Reconciling Description Logics and Rules

BORIS MOTIK
University of Oxford
and
RICCARDO ROSATI
Sapienza Università di Roma

Description logics (DLs) and rules are formalisms that emphasize different aspects of knowledge representation: whereas DLs are focused on specifying and reasoning about conceptual knowledge, rules are focused on nonmonotonic inference. Many applications, however, require features of both DLs and rules. Developing a formalism that integrates DLs and rules would be a natural outcome of a large body of research in knowledge representation and reasoning of the last two decades; however, achieving this goal is very challenging and the approaches proposed thus far have not fully reached it. In this paper, we present a hybrid formalism of *MKNF$^+$ knowledge bases*, which integrates DLs and rules in a coherent semantic framework. Achieving seamless integration is nontrivial, since DLs use an open-world assumption, while the rules are based on a closed-world assumption. We overcome this discrepancy by basing the semantics of our formalism on the logic of minimal knowledge and negation as failure (MKNF) by Lifschitz. We present several algorithms for reasoning with MKNF$^+$ knowledge bases, each suitable to different kinds of rules, and establish tight complexity bounds.

Categories and Subject Descriptors: F.4.1 [[**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Computational logic,Mechanical theorem proving*; I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving—*Answer/reason extraction*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Description Logics, Answer Set Programming, Combined Complexity, Data Complexity

1. INTRODUCTION

Background

The main research goal in the field of Knowledge Representation and Reasoning (KR&R) [Levesque 1984; Hayes 1979] is to define languages (or formalisms) for describing knowledge in a precise way. The formal description of such knowledge is usually called a *knowledge base* (KB), and it is given a *formal semantics* that

The results presented in this article are an extension of the results that were published in preliminary form in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, Hyderabad, India, January 6–12, 2007, and in *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, Athens, GA, USA, November 5–9, 2006.

Authors' addresses: B. Motik, Computing Laboratory, University of Oxford, Parks Road, Oxford, OX1 3QD, United Kingdom, e-mail: boris.motik@comlab.ox.ac.uk; R. Rosati, Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, Via Ariosto 25, 00185 Rome, Italy, e-mail: rosati@dis.uniroma1.it.

determines the *logical consequences* of the knowledge base. These consequences can often be derived automatically by means of various *reasoning* algorithms.

Numerous KR&R languages developed thus far can be classified into one of the following two main families, which have been developed in the past 30 years largely independently from each other.

(1) The first family encompasses formalisms that try to formally reconstruct popular but informal approaches, such as frame-based systems and semantic networks [Fikes and Kehler 1985; Hayes 1979; Woods and Schmolze 1992]. *Description Logics* (DLs) are prominent formalisms from this family. Most DLs can be seen as fragments of first-order logic that provide useful modeling constructs while keeping the basic reasoning problems decidable and sometimes even tractable [Baader et al. 2007]. The fundamental building blocks of DL knowledge bases are concepts (which can be seen as unary predicates), roles (which can be seen as binary predicates), and individuals (which can be seen as constants), and the relationships between concepts, roles, and individuals can be precisely described using axioms. The semantics of DL KBs is based on the *open-world assumption* of classical logic—that is, a DL knowledge base can be seen as an incomplete description of the world. DLs are strongly related to languages for expressing dependencies in relational databases [Calvanese et al. 1998]. Decidability of DLs is achieved by carefully restricting the syntactic form of the allowed axioms.

(2) The second family is centered on the idea of modeling knowledge as *rules.* *Logic programming* provides the fundamental rule-based formalism, and it is closely related to various approaches to *nonmonotonic reasoning* [Antoniou 1997; Marek and Truszczyński 1993]. Rules are typically implications, and they are often allowed to contain negation. Although they can be given a standard first-order semantics, rule-based KBs usually employ a semantics based on a suitable form of *closed-world assumption.* Such a semantics allows the rules to *introspect* the KB and derive conclusions based on the *absence* of information in the KB. Thus, rule-based formalisms are typically *nonmonotonic*: adding new information may invalidate previously derived conclusions.

The Problem: Integrating DLs and Rules

Both DLs and rules exhibit certain shortcomings that can be compensated by the features of the other formalism. Thus, complex knowledge representation problems often require features found in both DLs and rules.

The shortcomings of DLs are twofold. First, the syntactic restrictions used to ensure decidability of reasoning prevent DLs from axiomatizing non-tree-like relationships [Vardi 1996], such as "an uncle is the brother of one's father,"[1] which may prevent the derivation of certain desired consequences. Second, as most DLs are fragments of first-order logic, such DLs do not not provide for introspection and nonmonotonic inference and thus cannot express the following types of knowledge:

[1]The DL $\mathcal{SROIQ}$ [Kutz et al. 2006] provides role composition axioms, which can be used to address some, but by no means all use cases.

—they cannot axiomatize database-like integrity constraints (ICs) [Reiter 1992] (e.g., one cannot express a check whether the information record for each person occurring in a knowledge base explicitly contains a social security number);

—they cannot model closed-world reasoning (e.g., one cannot model the fact that someone is innocent unless proven guilty); and

—they cannot model exceptions [Reiter 1980] (e.g., one cannot express that the heart is located on the left in most people, but in some people it is on the right).

The shortcomings of rules are mainly due to the fact that rules typically cannot reason with unbounded or infinite domains and thus cannot represent many types of incomplete information. For example, in rule-based formalisms one typically cannot say that "every person has a father and a mother who are both persons" without listing all the parents explicitly. As a consequence, rules are typically not used for reasoning about conceptual schemata but are mainly applied to data-centric problems such as query answering.

The potential benefits of integrating DLs with rules have been recognized early on, and a significant body of research has been devoted to solving this problem. Initial studies, however, have shown that integrating DLs with first-order rules easily leads to undecidability of the basic reasoning problems [Levy and Rousset 1998] even if both the DL and the rule formalisms alone are decidable. Moreover, the differences in the semantic foundations of DLs and nonmonotonic rules present serious hurdles to defining a coherent semantics for the unified formalism. Thus, the development of a formalism that integrates DLs and rules is a very challenging goal that has eluded the KR&R community for some time.

Early Approaches

The idea of adding rules to structured knowledge representation systems dates back to the 80's and the early DL systems such as CLASSIC [Patel-Schneider et al. 1991], LOOM [MacGregor 1991], and CLASP [Yen et al. 1991]. These systems, however, were typically not based on a coherent semantic framework, and the integration of DLs and rules was procedural. In the 90's, attempts were made to study various integration approaches in a formally more rigorous way. The first approaches in this direction studied the problem of integrating DLs with datalog rules [Levy and Rousset 1998; Cadoli et al. 1997; Donini et al. 1998]. These works typically study *hybrid* knowledge bases consisting of a DL knowledge base (often called the *structural component*) and a set of datalog rules (often called a *program*). The interaction between the two components is obtained by allowing variables in datalog rules to range over the extensions of concepts and roles in the DL knowledge base.

Recent Approaches

A renewed interest in the integration of DLs and rules has been spurred by the research in *ontologies* and the advent of the *Semantic Web*. DLs are playing a central role in this field, as the Web Ontology Language (OWL)—the ontology modeling language standardized by the World Wide Web Consortium (W3C)—is based on DLs. Formal semantics and the availability of efficient and provably correct reasoning tools, such as Pellet [Parsia and Sirin 2004], FaCT++ [Tsarkov and Horrocks 2006], and RACER [Haarslev and Möller 2001], have made the OWL

DL variant of OWL the language of choice for practical applications in fields as diverse as biology [Sidhu et al. 2005], medicine [Golbreich et al. 2006], geography [Goodwin 2005], astronomy [Derriere et al. 2006], agriculture [Soergel et al. 2004], and defense [Lacy et al. 2005]. This extensive practical experience has confirmed the benefits of using DLs in knowledge representation, but it has also highlighted the already mentioned limitations of DLs.

In response, several approaches to integration of OWL and rules have been proposed recently. The Semantic Web Rule Language (SWRL) [Horrocks et al. 2005] is an extension of OWL with first-order rules that significantly increases the relational expressivity of OWL. SWRL is trivially undecidable; however, various syntactic restrictions on the rules can be used to regain decidability [Levy and Rousset 1998; Motik et al. 2005; Rosati 2006]. Several approaches to integrating OWL with nonmonotonic rules have been proposed as well [Eiter et al. 2008; Lukasiewicz 2007; Rosati 2005; de Bruijn et al. 2007; de Bruijn et al. 2007].

As mentioned above, integrating OWL with nonmonotonic rules is a nontrivial task, since the semantics of the two formalisms are quite different. Because of these difficulties, Kifer et al. [2005] claimed that true rule-based formalisms are intrinsically incompatible with OWL. As a consequence, they proposed to change the layering architecture of the Semantic Web: instead of building rules on top of OWL, rules and OWL should coexist side-by-side with semantic interoperability grounded in Description Logic Programs (DLP)—a straightforward intersection of OWL and first-order rules [Grosof et al. 2003]. Furthermore, OWL-Flight [de Bruijn et al. 2005], the Web Service Modeling Language [de Bruijn et al. 2006], and F-Logic [Kifer et al. 1995] were proposed as ontology languages based on the rule paradigm. Horrocks et al. [2005] criticized these approaches on the grounds that separating rules from OWL would essentially create two Semantic Webs with little or no semantic interoperability.

Limitations of Existing Approaches

In order to best use the advantages of DLs and rules, we argue that a formalism combining them should satisfy the following important criteria.

—*Faithfulness*: The integration between DLs and rules should preserve the semantics of both formalisms—that is, the semantics of a hybrid KB in which one component is empty should be the same as the semantics of the other component. In other words, the addition of rules to a DL should not change the semantics of the DL and vice versa.

—*Tightness*: Rules should not be layered on top of a DL or vice versa; rather, the integration between a DL and rules should be tight in the sense that both the DL and the rule component should be able to contribute to the consequences of the other component.

—*Flexibility*: The hybrid formalism should be flexible and allow one to view the same predicate under both open- and closed-world interpretation. This allows the rules to enrich a DL with nonmonotonic consequences, and a DL to enrich the rules with the capabilities of taxonomic reasoning.

—*Decidability*: To obtain a practically useful formalism that can be used in applications such as the Semantic Web, the hybrid formalism should be at least decidable, and preferably of low worst-case complexity.

As we will discuss in depth in Section 7, none of the approaches proposed so far possesses all of these qualities. The approaches that focus on the integration of DLs with first-order rules (such as those by Levy and Rousset [1998], Donini et al. [1998], and Horrocks et al. [2005]) do not extend DLs with capabilities of nonmonotonic reasoning. Furthermore, the approaches that consider nonmonotonic rules are inflexible [Rosati 2005; 2006] or not tight [Eiter et al. 2008], they use a restricted notion of faithfulness [Lukasiewicz 2007], or their computational properties are unknown [de Bruijn et al. 2007].

