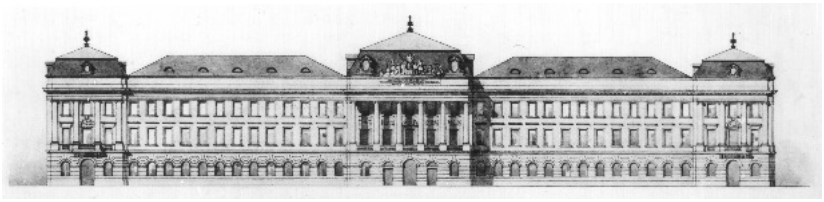
**I N F S Y S**
**R** E S E A R C H
**R** E P O R T



INSTITUT FÜR INFORMATIONSSYSTEME
ARBEITSBEREICH WISSENSBASIERTE SYSTEME

# VARIABLE-STRENGTH CONDITIONAL PREFERENCES FOR RANKING OBJECTS IN ONTOLOGIES

THOMAS LUKASIEWICZ JÖRG SCHELLHASE

INFSYS RESEARCH REPORT 1843-07-06
APRIL 2007

Institut für Informationssysteme
AB Wissensbasierte Systeme
Technische Universität Wien
Favoritenstraße 9-11
A-1040 Wien, Austria
Tel: +43-1-58801-18405
Fax: +43-1-58801-18493
sek@kr.tuwien.ac.at
www.kr.tuwien.ac.at

TU VIENNA

INFSYS RESEARCH REPORT

INFSYS RESEARCH REPORT 1843-07-06, APRIL 2007

# VARIABLE-STRENGTH CONDITIONAL PREFERENCES FOR RANKING OBJECTS IN ONTOLOGIES

APRIL 22, 2007

Thomas Lukasiewicz [1] Jörg Schellhase [2]

**Abstract.** We introduce conditional preference bases as a means for ranking objects in ontologies. Conditional preference bases consist of a description logic knowledge base and a finite set of conditional preferences, which are statements of the form "generally, in the context $\phi$, property $\alpha$ is preferred over property $\neg\alpha$ with strength $s$". They are inspired by variable-strength defaults in conditional knowledge bases. We define the notion of consistency for conditional preference bases, and we show how consistent conditional preference bases can be used for ranking objects in ontologies, where every object represents essentially a set of individuals that are sharing the same ranking-relevant properties. More concretely, we define two object rankings, denoted $\kappa^{sum}$ and $\kappa^{lex}$, which evaluate the strengths of conditional preferences in an additive and a lexicographic way, respectively. Furthermore, we provide algorithms for the main computational tasks for ranking objects under conditional preference bases, we analyze the complexity of these tasks, and we delineate a tractable special case. To give evidence of the usefulness of this approach in practice, we describe two applications in the areas of product and literature search, where it allows especially for a flexible user-defined ranking of the query results reflecting personal preferences.

[1]Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, Via Ariosto 25, I-00185 Rome, Italy; e-mail: lukasiewicz@dis.uniroma1.it. Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, A-1040 Vienna, Austria; e-mail: lukasiewicz@kr.tuwien.ac.at.

[2]Fachgebiet Wirtschaftsinformatik, Universität Kassel, Nora-Platiel-Straße 4, D-34127 Kassel, Germany; e-mail: schellhase@wirtschaft.uni-kassel.de.

**Acknowledgements**: This work has been partially supported by the German Research Foundation (DFG) under the Heisenberg Programme.

Copyright © 2007 by the authors

# Contents

# 1 Introduction

In their seminal works [44, 43], Poole and Smyth deal with the problem of matching instances against models of instances, which are both described at different levels of abstraction and at different levels of detail, using qualitative probability theory. Informally, such problems are as follows. Given an instance $I$ and a model of instances $M$, compute the qualitative probability that the instance $I$ is matching the model $M$ (that is, of $I$ given $M$). For example, in a geological exploration domain, we may want to determine whether there might be gold in an area. In this case, an instance $I$ may be given by the description of an area, while a model $M$ may be given by a description of areas where gold can be found, and the qualitative probability that $I$ is matching $M$ describes the likelihood that gold may be found in $I$.

In this paper, we continue this line of research. A serious drawback of the above works [44, 43] on matching instances against models of instances is that they only allow for expressing simple preferences of the form "property $\alpha$ is preferred over property $\neg\alpha$ with strength $s$" in models of instances. In particular, they do not allow for conditional preferences such as "generally, in the context $\phi$, property $\alpha$ is preferred over property $\neg\alpha$ with strength $s$". In this paper, we aim at filling this gap. We present a formalism for ranking objects in description logics that allows for expressing such conditional preferences in models of instances. In a companion paper [36], we present a generalization of it for matchmaking in description logics.

Like Poole and Smyth's work [44, 43], the ranking formalism in this paper is also based on qualitative probabilities. Differently from Poole and Smyth's work [44, 43], however, it requires a technically more involved way of computing qualitative probabilities, since our language for encoding models of instances is more expressive. We especially have to suitably handle *variable-strength conditional preferences*, which are the above statements of the form "generally, in the context $\phi$, property $\alpha$ is preferred over property $\neg\alpha$ with strength $s$" (also called *variable-strength conditional desires* [51]). They bear close similarity to *variable-strength defaults* of the form "generally, if $\phi$ then $\alpha$ with strength $s$" in conditional knowledge bases [24].

In this paper, we define a formal semantics for variable-strength conditional preferences, which is based on conditional knowledge bases (see Sections 3 and 10). We focus on the problem of ranking objects against a description of objects. Since we are especially interested in the Semantic Web as the main application context, we assume that objects and descriptions of objects are expressed in the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind the web ontology languages OWL Lite and OWL DL, respectively [26]. We assume that every object is a finite set of ranking-relevant properties of individuals and so essentially a set of individuals sharing the same ranking-relevant properties.

The Semantic Web [5, 17] aims at an extension of the current World Wide Web by standards and technologies that help machines to understand the information on the Web so that they can support richer discovery, data integration, navigation, and automation of tasks. The main ideas behind it are to add a machine-readable meaning to Web pages, to use ontologies for a precise definition of shared terms in Web resources, to make use of knowledge representation and reasoning technology for automated reasoning from Web resources, and to apply cooperative agent technology for processing the information of the Web. The Semantic Web consists of several hierarchical layers, where the Ontology layer, in form of the OWL Web Ontology Language [54, 27] (recommended by the W3C), is currently the highest layer of sufficient maturity. OWL consists of three increasingly expressive sublanguages, namely OWL Lite, OWL DL, and OWL Full. OWL Lite and OWL DL are essentially expressive description logics with an RDF syntax [27]. Ontology entailment in OWL Lite (resp., OWL DL) reduces to knowledge base (un)satisfiability in the description logic $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) [26].

The main contributions of this paper can be summarized as follows:

- We introduce conditional preference bases, which consist of a description logic knowledge base in

$\mathcal{SHIF}(\mathbf{D})$ or $\mathcal{SHOIN}(\mathbf{D})$, and a finite set of (variable-strength) conditional preferences. They are inspired by variable-strength defaults in conditional knowledge bases [24]. We define the notion of consistency for conditional preference bases, and show how consistent conditional preference bases can be used for ranking objects in ontologies, where every object is roughly a set of individuals sharing the same ranking-relevant properties. More concretely, we define two object rankings, denoted $\kappa^{sum}$ and $\kappa^{lex}$, which evaluate the strengths of conditional preferences in an additive and a lexicographic way, respectively.

- We provide algorithms for the main computational tasks related to conditional preference bases, namely, (i) for deciding whether a conditional preference base is consistent, (ii) for computing the $z$-partition of a consistent conditional preference base, and (iii) for computing the rankings $\kappa^{sum}$ and $\kappa^{lex}$ on a set of objects relative to a consistent conditional preference base. Each of these algorithms is based on a reduction to a polynomial number of tests whether a description logic knowledge base in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable.

- We analyze the complexity of the main computational tasks related to conditional preference bases. For conditional preference bases over $\mathcal{SHIF}(\mathbf{D})$, all tasks are complete for EXP (resp., FEXP), and thus have the same complexity as deciding knowledge base satisfiability in $\mathcal{SHIF}(\mathbf{D})$. For conditional preference bases over $\mathcal{SHOIN}(\mathbf{D})$, deciding consistency is complete for NEXP, and thus has the same complexity as deciding knowledge base satisfiability in $\mathcal{SHOIN}(\mathbf{D})$, while all the other tasks are in $\mathrm{FP}^{\mathrm{NEXP}}$, and thus can be done in the same time as the main reasoning tasks in description logic programs over $\mathcal{SHOIN}(\mathbf{D})$ [16].

- Taking inspiration from the recent description logic *DL-Lite*, which allows for polynomial-time description logic reasoning [9], we describe a special case of conditional preference bases (with concepts in *DL-Lite*) where the main computational tasks can all be done in polynomial time.

- We describe two applications of this approach in product and literature search. In the former, it allows for a flexible user-defined ranking of the query results, which reflects personal preferences. In the latter, it allows for both expressing sophisticated search strategies and a flexible user-defined ranking of the query results, which reflects personal preferences and quality measures. More generally, query languages of current search engines are very restricted in their expressive power. There are scientific search engines on the web, however, that have valuable metadata about publications, authors, organizations, and scientific events. We show that conditional preference bases allow for a more powerful query language, which can exploit this metadata better than the current approaches do.

The rest of this paper is organized as follows. In Section 2, we review the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$. In Section 3, we review conditional knowledge bases. Section 4 introduces conditional preference bases and the notions of consistency and $z$-entailment for conditional preference bases. In Section 5, we present the two object rankings $\kappa^{sum}$ and $\kappa^{lex}$ relative to a consistent conditional preference base. In Sections 6–8, we provide algorithms for the main computational tasks related to conditional preference bases, we analyze the complexity of these tasks, and we delineate tractable special cases. Section 9 describes a sample application in literature search, and Section 10 discusses related work. In Section 11, we summarize the main results and give an outlook on future research. Note that detailed proofs of all results in this paper are given in Appendices A–C.

# 2 The Description Logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$

In this section, we review the description logics $\mathcal{SHIF}(\mathbf{D})$ and $\mathcal{SHOIN}(\mathbf{D})$, which stand behind the web ontology languages OWL Lite and OWL DL, respectively. See especially [26] for further details and background. Intuitively, description logics model a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively. A description logic knowledge base encodes in particular subset relationships between classes of individuals, the membership of individuals to classes, and the membership of pairs of individuals to binary relations between classes.

## 2.1 Syntax

We first describe the syntax of $\mathcal{SHOIN}(\mathbf{D})$. We assume a set of *elementary datatypes* and a set of *data values*. A *datatype* is an elementary datatype or a set of data values (called *datatype oneOf*). A *datatype theory* $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a *datatype domain* $\Delta^{\mathbf{D}}$ and a mapping $\cdot^{\mathbf{D}}$ that assigns to each elementary datatype a subset of $\Delta^{\mathbf{D}}$ and to each data value an element of $\Delta^{\mathbf{D}}$. We extend $\cdot^{\mathbf{D}}$ to all datatypes by $\{v_1, \ldots\}^{\mathbf{D}} = \{v_1^{\mathbf{D}}, \ldots\}$. Let $\mathbf{A}$, $\mathbf{R}_A$, $\mathbf{R}_D$, and $\mathbf{I}$ be pairwise disjoint finite nonempty sets of *atomic concepts*, *abstract roles*, *datatype roles*, and *individuals*, respectively. We denote by $\mathbf{R}_A^-$ the set of inverses $R^-$ of all $R \in \mathbf{R}_A$.

A *role* is any element of $\mathbf{R}_A \cup \mathbf{R}_A^- \cup \mathbf{R}_D$. *Concepts* are inductively defined as follows. Every $\phi \in \mathbf{A}$ is a concept, and if $o_1, \ldots, o_n \in \mathbf{I}$, then $\{o_1, \ldots, o_n\}$ is a concept (called *oneOf*). If $\phi$, $\phi_1$, and $\phi_2$ are concepts and if $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, then also $\neg\phi$, $(\phi_1 \sqcap \phi_2)$, and $(\phi_1 \sqcup \phi_2)$ are concepts (called *negation*, *conjunction*, and *disjunction*, respectively), as well as $\exists R.\phi$, $\forall R.\phi$, $\geq nR$, and $\leq nR$ (called *exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geq 0$. If $D$ is a datatype and $U \in \mathbf{R}_D$, then $\exists U.D$, $\forall U.D$, $\geq nU$, and $\leq nU$ are concepts (called *datatype exists*, *value*, *atleast*, and *atmost restriction*, respectively) for an integer $n \geq 0$. We use $\top$ (resp., $\bot$) to abbreviate $\phi \sqcup \neg\phi$ (resp., $\phi \sqcap \neg\phi$), and eliminate parentheses as usual.

An *axiom* has one of the following forms: (1) $\phi \sqsubseteq \psi$ (called *concept inclusion axiom*), where $\phi$ and $\psi$ are concepts; (2) $R \sqsubseteq S$ (called *role inclusion axiom*), where either $R, S \in \mathbf{R}_A \cup \mathbf{R}_A^-$ or $R, S \in \mathbf{R}_D$; (3) $\mathrm{Trans}(R)$ (called *transitivity axiom*), where $R \in \mathbf{R}_A$; (4) $\phi(a)$ (called *concept membership axiom*), where $\phi$ is a concept and $a \in \mathbf{I}$; (5) $R(a, b)$ (resp., $U(a, v)$) (called *role membership axiom*), where $R \in \mathbf{R}_A$ (resp., $U \in \mathbf{R}_D$) and $a, b \in \mathbf{I}$ (resp., $a \in \mathbf{I}$ and $v$ is a data value); and (6) $a = b$ (resp., $a \neq b$) (*equality* (resp., *inequality*) *axiom*), where $a, b \in \mathbf{I}$. A *(description logic) knowledge base* $KB$ is a finite set of axioms. For decidability, number restrictions in $KB$ are restricted to *simple* abstract roles (see [28] for details).

The syntax of $\mathcal{SHIF}(\mathbf{D})$ is as the above syntax of $\mathcal{SHOIN}(\mathbf{D})$, but without the oneOf constructor and with the atleast and atmost constructors limited to 0 and 1.

