o_j \succ o_i$. We plot the accuracy of each of the six predictions against the ground truth of the testing set.

4.2 Results and Analysis

The recorded information for a variety of values to the experimental parameters is presented in Table 1 and Figures 2,3,4.

The first line of analysis of the empirical results that we pursue relates to properties of the learned CP-nets that derive from the PAC semantics enjoyed by the learning algorithm.

Observe that across all considered settings of Figure 2 and Table 1, the learned CP-net converges rather quickly to a high (above 90%) and then perfect (100%) performance on the testing set. The high speed of convergence can be best appreciated by considering the target CP-net with 20 variables associated with the top-right graph of Figure 2 and setting S_2 of Table 1. In that particular setting, high and perfect performance is achieved after only 200 and 700 training examples, respectively, even though the testing set itself contains 2829 comparisons, and there exist more than one million possible outcomes, and more than one trillion possible comparisons.

To further appreciate the efficiency, note that it is achieved despite the target CP-nets having complex structure, and being heavily utilized when constructing the set of training and testing comparisons. An illustration of the complex structure of a target CP-net can be found in Figure 3. In the right-most CP-net, one can observe: long chains between variables ($8 \rightarrow 9 \rightarrow 7 \rightarrow 5 \rightarrow 6 \rightarrow 2$); variables that act as hubs (variables 1 and 4 each have three incoming and two outgoing edges); multiple roots, leaves, and undirected cycles. The values for the utilization metrics U_1 and U_2 across all the experiments presented in Figure 2 can be seen in Table 1. This high utilization is an indication that the complex structure of the target CP-nets ends up being encoded in the constructed comparisons, which the learned CP-net needs to decode in order to achieve good performance. And it does so, quickly.

Performance of the learned CP-net is not equated with its structural distance from the target CP-net; structurally replicating the hidden CP-net is not a goal in itself. Rather, the goal is to sufficiently approximate the target CP-net in terms of the comparisons it entails. Figure 3 shows how the learned CP-net evolves in setting S_1 of Table 1, corresponding to the upper-left graph of Figure 2. The right-most CP-net of Figure 3 corresponds to the target CP-net. The first-from-left CP-net is trained on 10 instances, and although it exhibits very little structure, it achieves more than 55% performance. The second-from-left CP-net is trained on 100 instances, exhibits

