Querying Databases with Incomplete CP-nets

Paolo Ciaccia
DEIS, University of Bologna, Italy
pciaccia@deis.unibo.it

ABSTRACT

Preference queries aim to retrieve from large databases (DB's) those objects that better match user's requirements. With the aim of supporting modern DB applications, such as context-aware ones, in which conditional preferences are the rule, in this paper we investigate the possibility of adopting *conditional preference* networks (CP-nets) for DB querying. To this end, we also consider the relevant case in which CP-nets are not completely specified, a likely case for complex DB scenarios. We first show that the *ceteris paribus* (all else being equal) semantics, commonly associated with CP-nets, can lead to counterintuitive results if the CP-net is incomplete and the DB is incomplete as well. Then, we introduce a new *totalitarian* (i.e., not ceteris paribus) semantics and, rather surprisingly, prove that our semantics is equivalent to ceteris paribus for complete acyclic CP-nets and that yields the same optimal results if the DB is complete. Finally, we show that when both the CP-net and the underlying DB are incomplete the totalitarian semantics can lead to more accurate and intuitive results.

1. INTRODUCTION

The trend towards the personalization of information systems functionalities requires new models and techniques able to provide users with the "right information" at the "right time" in the "right place". Context-aware applications are a remarkable step towards achieving this goal, the key idea being that of taking into account context information when processing user requests. In particular, ranking the result of a query should be based on the current user context, rather than on some absolute criterion.

Example 1 *Consider the following database of hotels:*

Name	*Price*	*Stars*	*Rooms*	*Internet*
Jolly	*40*	*2*	*50*	*Yes*
Continental	*55*	*2*	*30*	*No*
Excelsior	*80*	*3*	*50*	*Yes*
Rome	*80*	*5*	*100*	*Yes*
Holiday	*60*	*4*	*20*	*No*

When travelling for work, the user does not care about price and number of rooms, she preferring hotels with at least 4 stars and an Internet connection. In this case the (only) best alternative is hotel Rome (5 stars and network-connected). However, if travelling for leisure, the user prefers small hotels (≤ 30 rooms) and whose price is at most 50 Euro. In this case no hotel satisfies both requirements, yet it can be argued that Continental, Jolly, and Holiday are the best available alternatives, since each of them satisfies one of the two user preferences. □

Frameworks proposed so far in the DB field [5, 10] have paid no specific attention to *conditional* preferences (see Section 5 for more details). On the other hand, these have been largely investigated by AI researchers, with a particular emphasis on *CP-nets* (Conditional Preference networks), see [2, 1, 14, 9], a graph-based formalism able to "factorize" the specification of preference statements over a set of attributes. A CP-net statement like $\varphi = p : a_i > a_j$, where a_i and a_j are values of attribute A and p is the value of other attributes P, is given a *ceteris paribus* interpretation, i.e., "given p prefer a_i to a_j only if values of other attributes are *equal*".

In order to use CP-nets for the purpose of DB querying, two major issues need to be addressed. First, since CP-nets are defined only for *finite* attribute domains, an extension to infinite domains, which are common in DB applications, is needed [6]. Second, the case in which the CP-net is not completely specified has to be considered. This is to cover the likely case in which there are many attributes, possibly with large domains, and the user only states a limited set of preferences.

In this paper we concentrate on this second issue and show that the ceteris paribus semantics yields counterintuitive results when the CP-net is incomplete and the DB is incomplete as well, i.e., it does not contain all the possible alternatives for the preference attributes. With the aim of preserving the strong points of CP-nets, in particular their capability of concisely representing conditional preferences, we study an alternative, so called *totalitarian*, semantics for CP-nets.[1] Our major formal result shows that, rather surprisingly, the new semantics is *equivalent* to ceteris paribus for complete acyclic CP-nets. Then we prove that for complete DB's the two semantics, although leading to different preferences, always yield the same set of optimal results. Finally, we consider the case of incomplete DB's *and* CP-nets and argue that the totalitarian semantics is better suited to exclude from the result those tuples that are apparently sub-optimal with respect to user preferences. Finally, we discuss possible extensions of the totalitarian semantics.

[1] In the literature the "CP" acronym is sometimes used to stand for "ceteris paribus" rather than for "conditional preference". In this paper we adhere to the original interpretation [2], thus we find no contradiction in defining a totalitarian semantics for CP-nets.

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

2. BACKGROUND ON CP-NETS

In this section we provide the necessary formal background on CP-nets. We adopt standard database terminology, the correspondence with the one commonly used for describing CP-nets being shown in Table 1.

DB terminology	CP-nets terminology	Symbol
attribute	variable	$A, B, C, \ldots, A_i$
schema	set of variables	X
domain of A	domain of A	$dom(A)$
tuple (over X)	outcome	$t \equiv t[X]$
relation	feasible outcomes	$r \subseteq dom(X)$

Table 1: Basic terminology

A CP-net over a set of attributes $X = \{A_1, \ldots, A_n\}$ is a pair $N = (G, CPT)$, where $G = (X, E)$ is a directed graph and CPT is a function that associates to each $A_i \in X$ a *conditional preference table*, $CPT(A_i)$. If the arc $(A_j, A_i) \in E$, then A_j is a *parent* of A_i. Let P be the set of parents of attribute A. Then, $CPT(A)$ consists of a set of preference statements φ of the form $\varphi = p : a_1 > a_2$,[2] where $p \in dom(P)$ and $a_1, a_2 \in dom(A)$.[3] This expresses the conditional preference of a_1 with respect to a_2 given p. If A has no parents, then the statement simplifies to $\varphi = \perp: a_1 > a_2 \equiv a_1 > a_2$, i.e., a_1 is unconditionally preferred to a_2.