**Example 2.1** *(Products)* An online store (such as *amazon.com*) may use a description logic knowledge base to classify and characterize its products. For example, assume the following relationships between products: (1) textbooks are books, (2) personal computers and laptops are mutually exclusive electronic products, (3) books and electronic products are mutually exclusive products, (4) objects on sale are products, (5) every product has at least one related product, (6) only products are related to each other, (7) *tb_ai* and *tb_lp* are textbooks, (8) which are related to each other, (9) *pc_ibm* and *pc_hp* are personal computers, (10) which are related to each other, and (11) *ibm* and *hp* are providers for *pc_ibm* resp. *pc_hp*. These relationships are expressed by the following description logic knowledge base $KB$:

(1) *Textbook* $\sqsubseteq$ *Book*;

(2) $PC \sqcup Laptop \sqsubseteq Electronics$; $PC \sqsubseteq \neg Laptop$;
(3) $Book \sqcup Electronics \sqsubseteq Product$; $Book \sqsubseteq \neg Electronics$;
(4) $Sale \sqsubseteq Product$;
(5) $Product \sqsubseteq \geq 1\, related$;
(6) $\geq 1\, related \sqcup \geq 1\, related^- \sqsubseteq Product$;
(7) $Textbook(tb\_ai)$; $Textbook(tb\_lp)$;
(8) $related(tb\_ai, tb\_lp)$;
(9) $PC(pc\_ibm)$; $PC(pc\_hp)$;
(10) $related(pc\_ibm, pc\_hp)$;
(11) $provides(ibm, pc\_ibm)$; $provides(hp, pc\_hp)$.

## 2.2 Semantics

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with respect to a datatype theory $\mathbf{D} = (\Delta^{\mathbf{D}}, \cdot^{\mathbf{D}})$ consists of a nonempty (*abstract*) *domain* $\Delta^{\mathcal{I}}$ disjoint from $\Delta^{\mathbf{D}}$, and a mapping $\cdot^{\mathcal{I}}$ that assigns to each atomic concept $\phi \in \mathbf{A}$ a subset of $\Delta^{\mathcal{I}}$, to each individual $o \in \mathbf{I}$ an element of $\Delta^{\mathcal{I}}$, to each abstract role $R \in \mathbf{R}_A$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to each datatype role $U \in \mathbf{R}_D$ a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathbf{D}}$. We extend $\cdot^{\mathcal{I}}$ to all concepts and roles as usual (where $\#S$ denotes the cardinality of a set $S$):

- $\{o_1, \ldots, o_n\}^{\mathcal{I}} = \{o_1^{\mathcal{I}}, \ldots, o_n^{\mathcal{I}}\}$; $(\neg\phi)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus \phi^{\mathcal{I}}$;
- $(\phi_1 \sqcap \phi_2)^{\mathcal{I}} = \phi_1^{\mathcal{I}} \cap \phi_2^{\mathcal{I}}$; $(\phi_1 \sqcup \phi_2)^{\mathcal{I}} = \phi_1^{\mathcal{I}} \cup \phi_2^{\mathcal{I}}$;
- $(\exists R.\phi)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\colon (x,y) \in R^{\mathcal{I}} \wedge y \in \phi^{\mathcal{I}}\}$;
- $(\forall R.\phi)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\colon (x,y) \in R^{\mathcal{I}} \rightarrow y \in \phi^{\mathcal{I}}\}$;
- $(\geq n R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\}) \geq n\}$;
- $(\leq n R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\}) \leq n\}$;
- $(\exists U.D)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\colon (x,y) \in U^{\mathcal{I}} \wedge y \in D^{\mathbf{D}}\}$;
- $(\forall U.D)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\colon (x,y) \in U^{\mathcal{I}} \rightarrow y \in D^{\mathbf{D}}\}$;
- $(\geq n U)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in U^{\mathcal{I}}\}) \geq n\}$;
- $(\leq n U)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \#(\{y \mid (x,y) \in U^{\mathcal{I}}\}) \leq n\}$.

The *satisfaction* of an axiom $F$ in an interpretation $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$, denoted $\mathcal{I} \models F$, is defined as follows: (1) $\mathcal{I} \models \phi \sqsubseteq \psi$ iff $\phi^{\mathcal{I}} \subseteq \psi^{\mathcal{I}}$; (2) $\mathcal{I} \models R \sqsubseteq S$ iff $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$; (3) $\mathcal{I} \models \mathrm{Trans}(R)$ iff $R^{\mathcal{I}}$ is transitive; (4) $\mathcal{I} \models \phi(a)$ iff $a^{\mathcal{I}} \in \phi^{\mathcal{I}}$; (5) $\mathcal{I} \models R(a,b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$; (6) $\mathcal{I} \models U(a,v)$ iff $(a^{\mathcal{I}}, v^{\mathbf{D}}) \in U^{\mathcal{I}}$; (7) $\mathcal{I} \models a = b$ iff $a^{\mathcal{I}} = b^{\mathcal{I}}$; and (8) $\mathcal{I} \models a \neq b$ iff $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. The interpretation $\mathcal{I}$ *satisfies* the axiom $F$, or $\mathcal{I}$ is a *model* of $F$, iff $\mathcal{I} \models F$. We say that $\mathcal{I}$ *satisfies* a knowledge base $KB$, or $\mathcal{I}$ is a *model* of $KB$, denoted $\mathcal{I} \models KB$, iff $\mathcal{I} \models F$ for all $F \in KB$. We say that $KB$ is *satisfiable* (resp., *unsatisfiable*) iff $KB$ has a (resp., no) model. An axiom $F$ is a *logical consequence* of $KB$, denoted $KB \models F$, iff each model of $KB$ satisfies $F$.

**Example 2.2** *(Products cont'd)* The description logic knowledge base $KB$ given in Example 3.1 is satisfiable, and some logical consequences of $KB$ are $Textbook \sqsubseteq \neg Electronics$, $PC \sqsubseteq Electronics$, $\neg Electronics(tb\_lp)$, and $Electronics(pc\_ibm)$.

# 3 Conditional Knowledge Bases

In this section, we review conditional knowledge bases and the notion of $\varepsilon$-consistency for conditional knowledge bases [1, 22]. Informally, a conditional knowledge base consists of a set of strict statements in classical logic of the form "$\psi \Leftarrow \phi$", which informally read as "if $\phi$ then $\psi$" and a set of defeasible rules (or defaults) of the form "$\psi \leftarrow \phi$", which informally read as "generally, if $\phi$ then $\psi$". The latter rules may have exceptions, which can be handled in different ways (see Section 10.1). Note that for ease of presentation, we here do not consider variable-strength defaults (which are of the form "$\psi \leftarrow_s \phi$" to express that "generally, if $\phi$ then $\psi$ with strength $s$"), but all the concepts and results revisited below are easily extended to conditional knowledge bases with variable-strength defaults [24].

## 3.1 Syntax

We first formally define the syntax of conditional knowledge bases. We assume a set of *basic events* $\Phi = \{p_1, \ldots, p_l\}$ with $l \geq 1$. We use $\bot$ and $\top$ to denote *false* and *true*, respectively. We define *events* by induction as follows. Every element of $\Phi \cup \{\bot, \top\}$ is an event. If $\phi$ and $\psi$ are events, then also $\neg\phi$ and $(\phi \wedge \psi)$. We use $(\phi \vee \psi)$ and $(\psi \Leftarrow \phi)$ to abbreviate the events $\neg(\neg\phi \wedge \neg\psi)$ and $\neg(\neg\psi \wedge \phi)$, respectively, and we adopt the usual conventions to eliminate parentheses. A *logical constraint* is an event of the form $\psi \Leftarrow \phi$. A *conditional rule* (or *default*) is an expression of the form $\psi \leftarrow \phi$, where $\phi$ and $\psi$ are events. A *conditional knowledge base* $KB = (L, D)$ consists of a finite set of logical constraints $L$ and a finite set of defaults $D$. The following example illustrates conditional knowledge bases.

**Example 3.1** *(Penguins)* The strict logical knowledge "all penguins are birds" and the default logical knowledge "generally, birds fly", "generally, penguins do not fly", and "generally, birds have wings" is encoded by the conditional knowledge base $KB = (\{\mathit{bird} \Leftarrow \mathit{penguin}\}, \{\mathit{fly} \leftarrow \mathit{bird}, \neg\mathit{fly} \leftarrow \mathit{penguin}, \mathit{wings} \leftarrow \mathit{bird}\})$.

## 3.2 Semantics

We next define the semantics of conditional knowledge bases in terms of admissible world rankings. A *world* $I$ associates with every basic event in $\Phi$ a binary truth value. We extend $I$ by induction to all events as usual. We denote by $\mathcal{I}_\Phi$ the set of all worlds for $\Phi$. A world $I$ *satisfies* an event $\phi$, or $I$ is a *model* of $\phi$, denoted $I \models \phi$, iff $I(\phi) = \mathbf{true}$. A world $I$ *satisfies* a default $\psi \leftarrow \phi$, or $I$ is a *model* of $\psi \leftarrow \phi$, denoted $I \models \psi \leftarrow \phi$, iff $I \models \psi \Leftarrow \phi$. We say that $I$ *verifies* $\psi \leftarrow \phi$ iff $I \models \phi \wedge \psi$. We say that $I$ *falsifies* $\psi \leftarrow \phi$ iff $I \models \phi \wedge \neg\psi$ (that is, $I \not\models \psi \leftarrow \phi$). We say that $I$ *satisfies* a set of events and defaults $K$, or $I$ is a *model* of $K$, denoted $I \models K$, iff $I$ satisfies every member of $K$. We say $K$ is *satisfiable* iff a model of $K$ exists. A set of defaults $D$ *tolerates* a default $d$ *under* a set of logical constraints $L$, or $d$ is *tolerated under* $L$ *by* $D$, iff $D \cup L$ has a model that verifies $d$. A set of defaults $D$ is *under* $L$ *in conflict* with a default $\psi \leftarrow \phi$ iff all models of $D \cup L \cup \{\phi\}$ satisfy $\neg\psi$. A *world ranking* $\kappa$ is a mapping $\kappa\colon \mathcal{I}_\Phi \rightarrow \{0, 1, \ldots\} \cup \{\infty\}$ such that $\kappa(I) = 0$ for at least one world $I$. It is extended to all events $\phi$ as follows. If $\phi$ is satisfiable, then $\kappa(\phi) = \min\{\kappa(I) \mid I \in \mathcal{I}_\Phi,\ I \models \phi\}$; otherwise, $\kappa(\phi) = \infty$. A world ranking $\kappa$ is *admissible* with a conditional knowledge base $KB = (L, D)$ iff $\kappa(\neg\phi) = \infty$ for all $\phi \in L$, and $\kappa(\phi) < \infty$ and $\kappa(\phi \wedge \psi) < \kappa(\phi \wedge \neg\psi)$ for all defaults $\psi \leftarrow \phi \in D$.

## 3.3 $\varepsilon$-Consistency

The notion of $\varepsilon$-consistency for conditional knowledge bases [1, 22] is defined as follows. A conditional knowledge base $KB$ is *$\varepsilon$-consistent* (resp., *$\varepsilon$-inconsistent*) iff a (resp., no) world ranking exists that is admis-

sible with $KB$.

The existence of a world ranking that is admissible with $KB = (L, D)$ is equivalent to (a) the existence of a default ranking on $D$ that is admissible with $KB$, if $D \neq \emptyset$, and to (b) the satisfiability of $D$, otherwise. Here, a *default ranking* $\sigma$ on a set of defaults $D$ maps each default $d \in D$ to a nonnegative integer. We say that $\sigma$ is *admissible* with a conditional knowledge base $KB = (L, D)$ iff each $D' \subseteq D$ that is under $L$ in conflict with some $d \in D$ contains a default $d'$ such that $\sigma(d') < \sigma(d)$.

Another characterization of the notion of $\varepsilon$-consistency for conditional knowledge bases is given as follows. A conditional knowledge base $KB = (L, D)$ with $D \neq \emptyset$ is $\varepsilon$-consistent iff there exists an ordered partition $(D_0, \ldots, D_k)$ of $D$ such that either (a) every $D_i$, $0 \leq i \leq k$, is the set of all $d \in \bigcup_{j=i}^{k} D_j$ tolerated under $L$ by $\bigcup_{j=i}^{k} D_j$, or (b) for every $i$, $0 \leq i \leq k$, each $d \in D_i$ is tolerated under $L$ by $\bigcup_{j=i}^{k} D_j$. The unique partition $(D_0, \ldots, D_k)$ of $D$ in (a) is called the *z-partition* of $KB$.

**Example 3.2** *(Penguins cont'd)* The conditional knowledge base $KB = (L, D)$ of Example 3.1 is $\varepsilon$-consistent. Its $z$-partition $(P_0, P_1)$, which satisfies both (a) and (b) above, is given by $(P_0, P_1) = (\{$*wings* $\leftarrow$*bird*, *fly* $\leftarrow$*bird*$\}, \{\neg$*fly* $\leftarrow$*penguin*$\})$.

The $z$-partition $(D_0, \ldots, D_k)$ of an $\varepsilon$-consistent conditional knowledge base $KB = (L, D)$ gives rise to a natural notion of entailment from $KB$, called *entailment in System Z* (due to Pearl [41, 24]), which is based on the following default ranking $z$ and world ranking $\kappa^z$. For every $j \in \{0, \ldots, k\}$, each $d \in D_j$ is assigned the value $j$ under $z$. The world ranking $\kappa^z$ on all worlds $I \in \mathcal{I}_\Phi$ is then defined by:

$$\kappa^z(I) \;=\; \begin{cases} \infty & \text{if } I \not\models L \\ 0 & \text{if } I \models L \cup D \\ 1 + \max\limits_{d \in D:\, I \not\models d} z(d) & \text{otherwise.} \end{cases}$$

Note that $z$ is a default ranking on $KB$ that is admissible with $KB$, and $\kappa^z$ is a world ranking that is admissible with $KB$. The notion of entailment in System $Z$ is then defined as follows. A default $\psi \leftarrow \phi$ is *entailed in System Z* by an $\varepsilon$-consistent conditional knowledge base $KB$ iff either $\kappa^z(\phi) = \infty$ or $\kappa^z(\phi \wedge \psi) < \kappa^z(\phi \wedge \neg\psi)$.

**Example 3.3** *(Penguins cont'd)* The conditional knowledge base $KB = (L, D)$ of Example 3.1 entails in System $Z$ the defaults *fly* $\leftarrow$*bird* and $\neg$*fly* $\leftarrow$*penguin*.

# 4 Conditional Preference Bases

In this section, we first define the syntax of conditional preferences, which are intuitively statements of the form "generally, if $\phi$ holds, then $\alpha$ is preferred over $\neg\alpha$ with strength $s$" (which are inspired by variable-strength defaults in conditional knowledge bases [24]). We then define their semantics in terms of object rankings (which is based on conditional knowledge bases; see Sections 3 and 10), and we introduce the notion of consistency for conditional preference bases.