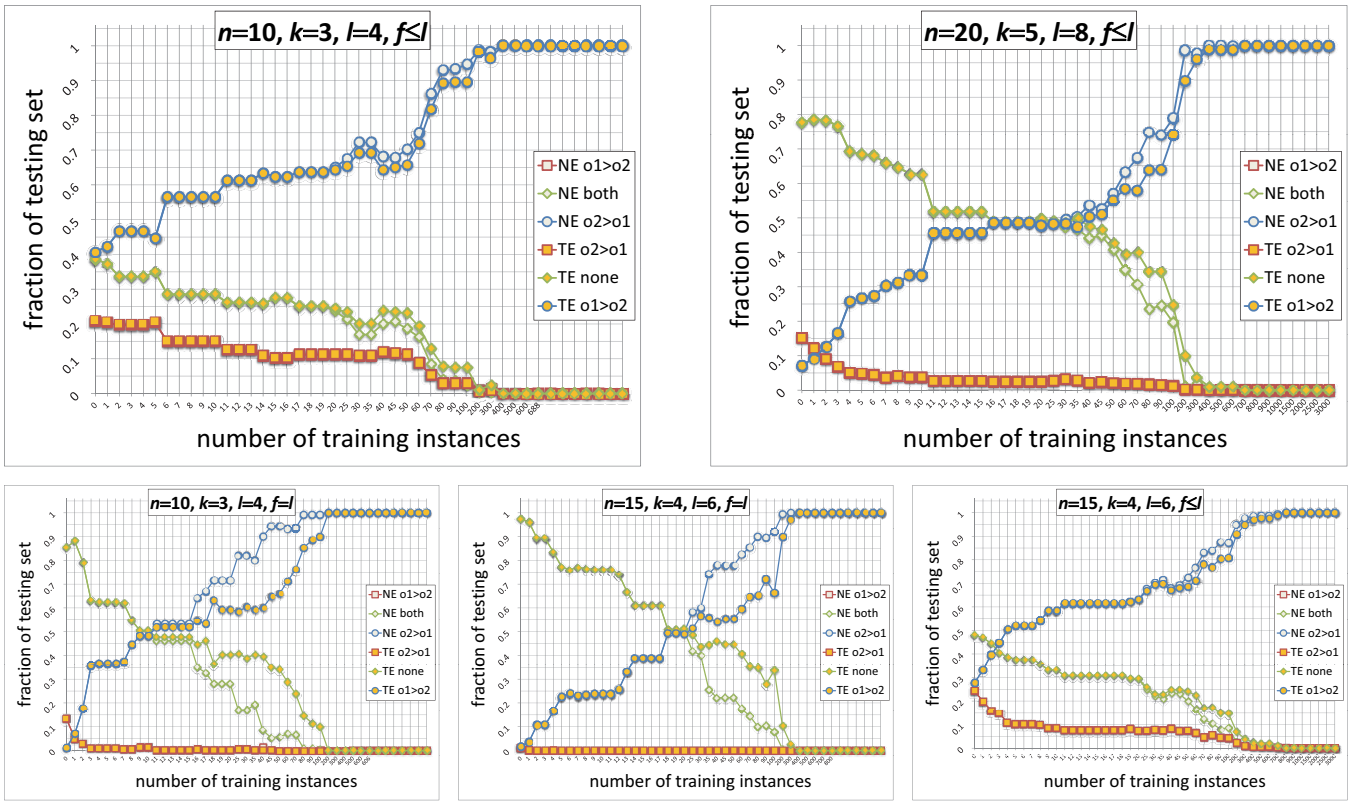

Figure 2: Performance of the learned CP-net h^s with increasing training examples, using the evaluation metrics discussed in the Methodology Section. Each graph shows the values assigned to parameters n, k, ℓ, and the "equality" bit. In the top-to-bottom order given in the legends, the first three plotted lines use ordering queries (NE means "does not entail"), and the last three plotted lines use Algorithm 1 (TE means "transparently entails"). In each case the target CP-net N is such that $N \models o_1 \succ o_2$.

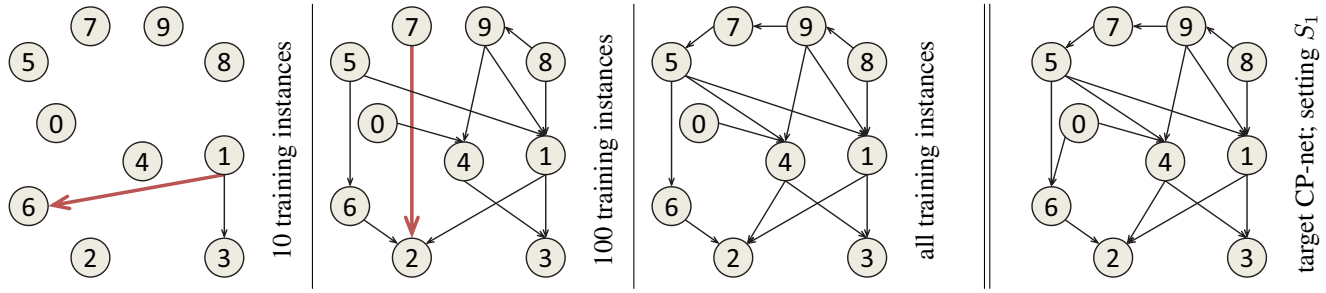

Figure 3: Structural evolution of a learned CP-net with increasing training examples, in juxtaposition to the target CP-net.

