Hierarchical CP-networks

Denis Mindolin
Dept. of Computer Science and Engineering,
University at Buffalo,
Buffalo, NY 14260-2000
mindolin@cse.buffalo.edu

Jan Chomicki
Dept. of Computer Science and Engineering,
University at Buffalo,
Buffalo, NY 14260-2000
chomicki@cse.buffalo.edu

ABSTRACT

We present here a variant of acyclic CP-networks. It allows not only finite but also infinite domain attributes. It also has the property that a preference over each attribute in the network has higher priority then all the descendants' preferences. We provide an algorithm of constructing a preference formula representing the order induced by a hierarchical CP-network, thus making it possible to work with hierarchical CP-networks in the database context. We also provide a complexity analysis of the size of preference formula constructed by the algorithm.

1. INTRODUCTION

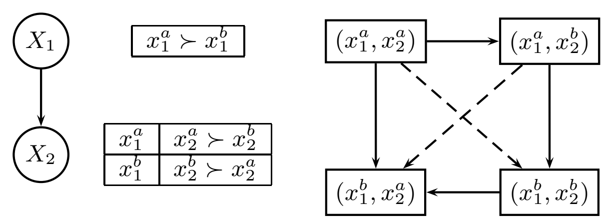
In making any kind of choice in the everyday's life, the notion of *preference* always comes to mind. A number of preference handling models have been developed. Two very popular ones are the *CP-network model*[1] and the *binary relation model*[4, 9].

Being very general, the *binary relation framework* can be used in different contexts like preference construction [9] or preference modification [6]. Moreover, the power of relational databases can be used to find the optimal outcomes of preferences represented as binary relations [4, 9]. At the same time, the *CP-network* model is very simple and intuitive. It represents a complex preference over objects using a set of atomic preferences each of which is a preference over a single object attribute given that the values of the other attributes are equal (the *ceteris paribus* principle). This set of atomic preferences is represented as a directed graph (sometimes called *the preference graph*) whose nodes are atomic preferences, and edges between nodes correspond to *conditional preferences* over attributes - i.e. the values of the parent attributes influence the preferences over the child attributes.

The *hierarchical CP-network* model, which is introduced in this paper, addresses some semantical and representational limitations of CP-networks: 1) it allows continuous attributes by representing conditional preference tables as binary preference relations, 2) the attributes in hierarchical CP-networks are *prioritized*: any attribute in preference graph is more important than its descendants; 3) the *ceteris paribus* principle can be selectively relaxed.

According to [1], in some CP-network instances edges between attributes in preference graph correspond to attribute priorities.

EXAMPLE 1. *Let a CP-net N_1 over the problem with two attributes $\mathcal{X} = \{X_1, X_2\}$ be defined as in Figure 1.*



(a) CP-network N_1

(b) The preference order implied by N_1. The preferences obtained from the CPTs are shown as solid arcs. The entailed preferences are shown as dashed arcs.

Figure 1: CP-net N_1 from Example 1

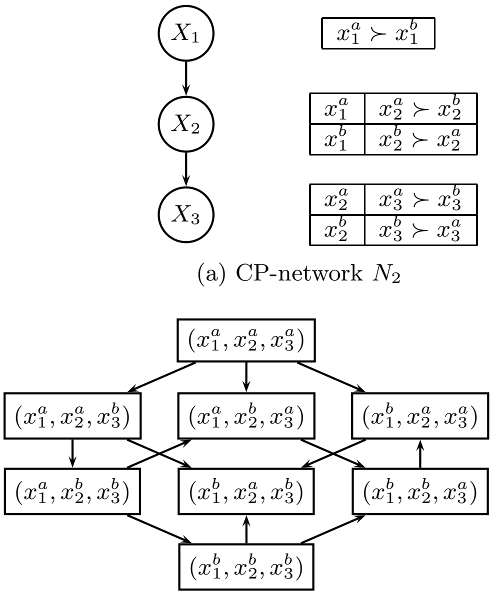
The CP-net N_1 induces a total ordering of all the outcomes (Figure 1.b) which implies that the most important is to satisfy the preferences over the attribute X_1, then over the attribute X_2 (the outcomes with values $X_1 = x_1^a$ are always preferred to those with $X_1 = x_1^b$).

However, attribute prioritization does not hold for all CP-network instances. In particular, it is not always the case that violation of a preference is worse than violation of *two or more* descendant preferences.

EXAMPLE 2. *Let CP-net N_2 over the problem with three attributes $\mathcal{X} = \{X_1, X_2, X_3\}$ be defined as in Figure 2. The entailed preference arcs are skipped in Figure 2.b for simplicity. However they can be obtained by performing the transitive closure of the graph.*

Note that in N_2, (x_1^a, x_2^b, x_3^a) is not preferred to (x_1^b, x_2^b, x_3^b) even though the the value x_1^a of X_1 is preferred to x_1^b.

VLDB '07, September 23-28, 2007, Vienna, Austria.



(a) CP-network N_2

(b) The preference order implied by CP-network N_2

Figure 2: CP-net N_2 for Example 2

Models in which descendant preferences are less important then their ancestors are natural for frameworks in which preferences are iteratively constructed by a user in a top-down manner; namely, when a new preference is added as a leaf node or a node starting a new preference subgraph. In this case, the nodes added as less important preferences (leaf nodes) are guaranteed not to violate more important, already introduced ancestor preferences.

Another drawback of CP-networks [12] is that sometimes the *ceteris paribus* semantics is too strict, and when one introduces a preference over an attribute, it's not always possible to say that this preference has to be of the *everything else being equal* kind.

EXAMPLE 3. *Assume a person wants to buy a car. Let a car seller have a database of cars which are described by the attributes* $\{make, year, price, mileage, engine\text{-}type\}$. *At the same time, the person only cares about the attributes* $\{make, year, price, mileage\}$. *However, in the CP-network framework there is no way to specify that engine-type is irrelevant: if a CP-network has a conditional preference table over engine-type, then engine-type is relevant; if it does not, then one outcome will be preferred to another according to the CP-network only if the engine-types of the outcomes are the same.*

According to the definition of CP-networks [1], each attribute involved in a CP-network is categorical. At the same time, there are many problems in which continuous attributes arise and for which it would be useful to have an approach similar to CP-networks.

