Computing k-Rank Answers with Ontological CP-Nets

Tommaso Di Noia[1], Thomas Lukasiewicz[2], Maria Vanina Martinez[2],
Gerardo I. Simari[2], and Oana Tifrea-Marciuska[2]

[1]Dipartimento di Ingegneria Elettrica e dell'Informazione, Politecnico di Bari, Italy
t.dinoia@poliba.it
[2]Department of Computer Science, University of Oxford, UK
{thomas.lukasiewicz,vanina.martinez,
gerardo.simari,oana.tifrea}@cs.ox.ac.uk

Abstract. The tastes of a user can be represented in a natural way by using qualitative preferences. In this paper, we describe how to combine ontological knowledge with CP-nets to represent preferences in a qualitative way and enriched with domain knowledge. Specifically, we focus on conjunctive query (CQ) answering under CP-net-based preferences. We define k-rank answers to CQs based on the user's preferences encoded in an ontological CP-net and we provide an algorithm for k-rank answering CQs.

1 Introduction

From its inception, the Web has been centered around the idea of linking information to make it more accessible and useful for users. Recently, however, the Web has evolved at an increasing pace towards the so-called Web 3.0, where classical linked information lives together with ontological knowledge and social interactions of users. While the former allows for more precise and rich results in search and query answering tasks, the latter can be used to provide a personalized access to information. This requires new techniques for ranking results based not only on the link structure of documents but also on ontological and user-centred data, i.e., user preferences. These techniques cannot only be used for users but also for computer applications (e.g., preferences over different data sources) and agents (e.g., planning with preferences).

The study of preferences has been carried out in many different areas, such as philosophy, economics, and choice theory. They can be modeled in a qualitative and in a quantitative way, where quantitative preferences are associated with a number representing their worth, while qualitative preferences are related to each other via pairwise comparisons. Users are more comfortable assessing their preferences qualitatively, therefore this work uses qualitative preferences.

In this paper, we focus on the problem of ranking answers for conjunctive queries (CQs) to Datalog+/– ontologies, based on user preferences encoded in CP-nets [6]. We chose CP-nets, because they provide a natural, concise, and flexible graphical representation of qualitative preferences, and they are a widespread formalism to represent and reason with qualitative preferences. The Datalog+/– ontology language was chosen, because it is more expressive than *DL-Lite* and has a more compact representation of the attributes of concepts and roles [10]. The integration between the ontology and the CP-net is tight: on the one hand, CP-net outcomes are constrained by the ontology, and on the other hand, they directly inform how answers to CQs are ranked.

The main contributions of this paper are briefly as follows:

- We introduce ontological CP-nets, which are a novel combination of Datalog+/– with CP-nets, modeling preferences over ground atoms in Datalog+/– ontologies.
- We define CP-net-based CQs and their skyline and k-rank answers. We also provide an algorithm for computing k-rank answers to CQs based on the preferences encoded in an ontological CP-net.
- We analyze the complexity of k-rank answering CP-net-based CQs, providing precise complexity and tractability results. In detail, we have data tractability, as long as query answering in the underlying classical Datalog+/– ontology is data tractable, the CP-net is a polytree, and the query is fixed-width, bounded, or atomic.

The rest of this paper is organized as follows. In Section 2, we briefly recall Datalog+/– and CP-nets. Section 3 introduces ontological CP-nets, where preferences are expressed over ground atoms from a Datalog+/– ontology, as well as the syntax and the semantics of conjunctive queries (CQs) based on such CP-nets. In Section 4, we describe how to compute k-rank answers to CP-net-based CQs. Section 5 provides general complexity and tractability results. In Section 6, we discuss related work. Finally, Section 7 summarizes the main results and gives an outlook on future work.

2 Preliminaries

Before introducing ontological CP-nets, we recall the basics on Datalog+/– and CP-nets. We first introduce the Datalog+/– ontology language used in our formalism.

2.1 Datalog+/–

We now recall the main concepts of Datalog+/– [10], namely, relational databases, (Boolean) conjunctive queries ((B)CQs), tuple- and equality-generating dependencies (TGDs and EGDs, respectively), negative constraints, the chase, and ontologies.

Databases and Queries. We assume (i) an infinite universe of *(data) constants* Δ (which constitute the "normal" domain of a database), (ii) an infinite set of *(labeled) nulls* Δ_N (used as "fresh" Skolem terms, which are placeholders for unknown values, and can thus be seen as variables), and (iii) an infinite set of variables $\mathcal{V}$ (used in queries, dependencies, and constraints). Different constants represent different values (*unique name assumption*), while different nulls may represent the same value. We denote by $\mathbf{X}$ sequences of variables $X_1, \ldots, X_k$ with $k \geqslant 0$. We assume a *relational schema* $\mathcal{R}$, which is a finite set of *predicate symbols* (or simply *predicates*).

A *term* t is a constant, null, or variable. An *atomic formula* (or *atom*) $\mathbf{a}$ has the form $P(t_1, ..., t_n)$, where P is an n-ary predicate, and $t_1, ..., t_n$ are terms. A term or atom is *ground* iff it contains no nulls and no variables. A *database (instance)* D for a relational schema $\mathcal{R}$ is a (possibly infinite) set of atoms with predicates from $\mathcal{R}$ and arguments from $\Delta \cup \Delta_N$. A *conjunctive query (CQ)* over $\mathcal{R}$ has the form

$Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ is a conjunction of atoms (possibly equalities, but not inequalities) with the variables $\mathbf{X}$ and $\mathbf{Y}$, and possibly constants, but without nulls. We use $sign(Q)$ to denote the set of predicates appearing in Φ. A *Boolean CQ (BCQ)* over $\mathcal{R}$ is a CQ of the form $Q()$, often written as the set of all its atoms, without quantifiers. Answers to CQs and BCQs are defined via *homomorphisms*, which are mappings $\mu\colon \Delta \cup \Delta_N \cup \mathcal{V} \rightarrow \Delta \cup \Delta_N \cup \mathcal{V}$ such that (i) $c \in \Delta$ implies $\mu(c) = c$, (ii) $c \in \Delta_N$ implies $\mu(c) \in \Delta \cup \Delta_N$, and (iii) μ is naturally extended to atoms, sets of atoms, and conjunctions of atoms. The set of all *answers* to a CQ $Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$ over a database D, denoted $Q(D)$, is the set of all tuples $\mathbf{t}$ over Δ for which there exists a homomorphism $\mu\colon \mathbf{X} \cup \mathbf{Y} \rightarrow \Delta \cup \Delta_N$ such that $\mu(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq D$ and $\mu(\mathbf{X}) = \mathbf{t}$. The *answer* to a BCQ $Q()$ over a database D is *Yes*, denoted $D \models Q$, iff $Q(D) \neq \emptyset$.