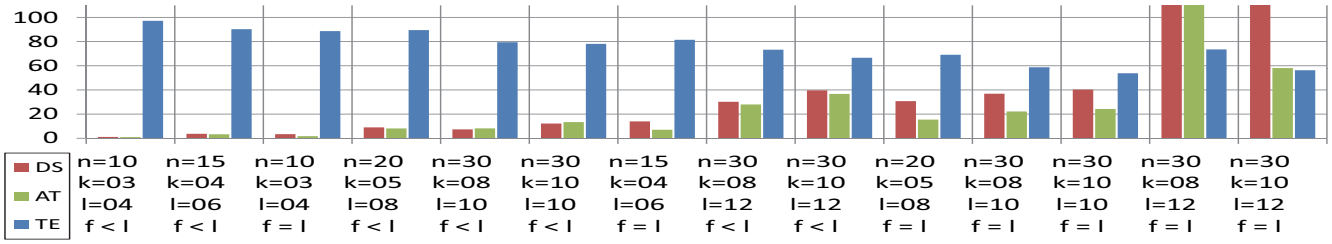

Figure 4: Characteristics of various target CP-nets while constructing 1000 entailed comparisons. DS (left columns) shows the average number of discarded outcomes for each retained comparison; scaled to $1/2$ of actual. AT (middle columns) shows the average number of attempted flips for each retained flip; scaled to $1/10$ of actual. TE (right columns) shows the percentage of comparisons that are transparently entailed. The settings are sorted by increasing expected value f of the flipping sequences.

	S_1	S_2	S_3	S_4	S_5
$\|\mathcal{P}_{all}\|$	1000	10000	1000	10000	10000
U_1	0.96	0.99	0.93	0.99	0.99
U_2	0.90	0.99	0.80	0.99	0.99
$\|\mathcal{P}\|/\|\mathcal{P}_{all}\|$	0.983	0.942	0.866	0.887	0.978
$\|\mathcal{P}_e\|$	295	2829	260	2662	2935
90% perf.	100	200	100	200	200
100% perf.	400	700	200	400	800

Table 1: The five settings considered in Figure 2 (from left to right, and from top to bottom), along with information on the utilization / performance of CP-nets in each setting. The interpretation of the first five rows of the table is as given in the Methodology Section. The last two rows report from Figure 2 the number of training examples that is sufficient (and some times overly so, due to the progressively coarser granularity of the horizontal axes in the graphs) to achieve high (above 90%) and perfect (100%) performance on the testing set.

much of the structure of the target CP-net, and achieves more than 90% performance. The third-from-left CP-net is trained on the entire training set, its structure almost replicates that of the target CP-net (except for $0 \rightarrow 6$), and achieves 100% performance. Even then, however, the learned CP-net still structurally differs from the target CP-net (due to the two CP-nets disagreeing on the entailment of comparisons that are unimportant / rare, and are not represented in the training set).

During the evolution of the learned CP-net, some spurious dependencies between variables might be included (shown with red / thick arrows in Figure 3), as they may suffice to "explain" the comparisons in the part of the training set that was observed thus far. In the process of receiving more training instances, these dependencies might be refuted, or become obsolete. In Figure 3, for instance: the dependency of variable 6 on variable 1 is removed when both 6 and 1 are found to have a common dependency on variable 5; the direct dependency of variable 2 on variable 7 is removed in the presence of an indirect dependency through the chain $7 \rightarrow 5 \rightarrow 6 \rightarrow 2$.

The second line of analysis of the empirical results that we pursue relates to the notion of transparent entailment.

Despite its seemingly restrictive nature, transparent entailment in no way restricts the complexity of the target CP-net; any target CP-net can be shown to transparently entail at least some (and, more precisely, exponentially many in n) comparisons. But how common is transparent entailment among the entailed comparisons of a target CP-net? Figure 4 presents this percentage for a variety of settings (including the ones found in Figure 2). Even as CP-nets get larger and more complex, and even as it becomes increasingly difficult to construct entailed comparisons (as shown by the increasing values of DS and AT), the percentage of the comparisons that are transparently entailed is rather high, and more so for comparisons with not too long supporting flipping sequences.