Due to these limitations, the preferences from Example 4 below cannot be represented as a CP-network. However, they can be represented using hierarchical CP-networks, as we show in Example 5.

EXAMPLE 4. *Assume a user wants to buy a car, and her preference over* make *has the same importance as the preference over* year, price, *and* mileage. *At the same time, assume that* year *is more important than* price *and* mileage.

Let the preference be constructed in a top-down manner from more to less important variables. So the first preference introduced by the user is over make*: given two cars with the same age, price and mileage, she prefers VW to Kia, and Kia to all the other makes. The second preference is over* year*: given two cars of the same make, she would buy the newer one. The third preference is over* mileage*: if the makes, ages, and prices are the same, but the cars are relatively new (year* ≥ 2004*), she prefers the one with less mileage (less than 60000). However, if the makes, ages, and prices are the same, but the cars are old (year* < 2004*), she prefers the one with the mileage less than 80000. The last preference is over* price*: if two cars have the same make, age, and mileage, but are new (year* ≥ 2006*), she prefers the cheaper one. However, given two cars with the same make, age, and mileage, but not new (year* < 2006*), she prefers to spend not more then 80000 on it.*

The paper is organized as follows. Section 2 contains the definition of *hierarchical CP-networks* and a discussion of some of their properties. In Section 3, we describe an algorithm constructing a preference formula representing the order induced by a hierarchical CP-network. This allows to work with hierarchical CP-networks in the database context. Section 4 contains the complexity analysis of the preference formula constructed by the algorithm. Section 5 concludes the paper with a discussion of related and future work.

2. NOTIONS

In this paper we adopt the notations from both approaches: *CP-networks* and *binary preference relations*.

Assume we have a problem over outcomes described by n *attributes* $\mathcal{X} = \{X_1, ..., X_n\}$ such that each attribute X_i is associated with a domain $\mathcal{D}(X_i)$ (categorical or continuous). Let us denote the *set of all possible outcomes* as $\mathcal{D}$

$$\mathcal{D} = \mathcal{D}(X_1) \times ... \times \mathcal{D}(X_n),$$

and the set of all possible *assignments* to the set of attributes $U \subseteq \mathcal{X}$ as

$$\mathcal{D}(U) = \prod_{X \in U} \mathcal{D}(X).$$

Then given an outcome $o \in \mathcal{D}$, we denote the value of an attribute $X \in \mathcal{X}$ of o as $o.X$. A relation instance is a finite set of outcomes.

We limit our attention to preference relations that are strict partial orders (SPOs): transitive and irreflexive binary relations.

2.1 Hierarchical CP-network

Let a *conditional preference table* $CPT(X)$ associated with an attribute $X \in \mathcal{X}$ be defined as a triple

$$CPT(X) = (\Phi_X, W_X, U_X)$$

in which $W_X \subset \mathcal{X}$, $U_X \subset \mathcal{X}$ such that $U_X \cap W_X = \emptyset$, $X \notin U_X, X \notin W_X$, and $\Phi_X = \{\varphi_1, ..., \varphi_k\}$.

Let φ be defined as $u_\varphi : R_\varphi$ where u_φ is a relation such that $u_\varphi \subseteq \mathcal{D}(U_X)$, R_φ is a strict partial order over $\mathcal{D}(X)$, and for all pairs of different $\varphi_1, \varphi_2 \in \Phi_X$, we have $u_{\varphi_1} \cap$

$u_{\varphi_2} = \emptyset$. Let P_{u_φ} be a *finite* formula representing the relation u_φ, and a P_{R_φ} be a *finite* formula representing the relation R_φ. In the complexity analysis, we will assume that P_{u_φ} and P_{R_φ} are such that they can be evaluated in polynomial time for given valuations of the free variables.

Define for each φ a binary relation over outcomes

$$\varphi^* = \{(uxwy, ux'w'y) : u \in \mathcal{D}(U_X), u_\varphi(u);\ w, w' \in \mathcal{D}(W_X);\ y \in \mathcal{D}(Y_X); (x, x') \in R_\varphi\}$$

where $Y_X = \mathcal{X} - (\{X\} \cup W_X \cup U_X)$. We define the binary relation induced by conditional preference table $CPT(X)$ as $CPT^*(X) = \bigcup_{\varphi \in \Phi_X} \varphi^*$. Thus from the expression for φ^* it follows that the preference over attribute X is conditionally dependent on the attributes from U_X. Moreover, the preference over the attribute X is independent of the attributes from W_X. In other words, instead of the CP-network principle *everything else being equal* we use the *everything else being equal except for the attributes* W_X principle which was introduced in [12].

Summarizing the notation introduced so far: U_X, W_X, and Y_X are such that 1) $U_X \cap W_X = \emptyset$; 2) $X \notin U_X$, $X \notin W_X$; 3) $Y_X = \mathcal{X} - (\{X\} \cup W_X \cup U_X)$.

PROPOSITION 1. *φ^* and $CPT^*(X)$ are strict partial orders.*

DEFINITION 1. *Let Γ be a set of conditional preference tables for the attributes $\mathcal{X}_\Gamma \subseteq \mathcal{X}$*

$$\Gamma = \{CPT(X) : X \in \mathcal{X}_\Gamma\}.$$

We define the preference order $\succ_\Gamma$ induced by Γ *as the transitive closure of* $\bigcup_{CPT(X)\in\Gamma} CPT^*(X)$

$$\succ_\Gamma \equiv TC(\bigcup_{CPT(X)\in\Gamma} CPT^*(X)).$$

Note that our definition of the order induced by Γ is different from the definition of the order *entailed* by CP-network [1].