## 4.1 Syntax

We assume a finite set $\mathcal{C}$ of concepts in $\mathcal{SHIF}(\mathbf{D})$ or $\mathcal{SHOIN}(\mathbf{D})$ such that $\top \in \mathcal{C}$. The elements of $\mathcal{C}$ are called *classification concepts*. Note that they are fully general concepts and not necessarily atomic concepts.

Intuitively, they are the relevant description logic concepts for defining preference relationships. A *conditional preference* is of the form $(\alpha|\phi)[s]$, where $\alpha$ and $\phi$ are concepts from $\mathcal{C}$ (called its *head* and its *body*, respectively), and $s$ is an integer from $\{1, \ldots, 100\}$ (called its *strength*). Informally, $(\alpha|\phi)[s]$ expresses that (i) generally, among the objects satisfying $\phi$, the ones satisfying $\alpha$ are preferred over those satisfying $\neg\alpha$, and (ii) this preference relationship holds with strength $s$. We often write $(\alpha)[s]$ to abbreviate $(\alpha|\top)[s]$.

**Definition 4.1** A *conditional preference base* is a triple $PB = (T, A, P)$, where $T$ is a knowledge base in $\mathcal{SHIF}(\mathbf{D})$ or $\mathcal{SHOIN}(\mathbf{D})$, $A$ is a finite set of concepts from $\mathcal{C}$, and $P$ is a finite set of conditional preferences.

Intuitively, conditional preference bases $PB = (T, A, P)$ encode rankings on a set of individuals $o$. Here, intuitively, $T$ represents background knowledge, and $A$ represents assertional knowledge about each $o$ (that is, $A$ represents the set of all $\alpha(o)$ such that $\alpha \in A$), while $P$ represents conditional preferences about each $o$ (that is, $P$ represents the set of all $(\alpha(o)|\phi(o))[s]$ such that $(\alpha|\phi)[s] \in P$). Observe also that the statements in $T$ and $A$ are *strict* (that is, they must always hold), while the ones in $P$ are *defeasible* (that is, they may have exceptions and thus do not always hold), since $P$ may not always be satisfiable as a whole.

**Example 4.2** *(Products cont'd)* The assertional knowledge "either a PC or a laptop" and the preference relationships "generally, PC's are preferred over laptops with strength 20", "generally, laptops on sale are preferred over PC's on sale with strength 70", and "generally, inexpensive products are preferred over expensive ones with strength 90" can be encoded by the conditional preference base $PB = (T, A, P)$, where $T$ is the description logic knowledge base $KB$ of Example 2.1, $A = \{\mathit{PC} \sqcup \mathit{Laptop}\}$, and $P = \{(\mathit{PC})[20], (\mathit{Laptop}|\mathit{Sale})[70], (\mathit{Inexpensive})[90]\}$. Note that, consequently, the set $\mathcal{C}$ of classification concepts contains in particular the description logic concepts $\mathit{PC} \sqcup \mathit{Laptop}$, *PC*, *Laptop*, *Sale*, and *Inexpensive*.

## 4.2 Semantics

We now define some basic semantic concepts, including objects (which are subsets of $\mathcal{C}$) and object rankings (which are certain functions that map every object to a rank from $\{0, 1, \ldots\} \cup \{\infty\}$), and we then associate with every conditional preference base a set of admissible object rankings as a formal semantics.

Formally, an *object* $o$ is a set of concepts from $\mathcal{C}$ such that $\{\phi(i) \mid \phi \in o\} \cup \{\neg\phi(i) \mid \phi \in \mathcal{C} \setminus o\}$ is satisfiable, where $i$ is a new individual. Informally, every object $o$ represents all individuals $i$ that are fully specified on $\mathcal{C}$ in the sense that $i$ belongs (resp., does not belong) to every concept $\phi \in o$ (resp., $\phi \in \mathcal{C} \setminus o$). That is, objects are essentially classifying individuals relative to the ranking-relevant properties specified in $\mathcal{C}$. We denote by $\mathcal{O}_\mathcal{C}$ the set of all objects relative to $\mathcal{C}$. An object *o satisfies* a description logic knowledge base $T$, denoted $o \models T$, iff $T \cup \{\phi(i) \mid \phi \in o\} \cup \{\neg\phi(i) \mid \phi \in \mathcal{C} \setminus o\}$ is satisfiable, where $i$ is a new individual. An object *o satisfies* a concept $\phi \in \mathcal{C}$, denoted $o \models \phi$, iff $\phi \in o$. An object *o satisfies* a set of concepts $A \subseteq \mathcal{C}$, denoted $o \models A$, iff $o$ satisfies all $\phi \in A$. A concept $\phi \in \mathcal{C}$ is *satisfiable* iff there exists an object $o \in \mathcal{O}_\mathcal{C}$ that satisfies $\phi$. The satisfaction of concepts by objects and the satisfiability of concepts are naturally extended to Boolean combinations of concepts from $\mathcal{C}$. An object *o satisfies* a conditional preference $(\alpha|\phi)[s]$, denoted $o \models (\alpha|\phi)[s]$, iff $o \models \neg\phi \sqcup \alpha$. We say that *o satisfies* a set of conditional preferences $P$, denoted $o \models P$, iff $o$ satisfies all $p \in P$. We say that *o verifies* $(\alpha|\phi)[s]$ iff $o \models \phi \sqcap \alpha$. We say that *o falsifies* $(\alpha|\phi)[s]$, denoted $o \not\models (\alpha|\phi)[s]$, iff $o \models \phi \sqcap \neg\alpha$. A set of conditional preferences $P$ *tolerates* a conditional preference $p$ *under* a description logic knowledge base $T$ and a set of classification concepts $A \subseteq \mathcal{C}$, or $p$ *is tolerated under* $T$ and $A$ *by* $P$, iff there exists an object $o$ that satisfies $T \cup A \cup P$ (that is, the object $o$ satisfies $T$, $A$, and $P$) and verifies $p$. We say that $P$ is *under* $T$ *and* $A$ *in conflict* with $p$ iff $P$ does not tolerate $p$ under $T$ and $A$.

An *object ranking* $\kappa$ is a mapping $\kappa\colon \mathcal{O}_\mathcal{C} \to \{0,1,\ldots\} \cup \{\infty\}$ such that $\kappa(o) = 0$ for at least one object $o \in \mathcal{O}_\mathcal{C}$. It is extended to all Boolean combinations $\phi$ of concepts from $\mathcal{C}$ as follows. If $\phi$ is satisfiable, then $\kappa(\phi) = \min\{\kappa(o) \mid o \in \mathcal{O}_\mathcal{C}, o \models \phi\}$; otherwise, $\kappa(\phi) = \infty$. We say that $\kappa$ is *admissible* with a description logic knowledge base $T$ (resp., a set of concepts $A$) iff $\kappa(o) = \infty$ for all $o \in \mathcal{O}_\mathcal{C}$ such that $o \not\models T$ (resp., $o \not\models A$). We say that $\kappa$ is *admissible* with a conditional preference $(\alpha|\phi)[s]$ iff either $\kappa(\phi) = \infty$ or $\kappa(\phi \sqcap \alpha) < \kappa(\phi \sqcap \neg\alpha)$. We say that $\kappa$ is *admissible* with $PB = (T, A, P)$ iff $\kappa$ is admissible with $T$, $A$, and all $p \in P$.

### 4.3 Consistency

We now define the consistency of conditional preference bases, which properly generalizes the $\varepsilon$-consistency of conditional knowledge bases (see Section 3.3).

**Definition 4.3** A conditional preference base $PB$ is *consistent* (resp., *inconsistent*) iff an (resp., no) object ranking $\kappa$ exists that is admissible with $PB$.

Observe that a conditional preference base $PB = (T, A, P)$ with $P = \emptyset$ is consistent iff $T \cup \{\alpha(i) \mid \alpha \in A\}$ is satisfiable. We now summarize some results that carry over from $\varepsilon$-consistency in conditional knowledge bases.

The following result shows that the existence of an object ranking that is admissible with $PB = (T, A, P)$, where $P \neq \emptyset$, is equivalent to the existence of a preference ranking on $P$ that is admissible with $PB$. Here, a *preference ranking* $\sigma$ on $P$ maps each $p \in P$ to a nonnegative integer. We say that a preference ranking $\sigma$ on $P$ is *admissible* with $PB = (T, A, P)$ iff every $P' \subseteq P$ that is under $T$ and $A$ in conflict with some $p \in P$ contains some $p'$ such that $\sigma(p') < \sigma(p)$.

**Theorem 4.4** *A conditional preference base $PB = (T, A, P)$ with $P \neq \emptyset$ is consistent iff there exists a preference ranking $\sigma$ on $P$ that is admissible with $PB$.*

The next result shows that the consistency of $PB = (T, A, P)$ is equivalent to the existence of an ordered partition of $P$ with certain properties. Here, recall that $p$ is tolerated under $T$ and $A$ by $\bigcup_{j=i}^{k} P_j$ iff there exists an object $o$ that satisfies $T \cup A \cup \bigcup_{j=i}^{k} P_j$ and verifies $p$ (see Section 4.2).

**Theorem 4.5** *A conditional preference base $PB = (T, A, P)$ with $P \neq \emptyset$ is consistent iff there exists an ordered partition $(P_0, \ldots, P_k)$ of $P$ such that either (a) every $P_i$, $0 \leq i \leq k$, is the set of all $p \in \bigcup_{j=i}^{k} P_j$ tolerated under $T$ and $A$ by $\bigcup_{j=i}^{k} P_j$, or (b) for every $i$, $0 \leq i \leq k$, each $p \in P_i$ is tolerated under $T$ and $A$ by $\bigcup_{j=i}^{k} P_j$.*

We call the unique partition $(P_0, \ldots, P_k)$ of $P$ in (a) the *z-partition* of $PB$. Note that the notion of a $z$-partition for conditional preference bases properly generalizes the notion of a $z$-partition for conditional knowledge bases (see Section 3.3).

**Example 4.6** *(Products cont'd)* It is not difficult to verify that the conditional preference base $PB$ of Example 4.2 is consistent, and that its $z$-partition is given by $(P_0, P_1) = (\{(\textit{PC})[20], (\textit{Inexpensive})[90]\}, \{(\textit{Laptop}|\textit{Sale})[70]\})$.

The $z$-partition $(P_0, \ldots, P_k)$ of $PB = (T, A, P)$ gives rise to a natural notion of entailment from $PB$, called *$z$-entailment*, which properly generalizes the notion of entailment in System $Z$ for conditional knowledge bases (see Section 3.3). The notion of $z$-entailment is based on the following preference ranking $z$ and object ranking $\kappa^z$. For every $j \in \{0, \ldots, k\}$, each $p \in P_j$ is assigned the value $j$ under $z$. The object ranking $\kappa^z$ on all objects $o \in \mathcal{O}_\mathcal{C}$ is then defined as follows:

$$\kappa^z(o) \;=\; \begin{cases} \infty & \text{if } o \not\models T \cup A \\ 0 & \text{if } o \models T \cup A \cup P \\ 1 + \max\limits_{p \in P:\, o \not\models p} z(p) & \text{otherwise.} \end{cases}$$

Note that $z$ is a preference ranking on $PB$ that is admissible with $PB$, and $\kappa^z$ is an object ranking that is admissible with $PB$. We define the notion of *$z$-entailment* as follows. A conditional preference $p = (\alpha|\phi)[s]$ is a *$z$-consequence* of $PB$, denoted $PB \mid\!\sim p$, iff either $\kappa^z(\phi) = \infty$ or $\kappa^z(\phi \wedge \alpha) < \kappa^z(\phi \wedge \neg\alpha)$.

Since the notion of $z$-entailment generalizes the notion of entailment in System $Z$, it has similar semantic properties. In particular, it realizes some inheritance of conditional preferences along subclass relationships, where conditional preferences of more specific classes override the ones of less specific classes.

**Example 4.7** *(Products cont'd)* Let the conditional preference base $PB$ be defined as in Example 4.2. Then, it is not difficult to verify that we obtain (*PC*)[20], (*Laptop*|*Sale*)[70], (*Inexpensive*)[90], and (*Inexpensive* | *Made_By_IBM*)[90] as some $z$-consequences of $PB$.

# 5 Ranking Objects under Conditional Preference Bases

In this section, we define the two object rankings $\kappa^{sum}$ and $\kappa^{lex}$, which reflect the conditional preferences of a consistent conditional preference base $PB = (T, A, P)$. Informally, every object $o$ that does not satisfy $T \cup A$ is associated with the rank $\infty$, while every object $o$ that satisfies $T \cup A$ is associated with a nonnegative integer as a rank, which depends on how well $o$ satisfies the conditional preferences in $P$, where the rank is low (resp., high) for more (resp., less) desired $o$ relative to $P$.

The main idea behind the first object ranking can be described as follows: $\kappa^{sum}$ interprets the strengths of the conditional preferences in $P$ as *costs* (e.g., monetary costs). Intuitively, if an object $o$ falsifies a conditional preference from $P$ of strength $s$, then this produces the cost $s$, and the overall rank of the object $o$ is then given by the sum of all these costs. That is, the rank of every object is given by the sum of the strengths of all falsified conditional preferences in $P$. For example, when looking for an apartment, we may rank a given collection of apartments according to whether they satisfy our conditional preferences, where every falsified conditional preference produces a degree of dissatisfaction, and the rank of an apartment is given by the sum of all such degrees of dissatisfaction.

The second object ranking is based on a different interpretation of the strengths: $\kappa^{lex}$ interprets the strengths of the conditional preferences in $P$ as *priorities* and not as costs. That is, falsifying a conditional preference from $P$ of strength $s$ is always worse than falsifying any set of conditional preferences from $P$ of strength at most $s - 1$. Thus, falsifying a collection of conditional preferences from $P$ of low strengths can never be worse than falsifying even a single conditional preference from $P$ of high strength. For example, when ranking a set of apartments $\mathcal{O}$ relative to a set of conditional preferences $P$, to obtain the apartments in $\mathcal{O}$ of lowest rank, we first select those in $\mathcal{O}$ that satisfy a maximal set of conditional preferences of highest strength $s$, among which we then select the ones that satisfy a maximal set conditional preferences of strength $s - 1$, and so on.

If $P$ contains only conditional preferences of the form $(\psi|\top)[s]$, then computing the above two object rankings is quite easy, since $\kappa^{sum}$ can be computed by summing up the strengths of all falsified conditional preferences in $P$, while $\kappa^{lex}$ can be computed by a lexicographic order relative to the strengths. However, if $P$ contains fully general conditional preferences of the form $(\psi|\phi)[s]$, then computing the above two object rankings is technically much more involved. The main technical difficulty is roughly that we want any conditional preference in $P$ of the form $(\psi|\phi)[s]$ to apply on all objects that satisfy $\phi$ and also on all objects that satisfy some more specific $\phi'$ *as long as this is compatible with* $P$ (that is, as long as $P$ contains no conditional preference $(\psi'|\phi')[s']$ that is incompatible with $(\psi|\phi)[s]$). That is, from another perspective, the set of conditional preferences $P$ *implicitly encodes some exceptions for the bodies of its conditional preferences*, which we have to make explicit in order to correctly compute the above two object rankings. To this end, we rewrite $P$ from a set of defeasible statements to a set of strict statements $P^\star$, which is done by adding exceptions to the bodies of the conditional preferences in $P$.

