Reasoning with DL-Based CP-Nets

Tommaso Di Noia[1], Thomas Lukasiewicz[2], and Gerardo I. Simari[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
firstname.lastname@cs.ox.ac.uk

Abstract. Preference representation and reasoning is a key issue in many real-world scenarios where a personalized access to information is needed. Currently, there are many approaches allowing a system to assess preferences in a qualitative or quantitative way, and among the qualitative ones the most prominent are CP-nets. Their clear graphical structure unifies an easy representation of user desires with nice computational properties when computing the best outcome. In this paper, we show how to reason with CP-nets when the attributes modeling the knowledge domain have an ontological structure or, in other words, variable values are DL formulas constrained relative to an underlying domain ontology. We also show how the computation of Pareto-optimal outcomes for an ontological CP-net can be reduced to the solution of constraint satisfaction problems.

1 Introduction

During the recent years, several revolutionary changes are taking place on the classical Web. First, the so-called Web of Data is more and more being realized as a special case of the Semantic Web. Second, as part of the Social Web, users are acting more and more as first-class citizens in the creation and delivery of contents on the Web. The combination of these two technological waves is called the *Social Semantic Web* (or also *Web 3.0*), where the classical Web of interlinked documents is more and more turning into (i) semantic data and tags constrained by ontologies, and (ii) social data, such as connections, interactions, reviews, and tags.

The Web is thus shifting away from data on linked Web pages towards less interlinked data in social networks on the Web relative to underlying ontologies. This requires new technologies for search and query answering, where the ranking of search results is not based on the link structure between Web pages anymore, but on the information available in the Social Semantic Web, in particular, the underlying ontological knowledge and the preferences of the users. Given a query, these latter play a fundamental role when a crisp yes/no answer is not enough to satisfy a user's needs, since there is a certain degree of uncertainty in possible answers [9].

We have two main ways of modeling preferences: (a) *quantitative preferences* are associated with a number representing their worth or they are represented as an ordered set of objects (e.g., "my preference for WiFi connection is 0.8" and "my preference for cable connection is 0.4"), while (b) *qualitative preferences* are related to each other via pairwise comparisons (e.g., "I prefer WiFi over cable connection").

In many applications in practice, the qualitative approach is a more natural way of representing preferences, since humans are often not very comfortable in expressing their "wishes" in terms of a numerical value. To have a quantitative representation of her preferences, the user needs to explicitly determine a value for a large number of alternatives usually described by more than one attribute. It is generally much easier to provide information about preferences as pairwise qualitative comparisons [9]. One of the most powerful qualitative frameworks for preference representation and reasoning are perhaps CP-nets [3]. They are a graphical language that unifies an easy representation of user desires with nice computational properties when computing the best outcome. Most of the work done with CP-nets and more generally with preference representation mainly deals with a propositional representation of preferences. In this paper, we propose an enhancement of CP-nets by adding ontological information associated to preferences. This is an initial step towards a new type of semantic search techniques that can go far beyond PageRank and similar algorithms. They will be able to exploit social information, e.g., information coming from social networks, and model it as semantic-enabled user preferences.

The rest of this paper is organized as follows. In Section 2, we briefly recall CP-nets. Section 3 introduces ontological CP-nets, i.e., CP-nets enriched with ontological descriptions, and it describes how to compute optimal outcomes. In Sections 4 and 5, we provide complexity results and discuss related work, respectively. Finally, we give a summary of the results in this paper and an outlook on future work.

2 Preliminaries

We start by introducing some notions and formalisms that are necessary to present our framework. Given a set of variables $\mathcal{V}$, an *outcome* $v \in Dom(\mathcal{V})$ is an assignment of a domain value $x \in Dom(X)$ to every variable $X \in \mathcal{V}$. A *preference relation* $\succeq$ is a total pre-order over the set of all outcomes. We write $o_1 \succ o_2$ (resp., $o_1 \succeq o_2$) to denote that o_1 is strictly preferred (resp., strictly or equally preferred) to o_2. If $o_1 \succ o_2$, then o_2 is *dominated* by o_1. If there is no outcome o such that $o \succ o_1$, then o_1 is *undominated*.

A *conditional preference* is an expression $(\alpha \succ \beta \mid \gamma)$, where α, β, and γ are formulas. It intuitively means that "*given γ, I prefer α over β*". In the following, we often write $(\alpha \mid \gamma)$ and $(\neg\alpha \mid \gamma)$ to denote $(\alpha \succ \neg\alpha \mid \gamma)$ and $(\neg\alpha \succ \alpha \mid \gamma)$, respectively, and we use $\tilde{\alpha}$ to represent one of the elements among α and $\neg\alpha$.