DEFINITION 2. [1] *Let Γ be a set of conditional preference tables. Let $\succ$ be a total order over $\mathcal{D}$. Then $\succ$ is said to* satisfy $CPT(X)$ *if for all $o, o' \in \mathcal{D}$: $(o, o') \in CPT^*(X)$ implies $o \succ o'$. A total order $\succ$ is said to* satisfy Γ *if it satisfies every $CPT(X) \in \Gamma$.*

PROPOSITION 2. *Let* the order $\succ'_\Gamma$ entailed by the hierarchical CP-network Γ *be the intersection of all total orders satisfying Γ. Then the order* $\succ_\Gamma$ induced *by Γ and the order* $\succ'_\Gamma$ entailed *by Γ are equivalent.*

Proof: *Prove that every linear extension of $\succ_\Gamma$ is a total order satisfying Γ and vice versa. Let $\succ$ be a linear extension of $\succ_\Gamma$, i.e. $TC(\bigcup_{X\in\mathcal{X}_\Gamma} CPT^*(X)) \subseteq \succ$. It implies $\bigcup_{X\in\mathcal{X}_\Gamma} CPT^*(X) \subseteq \succ$. Thus $\succ$ satisfies Γ.*

Let $\succ$ be a total order satisfying Γ, i.e. $\bigcup_{X\in\mathcal{X}_\Gamma} CPT^*(X) \subseteq \succ$. $\succ$ being an SPO implies $TC(\bigcup_{X\in\mathcal{X}_\Gamma} CPT^*(X)) \subseteq \succ$. Therefore $\succ$ is a linear extension of $\succ_\Gamma$.

We define now hierarchical CP-networks. Let $H(X) = \{(Y, X) : Y \in U_X\}$, where $H(X)$ can be viewed as a directed graph with incoming edges going from the attributes $Y \in U_X$ to a single attribute X. These edges correspond to the conditional preference of attribute X on attributes $Y \in U_X$. Then we define the *preference graph of* Γ as $H_\Gamma = \bigcup_{X\in\mathcal{X}_\Gamma} H(X)$.

Working with preference graph H_Γ, let us use the following notation: $Anc_\Gamma(X) = \{$ ancestors of X in H_Γ $\}$, $Anc\text{-}self_\Gamma(X) = \{$ ancestors of X in H_Γ or X $\}$, $Desc_\Gamma(X) = \{$ descendants of X in H_Γ $\}$, $Sibl_\Gamma(X) = \mathcal{X}_\Gamma - (Desc_\Gamma(X) \cup Anc_\Gamma(X) \cup \{X\})$, $Pa_\Gamma(X) = \{$ parents of X in $H_\Gamma\}$, $Ch_\Gamma(X) = \{$ children of X in H_Γ $\}$.

DEFINITION 3. *A set of conditional preference tables Γ is called a* hierarchical CP-network *if*

1. *for every X in $\mathcal{X}_\Gamma$, $W_X = \bigcup_{Z\in Ch_\Gamma(X)}(\{Z\} \cup W_Z)$;*
2. *if an attribute X has no child attributes in H_Γ, then $W_X = \mathcal{X} - \mathcal{X}_\Gamma$.*

From Definition 3 it follows that for each attribute X in the preference graph of a hierarchical CP-network Γ, $W_X = Desc_\Gamma(X) \cup (\mathcal{X} - \mathcal{X}_\Gamma)$. As a result, a preference over any attribute is more important than the preferences over this attribute's descendants in H_Γ.

Definition 3 also implies that a preference graph H_Γ of a hierarchical network is acyclic, otherwise W_X of some attribute X involved in a cycle would contain attribute X leading to a contradiction.

The formula representation P_{φ^*} of φ^* is defined as

$$P_{\varphi^*}(o, o') = [\bigwedge_{Z\in Anc(X)\cup Sibl(X)} o.Z = o'.Z] \wedge P_{u_\varphi}(o.U_X) \wedge P_{u_\varphi}(o'.U_X) \wedge P_{R_\varphi}(o.X, o'.X),$$

according to Definition 3. The formula representation $P_{CPT^*(X)}$ of CPT^* is defined as

$$P_{CPT^*(X)}(o, o') = \bigvee_{\varphi\in\Phi_X} P_{\varphi^*}(o, o'),$$

or using the expression for $P_{\varphi^*}(o, o')$,

$$P_{CPT^*(X)}(o, o') = [\bigwedge_{Z\in Anc(X)\cup Sibl(X)} o.Z = o'.Z] \wedge \bigvee_{\varphi\in\Phi_X} P_{u_\varphi}(o.U_X) \wedge P_{u_\varphi}(o'.U_X) \wedge P_{R_\varphi}(o.X, o'.X).$$

Let

$$Q_{CPT^*(X)}(o, o') = \bigvee_{\varphi\in\Phi_X} P_{u_\varphi}(o.U_X) \wedge P_{u_\varphi}(o'.U_X) \wedge P_{R_\varphi}(o.X, o'.X).$$

Then

$$P_{CPT^*(X)}(o, o') = [\bigwedge_{Z\in Anc(X)\cup Sibl(X)} o.Z = o'.Z] \wedge Q_{CPT^*(X)}(o, o').$$

Note that since P_{u_φ} and P_{R_φ} are finite formulas and can be evaluated in polynomial time, P_{φ^*}, $P_{CPT^*(X)}$, and $Q_{CPT^*(X)}$ are finite formulas and can be evaluated in polynomial time as well. We introduce formula $Q_{CPT^*(X)}$ here since it plays an important role further in the paper.

In our model, a conditional preference table $CPT(X)$ is graphically represented as a two-column table in which a row corresponds to a single $\varphi \in \Phi_X$. The first column of each row holds the formula P_{u_φ}, and the second column holds the formula P_{R_φ}. Compared to conditional preference tables of the traditional CP-network model, hierarchical CPTs differ in the following:

- the first column of a CP-net CPT is required to store a single assignment to U_X, while a hierarchical CPT has there a formula P_{u_φ}. As a result, since P_{u_φ} represents the relation u_φ, hierarchical CPTs can be defined for infinite domain attributes U_X;

- the second column of a CP-net $CPT(X)$ is required to store a total order of $\mathcal{D}(X)$. In our approach, the second column holds a formula P_{R_φ} representing an SPO relation over $\mathcal{D}(X)$. Therefore hierarchical CPTs can be defined for infinite domain attributes X:

EXAMPLE 5. *Take Example 4. A hierarchical CP-network Γ which corresponds to the preference from this example is shown in Figure 3. Note that according to Example 4 the preferences over the attributes y and m are unconditional, i.e. there is no attribute in $\mathcal{X}_\Gamma$ whose value influences the preference over y or m. Therefore $U_y = U_m = \emptyset$ and the first columns of $CPT(y)$ and $CPT(m)$ are skipped (shown filled with dashes).*

$CPT(m)$

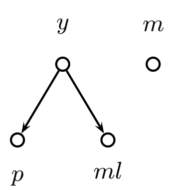
-	m
-	$[o.m = vw \wedge o'.m = kia] \vee$ $[o.m \in \{kia, vw\} \wedge o'.m \notin \{kia, vw\}]$

$CPT(y)$

-	y
-	$o.y > o'.y$

$CPT(ml)$

y	ml
$y \geq 2004$	$o.ml < 60000 \wedge o'.ml \geq 60000$
$y < 2004$	$o.ml < 80000 \wedge o'.ml \geq 80000$



$CPT(p)$

y	p
$y \geq 2006$	$o.p < o'.p$
$y < 2006$	$o.p \leq 8000 \wedge o'.p > 8000$

Figure 3: Hierarchical CP-net for Example 5.