Our Proposal and Contribution

In this paper we propose a formalism of *MKNF$^+$ knowledge bases*, which allows for a faithful, tight, and flexible integration of DLs and *answer set programming* (ASP) [Gelfond and Lifschitz 1991]—a well-known and popular rule-based formalism with a semantics grounded in circumscription [McCarthy 1980]. Our formalism is based on the logic of minimal knowledge and negation as failure (MKNF)—a formalism developed by Lifschitz [1991] with the goal of unifying most existing approaches to nonmonotonic reasoning. We believe that MKNF provides a natural framework for overcoming the mentioned technical differences in the semantics of DLs and ASP. Our contributions can be summarized as follows:

—We define a syntax and a model-theoretic semantics for MKNF$^+$ knowledge bases. Our semantics employs the *standard name assumption*—a modification of the semantics of MKNF that is needed in order to make the semantics of the hybrid formalism intuitive. We show that such a semantics indeed provides for a faithful, tight, and flexible integration of DLs and ASP.

—We thoroughly study the computational properties of the basic reasoning task for MKNF$^+$ knowledge bases—that is, the problem of checking whether a ground atom is true in all models of the knowledge base. Our results can be summarized as follows:
  - We prove that reasoning with MKNF$^+$ knowledge bases is undecidable even if standard restrictions on the DL knowledge base and the rules are employed. Therefore, to ensure decidability, we apply the well-known *DL-safety* restriction and define a suitable notion of knowledge base admissibility.
  - We present reasoning algorithms for the cases where ASP rules are unrestricted, positive, nondisjunctive positive, stratified, and nondisjunctive.
  - We establish precise combined and data complexity bounds for the basic reasoning problem. Interestingly, it turns out that our approach to the integration of DLs and ASP is in most cases computationally "optimal"—that is, the complexity of reasoning in MKNF$^+$ knowledge bases is computationally not worse than reasoning in the corresponding DL or answer set program alone.

—We show that our formalism is quite general and that it can capture many of the existing combinations of DLs and rules, as well as several existing approaches to extending DLs with nonmonotonic features.

We focus in this paper on description logics mainly due to the context by which our work is motivated; however, our approach can be used to integrate any first-order fragment $\mathcal{DL}$ with ASP. The semantics of our hybrid formalism does not rely on the fact that $\mathcal{DL}$ is a description logic. Furthermore, we clearly identify the types of entailments that must be decidable in $\mathcal{DL}$ in order to obtain a decidable hybrid formalism. The complexity of reasoning in the hybrid formalism, however, depends on the complexity of $\mathcal{DL}$, so we have studied the cases when $\mathcal{DL}$ is one of the DLs commonly used in practice.

Organization of the Paper

The rest of this paper is organized as follows. In Section 2, we present the basic definitions of DLs, ASP, and MKNF. In Section 3, we introduce the syntax and the semantics of MKNF$^+$ knowledge bases. In Section 4, we develop a suitable characterization of the models of MKNF$^+$ knowledge bases that provides the basis for a reasoning algorithm. In Section 5 we show that reasoning with MKNF$^+$ knowledge bases is undecidable under some common assumptions; therefore, in Section 6, we present a syntactic restriction that is sufficient for decidability, and we study the complexity of the resulting formalism. In Section 7, we discuss the related work and show that MKNF$^+$ knowledge bases can capture many of the existing combinations of DLs and rules.

2. PRELIMINARIES

In this section, we recapitulate the definitions of certain logical formalisms that we use in the rest of this paper.

2.1 Description Logics

In this paper, we consider an integration of an arbitrary first-order fragment $\mathcal{DL}$ with logic programming. This fragment does not need to be a description logic; however, our motivation stems from knowledge representation, so we call $\mathcal{DL}$ a description logic nonetheless. We present an overview of the features common to most DLs; precise definitions can be found in the literature [Baader et al. 2007].

A DL signature Σ consists of *atomic concepts*, *atomic roles*, and *individuals.* In first-order logic, atomic concepts correspond to unary predicates, atomic roles to binary predicates, and individuals to constants. DLs provide a rich set of constructors used to build complex concepts and roles from simpler ones. Table I shows the common constructors and the calligraphic letters used to identify them. A DL knowledge base $\mathcal{O}$ consists of a TBox $\mathcal{T}$ that describes the general structure of the world and an ABox $\mathcal{A}$ that describes particular objects in the world; both $\mathcal{T}$ and $\mathcal{A}$ must be finite. Table I summarizes the types of TBox and ABox axioms commonly found in DLs. The ABox axioms are usually called *assertions.* With $|\mathcal{O}|$ we denote the number of symbols needed to encode $\mathcal{O}$ on a tape of a Turing machine.

A DL knowledge base $\mathcal{O}$ is given semantics by interpreting it in a first-order structure—usually called an *interpretation*—where concepts correspond to unary predicates and roles correspond to binary predicates. More precisely, an interpretation I consists of a domain set $\triangle^I$ and an interpretation function $\cdot^I$ that interprets (i) each individual a as some element $a^I \in \triangle^I$, and (ii) concepts and roles as shown in Table I. An interpretation I is a model of $\mathcal{O}$ if it satisfies all TBox and ABox

Table I. Common DL Constructs and Their Semantics

Letters	Syntax	Name	Semantics
		Concepts and Roles	
	$\top$	Top concept	$\top^I = \triangle^I$
	$\bot$	Bottom concept	$\bot^I = \emptyset$
	A	Atomic concept	$A^I \subseteq \triangle^I$
	R	Atomic role	$R^I \subseteq \triangle^I \times \triangle^I$
$\mathcal{ALC}$	$\neg C$	Concept negation	$\triangle^I \setminus C^I$
	$C \sqcap D$	Concept intersection	$C^I \cap D^I$
	$C \sqcup D$	Concept union	$C^I \cup D^I$
	$\exists R.C$	Existential quantifier	$\{s \mid \exists t \in \triangle^I : \langle s,t\rangle \in R^I \wedge t \in C^I\}$
	$\forall R.C$	Universal quantifier	$\{s \mid \forall t \in \triangle^I : \langle s,t\rangle \in R^I \rightarrow t \in C^I\}$
$\mathcal{I}$	R^-	Inverse role	$\{\langle t,s\rangle \mid \langle s,t\rangle \in R^I\}$
$\mathcal{O}$	$\{a\}$	Nominals	$\{a^I\}$
$\mathcal{N}$	$\geq n\,R$	Unqualified number	$\{s \mid \sharp\{t \mid \langle s,t\rangle \in R^I\} \geq n\}$
	$\leq n\,R$	restrictions	$\{s \mid \sharp\{t \mid \langle s,t\rangle \in R^I\} \leq n\}$
$\mathcal{Q}$	$\geq n\,R.C$	Qualified number	$\{s \mid \sharp\{t \mid \langle s,t\rangle \in R^I \wedge t \in C^I\} \geq n\}$
	$\leq n\,R.C$	restrictions	$\{s \mid \sharp\{t \mid \langle s,t\rangle \in R^I \wedge t \in C^I\} \leq n\}$
		TBox Axioms	
	$C \sqsubseteq D$	Concept inclusion	$C^I \subseteq D^I$
$\mathcal{H}$	$R \sqsubseteq S$	Role inclusion	$R^I \subseteq S^I$
$\mathcal{S}$	$\mathsf{Trans}(R)$	Transitivity	$\forall s,t,u \in \triangle^I : \langle s,t\rangle \in R^I \wedge \langle t,u\rangle \in R^I \rightarrow \langle s,u\rangle \in R^I$
$\mathcal{R}$	$R \circ S \sqsubseteq T$	Role composition	$\forall s,t,u \in \triangle^I : \langle s,t\rangle \in R^I \wedge \langle t,u\rangle \in S^I \rightarrow \langle s,u\rangle \in T^I$
		ABox Axioms	
	$C(a)$	Concept assertion	$a^I \in C^I$
	$R(a,b)$	Positive role assertion	$\langle a^I, b^I\rangle \in R^I$
	$\neg R(a,b)$	Negative role assertion	$\langle a^I, b^I\rangle \notin R^I$
	$a \approx b$	Equality assertion	$a^I = b^I$
	$a \not\approx b$	Inequality assertion	$a^I \neq b^I$

axioms as shown in the bottom part of the table. Furthermore, $\mathcal{O}$ *entails* an axiom α, written $\mathcal{O} \models \alpha$, if α is true in all models of $\mathcal{O}$. It is well known that $\mathcal{O}$ can be translated into a first-order formula $\pi(\mathcal{O})$ that is satisfied in exactly the same models as $\mathcal{O}$ [Baader et al. 2007]; the translation of nominals and number restrictions requires counting quantifiers, which can be represented in first-order logic using the equality predicate $\approx$. The basic reasoning problem for $\mathcal{O}$ is checking its *satisfiability*—that is, checking whether $\mathcal{O}$ admits a model. Many DL-based applications also use the *entailment checking* problem, which is the problem of checking whether an axiom is entailed by a DL knowledge base. If the axiom is of the form $C \sqsubseteq D$ with C and D concepts, the problem is called *subsumption checking*, and if the axiom is an assertion, the problem is called *assertion checking*.

Conjunctive queries were proposed as an expressive query language for DLs [Calvanese et al. 1998]. For $\overline{x}$ and $\overline{y}$ disjoint vectors of *distinguished* and *nondistinguished* variables, respectively, a *conjunctive query* $Q(\overline{x})$ is a finite formula of the form $\exists \overline{y} : A_1 \wedge \ldots \wedge A_n$ where each A_i is a function-free first-order *atom* over predicates and constants from Σ and the variables from $\overline{x} \cup \overline{y}$. If $\overline{x}$ is empty, the conjunctive query is said to be *Boolean.* For $\overline{a}$ a vector of as many constants as there are variables in $\overline{x}$, with $Q(\overline{a})$ we denote the result of replacing in $Q(\overline{x})$ each $x_i \in \overline{x}$ with the corresponding $a_i \in \overline{a}$. We say that $\overline{a}$ is an *answer* to $Q(\overline{x})$ over a DL knowledge base $\mathcal{O}$, written $\mathcal{O} \models Q(\overline{a})$, if $Q(\overline{a})$ is true in every model of $\mathcal{O}$; furthermore, *query answering* is the problem of checking whether $\overline{a}$ is an answer to $Q(\overline{x})$. Decidability and complexity bounds of answering conjunctive queries are known for many DLs [Glimm et al. 2007; Calvanese et al. 1998; Calvanese et al. 2006; Ortiz et al. 2008; Krötzsch et al. 2008a].