2.1 CP-Nets

Conditional preferences networks (CP-nets) [3] are a formalism to represent and reason about qualitative preferences. They are a compact but powerful language, which allows the specification of preferences based on the notion of conditional preferential independence. Fundamental to CP-nets is the notion of *conditional preferential independence (CPI)* [14]. Let $A, B \in \mathcal{V}$ be two variables and $\mathcal{R} \subset \mathcal{V}$ be a set of variables such that A, B, and $\mathcal{R}$ partition $\mathcal{V}$, and $Dom(A)$, $Dom(B)$, and $Dom(\mathcal{R})$ represent all possible assignments to A, B, and all the variables in $\mathcal{R}$, respectively. We say that A is *conditionally preferentially independent (CPI)* of B given an assignment $\rho \in Dom(\mathcal{R})$ iff, for every $\alpha_1, \alpha_2 \in Dom(A)$ and $\beta_1, \beta_2 \in Dom(B)$, we have that $\alpha_1\beta_1\rho \succ \alpha_2\beta_1\rho$ iff $\alpha_1\beta_2\rho \succ \alpha_2\beta_2\rho$. Here, $\succ$ represents the preference order on assignments to sets of variables. CP-nets are a graphical language to model CPI statements. Formally, a ***CP-net*** N consists of a directed graph G over a set of variables $\mathcal{V} = \{A_i \mid i \in \{1, \ldots, n\}\}$ as nodes, along with a conditional preference table $CPT(A_i)$ for every variable A_i, which contains a preference for each pair of values of A_i conditioned to all possible assignments to the parents of A_i in G. Given a CP-net N, we denote by $\mathcal{CPT}_i$ the set of all conditional preferences represented by $CPT(A_i)$, and we define $\mathcal{CPT}_N = \{\mathcal{CPT}_i \mid i \in \{1, \ldots, n\}\}$. An example of a CP-net (over only binary variables) is shown in Fig. 1.

Example 1 (Hotel). A CP-net for representing preferences for hotel accommodations is shown in Fig. 1. Note that this is a toy example, whose purpose is to show the representational expressiveness of CP-nets in modeling user profiles. In this simple case, we use five binary variables of the following meaning:

α_1: the hotel is located near the sea;
α_2: the hotel is located in the city center;
α_3: scooters for rent;
α_4: parking available;
α_5: bikes for rent.

Looking at $CPT(A_3)$, e.g., we see that the user prefers to have a scooter for rent in case the hotel is located near the sea or in the city center, and that she prefers to not have a scooter for rent if the hotel is neither near the sea nor in the city center. ■

To establish an order among possible outcomes of a CP-net, we introduce the notion of *worsening flip*, which is a change in the value of a variable that worsens the satisfaction of user preferences. As an example, if we consider the CP-net in Fig. 1, we have a worsening flip moving from $\alpha_1\alpha_2\alpha_3\neg\alpha_4\neg\alpha_5$ to $\alpha_1\alpha_2\neg\alpha_3\neg\alpha_4\neg\alpha_5$. Indeed, given $\alpha_1\alpha_2\neg\alpha_4\neg\alpha_5$, we see that α_3 is preferred over $\neg\alpha_3$. Based on this notion, we can state that $\alpha_1\alpha_2\alpha_3\neg\alpha_4\neg\alpha_5 \succ \alpha_1\alpha_2\neg\alpha_3\neg\alpha_4\neg\alpha_5$.

Given a CP-net, the two main queries that one may ask are:

– *dominance query*: given two outcomes o_1 and o_2, does $o_1 \succ o_2$ hold?
– *outcome optimization*: compute an optimal (i.e., undominated) outcome for the preferences represented by a given CP-net.

Given an acyclic CP-net, one can compute the best outcome in linear time. The algorithm just follows the order among variables represented by the graph and assigns values to the variables A_i from top to bottom satisfying the preference order in the corresponding $CPT(A_i)$. For example, in the CP-net in Fig. 1, the optimal outcome is $\alpha_1\alpha_2\alpha_3\alpha_4\alpha_5$. Finding optimal outcomes in cyclic CP-nets is NP-hard.

2.2 Constrained CP-Nets

In constrained CP-nets [19, 4], constraints among variables are added to the basic formalism of CP-nets. Adding constraints among variables may reduce the set of possible outcomes $\mathcal{O}$. The approach to finding the optimal outcomes proposed in [19] relies on a reduction of the preferences represented in the CP-net to a set of hard constraints (which

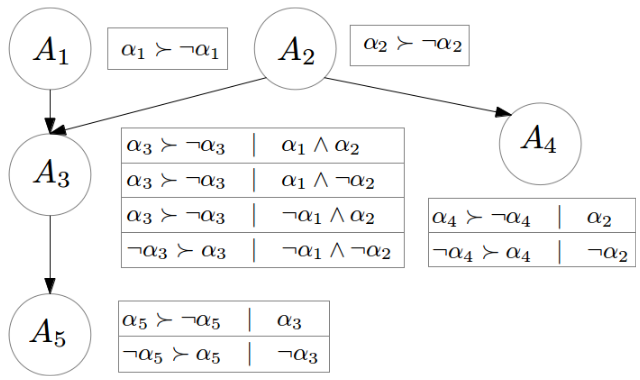

Fig. 1. An example of a CP-net over five binary variables.