Note that hierarchical CP-networks do not subsume CP-nets, although it may appear to be so. In CP-nets, $W_X = \emptyset$ but this may violate the first condition of Definition 3.

2.2 Subnet of a hierarchical CP-network

Loosely speaking, a subnet of a hierarchical CP-network Γ is just a subset of the conditional preference tables from Γ. However not any subset of Γ is a subnet. Its formal definition is given further.

DEFINITION 4. *Let Δ and Γ be two hierarchical CP-networks such that*

1. *$\mathcal{X}_\Delta \subset \mathcal{X}_\Gamma$: all attributes used in the network Δ are also used in the network Γ;*

2. *if some attribute X from $\mathcal{X}_\Gamma$ is in $\mathcal{X}_\Delta$ then all ancestors of X from H_Γ are in $\mathcal{X}_\Delta$;*

3. *Given two conditional preference tables $CPT_\Delta(X)$ and $CPT_\Gamma(X)$ for an attribute $X \in \Delta \cap \Gamma$ correspondingly, the first and the third components (namely, Φ_X and U_X) of the two conditional preference tables are equal. The W_X component of $CPT_\Delta(X)$ is set to $Desc_\Delta(X) \cup (\mathcal{X} - \mathcal{X}_\Delta)$, i.e. it is set according to the preference graph H_Δ of Δ.*

Then Δ is called a subnet *of a hierarchical CP-network Γ.*

The notion of *subnet* of a hierarchical CP-network will be used further in the paper to construct a preference formula representing the order induced by a hierarchical preference network.

3. FROM HIERARCHICAL CP-NETS TO PREFERENCE FORMULAS

Dealing with preferences, the two most common tasks are 1) given two outcomes, find the more preferred one, and 2) find the optimal outcomes from the given set of outcomes.

The first problem is called *dominance testing*. In the CP-network approach, this problem can be solved in polynomial time in the number of attributes when a preference graph is a directed tree or polytree. If the graph is directed-path singly-connected, dominance testing is NP-complete, and it is in NP if the number of paths between any pair of nodes in the graph is polynomially bounded [1]. In the hierarchical CP-network framework, this problem can be solved in polynomial time in the size of the hierarchical CP-network description by the following proposition which is analogous to a result in [12].

PROPOSITION 3. *Let Γ be a hierarchical CP-net. Let o, o' be two outcomes. Let $Diff = \{X_1, ..., X_l\} \subseteq \mathcal{X}_\Gamma$ be the attributes in whose values o and o' are different, and Top be the set of all nodes from $Diff$ which have no ancestors in $Diff$. Then*

$$o \succ_\Gamma o' \Leftrightarrow \forall X \in Top : Q_{CPT^*(X)}(o, o')$$

As shown in [12], the problem of finding the optimal outcomes can be also solved by Proposition 3 by taking every outcome and checking if there is any outcome dominating it. Thus the optimal outcomes can be computed in time polynomial in the size of the network description and the number of outcomes.

To find the optimal outcomes in the *relational database framework*, the *winnow operator* [4] can be used. It is an algebraic operator which picks from a given relation (containing all possible outcomes) the set of the *most preferred outcomes*, according to a given preference relation. Formally, it is defined as follows.

DEFINITION 5. *If R is a relation schema and $\succ$ a preference relation over R, then the* winnow operator *is written as $\omega_\succ(R)$, and for every instance r of R:*

$$\omega_\succ(r) = \{t \in r \mid \neg\exists t' \in r.\ t' \succ t\}$$

From Definition 5, it follows that using the winnow operator requires the order induced by a hierarchical CP-network to be represented as a binary preference relation.

According to Proposition 3, one of the ways to construct a preference formula representing the order induced by a hierarchical CP-network Γ is by 1) taking every nonempty subset $Diff_i$ of $\mathcal{X}_\Gamma$, 2) for every $Diff_i$, finding the corresponding Top_i, 3) for every $Diff_i$, writing down the formula

$$\begin{aligned} D_i(o, o') = {} & (\textstyle\bigwedge_{X \in \mathcal{X} - Diff_i} o.X = o'.X) \wedge \\ & (\textstyle\bigwedge_{X \in Diff_i - Top_i} o.X \neq o'.X) \\ & (\textstyle\bigwedge_{X \in Top_i} Q_{CPT^*(X)}(o, o')), \end{aligned}$$

and 4) finding the disjunction of all $D_i(o, o')$, which will be the order induced by Γ.

However, the formulas constructed by this algorithm will be clearly exponential in the size of the hierarchical CP-network description, since the algorithm requires enumerating all nonempty subsets of $\mathcal{X}_\Gamma$.

At the end of this section, we present another algorithm, Algorithm 1, for constructing a preference formula representing the order induced by a hierarchical CP-network. Algorithm 1 produces polynomial-size preference formulas for a large class of hierarchical CP-networks.

3.1 Disassembling a hierarchical CP-network into a set of connectives and subnets

To construct a preference formula for a hierarchical CP-net Γ, we consider its preference graph H_Γ as a set of *many-to-one* connectives and *parallel subnets.* The preference formula construction algorithm provided at the end of this section is iterative. It starts from the set of parallel subnets each of which consists of a topmost node of H_Γ. Each step of the algorithm joins two or more parallel subnets of Γ and/or extends the existing subnets with a node from $\mathcal{X}_\Gamma$.