As an example, consider the following DL knowledge base $\mathcal{O}$ about human anatomy. Axiom (1) states that the heart of each person is either on the left or on the right, and axiom (2) states that the heart cannot be both on the left and on the right. Furthermore, axiom (3) states that each person has a spinal column, and axiom (4) states that things that have a spinal column are vertebrates. Finally, axiom (5) states that *Bob* is an instance of *Person*. Then we can conclude $\mathcal{O} \models \mathit{Vertebrate}(\mathit{Bob})$—that is, that *Bob* is an instance of *Vertebrate*. In fact, we also have $\mathcal{O} \models \mathit{Person} \sqsubseteq \mathit{Vertebrate}$—that is, we can conclude that all people (even those not explicitly mentioned in $\mathcal{O}$) are vertebrates. Finally, for a Boolean conjunctive query $Q = \exists x : \mathit{SpinalColumn}(x)$, we have $\mathcal{O} \models Q$—that is, the axioms in $\mathcal{O}$ allow us to conclude that at least one spinal column exists even if we do not have an explicit name for it.

(1) $$\mathit{Person} \sqsubseteq \mathit{HeartOnLeft} \sqcup \mathit{HeartOnRight}$$
(2) $$\mathit{HeartOnLeft} \sqcap \mathit{HeartOnRight} \sqsubseteq \bot$$
(3) $$\mathit{Person} \sqsubseteq \exists \mathit{has}.\mathit{SpinalColumn}$$
(4) $$\exists \mathit{has}.\mathit{SpinalColumn} \sqsubseteq \mathit{Vertebrate}$$
(5) $$\mathit{Person}(\mathit{Bob})$$

2.2 Answer Set Programming

Answer set programming (ASP) [Gelfond and Lifschitz 1991] is nowadays one of the most popular nonmonotonic rule-based formalisms, mainly due to the availability of efficient answer set solvers such as DLV [Leone et al. 2006] and Smodels [Syrjänen and Niemelä 2001]. A *literal* is a formula of the form A or $\neg A$, where A is a function-free first-order atom. The negation $\neg$ is called *classical* (or sometimes also *strong*), and it is different from the *nonmonotonic* negation as failure, which is written as *not*. An answer set program $\mathcal{P}$ is a finite set of *rules* of the form

(6) $$H_1 \vee \ldots \vee H_k \leftarrow B_1, \ldots, B_m, \mathit{not}\, B_{m+1}, \ldots, \mathit{not}\, B_n$$

where B_i and H_j are *body* and *head* literals, respectively. A rule is *safe* if each variable in the rule occurs in an atom B_i for some $1 \leq i \leq m$; unless otherwise mentioned, we assume that all rules are safe. A rule is *positive* if $m = n$.[2] A rule

[2]Note that the term "positive" refers to the absence of nonmonotonic negation and not of the classical negation. Such usage of terminology is common in answer set programming, as literals can be understood as being "atomic" from the logical point of view.

with $k = 0$ is often written as having the single head literal false. Finally, a rule with $n = 0$ is often called a *fact*, and the $\leftarrow$ symbol is typically omitted in such cases. A rule is *ground* if it does not contain variables. These notions are extended to programs in the obvious way.

Roughly speaking, the semantics of ASP is obtained from the standard first-order semantics by considering only Herbrand models [Fitting 1996] (i.e., models in which ground terms are interpreted by themselves) and by considering only those models that contain a "minimal" amount of information necessary to satisfy the rules. We next recapitulate the formal definitions.

The *Herbrand base* of a ground program $\mathcal{P}$ is the set $HB_{\mathcal{P}}$ of all ground literals occurring in $\mathcal{P}$. An *interpretation* I for $\mathcal{P}$ is a consistent subset of $HB_{\mathcal{P}}$—that is, I is not allowed to contain both A and $\neg A$ for some ground atom A. An interpretation I *satisfies* a positive ground rule r of the form (6), written $I \models r$, if $B_i \in I$ for each $1 \leq i \leq m$ implies $H_j \in I$ for some $1 \leq j \leq k$. Let $\mathcal{P}$ be a positive ground program. Then, $I \models \mathcal{P}$ if and only if $I \models r$ for each $r \in \mathcal{P}$; furthermore, I is an *answer set* of $\mathcal{P}$ if $I \models \mathcal{P}$ and no interpretation $I' \subsetneq I$ exists such that $I' \models \mathcal{P}$.

Let $\mathcal{P}$ be a ground program in which the rules are allowed to contain *not*. The *reduct* of $\mathcal{P}$ w.r.t. an interpretation I is the positive ground program $\mathcal{P}^I$ obtained from $\mathcal{P}$ by deleting each rule of the form (6) such that $B_i \in I$ for some $m + 1 \leq i \leq n$, and by deleting all $\mathit{not}\, B_j$ in the remaining rules. An interpretation I is an *answer set* of $\mathcal{P}$ if I is an answer set of $\mathcal{P}^I$.

The *grounding* of a rule r w.r.t. a set of constants C is the set of rules $\mathsf{gr}(r, C)$ obtained by replacing in r all variables with the constants from C in all possible ways. The grounding of a program $\mathcal{P}$ w.r.t. a set of constants C is defined as $\mathsf{gr}(\mathcal{P}, C) = \bigcup_{r \in \mathcal{P}} \mathsf{gr}(r, C)$. An interpretation I is an *answer set* of a (not necessarily ground) program $\mathcal{P}$ if and only if I is an answer set of $\mathsf{gr}(\mathcal{P}, O_{\mathcal{P}})$, where $O_{\mathcal{P}}$ is the set of all constants occurring in $\mathcal{P}$.

As an example, consider the following program $\mathcal{P}$ that describes common knowledge about people. Rules (7)–(8) define an integrity constraint that checks whether each person has an explicitly specified social security number (SSN). One might try to express this statement by the DL axiom $\mathit{Person} \sqsubseteq \exists \mathit{hasSSN}.\top$; however, the latter axiom merely says that each each person has some (potentially unknown) SSN. In contrast, rules (7)–(8) behave as checks, and such checks cannot be expressed in first-order logic. Rule (9) is a default rule, which states that, unless the contrary can be proved, vertebrates have their hearts on the left.

$$\mathit{SSN\_OK}(x) \leftarrow \mathit{hasSSN}(x, y) \tag{7}$$

$$\mathsf{false} \leftarrow \mathit{Person}(x), \mathit{not}\, \mathit{SSN\_OK}(x) \tag{8}$$

$$\mathit{HeartOnLeft}(x) \leftarrow \mathit{Vertebrate}(x), \mathit{not}\, \neg \mathit{HeartOnLeft}(x) \tag{9}$$

If we extend $\mathcal{P}$ with a fact $\mathit{Verterbrate}(\mathit{Peter})$, then (9) derives $\mathit{HeartOnLeft}(\mathit{Peter})$. If, in addition, we explicitly state that Peter does not have his heart on the left—that is, if we add the fact $\neg \mathit{HeartOnLeft}(\mathit{Peter})$—then the default rule does not apply and we do not derive $\mathit{HeartOnLeft}(\mathit{Peter})$. Furthermore, if we provide a fact $\mathit{Person}(\mathit{Peter})$ without any additional information, then rules (7)–(8) derive a contradiction; we can prevent this by providing a fact such as $\mathit{hasSSN}(\mathit{Peter}, \mathit{ssn1})$.

Table II. Satisfaction of a Closed MKNF Formula in an MKNF Structure

$(I, M, N) \models \mathsf{true}$	for each triple (I, M, N)
$(I, M, N) \models P(t_1, \ldots, t_n)$	iff $\langle t_1^I, \ldots, t_n^I \rangle \in P^I$
$(I, M, N) \models \neg\varphi$	iff $(I, M, N) \not\models \varphi$
$(I, M, N) \models \varphi_1 \wedge \varphi_2$	iff $(I, M, N) \models \varphi_1$ and $(I, M, N) \models \varphi_2$
$(I, M, N) \models \exists x : \varphi$	iff $(I, M, N) \models \varphi[n_\alpha / x]$ for some $\alpha \in \triangle$
$(I, M, N) \models \mathbf{K}\,\varphi$	iff $(J, M, N) \models \varphi$ for all $J \in M$
$(I, M, N) \models \mathbf{not}\,\varphi$	iff $(J, M, N) \not\models \varphi$ for some $J \in N$

2.3 Minimal Knowledge and Negation as Failure

The logic of minimal knowledge and negation as failure (MKNF) [Lifschitz 1991] and the closely related logic of minimal belief and negation as failure (MBNF) [Lifschitz 1994] have been proposed as unifying frameworks for different nonmonotonic formalisms, such as default logic, autoepistemic logic, and logic programming.

Let $\Sigma = (\Sigma_c, \Sigma_f, \Sigma_p)$ be a first-order signature, where Σ_c is a set of constants, Σ_f is a set of function symbols, and Σ_p is a set of predicates containing the binary *equality* predicate $\approx$. The syntax of MKNF formulae over Σ is defined by the following grammar, where t_i are first-order terms and P is a predicate:

$$\varphi \leftarrow \mathsf{true} \mid P(t_1, \ldots, t_n) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \exists x : \varphi \mid \mathbf{K}\,\varphi \mid \mathbf{not}\,\varphi$$

Formulae of the form $P(t_1, \ldots, t_n)$ are called *first-order atoms*. Furthermore, formulae $\varphi_1 \vee \varphi_2$, $\forall x : \varphi$, $\varphi_1 \supset \varphi_2$, $\varphi_1 \equiv \varphi_2$, false, $t_1 \approx t_2$, and $t_1 \not\approx t_2$ are syntactic shortcuts for the formulae $\neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\neg(\exists x : \neg\varphi)$, $\neg\varphi_1 \vee \varphi_2$, $(\varphi_1 \supset \varphi_2) \wedge (\varphi_2 \supset \varphi_1)$, $\neg\mathsf{true}$, $\approx(t_1, t_2)$, and $\neg(t_1 \approx t_2)$, respectively. First-order atoms of the form $t_1 \approx t_2$ and $t_1 \not\approx t_2$ are called *equalities* and *inequalities*, respectively, and have a predefined interpretation [Fitting 1996]. A formula of the form $\mathbf{K}\,\varphi$ is a *modal* $\mathbf{K}$*-atom*, and a formula of the form $\mathbf{not}\,\varphi$ is a *modal* $\mathbf{not}$*-atom*; modal $\mathbf{K}$- and $\mathbf{not}$-atoms are *modal atoms*. An MKNF formula φ is *closed* if it has no free variables, and φ is *modally closed* if it is closed and all modal operators are applied in φ only to closed subformulae. With $\varphi[t_1/x_1, \ldots, t_n/x_n]$ we denote the result of simultaneously replacing in φ all free occurrences of the variables x_i with the terms t_i. An MKNF *theory* is a countable set of closed MKNF formulae.

