Relational Networks of Conditional Preferences

Frédéric Koriche

LIRMM, Université Montpellier II, France
frederic.koriche@lirmm.fr

Abstract. Much like relational probabilistic models, the need for relational preference models naturally arises in real-world applications involving multiple, heterogeneous, and richly interconnected objects. On the one hand, relational preference models should be expressive enough to represent preferences in a compact and transparent form. On the other hand, these models should include nontrivial subclasses which are tractable for reasoning and learning purposes. In this paper, we introduce the framework of conditional preference relational networks (*CPR-nets*), which maintains the spirit of CP-nets by expressing relational preferences in a natural way using the *ceteris paribus* semantics. We show that the class of *acyclic* CPR-nets supports tractable inference for outcome optimization and ranking tasks. In addition, we show that the subclass of *tree-structured* CPR-nets is efficiently online learnable from both optimization tasks and ranking tasks.

1 Introduction

A recurrent issue in AI is the development of intelligent agents capable of tailoring their actions and recommendations to the preferences of human users. The spectrum of applications that resort on this ability is extremely wide, ranging from adaptive interfaces and configuration softwares, to recommender systems and group decision-making. In essence, the crucial ingredients for addressing this issue are *representation*, *inference* and *learning*. In complex domains, we need a representation that offers a compact encoding of preference relations defined over large outcome spaces. We also need to be able to use this representation effectively in order to answer a broad range of queries. And, since the performance of decision makers is dependent on their aptitude to reflect users' preferences, we need to be able to predict and extract such preferences in an automatic way.

Among the different preference models that have been devised in the literature, conditional preference networks (*CP-nets*) have attracted a lot of attention by providing a natural representation of qualitative preferences [1,2,5]. By analogy with Bayes nets, CP-nets are graphical structures in which nodes describe variables of interest and edges capture preferential dependencies between variables. Each node is labeled with a table expressing the preference over alternative values of the node given different values of the parent nodes under a *ceteris paribus* assumption. For example, the entry $\texttt{shirt} : \texttt{red} \succ \texttt{black} \mid \texttt{jacket} = \texttt{black}, \texttt{pants} = \texttt{black}$ might state that, all other things equal, I prefer a red shirt to a black one if the color for both the jacket and the pants is black.

Despite their popularity, CP-nets are intrinsically limited to "attribute-value" domains. Many applications, however, are richly structured, involving objects of multiple types that are related to each other through a network of different types of relations. For example, flight recommender systems are usually defined over large databases involving various entities, such as passengers, itineraries, flights, airways

and aircrafts, each entity being specified with its own attributes, and related to others using appropriate types of references. Such applications pose new challenges for devising relational preference models endowed with expressive representations, efficient inference engines, and fast learning algorithms.

In this paper, we introduce the framework of *conditional preference relational networks* (*CPR-nets*) that enhances the expressive power of CP-nets to relational domains. Briefly, a CPR-net is a template over a relational schema which specifies a ground CP-net for each particular database of objets. The template allows us to use information about a group of objects in order to derive preferences about other, related objects. Based on the ceteris paribus semantics, the representations provided by CPR-nets are *transparent*, in that a human expert can easily evaluate their meaning. For instance, in a CPR-net for flight recommendation, the entry:

$$\texttt{fromAirport(z)}:\texttt{LHR} \succ \texttt{LGW} \succ \texttt{STN} \succ \texttt{LTN} \mid \texttt{next(x,y,z)}, \texttt{toAirport(y)} = \texttt{LHR}$$

might state that, all other things being equal, if London Heathrow is the destination airport of the flight y in my itinerary x then, concerning the departure airport for the next flight z in the itinerary x, my preferences in decreasing order of priority are Heathrow, Gatwick, Stansted, and Luton.

A key property of our framework is that the class of *acyclic* CPR-nets (with constant in-degree) supports efficient inference for two sorts of reasoning tasks of practical interest. Namely, in an *outcome optimization task*, the decision maker is given a relational outcome in which several attributes are left unspecified; the goal is to find a maximally preferred extension of this partial outcome. If the CPR-net is acyclic, such an extension is unique and can be found in polynomial time. In an *outcome ranking task*, the decision maker is given a set of relational outcomes, and the goal is to rank them in some non-decreasing order of user preference. Again, such an ordering can be found in polynomial time using an acyclic CPR-net.