Given a relational schema $\mathcal{R}$, a *tuple-generating dependency (TGD)* σ is a first-order formula of the form $\forall \mathbf{X} \forall \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \Psi(\mathbf{X}, \mathbf{Z})$, where $\Phi(\mathbf{X}, \mathbf{Y})$ and $\Psi(\mathbf{X}, \mathbf{Z})$ are conjunctions of atoms over $\mathcal{R}$ (without nulls), called the *body* and the *head* of σ, denoted $body(\sigma)$ and $head(\sigma)$, respectively. Such σ is satisfied in a database D for $\mathcal{R}$ iff, whenever there exists a homomorphism h that maps the atoms of $\Phi(\mathbf{X}, \mathbf{Y})$ to atoms of D, there exists an extension h' of h that maps the atoms of $\Psi(\mathbf{X}, \mathbf{Z})$ to atoms of D. Since TGDs can be reduced to TGDs with only single atoms in their heads (producing an equivalent set of TGDs), in the sequel, every TGD has w.l.o.g. a single atom in its head. A TGD σ is *guarded* iff it contains an atom in its body that contains all universally quantified variables of σ. A TGD σ is *linear* iff it contains only a single atom in its body. As set of TGDs is guarded (resp., linear) iff all its TGDs are guarded (resp., linear).

Query answering under TGDs, i.e., the evaluation of CQs and BCQs on databases under a set of TGDs is defined as follows. For a database D for $\mathcal{R}$, and a set of TGDs Σ on $\mathcal{R}$, the set of *models* of D and Σ, denoted $mods(D, \Sigma)$, is the set of all (possibly infinite) databases B such that (i) $D \subseteq B$ and (ii) every $\sigma \in \Sigma$ is satisfied in B. The set of *answers* for a CQ Q to D and Σ, denoted $ans(Q, D, \Sigma)$, is the set of all tuples $\mathbf{a}$ such that $\mathbf{a} \in Q(B)$ for all $B \in mods(D, \Sigma)$. The *answer* for a BCQ Q to D and Σ is *Yes*, denoted $D \cup \Sigma \models Q$, iff $ans(Q, D, \Sigma) \neq \emptyset$. Query answering under general TGDs is undecidable [1], even when $\mathcal{R}$ and Σ are fixed [8]. Decidability of query answering for the guarded case follows from a bounded tree-width property. Its data complexity in this case is P-complete.

A *negative constraint* (or simply *constraint*) γ is a first-order formula of the form $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow \bot$, where $\Phi(\mathbf{X})$ (called the *body* of γ) is a conjunction of atoms over $\mathcal{R}$ (without nulls). Under the standard semantics of query answering of BCQs in Datalog+/– with TGDs, adding negative constraints is computationally easy, as for each constraint $\forall \mathbf{X} \Phi(\mathbf{X}) \rightarrow \bot$, we only have to check that the BCQ $\exists \mathbf{X}\, \Phi(\mathbf{X})$ evaluates to false in D under Σ; if one of these checks fails, then the answer to the original BCQ Q is true, otherwise the constraints can simply be ignored when answering the BCQ Q.

As another component, the Datalog+/– language allows for special types of *equality-generating dependencies (EGDs)*. Since they can also be modeled via negative constraints, we omit them here, and we refer to [10] for their details. We usually omit the universal quantifiers in TGDs, negative constraints, and EGDs, and we implicitly assume that all sets of dependencies and/or constraints are finite.

The Chase. The *chase* was first introduced to enable checking implication of dependencies, and later also for checking query containment. By "chase", we refer both to the chase procedure and to its output. The TGD chase works on a database via so-called TGD *chase rules* (see [10] for further details and for an extended chase with also EGD chase rules). The (possibly infinite) chase of a database D relative to a set of TGDs Σ, denoted $chase(D,\Sigma)$, is a *universal model*, i.e., there is a homomorphism from $chase(D,\Sigma)$ onto every $B \in mods(D,\Sigma)$ [10]. Thus, BCQs Q over D and Σ can be evaluated on the chase for D and Σ, i.e., $D \cup \Sigma \models Q$ is equivalent to $chase(D,\Sigma) \models Q$. For guarded TGDs Σ, such BCQs Q can be evaluated on an initial fragment of $chase(D,\Sigma)$ of constant depth $k \cdot |Q|$, which is possible in polynomial time in the data complexity.

Datalog+/– Ontologies. A *Datalog+/– ontology* $O = (D,\Sigma)$, where $\Sigma = \Sigma_T \cup \Sigma_E \cup \Sigma_{NC}$, consists of a finite database D over Δ, a set of TGDs Σ_T, a set of non-conflicting EGDs Σ_E, and a set of negative constraints Σ_{NC}. We say O is *guarded* (resp., *linear*) iff Σ_T is guarded (resp., linear).