Let Σ be a signature and $\triangle$ a nonempty set called *universe*. Just like in first-order logic, a *first-order interpretation* I over Σ and $\triangle$ assigns an object $a^I \in \triangle$ to each constant $a \in \Sigma_c$, a function $f^I : \triangle^n \to \triangle$ to each n-ary function symbol $f \in \Sigma_f$, and a relation $P^I \subseteq \triangle^n$ to each n-ary predicate $P \in \Sigma_p$, and it interprets the predicate $\approx$ as equality—that is, for $\alpha, \beta \in \triangle$, we have $\langle \alpha, \beta \rangle \in \approx^I$ iff $\alpha = \beta$. Unlike in standard first-order logic, for each element $\alpha \in \triangle$, the signature Σ is required to contain a special constant n_α—called a *name*—such that $n_\alpha^I = \alpha$. The interpretation of a variable-free term $t = f(s_1, \ldots, s_n)$ is defined recursively as $t^I = f^I(s_1^I, \ldots, s_n^I)$.

The semantics of an MKNF formula over a signature Σ (henceforth considered implicit in all definitions) is defined as follows. An *MKNF triple* over a universe $\triangle$ is a triple (I, M, N), where I is a first-order interpretation over $\triangle$ and Σ, and M and N are nonempty sets of first-order interpretations over $\triangle$ and Σ. Satisfiability of a closed MKNF formula in (I, M, N) is defined as shown in Table II.

An *MKNF interpretation* M over a universe $\triangle$ is a nonempty set of first-order interpretations over $\triangle$. For a closed MKNF formula φ, we say that M is an S5 *model* of φ, written $M \models \varphi$, if $(I, M, M) \models \varphi$ for each $I \in M$. As its name suggests, S5 models of φ are obtained by interpreting φ as a first-order modal formula in the modal logic S5 while taking **not** to be a shortcut for $\neg\,\mathbf{K}$.[3]

The nonmonotonic semantics of MKNF is obtained by preferring some S5 models over others: an MKNF interpretation M over $\triangle$ is an MKNF model of φ if (*i*) M is an S5 model of φ, and (*ii*) for each set of first-order interpretations M' over $\triangle$ such that $M' \supsetneq M$, we have $(I', M', M) \not\models \varphi$ for some $I' \in M'$.

An MKNF formula φ is MKNF *satisfiable* if an MKNF model of φ exists; otherwise, φ is MKNF *unsatisfiable*. Furthermore, φ MKNF *entails* ψ, written $\varphi \models_{\mathsf{MKNF}} \psi$, if $M \models \psi$ for each MKNF model M of φ. The S5 (un)satisfiability and entailment (written $\varphi \models_{\mathsf{S5}} \psi$), are defined analogously by considering S5 instead of MKNF models.

The definitions of models, (un)satisfiability, and entailment are extended to MKNF theories as usual; for example, an MKNF interpretation M is an S5 model of an MKNF theory T, written $M \models T$, if $M \models \varphi$ for each $\varphi \in T$.

Standard equivalences of first-order logic, such as de Morgan laws, are applicable to MKNF. Furthermore, let φ and ψ be arbitrary MKNF formulae, and let κ be a formula in which all first-order atoms occur within the scope of a modal operator. By the properties of S5, the formulae on the left- and the right-hand side of $\Leftrightarrow$ in the following table have the same truth values in any MKNF triple.

$$\begin{array}{rcl@{\qquad\qquad}rcl}
\mathbf{K}(\varphi \wedge \psi) & \Leftrightarrow & \mathbf{K}\,\varphi \wedge \mathbf{K}\,\psi & \mathbf{not}(\varphi \wedge \psi) & \Leftrightarrow & \mathbf{not}\,\varphi \vee \mathbf{not}\,\psi \\
\mathbf{K}(\kappa \vee \varphi) & \Leftrightarrow & \kappa \vee \mathbf{K}\,\varphi & \mathbf{not}(\kappa \vee \varphi) & \Leftrightarrow & \neg\kappa \wedge \mathbf{not}\,\varphi \\
\mathbf{K}(\forall x : \varphi) & \Leftrightarrow & \forall x : \mathbf{K}\,\varphi & \mathbf{not}(\forall x : \varphi) & \Leftrightarrow & \exists x : \mathbf{not}\,\varphi
\end{array}$$

Hence, we can replace any subformula of an MKNF formula of the form on the left with the formula on the right and vice versa.

Each closed MKNF formula φ clearly has the same MKNF models as $\mathbf{K}\,\varphi$; hence, for a closed MKNF formula ψ, we have $\varphi \models_{\mathsf{MKNF}} \psi$ iff $\mathbf{K}\,\varphi \models_{\mathsf{MKNF}} \psi$ iff $\varphi \models_{\mathsf{MKNF}} \mathbf{K}\,\psi$. Furthermore, if all first-order atoms occur in ψ within the scope of a modal operator, then $\varphi \models_{\mathsf{MKNF}} \psi$ iff $\varphi \wedge \neg\psi'$ is MKNF unsatisfiable, where ψ' is obtained by replacing in ψ each occurrence of $\mathbf{K}$ with $\neg\,\mathbf{not}$ [Rosati 2003]. Finally, if σ is a first-order subformula of φ, then without affecting MKNF satisfiability of φ we can introduce a fresh predicate Q, replace the occurrences of σ in φ with $Q(\overline{x})$, and add a definition $\forall \overline{x} : Q(\overline{x}) \equiv \sigma$, where $\overline{x}$ are the free variables of σ. This equivalence does not necessarily hold if σ contains modal operators [Lifschitz 1991].

Answer set programming can be embedded into MKNF. An ASP rule r of the form (6) with free variables $\overline{x}$ is transformed into an MKNF formula $\pi(r)$ as follows:

$$(10) \qquad \pi(r) = \forall \overline{x} : \left[\bigwedge_{1 \leq i \leq m} \mathbf{K}\,B_i \wedge \bigwedge_{m+1 \leq j \leq n} \mathbf{not}\,B_j \supset \bigvee_{1 \leq \ell \leq k} \mathbf{K}\,H_\ell \right]$$

[3]In first-order modal logics, possible worlds corresponds to first-order interpretations; furthermore, since each world is accessible from any other world in $S5$, a Kripke structure can be represented as a set M of first-order models.

An answer set program $\mathcal{P}$ is translated into the following MKNF formula:

$$\pi(\mathcal{P}) = \left[\bigwedge_{r\in\mathcal{P}} \pi(r)\right] \wedge \mathit{UNA}_\Sigma \wedge \mathit{RDA}_\Sigma \tag{11}$$

$$\mathit{UNA}_\Sigma = \bigwedge_{a,b\,\in\,\Sigma_c \text{ and } a\neq b} \mathbf{K}(a \not\approx b) \tag{12}$$

$$\mathit{RDA}_\Sigma = \bigwedge_{a\,\in\,\Sigma_c} \exists x : \mathbf{K}(x \approx a) \tag{13}$$

The MKNF models of $\pi(\mathcal{P})$ encode the answer sets of $\mathcal{P}$: for each answer set I of $\mathcal{P}$, there is an MKNF model M of $\pi(\mathcal{P})$ such that $M = \{J \mid J \models I\}$; conversely, for each MKNF model M of $\pi(\mathcal{P})$, the set of ground literals $I = \{L \mid M \models \mathbf{K}\, L\}$ is an answer set of $\mathcal{P}$. Lifschitz [1991] has shown this equivalence for propositional answer set programs; furthermore, by relying on the results by Lifschitz [1994], it is not difficult to see that the relationship holds for programs with variables as well if one additionally axiomatizes the unique name assumption (UNA_Σ) and ensures that each constant a is interpreted rigidly (RDA_Σ).

2.4 Complexity Classes

We use standard definitions of the complexity classes PTime, NP, and coNP [Papadimitriou 1993]. Furthermore, the classes NExpTime and N2ExpTime contain decision problems that can be solved by a nondeterministic Turing machine in single and double exponential time, respectively. The complexity class DP contains all decision problems that can be solved by solving one decision problem in NP and one decision problem in coNP. For complexity classes $\mathcal{C}$ and $\mathcal{E}$, with $\mathcal{E}^{\mathcal{C}}$ we denote the class of decision problems that can be solved by a Turing machine running in $\mathcal{E}$ and using an oracle for decision problems in $\mathcal{C}$. The polynomial hierarchy is defined inductively as follows:

$$\Delta_0^p = \Sigma_0^p = \Pi_0^p = \text{PTime}, \quad \Delta_{k+1}^p = \text{PTime}^{\Sigma_k^p}, \quad \Sigma_{k+1}^p = \text{NP}^{\Sigma_k^p}, \quad \Pi_{k+1}^p = \text{co}\Sigma_{k+1}^p.$$

3. A FRAMEWORK FOR COMBINING DESCRIPTION LOGICS WITH RULES

In this section we define the syntax and the semantics of MKNF$^+$ knowledge bases, present an example, and discuss some of their semantic properties. We start, however, with a simple example that motivates our work. Consider a DL knowledge base $\mathcal{O}$ consisting of axioms (1)–(4) and ABox facts (14)–(17), and an answer set program $\mathcal{P}$ consisting of rules (7)–(9).

$$\mathit{Person}(\mathit{Peter}) \tag{14}$$

$$\mathit{HeartOnRight}(\mathit{Peter}) \tag{15}$$

$$\mathit{Person}(\mathit{Paul}) \tag{16}$$

$$\mathit{hasSSN}(\mathit{Paul}, \mathit{ssn}_1) \tag{17}$$

We now present several inferences that one might intuitively expect from a hybrid formalism integrating $\mathcal{O}$ and $\mathcal{P}$.

DLs can reason about objects not explicitly mentioned in either $\mathcal{O}$ or $\mathcal{P}$, and this capability should be preserved. Thus, we should be able to derive from axioms

(3) and (4) that *Peter* and *Paul* are instances of *Vertebrate*. Note that axiom (3) contains an existential quantifier, so we need standard open-domain reasoning of first-order logic to draw the desired conclusion.

Nonmonotonic rules should provide a way for expressing integrity constraints over the DL knowledge base and thus allow for checking whether all information has been entered properly. For example, rules (7)–(8) express a constraint requiring that the SSN must be known for each object in the knowledge base. Axiom (17) specifies the SSN of *Paul*, but there is no such axiom for *Peter*, so the integrity constraint encoded by rules (7)–(8) should detect an inconsistency for *Peter*.

Rule (9) can be understood as a default rule. Both *Peter* and *Paul* are instances of *Vertebrate* due to $\mathcal{O}$, so (9) should be applicable to both *Peter* and *Paul*. Furthermore, axioms (14)–(17) specify no information about the location of the heart of *Paul*, so we expect to derive by (9) that his heart is on the left; in contrast, (15) explicitly says that the heart of *Peter* is on the right, so rule (9) should not "fire."