3.1.1 Parallel subnets

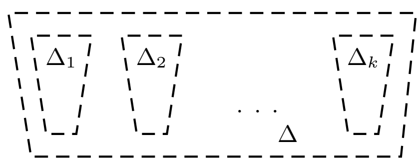


Figure 4: Parallel subnets

Proposition 4. *Let $\Delta_1, ..., \Delta_k$ be some subnets of Γ and $\succ_{\Delta_1}, ..., \succ_{\Delta_k}$ be SPO relations representing the orders induced by $\Delta_1, ..., \Delta_k$ correspondingly. Let $P_{\succ_{\Delta 1}}, ..., P_{\succ_{\Delta k}}$ be formula representations of $\succ_{\Delta_1}, ..., \succ_{\Delta_k}$ correspondingly. Let also Δ be a subnet of Γ such that $\mathcal{X}_\Delta = \mathcal{X}_{\Delta_1} \cup ... \cup \mathcal{X}_{\Delta_k}$. Then the formula $P'_{\succ_\Delta}$ defined as*

$$\begin{aligned} P'_{\succ_\Delta}(o, o') \equiv\ & (P_{\succ_{\Delta 1}}(o, o') \vee \textstyle\bigwedge_{Z \in \mathcal{X}_{\Delta_1}} o.Z = o'.Z) \wedge ... \wedge \\ & (P_{\succ_{\Delta k}}(o, o') \vee \textstyle\bigwedge_{Z \in \mathcal{X}_{\Delta_k}} o.Z = o'.Z) \wedge \\ & \neg \textstyle\bigwedge_{Z \in \mathcal{X}_\Delta} o.Z = o'.Z \end{aligned}$$

defines an SPO, and $P'_{\succ_\Delta} \equiv P_{\succ_\Delta}$.

By Proposition 4, if we know preference formulas representing the orders induced by a set of parallel subnets, we can easily find a preference formula representing the order induced by the union of these subnets.

Note that according to Proposition 4, the subnets $\Delta_1, ..., \Delta_k$ are not required to be disjoint (i.e. if some subnet Δ_i contains an attribute, then the other subnets can also contain the same attribute).

Example 6. *Take the hierarchical CP-network from Example 5. It can be considered as a union of two subnets Δ_1 consisting of the attributes $\{p, y, ml\}$, and Δ_2 consisting of a single attribute $\{m\}$.*

Then according to Proposition 4, the preference formula representing the order induced by Γ is

$$\begin{aligned} P_{\succ_\Gamma}(o, o') \equiv\ & (P_{\succ_{\Delta 1}}(o, o') \bigvee \\ & o.p = o'.p \wedge o.y = o'.y \wedge o.ml = o'.ml) \bigwedge \\ & (P_{\succ_{\Delta 2}}(o, o') \bigvee o.m = o'.m) \bigwedge \\ & \neg[o.p = o'.p \wedge o.y = o'.y \wedge \\ & o.ml = o'.ml \wedge o.m = o'.m] \end{aligned}$$

where $P_{\succ_{\Delta 1}}$ is a preference formula representing the order induced by Δ_1, and $P_{\succ_{\Delta 2}}$ is a preference formula representing the order induced by Δ_2 which is defined as

$$\begin{aligned} P_{\succ_{\Delta 2}} \equiv\ & o.m = vw \wedge o'.m = kia \bigvee \\ & o.m \in \{kia, vw\} \wedge o'.m \notin \{kia, vw\} \end{aligned}$$

3.1.2 Many-to-one connectives

Let $\Delta_1, ..., \Delta_k$ be some subnets of Γ. Let also X_e be such an attribute from $\mathcal{X}_\Gamma - (\mathcal{X}_{\Delta_1} \cup ... \cup \mathcal{X}_{\Delta_k})$ that

- each parent of X_e in H_Γ is in one of $\mathcal{X}_{\Delta 1}, ..., \mathcal{X}_{\Delta k}$;
- if an attribute $Y \in \mathcal{X}_{\Delta i}$ is a parent of X_e in H_Γ, then all the other nodes in $\mathcal{X}_{\Delta_i}$ are the ancestors of Y in $H_{\Delta i}$ (i.e. Y is the bottom most node of H_{Δ_i}).

Then the set of $\Delta_1, ..., \Delta_k$ along with X_e is called *many-to-one connective.*

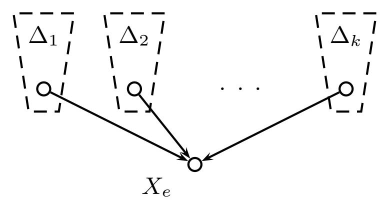


Figure 5: Many-to-one connective

Proposition 5. *Let a set of subnets $\Delta_1, ..., \Delta_k$ of Γ and an attribute $X_e \in \mathcal{X}_\Gamma$ be a many-to-one connective and $\succ_{\Delta_1}, ..., \succ_{\Delta_k}$ be SPO relations representing the orders induced by $\Delta_1, ..., \Delta_k$ correspondingly. Let $P_{\succ_{\Delta 1}}, ..., P_{\succ_{\Delta k}}$ be formula representations of $\succ_{\Delta_1}, ..., \succ_{\Delta_k}$ correspondingly. Let also Δ be such a subnet of Γ that $\mathcal{X}_\Delta = \mathcal{X}_{\Delta_1} \cup ... \cup \mathcal{X}_{\Delta_k} \cup \{X_e\}$. Then the formula $P'_{\succ_\Delta}$ defined as*

$$\begin{aligned} P'_{\succ_\Delta}(o, o') \equiv\ & (P_{\succ_{\Delta 1}}(o, o') \vee [\textstyle\bigwedge_{Z \in \mathcal{X}_{\Delta 1}} o.Z = o'.Z]) \wedge ... \wedge \\ & (P_{\succ_{\Delta k}}(o, o') \vee [\textstyle\bigwedge_{Z \in \mathcal{X}_{\Delta k}} o.Z = o'.Z]) \wedge \\ & \neg[\textstyle\bigwedge_{\mathcal{X}_{\Delta 1} \cup ... \cup \mathcal{X}_{\Delta k}} o.Z = o'.Z] \bigvee \\ & [\textstyle\bigwedge_{Z \in \mathcal{X}_{\Delta 1} \cup ... \cup \mathcal{X}_{\Delta k}} o.Z = o'.Z] \wedge Q_{CPT^*(X_e)}(o, o') \end{aligned}$$

defines an SPO, and $P'_{\succ_\Delta} \equiv P_{\succ_\Delta}$.

By Proposition 5, if we know preference formulas representing the orders induced by all components of a many-to-one connective, we can easily find a preference formula representing the order induced by the entire connective.