can be represented in clause form for binary variables), taking into account the variables occurring in the preferences. Given a CP-net N and a set of constraints $\mathcal{C}$, an outcome o is *feasible* iff it satisfies all the constraints in $\mathcal{C}$. A feasible outcome is *Pareto optimal* [4] iff it is undominated among all feasible outcomes. These optimal outcomes now correspond to the solutions of a constraint satisfaction problem. For binary variables, given a conditional preference $(\alpha_{n+1} \mid \alpha_1 \wedge \cdots \wedge \alpha_n)$, the corresponding constraint is the clause

$$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \alpha_{n+1}. \tag{1}$$

Given a CP-net N and a set of constraints $\mathcal{C}$, a feasible Pareto optimal outcome is exactly an assignment satisfying the corresponding set of clauses and all constraints in $\mathcal{C}$. We refer the reader to [19, 4] for further details, including examples.

3 Ontological CP-Nets

We now introduce a framework for preference representation that is harnessing the technologies described in the previous section. The idea is to combine CP-nets and DLs. In the framework that we propose here, variable values are satisfiable DL formulas.

We consider only binary variables here. Two conditional preferences $(\alpha \mid \gamma)$ and $(\alpha' \mid \gamma')$ are *equivalent* under an ontology $\mathcal{T}$ iff $\gamma \equiv_{\mathcal{T}} \gamma'$ and $\alpha \equiv_{\mathcal{T}} \alpha'$.

Definition 1 (ontological CP-net). An *ontological CP-net* $(N, \mathcal{T})$ consists of a CP-net N and an ontology $\mathcal{T}$ such that:

(i) for each variable $A \in \mathcal{V}$, it holds that $Dom(A) = \{\alpha, \neg\alpha\}$, where both α and $\neg\alpha$ are DL formulas that are satisfiable relative to $\mathcal{T}$;
(ii) all the conditional preferences in $\mathcal{CPT}_N$ are pairwise not equivalent.

Note that even if we do not have any explicit hard constraint expressed among the variables of the CP-net, due to the underlying ontology, we have a set of implicit constraints among the values of the variables $\mathcal{V}$ in the CP-net. We show in Section 3.2 how to explicitly encode such constraints to compute an optimal outcome.

Example 2 (Hotel cont'd). Consider a simple ontology $\mathcal{T}$, describing the services offered by a hotel, and consisting of the following four axioms:

$$functional(\textit{rent});$$
$$\textit{Scooter} \sqsubseteq \textit{Motorcycle};$$
$$\textit{Motorcycle} \sqsubseteq \neg\textit{Bike};$$
$$\exists\textit{rent}.\textit{Scooter} \sqsubseteq \exists\textit{facilities}.(\textit{Parking} \sqcap \exists\textit{payment} \sqcap \forall\textit{payment}.\textit{Free}).$$

Suppose that we have the variables A_3, A_4, and A_5 of the CP-net of Fig. 1 with the domains $Dom(A_3) = \{\alpha_3, \neg\alpha_3\}$, $Dom(A_4) = \{\alpha_4, \neg\alpha_4\}$, and $Dom(A_5) = \{\alpha_5, \neg\alpha_5\}$, respectively, where:

$$\alpha_3 = \exists\textit{rent}.\textit{Scooter};$$
$$\alpha_4 = \exists\textit{facilities}.\textit{Parking};$$
$$\alpha_5 = \exists\textit{rent}.\textit{Bike}.$$

It is then not difficult to verify that $\alpha_3 \sqcap \alpha_5 \sqsubseteq_{\mathcal{T}} \bot$ and $\alpha_3 \sqsubseteq_{\mathcal{T}} \alpha_4$. Hence, A_3 and A_5 constrain each other, as well as A_3 and A_4. ■

Following [19], to compute the outcomes of a CP-net N, we can transform N into a set of constraints represented in clausal form. For each conditional preference $\Phi = (\tilde{\alpha} \mid \gamma)$ in $\mathcal{CPT}_N$, we write the following clause:

$$\gamma \to \tilde{\alpha}. \tag{2}$$

In a constrained CP-net, if we had propositional *true*/*false* variables, an outcome would be a model, i.e., a *true*/*false* assignment that satisfies all the constraints and some of the clauses built, starting from the preferences represented in $\mathcal{CPT}_N$. In ontological CP-nets, we also represent an outcome as a model satisfying a preference. Without loss of generality, we write an *interpretation* $\mathcal{I}$ as a conjunction of concept names α_i and negated concept names $\neg\alpha_i$, one such literal for each variable with the domain $\{\alpha_i, \neg\alpha_i\}$ in the CP-net. We say $\mathcal{I}$ *satisfies* a concept α under $\mathcal{T}$, denoted $\mathcal{I} \models_{\mathcal{T}} \alpha$, iff $\mathcal{T} \models \mathcal{I} \sqsubseteq \alpha$. We say α is *satisfiable* under $\mathcal{T}$ iff an interpretation $\mathcal{I}$ exists such that $\mathcal{I} \models_{\mathcal{T}} \alpha$. Finally, we use $\mathcal{I} \models \alpha \sqsubseteq \beta$ to say that $\mathcal{I} \sqsubseteq \neg\alpha \sqcup \beta$, and $\mathcal{I} \models \mathcal{T}$ iff $\mathcal{I} \models \alpha \sqsubseteq \beta$ for each axiom $\alpha \sqsubseteq \beta \in \mathcal{T}$.