**Example 5.1** *(Products cont'd)* Consider again $PB = (T, A, P)$ of Example 4.2. Ignoring the strengths, $P$ encodes that (i) "PCs are preferred over laptops (with strength 20), as long as they are not on sale, because in that case, laptops are preferred over PCs (with strength 70)" and (ii) "inexpensive products are preferred over expensive ones (with strength 90)". Hence, for technical reasons, laptops on sale always falsify the conditional preference $p = (\textit{PC})[20]$. However, this is not desired. Thus, when computing the rank of laptops on sale, we have to avoid such falsifications. We do this by rewriting $p$ and thus $PB$. The rewritten conditional preference base $PB^\star = (T, A, P^\star)$ is given by $P^\star = \{(\textit{PC}|\neg\textit{Sale})[20],\ (\textit{Laptop}|\textit{Sale})[70],\ (\textit{Inexpensive})[90]\}$. It is obtained from $PB$ by adding the exception $\neg\textit{Sale}$ to the body of the conditional preference $(\textit{PC})[20]$.

To make explicit the above exceptions in the bodies of conditional preferences in $P$, we use a sophisticated and well-explored machinery from conditional knowledge bases, which is formally described as follows. A conditional preference base $PB = (T, A, P)$ is *flat* iff its $z$-partition is given by $(P)$ and thus consists only of one component. Given a conditional preference base $PB = (T, A, P)$, a *non-defeasible equivalent* $PB^\star = (T, A, P^\star)$ to $PB$ satisfies the properties that (i) $PB^\star$ is flat, (ii) $PB \mid\!\sim p$ for all $p \in P^\star$, and (iii) $P^\star = \{(\alpha|\phi \sqcap \psi_p)[s] \mid p = (\alpha|\phi)[s] \in P\}$, where $\psi_p$ is a conjunction of negated bodies that occur in $P$. Informally, (iii) says that the rewriting from $P$ to $P^\star$ adds exceptions to the bodies of conditional preferences in $P$, (ii) says that the rewriting does not change the semantic meaning of the set of conditional preferences $P$, and (i) says that $P^\star$ encodes no further hidden exceptions. In Section 6, we present Algorithm *flatten*, which transforms a consistent conditional preference base $PB$ into a non-defeasible equivalent $PB^\star$.

We are now ready to define the object rankings $\kappa^{sum}$ and $\kappa^{lex}$. Informally, the object ranking $\kappa^{sum}$ associates with every object the sum of the strengths of all conditional preferences in $P^\star$ that are falsified by $o$. Roughly, objects with smaller values under $\kappa^{sum}$ are those that satisfy more conditional preferences with larger strengths.

**Definition 5.2** Let $PB = (T, A, P)$ be a consistent conditional preference base, and let $PB^\star = (T, A, P^\star)$ be its non-defeasible equivalent (computed by *flatten*). Then, the object ranking $\kappa^{sum}$ is defined as follows for all objects $o \in \mathcal{O}_\mathcal{C}$:

$$\kappa^{sum}(o) \;=\; \begin{cases} \infty & \text{if } o \not\models T \cup A \\ \displaystyle\sum_{p=(\alpha|\phi)[s] \in P^\star :\ o \not\models p} s & \text{otherwise.} \end{cases} \tag{1}$$

The object ranking $\kappa^{lex}$, in contrast, is based on a lexicographic order. Roughly, objects with smaller ranks are those that satisfy more conditional preferences with larger strengths, where satisfying one conditional preference of strength $s$ is strictly preferred to satisfying any set of conditional preferences of strength at most $s-1$.

**Definition 5.3** Let $PB=(T,A,P)$ be a consistent conditional preference base, and let $PB^\star=(T,A,P^\star)$ be its non-defeasible equivalent (computed by *flatten*). Then, the object ranking $\kappa^{lex}$ is defined as follows for all objects $o\in\mathcal{O}_\mathcal{C}$ (where $n_j$ with $j\in\{1,\ldots,100\}$ is the number of all $p\in P^\star$ of strength $j$):

$$\kappa^{lex}(o) \;=\; \begin{cases} \infty & \text{if } o\not\models T\cup A \\ \sum\limits_{i=1}^{100} |\{p=(\alpha|\phi)[i]\in P^\star \mid o\not\models p\}|\cdot \Pi_{j=1}^{i-1}(n_j+1) & \text{otherwise.} \end{cases} \tag{2}$$

**Example 5.4** *(Products cont'd)* Consider again $PB=(T,A,P)$ of Example 2.1. Recall from Example 5.1 that the rewritten conditional preference base $PB^\star=(T,A,P^\star)$ is given by $P^\star=\{(PC|\neg Sale)[20],(Laptop\mid Sale)[70],(Inexpensive)[90]\}$. The object rankings $\kappa^{sum}$ and $\kappa^{lex}$ for $PB$ are shown in Fig. 1. For example, the rank of $o_3$ is $\infty$, since it does not satisfy $A$, and the ranks of $o_5$, $o_8$, and $o_{11}$ under $\kappa^{sum}$ and $\kappa^{lex}$ are calculated as follows:

$$\begin{array}{lcrcrcrcl}
\kappa^{sum}(o_5) &=& 1\cdot 20 &+& 0\cdot 70 &+& 1\cdot 90 &=& 110,\\
\kappa^{lex}(o_5) &=& 1\cdot 1 &+& 0\cdot(1+1) &+& 1\cdot(1+1)\cdot(1+1) &=& 5,\\
\kappa^{sum}(o_8) &=& 0\cdot 20 &+& 0\cdot 70 &+& 0\cdot 90 &=& 0,\\
\kappa^{lex}(o_8) &=& 0\cdot 1 &+& 0\cdot(1+1) &+& 0\cdot(1+1)\cdot(1+1) &=& 0,\\
\kappa^{sum}(o_{11}) &=& 0\cdot 20 &+& 1\cdot 70 &+& 1\cdot 90 &=& 160,\\
\kappa^{lex}(o_{11}) &=& 0\cdot 1 &+& 1\cdot(1+1) &+& 1\cdot(1+1)\cdot(1+1) &=& 6,
\end{array}$$

since $o_5,o_8,o_{11}\models T\cup A$, and $o_5\models \neg PC\sqcap\neg Sale$, $o_5\models Laptop\sqcup\neg Sale$, $o_5\models\neg Inexpensive$, $o_8\models PC\sqcup Sale$, $o_8\models Laptop\sqcup\neg Sale$, $o_8\models Inexpensive$, $o_{11}\models PC\sqcup Sale$, $o_{11}\models\neg Laptop\sqcap Sale$, and $o_{11}\models\neg Inexpensive$ (and so $o_5\not\models(PC|\neg Sale)[20]$, $o_5\models(Laptop|Sale)[70]$, $o_5\not\models(Inexpensive)[90]$, $o_8\models(PC|\neg Sale)[20]$, $o_8\models(Laptop|Sale)[70]$, $o_8\models(Inexpensive)[90]$, $o_{11}\models(PC|\neg Sale)[20]$, $o_{11}\not\models(Laptop|Sale)[70]$, and $o_{11}\not\models(Inexpensive)[90]$, respectively).

We provide another example, which shows that the two object rankings $\kappa^{sum}$ and $\kappa^{lex}$ generally indeed produce two different orderings on the objects.

**Example 5.5** *(Products cont'd)* Let $PB=(T,A,P)$ be obtained from $PB$ of Example 4.2 by adding $(Made\_By\_IBM)[70]$ to $P$. The rewritten conditional preference base $PB^\star=(T,A,P^\star)$ is obtained from the one in Example 5.1 by adding $(Made\_By\_IBM)[70]$ to $P^\star$. For the objects $o=\{PC\sqcup Laptop, PC, Sale, Inexpensive\}$ and $o'=\{PC\sqcup Laptop, Laptop, Sale, Made\_By\_IBM\}$, we then obtain the ranks $\kappa^{sum}(o)=140>90=\kappa^{sum}(o')$ and $\kappa^{lex}(o)=4<6=\kappa^{lex}(o')$.

Summarizing, every object ranking $\kappa\in\{\kappa^{sum},\kappa^{lex}\}$ of a conditional preference base $PB$ represents the preference relationships encoded in $PB$. For every (fully specified) object $o$, the *rank* of $o$ under $PB$ is given by $\kappa(o)$. Every object ranking $\kappa\in\{\kappa^{sum},\kappa^{lex}\}$ can also be used to compare two (fully specified) objects $o,o'\in\mathcal{O}_\mathcal{C}$ as follows. The *distance* between the objects $o$ and $o'$ under $PB$ is defined as $|\kappa(o)-\kappa(o')|$. Furthermore, the (*credulous*) *rank* of a *partially specified object* $\phi$ (which is a Boolean combination of

Table 1: The object rankings $\kappa^{sum}$ and $\kappa^{lex}$.

| | *PC* | *Laptop* | *Sale* | *Inexpensive* | $\kappa^{sum}$ | $\kappa^{lex}$ |
|---|---|---|---|---|---|---|
| $o_1$ | **false** | **false** | **false** | **false** | $\infty$ | $\infty$ |
| $o_2$ | **false** | **false** | **false** | **true** | $\infty$ | $\infty$ |
| $o_3$ | **false** | **false** | **true** | **false** | $\infty$ | $\infty$ |
| $o_4$ | **false** | **false** | **true** | **true** | $\infty$ | $\infty$ |
| $o_5$ | **false** | **true** | **false** | **false** | 110 | 5 |
| $o_6$ | **false** | **true** | **false** | **true** | 20 | 1 |
| $o_7$ | **false** | **true** | **true** | **false** | 90 | 4 |
| $o_8$ | **false** | **true** | **true** | **true** | 0 | 0 |
| $o_9$ | **true** | **false** | **false** | **false** | 90 | 4 |
| $o_{10}$ | **true** | **false** | **false** | **true** | 0 | 0 |
| $o_{11}$ | **true** | **false** | **true** | **false** | 160 | 6 |
| $o_{12}$ | **true** | **false** | **true** | **true** | 70 | 2 |
| $o_{13}$ | **true** | **true** | **false** | **false** | $\infty$ | $\infty$ |
| $o_{14}$ | **true** | **true** | **false** | **true** | $\infty$ | $\infty$ |
| $o_{15}$ | **true** | **true** | **true** | **false** | $\infty$ | $\infty$ |
| $o_{16}$ | **true** | **true** | **true** | **true** | $\infty$ | $\infty$ |

concepts from $\mathcal{C}$) under $PB$ is defined as $\min_{o \in \mathcal{O}_\mathcal{C} : o \models \phi} \kappa(o)$. Finally, the (*credulous*) *distance* between two partially specified objects $\phi$ and $\phi'$ is defined as $\min_{o,o' \in \mathcal{O}_\mathcal{C} : o \models \phi, o' \models \phi'} |\kappa(o) - \kappa(o')|$.

# 6 Algorithms

In this section, we formally specify the main computational tasks related to conditional preference bases, and we provide algorithms for solving them. This shows in particular that the tasks are all decidable. The main computational tasks for conditional preference bases are formally given as follows:

CONSISTENCY: Given a conditional preference base $PB$, decide whether $PB$ is consistent.

Z-PARTITION: Given a conditional preference base $PB$, compute the $z$-partition of $PB$ (if it exists).

FLATTEN: Given a conditional preference base $PB$, compute a non-defeasible equivalent $PB^\star$ to $PB$ (if one exists).

$s$-RANKING: Given a conditional preference base $PB$ and a set of objects $\mathcal{O} \subseteq \mathcal{O}_\mathcal{C}$, compute the ranking $\kappa^s$ on $\mathcal{O}$ for $PB$ (if it exists), where $s \in \{sum, lex\}$.

The problem of deciding the consistency (and computing the $z$-partition) of a conditional preference base $PB$ is solved by Algorithm *z-partition* in Fig. 1, which generalizes an algorithm for deciding the $\varepsilon$-consistency (and computing the $z$-partition) of a conditional knowledge base in default reasoning [22]. The algorithm takes as input a conditional preference base $PB = (T, A, P)$, and it returns as output the $z$-partition of $PB$, if $PB$ is consistent, and *nil*, otherwise. In lines (1) and (2), it deals with the cases where $T \cup A$ is unsatisfiable and $P = \emptyset$, respectively. In lines (3)–(11), it computes and returns the $z$-partition of $PB$ as specified in Theorem 4.5 (a).

**Example 6.1** *(Products cont'd)* Consider again $PB = (T, A, P)$ of Example 4.2. Since $T \cup A$ is satisfiable and $P \neq \emptyset$, we set $H := P$ and $i := -1$ in lines (3) and (4), respectively, and enter the loop in lines (5)–(9). After the first run of the loop, $i = 0$, $P_0 = \{(\textit{PC})[20], (\textit{Inexpensive})[90]\}$, and $H = \{(\textit{Laptop}|\textit{Sale})[70]\}$. After the second run, $i = 1$, $P_1 = \{(\textit{Laptop}|\textit{Sale})[70]\}$, and $H = \emptyset$. We thus leave the loop, and return $(P_0, P_1)$ as the $z$-partition of $PB$, which also shows that $PB$ is consistent.

**Algorithm** *z-partition*

**Input**: conditional preference base $PB = (T, A, P)$.
**Output**: $z$-partition $(P_0, \ldots, P_n)$ of $PB$, if $PB$ is consistent; *nil*, otherwise.

1. **if** $T \cup A$ is unsatisfiable **then return** *nil*;
2. **if** $P = \emptyset$ **then return** $()$;
3. $H := P$;
4. $i := -1$;
5. **repeat**
6. $\quad i := i + 1$;
7. $\quad P_i := \{p \in H \mid p$ is tolerated under $T$ and $A$ by $H\}$;
8. $\quad H := H \setminus P_i$
9. **until** $H = \emptyset$ **or** $P_i = \emptyset$;
10. **if** $H = \emptyset$ **then return** $(P_0, \ldots, P_i)$
11. $\quad$ **else return** *nil*.

Figure 1: Algorithm *z-partition*.