The learnability of CPR-nets is analyzed within the *online learning model* [3] which has become the mainstream setting for structured prediction. In this model, the decision maker observes instances of a reasoning task in a sequential manner. At trial t, the algorithm attempts to predict the solution associated with the tth instance using its current CPR-net. Once the algorithm has predicted, it receives the correct solution and incurs a loss measuring the discrepancy between its prediction and the response. The goal is to achieve a vanishing per-round regret with respect to the best CPR-net in the hypothesis class. Based on this model, we show that the class of *tree-structured* CPR-nets (with constant clause and domain size) is *efficiently learnable* from both optimization tasks and ranking tasks.

2 Language and Semantics

The reasoning problems under consideration in this study assume a prefixed and known *relational schema* which consists in a set of *object types*, a set of *value types*, a set of *attributes*, and a set of *references*. Each reference $\mathtt{r}$ is associated with a tuple of object types $dom(\mathtt{r})$ specifying its domain. Each attribute $\mathtt{a}$ is associated with a tuple of object types $dom(\mathtt{a})$ and a value type $ran(\mathtt{a})$ specifying its domain and range, respectively. Finally, each value type $\mathtt{t}$ is associated with a set of values $\mathtt{V_t}$ and an *aggregator* $\gamma_\mathtt{t} : \wp(\mathtt{V_t}) \to \mathtt{V_t}$ that maps any subset of $\mathtt{V_t}$ into an element of $\mathtt{V_t}$. In what follows, $\mathtt{V}_{ran(\mathtt{a})}$ and $\gamma_{ran(\mathtt{a})}$ are abbreviated as $\mathtt{V_a}$ and $\gamma_\mathtt{a}$. Similarly, for a set $par(\mathtt{a}) = \{\mathtt{a}_1, \cdots, \mathtt{a}_p\}$, we write $\mathtt{V}_{par(\mathtt{a})}$ as an abbreviation of $\mathtt{V}_{\mathtt{a}_1} \times \cdots \times \mathtt{V}_{\mathtt{a}_p}$, and $\gamma_{par(\mathtt{a})}$ as an abbreviation of the componentwise function $\langle \gamma_{\mathtt{a}_1}, \cdots, \gamma_{\mathtt{a}_p} \rangle$.

We assume that the arities of attributes and references are bounded by a constant k. The input dimension of the schema is thus specified by the number a of attributes, the number r of references, and the largest size d of the sets of values.

Language. Given a countable set of *variables*, each associated with an object type or a label type, the notions of *attribute atoms* and *reference atoms* are defined in the obvious way. A *conditional preference clause* for an attribute $\mathtt{a}$ with respect to a set of attributes $par(\mathtt{a}) = \{\mathtt{a}_1, \cdots, \mathtt{a}_p\}$ is an expression $\mathtt{C}_\mathtt{a}$ of the form

$$\mathtt{a}(\mathbf{x}) : \mathtt{y}_1 \succ \cdots \succ \mathtt{y}_{d_\mathtt{a}} \mid \mathtt{r}_1(\mathbf{x}_1), \cdots, \mathtt{r}_q(\mathbf{x}_q), \mathtt{a}_1(\mathbf{x}'_1) = \mathtt{y}'_1, \cdots, \mathtt{a}_p(\mathbf{x}'_p) = \mathtt{y}'_p$$

where the head is formed by a preference atom involving the attribute $\mathtt{a}$, a tuple of variables $\mathbf{x}$ of type $dom(\mathtt{a})$, and a set of $d_\mathtt{a} = |\mathtt{V}_\mathtt{a}|$ variables $\mathtt{y}_i$ of type $ran(\mathtt{a})$; the body is formed by attribute atoms over $par(\mathtt{a})$ and reference atoms. We assume that any object variable occurring in the head of the clause must also occur in its body. The *size* of $\mathtt{C}_\mathtt{a}$ is the number $p + q$ of atoms in its body. A preference clause is *acyclic* if its body is representable by a join tree.