Example 2 *Figure 1 shows a simple CP-net over attributes RestaurantType (R), Table (T), and Price (P), thus* $X = \{R, T, P\}$. *For simplicity, all attributes have binary domains, in particular:* $dom(R) = \{it, chn\}$ *(Italian or Chinese),* $dom(T) = \{in, out\}$ *(inside or outside), and* $dom(P) = \{low, high\}$. *My preferences unconditionally go to Italian restaurants (*$it > chn$*), for which I prefer to have a table inside (*$it : in > out$*) and pay the less (*$it : low > high$*). On other hand, in a Chinese restaurant I prefer to sit outside (*$chn : out > in$*) and to pay more (*$chn : high > low$*).* □

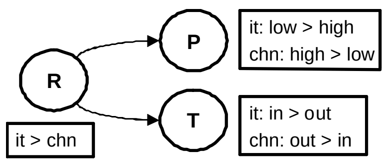


Figure 1: The CP-net of Example 2

Definition 1 *A CP-net* $N = (G, CPT)$ *is:*

- acyclic *iff* G *is acyclic;*
- locally consistent *iff, for each attribute* A *with parents* P *and for each* $p \in dom(P)$*, the order* $>$ *induced by* p *on the values of* $dom(A)$ *is transitive and asymmetric, i.e., (at least) a strict partial order.*
- complete *iff, for each* A *with parents* P *and for each* $p \in dom(P)$*,* $>$ *is a (strict) total order, i.e., a strict partial order such that for each* $a_1, a_2 \in dom(A)$*,* $a_1 \neq a_2$*, either* $p : a_1 > a_2$ *or* $p : a_2 > a_1$.

The CP-net in Figure 1 is acyclic and complete (thus, also locally consistent). Should we drop one statement (e.g., $it : in > out$), the CP-net would still be locally consistent, but incomplete. If the CP-net is locally consistent, no contradiction is present as long as we consider preferences over any single attribute. In the following we only consider acyclic and locally consistent CP-nets.

2.1 The Ceteris Paribus Semantics

The standard *ceteris paribus* interpretation of a statement $\varphi = p : a_1 > a_2$, $\varphi \in CPT(A)$, is the set of pairs of tuples over X:

$$\varphi^*_{\mathsf{cp}} = \{((p, a_1, y), (p, a_2, y)) | y \in dom(X - P - \{A\})\} \quad (1)$$

in which y is any value of $dom(Y)$, $Y = X - P - \{A\}$ being the set of attributes not involved in φ. Thus, each preference induced by φ concerns two tuples that differ *only* in the value of A.

Let $\Phi^*_{A_i,\mathsf{cp}} = \bigcup_{\varphi \in CPT(A_i)} \varphi^*_{\mathsf{cp}}$ denote all preferences induced by $CPT(A_i)$. Since the CP-net is locally consistent, no conflicts are present in $\Phi^*_{A_i,\mathsf{cp}}$. Further, it is easy to see that, due to the ceteris paribus (cp) semantics, any two tuples t_1 and t_2 are ordered by at most one $\Phi^*_{A_i,\mathsf{cp}}$ set. Taking the union of such sets leads to:

$$\Phi^*_{\mathsf{cpu}} = \bigcup_{A_i \in X} \Phi^*_{A_i,\mathsf{cp}} \quad (2)$$

Finally, let $\succ_{\mathsf{cpu}}$ stand for the order obtained by taking the transitive closure of Φ^*_{cpu}.[4] We say that tuple t_1 *dominates* tuple t_2 (according to the *ceteris paribus union* (cpu) semantics) iff $t_1 \succ_{\mathsf{cpu}} t_2$, and that t_1 is *optimal* in a relation $r \subseteq dom(X)$ if it is undominated in r.

A basic result on acyclic CP-nets is that $\succ_{\mathsf{cpu}}$ is always a strict partial order, which guarantees that at least one optimal tuple exists. Further, if the CP-net is complete there is exactly one optimal tuple in $dom(X)$.

Example 3 *Figure 2 shows the* preference graph *for the CP-net in Figure 1, where there is an arc from* t_1 *to* t_2 *iff the pair* (t_1, t_2) *is in* Φ^*_{cpu}*. Due to the ceteris paribus semantics, arcs exist only between tuples that differ in the value of a single attribute [14]. There is a* path *in the graph from* t_1 *to* t_2 *iff* $t_1 \succ_{\mathsf{cpu}} t_2$ *(e.g.,* $(it, out, low) \succ_{\mathsf{cpu}} (chn, in, low)$ *through the path* (it, out, low)*,* (chn, out, low)*,* (chn, in, low)*). Since the CP-net is complete there is one optimal tuple in* $dom(X)$*, namely* (it, in, low)*.* □

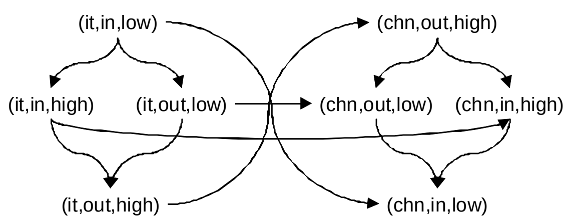


Figure 2: The $\succ_{\mathsf{cpu}}$ order induced over tuples by the CP-net in Figure 1

[2] In practice, each statement might specify a conjunction of pair orderings of the form $a_j > a_k$, given a set of values from $dom(P)$.

[3] Note that we simplify the notation and write $>$ in place of $>^A_p$, since p and A will always be clear from the context.

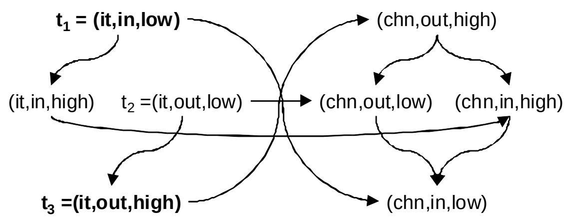
[4] Our graph-based formalization of the cpu semantics is quite different from that in [1], which is based on the notion of *sequence of worsening flips*, yet it is equivalent and more suitable to the purpose of this paper.

Concerning the proof procedure of CP-nets, needed to check if $t_1 \succ_{\text{cpu}} t_2$, it is known that for acyclic CP-nets its complexity can be exponential in the number of attributes, depending on the structure of the G graph and on how CPT's are specified [1].

2.2 Incomplete CP-nets

Although the cpu semantics is adequate in many situations, it is a fact that in most cases a *complete CP-net* is assumed. When preferences are over many attributes and/or domains have large cardinalities, it is unrealistic to expect that a user will completely specify all the CPT's.

The effects of having an incomplete CP-net can be seen by referring to Example 2. Assume the user has specified no preference on where to sit in Italian restaurants. Then, $it : in > out$ is dropped from Figure 1 and the following preference graph results:



If the DB relation is *complete*, i.e., $r = dom(X)$, then the optimal tuples are $t_1 = (it, in, low)$ and $t_2 = (it, out, low)$, which is perfectly reasonable given the absence of preference on where to sit. Assume now that $r = \{t_1 = (it, in, low), t_3 = (it, out, high)\}$. Since $t_1 \not\succ_{\text{cpu}} t_3$ (there is no path from t_1 to t_3 in the above graph), we conclude that both t_1 and t_3 are optimal in r. We find this quite counterintuitive, since t_3 has a high price, which contradicts the preference $it : low > high$. Ideally, we would like to have that t_1 dominates t_3 even if the CP-net is incomplete.