The problem of rewriting $PB$ to a non-defeasible equivalent $PB^\star$ (for the object rankings $\kappa^{sum}$ and $\kappa^{lex}$ in Section 5) is solved by Algorithm *flatten* in Fig. 2, which is related to a rewriting algorithm in fuzzy default reasoning [15]. The algorithm takes as input a conditional preference base $PB = (T, A, P)$, and it returns as output a non-defeasible equivalent $PB^\star$ to $PB$, if $PB$ is consistent, and *nil*, otherwise. In lines (1)–(3), the algorithm handles the cases where $PB$ is inconsistent and $P = \emptyset$, and it computes the $z$-partition of $PB$ (if it exists). In lines (4)–(14), it adds for every $i \in \{1, \ldots, n\}$ to some bodies of conditional preferences in $P_0 \cup \cdots \cup P_{i-1}$ negations of bodies from $P_i$ and returns the thus computed conditional preference base (see Theorem 6.4 for the correctness of Algorithm *flatten*).

**Example 6.2** *(Products cont'd)* Consider again $PB = (T, A, P)$ of Example 4.2. We first run Algorithm *z-partition* to compute the $z$-partition $(P_0, P_1)$ of $PB$ (and thus also to verify that $PB$ is consistent). Since this is successful (see Example 6.1) and $P \neq \emptyset$, we then set $D := P_0$ in line (4). We then run once (for $i = 1$) through the loop in lines (5)–(13), where we set $H := \emptyset$ and run twice (for $p_1 = (\mathit{PC})[20]$ and $p_2 = (\mathit{Inexpensive})[90]$) through the loop in lines (7)–(11). Here, $F_{p_1} = \{\mathit{Sale}\}$ and $H = \{(\mathit{PC}|\neg\mathit{Sale})[20]\}$ after the first run, and $F_{p_2} = \emptyset$ and $H = \{(\mathit{PC}|\neg\mathit{Sale})[20], (\mathit{Inexpensive})[90]\}$ after the second run. Hence, $D = \{(\mathit{PC}|\neg\mathit{Sale})[20], (\mathit{Inexpensive})[90], (\mathit{Laptop}|\mathit{Sale})[70]\}$ after line (12), and we return $(T, A, D)$ as the result.

Finally, computing the ranking functions $\kappa^{sum}$ and $\kappa^{lex}$ is done by Algorithms *sum-ranking* and *lex-ranking* in Figs. 3 and 4, respectively. In lines (1)–(4), the two algorithms handle the cases where $PB$ is inconsistent and $P = \emptyset$, and they compute a non-defeasible equivalent $PB^\star$ to $PB$ (if it exists). In lines (5)–(7) and (5)–(14), the two algorithms compute (and return) the ranking functions $\kappa^{sum}$ and $\kappa^{lex}$ for the case where $PB$ is consistent and $P \neq \emptyset$ using equations (1) and (2), respectively.

**Example 6.3** *(Products cont'd)* Consider again $PB = (T, A, P)$ of Example 4.2. Suppose we want to rank the objects $o_3 = \{\mathit{Sale}\}$ and $o_{11} = \{\mathit{PC} \sqcup \mathit{Laptop}, \mathit{PC}, \mathit{Sale}\}$. In both *sum-ranking* and *lex-ranking*, we first

**Algorithm** *flatten*

**Input**: conditional preference base $PB = (T, A, P)$.
**Output**: non-defeasible equivalent $PB^\star$ to $PB$, if $PB$ is consistent; *nil*, otherwise.

1. **if** *z-partition*$(PB) \neq$ *nil* **then** $(P_0, \ldots, P_n) :=$ *z-partition*$(PB)$
2. **else return** *nil*;
3. **if** $P = \emptyset$ **then return** $PB$;
4. $D := P_0$;
5. **for** $i := 1$ **to** $n$ **do begin**
6. $H := \emptyset$;
7. **for each** $p = (\alpha|\phi)[s] \in D$ **do begin**
8. $F_p := \{\psi \mid \exists(\gamma|\psi)[r] \in P_i\colon\ p$ is not tolerated under $T$ and $A \cup \{\psi\}$ by $D \cup P_i \cup \cdots \cup P_n\}$;
10. $H := H \cup \{(\alpha|\phi \sqcap \neg\psi_1 \sqcap \cdots \sqcap \neg\psi_l)[s]\}$, where $\{\psi_1, \ldots, \psi_l\} = F_p$
11. **end**;
12. $D := H \cup P_i$
13. **end**;
14. **return** $(T, A, D)$.

Figure 2: Algorithm *flatten*.

rewrite $PB = (T, A, P)$ using Algorithm *flatten*. Since this is successful, and $o_3$ does not satisfy (resp., $o_{11}$ satisfies) $T \cup A$, we then set the rank of $o_3$ (resp., $o_{11}$) to $\infty$ (resp., 0) in line (3). Since $P \neq \emptyset$ in line (4), we continue running once (for $o = o_{11}$) through the loop starting in line (5). In Algorithm *sum-ranking*, we then run three times (for $p_1 = (PC|\neg Sale)[20]$, $p_2 = (Laptop|Sale)[70]$, and $p_3 = (Inexpensive)[90]$) through the loop in line (6). Since $o_{11}$ satisfies $p_1$ and falsifies $p_2$ and $p_3$, its rank is $70 + 90 = 160$. Instead, in Algorithm *lex-ranking*, we set $n := 1$ and we run 100 times (for $i \in \{1, \ldots, 100\}$) through the loop in lines (7)–(12). Here, for $i = 20$, $i = 70$, and $i = 90$, we run once through the loop in (9), producing $h = 0$, $h = 1$, and $h = 1$. Since at the same time $n = 1$, $n = 1 \cdot 2 = 2$, and $n = 2 \cdot 2 = 4$, respectively, $o_{11}$'s rank is $0 \cdot 1 + 1 \cdot 2 + 1 \cdot 4 = 6$. Finally, we return the ranking on $o_3$ and $o_{11}$ in line (7).

The following result shows the correctness of all the above algorithms.

**Theorem 6.4** *Algorithms z-partition, flatten, sum-ranking, and lex-ranking are all correct, that is, they solve the problems* Z-PARTITION *(and* CONSISTENCY*),* FLATTEN*,* $sum$-RANKING*, and* $lex$-RANKING*, respectively.*

The next result shows that all the above algorithms can be reduced to a polynomial number of checks whether a description logic knowledge base is satisfiable.

**Theorem 6.5** *Given a conditional preference base $PB = (T, A, P)$, Algorithms z-partition and flatten can be done in $O(|P|^2)$ description logic satisfiability tests. Given additionally a set of objects $\mathcal{O} \subseteq \mathcal{O}_\mathcal{C}$, Algorithms sum-ranking and lex-ranking can be done in $O(|P|^2 + |\mathcal{O}|)$ description logic satisfiability tests.*

Observe here that the $O(|P|^2 + |\mathcal{O}|)$ description logic satisfiability tests for computing $\kappa^{sum}$ and $\kappa^{lex}$ are essentially used to transform the given $PB = (T, A, P)$ into a non-defeasible equivalent $PB^\star = (T, A, P^\star)$,

**Algorithm** *sum-ranking*

**Input**: conditional preference base $PB = (T, A, P)$ and set of objects $\mathcal{O} \subseteq \mathcal{O}_\mathcal{C}$.
**Output**: ranking $\kappa^{sum}$ on $\mathcal{O}$ for $PB$, if $PB$ is consistent; *nil*, otherwise.

1. **if** *flatten*$(T, A, P) \neq$ *nil* **then** $(T, A, P) :=$ *flatten*$(T, A, P)$
2. **else return** *nil*;
3. **for each** $o \in \mathcal{O}$ **do if** $o \models A \cup T$ **then** $\kappa(o) := 0$ **else** $\kappa(o) := \infty$;
4. **if** $P = \emptyset$ **then return** $\kappa$;
5. **for each** $o \in \mathcal{O}$ such that $\kappa(o) \neq \infty$ **do**
6. **for each** $p = (\alpha|\phi)[s] \in P$ **do if** $o \not\models p$ **then** $\kappa(o) := \kappa(o) + s + 1$;
7. **return** $\kappa$.

Figure 3: Algorithm *sum-ranking*.

and to decide which of the objects in the given $\mathcal{O}$ satisfy $T$. That is, the description logic satisfiability tests are part of a preprocessing step. The actual computation of $\kappa^{sum}$ and $\kappa^{lex}$ can then be done in time $O(|\mathcal{O}| \cdot |\mathcal{C}| \cdot (|A| + |P|))$, without further description logic satisfiability tests.

By Theorem 6.5, under the assumption that $|P|$ is bounded by a constant (which is a reasonable assumption in the application in literature search in Section 9), deciding whether $PB$ is consistent, computing the $z$-partition of $PB$, and computing a non-defeasible equivalent to $PB$ can all be done in a constant number of description logic satisfiability tests. Under the same assumption, computing the rankings $\kappa^{sum}$ and $\kappa^{lex}$ can be done in $O(|\mathcal{O}|)$ description logic satisfiability tests.

# 7 Complexity

In this section, we address the complexity of conditional preference bases. We first recall some necessary complexity classes, and previous complexity results on description logic satisfiability. We then provide our complexity results.

## 7.1 Complexity Classes and Previous Results

We assume that the reader has some elementary background in complexity theory, and is familiar with the concepts of Turing machines and oracle calls, polynomial-time transformations among problems, and the hardness and completeness of a problem for a complexity class [30, 31, 40]. We now briefly recall the complexity classes that we encounter in our complexity results below.

The class EXP (resp., NEXP) contains all decision problems that can be solved in exponential time on a deterministic (resp., nondeterministic) Turing machine. The class $\mathrm{P}^{\mathrm{NEXP}}$ contains all problems that are decidable in polynomial time on a deterministic Turing machine with the help of a NEXP oracle. The class $\mathrm{P}_{\|}^{\mathrm{NEXP}}$ contains the problems in $\mathrm{P}^{\mathrm{NEXP}}$ that are solvable in such a way that all oracle calls are done in parallel. The above complexity classes along with their inclusion relationships (all of which are currently believed to be strict) are summarized as follows:

$$\mathrm{EXP} \subseteq \mathrm{NEXP} \subseteq \mathrm{P}_{\|}^{\mathrm{NEXP}} \subseteq \mathrm{P}^{\mathrm{NEXP}}.$$

**Algorithm** *lex-ranking*

**Input**: conditional preference base $PB = (T, A, P)$ and set of objects $\mathcal{O} \subseteq \mathcal{O}_\mathcal{C}$.
**Output**: ranking $\kappa^{lex}$ on $\mathcal{O}$ for $PB$, if $PB$ is consistent; *nil*, otherwise.

1. **if** *flatten*$(T, A, P) \neq$ *nil* **then** $(T, A, P) :=$ *flatten*$(T, A, P)$
2. **else return** *nil*;
3. **for each** $o \in \mathcal{O}$ **do if** $o \models A \cup T$ **then** $\kappa(o) := 0$ **else** $\kappa(o) := \infty$;
4. **if** $P = \emptyset$ **then return** $\kappa$;
5. **for each** $o \in \mathcal{O}$ such that $\kappa(o) \neq \infty$ **do begin**
6. $n := 1$;
7. **for each** $i := 1$ **to** $100$ **do begin**
8. $h := 0$;
9. **for each** $p = (\alpha|\phi)[i] \in P$ **do if** $o \not\models p$ **then** $h := h + 1$
10. $\kappa(o) := \kappa(o) + h \cdot n$;
11. $n := n \cdot (|\{(\alpha|\phi)[s] \in P \mid s = i\}| + 1)$
12. **end**
13. **end**;
14. **return** $\kappa$.

Figure 4: Algorithm *lex-ranking*.

For classifying problems that compute an output value, function classes similar to the classes above have been introduced [45, 30]. In particular, FEXP, $\text{FP}_{\|}^{\text{NEXP}}$, and $\text{FP}^{\text{NEXP}}$ are the functional analogs of EXP, $\text{P}_{\|}^{\text{NEXP}}$, and $\text{P}^{\text{NEXP}}$, respectively.

Finally, we recall that the problem of deciding whether a knowledge base $L$ in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable is complete for EXP [52, 26] (resp., NEXP, assuming unary number encoding; see [26] and the NEXP-hardness proof for $\mathcal{ALCQI}$ in [52], which implies the NEXP-hardness of $\mathcal{SHOIN}(\mathbf{D})$).

## 7.2 Complexity Results

Our complexity results for the main computational tasks related to conditional preference bases are compactly summarized in Table 2. More concretely, for conditional preference bases $PB$ over $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$), the decision problem CONSISTENCY is complete for EXP (resp., NEXP), while the computation problems Z-PARTITION, FLATTEN, and $s$-RANKING, where $s \in \{sum, lex\}$, are complete for FEXP (resp., in $\text{FP}_{\|}^{\text{NEXP}}$, $\text{FP}^{\text{NEXP}}$, and $\text{FP}^{\text{NEXP}}$).

The following theorem formally states the complexity results for the decision problem CONSISTENCY. Hardness for EXP (resp., NEXP) follows from the hardness for EXP (resp., NEXP) of deciding description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$), while membership in EXP (resp., NEXP) follows from Theorem 6.5 (resp., 4.5 (b)) and the membership in EXP (resp., NEXP) of deciding description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$).

**Theorem 7.1** *The decision problem* CONSISTENCY *is complete for* EXP *(resp.,* NEXP*) when $PB$ is defined over $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$).*

Table 2: Complexity results.

| | $\mathcal{SHIF}(\mathbf{D})$ | $\mathcal{SHOIN}(\mathbf{D})$ |
|---|---|---|
| CONSISTENCY | EXP-complete | NEXP-complete |
| Z-PARTITION | FEXP-complete | in $\text{FP}_{\parallel}^{\text{NEXP}}$ |
| FLATTEN | FEXP-complete | in $\text{FP}^{\text{NEXP}}$ |
| $s$-RANKING | FEXP-complete | in $\text{FP}^{\text{NEXP}}$ |

The next theorem formally states the complexity results for Z-PARTITION, FLATTEN, and $s$-RANKING. These results follow from Theorem 6.5 and the complexity of deciding description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$).

**Theorem 7.2** *The computation problems* Z-PARTITION, FLATTEN, *and* $s$-RANKING, *where* $s \in \{sum, lex\}$, *are complete for* FEXP *(resp., in* $\text{FP}_{\parallel}^{\text{NEXP}}$, $\text{FP}^{\text{NEXP}}$, *and* $\text{FP}^{\text{NEXP}}$*) when PB is defined over* $\mathcal{SHIF}(\mathbf{D})$ *(resp.,* $\mathcal{SHOIN}(\mathbf{D})$*).*

# 8 Tractable Special Case

In this section, we present a special case in which the problems CONSISTENCY, Z-PARTITION, FLATTEN, and $s$-RANKING, where $s \in \{sum, lex\}$, can all be solved in polynomial time. The main idea behind it is to restrict the class of concepts that may occur in $PB$ in such a way that the description logic satisfiability tests in Algorithms *z-partition*, *flatten*, *sum-ranking*, and *lex-ranking* can be done in polynomial time. Here, we take inspiration from the description logic *DL-Lite* [9] where deciding whether a knowledge base is satisfiable can be done in polynomial time.