As this example demonstrates, our hybrid formalism should enable a mix of open- and closed-world reasoning, which is nontrivial to realize. Furthermore, our goal is to obtain a general framework for integrating DLs with both first-order and nonmonotonic rules that is capable of generalizing many of the existing proposals.

MKNF [Lifschitz 1991] has been specifically designed to capture first-order logic as well as many existing approaches to nonmonotonic reasoning. A DL knowledge base can straightforwardly be embedded into MKNF, and the same is the case for first-order rules. Furthermore, as shown in Section 2.3, ASP can be embedded into MKNF in a simple way as well. Therefore, it is natural to try to use MKNF as a semantic framework for an integration of DLs and ASP.

We proceed as follows. In Section 3.1, we argue that the semantics of MKNF as defined by Lifschitz [1991] must be extended with the *standard name assumption* to obtain the desired consequences for our hybrid formalism. Next, in Section 3.2 we introduce *MKNF*$^+$ *knowledge bases*—a very general combination of a first-order fragment and rules. In section 3.3 we show that the syntax of the formalism can be simplified without losing expressivity. In Section 3.4 we discuss a nonobvious interaction between modal atoms in the rules and existential quantifiers in the DL component. Finally, in Section 3.5 we present a nontrivial example.

3.1 Standard Name Assumption

Roughly speaking, a nonmonotonic rule in our formalism will be of the form (18), and it will be given semantics by translating it into the MKNF formula (19).

$$\mathbf{K}\,H_1 \vee \ldots \vee \mathbf{K}\,H_k \leftarrow \mathbf{K}\,B_1, \ldots, \mathbf{K}\,B_m, \mathbf{not}\,B_{m+1}, \ldots, \mathbf{not}\,B_n \tag{18}$$

$$\forall \vec{x} : \mathbf{K}\,H_1 \vee \ldots \vee \mathbf{K}\,H_k \subset \mathbf{K}\,B_1 \wedge \ldots \wedge \mathbf{K}\,B_m \wedge \mathbf{not}\,B_{m+1} \wedge \ldots \wedge \mathbf{not}\,B_n \tag{19}$$

The semantics of MKNF as defined by Lifschitz [1991], however, exhibits two undesirable properties that make such a straightforward definition undesirable.

The first problem arises because the semantics of MKNF considers arbitrary universes. Let $\varphi = \varphi_1 \wedge \varphi_2$, where $\varphi_1 = \mathbf{K}\,A(a)$ and $\varphi_2 = \mathbf{not}\,A(b) \supset \mathsf{false}$. The formula φ_2 can be understood as "it is an error not to derive $A(b)$," and the formula φ_1 provides no evidence that $A(b)$ is derivable; therefore, one might expect φ to be unsatisfiable similarly as it would be the case in ASP. The universe $\triangle$ of MKNF interpretations is, however, not fixed, so we can consider a universe $\triangle$ that contains only one object. Then, both a and b must be interpreted as the same objects, so φ is satisfied. Hence, instead of being unsatisfiable, we have $\varphi \models_{\mathsf{MKNF}} a \approx b$, which is not intuitive: we want to interpret φ_2 as in ASP—that is, as a constraint that only affects the consistency of φ and has no other side-effects. To avoid such problems, the semantics of ASP is defined w.r.t. Herbrand models, in which each constant is interpreted by itself.

The second problem is due to the semantics of quantification. Let $\varphi_1 = \mathbf{K}\, A(a)$ and $\varphi_2 = \exists x : \mathbf{K}\, A(x)$. Intuitively, we expect $\varphi_1 \models_{\mathsf{MKNF}} \varphi_2$ to hold; however, under the original semantics of MKNF, this is not the case. Consider an MKNF interpretation M such that $\{I_1, I_2\} \subseteq M$ where I_1 is a first-order interpretation in which a is interpreted as a name α_1 and $I_1 \models A(\alpha_1)$, and I_2 is a first-order interpretation in which a is interpreted as some other name α_2 and $I_2 \models A(\alpha_2)$. Clearly, $M \models \varphi_1$. For $M \models \varphi_2$ to hold, we should find a name α such that $I \models A(\alpha)$ for each $I \in M$. If we choose $\alpha = \alpha_1$, then $I_2 \not\models A(\alpha)$ and, similarly, if we choose $\alpha = \alpha_2$, then $I_1 \not\models A(\alpha)$; hence, $M \not\models \varphi_2$. The problem arises because a can be mapped in different interpretations in M to different domain objects. This can be corrected if we make a *rigid* using the MKNF formula $\exists x : \mathbf{K}(x \approx a)$: this ensures that a is interpreted in all first-order interpretations in M as the same name α, so $\varphi_1 \wedge \exists x : \mathbf{K}(x \approx a) \models_{\mathsf{MKNF}} \varphi_2$. This observation also explains why RDA_Σ is needed in the encoding of ASP into MKNF in Section 2.3. Consider the answer set program $\mathcal{P} = \{A(a),\ \mathsf{false} \leftarrow A(x)\}$; clearly, $\mathcal{P}$ does not have an answer set. Without RDA_Σ, the program $\mathcal{P}$ corresponds to the MKNF formula $\mathbf{K}\, A(a) \wedge \forall x : [\mathbf{K}\, A(x) \supset \mathsf{false}]$, which is MKNF equivalent to the formula $\psi = \mathbf{K}\, A(a) \wedge \neg[\exists x : \mathbf{K}\, A(x)] = \varphi_1 \wedge \neg\varphi_2$. Thus, ψ is MKNF satisfiable; however, $\psi \wedge \mathit{RDA}_\Sigma$ is MKNF unsatisfiable, which gives us the desired behavior.

The first problem could seemingly be solved by applying the unique name assumption (UNA)—that is, by including a fact $\mathbf{K}(a \not\approx b)$ for each pair of constants such that $a \neq b$. Such a semantics, however, is incompatible with first-order fragments that do not require UNA; therefore, such a solution is unsatisfactory as it may require a change to the semantics of $\mathcal{DL}$. Furthermore, UNA does not address the second problem, so we adopt a different solution.

Definition 3.1 *(Standard Name Assumption).* A first-order interpretation I over a signature Σ employs the *standard name assumption* if

(1) the universe $\triangle$ of I contains all constants of Σ and a countably infinite number of additional constants called *parameters*,

(2) $t^I = t$ for each ground term t constructed using the function symbols from Σ and the constants from $\triangle$, and

(3) the predicate $\approx$ is interpreted in I as a congruence relation—that is, it is reflexive, symmetric, transitive, and it allows the replacement of equals by equals [Fitting 1996].

Property (1) of Definition 3.1 avoids the first problem mentioned previously by fixing the universe $\triangle$. Property (2) avoids the second problem by requiring each constant to be rigid. Together, (1)–(2) make each model I equal to a Herbrand model with an infinite supply of constants. This, however, produces a nonstandard semantics that is not equivalent to first-order logic. For example, formula $\varphi = \forall x : (x \approx a)$ is unsatisfiable if we require (1)–(2) to hold in each interpretation and additionally interpret $\approx$ as true equality: (1) requires the universe $\triangle$ to be infinite, while φ requires it to contain at most one element. Property (3) remedies this problem by treating $\approx$ not as true equality, but as a congruence relation. Formula φ is satisfied in an interpretation that satisfies (1)–(3): the universe $\triangle$ is infinite, but $\alpha \approx \beta$ will be true in the interpretation for all pairs of elements $\alpha, \beta \in \triangle$. In the following sections, we implicitly consider only MKNF interpretations that satisfy the standard name assumption—that is, we always apply the standard name assumption in addition to the definitions presented in Section 2.3.

PROPOSITION 3.2. *Each first-order formula is satisfiable if and only if it is satisfiable in a model that employs the standard name assumption.*

PROOF. An equality-free first-order formula is satisfiable in an arbitrary model if and only if it is satisfiable in a Herbrand model with an infinite supply of constants not occurring in the formula [Fitting 1996, Theorem 5.9.4]. Hence, without equality we cannot distinguish satisfiability in arbitrary models from satisfiability only in Herbrand models. Furthermore, a first-order formula with equality is satisfiable in a model with true equality if and only if it is satisfiable in a model where $\approx$ is interpreted as a congruence relation [Fitting 1996, Theorem 9.3.9]. $\square$

By Proposition 3.2, we cannot distinguish consequences that first-order formulae have under the standard first-order semantics and the standard name assumption. Technically speaking, all first-order inferences that we mention in the following sections use the standard name assumption; however, we can consider them to be ordinary first-order inferences, as we cannot tell the difference.

The standard name assumption clearly makes all constants rigid, thus achieving the effect of the formula RDA_Σ from Section 2.3. The standard name assumption does not achieve the effect of UNA_Σ from Section 2.3; however, it makes constants "unique by default." Consider the formula $\varphi = \mathbf{not}(a \approx b) \supset \mathbf{K}\, Q(c)$. Due to Property (3) and the fact that $\approx$ is a standard first-order predicate, the extension of $\approx$ is minimized just like for any other predicate. Thus, a and b are assumed to be different because there is no evidence to the contrary, and $\varphi \models_{\mathsf{MKNF}} \mathbf{K}\, Q(c)$. This, however, does not prevent us from explicitly making a and b equal. If $\mathcal{P}$ is an answer set program that does not contain equality in the head of a rule (which is a standard restriction in ASP), then $\mathcal{P} \models A$ if and only if $\bigwedge_{r \in \mathcal{P}} \pi(r) \models_{\mathsf{MKNF}} \mathbf{K}\, A$, where π is the transformation from Section 2.3—that is, we do not need to include RDA_Σ and UNA_Σ in the translation.

3.2 MKNF$^+$ Knowledge Bases

In order to capture several existing combinations of DLs and rules, we define MKNF$^+$ knowledge bases as a very general formalism. The generality is achieved in the following two ways.

—In order to capture languages such as *EQL-Lite*$(\mathcal{Q})$ [Calvanese et al. 2007], dl-programs by Eiter et al. [2008], and dl-programs by Lukasiewicz [2007], we generalize the notion of literals in the rules and allow them to be arbitrary first-order formulae. For example, we allow literals to be conjunctive queries over $\mathcal{DL}$.

—In order to capture first-order combinations of DLs and rules, as well as approaches that allow for open- and closed-world reasoning in the rules such as $\mathcal{DL}$+log [Rosati 2005], we allow the rules to contain a mix of modal and nonmodal atoms.

All definitions in this and the following sections are parameterized by a signature Σ that contains the equality predicate $\approx$.