A *conditional preference table* for $\mathtt{C}_\mathtt{a}$ is a map $cpt(\mathtt{a})$ that assigns to each tuple in $\mathtt{V}_{par(\mathtt{a})}$ a total ordering of values in $\mathtt{V}_\mathtt{a}$. Any entry of $cpt(\mathtt{a})$ can be viewed as an instance of $\mathtt{C}_\mathtt{a}$ obtained by replacing each variable $\mathtt{y}'_i$ in the body of $\mathtt{C}_\mathtt{a}$ with a value $\mathtt{v}'_i$ in $\mathtt{V}_{\mathtt{a}_i}$ and each variable $\mathtt{y}_i$ in the head of $\mathtt{C}_\mathtt{a}$ with a distinct value $\mathtt{v}_i$ in $\mathtt{V}_\mathtt{a}$.

Finally, a *conditional preference relational network* (CPR-net) is a map N that associates to each attribute $\mathtt{a}$ in the relational schema a preference clause $\mathtt{C}_\mathtt{a}$ and a preference table $cpt(\mathtt{a})$ for $\mathtt{C}_\mathtt{a}$. The *dependency graph* of N is the digraph obtained by associating an edge to each pair $(\mathtt{a}, \mathtt{a}_i)$ of attributes such that $\mathtt{a}_i$ occurs in the body of $\mathtt{C}_\mathtt{a}$. The *in-degree* of N is the maximum of the in-degrees of its dependency graph and the *clause size* of N is the maximum of the sizes of its preference clauses. N is *acyclic* if its dependency graph is acyclic and its preference clauses are all acyclic; N is *tree-structured* if it is acyclic and of in-degree 1.

Semantics. A *skeleton* for is a map K that assigns to each object type $\mathtt{t}$ in the relational schema a finite set $\mathtt{O}_\mathtt{t}$ of *objects*, and to each reference $\mathtt{r}$ with domain $\mathbf{t} = (\mathtt{t}_1, \cdots, \mathtt{t}_k)$ in the schema a subset of $\mathtt{O}_\mathbf{t} = \mathtt{O}_{\mathtt{t}_1} \times \cdots \times \mathtt{O}_{\mathtt{t}_k}$.

An *outcome* or *interpretation* for a skeleton K is a map I that extends K by assigning to each attribute $\mathtt{a}$ with domain $\mathbf{t}$ in the schema a function $\mathtt{a}_I$ from $\mathtt{O}_\mathbf{t}$ into $\mathtt{V}_\mathtt{a}$. If $\mathtt{C}_\mathtt{a}$ is a preference clause for the attribute $\mathtt{a}$ with respect to the parents $par(\mathtt{a}) = \{\mathtt{a}_1, \cdots, \mathtt{a}_p\}$, and $\mathbf{o}$ is a tuple of objects of type $dom(\mathtt{a})$, we denote by $[\mathtt{C}_\mathtt{a}(\mathbf{o})]_I$ the set of all tuples $\mathbf{v}$ in $\mathtt{V}_{par(\mathtt{a})}$ for which the existential closure of the body of $\mathtt{C}_\mathtt{a}$ grounded onto $\mathbf{o}$ and $\mathbf{v}$ is true in I.

Based on these notions, the *ceteris paribus* preference semantics for relational networks is defined as follows. Consider a CPR-net N and a skeleton K. Given an attribute $\mathtt{a}$ and a tuple of objects $\mathbf{o}$ iof type $dom(\mathtt{a})$, a pair (I, I') of interpretations for K is called a *flip* on $\mathtt{a}(\mathbf{o})$ if I and I' are everywhere identical excepted for the values $\mathtt{a}_I(\mathbf{o})$ and $\mathtt{a}_{I'}(\mathbf{o})$. A flip (I, I') on $\mathtt{a}(\mathbf{o})$ is a *model* of N if $\mathtt{a}_I(\mathbf{o})$ is preferred to $\mathtt{a}_{I'}(\mathbf{o})$ in the ordering specified by $cpt(\mathtt{a})$ for the aggregated tuple $\gamma_{par(\mathtt{a})}[\mathtt{C}_\mathtt{a}(\mathbf{o})]_I$.