We first recall knowledge bases in *DL-Lite* [9], which are a restricted class of description logic knowledge bases. Let $\mathbf{A}$, $\mathbf{R}_A$, and $\mathbf{I}$ be pairwise disjoint finite nonempty sets of atomic concepts, abstract roles, and individuals, respectively. A *basic concept in DL-Lite* is either an atomic concept from $\mathbf{A}$ or an exists restriction on roles of the form $\exists R.\top$ (abbreviated as $\exists R$), where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$. *Concepts in DL-Lite* are defined by induction as follows. Every basic concept in *DL-Lite* is a concept in *DL-Lite*. If $b$ is a basic concept in *DL-Lite*, and $\phi_1$ and $\phi_2$ are concepts in *DL-Lite*, then $\neg b$ and $\phi_1 \sqcap \phi_2$ are also concepts in *DL-Lite*. An *axiom in DL-Lite* is either (1) a concept inclusion axiom of the form $b \sqsubseteq \psi$, where $b$ is a basic concept in *DL-Lite* and $\phi$ is a concept in *DL-Lite*, or (2) a *functionality axiom* $(\mathsf{funct}\ R)$, where $R \in \mathbf{R}_A \cup \mathbf{R}_A^-$, or (3) a concept membership axiom $b(a)$, where $b$ is a basic concept in *DL-Lite* and $a \in \mathbf{I}$, or (4) a role membership axiom $R(a, c)$, where $R \in \mathbf{R}_A$ and $a, c \in \mathbf{I}$. A *knowledge base in DL-Lite* is a finite set of axioms in *DL-Lite*. We recall the following result from [9], which says that deciding whether a knowledge base in *DL-Lite* is satisfiable can be done in polynomial time.

**Theorem 8.1 (see [9])** *Given a knowledge base in DL-Lite KB, deciding whether KB is satisfiable can be done in polynomial time.*

We are now ready to define a similarly restricted class of conditional preference bases. A *literal in DL-Lite* is either a basic concept in *DL-Lite* $b$ or the negation of a basic concept in *DL-Lite* $\neg b$. A *conjunctive*

*concept in DL-Lite* is either $\bot$, or $\top$, or a conjunction of literals in *DL-Lite*. A conditional preference base $PB = (T, A, P)$ is defined *over DL-Lite* iff $T$ is a description logic knowledge base in *DL-Lite*, $A$ is a set of literals in *DL-Lite*, and $P$ is a set of conditional preferences of the form $(\psi|\phi)[s]$, where $\psi$ and $\phi$ are conjunctive concepts in *DL-Lite*. Given a conditional preference base over DL-Lite $PB = (T, A, P)$, we say that $P$ is *bounded* iff the input size of $P$ is bounded by a constant (which is a reasonable assumption in the application in literature search in Section 9).

The following theorem shows that for conditional preference bases over *DL-Lite*, the problems CONSISTENCY, Z-PARTITION, and FLATTEN can all be solved in polynomial time when $P$ is bounded. This result follows from Theorems 6.5 and 8.1, and the observation that here every description logic satisfiability test in Algorithms *z-partition* and *flatten* can be reduced to a constant number of description logic satisfiability tests on knowledge bases in *DL-Lite*.

**Theorem 8.2** *Given a conditional preference base over DL-Lite $PB = (T, A, P)$, where $P$ is bounded, (a) deciding whether $PB$ is consistent, (b) computing the $z$-partition of $PB$ (if it exists), and (c) computing a non-defeasible equivalent $PB^\star$ to $PB$ (if one exists) can all be done in polynomial time.*

The next theorem shows that for conditional preference bases over *DL-Lite*, also the problems $sum$- and $lex$-RANKING can both be solved in polynomial time when $P$ is bounded. Thus, for conditional preference bases over *DL-Lite*, the problems $sum$- and $lex$-RANKING have both a polynomial data complexity (where the input size of the whole conditional preference base $PB$ is bounded by a constant).

**Theorem 8.3** *Given a conditional preference base over DL-Lite $PB = (T, A, P)$, where $P$ is bounded, and a set of objects $\mathcal{O} \subseteq \mathcal{O}_\mathcal{C}$, where $\mathcal{C}$ is the set of all concepts that occur in $A$ and $P$, computing the object rankings $\kappa^{sum}$ and $\kappa^{lex}$ on $\mathcal{O}$ for $PB$ can be done in polynomial time (if they exist).*

# 9 Literature Search

In the previous sections, we have already described the application of conditional preference bases for a flexible user-defined ranking of the query results when searching product databases. In this section, we describe a further application of this approach in literature search, which shows in particular that the combination of description logics and variable-strength conditional preferences nicely allows for both expressing sophisticated search strategies and a flexible user-defined ranking of the query results, which reflects personal preferences and quality measures.

## 9.1 Background

A very important and time consuming task of researchers is finding publications. There exist a lot of possibilities to find relevant research publications over the internet. For instance, there are portals for research publications, portals for ejournals, special purpose search engines for researchers (for example, CiteSeer and Google Scholar), specialized databases, publication databases of institutions, and bibliographic online catalogues. It seems that there is a trend to more diversity and quality regarding online search engines. On the other hand, the "tremendous increase in the quantity and diversity of easily available research publications has exacerbated the problems of information overload for researchers attempting to keep abreast of new relevant research, especially in rapidly advancing fields" [6].

There are a lot of good search strategies for the task of finding relevant scientific publications. Bates [2] has identified the following six important information search strategies: (1) *Footnote chasing:* Following

up footnotes (that is, references) found in publications. This can be done in successive leaps. (2) *Citation searching:* Looking for publications that cite certain publications. (3) *Journal run:* Identification of a central journal in a research area and then looking up publications in relevant volumes. (4) *Area scanning:* Browsing resources that are physically collocated with resources that are regarded as relevant. A good example is a bookshelf in a library. In a digital library, one could exploit the classification of resources. (5) *Subject searches:* The usage of subject descriptors such as keywords to find relevant publications. (6) *Author searching:* To find other publications of an author, which may have a similar topic as a publication one already knows of.

It would be helpful if the above search strategies could be explicitly supported by the query languages of search engines. Each existing search engine (such as CiteSeer, Google Scholar, and Google), however, supports only some of these search strategies, but not all of them. Furthermore, there is currently no way to exploit the search strategies by the formulation of search queries. What is currently also not supported is the possibility to exploit relationships like citations or co-authorships by the formulation of queries. Finally, to date, search query languages of most web search engines have little expressive power for formulating semantic queries.

Another limitation of current search engines is that they provide the user only with restricted possibilities to influence the ranking of the returned query results. For example, Google is based on the PageRank ranking [8], which is calculated from the link structure between the objects, and thus somehow reflects the relative importance of the objects. However, especially in literature search, the user should be provided with more possibilities to explicitly influence the ranking of the query results, to express personal preferences and quality measures for the query results.

## 9.2 Literature Search via Conditional Preference Bases

We express each search query $Q$ by a conditional preference base $PB_Q = (T, A, P)$, where the background knowledge $T$ is informally described as follows (using self-explaining names). We assume the concepts *Publication*, *JournalPublication*, *ConfPublication*, *Person*, *Publicationmedium*, *Journal*, *Proceedings*, *Keyword*, *Event*, *Conference*, and *Workshop*, which are related by the concept inclusion axioms *JournalPublication* $\sqsubseteq$ *Publication*, *ConfPublication* $\sqsubseteq$ *Publication*, *Conference* $\sqsubseteq$ *Event*, *Workshop* $\sqsubseteq$ *Event*, *Journal* $\sqsubseteq$ *Publicationmedium*, and *Proceedings* $\sqsubseteq$ *Publicationmedium*. We assume the roles *Author* (relating *Publication* and *Person*), *Coauthor* (on *Person*), *Cite* (on *Publication*), *Publishedin* (relating *Publication* and *Publicationmedium*), *Keywords* (relating *Publication* and *Keyword*), *hasPublicationmedium* (relating *Event* and *Publicationmedium*), and *in_title* (relating *Publication* and string). Here, *in_title* relates every publication with all substrings of its title. Finally, the concept *Publication* has the attributes *year*, *title*, *publishedat*, and *type*.

We assume that the standard semantics of $PB_Q = (T, A, P)$ is given by its object ranking $\kappa^{sum}$, and we abbreviate $PB_Q = (T, A, P)$ by the conjunction $C_Q$ of all elements in $A \cup P$. For example, consider the following query $Q$: We are looking for publications with the keyword "Semantic Web"; furthermore, the publications should also contain the keywords "OWL" and "DAML+OIL" with the strengths 70 and 20, respectively. This query $Q$ is expressed by the following $PB_Q = (T, A, P)$:

$T$ as informally described above, $A = \{\exists$*Keywords*.{"Semantic Web"}$\}$, and
$P = \{(\exists$*Keywords*.{"OWL"})[70], ($\exists$*Keywords*.{"DAML+OIL"})[20]$\}$,

which is abbreviated by the following conjunction $C_Q$:

$$\exists \mathit{Keywords}.\{\text{“Semantic Web”}\} \sqcap \\ (\exists \mathit{Keywords}.\{\text{“OWL”}\})[70] \sqcap (\exists \mathit{Keywords}.\{\text{“DAML+OIL”}\})[20] .$$

This query also illustrates the use of (user-defined) strengths in simple conditional preferences: Publications that contain all the keywords have the rank $0$, whereas publications that have “Semantic Web” and “OWL” but not “DAML+OIL” as keywords have the rank $20$, since they falsify the second conditional preference, while those that have “Semantic Web” and “DAML+OIL” but not “OWL” as keywords have the rank $70$, since they falsify the first conditional preference. Finally, publications that have “Semantic Web” but not “DAML+OIL” and “OWL” as keywords have the rank $70 + 20 = 90$, since they falsify both conditional preferences.

We now provide some examples, which show in particular that search queries based on conditional preferences bases allow for both expressing nearly all the above-mentioned search strategies and a flexible user-defined ranking of the query results, which reflects personal preferences and quality measures for the query results. Note that further such examples are given in the ESWC-2006 abstract of this paper [37].

*Footnote chasing:* A highly relevant publication can be a very good starting point for the identification of further highly relevant publications. For example, we may be looking for all publications cited in “Weaving the Web” by “Tim Berners-Lee”:

$$\exists \mathit{Cite}^{-}.(\exists \mathit{title}.\{\text{“Weaving the Web”}\} \sqcap \exists \mathit{Author}.\{\text{“Tim Berners-Lee”}\}) .$$

*Citation searching:* Suppose we want to know which of the papers at ISWC-2003 were cited and how the topics of this conference evolved over time, that is, we may be looking for publications that cite publications of ISWC-2003:

$$\exists \mathit{Cite}.(\mathit{ConfPublication} \sqcap \exists \mathit{publishedat}.\{\text{“ISWC”}\} \sqcap \exists \mathit{year}.\{2003\}) .$$

*Journal run:* Suppose we are looking for conferences that are relevant to the topics “elearning” and “Semantic Web”, that is, we are looking for conferences that have publications with the keywords “elearning” and “Semantic Web”:

$$\mathit{Conference} \sqcap \exists \mathit{hasPublicationmedium}.\exists \mathit{Publishedin}^{-} . \\ (\exists \mathit{Keywords}.\{\text{“elearning”}\} \sqcap \exists \mathit{Keywords}.\{\text{“Semantic Web”}\}) .$$

*Subject searches:* Suppose next we are looking for publications that have the keyword “matching”, where (i) among the non-journal publications we prefer the ones that cite at least five journal publications that are cited at least eight times to those without this property with strength $40$, (ii) among the journal publications we prefer the ones that cite at least four publications that are cited at least seven times to those without this poperty with strength $50$, and (iii) we prefer journal publications to non-journal publications with strength $10$:

$$\exists \mathit{Keywords}.\{\text{“matching”}\} \sqcap \\ (\geq_5 \mathit{Cite}.(\geq_8 \mathit{Cite}^{-} \sqcap \mathit{JournalPublication}) \mid \neg \mathit{JournalPublication})[40] \sqcap \\ (\geq_4 \mathit{Cite}.(\geq_7 \mathit{Cite}^{-}) \mid \mathit{JournalPublication})[50] \sqcap (\mathit{JournalPublication})[10] .$$

Note that the object ranking of this query encodes a (user-defined) quality measure for the publications. More concretely, the query ranks the returned publications as follows. First, journal publications that cite at least four publications that are cited at least seven times have the rank $0$. Second, non-journal publications that cite at least five journal publications that are cited at least eight times have the rank $10$. Third, all the other journal and non-journal publications have the rank $50$.

*Author searching:* One possibility to judge the scientific famousness of a researcher is to find out how often and from whom the researcher is cited. For example, we may be looking for authors that cite publications of Ian Horrocks:

$$\exists \mathit{Author}^{-}.\exists \mathit{Cite}.\exists \mathit{Author}.\{\text{“Ian Horrocks”}\}\,.$$

# 10 Related Work

In this section, we give a brief overview on the area of default reasoning from conditional knowledge bases, and we discuss related work on combining formalisms for reasoning about uncertainty with description logics and ontologies.

## 10.1 Default Reasoning from Conditional Knowledge Bases

The literature contains several different proposals for default reasoning from conditional knowledge bases and extensive work on its desired properties. The core of these properties are the rationality postulates of System $P$ by Kraus et al. [33], which constitute a sound and complete axiom system for several classical model-theoretic entailment relations under uncertainty measures on worlds. They characterize classical model-theoretic entailment under preferential structures, infinitesimal probabilities, possibility measures [14], and world rankings [46, 23]. They also characterize an entailment relation based on conditional objects [13]. A survey of all these relationships is given in [3, 19]. Mainly to solve problems with irrelevant information, the notion of rational closure as a more adventurous notion of entailment was introduced by Lehmann [34]. It is in particular equivalent to entailment in System $Z$ by Pearl [41] (which is generalized to variable-strength defaults in System $Z^{+}$ by Goldszmidt and Pearl [21, 24]) and to the least specific possibility entailment by Benferhat et al. [4]. Another sophisticated quasi-probabilistic formalism for reasoning about variable-strength defaults is Weydert's System $JLZ$ [53]. Recently, also generalizations of many of the above approaches to probabilistic and fuzzy default reasoning have been proposed (see especially [35] resp. [15]).