Given two DL formulas α and γ, we call $\gamma \to \alpha$ a *DL clause*. Here, we use "$\to$" with the usual standard semantics.[1] An outcome $\mathcal{I}$ satisfies the conditional preference Φ under $\mathcal{T}$, denoted $\mathcal{I} \models_{\mathcal{T}} \Phi$, iff $\mathcal{I} \models_{\mathcal{T}} \gamma \to \tilde{\alpha}$. Using a notation similar to the one proposed in [19], we call *DL-opt(N)* the set of DL clauses corresponding to all the preferences in $\mathcal{CPT}_N$.

Definition 2 (feasible outcome and dominance). Given an ontological CP-net $(N, \mathcal{T})$, an outcome $\mathcal{I}$ is *feasible* iff $\mathcal{I} \models \mathcal{T}$. A feasible outcome $\mathcal{I}$ is *undominated* iff no feasible outcome $\mathcal{I}'$ exists such that $\mathcal{I}' \succ \mathcal{I}$.

[1] Hence, we have the equivalence $\gamma \to \alpha \equiv \neg\gamma \sqcup \alpha$.

3.1 Propositional Compilation of DL Formulas

Given a set of satisfiable DL formulas $\mathcal{F} = \{\phi_i \mid i \in \{1, \ldots, n\}\}$, some of them may *constrain* others, because of their logical relationships. For example, we may have $\phi_i \sqcap \neg\phi_j \sqsubseteq \phi_k$ or $\phi_i \sqcap \phi_j \sqsubseteq \bot$. By the equivalence $\alpha \sqsubseteq \beta \equiv \top \sqsubseteq \neg\alpha \sqcup \beta$, we can always represent each constraint in its logically equivalent clausal form. The previous constraints are then equivalent to $\top \sqsubseteq \neg\phi_i \sqcup \phi_j \sqcup \phi_k$ and $\top \sqsubseteq \neg\phi_i \sqcup \neg\phi_j$, respectively. In the following, we represent a clause ψ either as a logical disjunctive formula $\psi = \tilde{\phi_1} \sqcup \cdots \sqcup \tilde{\phi_n}$ or as a set of formulas $\psi = \{\tilde{\phi_1}, \ldots, \tilde{\phi_n}\}$. Moreover, we often write $\tilde{\phi_1} \sqcup \cdots \sqcup \tilde{\phi_n}$ to denote $\top \sqsubseteq \tilde{\phi_1} \sqcup \cdots \sqcup \tilde{\phi_n}$.

A DL ontology can be seen as a set of logical constraints that reduces the set of models for a formula. Given a set of DL formulas $\mathcal{F}$, in the following, we show how to compute a compact representation of an ontology $\mathcal{T}$ as a set of clauses whose variables have a one-to-one mapping to the formulas in $\mathcal{F}$.

Definition 3 (ontological constraint). Given an ontology $\mathcal{T}$ and a set of formulas $\mathcal{F} = \{\phi_i \mid i \in \{1, \ldots, n\}\}$ satisfiable w.r.t. $\mathcal{T}$, we say that $\mathcal{F}$ is *minimally constrained* w.r.t. $\mathcal{T}$ iff

1. there exists a formula $\tilde{\phi_1} \sqcup \ldots \sqcup \tilde{\phi_n}$ such that $\mathcal{T} \models \top \sqsubseteq \tilde{\phi_1} \sqcup \ldots \sqcup \tilde{\phi_n}$;
2. there is no proper subset $\mathcal{E} \subset \mathcal{F}$ such that the previous condition holds.

The formula $\top \sqsubseteq \tilde{\phi_1} \sqcup \ldots \sqcup \tilde{\phi_n}$ is called an *ontological constraint*.

An ontological constraint is an explicit representation of the constraints existing among a set of formulas, due to the information encoded in the ontology $\mathcal{T}$.

Definition 4 (ontological closure). Given an ontology $\mathcal{T}$ and a set of formulas $\mathcal{F} = \{\phi_i \mid i \in \{1, \ldots, n\}\}$ satisfiable w.r.t. $\mathcal{T}$, we call *ontological closure* of $\mathcal{F}$, denoted $\mathcal{OCL}(\mathcal{F}, \mathcal{T})$, the set of ontological constraints built, if any, for each set in $2^{\mathcal{F}}$.

The ontological constraint is an explicit representation of all the logical constraints considering also an underlying ontology. If we are interested only in the relationships between predefined formulas (due to $\mathcal{T}$), then the corresponding ontological closure is a compact and complete representation.