Example 1. Consider the following relational schema $\mathcal{R}$ representing booking information about airlines and their flights:

$$\begin{aligned}&\mathit{flight}(\mathit{id}, \mathit{d\text{-}airport}, \mathit{a\text{-}airport}, \mathit{dTime}, \mathit{aTime}, \mathit{plane\text{-}id}, \mathit{company});\\ &\mathit{plane}(\mathit{id}, \mathit{capacity});\quad \mathit{dPlace}(\mathit{city}, \mathit{airport});\\ &\mathit{book}(\mathit{flight\text{-}id}, \mathit{departure\text{-}date}, \mathit{class}).\end{aligned}$$

For example, the *flight* relation contains information about the flight such as the departure airport, plane id and the company that is responsible; A simple Datalog+/– ontology $O = (D,\Sigma)$ for flights is given below. Intuitively, the database D encodes that for example *f1*, *f2* and *f3* are 3 flights and *ma*,*na* are arrival times, etc. The set of constraints Σ encodes the domain and range of flights, the fact that a flight cannot have the same departure and arrival city, and an inverse relation between *flight* and *hasFlight*.

$$\begin{aligned}D = \{&\mathit{flight}(f_1, l_2, l_3, \mathit{nd}, \mathit{na}, p_2, a_1),\ \mathit{book}(f_1, d_1, e),\ \mathit{aTime}(\mathit{ma}),\\ &\mathit{flight}(f_2, l_2, l_1, \mathit{md}, \mathit{ma}, p_1, a_2),\ \mathit{book}(f_2, d_1, b),\ \mathit{aTime}(\mathit{na}),\\ &\mathit{flight}(f_3, l_3, l_2, \mathit{nd}, \mathit{na}, p_3, a_2),\ \mathit{book}(f_3, d_1, b),\ \mathit{dTime}(\mathit{nd}),\\ &\mathit{dTime}(\mathit{md}),\ \mathit{class}(b),\ \mathit{class}(e),\ \mathit{dPlace}(c_1, l_1),\ \mathit{airline}(a_1),\\ &\mathit{dPlace}(c_1, l_2),\ \mathit{dPlace}(c_2, l_3),\ \mathit{airline}(a_2)\};\end{aligned}$$

$$\begin{aligned}\Sigma = \{&\mathit{flight}(A,B,C,D,E,F,G) \rightarrow \exists Y\ \mathit{dPlace}(Y,B)\\ &\mathit{flight}(A,B,C,D,E,F,G) \rightarrow \mathit{dTime}(D) \wedge \mathit{aTime}(E)\\ &\mathit{flight}(A,B,C,D,E,F,G) \rightarrow \exists Y\ \mathit{plane}(F,Y)\\ &\mathit{flight}(A,B,C,D,E,F,G) \rightarrow \exists X\ \mathit{airline}(G,X)\\ &\mathit{flight}(A,B,B,D,E,F,G) \rightarrow \bot\\ &\mathit{flight}(A,B,C,D,E,F,G) \rightarrow \mathit{hasFlight}(G,A)\\ &\mathit{book}(A,B,C) \rightarrow \exists X\ \mathit{hasFlight}(X,A)\\ &\mathit{book}(A,B,C) \rightarrow \mathit{class}(C)\}. \qquad \blacksquare\end{aligned}$$

2.2 CP-Nets

Conditional preferences networks (CP-nets) [5] are a formalism to represent and reason with qualitative preferences. They allow the specification of preferences based on the notion of conditional preferential independence (CPI) [13].

We assume a set of variables $\mathbf{V}$, where each variable $X_i \in \mathbf{V}$ is associated with a *domain of values*, denoted $dom(X_i)$. The *domain of values* $\mathbf{x}$ of a set of variables $\mathbf{X} = \{X_1, \ldots, X_k\} \subseteq \mathbf{V}$, denoted $dom(\mathbf{X})$, is defined as $dom(X_1) \times \cdots \times dom(X_k)$. If $\mathbf{X} = \mathbf{V}$, then $\mathbf{x}$ is a *complete assignment* (*outcome*), otherwise it is a *partial assignment*. If **x** and **y** are assignments to disjoints sets $\mathbf{X}$ and $\mathbf{Y}$, then we denote the combination of **x** and **y** by **xy**.

A preference relation $\succeq$ is a total pre-order over the set of outcomes. We write $o_1 \succ o_2$ to state that o_1 is strictly preferred over o_2 while we write $o_1 \succeq o_2$ if o_1 is strictly or equally preferred to o_2. We say that o_2 is *dominated* by o_1 if $o_1 \succ o_2$ and that o_2 is *directly dominated* by o_1, $o_1 \succ_d o_2$ if o_2 is dominated by o_1 and there is no outcome o such that $o_1 \succ o$ and $o \succ o_2$. If there is no outcome o such that $o \succ o_1$ we say that o_1 is *undominated.* If both $o_1 \succeq o_2$ and $o_2 \succeq o_1$ hold, we say there is an *indifference* situation.

A conditional preference is represented as $(\mathbf{x} \succ \mathbf{x}' \mid \mathbf{z})$ meaning that "*given* $\mathbf{z}$*, I prefer* $\mathbf{x}$ *over* $\mathbf{x}'$". Let $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$ be nonempty sets that partition $\mathbf{V}$ and $\succ$ a preference relation over *dom*$(\mathbf{V})$. $\mathbf{X}$ is *conditionally preferentially independent* (CPI) of $\mathbf{Y}$ given $\mathbf{Z}$ iff for all **x, x'** $\in$ *dom*$(\mathbf{X}), \mathbf{y}, \mathbf{y}'$ $\in$*dom*$\mathbf{Y}$), **z** $\in$*dom*$(\mathbf{Z})$, we have $\mathbf{xyz} \succ \mathbf{x}'\mathbf{yz}$ iff $\mathbf{xy}'\mathbf{z} \succ \mathbf{x}'\mathbf{y}'\mathbf{z}$. Using CP-nets, one can model CPI statements. Formally, a CP-net $\mathcal{N}$ over $\mathbf{V}$ consists of an annotated directed graph $\mathcal{G}$ over $\{X_1, \ldots, X_n\}$ in which nodes stand for problem variables and edges represent conditioning among variables X_i. Each node X_i is annotated with a conditional preference table $CPT(X_i)$, that associates a total order $\succ^{X_i|\mathbf{u}}$ with each instantiation $\mathbf{u}$ of X_i's parents $Pa(X_i)$, i.e., $\mathbf{u} \in dom(Pa(X_i))$. In the sequel, we use $o \in \mathcal{N}$ to denote that o is an outcome of $\mathcal{N}$. The following are the two main computational tasks for CP-nets:

- *Dominance query*: given a CP-net and two outcomes o_1 and o_2, decide whether $o_1 \succ o_2$.
- *Outcome optimization*: given a CP-net, compute an undominated outcome.