Similarly, consider the preferences of Example 1 in the Introduction. They can be represented by a CP-net over 5 attributes, namely: Price (P), Stars (S), Rooms (R), Internet (I), and Travel (T), where the domain of Travel includes work (w) and leisure (l) values. Travel, although not an attribute in the database, is a *context* attribute needed to properly model user preferences. Figure 3 shows the CP-net for this example, in which statements are expressed in a compact form. For instance, the preference on hotels with at least 4 stars is written as $\{\geq 4\} > \{< 4\}$. This net is highly incomplete, since no domain is totally ordered for all parent (i.e., Travel) values.

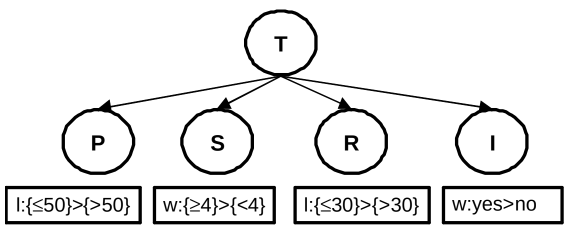


Figure 3: The CP-net for Example 1

Consider the work context. From the relation in Example 1 we see that the (only) best alternative is Rome (5 stars and Internet connection, as required). The ceteris paribus semantics, on the other hand, would return as optimal tuples the set {Jolly, Rome, Holiday}. Hotel Jolly is in the answer since its Price value cannot be compared to that of Rome (there is no preference on Price for the work context), thus the cpu semantics is unable to infer that Rome is definitely better than Jolly. Similar arguments hold for Holiday as well as for the leisure context.

Note that, although in this simple example one could build one different CP-net for each context value, in the general case this would lead to combinatorial explosion. Further, this approach would not exploit at all preferences shared by different contexts (i.e., parents' values).

Before introducing our novel semantics, a brief digression about the interpretation of incomplete CP-nets is needed. Indeed, referring to the first of our examples, it is apparent the ceteris paribus semantics assumes that the absence of preference on where to sit in Italian restaurants has to be interpreted as *incomparability* [1]: "given it, values in and out are incomparable, thus any two tuples (it, in, p_1) and (it, out, p_2), with $p_1, p_2 \in dom(P)$, cannot be ordered." As above argued, equating absence of preference to incomparability does not seem to be the best choice to make, at least from the user point of view.

Interpreting a missing preference in terms of *indifference*, although not considered in CP-nets literature, in which indifference between values is assumed to be explicitly declared (see [1]), seems to be the only other possible alternative if one wants to stay with the ceteris paribus semantics: "if the user has not specified a preference on where to sit, then values in and out are *indifferent* given the it context." In terms of preference graph this interpretation amounts to add bidirectional arcs among tuples when they differ by values that are not ordered. In the example, this strategy would add, among others, an arc from t_1 to t_2 and vice versa. Although this works in our example (when only t_1 and t_3 are in r, now there is a path from t_1 to t_3 going through t_2), it brings up with new, more complex, problems to deal with. First, no strict partial order on tuples can be defined anymore (since the preference graph contains cycles), thus there is no consistent way to rank tuples. Second, indifference easily leads to the loss of "natural" dominance relations. As a simple example (taken from [1]), consider the CP-net over $X = \{A, B\}$ with statements $a_1 : b_1 > b_2$ and $a_2 : b_2 > b_1$ (both domains are binary). The optimal tuples in $dom(X)$ are (a_1, b_1) and (a_2, b_2). If we consider a_1 and a_2 indifferent, thus adding bidirectional arcs between (a_1, b_1) and (a_2, b_1) and between (a_1, b_2) and (a_2, b_2), there would be a path from any tuple t to any other tuple t', thus all tuples would be considered equally good.

It seems therefore that no adequate solution exists that is able to guarantee that apparently worse tuples are excluded from the optimal results and, at the same time, that "natural" preferences implied by the CP-net are preserved. The incomparability interpretation of missing preferences satisfies the second requirement but not the first one, the opposite is true for the indifference interpretation.

In the following we pursue an approach that tries to solve above dichotomy in two steps. First, in Section 3 we redefine the semantics of preference statements *and* the way the so-resulting preferences have to be combined, and show that the ceteris paribus semantics is not the only possible one for CP-nets. This provides us with a new, so called *totalitarian* (as opposed to ceteris paribus), semantics which is equivalent to cpu for complete CP-nets, yet it is more flexible if the CP-net is only partially specified. As a second step, in Section 4 we show how our semantics behaves on incomplete CP-nets and discuss possible extensions.

3. TOTALITARIAN SEMANTICS FOR CP-NETS

We start with a first, quite natural, way of interpreting preference statements in a totalitarian way.

Definition 2 *Let $\varphi = p : a_{i,1} > a_{i,2}$ be a statement in $CPT(A)$. The* strong totalitarian *(st) interpretation of φ is the set of pairs of tuples:*

$$\varphi^*_{st} = \{((p, a_1, y), (p, a_2, y')) | y, y' \in dom(X - P - \{A\})\}$$

Thus, tuples ordered by φ differ in the value of A and, possibly, also in the values of attributes Y not involved in φ.

Since the CP-net is locally consistent, the sets $\Phi^*_{A_i,\text{st}} = \bigcup_{\varphi \in CPT(A_i)} \varphi^*_{\text{st}}$ of preferences induced by $CPT(A_i)$ still have no conflicts inside. However, two tuples t_1 and t_2 might now be differently ordered by two $\Phi^*_{A_i,\text{st}}$ sets, thus taking their union could introduce cycles in the preference graph. As an example, given $\varphi = it : in > out$ and $\varphi' = it : low > high$ and the tuples $t_1 = (it, in, high)$ and $t_2 = (it, out, low)$, we have that $(t_1, t_2) \in \Phi^*_{T,\text{st}}$ and $(t_2, t_1) \in \Phi^*_{P,\text{st}}$, i.e., a cycle if we take the union of $\Phi^*_{T,\text{st}}$ and $\Phi^*_{P,\text{st}}$.