Example 3 (Hotel cont'd). Given the set $\mathcal{F} = \{\alpha_3, \alpha_4, \alpha_5\}$, due to the axioms in the ontology, we have the two minimally constrained sets $\mathcal{F}' = \{\alpha_3, \alpha_5\}$ and $\mathcal{F}'' = \{\alpha_3, \alpha_4\}$ and the two corresponding ontological constraints $\neg\alpha_3 \sqcup \neg\alpha_5$ (indeed $\alpha_3 \sqcap \alpha_5 \sqsubseteq_{\mathcal{T}} \bot$) and $\neg\alpha_3 \sqcup \alpha_4$ (indeed $\alpha_3 \sqsubseteq_{\mathcal{T}} \alpha_4$). The corresponding ontological closure is then $\mathcal{OCL}(\mathcal{F}, \mathcal{T}) = \{\neg\alpha_3 \sqcup \neg\alpha_5, \neg\alpha_3 \sqcup \alpha_4\}$. ■

Proposition 1. *Given a set $\mathcal{F} = \{\phi_i \mid i \in \{1, \ldots, n\}\}$ of satisfiable formulas, if $\mathcal{T} \models \bigsqcap \tilde{\phi_i} \sqsubseteq \bot$, then $\mathcal{OCL}(\mathcal{F}, \mathcal{T}) \models \bigsqcap \tilde{\phi_i} \sqsubseteq \bot$.*

We say that the set $\tilde{\mathcal{F}} = \{\tilde{\phi_i} \mid i \in \{1, \ldots, n\}\}$ is a *feasible* assignment for $\mathcal{F}$ iff

$$\mathcal{OCL}(\mathcal{F}, \mathcal{T}) \not\models \bigsqcap_i \tilde{\phi_i} \sqsubseteq \bot.$$

Note that by Proposition 1, we have that if $\tilde{\mathcal{F}}$ is a feasible assignment for $\mathcal{F}$, then we have $\mathcal{T} \not\models \bigsqcap_i \tilde{\phi_i} \sqsubseteq \bot$, i.e., $\bigsqcap_i \tilde{\phi_i}$ is satisfiable w.r.t. $\mathcal{T}$.

Proposition 2. *For each set of satisfiable formulas $\mathcal{F}$, there always exists a feasible assignment.*

We are interested in feasible assignments since, as we will show in the following, they represent feasible outcomes for an ontological CP-net.

3.2 Computing Optimal Outcomes

The main task that we want to solve with our framework is finding an undominated feasible outcome. In this section, we show how to compute it, given an ontological CP-net. The approach mainly relies on the HARD-PARETO algorithm of [19] (see Algorithm 1).

If we have an ontological CP-net $(N, \mathcal{T})$, the variable values (formulas) in a set $\mathcal{F}$ may constrain each other, and the corresponding constraints are encoded in $\mathcal{OCL}(\mathcal{F}, \mathcal{T})$. The ontological closure of a set of formulas explicitly represents all the logical constraints among them with respect to an underlying ontology. The computation of all feasible Pareto optimal solutions for an ontological CP-net goes through the Boolean encoding of both the ontology $\mathcal{T}$ and of the clauses corresponding to the preferences represented in $\mathcal{CPT}_N$ for each variable $A_j \in \mathcal{V}$. To use HARD-PARETO, we need a few pre-processing steps. Given the ontological CP-net $(N, \mathcal{T})$:

1. for each $A_j \in \mathcal{V}$ with $Dom(A_j) = \{\alpha_j, \neg\alpha_j\}$, choose a fresh concept name V_j;
2. define the ontology $\mathcal{T}' = \mathcal{T} \cup \{V_j \equiv \alpha_j \mid j \in \{1, \ldots, |\mathcal{V}|\}\}$;
3. define the ontological CP-net $(N', \mathcal{T}')$, where N' is the same CP-net as N but for the domain of its variables. In particular, in N', we have $Dom(A_j) = \{V_j, \neg V_j\}$;
4. define $\mathcal{F} = \{V_j \mid j \in \{1, \ldots, |\mathcal{V}|\}\}$, where the V_j's are the concept names introduced in step 1;
5. compute $\mathcal{OCL}(\mathcal{F}, \mathcal{T}')$;
6. introduce a Boolean variable v_j for each $V_j \in \mathcal{F}$;
7. transform $\mathcal{OCL}(\mathcal{F}, \mathcal{T}')$ into the corresponding set of Boolean clauses $\mathcal{C}$ by replacing V_j with the corresponding binary variable v_j;
8. transform *DL-opt(N')* into the set of Boolean clauses *opt(N')* by replacing $V_j \in Dom(A_j)$ with the corresponding variable v_j.

Note that $\mathcal{T}$ is logically equivalent to $\mathcal{T}'$. Indeed, we just introduced equivalence axioms to define new concept names V_j used as synonyms of complex formulas α_j. The same holds for $(N, \mathcal{T})$ and $(N', \mathcal{T}')$, since we just rewrite formulas in $Dom(A_j)$ with an equivalent concept name.