## 10.2 Uncertainty Reasoning in Description Logics and Ontologies

Related formalisms to uncertainty reasoning in description logics and ontologies can roughly be divided into fuzzy description logics, probabilistic description logics, and probabilistic web ontology languages. In particular, Straccia [47] presents a fuzzy extension of the description logic $\mathcal{ALC}$, which is based on Zadeh's fuzzy logic, and which is directed towards applications in multimedia information retrieval. Recent works by Straccia also introduce a fuzzy description logic with concrete domains [48] and a fuzzy description logic for the Semantic Web [49]. Closely related to the latter is the work by Stoilos et al. [50], which combines the description logic $\mathcal{SHIN}$ with fuzzy set theory for the Semantic Web. As for probabilistic description logics, in [20], Giugno and Lukasiewicz present a probabilistic generalization of the expressive description logic $\mathcal{SHOQ}(\mathbf{D})$ that stands behind DAML+OIL, which is based on lexicographic probabilistic

reasoning. In earlier work, Heinsohn [25] and Jaeger [29] present probabilistic extensions to the description logic $\mathcal{ALC}$, which are essentially based on probabilistic reasoning in probabilistic logics. Koller et al. [32] present a probabilistic generalization of the CLASSIC description logic, which uses Bayesian networks as underlying probabilistic reasoning formalism. Finally, as for probabilistic web ontology languages, there are especially the works by Costa [10], Pool and Aikin [42], and Ding and Peng [12], which present probabilistic generalizations of the web ontology language OWL. In particular, Costa's work [10] is semantically based on multi-entity Bayesian networks, while [12] has a semantics in standard Bayesian networks. In closely related work, Fukushige [18] proposes a basic framework for representing probabilistic relationships in RDF. Finally, Nottelmann and Fuhr [39] present pDAML+OIL, which is a probabilistic generalization of the web ontology language DAML+OIL, along with a mapping to stratified probabilistic datalog.

# 11 Conclusion

We have introduced conditional preference bases as a means for ranking objects in ontologies. Conditional preference bases consist of a description logic knowledge base and a finite set of conditional preferences, which are statements of the form "generally, in the context $\phi$, property $\alpha$ is preferred over property $\neg\alpha$ with strength $s$". They are given a qualitative probabilistic formal semantics that is based on conditional knowledge bases. We have defined the consistency of conditional preference bases and shown how consistent conditional preference bases can be used for defining two object rankings, denoted $\kappa^{sum}$ and $\kappa^{lex}$, which evaluate the strengths of conditional preferences in an additive and a lexicographic way, respectively. Furthermore, we have provided algorithms for the main computational tasks for ranking objects under conditional preference bases, analyzed the complexity of these tasks, and described tractable special cases.

We have described two applications of the presented approach in product and literature search. In the former, it allows for a flexible user-defined ranking of the query results, which reflects personal preferences. In the latter, it allows for both expressing sophisticated search strategies and a flexible user-defined ranking of the query results, which reflects personal preferences and quality measures. More generally, query languages of current search engines are very restricted in their expressive power. There are scientific search engines on the web, however, that have valuable metadata about publications, authors, organizations, and scientific events. We have shown that conditional preference bases allow for a more powerful query language, which can exploit this metadata better than the current approaches do.

There are many other applications (especially in the Web), where ranking a selection of objects relative to personal preferences — as realized by our approach based on conditional preference bases — plays an important role, such as, e.g., product configuration, meeting scheduling, and searching for a partner, a holiday place, a hotel, travel means, a restaurant, an event, a house, a job, or an employee.

An interesting topic of future research is to explore whether the presented approach can also be used for personalization tasks and in recommender systems. Another interesting issue is an extension in such a way that the user-defined preference ranking on objects is combined with an importance ranking on objects (such as PageRank [8]). It would also be interesting to combine the presented formalism with probabilistic ontologies [32, 20, 10] to compute the ranking of incompletely specified objects. A first effort in the latter two directions is [38]. Finally, it would be interesting to generalize the formalism to the multi-agent framework.

# A Appendix: Proofs for Section 6

**Proof of Theorem 6.4.** The correctness of Algorithm *z-partition* for solving the computation problem z-PARTITION (and the decision problem CONSISTENCY) follows immediately from Theorem 4.5 (a). It is also not difficult to verify that Algorithms *sum-ranking* and *lex-ranking* correctly implement the declarative definitions of the object rankings $\kappa^{sum}$ and $\kappa^{lex}$ in equations (1) and (2), respectively.

It thus only remains to prove the correctness of Algorithm *flatten* for computing a non-defeasible equivalent of a consistent conditional preference base. In the sequel, let $PB = (T, A, P)$ be a consistent conditional preference base, and let $PB' = (T, A, P')$ be the output of Algorithm *flatten* on $PB$ as input. We now show that (i) $PB'$ is flat, (ii) $PB \mathrel{|\!\sim} p$ for all $p \in P'$, and (iii) $P' = \{(\alpha|\phi \sqcap \psi_p)[s] \mid p = (\alpha|\phi)[s] \in P\}$, where $\psi_p$ is a conjunction of negated bodies that occur in $P$.

We first prove that (i) holds. Let $D_i$ and $Q_i$, for every $i \in \{0, \ldots, n\}$, denote $D$ and $D \cup P_{i+1} \cup \cdots \cup P_n$ after the $i$-th iteration step of the **for**-loop in lines (5)–(13) of Algorithm *flatten*, respectively. We now prove that $(\star)$ for every $i \in \{0, \ldots, n\}$, every $p \in D_i$ is tolerated under $T \cup A$ by $Q_i = D_i \cup P_{i+1} \cup \cdots \cup P_n$. This then shows in particular that after the $n$-th iteration step, every $p \in D$ is tolerated under $T \cup A$ by $D$, which says that $(T, A, D) = PB'$ is flat, that is, (i) holds. We prove $(\star)$ by induction on the number of iteration steps $i \in \{0, \ldots, n\}$ as follows:

*Basis:* Let $i = 0$. Since $(P_0, \ldots, P_n)$ is the $z$-partition of $PB$, every $p \in D_0 = P_0$ is tolerated under $T \cup A$ by $Q_0 = D_0 \cup P_1 \cup \cdots \cup P_n = P$.

*Induction:* Let $i > 0$. Suppose that after the $i-1$-th iteration step, every $p \in D_{i-1}$ is tolerated under $T \cup A$ by $Q_{i-1} = D_{i-1} \cup P_i \cup \cdots \cup P_n$. We now show that after $i$-th iteration step, every $p \in D_i$ is tolerated under $T \cup A$ by $Q_i = D_i \cup P_{i+1} \cup \cdots \cup P_n$. Consider first any $p \in D_i \setminus P_i$, which either coincides with some $p' \in D_{i-1}$ or is obtained from some $p' \in D_{i-1}$ by replacing its body $\phi$ by some $\phi \sqcap \neg\psi_1 \sqcap \cdots \sqcap \neg\psi_l$. By the induction hypothesis, $p'$ is tolerated under $T \cup A$ by $Q_{i-1}$. That is, there exists an object $o$ that satisfies $T \cup A \cup Q_{i-1}$ and verifies $p'$. By the construction of the $\psi_j$'s, it follows that $o$ satisfies every $\neg\psi_j$ and all $q \in D_i \setminus (D_{i-1} \cup P_i)$. That is, $o$ satisfies $T \cup A \cup Q_i$ and verifies $p$. That is, $p$ is tolerated under $T \cup A$ by $Q_i$. Consider next any $p \in D_i \cap P_i = P_i$ with body $\phi$. Observe first that $p$ is tolerated under $T \cup A$ by $P_i \cup \cdots \cup P_n$, since $(P_0, \ldots, P_n)$ is the $z$-partition of $PB$. Thus, if every $q \in D_i \setminus P_i$ has $\neg\phi$ in its body, then $p$ is also tolerated under $T \cup A$ by $Q_i$. Otherwise, there exists an object $o$ that satisfies $T \cup A \cup Q_{i-1}$ and verifies $p$. Hence, $o$ satisfies also $T \cup A \cup Q_i$ and verifies $p$. That is, $p$ is tolerated under $T \cup A$ by $Q_i$.

As for (iii), by induction on the number of iteration steps $i \in \{0, \ldots, n\}$ of the **for**-loop in lines (5)–(13) of Algorithm *flatten*, it is easy to verify that $P' = \{(\alpha|\phi \sqcap \psi_p)[s] \mid p = (\alpha|\phi)[s] \in P\}$, where $\psi_p$ is a conjunction of negated bodies in $P$.

Finally, we prove that (ii) holds, that is, $PB \mathrel{|\!\sim} p$ for every $p \in P'$. Observe first that every $p \in P$ is a $z$-consequence of $PB$. This is due to the fact that $(P_0, \ldots, P_n)$ is the $z$-partition of $PB$, and thus every $p = (\alpha|\phi)[s] \in P_i$ is tolerated under $T \cup A$ by $P_i \cup \cdots \cup P_n$, for every $i \in \{0, \ldots, n\}$. Hence, there exists an object $o$ that satisfies $\phi$, $T \cup A$, and $P_i \cup \cdots \cup P_n$, which implies that $PB \mathrel{|\!\sim} p$. It thus only remains to prove that $PB \mathrel{|\!\sim} p'$ for every $p' \in P' \setminus P$. By (iii), $p'$ is obtained from some $p \in P$ by replacing its body $\phi$ by some $\phi \sqcap \psi_p$, where $\psi_p$ is a conjunction of negated bodies in $P$. Notice then that $\psi_p$ is even a conjunction of negated bodies in $P_{i+1} \cup \cdots \cup P_n$, where $i \in \{0, \ldots, n\}$ such that $p \in P_i$. Since $(P_0, \ldots, P_n)$ is the $z$-partition of $PB$, it follows that $p$ is tolerated under $T \cup A$ by $P_i \cup \cdots \cup P_n$, and every $q \in P_{i+1} \cup \cdots \cup P_n$ is *not* tolerated under $T \cup A$ by $P_i \cup \cdots \cup P_n$. Hence, there exists an object $o$ that satisfies $\phi \sqcap \psi_p$, $T \cup A$, and $P_i \cup \cdots \cup P_n$, which implies that $PB \mathrel{|\!\sim} p'$. $\square$

**Proof of Theorem 6.5.** As for Algorithm *z-partition*, in line (1), we decide one time whether $T \cup \{\alpha(i) \mid \alpha \in A\}$ is satisfiable, and in the iteration in lines (5)–(9), we decide at most $O(|P|^2)$ times whether $T \cup \{\alpha(i) \mid \alpha \in A\} \cup \{\beta(i)\} \cup \{(\psi \sqcup \neg\phi)(i) \mid (\psi|\phi)[s] \in H\}$ is satisfiable, where $(\gamma|\beta)[t] = p$. In summary, Algorithm *z-partition* can be done in $O(|P|^2)$ description logic satisfiability tests.

As for Algorithm *flatten*, line (1) can be done in $O(|P|^2)$ description logic satisfiability tests, as argued above, and the iteration in lines (5)–(13) can be done in $\sum_{i=1}^{n} |P_i| \cdot (\sum_{j=0}^{i-1} |P_j|) \leq |P|^2$ description logic satisfiability tests. In summary, Algorithm *z-flatten* can also be done in $O(|P|^2)$ description logic satisfiability tests.

As for Algorithms *sum-ranking* and *lex-ranking*, line (1) can be done in $O(|P|^2)$ description logic satisfiability tests, as argued above, and in line (3), we decide $|\mathcal{O}|$ times whether $T \cup \{\alpha(i) \mid \alpha \in A\} \cup \{\beta(i) \mid \beta \in o\} \cup \{\neg\beta(i) \mid \beta \in \mathcal{C} \setminus o\}$ is satisfiable. In summary, Algorithms *sum-ranking* and *lex-ranking* can be both done in $O(|P|^2 + |\mathcal{O}|)$ description logic satisfiability tests. □

# B Appendix: Proofs for Section 7

**Proof of Theorem 7.1.** We first prove that CONSISTENCY is in EXP (resp., NEXP). For $PB$ over $\mathcal{SHIF}(\mathbf{D})$, membership in EXP follows immediately from Theorem 6.5 and the membership in EXP of deciding description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$. For $PB$ over $\mathcal{SHOIN}(\mathbf{D})$, by Theorem 4.5 (b), $PB = (T, A, P)$ is consistent iff either (i) $P = \emptyset$ and $T \cup A$ is satisfiable, or (ii) $P \neq \emptyset$ and there exists an ordered partition $(P_0, \ldots, P_k)$ of $P$ such that for every $i$, $0 \leq i \leq k$, each $p \in P_i$ is tolerated under $T$ and $A$ by $\bigcup_{j=i}^{k} P_j$. Since deciding description logic satisfiability in $\mathcal{SHOIN}(\mathbf{D})$ is in NEXP, (i) deciding whether $T \cup \{\alpha(o) \mid \alpha \in A\}$ is satisfiable is in NEXP, and (ii) guessing an ordered partition $(P_0, \ldots, P_k)$ of $P$ and verifying that $T \cup \{\alpha(o) \mid \alpha \in A\} \cup \{\beta(o)\} \cup \{(\psi \sqcup \neg\phi)(o) \mid (\psi|\phi)[s] \in \bigcup_{j=i}^{k} P_j\}$ is satisfiable, for every $i \in \{0, \ldots, k\}$ and every $p = (\gamma|\beta)[t] \in P_i$, is in NEXP. In summary, deciding whether a given $PB$ over $\mathcal{SHOIN}(\mathbf{D})$ is consistent is in NEXP.

We next prove that CONSISTENCY is hard for EXP (resp., NEXP). Since a description logic knowledge base $T$ in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$) is satisfiable iff the conditional preference base $PB = (T, \emptyset, \emptyset)$ is consistent, hardness for EXP (resp., NEXP) follows immediately from the hardness for EXP (resp., NEXP) of deciding description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$ (resp., $\mathcal{SHOIN}(\mathbf{D})$). □

**Proof of Theorem 7.2.** For $PB$ over $\mathcal{SHIF}(\mathbf{D})$, membership in FEXP of Z-PARTITION, FLATTEN, and $s$-RANKING follows from Theorem 6.5 and the membership in EXP of description logic satisfiability in $\mathcal{SHIF}(\mathbf{D})$. Hardness for FEXP follows from the hardness for EXP of CONSISTENCY for $PB$ over $\mathcal{SHIF}(\mathbf{D})$.