Example 2. The CP-net in Figure 1 encodes that the morning departure (*md*) is preferred over night departure (*nd*), and morning arrival (*ma*) is preferred over night arrival (*na*). Business class (*b*) is preferred over economy class (*e*) when there is a night departure and a morning arrival, and the other way around, otherwise. ■

To establish an order among possible outcomes of a CP-net, we introduce the notion of *worsening flip*. This is a change in the value of a variable that worsens the satisfaction of the user's preferences. For example, in Figure 1, we have that $md\ ma\ e \succ md\ na\ e$. Based on the notion of *worsening flip*, we can derive a preference graph representing the transitive reduction of the preference relations among possible outcomes. In Figure 2, the preference graph related to the CP-net in Figure 1 is represented. Given two outcomes o_1 and o_2, an edge going from o_1 to o_2 means that $o_1 \succ o_2$.

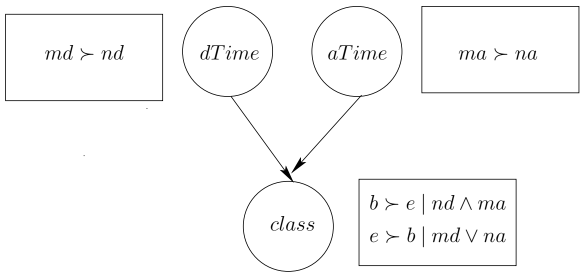


Fig. 1. An example of a CP-net with three variables.

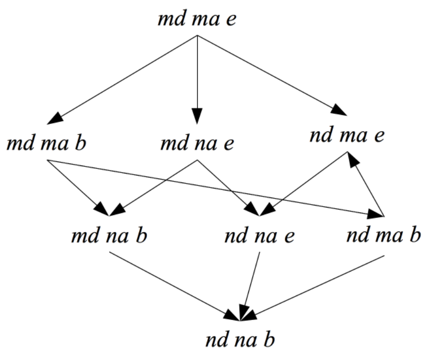


Fig. 2. The preference graph induced by the CP-net in Figure 1.

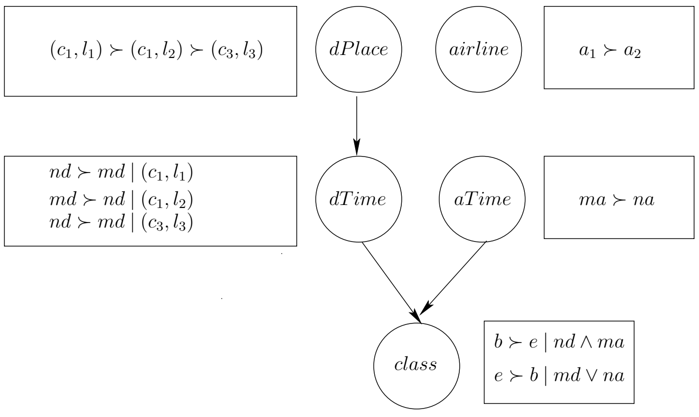
3 CP-Net-Based Conjunctive Queries

In this section, we introduce CP-Datalog+/–, which is an extension of Datalog+/– by CP-net based preferences. We first introduce the syntax and then the semantics of CP-Datalog+/–.

3.1 Syntax

We first define CP-net-based conjunctive queries, which are conjunctive queries along with a CP-net for defining a preference relation among the ground instances of the queries' atoms and thus among the queries' answers. To this end, we introduce ontological CP-nets, which informally define preference relations between conjunctions of ground atoms (which have as arguments constants from Δ as well as the special constant ν, which is a placeholder for any null). The preference relation is defined relative to an underlying ontology.

Definition 1 (Ontological CP-Net). Let O be a Datalog+/– ontology over the set of constants Δ. An *ontological CP-net* over O is a CP-net $\mathcal{N}$, which has as set of variables $\mathbf{V}$ a set of predicates from O, and as the *domain* of each $P \in \mathbf{V}$, denoted *dom*(P), a finite set of at least two different atoms $P(t_1, \ldots, t_k)$ with $t_1, \ldots, t_k \in \Delta \cup \{\nu\}$.



$(c_1, l_1) \succ (c_1, l_2) \succ (c_3, l_3)$
dPlace
airline
$a_1 \succ a_2$
$nd \succ md \mid (c_1, l_1)$
$md \succ nd \mid (c_1, l_2)$
$nd \succ md \mid (c_3, l_3)$
dTime
aTime
$ma \succ na$
class
$b \succ e \mid nd \wedge ma$
$e \succ b \mid md \vee na$

Fig. 3. An example of CP-net with five variables.

Observe that every outcome o of an ontological CP-net is a conjunction of grounded atoms. The following example illustrates the concept of ontological CP-net.

Example 3. Given the ontology O in Example 1, an ontological CP-net $\mathcal{N}$ over O is shown in Figure 3, where $\mathbf{V} = \{\mathit{dPlace}, \mathit{airline}, \mathit{dTime}, \mathit{aTime}, \mathit{class}\}$ and, for instance, $\mathit{dom}(\mathit{aTime}) = \{\mathit{airline}(a_1), \mathit{airline}(a_2)\}$. For clarity, in the CPT associated with the variable P, we use $\mathbf{t_i}$ to denote $P(\mathbf{t_i})$. Then, for instance, $\mathbf{t_i} \succ \mathbf{t_j}$ means $P(\mathbf{t_i}) \succ P(\mathbf{t_j})$.