Example 4 (Hotel cont'd). With respect to the CP-net in Fig. 1, if we consider

$$\alpha_1 = \exists \mathit{location}.\mathit{OnTheSea},$$
$$\alpha_2 = \exists \mathit{location}.\mathit{CityCenter},$$

then we obtain:

- $\mathcal{T}' = \mathcal{T} \cup \{V_1 \equiv \exists \mathit{location}.\mathit{OnTheSea}, \ldots, V_5 \equiv \exists \mathit{rent}.\mathit{Bike}\}$;
- $\mathcal{C} = \{\neg v_3 \vee \neg v_5, \neg v_3 \vee v_4\}$;
- *opt(N')* $= \{v_1, v_2, v_1 \wedge v_2 \rightarrow v_3, \ldots, v_2 \rightarrow v_4, \ldots, v_3 \rightarrow v_5, \ldots\}$. ∎

Once we have $\mathcal{C}$ and *opt(N')*, we can compute the optimal outcome of $(N, \mathcal{T})$ by using the slightly modified version of HARD-PARETO represented in Algorithm 1. The function $sol(\cdot)$ used in Algorithm 1 computes all the solutions for the Boolean constraint satisfaction problem represented by $\mathcal{C}$, *opt(N')* and $\mathcal{C} \cup$ *opt(N')*. Differently from the original HARD-PARETO, by Proposition 2, we know that $\mathcal{C}$ is always consistent, and so we do not need to check its consistency at the beginning of the algorithm. Moreover, note that the algorithm works with propositional variables although we are computing Pareto optimal solutions for an ontological CP-net. This means that the dominance test in line 11 can be computed using well-known techniques for Boolean problems.

Input: *opt(N')* and $\mathcal{C}$

1 $S_{opt} \leftarrow sol(\mathcal{C} \cup opt(N'))$;
2 **if** $S_{opt} = sol(\mathcal{C})$ **then**
3   **return** S_{opt};
4 **end**
5 **if** $sol(opt(N')) \neq \emptyset$ *and* $S_{opt} = sol(opt(N'))$ **then**
6   **return** S_{opt};
7 **end**
8 $S \leftarrow sol(\mathcal{C}) - S_{opt}$;
9 **repeat**
10   choose $o \in S$;
11   **if** $\forall o' \in sol(\mathcal{C}) - o, o' \not\succ o$ **then**
12         $S_{opt} \leftarrow S_{opt} \cup \{o\}$;
13   **end**
14     $S \leftarrow S - \{o\}$;
15 **until** $S = \emptyset$;
16 **return** S_{opt}.

Algorithm 1: Algorithm HARD-PARETO adapted to ontological CP-nets.

The outcomes returned by Algorithm 1 in S_{opt} are *true*/*false* assignments to the Boolean variables v_j. To compute undominated outcomes for the original ontological CP-net $(N, \mathcal{T})$, we need to revert to a DL setting. Hence, we build the set *DL*-S_{opt}, where for each outcome $o_i \in S_{opt}$, we add to *DL*-S_{opt} the following formula:

$$\mathcal{I}_i = \bigsqcap\{V_j \mid v_j = \mathit{true} \text{ in } o_i\} \sqcap \bigsqcap\{\neg V_j \mid v_j = \mathit{false} \text{ in } o_i\}.$$

Theorem 1. *Given an ontological CP-net* $(N, \mathcal{T})$*, the formulas* $\mathcal{I}_i \in DL\text{-}S_{opt}$ *are undominated outcomes for* $(N, \mathcal{T})$*.*

4 Computational Complexity

We now explore the complexity of the main computational problems in ontological CP-nets for underlying ontological languages with typical complexity of deciding knowledge base satisfiability, namely, tractability and completeness for EXP and NEXP. We also provide some special tractable cases of dominance testing in ontological CP-nets.

4.1 General Results

For tractable ontology languages (i.e., those for which deciding knowledge base satsfiability is tractable), the complexity of ontological CP-nets is dominated by the complexity of CP-nets. That is, deciding (a) consistency, (b) whether a given outcome is undominated, and (c) dominance of two given outcomes are all PSPACE-complete. Here, the lower bounds follow from the fact that ontological CP-nets generalize CP-nets, for which these problems are all PSPACE-complete [11]. As for the upper bounds, compared to standard CP-nets, these problems additionally involve knowledge base satisfiability checks, which can all be done in polynomial time and thus also in polynomial space. Note that in (a) (resp., (b)), one has to go through all outcomes o' and check that it is not the case that $o \succ o'$ (resp., $o' \succ o$), which can each and thus overall be done in polynomial space. Note also that the same complexity results hold for ontology languages with PSPACE-complete knowledge base satisfiability checks and that even computing the set of all undominated outcomes (generalizing (b)) is PSPACE-complete under the condition that there are only polynomially many of them.

Theorem 2. *Given an ontological CP-net $(N, \mathcal{T})$ over a tractable ontology language,*

(a) deciding whether $(N, \mathcal{T})$ is consistent,
(b) deciding whether a given outcome o is undominated,
(c) deciding whether $o \prec o'$ for two given outcomes o and o'

are all PSPACE-complete.