Definition 3.3 *(Syntax).* A *generalized atom* is a first-order formula. A generalized atom is *ground* if it does not contain free variables.[4] A generalized atom ξ_G is a *grounding* of a generalized atom ξ if ξ_G is obtained from ξ by replacing its free variables with constants. A *generalized atom base* $\mathcal{B}$ is a set of generalized atoms such that $\xi \in \mathcal{B}$ implies $\xi_G \in \mathcal{B}$ for each grounding ξ_G of ξ.

An *MKNF*$^+$ *atom* over $\mathcal{B}$ is either a *nonmodal atom* of the form ξ, a **K**-*atom* of the form $\mathbf{K}\,\xi$, or a **not**-*atom* of the form $\mathbf{not}\,\xi$, where $\xi \in \mathcal{B}$. The **K**- and **not**-atoms are collectively called *modal atoms.* An *MKNF*$^+$ *rule* r over $\mathcal{B}$ is a formula of the following form, where each H_i is a nonmodal or a **K**-atom over $\mathcal{B}$ and each B_i is either a nonmodal, a **K**-, or a **not**-atom over $\mathcal{B}$.

$$H_1 \vee \ldots \vee H_m \leftarrow B_1, \ldots, B_n \tag{20}$$

As usual, the set of atoms $\mathsf{head}(r) = \{H_i \mid 1 \leq i \leq m\}$ is called the *head* of r, and the set of atoms $\mathsf{body}(r) = \{B_i \mid 1 \leq i \leq n\}$ is called the *body* of r. If $m = 1$, the rule is *nondisjunctive*; if $n = 0$, it is a *fact*; and if it does not contain **not**-atoms, it is *positive.* The empty head ($m = 0$) is written as false. A rule is *safe* if each variable that occurs free in some rule atom also occurs free in a body **K**-atom. A *program* $\mathcal{P}$ over $\mathcal{B}$ is a finite set of MKNF$^+$ rules. The *size* of $\mathcal{P}$, written $|\mathcal{P}|$, is the number of symbols needed to encode $\mathcal{P}$ on a tape of a Turing machine.

Let $\mathcal{DL}$ be a description logic and let $\mathcal{B}$ be a generalized atom base. An *MKNF*$^+$ *knowledge base* over $\mathcal{DL}$ and $\mathcal{B}$ is a pair $\mathcal{K} = (\mathcal{O}, \mathcal{P})$, where $\mathcal{O} \in \mathcal{DL}$ is a DL knowledge base and $\mathcal{P}$ is a program over $\mathcal{B}$. The *size* of $\mathcal{K}$ is defined as $|\mathcal{K}| = |\mathcal{O}| + |\mathcal{P}|$. Given $\mathcal{K}$, with $O_\mathcal{K}$ we denote the set of all constants occurring in $\mathcal{K}$; if $\mathcal{K}$ does not contain constants, then $O_\mathcal{K}$ contains one arbitrary constant.

In the rest of this paper, all references to MKNF$^+$ knowledge bases are assumed to be implicitly parameterized by a description logic $\mathcal{DL}$ and a generalized atom base $\mathcal{B}$. To obtain a useful formalism in practice, $\mathcal{B}$ should at least include the standard function-free first-order atoms of the form $P(t_1, \ldots, t_n)$. Furthermore, to capture answer set programming, $\mathcal{B}$ should include negative atoms of the form $\neg P(t_1, \ldots, t_n)$. It is also useful to extend $\mathcal{B}$ with conjunctive queries over $\mathcal{DL}$, as this allows MKNF$^+$ rules to be used as an expressive query language for $\mathcal{DL}$ (see Section 7.6 for more information). These three types of generalized atoms are most likely to be relevant for practical usage.

We next define the semantics of MKNF$^+$ knowledge bases.

Definition 3.4 *(Semantics).* Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be an MKNF$^+$ knowledge base and let $\pi(\mathcal{O})$ be the translation of $\mathcal{O}$ into a formula of first-order logic with equality.

[4]Note that a ground generalized atom can contain variables bound by a quantifier.

(Such translations are known for most DLs [Baader et al. 2007].) We extend π to a rule r of the form (20) with the free variables $\overline{x}$ and to $\mathcal{K}$ as follows.

$$\begin{aligned}\pi(r) &= \forall \overline{x} : (B_1 \wedge \ldots \wedge B_n \supset H_1 \vee \ldots \vee H_m)\\ \pi(\mathcal{P}) &= \textstyle\bigwedge_{r \in \mathcal{P}} \pi(r)\\ \pi(\mathcal{K}) &= \mathbf{K}\,\pi(\mathcal{O}) \wedge \pi(\mathcal{P})\end{aligned}$$

An MKNF$^+$ knowledge base $\mathcal{K}$ is *MKNF satisfiable* if and only if an MKNF model of $\pi(\mathcal{K})$ exists. Furthermore, $\mathcal{K}$ *MKNF entails* an MKNF formula ψ, written $\mathcal{K} \models_{\mathsf{MKNF}} \psi$, if and only if $\pi(\mathcal{K}) \models_{\mathsf{MKNF}} \psi$.

To simplify the notation, we usually identify an MKNF knowledge base $\mathcal{K}$, a DL knowledge base $\mathcal{O}$, a program $\mathcal{P}$, and a rule r with the corresponding formula $\pi(\mathcal{K})$, $\pi(\mathcal{O})$, $\pi(\mathcal{P})$, and $\pi(r)$, respectively. Therefore, instead of writing, say, $M \models \pi(\mathcal{K})$, whenever no confusion can arise we simply write $M \models \mathcal{K}$.

For $\mathcal{K} = (\mathcal{O}, \emptyset)$ and ψ a closed first-order formula, clearly $\mathcal{K} \models_{\mathsf{MKNF}} \psi$ iff $\mathcal{O} \models \psi$. Furthermore, for $\mathcal{K} = (\emptyset, \mathcal{P})$, A a ground literal, and $\mathcal{P}'$ the answer set program corresponding to $\mathcal{P}$ as described in Section 2.2, $\mathcal{K} \models_{\mathsf{MKNF}} A$ if and only if $\mathcal{P}' \models A$ under the answer set semantics. Hence, the semantics of MKNF$^+$ knowledge bases is faithful w.r.t. the criteria put forth in the introduction.

3.3 Nonmodal Atoms in Rules

As we show in Section 7, the ability of MKNF$^+$ rules to incorporate both modal and nonmodal atoms is necessary to capture several existing approaches to integration of DLs and rules. This generality, however, adds a degree of complexity to our formalism because it mixes the first-order with the nonmonotonic aspects of reasoning. We next introduce a restricted version of our formalism in which the two types of concerns are more clearly separated.

Definition 3.5 *(MKNF Knowledge Bases).* MKNF rules and MKNF knowledge bases are defined as in Definition 3.3 with the difference that each atom in each rule must be a **K**- or **not**-atom.

Each MKNF$^+$ knowledge base can be transformed into an MKNF knowledge base, provided that we extend the generalized atom base appropriately.

Proposition 3.6. *For a generalized atom base $\mathcal{B}$, let $\mathcal{B}'$ be the smallest generalized atom base such that, for each nonempty finite subset $\{\varphi_1, \ldots, \varphi_n\} \subseteq \mathcal{B}$, each formula ψ of the form $(\neg)\varphi_1 \vee \ldots \vee (\neg)\varphi_n$, and each not necessarily proper and possibly empty subset $\overline{y}$ of the free variables of ψ, we have $\forall \overline{y} : \psi \in \mathcal{B}'$.*

For each MKNF$^+$ knowledge base $\mathcal{K}^+ = (\mathcal{O}, \mathcal{P}^+)$ over $\mathcal{DL}$ and $\mathcal{B}$, an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ over $\mathcal{DL}$ and $\mathcal{B}'$ exists such that $\mathcal{K}^+$ and $\mathcal{K}$ have exactly the same MKNF models.

Proof. By the identities listed in Section 2.3, $\pi(\mathcal{K}^+) = \mathbf{K}\,\mathcal{O} \wedge \bigwedge_{r \in \mathcal{P}^+} \pi(r)$ is MKNF equivalent to the formula $\mathbf{K}[\mathbf{K}\,\mathcal{O} \wedge \bigwedge_{r \in \mathcal{P}^+} \pi(r)]$; the latter is in turn MKNF equivalent to the formula $\mathbf{K}\,\mathcal{O} \wedge \bigwedge_{r \in \mathcal{P}^+} \mathbf{K}\,\pi(r)$, where r denotes rules of the following form containing both modal and nonmodal atoms:

$$\bigvee H_i \vee \bigvee \mathbf{K}\,H_j \leftarrow \bigwedge B_k \wedge \bigwedge \mathbf{K}\,B_m \wedge \bigwedge \mathbf{not}\,B_n$$

For each such r, let $\overline{y}$ be the list of variables that occur only in H_i and B_k, and let $\overline{x}$ be the list of all other variables in r. Then, the following equivalences hold.

$$\begin{aligned}
\mathbf{K}\,\pi(r) = \; & \mathbf{K}\,\forall\overline{x},\overline{y} : \{\bigvee H_i \vee \bigvee \mathbf{K}\,H_j \subset \bigwedge B_k \wedge \bigwedge \mathbf{K}\,B_m \wedge \bigwedge \mathbf{not}\,B_n\} && \Leftrightarrow \\
& \forall\overline{x} : \mathbf{K}\,\forall\overline{y} : \{\bigvee H_i \vee \bigvee \mathbf{K}\,H_j \subset \bigwedge B_k \wedge \bigwedge \mathbf{K}\,B_m \wedge \bigwedge \mathbf{not}\,B_n\} && \Leftrightarrow \\
& \forall\overline{x} : \{\mathbf{K}\,(\forall\overline{y} : \bigvee H_i \vee \bigvee \neg B_k) \vee \bigvee \mathbf{K}\,H_j \subset \bigwedge \mathbf{K}\,B_m \wedge \bigwedge \mathbf{not}\,B_n\}
\end{aligned}$$

We have thus transformed an MKNF$^+$ knowledge base over $\mathcal{DL}$ and $\mathcal{B}$ into an MKNF knowledge base over $\mathcal{DL}$ and $\mathcal{B}'$ while preserving the MKNF models. □

Intuitively, Lemma 3.6 allows us to "pull" the first-order aspects of MKNF$^+$ rules into the modal atoms; this is possible if we extend the set of modal atoms from $\mathcal{B}$ to $\mathcal{B}'$. In this way, we can cleanly delineate first-order from nonmonotonic reasoning: the former is restricted to the formulae in $\mathcal{B}'$ and $\mathcal{DL}$, and the latter is restricted to the modal MKNF rules. This allows us to isolate the influence of either component on the computational properties of reasoning. For example, in Section 6 we establish decidability of reasoning with MKNF knowledge bases by making only very general assumptions on the properties of reasoning in the first-order component.