A way to preserve the strict partial order properties is to compose preferences in the $\Phi^*_{A_i,\text{st}}$ sets using a *Pareto rule*. Intuitively, this is to say that tuple t dominates t' iff it does so over at least one attribute and is never the case that this is true also for t'. More precisely, we have that $(t, t') \in \Phi^*_{\text{stp}}$ iff there exists an attribute A_i such that $(t, t') \in \Phi^*_{A_i,\text{st}}$ and for no attribute A_j it is $(t', t) \in \Phi^*_{A_j,\text{st}}$. The *strong totalitarian Pareto* (stp) order $\succ_{\text{stp}}$ is then defined as the transitive closure of Φ^*_{stp}.

Theorem 1 *For any complete acyclic CP-net N $\succ_{stp}$ is a strict partial order.*

PROOF. The proof is basically the same as that of Theorem 1 in [1], which proves that $\succ_{\text{cpu}}$ is a strict partial order for complete acyclic CP-nets. Any transitive relation $\succ$ (as, by definition, $\succ_{\text{stp}}$ is) can be *extended* into at least one total order $\sqsupset$ iff it is asymmetric, where the extension satisfies: if $t \succ t'$ then $t \sqsupset t'$. Then, it suffices to show that an extension exists for $\succ_{\text{stp}}$. The proof is by induction on the number of attributes in X. The base case, $n = 1$, follows from the hypothesis of completeness of the CP-net, which in turn implies local consistency. Assume the result holds for all CP-nets over $n - 1$ attributes and let N be a net with n attributes. Since N is acyclic, there is at least one attribute A with no parents. Let $dom(A) = \{a_1, a_2, \ldots, a_m\}$, with $a_1 > a_2 > \ldots > a_m$ being the total order specified by $CPT(A)$. Let $Y = X - \{A\}$. For each $a_i \in dom(A)$, consider the CP-net N_i over Y obtained from N by removing attribute A and in which all attributes having A as a parent have their CPT's restricted by keeping only statements in which $A = a_i$. By the inductive hypothesis we have that each N_i induces a strict partial order $\succ_{i,\text{stp}}$ over $dom(Y)$. Let $\sqsupset_i$ be an extension of $\succ_{i,\text{stp}}$. Consider now the total order $\sqsupset$ defined as follows: if $t[A] > t'[A]$ then $t \sqsupset t'$, if $t[A] = t'[A] = a_i$ and $t[Y] \sqsupset_i t'[Y]$ then $t \sqsupset t'$. It is immediate to verify that $t \succ_{\text{stp}} t'$ implies $t \sqsupset t'$. It follows that $\succ_{\text{stp}}$ is asymmetric, thus a strict partial order. □

Lemma 1 *For any complete acyclic CP-net N, $\succ_{cpu} \subseteq \succ_{stp}$.*

PROOF. We prove that $\Phi^*_{\text{cpu}} \subseteq \Phi^*_{\text{stp}}$, from which the result follows due to the monotonicity of the transitive closure operator. Let $(t, t') \in \Phi^*_{\text{cpu}}$. Thus, there is an attribute A such that $(t, t') \in \Phi^*_{A,\text{cp}}$. Clearly, (t, t') also belongs to $\Phi^*_{A,\text{st}}$, since $\Phi^*_{A,\text{cp}} \subseteq \Phi^*_{A,\text{st}}$ always holds. Since, due to the cpu semantics, t and t' differ only in the value of A, there can be no other attribute B such that $(t', t) \in \Phi^*_{B,\text{st}}$. It follows that $(t, t') \in \Phi^*_{\text{stp}}$. □

Above lemma shows that the strong totalitarian semantics includes *all* the ceteris paribus preferences. In many cases[5] it is also true that all the additional preferences in $\Phi^*_{\text{stp}} - \Phi^*_{\text{cpu}}$ are in the transitive closure of Φ^*_{cpu}, thus $\succ_{\text{stp}} = \succ_{\text{cpu}}$. For instance, this happens for the CP-net in Figure 1. However, this does not hold in general, since stp is sometimes unable to discover some *preference violations*, as the following example demonstrates.

Example 4 *Consider the CP-net in Figure 4, along with the preference graph of Φ^*_{cpu} (solid arcs). The figure also shows as dashed arcs 3 of the preferences in $\Phi^*_{stp} - \Phi^*_{cpu}$. While the one from (a_1, b_1, c_1) to (a_2, b_2, c_1), although not in Φ^*_{cpu} is in $\succ_{cpu}$ (there is a path in the Φ^*_{cpu} graph) the other two are* not *derivable using the cpu semantics. For instance, consider the pair $(t, t') = ((a_1, b_2, c_1), (a_2, b_2, c_2))$. This is in Φ^*_{stp} since t is better than t' on A, $t[B] = t'[B]$, and on attribute C the two tuples cannot be compared, since they have different parent values ($(a1, b_2)$ and (a_2, b_2), respectively). However, $CPT(C)$, written in the figure in a compact form, asserts that if $A = a_2$ or $B = b_2$ then preference is given to c_2 rather than to c_1. We have $t'[B] = t[B] = b_2$, $t'[C] = c_2$, and $t[C] = c_1$, thus t' should be better than t on attribute C, yet stp is unable to discover it.* □

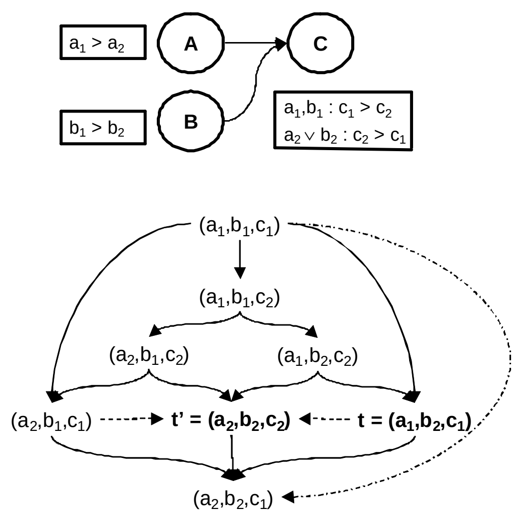


Figure 4: A CP-net for which the cpu and stp semantics do not coincide

Above example motivates the introduction of a new (weak) totalitarian semantics for interpreting the statements in a CPT.[6]

Definition 3 (Weak totalitarian Pareto semantics)
Let $a_1, a_2 \in dom(A)$ and t_1 and t_2 be two tuples with $t_1[A] = a_1$ and $t_2[A] = a_2$. Let P be the parents of A, with $p_1 = t_1[P]$ and $p_2 = t_2[P]$. If $CPT(A)$ includes statements (not necessarily distinct) $\varphi_1 = p_1 : a_1 > a_2$ and $\varphi_2 = p_2 : a_1 > a_2$ then $(t_1, t_2) \in \Phi^*_{A,wt}$.