By extension, any pair (I, I') of interpretations for K is a *model* of N if there is a sequence $(I_1, \cdots, I_n)$ of flips such that $I_1 = I$, $I_n = I'$, and (I_i, I_{i+1}) is a model of N for for $1 \leq i < n$. In this case, we say that I *dominates* I' in N, and write $I \succ_N I'$. A CPR-net N is *coherent* if for any skeleton K the relation $\succ_N$ is a strict partial order over the space of interpretations extending K.

Theorem 1. *Any acyclic CPR-net is coherent.*

3 Preference Reasoning

In the setting suggested by our framework, a *preference reasoning problem* consists in a class $\mathcal{N}$ of CPR-nets, a set $\mathcal{X}$ of instances, and a set $\mathcal{Y}$ of solutions. For a representation class $\mathcal{N}$, we denote by $\mathcal{N}[\boldsymbol{p}]$ the parameterized subclass of $\mathcal{N}$ where $\boldsymbol{p}$ is a tuple of parameters, referring to the language or the structure, and taken as constants. Notably, we shall concentrate here on the class $\mathcal{N}_{\mathtt{acy}}[p]$ of acyclic CPR-nets with constant in-degree p.

Lemma 1. *Let $N \in \mathcal{N}_{\mathtt{acy}}[p]$ be an acyclic CPR-net of clause size c, and K be a skeleton associating at most n objects per (object) type. Then, for any interpretation I extending K and any ground attribute $\mathtt{a}(\mathbf{o})$, the aggregated tuple $\gamma_{\mathtt{a}}[\mathtt{R}_{\mathtt{a}}(\mathbf{o})]_I$ can be computed in $\mathcal{O}(q)$ time, where $q = cd^p n^k \log_2(n^k)$.*

A *partial interpretation* is a map that extends a skeleton by associating to each attribute $\mathtt{a}$ of domain $\mathbf{t}$ a function $\mathtt{a}_I$ from $\mathtt{O}_{\mathbf{t}}$ into $\mathtt{V}_{\mathtt{a}} \cup \{*\}$. The symbol $*$ refers to the *unknown value* and captures the fact that some ground attributes may not be observed. The spaces of partial interpretations and total interpretations defined over any skeleton are denoted $\mathcal{I}^*$ and $\mathcal{I}$, respectively. For a partial interpretation $I \in \mathcal{I}^*$, a *completion* of I is a total interpretation $J \in \mathcal{I}$ that replaces each unknown value in I with a value of appropriate type. An *outcome optimization problem* is a tuple $(\mathcal{N}, \mathcal{I}^*, \mathcal{I})$. Given a CPR-net N and a partial interpretation I, the task is to find a completion J of I which is maximally preferred with respect to $\succ_N$.

For acyclic networks, the optimal extension of any partial outcome I is unique and can be found using the following greedy algorithm: we first construct a topological ordering of the a attributes in the dependency graph of N; then, starting from $J = I$, we instantiate each ground attribute $\mathtt{a}(\mathbf{o})$ in turn to its maximal value in $cpt(\mathtt{a})$ given the known parent assignment $\gamma_{par(\mathtt{a})}[\mathtt{C}_{\mathtt{a}}(\mathbf{o})]_J$.

Theorem 2. *Let $N \in \mathcal{N}_{\mathtt{acy}}[p]$ be an acyclic CPR-net of clause size c and $I \in \mathcal{I}^*$ a partial outcome with at most n objects per type. Then, the optimal completion of I with respect to N can be inferred in $\mathcal{O}(an^k q)$ time.*

An *outcome set* is a collection of interpretations extending the same skeleton K. Let $\mathcal{I}_m$ be the space of all outcome sets of size m, and $\mathbb{S}_m$ denote the symmetric group of all permutations over m elements. Then, an *outcome ranking problem* is a tuple $(\mathcal{N}, \mathcal{I}_m, \mathbb{S}_m)$. Given a CPR-net N and an outcome set S, the task is to find a permutation $\pi(S)$ which is consistent with N. Namely, $\pi(S)$ is *consistent* with N if, for any pair $\{I, J\}$ in S, I occurs before J in $\pi(S)$ whenever $I \succ_N J$.