One could argue that short flipping sequences are more likely to occur in certain real-world settings, and that, by extension, the preferences with which one has to deal in such settings would be reasonably expected to often be (transparently entailed and thus) amenable to the treatment followed in this work. We shall not attempt to make this argument herein, other than state that finding empirical evidence to support it would constitute an intriguing direction for further research.

Finally, we emphasize the advantage of the new reasoning algorithm over the existing one when checking for transparent entailment. All graphs of Figure 2 show that the use of ordering queries overestimates the preference (blue / circle lines) and underestimates the indifference (green / rhombus lines) of the learned CP-net against what is the case. This departure from reality is at times rather dramatic, reaching a distance of 30 percentile units (bottom-left graph of Figure 2). Although further quantification of this phenomenon is needed, it is clear that the new algorithm is a more reliable and equally efficient alternative, when transparent entailment is concerned.

5 Conclusions

This work presents the first systematic empirical investigation of CP-net learnability when the reliability, efficiency, and non-intrusiveness of the learning process are important. It is demonstrated that the learned preferences have high predictive ability on previously unseen situations, after only a very short training phase, and without the active involvement of a user. Equally importantly, the learned preferences are shown to be amenable to efficient reasoning, so that their employment for decision support and recommendations is not merely a conceptual possibility, but a practical and realistic option.

We are aware only of one other (very recent) empirical investigation of CP-net learnability [Liu *et al.*, 2013]. In that work, the authors propose a statistical test that can be used to learn CP-nets in the presence of noisy training examples. Although that work shares the non-intrusiveness property of our investigated algorithm, it does not respect reliability and efficiency. Despite the authors' claim that their learning process is polynomial-time, it seems not to be so in the learning parameters that are typically used in Machine Learning, leading to an algorithm that is ultimately not scalable. Furthermore, the predictive ability of the learned preferences is not evaluated on a distinguished testing set, as done typically in Machine Learning, but only on the training set. Their empirical results exemplify the departure from efficiency and reliability, showing that even for tiny CP-nets over 3 or 5 variables, a few hundred training instances (which corresponds to a significant fraction of all possible training instances) is needed to get a respectable (training set) accuracy. Finally, the learned preferences (irrespectively of any concerns on how they were obtained) cannot be efficiently reasoned with, unlike our case.

Nonetheless, the aforementioned work raises two interesting points: *(i)* how can learning deal with noise, and *(ii)* how often can transparent entailment be reasonably assumed? Regarding the former point, our investigated learning algorithm relies on a reduction of training instances to satisfiability instances and the use of a SAT solver [Dimopoulos *et al.*, 2009]. Dealing with noise can, then, be accommodated by using a MAX-SAT solver and following the standard practice of identifying a best-fit hypothesis when learning from noisy data. Regarding the latter point, our empirical results already show that transparent entailment is a rather common property. Both points are certainly worth looking into as part of future work.

References

[Boutilier *et al.*, 2004] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Poole. CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research*, 21:135–191, January–June 2004.

[Dimopoulos *et al.*, 2009] Yannis Dimopoulos, Loizos Michael, and Fani Athienitou. Ceteris Paribus Preference Elicitation with Predictive Guarantees. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 1890–1895, July 2009.

[Koriche and Zanuttini, 2010] Frédéric Koriche and Bruno Zanuttini. Learning Conditional Preference Networks. *Artificial Intelligence*, 174(11):685–703, July 2010.

[Lang and Mengin, 2009] Jérôme Lang and Jérôme Mengin. The Complexity of Learning Separable Ceteris Paribus Preferences. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, pages 848–853, July 2009.

[Liu *et al.*, 2013] Juntao Liu, Zhijun Yao, Yi Xiong, Wenyu Liu, and Caihua Wu. Learning Conditional Preference Network from Noisy Samples using Hypothesis Testing. *Knowledge-Based Systems*, 40:7–16, March 2013.

[Valiant, 1984] Leslie G. Valiant. A Theory of the Learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.