We next consider the case where $PB$ is defined over $\mathcal{SHOIN}(\mathbf{D})$. As for the problem Z-PARTITION, observe that the $z$-partition $(P_0, \ldots, P_k)$ of a consistent $PB = (T, A, P)$ is the unique ordered partition $(P_0, \ldots, P_k)$ of $P$ such that (i) for every $i \in \{0, \ldots, k\}$, every $p \in P_i$ is tolerated under $T$ and $A$ by $\bigcup_{j=i}^{k} P_j$, and (ii) $\sum_{p \in P} r(p)$ (called the *weight* of $(P_0, \ldots, P_k)$) is minimal, where every $p \in P_j$ is assigned the value $j$ under $r$, for every $j \in \{0, \ldots, k\}$. As argued in the proof of Theorem 7.1, guessing an ordered partition $(P_0, \ldots, P_k)$ of $P$ and verifying that (i) holds is in NEXP. Thus, for each triple $(i, p, j)$, where $i \in \{0, 1, \ldots, |P|(|P|-1)/2\}$, $p \in P$, and $j \in \{0, 1, \ldots, |P|-1\}$, deciding whether $P$ has an ordered partition $(P_0, \ldots, P_k)$ that satisfies (i), has weight $i$, and such that $p \in P_j$ is in NEXP. It is now easy to see that for the $z$-partition $(P_0, \ldots, P_k)$ of a consistent $PB$, it holds that $p \in P_j$ iff the query for $(i, p, j)$ is a query with smallest value for $i$ that is answered "Yes", for every $j \in \{0, \ldots, k\}$. Furthermore, $PB$ is inconsistent

iff all queries are answered "No". In summary, Z-PARTITION is in $\mathrm{FP}_{\|}^{\mathrm{NEXP}}$. Finally, membership in $\mathrm{FP}^{\mathrm{NEXP}}$ of FLATTEN and $s$-RANKING follows from Theorem 6.5 and the membership in NEXP of description logic satisfiability in $\mathcal{SHOIN}(\mathbf{D})$. □

# C Appendix: Proofs for Section 8

**Proof of Theorem 8.2.** (a), (b) Algorithm *z-partition* decides whether $PB$ is consistent and computes the $z$-partition of $PB$ (if it exists). By Theorem 6.5, it can be done in $O(|P|^2)$ description logic satisfiability tests. More concretely, in the iteration in lines (5)–(9), we decide at most $O(|P|^2)$ times whether $T \cup \{\alpha(i) \mid \alpha \in A\} \cup \{\beta(i)\} \cup \{(\psi \sqcup \neg\phi)(i) \mid (\psi|\phi)[s] \in H\}$ (where $H \subseteq P$) is satisfiable, where $(\gamma|\beta)[t] = p$. Each of these satisfiability tests is equivalent to $(1+k)^{|H|}$ satisfiability tests on knowledge bases of the form $KB = T \cup \{b(i) \mid b \in B^+\} \cup \{\neg b(i) \mid b \in B^-\}$, where $k$ is the maximal number of literals that occur in the bodies of conditional preferences in $H$, and $B^+$ and $B^-$ are sets of basic concepts in *DL-Lite*. Notice then that (i) $KB$ is satisfiable iff $KB' = T \cup \{b(i) \mid b \in B^+\} \cup \{b^\neg(i) \mid b \in B^-\} \cup \{b^\neg \sqsubseteq \neg b \mid b \in B^-\}$ is satisfiable, where the $b^\neg$'s are new basic concepts in *DL-Lite* that do not occur in $KB$, and (ii) $KB'$ is a knowledge base in *DL-Lite*. By Theorem 8.1, deciding whether $KB'$ is satisfiable can be done in polynomial time. Since $k$ and $|H|$ are bounded by a constant, each satisfiability test in Algorithm *z-partition* is possible in polynomial time, and thus the whole algorithm is possible in polynomial time.

(c) Algorithm *flatten* computes a non-defeasible equivalent $PB^\star$ to $PB$ (if one exists). By (a), lines (1) and (2) can be done in polynomial time. As argued in the proof of Theorem 6.5, the iteration in lines (5)–(13) can be done in $O(|P|^2)$ description logic satisfiability tests. Since $P$ is bounded, and every rewritten conditional preference contains in its body at most $|P| - 1$ negated bodies of other conditional preferences from $P$, each of these satisfiability tests is equivalent to a constant number of satisfiability tests on knowledge bases of the form $KB = T \cup \{b(i) \mid b \in B^+\} \cup \{\neg b(i) \mid b \in B^-\}$, where $B^+$ and $B^-$ are sets of basic concepts in *DL-Lite*. As argued in (a), each of the latter satisfiability tests is equivalent to a satisfiability test on a knowledge base in *DL-Lite*, which can in turn be done in polynomial time, by Theorem 8.1. Thus, every of the $O(|P|^2)$ satisfiability tests can be done in polynomial time. In summary, Algorithm *flatten* can be done in polynomial time. □

**Proof of Theorem 8.3.** Algorithms *sum-ranking* and *lex-ranking* compute the object rankings $\kappa^{sum}$ and $\kappa^{lex}$ (if they exist), respectively. By Theorem 8.2 (c), lines (1) and (2) can be done in polynomial time. As argued in the proof of Theorem 6.5, in line (3), we decide $|\mathcal{O}|$ times whether $T \cup \{\alpha(i) \mid \alpha \in A\} \cup \{\beta(i) \mid \beta \in o\} \cup \{\neg\beta(i) \mid \beta \in \mathcal{C} \setminus o\}$ is satisfiable. Since $P$ is bounded, and $\mathcal{C}$ contains only concepts from $A$ and $P$, each of these satisfiability tests is equivalent to a constant number of satisfiability tests on knowledge bases $KB = T \cup \{b(i) \mid b \in B^+\} \cup \{\neg b(i) \mid b \in B^-\}$, where $B^+$ and $B^-$ are sets of basic concepts in *DL-Lite*. As argued in the proof of Theorem 8.2 (a), each of the latter tests is equivalent to a satisfiability test on a knowledge base in *DL-Lite*, which can in turn be done in polynomial time, by Theorem 8.1. Hence, each satisfiability test in Algorithms *sum-* and *lex-ranking* is possible in polynomial time, and since all their other computations can also be done in polynomial time, the whole algorithms can be done in polynomial time. □

# References

[1] E. W. Adams. *The Logic of Conditionals*, volume 86 of *Synthese Library*. D. Reidel, Dordrecht, Netherlands, 1975.

[2] M. Bates. The design of browsing and berrypicking techniques for the on-line search interface. *Online Review*, 13(5):407–431, 1989.

[3] S. Benferhat, D. Dubois, and H. Prade. Nonmonotonic reasoning, conditional objects and possibility theory. *Artif. Intell.*, 92(1–2):259–276, 1997.

[4] S. Benferhat, D. Dubois, and H. Prade. Representing default rules in possibilistic logic. In *Proceedings KR-1992*, pp. 673–684. Morgan Kaufmann, 1992.

[5] T. Berners-Lee. *Weaving the Web*. Harper, San Francisco, 1999.

[6] K. D. Bollacker, S. Lawrence, and C. L. Giles. Discovering relevant scientific literature on the web. *IEEE Intelligent Systems*, 15(2):42–47, 2000.

[7] 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.

[8] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Computer Networks*, 30(1–7):107–117, 1998.

[9] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. *DL-Lite*: Tractable description logics for ontologies. In *Proceedings AAAI-2005*, pp. 602–607. AAAI Press / MIT Press, 2005.

[10] P. C. G. da Costa. Bayesian semantics for the Semantic Web. Doctoral Dissertation, George Mason University, Fairfax, VA, USA, 2005.

[11] T. Di Noia, E. Di Sciascio, F. M. Donini, and M. Mongiello. Abductive matchmaking using description logics. In *Proceedings IJCAI-2003*, pp. 337–342. Morgan Kaufmann, 2003.

[12] Z. Ding and Y. Peng. A Probabilistic extension to ontology language OWL. In *Proceedings HICSS-2004*, 2004.

[13] D. Dubois and H. Prade. Conditional objects as nonmonotonic consequence relationships. *IEEE Trans. Syst. Man Cybern.*, 24(12):1724–1740, 1994.

[14] D. Dubois and H. Prade. Possibilistic logic, preferential models, non-monotonicity and related issues. In *Proceedings IJCAI-1991*, pp. 419–424. Morgan Kaufmann, 1991.

[15] F. Dupin de Saint-Cyr and H. Prade. Possibilistic handling of uncertain default rules with applications to persistence modeling and fuzzy default reasoning. In *Proceedings KR-2006*, pp. 440–451. AAAI Press, 2006.

[16] T. Eiter, T. Lukasiewicz, R. Schindlauer, and H. Tompits. Combining answer set programming with description logics for the Semantic Web. In *Proceedings KR-2004*, pp. 141–151. AAAI Press, 2004.

[17] D. Fensel, W. Wahlster, H. Lieberman, and J. Hendler, editors. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2002.

[18] Y. Fukushige. Representing probabilistic knowledge in the Semantic Web. In *Proceedings of the W3C Workshop on Semantic Web for Life Sciences*, Cambridge, MA, USA, 2004.

[19] D. M. Gabbay and P. Smets, editors. *Handbook on Defeasible Reasoning and Uncertainty Management Systems*. Kluwer Academic, Dordrecht, Netherlands, 1998.

[20] R. Giugno and T. Lukasiewicz. P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the Semantic Web. In *Proceedings JELIA-2002*, pp. 86–97, *LNCS* 2424, Springer, 2002.

[21] M. Goldszmidt and J. Pearl. System $Z^+$: A formalism for reasoning with variable strength defaults. In *Proceedings AAAI-1991*, pp. 399–404. AAAI Press / MIT Press, 1991.

[22] M. Goldszmidt and J. Pearl. On the consistency of defeasible databases. *Artif. Intell.*, 52(2):121–149, 1991.

[23] M. Goldszmidt and J. Pearl. Rank-based systems: A simple approach to belief revision, belief update and reasoning about evidence and actions. In *Proceedings KR-1992*, pp. 661–672. Morgan Kaufmann, 1992.

[24] M. Goldszmidt and J. Pearl. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *Artif. Intell.*, 84(1–2):57–112, 1996.

[25] J. Heinsohn. Probabilistic description logics. In *Proceedings UAI-1994*, pp. 311–318. Morgan Kaufmann, 1994.

[26] I. Horrocks and P. F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. *J. Web Semantics*, 1(4):345–357, 2004.

[27] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. Web Semantics*, 1(1):7–26, 2003.

[28] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proceedings LPAR-1999*, pp. 161–180. *LNCS* 1705, Springer, 1999.

[29] M. Jaeger. Probabilistic reasoning in terminological logics. In *Proceedings KR-1994*, pp. 305–316. Morgan Kaufmann, 1994.

[30] B. Jenner and J. Toran. Computing functions with parallel queries to NP. *Theor. Comput. Sci.*, 141:175–193, 1995.

[31] D. S. Johnson. A catalog of complexity classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A, chapter 2, pp. 67–161. MIT Press, 1990.

[32] D. Koller, A. Levy, and A. Pfeffer. P-CLASSIC: A tractable probabilistic description logic. In *Proceedings AAAI-1997*, pp. 390–397. AAAI Press / MIT Press, 1997.

[33] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artif. Intell.*, 14(1):167–207, 1990.

[34] D. Lehmann and M. Magidor. What does a conditional knowledge base entail? *Artif. Intell.*, 55(1):1–60, 1992.

[35] T. Lukasiewicz. Weak nonmonotonic probabilistic logics. *Artif. Intell.*, 168(1–2):119–161, 2005.

[36] T. Lukasiewicz and J. Schellhase. Variable-strength conditional preferences for match- making in description logics. In *Proceedings KR-2006*, pp. 164–174. AAAI Press, 2006.

[37] T. Lukasiewicz and J. Schellhase. Variable-strength conditional preferences for ranking objects in ontologies. In *Proceedings ESWC-2006*, pp. 288–302, *LNCS* 4011, Springer, 2006.

[38] T. Lukasiewicz and J. Schellhase. Preferences, links, and probabilities for ranking objects in ontologies. In *Proceedings URSW-2006*, *CEUR Workshop Proceedings* 218, CEUR-WS.org, 2006.

[39] H. Nottelmann and N. Fuhr. pDAML+OIL: A probabilistic extension to DAML+OIL based on probabilistic Datalog. In *Proceedings IPMU-2004*, 2004.

[40] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, Reading, MA, 1994.

[41] J. Pearl. System $Z$: A natural ordering of defaults with tractable applications to default reasoning. In *Proceedings TARK-1990*, pp. 121–135. Morgan Kaufmann, 1990.

[42] M. Pool and J. Aikin. KEEPER and Protégé: An elicitation environment for Bayesian inference tools. In *Proceedings of the Workshop on Protégé and Reasoning* held at the *7th International Protégé Conference*, 2004.

[43] D. Poole and C. Smyth. Type uncertainty in ontologically-grounded qualitative probabilistic matching. In *Proceedings ECSQARU-2005*, pp. 763–774, *LNCS* 3571, Springer, 2005.

[44] C. Smyth and D. Poole. Qualitative probabilistic matching with hierarchical descriptions. In *Proceedings KR-2004*, pp. 479–487. AAAI Press, 2004.

[45] A. Selman. A taxonomy of complexity classes of functions. *J. Computer and System Sciences*, 48:357–381, 1994.

[46] W. Spohn. Ordinal conditional functions: A dynamic theory of epistemic states. In W. Harper and B. Skyrms, editors, *Causation in Decision, Belief Change, and Statistics*, volume 2, pp. 105–134. Reidel, Dordrecht, Netherlands, 1988.

[47] U. Straccia. Reasoning within fuzzy description logics. *J. Artif. Intell. Res.*, 14:137–166, 2001.

[48] U. Straccia. Description logics with fuzzy concrete domains. In *Proceedings UAI-2005*, pp. 559–567. AUAI Press, 2005.

[49] U. Straccia. Towards a fuzzy description logic for the Semantic Web (preliminary report). In *Proceedings ESWC-2005*, pp. 167–181, *LNCS* 3532, Springer, 2005.

[50] G. Stoilos, G. B. Stamou, V. Tzouvaras, J. Z. Pan, and I. Horrocks. The fuzzy description logic $f$-$\mathcal{SHIN}$. In *Proceedings URSW-2005*, pp. 67–76, 2005.

[51] S.-W. Tan and J. Pearl. Specification and evaluation of preferences under uncertainty. In *Proceedings KR-1994*, pp. 530–539. Morgan Kaufmann, 1994.

[52] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD Thesis, RWTH Aachen, Germany, 2001.

[53] E. Weydert. System JLZ — Rational default reasoning by minimal ranking constructions. *J. Applied Logic* 1(3–4):273–308, 2003.

[54] W3C. OWL web ontology language overview. W3C Recommendation (10 Feb. 2004). `http://www.w3.org/TR/2004/REC-owl-features-20040210/`.