The CP-net shown encodes that the departure place (*dPlace*) that maps city c_1 with airport location l_1 is more preferred than (c_1, l_2), that is more preferred than (c_3, l_3) (this might be dependent on where the user lives). Now, given the departure place, there are preferences on the departure time (how long it takes to reach the place, etc.). Thus, if the departure place is (c_1, l_1) or (c_3, l_3), then a night departure (*nd*) is preferred over a morning departure (*md*). If the departure place is (c_1, l_2), then a morning departure is preferred over a night departure. Business class (b) is preferred over economy class (e) when there is a night departure and a morning arrival, and the other way around, otherwise. Airline a_1 is preferred over a_2. ■

As a consequence of the underlying ontology, some of the outcomes of an ontological CP-net may be inconsistent, and some other outcomes may be ontologically equivalent. We thus have to assure that the preference relation encoded in an ontological CP-net is well-defined, which is expressed in the notion of consistency of ontological CP-nets. In the sequel, let $\succ^+$ denote the transitive closure of the preference relation $\succ$ induced by the ontological CP-net for its outcomes, and $\succ^+_\sim$ its restriction to the classes of equivalent consistent outcomes. The notion of consistency of CP-nets then describes the acyclicity of this preference relation $\succ^+_\sim$ and the $\succ^+$-incomparability of any two equivalent consistent outcomes. That is, to obtain $\succ^+_\sim$, we remove from $\succ^+$ all inconsistent outcomes relative to the ontology O, and we do not admit equivalent outcomes relative to O.

Definition 2 (Consistency of Ontological CP-Nets). Let O be a Datalog+/– ontology over $\varDelta$, and let $\mathcal{N}$ be an ontological CP-net over O. Then, $\mathcal{N}$ is *consistent* iff (i) $\succ^+_\sim$

is acyclic, and (ii) $o_1 \succ^+ o_2$ for no two outcomes o_1 and o_2 of $\mathcal{N}$ that are equivalent under O.

We next define CP-Net-based conjunctive queries, which are informally conjunctive queries along with an ontological CP-net for defining a preference relation among the queries' answers.

Definition 3 (CP-Net-Based Conjunctive Query). Let O be a Datalog+/– ontology. Then, a *CP-net-based conjunctive query (CP-net-based CQ)* $(Q, \mathcal{N})$ consists of a conjunctive query Q and a consistent ontological CP-net $\mathcal{N}$ over O such that all variables in $\mathcal{N}$ occur in Q.

W.l.o.g., all the atoms in Q have different predicates (which can be easily achieved by a predicate renaming), and thus there exists a bijection β from atoms in Q to the vertices of $\mathcal{N}$.

3.2 Semantics

We next define the semantics of CP-net-based CQs $(Q, \mathcal{N})$. The following definition formalizes answers to Q in the usual way and relates these answers to the outcomes of the CP-net $\mathcal{N}$.

Definition 4 (Answer to a CP-Net-based CQ). Let $O = (D, \Sigma)$ be a Datalog+/– ontology, and let $(Q, \mathcal{N})$ be a CP-net-based CQ with $Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$. Then, the set of all *answers* for $(Q, \mathcal{N})$ to O under the outcome o of $\mathcal{N}$, denoted $ans(Q, \mathcal{N}, O, o)$, is the set of all tuples $\mathbf{a}$ over Δ for which there exists a homomorphism $\mu \colon \mathbf{X} \cup \mathbf{Y} \rightarrow \Delta \cup \Delta_N$ such that (i) $\mu(\Phi(\mathbf{X}, \mathbf{Y})) \subseteq chase(D, \Sigma)$ and $\mu(\mathbf{X}) = \mathbf{a}$, and (ii) $o \subseteq \mu'(\Phi(\mathbf{X}, \mathbf{Y}))$, where (a) $\mu'|_{\mathbf{X}} = \mu|_{\mathbf{X}}$, (b) $\mu'(Y) = \mu(Y)$ for all $Y \in \mathbf{Y}$ such that $\mu(Y) \notin \Delta_N$ and (c) $\mu'(Y) = \nu$ for all $Y \in \mathbf{Y}$ such that $\mu(Y) \in \Delta_N$. The set of all *answers* for $(Q, \mathcal{N})$ to O, denoted $ans(Q, \mathcal{N}, O)$, is the set of all answers for $(Q, \mathcal{N})$ to O under some outcome o of $\mathcal{N}$.

Example 4. Consider again Example 3, which describes a consistent ontological CP-net $\mathcal{N}$, and let $Q(B, A, Z, C) =$

$$\exists X, Y\ \mathit{dPlace}(X, B) \wedge \mathit{airline}(A) \wedge \mathit{dTime}(Z) \wedge \mathit{aTime}(Y) \wedge \mathit{class}(C).$$

Then, $(Q, \mathcal{N})$ is a CP-net-based CQ, and $\langle l_1, a_1, \mathit{nd}, b\rangle$ is an answer to Q – note that its outcome in $\mathcal{N}$ is $o = \{\mathit{dPlace}(c_1, l_1), \mathit{airline}(a_1), \mathit{dTime}(\mathit{nd}), \mathit{aTime}(\mathit{ma}), \mathit{class}(b)\}$. ■

We now focus on how to order these answers based on the preferences of the user. We concentrate on skyline queries [3], a well-known class of queries for preference-based formalisms, and the iterated computation of skyline answers that allows us to assign a *rank* to every atom (using the CP-net); we refer to these as k-rank answers.

Definition 5 (Ordering Answers). Let $O = (D, \Sigma)$ be a Datalog+/– ontology, and let $(Q, \mathcal{N})$ be a CP-net-based CQ with $Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$. Let $\mathbf{a_1}$ and $\mathbf{a_2}$ be two different answers for $(Q, \mathcal{N})$ to O under the outcomes o_1 and o_2 of $\mathcal{N}$, respectively, such that $o_1 \succ o_2$. Then, we say that $\mathbf{a_1}$ is *ranked better than* $\mathbf{a_2}$.