We finish this section with a brief discussion of why MKNF is more suitable for our purposes than the closely related logic MBNF [Lifschitz 1994]. The latter logic differs from MKNF in the definition of models: an *MBNF model* is a pair (I, M) where I is a first-order interpretation and M is a set of first-order interpretations such that $(I, M, M) \models \varphi$ and no $M' \supsetneq M$ and I' exist such that $(I', M', M) \models \varphi$; note that I and I' are not required to be elements of M and M', respectively. Under MBNF semantics, $\mathbf{K}\,\pi(\mathcal{O})$ and nonmodal atoms in the rules would not interact, so the MBNF-based formalism would not correctly capture the semantics of first-order rule extensions of DLs such as SWRL. Furthermore, φ and $\mathbf{K}\,\varphi$ are *not* equivalent in MBNF, which would invalidate Proposition 3.6.

3.4 Existential Quantification in MKNF

The semantics of MKNF rules is quite different from the standard first-order semantics. Consider an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ where $\mathcal{O}$ contains the axiom (21)[5] and $\mathcal{P}$ contains the rule (22) in which B is a propositional variable.

$$\exists x : A(x) \tag{21}$$

$$\mathbf{K}\,p \leftarrow \mathbf{K}\,A(x) \tag{22}$$

One might intuitively expect $\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\,p$ to hold: (21) says that A is not empty, and (22) derives $\mathbf{K}\,p$ provided that $A(x)$ holds for some x. Such reasoning, however, is incorrect. The MKNF model M of $\mathcal{K}$ consists of first-order interpretations $I \in M$ such that $I \models A(\alpha)$ for some domain object $\alpha \in \triangle$, but this α varies across the interpretations in M. If, however, $M \models \mathbf{K}\,A(\alpha)$ were to hold, the object α should be the same across all interpretations in M. As the following proposition shows, this is true only in pathological cases.

[5]Since $\mathcal{DL}$ can be an arbitrary first-order fragment, we use a first-order formula in (21) rather than a DL axiom for the sake of simplicity.

PROPOSITION 3.7. *Let $\mathcal{O}$ be a DL knowledge base, let $a_1, \dots, a_n$ be arbitrary constants occurring in $\mathcal{O}$, and let ξ be a generalized atom with one free variable x such that all constants in ξ occur in $\mathcal{O}$. If $\mathcal{O} \models \xi[\alpha/x]$ for some $\alpha \in \triangle$ that does not occur in $\mathcal{O}$, then $\mathcal{O} \models \forall x : [\bigwedge_{1 \leq i \leq n} x \not\approx a_i] \supset \xi$.*[6]

PROOF. Assume that $\mathcal{O} \models \xi[\alpha/x]$ for some $\alpha \in \triangle$ not occurring in $\mathcal{O}$, but at the same time $\mathcal{O} \not\models \forall x : [\xi \vee \bigvee_{1 \leq i \leq n} x \approx a_i]$. Then, a model I of $\mathcal{O}$ and $\beta \in \triangle$ exist such that $I \not\models \xi[\beta/x]$ and $I \not\models \beta \approx a_i$ for all $1 \leq i \leq n$; by the latter condition, β does not occur in $\mathcal{O}$ or ξ. Let I' be an interpretation obtained from I by swapping α with β. Since I and I' are isomorphic and neither α nor β occurs in $\mathcal{O}$ and ξ, then $I' \models \mathcal{O}$ and $I' \not\models \xi[\alpha/x]$, which contradicts the assumption that $\mathcal{O} \models \xi[\alpha/x]$. □

Thus, $\mathbf{K}\,\xi[\alpha/x]$ can become true for an unnamed object α only if ξ is true for all $\triangle \setminus \{a_1, \dots, a_n\}$ in all models of $\mathcal{O}$. The latter, however, is true only when $\mathcal{O}$ ensures that ξ holds for "most" elements of the universe or when ξ is a tautology. Consequently, the MKNF rules can intuitively be understood as being applicable (mostly) to the individuals that are explicitly given a name either in the DL or the rule part. This limits the ability of MKNF rules to derive terminological consequences. For example, rule (23) does not imply the first-order formula (24): unlike the first-order atom $A(x)$ in the antecedent of (24), the atom $\mathbf{K}\,A(x)$ in the body of (23) is not applicable to the unnamed objects in the domain $\triangle$.

$$\mathbf{K}\,B(x) \leftarrow \mathbf{K}\,A(x) \tag{23}$$

$$\forall x : [A(x) \supset B(x)] \tag{24}$$

Intuitively, rule (23) "extends" the DL knowledge base with $B(a)$ for each named object a for which $A(a)$ is derivable. Depending on the expressivity of the DL, this may affect terminological consequences. For example, let $\mathcal{O}$ be the DL knowledge base containing axioms (25)–(27).

$$\forall x : [C(x) \supset D(x) \vee x \approx a] \tag{25}$$

$$\forall x : [B(x) \supset \neg C(x)] \tag{26}$$

$$A(a) \tag{27}$$

Let $\mathcal{P}$ contain solely rule (23), and let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$. Then, since assertion (27) makes $A(a)$ a consequence of $\mathcal{K}$, rule (23) "extends" $\mathcal{O}$ with $B(a)$. But then, since $\mathcal{O} \cup \{B(a)\} \models \forall x : [C(x) \supset D(x)]$, we conclude $\mathcal{K} \models_{\mathsf{MKNF}} \forall x : [C(x) \supset D(x)]$. Thus, while MKNF rules do not derive terminological consequences as the analogous first-order rules do, MKNF rules can affect terminological consequences by asserting new ground generalized atoms. Note that the latter can be first-order sentences and not just simple facts of the form $B(a)$. Thus, MKNF rules can be seen as a powerful mechanism for manipulating consequences of a first-order theory. Whether this leads to new terminological consequences depends on the structure of the generalized atoms and the DL knowledge base.

We next summarize the relationship between MKNF$^+$ and MKNF, and the types of consequences each formalism can derive.

[6]Note that, due to the standard name assumption, the elements of $\triangle$ are constants, so $\xi[\alpha/x]$ is a valid expression.

Table III. An MKNF Knowledge Base about Cities

(28)	$\mathit{portCity}(\mathit{Barcelona})$	Barcelona is a city with a port.
(29)	$\mathit{onSea}(\mathit{Barcelona}, \mathit{Mediterranean})$	Barcelona is on the Mediterranean.
(30)	$\mathit{portCity}(\mathit{Hamburg})$	Hamburg is a city with a port.
(31)	$\neg \mathit{seasideCity}(\mathit{Hamburg})$	Hamburg is not a seaside city.
(32)	$\mathit{rainyCity}(\mathit{Manchester})$	Manchester is a rainy city.
(33)	$\mathit{has}(\mathit{Manchester}, \mathit{AquaticsCentre})$	Manchester has the Aquatics Centre.
(34)	$\mathit{recreational}(\mathit{AquaticsCentre})$	Aquatics Centre is for recreation.
(35)	$\mathit{seasideCity} \sqsubseteq \exists \mathit{has}.\mathit{beach}$	Seaside cities have a beach.
(36)	$\mathit{beach} \sqsubseteq \mathit{recreational}$	Beaches are for recreation.
(37)	$\mathbf{K}\, \mathit{seasideCity}(x) \leftarrow \mathbf{K}\, \mathit{portCity}(x), \mathbf{not}\, \neg \mathit{seasideCity}(x)$	Port cities are usually at the seaside.
(38)	$\mathbf{K}\, \mathit{InterestingCity}(x) \leftarrow \mathbf{K}[\exists y : \mathit{has}(x, y) \wedge \mathit{recreational}(y)], \mathbf{not}\, \mathit{rainyCity}(x)$	Cities that have recreational facilities and are known not to be rainy are interesting.
(39)	$\mathbf{K}\, \mathit{HasOnSea}(x) \leftarrow \mathbf{K}\, \mathit{onSea}(x, y)$	For each seaside city we must...
(40)	$\mathsf{false} \leftarrow \mathbf{K}\, \mathit{seasideCity}(x), \mathbf{not}\, \mathit{HasOnSea}(x)$	...know on which sea the city is.
(41)	$\mathbf{K}\, \mathit{SummerDestination}(x, y) \leftarrow \mathbf{K}\, \mathit{InterestingCity}(x), \mathbf{K}\, \mathit{onSea}(x, y)$	Create a list of destinations.

Note: Predicates from $\mathcal{O}$ start with a lowercase and others with an uppercase letter.

—By allowing for a mix of nonmodal and modal atoms, MKNF$^+$ rules provide us with a very general semantic framework capable of capturing both first-order and nonmonotonic reasoning.

—The transformation of MKNF$^+$ to MKNF knowledge bases from the Section 3.3 allows us to cleanly separate first-order from nonmonotonic reasoning.

—The results in this section show the inherent limitations of the MKNF framework regarding the types of consequences one can derive.

In the rest of this paper, we focus primarily on the nonmonotonic aspects of reasoning, and leave the investigation of the first-order aspects to related work. Consequently, we focus for the most part on MKNF knowledge bases and use MKNF$^+$ only to establish relationships with other formalisms.

3.5 An Example

Imagine a knowledge-based recommender system helping users to decide where to go on holiday, based on the MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ presented in Table III. The intuitive meaning of each axiom is paraphrased on the right-hand side of the table. The DL part $\mathcal{O}$ consists of axioms (28)–(36) that state some basic facts about several European cities. The program $\mathcal{P}$ consists of rules (37)–(41) that derive a list of possible holiday destinations.

Rule (37) says that port cities are on the seaside by default—that is, unless there is evidence to the contrary. For example, Barcelona is a port city by (28), and there is no evidence that it is not a seaside city, so we derive $\mathit{seasideCity}(\mathit{Barcelona})$. In contrast, (31) explicitly says that Hamburg is not a seaside city (namely, Hamburg is located on the river Elbe). Hence, Hamburg is an exception to rule (37): the atom $\mathbf{not}\, \neg \mathit{seasideCity}(\mathit{Hamburg})$ is false, so the rule does not “fire.” We discuss the usage of MKNF rules to represent defaults in Section 7.7.