The set of all preferences, Φ^*_{wtp}, is the n-ary Pareto composition of the $\Phi^*_{A_i,wt}$ sets, and the weak totalitarian Pareto *(wtp) order $\succ_{wtp}$ is the transitive closure of Φ^*_{wt}.*

[5]A precise characterization of the CP-nets for which this occurs seems to be a difficult problem, since it depends not only on the net structure, but also on its CPT's.

[6]Indeed, this new semantics induces more preferences than the strong one from the CPT's. However, the net effect is that *less* preferences among tuples survive after the Pareto composition, as Theorem 2 proves. This is why we say it is "weak".

Consider again Figure 4. In $CPT(C)$ there are two statements (once we write them in extended form), $\varphi_1 = a_1, b_2 : c_2 > c_1$ and $\varphi_2 = a_2, b_2 : c_2 > c_1$, from which we conclude, according to the above definition, that the pair $(t', t) \in \Phi^*_{C,\mathsf{wt}}$. Since $(t, t') \in \Phi^*_{A,\mathsf{wt}}$ still holds, it follows that $(t, t') \notin \Phi^*_{\mathsf{wtp}}$.

Since checking if $t \succ_{\mathsf{cpu}} t'$ is NP-hard for arbitrary complete acyclic CP-nets with binary domains [1],[7] it is clear that direct comparison of tuples, based on Definition 3, cannot be a conclusive test for checking dominance according to the wtp semantics. Indeed, Definition 3 offers a polynomial-time method able to check only if $(t, t') \in \Phi^*_{\mathsf{wtp}}$, yet it can be well the case that $(t, t') \notin \Phi^*_{\mathsf{wtp}}$ but $t \succ_{\mathsf{wtp}} t'$. The following example shows that reasoning on the transitive closure of Φ^*_{wtp} is indeed needed.

Example 5 *Consider the CP-net in Figure 5 over attributes $X = \{A, B, C\}$, along with the preference graph of Φ^*_{cpu}. Let $t = (a_1, b_1, c_2)$ and $t' = (a_2, b_1, c_1)$.*

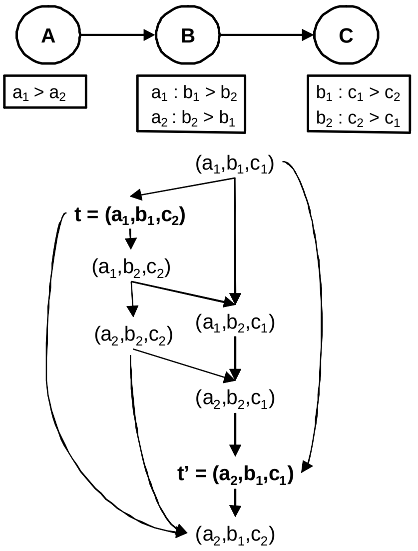


Figure 5: The CP-net for Example 5

It is easy to see that $(t, t') \notin \Phi^*_{wtp}$, since t is better than t' on A ($a_1 > a_2$), whereas t' is better than t on C ($b_1 : c_1 > c_2$). However, $t \succ_{wtp} t'$ holds, since both $(t, (a_2, b_2, c_1))$ and $((a_2, b_2, c_1), t')$ are in Φ^*_{wtp}. □

Given that we have redefined both statements' interpretation and the preference composition rule, the following is indeed rather surprising:

Theorem 2 *For any complete acyclic CP-net N, $\succ_{cpu} = \succ_{wtp}$.*

PROOF. We prove that 1) $\Phi^*_{\mathsf{cpu}} \subseteq \Phi^*_{\mathsf{wtp}}$, and 2) $\Phi^*_{\mathsf{wtp}} \subseteq \succ_{\mathsf{cpu}}$. The result then follows from monotonicity and idempotence of transitive closure.

($\Phi^*_{\mathsf{cpu}} \subseteq \Phi^*_{\mathsf{wtp}}$) The arguments in the proof of Lemma 1 still apply here.

($\Phi^*_{\mathsf{wtp}} \subseteq \succ_{\mathsf{cpu}}$). Let $(t, t') \in \Phi^*_{\mathsf{wtp}}$. We prove that there is a path from t to t' in the graph of Φ^*_{cpu}. The proof is by *double* induction, first on the number, m, of attributes on which t is better than t', while assuming that on the other $n - m$ attributes t and t' have the same values. Second, on the number, q, of attributes on which t and t' differ, yet t is not better than t' on such attributes. This covers all possibilities.

($q = 0$) The base case ($m = 1$), is when t and t' differ on a single attribute A_1, and t is better than t' on A_1. This is the case in which $(t, t') \in \Phi^*_{A_1,\mathsf{cp}}$. For the inductive step, assume the result holds for $m - 1$ attributes $Y = \{A_1, \ldots, A_{m-1}\}$ and consider that t is better than t' also on A_m, whose parents are P. Let $Z = X - Y - \{A_m\}$, $Z_P = Z \cap P$, and $Y_P = Y \cap P$. Since $q = 0$ it is is $t = (z, y, a_m)$ and $t' = (z, y', a'_m)$. For t being better than t' on A_m, $CPT(A_m)$ must have two statements $\varphi_1 = z_P, y_P : a_m > a'_m$ and $\varphi_2 = z_P, y'_P : a_m > a'_m$, where $z_P = z[Z_P]$, $y_P = y[Y_P]$, and $y'_P = y'[Y_P]$. From φ_1 we derive that $\Phi^*_{A_m,\mathsf{cp}}$ includes the pair (t, t''), where $t'' = (z, y, a'_m)$ differs from t just in the (worse) value a'_m. By the inductive hypothesis we have that $t'' \succ_{\mathsf{cpu}} t'$, thus $t \succ_{\mathsf{cpu}} t'$.