We next define skyline and k-rank answers to CP-net-based CQs.

Definition 6 (Skyline Answer). Let $O = (D, \Sigma)$ be a Datalog+/– ontology, and let $(Q, \mathcal{N})$ be a CP-net-based CQ with $Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$. A *skyline answer* for $(Q, \mathcal{N})$ to O is any tuple $\mathbf{a}$ in $ans(Q, \mathcal{N}, O, o)$ for some undominated outcome o of $\mathcal{N}$.

Observe that skyline answers are not unique. Indeed, we may have more than one undominated outcome o of $\mathcal{N}$, and we may also have more than one homomorphism μ that satisfies the conditions (i) and (ii) of Definition 4 for the same undominated outcome o of $\mathcal{N}$.

Definition 7 (k-Rank Answer). Let $O = (D, \Sigma)$ be a Datalog+/– ontology, and let $(Q, \mathcal{N})$ be a CP-net-based CQ with $Q(\mathbf{X}) = \exists \mathbf{Y}\, \Phi(\mathbf{X}, \mathbf{Y})$. A *$k$-rank answer* for $(Q, \mathcal{N})$ to O *outside* a set of ground atoms S is a sequence $\langle \mathbf{a_1}, \ldots, \mathbf{a_k} \rangle$ such that either:

(a) $\mathbf{a_1}, \ldots, \mathbf{a_k}$ are k different skyline answers for $(Q, \mathcal{N})$ to O that do not belong to S, if $m \geqslant k$ such answers exist; or
(b) (1) $\mathbf{a_1}, \ldots, \mathbf{a_i}$ are all i different skyline answers for $(Q, \mathcal{N})$ to O that do not belong to S, and (2) $\langle \mathbf{a_{i+1}}, \ldots, \mathbf{a_k} \rangle$ is a $k{-}i$-rank answer for $(Q, \mathcal{N}{-}\{o\})$ to O outside $S \cup \{\mathbf{a_1}, \ldots, \mathbf{a_i}\}$, where o is an undominated outcome of $\mathcal{N}$, otherwise.

A *k-rank answer* for $(Q, \mathcal{N})$ to O is a k-rank answer for $(Q, \mathcal{N})$ to O outside $\emptyset$.

Note that when no answer for $(Q, \mathcal{N})$ to O exists, then $\langle\rangle$ is its unique k-rank answer. A k-rank answer is a sequence of k answers to a CP-net-based CQ ranked by following the order among outcomes induced by the ontological CP-net. Given two following answers in the sequence, they can be related to the same outcome (as for the skyline situation, the answer related to a single outcome may not be unique) or to two different outcomes in *directly dominated* relation. Finally, the computed answer is not deterministic. Indeed, a CP-net induces a partial order over the set of possible outcomes.

4 CP-Net-based Query Answering

We now present an algorithm to compute k-rank answers in an ontological CP-net-based setting. The procedure in Algorithm 1 exploits $\succ^{+}_{\sim}$ (the transitive closure of outcomes restricted to the class of equivalent consistent outcomes) to incrementally compute ordered answers to a conjunctive query Q over an ontology O and an ontological CP-net $\mathcal{N}$ for O. The algorithm stops the computation of answers when it reaches k different answers to Q. The most preferred solutions are the ones related to the undominated outcome of $\mathcal{N}$. This is the reason why the computation starts by adding o_u to the set *Outcomes* (line 7). Then, the algorithm incrementally adds directly dominated outcomes to *Outcomes* and finds answers to Q that are related to them. To preserve the preference order in $\succ^{+}_{\sim}$, in line 9, there is a check needed to avoid that an outcome in *Outcomes* is selected before we have answers related to other better outcomes.

For a better understanding of line 9, we refer to the preference graph in Figure 2. Suppose that *Outcomes* $= \{$*md ma* e, *md ma* b, *nd ma* b, *md na* b, *nd ma* b, *nd na* b,

```
Input: $O$ is a Datalog+/– ontology, $(Q,\mathcal{N})$ is a CP-net-based CQ, and $k \geqslant 0$.
Output: $k$-rank answers $\{\mathbf{a}_1,\ldots,\mathbf{a}_k\}$ to $(Q,\mathcal{N})$.
1  Result $\leftarrow \langle\rangle$;
2  Outcomes $\leftarrow \emptyset$;
3  Checked $\leftarrow \emptyset$ ;
4  k-reached $\leftarrow$ false;
5  Compute $\succ_{\sim}^{+}$ for $\mathcal{N}$;
6  Let $o_u \in \mathcal{N}$ be an undominated outcome;
7  Outcomes $\leftarrow$ Outcomes $\cup \{o_u\}$;
8  while k-reached = false do
9      choose $o \in$ Outcomes such that $\nexists o' \in$ Outcomes $-$ Checked , $o' \succ o$ and $o' \not\succ_d o$;
10     Checked $\leftarrow$ Checked $\cup \{o\}$;
11     foreach $\mu$ such that $\mu(\mathbf{X}) \in ans(Q,\mathcal{N},O,o)$ do
12         if k-reached = false and $\mu(\mathbf{X}) \notin$ Result then
13             Result $\leftarrow$ Result $\circ \langle\mu(\mathbf{X})\rangle$;
14             if $length($Result$) = k$ then
15                 k-reached $\leftarrow$ true;
16     foreach $o \in \mathcal{N}$ such that $\exists o'' \in$ Outcomes and $o'' \succ_d o$ do
17         Outcomes $\leftarrow$ Outcomes $\cup \{o\}$;
18 return Result .
```

Algorithm 1: k-Rank-Prefs $(O,\mathcal{N},Q,k)$

md na $e\}$ and *Checked* $=\{$*md ma* e, *md ma* b, *nd ma* b, *md na* $b\}$. Then, we have to select an outcome in $\{$*nd ma* b, *nd na* b, *md na* $e\}$. By Figure 2, *nd na* b is the worst choice. Indeed, although it is directly dominated by *nd ma* b, it is also dominated (not directly) by *md na* e.