In particular, if the ontological CP-net is defined over a DL of the *DL-Lite* family [7] (which all allow for deciding knowledge base satisfiability in polynomial time, such as *DL-Lite*$_{\mathcal{R}}$, which stands behind the important OWL 2 QL profile [17]), deciding (a) consistency, (b) whether a given outcome is undominated, and (c) dominance of two given outcomes are all PSPACE-complete.

Corollary 1. *Given an ontological CP-net $(N, \mathcal{T})$ over a DL from the DL-Lite family,*

(a) deciding whether $(N, \mathcal{T})$ is consistent,
(b) deciding whether a given outcome o is undominated,
(c) deciding whether $o \prec o'$ for two given outcomes o and o'

are all PSPACE-complete.

For EXP (resp., NEXP) complete ontology languages (i.e., those for which knowledge base satisfiability is complete for EXP (resp., NEXP)), the complexity of ontological CP-nets is dominated by the complexity of the ontology languages. That is, deciding (a) inconsistency, (b) whether a given outcome is dominated, and (c) dominance of two given outcomes are all complete for EXP (resp., NEXP). Here, the lower bounds follow from the fact that all three problems in ontological CP-nets can be used to decide knowledge base satisfiability in the underlying ontology language. As for the upper bounds, in (a) and (b), we have to go through all outcomes, which is in EXP (resp., NEXP). Then, we have to perform knowledge base satisfiability checks, which are also in EXP (resp., NEXP), and dominance checks in standard CP-nets, which are in PSPACE, and thus also in EXP (resp., NEXP). Overall, (a) to (c) are thus in EXP (resp., NEXP). Note that computing the set of all undominated outcomes (generalizing (b)) is also EXP-complete for EXP-complete ontology languages.

Theorem 3. *Given an ontological CP-net* $(N, \mathcal{T})$ *over an EXP (resp., NEXP) complete ontology language,*

(a) deciding whether $(N, \mathcal{T})$ *is inconsistent,*
(b) deciding whether a given outcome o *is dominated,*
(c) deciding whether $o \prec o'$ *for two given outcomes* o *and* o'

are all complete for EXP (resp., NEXP).

In particular, if the ontological CP-net is defined over the expressive DL $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) [13] (which stands behind OWL Lite (resp., OWL DL) [16, 12], and allows for deciding knowledge base satisfiability in EXP [13, 21] (resp., NEXP, for both unary and binary number encoding; see [18, 21] and the NEXP-hardness proof for $\mathcal{ALCQIO}$ in [21], which implies the NEXP-hardness of $\mathcal{SHOIN}(\mathbf{D})$)), deciding (a) inconsistency, (b) whether a given outcome is dominated, and (c) dominance of two given outcomes are all complete for EXP (resp., NEXP).

Corollary 2. *Given an ontological CP-net* $(N, \mathcal{T})$ *over the DL* $\mathcal{SHIF}(\mathbf{D})$ *(resp.,* $\mathcal{SHOIN}(\mathbf{D})$*),*

(a) deciding whether $(N, \mathcal{T})$ *is inconsistent,*
(b) deciding whether a given outcome o *is dominated,*
(c) deciding whether $o \prec o'$ *for two given outcomes* o *and* o'

are all complete for EXP (resp., NEXP).

4.2 Tractability Results

If the ontological CP-net is a polytree and defined over a tractable ontology language, deciding dominance of two outcomes can be done in polynomial time, which follows from the fact that for standard polytree CP-nets, dominance can be decided in polynomial time [3]. Note that polytree ontological CP-nets are always consistent.

Theorem 4. *Given an ontological CP-net* $(N, \mathcal{T})$ *over a tractable ontology language, where* N *is a polytree, deciding whether* $o \prec o'$ *for two given outcomes* o *and* o' *can be done in polynomial time.*

In particular, if the ontological CP-net is a polytree and defined over a DL of the *DL-Lite* family, deciding dominance of two outcomes can be done in polynomial time.

Corollary 3. *Given an ontological CP-net* $(N, \mathcal{T})$ *over a DL from the DL-Lite family, where* N *is a polytree, deciding whether* $o \prec o'$ *for two given outcomes* o *and* o' *can be done in polynomial time.*

5 Related Work

Constrained CP-nets were originally proposed in [4], along with the algorithm SEARCH-CP, which uses branch and bound to compute undominated outcomes. The algorithm has an anytime behavior: it can be stopped at any time, and the set of computed solutions are a subset of the set containing all the undominated outcomes. This means that in case one is interested in any undominated outcome, one can use the first one returned by SEARCH-CP. In [19], HARD-PARETO is presented. The most notable difference is that HARD-PARETO does not rely on topological information like SEARCH-CP, but it exploits only the CP-statements, thus allowing to work also with cyclic CP-nets. Differently from the previous two papers, in our work, we allow the variable domains to contain DL formulas constrained via ontological axioms.

There are a very few papers describing how to combine Semantic Web technologies with preference representation and reasoning using CP-nets. To our knowledge, the most notable work is [2]. Here, in an information retrieval context, Wordnet is used to add a semantics to CP-net variables. Another interesting approach to mixing qualitative preferences with a Semantic Web technology is presented in [20], where the authors describe an extension of SPARQL, which can encode user preferences in the query.