For acyclic networks, such a permutation can be found in polynomial time using the following algorithm: we construct a digraph G over S for which the edge set is initialized to the empty set. For each pair $\{I, J\}$ of outcomes in S, we identify the set A of ground atoms $\mathtt{a}(\mathbf{o})$ such that $\gamma_{par(\mathtt{a})}[\mathtt{C}_{\mathtt{a}}(\mathbf{o})]_I = \gamma_{par(\mathtt{a})}[\mathtt{C}_{\mathtt{a}}(\mathbf{o})]_J$. If for all $\mathtt{a}(\mathbf{o}) \in A$, $\mathtt{a}_I(\mathbf{o})$ is preferred to $\mathtt{a}_J(\mathbf{o})$ in the table $cpt(\mathtt{a})$, then we expand G with (I, J). Dually, if for all $\mathtt{a}(\mathbf{o}) \in A$, $\mathtt{a}_J(\mathbf{o})$ is preferred to $\mathtt{a}_I(\mathbf{o})$, then we expand G with (J, I). Because N is acyclic, G is guaranteed to be acyclic. This, together with the fact that the projection of $\succ_N$ onto S is a subset of G, implies that any linear extension of G is consistent with $\succ_N$.

Theorem 3. *Let $N \in \mathcal{N}_{\mathtt{acy}}[p]$ be an acyclic CPR-net of clause size c and $S \in \mathcal{I}_m$ be an outcome set defined over a skeleton with at most n objects per type. Then, finding a permutation of S consistent with N can be done in $\mathcal{O}(am^2 n^k q)$ time.*

4 Preference Learning

By extending our previous considerations, a *preference learning problem* consists in a class $\mathcal{N}$ of CPR-nets, a space $\mathcal{X}$ of instances, a space $\mathcal{Y}$ of solutions, and a nonnegative and bounded *loss function* $\ell : \mathcal{N} \times \mathcal{X} \times \mathcal{Y} \to [0, \lambda]$. In the online learning model, the decision maker is a learning algorithm that observes instances in a sequence of trials. At trial t, the algorithm first receives an instance $x_t \in \mathcal{X}$ and is then required to predict a solution $\hat{y}_t \in \mathcal{Y}$ associated to this instance using its current hypothesis $N_t \in \mathcal{N}$. Once the algorithm has predicted, it receives the correct solution $y_t \in \mathcal{Y}$ and incurs a loss $\ell(N_t; x_t, y_t)$ which assesses the quality of the hypothesis N_t on the example (x_t, y_t). In light of this information, the decision maker is allowed to choose a new hypothesis, possibly using a randomized strategy.

As a common thread in the analysis of online algorithms, we make no assumptions regarding the sequence of examples. In this general setting, the performance of the algorithm is measured relatively to the performance of the best hypothesis in $\mathcal{N}$. Namely, the *regret* of an online algorithm with respect to a sequence $\{(x_t, y_t)\}$ of examples is given by the difference between the expected cumulative loss of the algorithm $\mathbb{E}[\sum_t \ell(N_t; x_t, y_t)]$ and the cumulative loss of the best hypothesis $\inf_{N \in \mathcal{N}} \sum_t \ell(N; x_t, y_t)$ chosen with the benefit of hindsight.

A class of hypotheses $\mathcal{N}$ is *online learnable* with respect to a class of reasoning tasks $(\mathcal{X}, \mathcal{Y}, \ell)$ if there exists an algorithm A such that, for any sequence of T examples, the regret of A is *sublinear* as a function of T. This condition implies that "on the average" the algorithm performs as good as the best fixed hypothesis in hindsight. If, in addition, the computational complexity of A is polynomial in the parameters associated to $\mathcal{N}$, $\mathcal{X}$, $\mathcal{Y}$ and ℓ, then $\mathcal{N}$ is *efficiently* learnable.