Note that we may have two sources of non-determinism in the algorithm. In particular, in line 9, we may choose arbitrarily among different incomparable outcomes. In the situation that we have just discussed, we can select either *nd ma* b or *md na* e. The other source of non-determinism is in line 11. Here, we may have multiple equivalent (from the outcomes ordering point of view) μ, and after selecting some of them, we reach *length*(*Result*) $= k$.

Finally, an algorithm to compute skyline answers can be easily modeled by stopping the computation of answers to the ones related to the undominated outcome o_u.

5 Computational Complexity

In this section, we analyze the computational complexity of skyline and k-rank answering CP-net-based CQs $(Q,\mathcal{N})$ to Datalog+/– ontologies. We also delineate some tractable special cases.

5.1 General Results

The following theorem shows that k-rank answering CP-net-based CQs $(Q,\mathcal{N})$ to linear (resp., guarded) Datalog+/– ontologies is both complete for PSPACE (resp., 2EX-

PTIME) and that hardness holds even when Q has a bounded width, is fixed, or is an atom. The lower complexity bounds follow from the result that the more specialized problem of answering BCQs to linear (resp., guarded) Datalog+/– ontologies is complete for PSPACE (resp., 2EXPTIME) [9], where hardness holds even in the case of bounded-width, fixed, or atomic BCQs. As for the upper complexity bound, we first have to decide the consistency of $\mathcal{N}$, which can be done in PSPACE in the linear case (despite the number of outcomes of $\mathcal{N}$ being exponential, as we only have to store maximally four outcomes) and in 2EXPTIME in the guarded case, by (i) deciding BCQs to linear (resp., guarded) Datalog+/– ontologies, which is complete for PSPACE (resp., 2EXPTIME) [10], and (ii) deciding dominance between two outcomes in a standard CP-net, which is PSPACE-complete [11]. For the actual k-rank answering, intuitively, we first compute k ordered top outcomes of $\mathcal{N}$, which can also be done in PSPACE (resp., 2EXPTIME), following a similar line of argumentation as for consistency, and we then use these outcomes to instantiate Q and evaluate the resulting CQ Q' over O, which is in PSPACE (resp., 2EXPTIME).

Theorem 1. *Let $k \geqslant 0$ be fixed. Given a linear (resp., guarded) Datalog+/– ontology O and a CP-net-based CQ $(Q, \mathcal{N})$, computing the k-rank answer for $(Q, \mathcal{N})$ is complete for* PSPACE *(resp.,* 2EXPTIME*). Hardness holds even when Q has a bounded width, is fixed, or is an atom.*

The next theorem shows that k-rank answering CP-net-based CQs $(Q, \mathcal{N})$ to linear (resp., guarded) Datalog+/– ontologies in the data complexity (where Σ is fixed) is both complete for PSPACE. The lower complexity bounds follow from the result that the more specialized problem of deciding dominance between two outcomes in a standard CP-net is PSPACE-complete [11]. As for the upper complexity bound, we first have to decide the consistency of $\mathcal{N}$, which can be done in PSPACE, by (i) deciding BCQs to linear or guarded Datalog+/– ontologies, which is NP-complete in the data complexity [10], and (ii) deciding dominance between two outcomes in a standard CP-net, which is PSPACE-complete [11]. For the actual k-rank answering, intuitively, we first compute k ordered top outcomes, which is in PSPACE, by a similar argumentation as for consistency, and we then use these outcomes to instantiate Q and evaluate the resulting CQ Q' over O, which is in NP and thus in PSPACE.

Theorem 2. *Let $k \geqslant 0$ be fixed. Given a linear or guarded Datalog+/– ontology O, and a CP-net-based CQ $(Q, \mathcal{N})$, computing the k-rank answer for $(Q, \mathcal{N})$ is data complete for* PSPACE.

5.2 Tractability Results

We now delineate special cases where skyline and k-rank answering CP-net-based CQs $(Q, \mathcal{N})$ to Datalog+/– ontologies $O = (D, \Sigma)$ is tractable in the data complexity (where Σ is fixed). More precisely, the following result shows that these two problems are tractable in the data complexity, when (i) O is linear or guarded, (ii) $\mathcal{N}$ is a polytree, and (iii) Q has a bounded width, is fixed, or is an atom (which also implies that $\mathcal{N}$ is bounded). It follows from the results (1) that answering bounded-width, fixed, or atomic BCQs to linear or guarded Datalog+/– ontologies can be done in polynomial time [10], and (2) that for standard polytree CP-nets (of bounded node in-degree), dominance between two outcomes can be decided in polynomial time [5]. Intuitively, we first decide whether $\mathcal{N}$ is consistent, which can be done in polynomial time, since the number of outcomes of $\mathcal{N}$ is polynomial (by the above assumptions), deciding equivalence and inconsistency of outcomes can be done in polynomial time, and deciding dominance of two outcomes can also be done in polynomial time. We then order the outcomes of $\mathcal{N}$ along their preferences, and use them in this order to instantiate Q and evaluate the resulting CQ Q' over O, which can be done in polynomial time.

Theorem 3. *Let $k \geqslant 0$ be fixed. Given a linear or guarded Datalog+/– ontology O, and a CP-net-based CQ $(Q, \mathcal{N})$, where $\mathcal{N}$ is a polytree, and Q has a bounded width, is fixed, or is an atom, computing the skyline and the k-rank answer for $(Q, \mathcal{N})$ can both be done in polynomial time in the data complexity.*

6 Related Work

Modeling and dealing with preferences in databases has been studied for almost three decades, since the seminal work of [14]; see [19] for a survey of notable works in this line. Work has also been carried out in the intersection with databases and knowledge representation and reasoning, such as preference logic programs [12], incorporation of preferences into formalisms such as answer set programs [7], and answering k-rank queries in ontological languages [15].