($q > 0$) The base case, $q = 0$, has been just proved. Then, assume the result holds for an arbitrary value of m and for $q - 1$ attributes W on which t and t' differ, yet t is not better than t' on them. We show how the result extends to q. Now we have that t and t' can be written, respectively, as $t = (z, y, w, b_q)$ and $t' = (z, y', w', b'_q)$, where z, y, and y' are as above, $w = t[W]$, $w' = t'[W]$, $b_q = t[B_q]$, and $b'_q = t'[B_q]$. Since, by hypothesis, $(t, t') \notin \Phi^*_{B_q,\mathsf{wt}}$, there are two statements matching t and t' parent values that order differently b_q and b'_q. Letting $W_P = W \cap P$, the statements are either $\varphi_1 = z_P, y_P, w_P : b_q > b'_q$ and $\varphi_2 = z_P, y'_P, w'_P : b'_q > b_q$, or $\varphi_3 = z_P, y_P, w_P : b'_q > b_q$ and $\varphi_4 = z_P, y'_P, w'_P : b_q > b'_q$, where $w_P = w[W_P]$ and $w'_P = w'[W_P]$. In the first case, from φ_1 we derive that $\Phi^*_{B_q,\mathsf{cp}}$ includes the pair (t, t''), where $t'' = (z, y, w, b'_q)$ differs from t just in the value of B_q. By the inductive hypothesis we have that $t'' \succ_{\mathsf{cpu}} t'$, thus $t \succ_{\mathsf{cpu}} t'$. Similarly, in the second case we infer from φ_4 that the pair (t''', t') is in $\Phi^*_{B_q,\mathsf{cp}}$, where $t''' = (z, y', w', b_q)$. By hypothesis it $t \succ_{\mathsf{cpu}} t'''$, thus $t \succ_{\mathsf{cpu}} t'$. □

Having shown that the weak totalitarian semantics is equivalent to ceteris paribus for complete CP-nets is an important result by itself. Indeed, since the usual interpretation of preferences in the DB field is totalitarian (see, e.g. [5, 10, 13]), our result shows that, at least for the case of complete CP-nets, this makes no difference at all, thus providing a contribution to bridge the gap between the AI and the DB approaches.

4. DEALING WITH INCOMPLETE CP-NETS

Let us now consider how the wtp semantics behaves on incomplete CP-nets. The following is immediate:

Lemma 2 *For any, possibly incomplete, acyclic CP-net N it holds that $\succ_{cpu} \subseteq \succ_{wtp}$.*

PROOF. The arguments in the proof of Lemma 1 still apply here, since they make no assumption on the completeness of the CP-net. □

Inclusion in the other direction, i.e., $\succ_{\mathsf{wtp}} \subseteq \succ_{\mathsf{cpu}}$, does not hold anymore. For instance, in the example at the beginning of this section it is $t_1 = (it, in, low) \succ_{\mathsf{wtp}} t_3 = (it, out, high)$ even if the statement $it : in > out$ has not been specified. This follows since $(t_1, t_3) \in \Phi^*_{P,\mathsf{wt}}$ (t_1 is better on price than t_3) and no preferences over other attributes involve these two tuples. Thus, wtp, unlike cpu, can indeed derive that t_3 is apparently sub-optimal given

[7] However, polynomial time algorithms exist for particular types of CP-net graphs G [1].

t_1. Similarly, for the CP-net in Figure 3 the wtp semantics better matches user intuition, in particular returning hotel Rome for the work context and hotels Continental, Jolly, and Holiday for the leisure context.

The fact that the optimal tuples in a relation r obtained from a CP-net N under the $\succ_{\text{wtp}}$ semantics, denoted as $Opt_{\text{wtp}}(r; N)$, are a subset of those of $\succ_{\text{cpu}}$, $Opt_{\text{cpu}}(r; N)$, is not a coincidence.

Corollary 1 *For any acyclic CP-net N and any relation r it holds that $Opt_{wtp}(r; N) \subseteq Opt_{cpu}(r; N)$.*

The result immediately follows from $\succ_{\text{cpu}}$ being a subset of $\succ_{\text{wtp}}$, and can be refined in the case of complete relations, $r = dom(X)$.

Theorem 3 *For any acyclic CP-net N it holds that $Opt_{wtp}(dom(X); N) = Opt_{cpu}(dom(X); N)$.*

PROOF. Thanks to Corollary 1, we only have to show that $Opt_{\text{cpu}}(dom(X); N) \subseteq Opt_{\text{wtp}}(dom(X); N)$, i.e., $t' \notin Opt_{\text{wtp}}(dom(X); N)$ implies $t' \notin Opt_{\text{cpu}}(dom(X); N)$.

If t' is not optimal with the wtp semantics, there exists at least one tuple t such that $(t, t') \in \Phi^*_{\text{wtp}}$. This is also to say that there exists an attribute A with parents P such that $(t, t') \in \Phi^*_{A,\text{wt}}$ and for no other attribute B it is $(t', t) \in \Phi^*_{B,\text{wt}}$. Let $t' = (y, p, a')$, where $p = t'[P]$ and $a' = t'[A]$. By hypothesis, $CPT(A)$ includes a statement $\varphi = p : a > a'$ (otherwise t' could not be dominated on attribute A by any tuple). Since r is complete, the tuple $t = (y, p, a)$ is in r. It follows that $(t, t') \in \Phi^*_{A,\text{cp}}$, thus t' is not optimal according to the cpu semantics. □

Thus, when all alternatives are available, optimal results of $\succ_{\text{cpu}}$ and $\succ_{\text{wtp}}$ are the same, regardless of the amount of incompleteness in the CP-net.

Although wtp appears to return qualitatively better results than cpu, in the general case it is *not* a strict partial order, as the following example shows.

Example 6 *Consider the CP-net in Figure 6 over attributes RestaurantType (R), Table (T), and SmokingArea (S), $dom(S) = \{yes, no\}$. As in Example 2, we have $it : in > out$ and $chn : out > in$, but now there is no preference on R (i.e., it and chn are not ordered). Preferences on S are conditional on T: if sitting inside, I do not want to stay in a smoking area ($in : no > yes$), but my preferences change should the table be outside ($out : yes > no$). According to Definition 3, we derive the following cycle of preferences (see also Figure 6):*

$$\begin{array}{lll} 1) & (it, in, yes) \succ_{wtp} (it, out, yes) \\ 2) & (it, out, yes) \succ_{wtp} (chn, out, no) \\ 3) & (chn, out, no) \succ_{wtp} (chn, in, no) \\ 4) & (chn, in, no) \succ_{wtp} (it, in, yes) \end{array}$$

Notice that 1) and 3) are also in $\succ_{cpu}$, whereas this is not the case for 2) and 4). □

A simple solution to avoid above problem would be to inhibit ordering tuples when they have unordered values in some attributes, i.e., assuming incomparability. This is exactly what the cpu semantics would do and, as argued at the beginning of Section 3, can easily lead to counterintuitive results.