The rest of this section is devoted to the class $\mathcal{N}_{\mathtt{tree}}[c, d]$ of tree-structured CPR-nets of constant clause size c defined over a relational schema for which the number of values per domain is bounded by the constant d. For this class, the dependency graph of any network is a directed forest over a attributes or, equivalently, a directed tree with fixed root over $a + 1$ attributes obtained by adjoining a new attribute $\top$ (with domain $\mathtt{V}_\top = \{1\}$) as the fixed root. Any network N in $\mathcal{N}_{\mathtt{tree}}$ is encoded as a set of entries of the form $\langle \mathtt{C_a}, v', \pi(\mathtt{V_a}) \rangle$, where $\mathtt{C_a}$ is a preference clause for some attribute $\mathtt{a}$ and $\pi(\mathtt{V_a})$ is a permutation of the values in $\mathtt{V_a}$. If $\mathtt{C_a}$ includes a parent $\mathtt{a}'$ of $\mathtt{a}$ then $\mathtt{v}'$ is a value in $\mathtt{V}_{\mathtt{a}'}$; otherwise $\mathtt{v}' = 1$. Let $\mathcal{E}_{\mathtt{tree}}$ denote the set of all distinct entries which may occur in any network of $\mathcal{N}_{\mathtt{tree}}$. We say that a loss function ℓ is *linear* for $\mathcal{N}_{\mathtt{tree}}$ if $\ell(N; x, y) = \sum_{e \in N} \ell(e; x, y)$ where $\ell(e; x, y)$ is the loss incurred by the entry e on the example (x, y).

Our online learning algorithm is based on a variant the Hedge algorithm [4] which is adapted to structured concepts with linear loss functions. This variant is often referred as to the *Expanded Hedge* algorithm [6]. The overall idea is to maintain a loss vector L_t over $\mathcal{E}_{\mathtt{tree}}$ which is initially set to $L_1(e) = 0$. At trial t, the algorithm predicts with a network N_t chosen at random according to the probability distribution $\mathbb{P}_t(N) \sim \exp[-\sum_{e \in N} L_t(e)]$. At the end of the trial, the loss vector L_t is updated using the rule $L_{t+1}(e) = L_t(e) + \eta_t \ell(e; x_t, y_t)$, where η_t is an adaptive learning rate.

Lemma 2. *The regret of the Expanded Hedge algorithm for the class $\mathcal{N}_{\mathtt{tree}}[c, d]$ with respect to any loss function that is linear for $\mathcal{N}_{\mathtt{tree}}$ is bounded by*

$$\lambda \sqrt{\frac{\ln |\mathcal{N}_{\mathtt{tree}}|}{T}} \quad \textit{where } |\mathcal{N}_{\mathtt{tree}}| \leq (a+1)^{a-1} a^{a^2 r^c} (d!)^d$$

Because $\mathcal{N}_{\mathtt{tree}}[c,d]$ grows exponentially with the number of attributes and the number of references, the key computational difficulty is to generate hypotheses at random according to the distribution specified by the Expanded Hedge algorithm.

This difficulty can be circumvented using the following strategy. Let G be a (multi-)digraph for which the nodes include $\top$ and the attributes in the schema. For each clause $\mathtt{C_a}$ occurring in any entry of $\mathcal{E}_{\mathtt{tree}}$, the graph G includes a directed edge labeled by $\mathtt{C_a}$. Namely, if $\mathtt{a}$ has a parent $\mathtt{a}'$ in $\mathtt{C_a}$ then the edge of $\mathtt{C_a}$ is $(\mathtt{a},\mathtt{a}')$; otherwise, the edge is $(\mathtt{a},\top)$. Now, suppose that we wish to generate a hypothesis N_t according to $\mathbb{P}(N_t) \sim \exp[-\sum_{e\in N_t} L_t(e)]$. Let w_t be the weight vector over $\mathcal{E}_{\mathtt{tree}}$ defined by $w_t(e) = \exp(-L_t(e))$. The weight of each edge labeled by $\mathtt{C_a}$ in the graph G is given by $w_t(\mathtt{C_a}) = \sum_{\mathtt{v}'} \prod_{\pi(\mathtt{V_a})} w_t\langle \mathtt{C_a}, \mathtt{v}', \pi(\mathtt{V_a})\rangle$. Importantly, $w_t(\mathtt{C_a})$ is equal to the sum of weights of all preference tables for $\mathtt{C_a}$. Based on the Matrix-Tree Theorem [8], a random spanning tree of G can be generated in time polynomial in the size of G [7]. If $\mathtt{C}_a$ is a clause selected by the tree, then the corresponding table $cpt(\mathtt{a})$ in N is filled by picking, for each value $\mathtt{v}' \in \mathtt{V_{a'}}$, a permutation $\pi(\mathtt{V_a})$ at random according to the projection of w_t onto the entries containing $\mathtt{C}_a$ and $\mathtt{v}'$.