In the philosophical tradition, preferences are usually expressed over mutually exclusive "worlds", such as truth assignments to formulas. The work of [2] is framed in this interpretation of preferences, aiming at bridging the gap between several formalisms from the AI community such as CP-nets and those studied traditionally in philosophy. In this regard, CP-nets [5] is one of the most widely known formalisms. Most of the work on CP-nets has focused on the computation of optimal outcomes and the problem of dominance testing, *i.e.*, to check if one outcome of the CP-net is preferred to another. More recently, the work of Wang et al. [20] proposes an efficient algorithm and indexing scheme for top k retrieval in CP-nets.

Recently, there has been some interest regarding the combination of Semantic Web technologies with preference representation and reasoning. A combination of conditional preferences (very different from CP -nets) with description logic (DL) reasoning for ranking objects is presented in [16]. There, conditional preferences are exploited in the definition of a ranking function that allows to perform a semantic personalized search and ranking over a set of resources annotated via an ontological description. In [15], Datalog+/– is extended with preference management formalisms closely related to those previously studied for relational databases. The authors focus on two kinds of answers to queries, skyline and k-rank (a generalization of top-k queries), and develop algorithms for disjunctions of atomic queries and conjunctive queries.

Closest in spirit to this paper is perhaps the preference formalism that combines CP-nets and DLs in [17], where variable values of CP-nets are satisfiable DL formulas. The main difference between this and the proposal here lies in the relationship between the ontology and the CP-net. While [17] uses ontological axioms to restrict CP-net outcomes, here we use the preference information contained in the CP-net to inform how answers to queries over the ontology should be ranked. Finally, in an information retrieval context in [4], Wordnet is used to add a semantics to CP-net variables. Another interesting approach to mixing qualitative preferences with Semantic Web technology is presented in [18], where an extension of SPARQL is studied that can encode user preferences in the query.

7 Summary and Outlook

We have introduced ontological CP-nets, which are a novel combination of Datalog+/– ontologies with CP-nets. We have defined CP-net-based CQs and their skyline and k-rank answers on top of ontological CP-nets. We have also provided an algorithm for computing k-rank answers to CP-net-based CQs. Furthermore, we have provided precise complexity and tractability results for this problem.

Interesting topics of ongoing and future research include the implementation and experimental evaluation of the approach, as well as a more complete complexity analysis for other Datalog+/– variants and ontology languages as well as other special cases of CP-nets.

Acknowledgments. This work was supported by the UK EPSRC grant EP/J008346/1 ("PrOQAW"), by a Google European Doctoral Fellowship, by the EU (FP7/2007-2013) ERC grant 246858 ("DIADEM"), by a Yahoo! Research Fellowship, and by the grant PON02_00563_3470993 ("VINCENTE").

References

1. C. Beeri and M. Y. Vardi. The implication problem for data dependencies. In *Proc. of ICALP*, pages 73–85, 1981.
2. M. Bienvenu, J. Lang, and N. Wilson. From preference logics to preference languages, and back. In *Proc. of KR*, pages 214–224, 2010.
3. S. Börzsönyi, D. Kossmann, and K. Stocker. The skyline operator. In *Proc. ICDE*, pages 421–430, 2001.
4. F. Boubekeur, M. Boughanem, and L. Tamine-Lechani. Semantic information retrieval based on CP-Nets. In *Proc.of FUZZ-IEEE*, pages 1 –7, 2007.
5. 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. *JAIR*, 21:135–191, 2004.
6. C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *Proc. of UAI*, pages 71–80, 1999.
7. G. Brewka. Preferences, contexts and answer sets. In *Proc. of ICLP*, page 22, 2007.
8. A. Calì, G. Gottlob, and M. Kifer. Taming the infinite chase: Query answering under expressive relational constraints. In *Proc. of KR*, pages 70–80. AAAI Press, 2008.
9. A. Calì, G. Gottlob, and T. Lukasiewicz. Datalog$^{\pm}$: A unified approach to ontologies and integrity constraints. In *Proc. of ICDT*, pages 14–30, 2009.
10. A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *J. Web Sem.*, 14:57–83, 2012.

11. J. Goldsmith, J. Lang, M. Truszczynski, and N. Wilson. The Computational Complexity of Dominance and Consistency in CP-Nets. *JAIR*, 33:403–432, 2008.
12. K. Govindarajan, B. Jayaraman, and S. Mantha. Preference logic programming. In *Proc. of ICLP*, pages 731–745, 1995.
13. R. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-Offs*. Cambridge University Press, 1993.
14. M. Lacroix and P. Lavency. Preferences: Putting more knowledge into queries. In *Proc. of VLDB*, pages 1–4. Morgan Kaufmann, 1987.
15. T. Lukasiewicz, M. Martinez, and G. I. Simari. Preference-Based Query Answering in Datalog+/- Ontologies. In *Proc. of IJCAI*, pages 501–518, 2013.
16. T. Lukasiewicz and J. Schellhase. Variable-strength conditional preferences for ranking objects in ontologies. *J. Web Sem*, 5(3):180–194, 2007.
17. T. D. Noia, T. Lukasiewicz, and G. Simari. Reasoning with semantic-enabled qualitative preferences. In *Proc. of SUM*, pages 374–386. 2013.
18. W. Siberski, J. Z. Pan, and U. Thaden. Querying the Semantic Web with preferences. In *Proc. of ISWC*, pages 612–624, 2006.
19. K. Stefanidis, G. Koutrika, and E. Pitoura. A survey on representation, composition and application of preferences in database systems. *ACM TODS*, 36, 2011.
20. H. Wang, X. Zhou, W. Chen, and P. Ma. Top-k retrieval using conditional preference networks. In *Proc. of CIKM*, pages 2075–2079, 2012.