Note that if the preference graph of a hierarchical CP-network Γ can be represented as a set of DAGs in which each node has at most one outgoing edge, Propositions 4 and 5 are enough to iteratively construct a preference formula for Γ. Namely, for each DAG in the set, perform the following steps 1) start with the subnets each of which consists of a single topmost attribute of the DAG, 2) iteratively merge and extend the existing subnets by one child node in each iteration (i.e. apply Proposition 5), 3) finally, apply Proposition 4 to the parallel subnets with no descendant nodes. However this algorithm is not always applicable because a hierarchical CP-network can have attributes with more than one outgoing edge (e.g. Γ from Example 5).

3.1.3 One-to-many connectives

In this section, we present a method which allows to transform a hierarchical network whose preference graph contains nodes with more then one outgoing edge (*one-to-many connective*) to a set of parallel subnets whose preference graphs have no nodes with more than one outgoing edge.

Let Δ_1 be a subnet of a hierarchical CP-network of Γ and X_s be such attribute in H_{Δ_1} that all the other attributes in Δ_1 are its ancestors (i.e. X_s is the bottom most node in H_{Δ_1}). Let X_s have k (where $k \geq 1$) outgoing edges in H_Γ. Formally, let there exist such attributes $X_j, ..., X_{j+k} \in \mathcal{X}_\Gamma$ that X_s is their parent (possibly one of many parents).

In order to avoid the situation when a node has more than one outgoing edge, we will *make* $k-1$ *copies* of the subnet Δ_1. As a result, we will make subnets $\Delta_2, ..., \Delta_k$ each of which is a copy of Δ_1. Note that the order induced by the subnet which is a union of $\Delta_1, ..., \Delta_k$ will be the same as the order induced by only Δ_1 (by Proposition 4). After that, we will make each attribute X_s of each of $\Delta_1, ..., \Delta_k$ a parent of *only one* attribute $X_j, ..., X_{j+k}$ correspondingly. As a result, each copy of X_s now has only one outgoing edge. Note also that this operation does not violate the semantics of the hierarchical CP-network because the preference over each of $X_j, ..., X_{j+k}$ is still conditionally dependent on X_s. We call this process *one-to-many connective elimination*.

EXAMPLE 7. *Take the subnet Δ_1 from Example 6. It consists of the attributes $\{p, y, ml\}$ such that y has two outgoing edges to p and ml. The one-to-many connective elimination technique splits this subnet into two parallel subnets Δ_3 and Δ_4 as it is shown in Figure 6.*

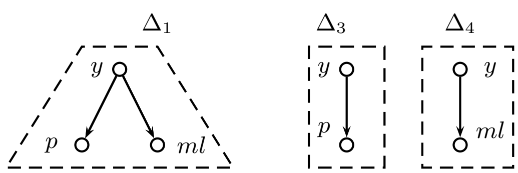


Figure 6: Eliminating the one-to-many connective from Δ_1.

The hierarchical CP-networks Δ_3 and Δ_4 are defined as in Figure 7.

Given a hierarchical CP-network with many-to-one connectives, this technique produces a set of subnets without one-to-many connectives. Each of the subnets is a hierarchical CP-network and thus one can apply Proposition 5 to construct a preference formula for each of them. Finally, Proposition 4 can be used to merge the subnets back and produce a preference formula for it.

3.2 Constructing preference formulas

Below we provide an algorithm for constructing an SPO preference formula representing the order induced by a hierarchical CP-network. It takes two parameters: a hierarchical CP-network Γ and the graph H'_Γ which is the result of performing the *one-to-many connective elimination* on H_Γ.

ALGORITHM 1.

`FormulaCons`(Γ, H'_Γ)

1. `S = topologically sorted sequence of nodes` H'_Γ;
2. `For each node` X `in` S,
3. `if` X `has no parents in` H'_Γ, $PA[X] = \{X\}$;
4. `else` $PA[X] = PA[Y_1] \cup ... \cup PA[Y_k] \cup \{X\}$, `where`
5. $\quad Y_1, ..., Y_k$ `are the parents of` X `in` H'_Γ;
6. `For each node` X `from` S `do`
7. `If` X `has no parents in` H'_Γ, `then`
8. $\quad P[X] = P^*_{CPT(X)}$
9. `If` X `has parents` $Y_1, ..., Y_k$, `then`
10. $\quad P[X] = (P[Y_1] \vee [\bigwedge_{Z \in PA[Y_1]} o.Z = o'.Z]) \wedge$
11. $\quad\quad ...$
12. $\quad\quad (P[Y_k] \vee [\bigwedge_{Z \in PA[Y_k]} o.Z = o'.Z]) \vee$
13. $\quad\quad ([\bigwedge_{Z \in PA[Y_1] \cup ... \cup PA[Y_k]} o.Z = o'.Z] \wedge$
14. $\quad\quad P^*_{CPT(X)})$
15. `Let` D `be the sequence of all nodes from` H'_Γ
16. `with no outgoing edges`
17. `If` $|D| = 1$, `then`
18. $\quad OUT = P[D[1]]$
19. `else` ($|D| = k > 1$)
20. $\quad OUT = (P[D[1]] \vee [\bigwedge_{Z \in PA[D[1]]} o.Z = o'.Z]) \wedge$
21. $\quad\quad ...$
22. $\quad\quad (P[D[k]] \vee [\bigwedge_{Z \in PA[D[k]]} o.Z = o'.Z]) \wedge$
23. $\quad\quad (\neg[\bigwedge_{Z \in PA[D[1]] \cup ... \cup PA[D[k]]} o.Z = o'.Z])$
24. `return` OUT;

$\Delta_3 = \{CPT_{\Delta_3}(y), CPT_{\Delta_3}(p)\}$

$CPT_{\Delta_3}(y)$:

-	y
-	$o.y > o'.y$

$CPT_{\Delta_3}(p)$:

y	p
$y \geq 2006$	$o.p < o'.p$
$y < 2006$	$o.p \leq 8000 \wedge o'.p > 8000$

$\Delta_4 = \{CPT_{\Delta_4}(y), CPT_{\Delta_4}(ml)\}$

$CPT_{\Delta_4}(y)$:

-	y
-	$o.y > o'.y$

$CPT_{\Delta_4}(ml)$:

y	ml
$y \geq 2004$	$o.ml < 60000 \wedge o'.ml \geq 60000$
$y < 2004$	$o.ml < 80000 \wedge o'.ml \geq 80000$

Figure 7: The hierarchical CP-networks Δ_3 and Δ_4.