We envision two, possibly complementary, approaches to solve the above problem. The first one is in the line of [3], in which an extended (ceteris paribus) *weak* preference semantics for CP-nets was proposed. This was motivated by the need of dealing with CP-nets $N = (G, CPT)$ in which the G graph is cyclic, thus cycles in

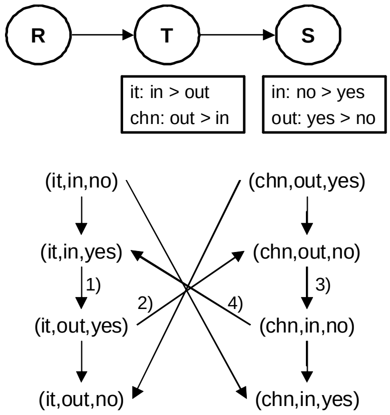


Figure 6: The CP-net for Example 6 and its $\succ_{\text{wtp}}$ order

the Φ^*_{cpu} graph might arise. Essentially, the idea is to replace *strict* preferences, like $t_1 \succ_{\text{cpu}} t_2$ (t_1 is strictly better than t_2), with *weak* ones, i.e., $t_1 \succeq_{\text{cpu}} t_2$, whose interpretation is "t_1 is at least as good as t_2". Then, it is $t_1 \succ_{\text{cpu}} t_2$ iff $t_1 \succeq_{\text{cpu}} t_2$ but not $t_2 \succeq_{\text{cpu}} t_1$. Essentially, with this weak semantics all tuples such that neither is strictly preferred to the others (i.e., they form a cycle in the Φ^*_{cpu} graph) are considered to be equivalent, and the optimal results are those tuples t for which either no tuple t' is at least as good as t (*strong* optimals, not involved in any cycle) or no tuple t' is strictly better than t (*weak* optimals).

Applying this approach to the case of wtp is conceptually straightforward. For instance, the CP-net in Figure 6 would have two strong optimal tuples ((it, in, no) and (chn, out, yes)). Should these not be in the database, the weak optimal results would be the four tuples involved in the cycle. We plan to carefully investigate this approach in the prosecution of our work. Here we just want to stress that, even when working with weak (rather than strict) preferences, the cpu and wtp semantics will not be equivalent on incomplete CP-nets. This can be immediately seen from Figure 6, in which cpu cannot order tuples with different restaurant types.

A second possibility for solving the problem is to extend the wtp semantics by: a) slightly revising the notion of what "being better on an attribute" means, and b) limiting the type of incompleteness in the CPT's. In the following we sketch the basic ideas of this alternative approach.

Consider first issue a). Referring to preference 2) in Example 6 (similar arguments hold for 4)), we see that the two tuples can be ordered only on S. However, looking at attribute T we might argue that, *since the parent values are unordered*, one should better consider comparing (it, out) and (chn, out) *as a whole*. Under this perspective, we could argue that (chn, out) is *better* than (it, out), since the latter does not respect the corresponding statement $it : in > out$. In other terms, when being unable to order parents' values, and only in this case, one should look at how good is the attribute value under consideration (out) within the two different contexts (it and chn).

Let us now turn to issue b), i.e., the type of incompleteness in the CPT's, and, for the sake of definiteness, consider first a CP-net in which all attributes have no parents (thus, all preferences are unconditional). In the most "liberal" case, the statements in $CPT(A)$ might induce a generic strict partial order on $dom(A)$. However, it is well known [5] that the Pareto composition of strict partial orders is *not* a strict partial order anymore.[8] As a simple example, if we have attributes A and B, and statements $a_1 > a_2$, $a_3 > a_4$, $b_2 > b_3$, and $b_4 > b_1$, these would lead to the cycle $(a_1, b_1) \succ_{\mathsf{wtp}} (a_2, b_2) \succ_{\mathsf{wtp}} (a_3, b_3) \succ_{\mathsf{wtp}} (a_4, b_4) \succ_{\mathsf{wtp}} (a_1, b_1)$.

This immediately rules out the possibility of having an uncontrolled amount of incompleteness. On the positive side, if the $CPT(A_i)$'s induce *weak orders*, their Pareto composition *is* a strict partial order. We remind that a weak order is a strict partial order that is also negatively transitive, i.e., for each triple of values a, b, c, if $a \not> b$ and $b \not> c$, then $a \not> c$. Clearly, a total order is also a weak order, but the converse is not necessarily true. More intuitively, a weak order can be viewed as a "linear order with ties". This is also to say that if a_1 and a_2 are not ordered, and $a_1 > a_3$, then it should be $a_2 > a_3$ as well. When attribute A has parents P, this restriction applies to each value in $dom(P)$. Clearly, if $p, p' \in dom(P)$ then the two weak orders they induce on $dom(A)$ need not to be the same. Finally, note that if no statement matching p is present in $CPT(A)$, then this induces a weak order in which all values are unordered.

For the CP-net in Example 6 (Figure 6), Figure 7 shows the (transitively reduced) preference graph that one would obtain from the, above informally defined, extended wtp semantics, which we might conveniently call wtp_I (the I subscript stays there to remind that this semantics is for incomplete CP-nets). It can be observed that tuples with unordered values can still be compared and that optimal tuples for the two R contexts also dominate sub-optimal tuples in the other context (e.g., $(chn, out, yes) \succ_{\mathsf{wtp}_I} (it, out, yes)$). Further, now we have $(it, out, yes) \not\succ_{\mathsf{wtp}_I} (chn, out, no)$ (this is preference 2) in Example 6), since the first tuple is (still) better on S, yet the second is now better on T, being it and chn unordered and out better in the chn context than in the it one.

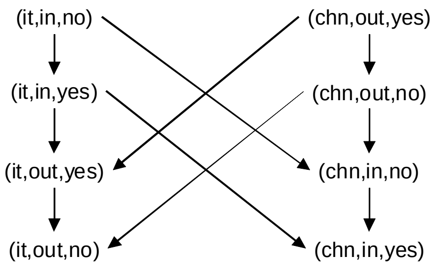


Figure 7: The $\succ_{\mathsf{wtp}_I}$ order induced by the CP-net in Figure 6

We have not worked out yet how the notion of "relative goodness", to be applied if parents' values are unordered, should be defined in the general case. At present, it is also not clear if this notion alone is sufficient to always guarantee acyclicity (we guess it is not) or if some further hypothesis (possibly on the CP-net topology) is needed. Nonetheless, combining relative goodness with a relaxed notion of preferences, in the line of [3], appears to be really promising for effectively dealing with incomplete CP-nets.