Lemma 3. *During each trial t of the Expanded Hedge algorithm, any hypothesis $N_t \in \mathcal{N}_{\mathtt{tree}}[c,d]$ can be generated at random in $\mathcal{O}(a^5 r^c)$ time.*

We now have all ingredients in hand to derive the main learnability results. Let $(\mathcal{I}^*, \mathcal{I}, \ell_{\mathtt{opt}})$ be the class of optimization tasks where $\ell_{\mathtt{opt}}$ is decomposed as follows: if $e = \langle \mathtt{C_a}, \mathtt{v}', \pi(\mathtt{V_a})\rangle$, then $\ell_{\mathtt{opt}}(e; I, J) = 1$ if there is a tuple of objects $\mathbf{o}$ such that $\gamma_{par(\mathtt{a})}[\mathtt{C_a}(\mathbf{o})]_J = \mathtt{v}'$, $\mathtt{a}_I(\mathbf{o}) = *$ and $\mathtt{a}_J(\mathbf{o})$ is suboptimal with respect to $\pi(\mathtt{V_a})$. Otherwise $\ell_{\mathtt{opt}}(e; I, J) = 0$. Note that $\ell_{\mathtt{opt}}$ is bounded by $\lambda = a$.

Theorem 4. *$\mathcal{N}_{\mathtt{tree}}[c,d]$ is efficiently learnable with respect to $(\mathcal{I}^*, \mathcal{I}, \ell_{\mathtt{opt}})$.*

Let $(\mathcal{I}_m, \mathbb{S}_m, \ell_{\mathtt{rank}})$ be the class of ranking tasks for which $\ell_{\mathtt{rank}}$ is decomposed as follows: if $S = \{I, J\}$ and $\pi(S) = (J, I)$, then $\ell_{\mathtt{rank}}(e; S, \pi(S)) = 1$ if there is a tuple $\mathbf{o}$ such that $\gamma_{par(\mathtt{a})}[\mathtt{C_a}(\mathbf{o})]_I = \gamma_{par(\mathtt{a})}[\mathtt{C_a}(\mathbf{o})]_J$ and $\mathtt{a}_I(\mathbf{o})$ is preferred to $\mathtt{a}_J(\mathbf{o})$ in $\pi(\mathtt{V_a})$. Otherwise, $\ell_{\mathtt{rank}}(e; S, \pi(S)) = 0$. The case $\pi(S) = (I, J)$ is handled symmetrically. By extending $\{I, J\}$ to an outcome set S of size m and applying the same policy to each pair in S, the loss $\ell_{\mathtt{rank}}$ is bounded by $\lambda = am^2$.

Theorem 5. *$\mathcal{N}_{\mathtt{tree}}[c,d]$ is efficiently learnable with respect to $(\mathcal{I}_m, \mathbb{S}_m, \ell_{\mathtt{rank}})$.*

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. J. Artif. Intell. Res. 21, 135–191 (2004)
2. Brafman, R.I., Domshlak, C., Shimony, S.E.: On graphical modeling of preference and importance. J. Artif. Intell. Res. 25, 389–424 (2006)
3. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge (2006)
4. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55(1), 119–139 (1997)
5. Goldsmith, J., Lang, J., Truszczynski, M., Wilson, N.: The computational complexity of dominance and consistency in CP-nets. J. Artif. Intell. Res. 33, 403–432 (2008)
6. Koolen, W.M., Warmuth, M.K., Kivinen, J.: Hedging structured concepts. In: Proceedings of the 23th Conference on Learning Theory (COLT'10). pp. 93–105 (2010)
7. Kulkarni, V.G.: Generating random combinatorial objects. J. Algorithms 11(2), 185–207 (1990)
8. Tutte, W.T.: Graph Theory. Cambridge (1984)