Since a preference graph H_Γ is acyclic, H'_Γ is also acyclic. Therefore the sequence D constructed in line 15 is nonempty.

The algorithm outputs relation $OUT(o, o')$ which is an SPO preference formula representing the order induced by Γ. Below we provide an example of applying the algorithm to the hierarchical CP-network from Example 5.

EXAMPLE 8. *Given a hierarchical CP-network Γ and a preference graph from Example 5, we apply the one-to-many elimination technique and decompose the preference graph into two parallel subnets: Δ_1 and Δ_2 as it is in Example 6. By applying one-to-many elimination technique, Δ_1 was decomposed into Δ_3 and Δ_4 as it is in Example 7. By proposition 5, the formula $P_{\succ_{\Delta_3}}$ representing the order induced by Δ_3 is*

$$P_{\succ_{\Delta_3}}(o, o') \equiv o.y > o'.y \vee$$
$$(o.y = o'.y \wedge (o.y \geq 2006 \wedge o'.y \geq 2006 \wedge o.p < o'.p \vee$$
$$o.y < 2006 \wedge o'.y < 2006 \wedge o.p \leq 8000 \wedge o'.y > 8000)),$$

and the formula $P_{\succ_{\Delta_4}}$ representing the order induced by Δ_4 is

$$P_{\succ_{\Delta 4}}(o,o') \equiv o.y > o'.y \vee (o.y = o'.y \wedge (\\ o.y \geq 2004 \wedge o'.y \geq 2004 \wedge\ o.ml < 60000 \wedge \\ o'.ml \geq 60000 \vee o.y < 2004 \wedge o'.y < 2004 \wedge \\ o.ml < 80000 \wedge o'.ml \geq 80000)).$$

Thus by Proposition 4, the formula $P_{\succ_{\Delta 1}}$ representing the order induced by Δ_1 is

$$P_{\succ_{\Delta 1}}(o,o') \equiv (P_{\succ_{\Delta_3}}(o,o') \vee o.y = o'.y \wedge o.p = o'.p) \wedge \\ (P_{\succ_{\Delta_4}}(o,o') \vee o.y = o'.y \wedge o.ml = o'.ml) \wedge \\ \neg(o.y = o'.y \wedge o.p = o'.p \wedge o.ml = o'.ml).$$

And finally, a preference formula representing the order induced by Γ can be computed by the expression provided in Example 6, i.e.

$$P_{\succ_\Gamma}(o,o') \equiv (P_{\succ_{\Delta 1}}(o,o') \bigvee \\ o.p = o'.p \wedge o.y = o'.y \wedge o.ml = o'.ml) \bigwedge \\ (P_{\succ_{\Delta 2}}(o,o') \bigvee o.m = o'.m) \bigwedge \\ \neg[o.p = o'.p \wedge o.y = o'.y \wedge \\ o.ml = o'.ml \wedge o.m = o'.m]$$

4. COMPLEXITY ANALYSIS

The purpose of this section is to provide an analysis of the size of the preference formula produced by Algorithm 1. We divide this analysis into two parts:

- the analysis of the size of the constructed preference formula as a function of the number of nodes in its preference graph H_Γ assuming that H_Γ has no one-to-many connectives;
- the analysis of size of a preference graph H'_Γ produced by the one-to-many connective elimination technique applied to the original preference graph H_Γ;

According to Proposition 6 below, the size of the preference formula is polynomial in the number of conditional preference tables in a hierarchical CP-network, given its preference graph has no one-to-many connectives.

Proposition 6. *The size of the preference formula produced by Algorithm 1 for a hierarchical CP-network Γ such that H_Γ does not have one-to-many connectives is $\Theta(|\mathcal{X}_\Gamma|^2 + |\mathcal{X}_\Gamma| B_{max})$, where B_{max} is the size of the longest formula $Q_{CPT^*(X)}$ among all $X \in \mathcal{X}_\Gamma$.*

However, given a preference graph H_Γ, its version H'_Γ without one-to-many connectives may have exponential size as shown below. Thus in general the size of the preference formula constructed by Algorithm 1 may be exponential in the number of conditional preference tables in a hierarchical CP-network.

Proposition 7. *Given a hierarchical CP-network Γ, the size of the preference graph H'_Γ produced from H_Γ by the one-to-many elimination technique is $\Theta(2^{|\mathcal{X}_\Gamma|})$.*

The exponential number of nodes in the preference graph after eliminating one-to-many connectives is caused by the fact that the number of copies of each node in the modified graph (the graph without one-to-many connectives) is

$$N'(X) = N_{out}(X) - 1 + \sum_{Y \in Ch_\Gamma(X)} N'(Y),$$

where $N_{out}(X)$ is the number of outgoing edges from node X in H_Γ. Therefore, in the worst case, eliminating one-to-many connectives for each node of H_Γ multiplies by c (where $c \geq 2$) the number of nodes in H'_Γ.

Example 9. *Consider a hierarchical CP-network Γ_n whose preference graph has n nodes (where n is an even number) connected as follows: the nodes X_1 and X_2 have no outgoing edges; and for any $i \in [3,n]$, 1) X_i is a parent of the nodes X_{i-3} and X_{i-2}, if i is even, and 2) X_i is a parent of the nodes X_{i-2} and X_{i-1}, if i is odd. An example of such network is shown below.*

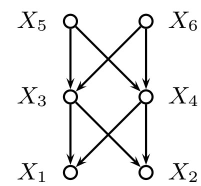


Figure 8: Hierarchical CP-net for Example 9.

Since the nodes X_1 and X_2 have no outgoing edges, no new copies of them will be produced by the one-to-many elimination. However, for each $i \in [3,n]$, the number of copies of X_i will be $N'(X_i) = 1 + 2N'(X_{i-2})$. Thus the total number of nodes in the network obtained from Γ_n by eliminating one-to-many connectives is clearly exponential in n.

However, by restricting the structure of a preference graph, the exponential blowup can be avoided.