5. RELATED WORKS

In the DB field two major approaches have been proposed for the specification of qualitative (i.e., non-numeric) preferences. The logic-based approach by J. Chomicki [5] views preferences as expressed by a first-order binary formula P, where $t_1 \succ_P t_2$ iff $P(t_1, t_2)$ is true. In [6] it is conjectured that CP-nets can be represented in this formalism by making explicit their ceteris paribus semantics. However, no specific results for CP-nets or, more in general, for conditional preferences, are given.

A second approach, pioneered by W. Kießling [10], is based on an algebraic formalism through which a preference P is obtained by composing simpler preferences. Recently, Endres and Kießling have shown how the ceteris paribus semantics of CP-nets, and TCP-nets as well,[9] can be captured in this algebraic framework by means of a specific operator [8]. In the light of our result on the equivalence of totalitarian and ceteris paribus semantics, it would be interesting to see if the approach in [8] could possibly be simplified, with the aim of having a more compact algebraic CP-net representation.

The work by Pini et al. [11] considers (although not in a CP-net context) incompleteness and incomparability in the preferences of multiple agents that are to be aggregated, and focuses on the computation of *possible* and *necessary* winners (i.e., best alternatives). Our aim is somewhat different, since we have a single agent (user) and we work with conditional preferences. Further, [11] does not consider constraints, i.e., all alternatives are available (a complete DB, using our terminology).

The idea of "relative goodness", that we have informally introduced in Section 4, is somehow inspired to the work of Rossi et al. [12] on *partial* CP-nets, i.e., CP-nets in which some attributes are not ordered at all. However, [12] does not enter into details of partial CP-nets, nor it considers the general case in which an attribute is partially ordered (and possibly only for some parents' values), the focus being again on aggregation of multiple agents' preferences.

6. CONCLUSIONS

In this paper we have considered CP-nets as a viable tool to express user preferences in database queries, and have shown that their strength in compactly representing conditional preferences can be decoupled from the ceteris paribus (cpu) semantics. Our results show that one can use an alternative, totalitarian, semantics that is equivalent to cpu for complete CP-nets and that overcomes some limitation of cpu when preferences are only partially specified.

This being the first work that investigates the use of (incomplete) CP-nets for querying databases, many issues need to be addressed. In particular, a complete proof procedure for dominance testing under our semantics is needed, since this is at the basis of all algorithms for computing the optimal tuples in a relation [5, 10, 7]. We would also like to better understand the implications of having an explicit distinction between DB and context attributes, the intuition being that the latter are, for any given query, set to constant values (or to a set of constants). Finally, although we have provided evidence by means of specific examples that the totalitarian semantics is able to remedy some counterintuitive effects of cpu, a detailed analysis is still missing. This would provide us with a precise characterization of which kind of additional preferences are derived and how they can modify the result of a query with respect to the cpu semantics. Theorem 3, which shows that no modification will arise if the DB is complete, is a preliminary yet important step along this direction.

---

[8] W. Kießling defines Pareto composition in a more restrictive way, which guarantees that strict partial orders properties are preserved. However, his definition has the same problems of cpu semantics in being unable to order tuples when some attributes' values are unordered.

[9] TCP-nets, or *tradeoffs-enhanced* CP-nets [4], are an extension of CP-nets in which it is possible to have *relative importance* statements among attributes.

7. REFERENCES

[1] C. Boutilier, R. I. Brafman, C. Domshlak, H. 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] C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole. Reasoning With Conditional Ceteris Paribus Preference Statements. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI '99)*, pages 71–80, Stockholm, Sweden, July 1999.

[3] R. I. Brafman and Y. Dimopoulos. Extended Semantics and Optimization Algorithms for CP-networks. *Computational Intelligence*, 20(2):218–245, 2004.

[4] R. I. Brafman and C. Domshlak. Introducing Variable Importance Tradeoffs into CP-Nets. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI '02)*, pages 69–76, Edmonton, Alberta, Canada, Aug. 2002.

[5] J. Chomicki. Querying with Intrinsic Preferences. In *Proceedings of the 8th International Conference on Extending Database Technology (EDBT 2002)*, pages 34–51, Prague, Czech Republic, Mar. 2002.

[6] J. Chomicki and J. Song. Monotonic and Nonmonotonic Preference Revisions, Mar. 2005. arXiv:cs.DB/0503092v1.

[7] P. Ciaccia. Processing Preference Queries in Standard Database Systems. In *Proceedings of 4th International Conference on Advances in Information Systems (ADVIS 2006)*, pages 1–12, Izmir, Turkey, Oct. 2006. Invited paper.

[8] M. Endres and W. Kießling. Transformation of TCP-Net Queries into Preference Database Queries. In *Proceedings of the ECAI 2006 Multidisciplinary Workshop on Advances in Preference Handling*, pages 23–30, Riva del Garda, Italy, Aug. 2006.

[9] J. Goldsmith, J. Lang, M. Truszczynski, and N. Wilson. The Computational Complexity of Dominance and Consistency in CP-nets. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 144–149, Edinburgh, Scotland, UK, July 2005.

[10] W. Kießling. Foundations of Preferences in Database Systems. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 2002)*, pages 311–322, Hong Kong, China, Aug. 2002.

[11] M. S. Pini, F. Rossi, K. B. Venable, and T. Walsh. Incompleteness and Incomparability in Preference Aggregation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 1464–1469, Hyderabad, India, Jan. 2007.

[12] F. Rossi, K. B. Venable, and T. Walsh. mCP Nets: Representing and Reasoning with Preferences of Multiple Agents. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*, pages 729–734, San Jose, CA, July 2004.

[13] R. Torlone and P. Ciaccia. Which Are My Preferred Items? In *AH2002 Workshop on Recommendation and Personalization in eCommerce (RPeC 2002)*, pages 1–9, Malaga, Spain, May 2002.

[14] N. Wilson. Extending CP-Nets with Stronger Conditional Preference Statements. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*, pages 735–741, San Jose, CA, USA, July 2004.