A combination of conditional preferences (very different from CP-nets) with DL reasoning for ranking objects is introduced in [15]. A ranking function is described that exploits conditional preferences to perform a semantic personalized search and ranking over a set of resources annotated via an ontological description.

6 Summary and Outlook

In classical decision theory and analysis, the preferences of decision makers are modeled by utility functions. Unfortunately, the effort needed to obtain a good utility function requires a significant involvement of the user [10]. This is one of the main reasons behind the success obtained by CP-nets since they were originally proposed [5]: they are compact, easily understandable and well-suited for combinatorial domains, such as multi-attribute ones. In this paper, we have described how to reason with CP-nets whose variable values are DL formulas that refer to a common ontology. The proposed framework is very useful in many semantic retrieval scenarios such as semantic search.

After the introduction of CP-nets, other related formalisms have been proposed such as TCP-nets (Trade-off CP-nets) [6] or CP-theories [22]. TCP-nets extend CP-nets by allowing to express also relative important statements between variables. With TCP-nets, the user is allowed to express her preferences over compromises that sometimes may be required. CP-theories generalize (T)CP-nets allowing conditional preference statements on the values of a variable, along with a set of variables that are allowed to vary when interpreting the preference statement. In future work, we plan to enrich these frameworks by introducing ontological descriptions and reasoning, thus allowing the development of more powerful semantic-enabled preference-based retrieval systems.

Acknowledgments. This work was supported by the UK EPSRC grant EP/J008346/1 "PrOQAW: Probabilistic Ontological Query Answering on the Web", the ERC (FP7/ 2007-2013) grant 246858 ("DIADEM"), and by a Yahoo! Research Fellowship.

References

1. F. Baader, D. Calvanese, D. Mc Guinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2002.
2. F. Boubekeur, M. Boughanem, and L. Tamine-Lechani. Semantic information retrieval based on CP-nets. In *Proc. FUZZ-IEEE-2007*, pp. 1–7. IEEE Computer Society, 2007.
3. 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. *J. Artif. Intell. Res.*, 21:135–191, 2004.
4. C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole. Preference-based constrained optimization with CP-nets. *Computat. Intell.*, 20(2):137–157, 2004.
5. C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *Proc. UAI-1999*, pp. 71–80. Morgan Kaufmann, 1999.
6. R. I. Brafman, C. Domshlak, and S. E. Shimony. On graphical modeling of preference and importance. *J. Artif. Intell. Res.*, 25(1):389–424, 2006.
7. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.
8. T. Di Noia and T. Lukasiewicz. Introducing ontological CP-nets. In *Proc. URSW-2013*, volume 900 of *CEUR Workshop Proceedings*, pp. 90–93. CEUR-WS.org, 2012.
9. C. Domshlak, E. Hüllermeier, S. Kaci, and H. Prade. Preferences in AI: An overview. *Artif. Intell.*, 175(7/8):1037–1052, 2011.
10. S. French. *Decision Theory: An Introduction to the Mathematics of Rationality*. Ellis Horwood Series in Mathematics and its Applications. Prentice Hall, 1988.
11. J. Goldsmith, J. Lang, M. Truszczynski, and N. Wilson. The computational complexity of dominance and consistency in CP-nets. *J. Artif. Intell. Res.*, 33:403–432, 2008.
12. P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, editors. *OWL 2 Web Ontology Language Primer (Second Edition)*. W3C Recommendation, 11 December 2012. Available at `http://www.w3.org/TR/owl2-primer/`.
13. I. Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *Proc. ISWC-2003*, volume 2870 of *LNCS*, pp. 17–29. Springer, 2003.
14. R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge University Press, 1993.
15. T. Lukasiewicz and J. Schellhase. Variable-strength conditional preferences for ranking objects in ontologies. *J. Web Sem.*, 5(3):180–194, 2007.
16. D. L. McGuinness and F. van Harmelen, editors. OWL Web Ontology Language Overview. W3C Recommendation, 10 February 2004. Available at `http://www.w3.org/TR/2004/REC-owl-features-20040210/`.
17. B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz. OWL 2 Web Ontology Language Profiles (Second Edition). W3C Recommendation, 11 December 2012. `http://www.w3.org/TR/owl2-profiles/`.
18. I. Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *Journal of Logic, Language and Information*, 14(3):369–395, 2005.
19. S. D. Prestwich, F. Rossi, K. Brent Venable, and T. Walsh. Constraint-based preferential optimization. In *Proc. AAAI/IAAI-2005*, pp. 461–466. AAAI Press / MIT Press, 2005.
20. W. Siberski, J. Z. Pan, and U. Thaden. Querying the Semantic Web with preferences. In *Proc. ISWC-2006*, volume 4273 of *LNCS*, pp. 612–624. Springer, 2006.
21. S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. Doctoral Dissertation, RWTH Aachen, Germany, 2001.
22. N. Wilson. Extending CP-nets with stronger conditional preference statements. In *Proc. AAAI-2004*, pp. 735–741. AAAI Press, 2004.