Proposition 8. *Let Γ be such a hierarchical CP-network that every node in H_Γ has at most one child node with outgoing edges. Then the size of the preference graph H'_Γ produced from H_Γ by the one-to-many elimination technique is $O(|\mathcal{X}_\Gamma|^3)$.*

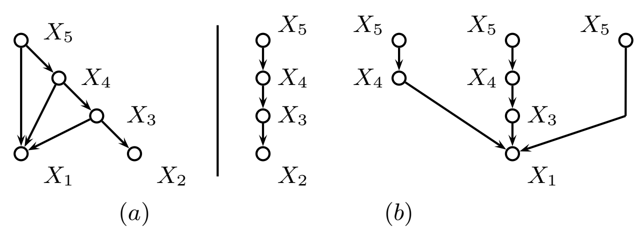


Figure 9: (a) A preference graph of a hierarchical CP-network satisfying Proposition 8, and (b) the network resulting from one-to-many connective elimination

As it follows from the analysis, given any hierarchical CP-network, performing dominance testing via preference formula constructed by our algorithm will require at most exponential time. At the same time, as it was discussed above, using Proposition 3 one can do it in polynomial time. The same applies to the problem of finding the optimal outcomes. However, having constructed a preference formula, one can use hierarchical CP-networks in the relational framework, where using the *winnow* operator creates opportunities for efficient evaluation and algebraic query optimization [3, 4].

5. RELATED AND FUTURE WORK

In this paper, we propose a variant of CP-networks[1] that addresses some of the limitations of the original proposal. Namely, *preference priority* can be expressed i.e. an edge in a preference graph captures not only conditional dependence between different attributes but also the relative preference importance - the higher an attribute is in the preference graph, the higher priority it has. Moreover, our framework can be used with *infinite* domain attributes. We also believe that the *everything else being equal* semantics is too strict and sometimes might not be applicable. By introducing a new preference with the *ceteris paribus* semantics, the user is required to consider all possible values of all other attributes, which is not always feasible. Thus in such situations, using our framework will be preferred to using CP-networks.

TCP-networks [2] is an extension of CP-networks which adds *attribute importance* to them. In TCP-networks, two special kinds of arcs are added to preference graphs: 1) an *i*-arc from one attribute to another implies that the first attribute is more important than the second; 2) a *ci*-arc between two attributes implies that relative importance of the connected attributes is dependent on the values of some other attributes. In contrast to that, relative importance of attributes in hierarchical CP-networks is dictated by the preference graph structure, thus *i*-arcs would be redundant here. However, to represent *conditional* attribute importance, the hierarchical CP-network model needs to be extended.

The idea of extending CP-networks with irrelevant attributes was introduced in [12]. The class of networks considered in [12] is more general than hierarchical CP-networks. In this model, W_X is not limited to the set of siblings and descendants of X and can be any subset of $\mathcal{X} - \{X\} - Pa(X)$. However, [12] does not consider how to deal with infinite domain attributes, and it does not have the notion of conditional preference table, which is essential for CP-networks. [12] proves that the orders induced by some classes of the extended CP-networks are strict partial orders. However, it does not provide an algorithm of constructing preference formulas for them.

The observation that the semantics of CP-nets can be captured using Constraint Datalog programs was first made in [6]. It was further developed into a method of bridging the TCP-network framework and the framework of preference relations in [7]. In [7] a binary preference formula is constructed for a given TCP-network using a set of preference constructors. The preference order is essentially computed as transitive closure of the union of all relations representing the orders induced by each conditional preference table. The transitive closure is computed using Constraint Datalog. When Constraint Datalog is used to compute transitive closure, one needs to show that the evaluation terminates (which is the case only for some constraint theories [8]). Moreover, one typically obtains at best exponential bounds on the resulting formula size. In contrast to that, we describe classes of hierarchical CP-networks for which our algorithm can construct polynomial-size preference formula representations. [7] also gives an example of embedding preferences over infinite domain attributes into TCP-networks, namely it shows how can one represent CPT(X) of an infinite domain attribute X using a limited set of *preference constructors* for which a transitive closure Datalog program terminates. In our work we have no limitations on the class of SPO preferences formulas used in CPT(X) representation.

In this paper, we define hierarchical CP-networks that are based on a relaxed *ceteris paribus* principle (i.e. *some other attributes being equal*). An interesting direction of the future work is to extend our semantics using the *some other attributes being equivalent* principle where instead of equality attribute values are considered to be *equivalent* according to some indifference relation [11]. One should also consider applying the existing preference modification and construction techniques [5, 10] to hierarchical CP-networks.

6. ACKNOWLEDGMENT

Research supported by NSF grant IIS-0307434.

7. REFERENCES

[1] 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. *Journal of Artificial Intelligence Research (JAIR)*, 21:135–191, 2004.

[2] R. I. Brafman, C. Domshlak, and E. Shimony. Introducing variable importance tradeoffs into CP-nets. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI02)*, pages 69–76, 2002.

[3] J. Chomicki. Semantic optimization techniques for preference queries. *Information Systems, 32(5)*, July 2007, pages 670–684.

[4] J. Chomicki. Preference formulas in relational queries. *ACM Transactions on Database Systems*, pages 427–466, 2003.

[5] J. Chomicki. Iterative modification and incremental evaluation of preference queries. *Proc. 4th International Symposium on Foundations of Information and Knowledge Systems (FoIKS), LNCS 3861*, pages 63–82, 2006, full version to appear in Annals of Mathematics and Artificial Intelligence.

[6] J. Chomicki and J. Song. Monotonic and nonmonotonic preference revision. *arXiv:cs.DB/0503092*, 31 Mar 2005.

[7] M. Endres and W. Kießling. Transformation of TCP-net queries into preference database queries. *Proceedings of the ECAI 2006 Multidisciplinary Workshop on Advances in Preference Handling*, 2006.

[8] Paris C. Kanellakis, Gabriel M. Kuper, and Peter Z. Revesz. Constraint query languages. *Journal of Computer and System Sciences*, pages 26–52, 1995.

[9] W. Kießling. Foundations of preferences in database systems. *International Conference on Very Large Data Bases (VLDB)*, pages 311–322, 2002.

[10] W. Kießling. Preference constructors for deeply personalized database queries. *Technical Report, Institute of Computer Science, University of Augsburg*, 2004.

[11] W. Kießling. Preference queries with SV-semantics. *11th International Conference on Management of Data (COMAD 2005)*, pages 15–26, 2005.

[12] N. Wilson. Extending CP-nets with stronger conditional preference statements. *Proc. AAAI 2004*, pages 735–741, 2004.