Rule (38) demonstrates two important points. The first body atom of (38) selects all things with a recreational facility. From $\mathit{seasideCity}(\mathit{Barcelona})$ and (35), we conclude that Barcelona has some beach, and by (36) we conclude that this beach is a recreational facility. Note that we do not know the name of the beach, but we know that it exists; nevertheless, this information suffices to make the **K**-atom $\mathbf{K}[\exists y : \mathit{has}(x, y) \wedge \mathit{recreational}(y)]$ true for $x = \mathit{Barcelona}$. Had we replaced this atom with the conjunction $\psi = \mathbf{K}\, \mathit{has}(x, y) \wedge \mathbf{K}\, \mathit{recreational}(y)$, we would not get this consequence because, for $M \models \mathbf{K}\, \mathit{recreational}(y)$ to hold for some y, the value of y must be the same in all first-order interpretations in M. Axiom (35) implies existence of a beach; however, the beach is not known by name and it can be a different individual in different first-order interpretations in M, thus preventing the variable y in ψ to be bound to a fixed element of the domain $\triangle$. Thus, an atom $\mathbf{K}\, \xi$ in the body of a rule can intuitively be understood as a query that checks whether $\mathcal{O}$ and the facts derived from all the rules imply ξ. Generalized atoms (such as conjunctive queries) in rules are thus an important generalization, since they allow us to pose general first-order, and not just atomic queries over $\mathcal{O}$.

The second body atom of rule (38) demonstrates the use of negation as failure and closed-world reasoning. The first atom of rule (38) also holds for $x = \mathit{Manchester}$ due to rules (33)–(34); however, assertion (32) explicitly says that Manchester is a rainy city, so **not** $\mathit{rainyCity}(x)$ is false and rule (38) does not "fire." In contrast, we have no information that Barcelona is a rainy city, so we make a default conjecture that it is not. Thus, rule (38) derives $\mathit{InterestingCity}(\mathit{Barcelona})$, but not $\mathit{InterestingCity}(\mathit{Manchester})$.

In our application, it might be useful to check whether $\mathcal{K}$ contains all relevant data—that is, whether it contains the name of the sea for each seaside city. We might try to use the axiom $\mathit{seasideCity} \sqsubseteq \exists \mathit{onSea}.\top$ for this purpose; however, this axiom merely derives that each seaside city is on some sea and does not really check whether the sea is explicitly present in $\mathcal{K}$. To solve this problem, we need a database-like integrity constraint that constrains the state of $\mathcal{K}$ rather than the world being modeled [Reiter 1992]. Rules (39)–(40) define such an integrity constraint: the first rule projects the second argument from the *onSea* relation, and the second rule performs the required check. Due to (29), the integrity constraint is satisfied for Barcelona; furthermore, there are no other seaside cities, so the integrity constraint is satisfied for all of $\mathcal{K}$.

Rule (41) finally produces a list of possible summer destinations, consisting of a city and a sea that the city is on. Since the integrity constraint (39)–(40) is satisfied, we know the sea of each seaside city, so we are sure that no interesting city has been dropped from the list just because the name of the sea is not known.

4. CHARACTERIZING MKNF MODELS

In this section we show that each MKNF model M of an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ can be characterized by a (possibly infinite) first-order theory T whose set of first-order models is exactly M—that is, $M = \{I \mid I \models T\}$. The theory T also characterizes MKNF entailments of the form $\mathcal{K} \not\models_{\mathsf{MKNF}} \psi$, where ψ is a modally closed MKNF formula. By Proposition 3.6, this characterization is applicable to MKNF$^+$ knowledge bases as well. This result is interesting for the following reasons.

—It reveals a close correspondence between MKNF knowledge bases and ASP: an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ can be intuitively understood as an answer set program $\mathcal{P}$ with an additional filter $\mathcal{O}$.

—We use this characterization in Section 6 to obtain decidability and complexity results of reasoning with a particular class of MKNF knowledge bases.

—Our characterization may provide a starting point for the implementation of practical reasoning systems.

For the sake of generality, we extend the definition of an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ and require the program $\mathcal{P}$ to be countable, but not necessarily finite; $\mathcal{P}$ and $\mathcal{K}$ are then given semantics by translating then into MKNF theories as follows:

$$\pi(\mathcal{P}) = \{\pi(r) \mid r \in \mathcal{P}\} \qquad \pi(\mathcal{K}) = \{\pi(\mathcal{O})\} \cup \pi(\mathcal{P})$$

If $\mathcal{P}$ is finite, this translation is equivalent to the one given in Definition 3.4, which justifies overloading the operator π. By allowing for infinite rule sets, we obtain the possibility of considering the ground knowledge base $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ in which $\mathcal{P}_G$ is the grounding of $\mathcal{P}$ w.r.t. the countable universe $\triangle$. By the semantics of MKNF, $\mathcal{K}$ has exactly the same models as $\mathcal{K}_G$; therefore, we develop our characterization only for ground MKNF knowledge bases.

For different types of MKNF rules, we present different strategies for identifying the theory that represents the MKNF models of $\mathcal{K}_G$. For general rules, we must use a guess-and-check approach. For the case of (stratified and nonstratified) nondisjunctive MKNF rules, we can construct the formula in a bottom-up fashion, much like this is done in ordinary datalog. These results are related to the characterization of MKNF models of propositional MKNF formulae [Rosati 1999] and of MKNF-based multiagent systems [Rosati 2003].

4.1 General Rules

We now develop our characterization of MKNF models by a first-order theory. We use the following MKNF knowledge base $\mathcal{K}_G^{ex} = (\mathcal{O}^{ex}, \mathcal{P}_G^{ex})$ to illustrate the concepts we introduce. The DL knowledge base $\mathcal{O}^{ex}$ contains only the propositional axiom (42),[7] and the program $\mathcal{P}_G^{ex}$ contains the MKNF rules (43)–(44).

$$q \supset r \tag{42}$$

$$\mathbf{K}\, q \leftarrow \mathbf{not}\, p \tag{43}$$

$$\mathbf{K}\, s \leftarrow \mathbf{not}\, r \tag{44}$$

Let $M^{ex} = \{I \mid I \models q \wedge (q \supset r)\}$. Clearly, we have $M^{ex} \models \mathbf{not}\, p$, $M^{ex} \models \mathbf{K}\, q$, $M^{ex} \models \mathbf{K}\, r$, $M^{ex} \not\models \mathbf{not}\, r$, and $M^{ex} \not\models \mathbf{K}\, s$. Furthermore, since M^{ex} contains all first-order interpretations in which q is true, for each MKNF interpretation M' such that $M' \supsetneq M^{ex}$, a first-order interpretation $I' \in M' \setminus M^{ex}$ exists such that $I' \not\models q$; thus, $M' \not\models \mathbf{K}\, q$, so $(I', M', M^{ex}) \not\models \mathcal{K}_G^{ex}$, and M^{ex} is an MKNF model of $\mathcal{K}_G^{ex}$. It is possible to see that $\mathcal{K}_G^{ex}$ has no other MKNF models.

Our formalism is related to ASP, so we next remind the reader of the reasoning algorithms used by ASP solvers. One can check whether a propositional answer set program $\mathcal{P}$ admits an answer set as follows:

[7]We take $\mathcal{DL}$ to be propositional logic for the sake of simplicity.

(1) Guess an interpretation I for $\mathcal{P}$.
(2) Check whether $I \models \mathcal{P}$.
(3) Check whether I is an answer set of $\mathcal{P}^I$. This can be done by guessing an interpretation $I' \subsetneq I$ and checking whether $I' \models \mathcal{P}^I$.

By applying these checks to the ASP program corresponding to $\mathcal{P}_G^{ex}$, we obtain $I = \{q, s\}$ as the only answer set.

Our characterization is similar in spirit to the ASP algorithm: it guesses an MKNF interpretation M for $\mathcal{K}_G$, checks whether $M \models \mathcal{K}_G$, and then checks whether an MKNF interpretation $M' \supsetneq M$ exists such that $(I', M', M) \models \mathcal{K}_G$. The main problem is that an MKNF interpretation is an infinite set of first-order interpretations, so it is cumbersome to work with. We next show, however, that each MKNF model M of $\mathcal{K}_G$ can be described by a first-order theory. We first introduce the notation that we use in the rest of this paper.

Definition 4.1. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground MKNF knowledge base. The set $\mathsf{KA}(\mathcal{K}_G)$ is the smallest set containing (i) all ground modal atoms $\mathbf{K}\,\xi$ occurring in $\mathcal{P}_G$, and (ii) an atom $\mathbf{K}\,\xi$ for each ground modal atom $\mathbf{not}\,\xi$ occurring in $\mathcal{P}_G$. Furthermore, $\mathsf{HA}(\mathcal{K}_G)$ is the subset of $\mathsf{KA}(\mathcal{K}_G)$ that contains all $\mathbf{K}$-atoms occurring in the head of some rule in $\mathcal{P}_G$.

Let P and N be disjoint sets of $\mathbf{K}$-atoms, and let $r_G \in \mathcal{P}_G$ be a ground rule. The rule $r_G[\mathbf{K}, P, N]$ is obtained by replacing each modal atom $\mathbf{K}\,\xi$ in r_G with true if $\mathbf{K}\,\xi \in P$ or with false if $\mathbf{K}\,\xi \in N$. Similarly, the rule $r_G[\mathbf{not}, P, N]$ is obtained by replacing each modal atom $\mathbf{not}\,\xi$ in r_G with true if $\mathbf{K}\,\xi \in N$ or with false if $\mathbf{K}\,\xi \in P$. Finally, $r_G[P, N] = r_G[\mathbf{K}, P, N][\mathbf{not}, P, N]$. In all these cases, the result is simplified as follows:

—If the rule contains the atom true in the head or the atom false in the body, the rule is replaced with $\mathsf{true} \leftarrow$.

—If all the head atoms in the rule are false and all body atoms in the rule are true, the rule is replaced with $\mathsf{false} \leftarrow$.

The programs $\mathcal{P}_G[\mathbf{K}, P, N]$, $\mathcal{P}_G[\mathbf{not}, P, N]$, and $\mathcal{P}_G[P, N]$ are obtained by replacing each rule $r_G \in \mathcal{P}_G$ with $r_G[\mathbf{K}, P, N]$, $r_G[\mathbf{not}, P, N]$, and $r_G[P, N]$, respectively.

For a set of rules $\mathcal{P}$, we write $\mathcal{P} = \mathsf{true}$ if $\mathcal{P} = \emptyset$ or if each rule in $\mathcal{P}$ is of the form $\mathsf{true} \leftarrow$; similarly, we write $\mathcal{P} = \mathsf{false}$ if $\mathcal{P}$ contains the rule $\mathsf{false} \leftarrow$.

Intuitively, $\mathsf{HA}(\mathcal{K}_G)$ determines the building blocks of our representation—that is, each MKNF model of $\mathcal{K}_G$ will be represented by a subset P_h of $\mathsf{HA}(\mathcal{K}_G)$. The set $\mathsf{KA}(\mathcal{K}_G)$ determines the modal atoms that need to be evaluated in the first-order theory. In our running example, we have $\mathsf{KA}(\mathcal{K}^{ex}) = \{\mathbf{K}\,p, \mathbf{K}\,q, \mathbf{K}\,r, \mathbf{K}\,s\}$ and $\mathsf{HA}(\mathcal{K}^{ex}) = \{\mathbf{K}\,q, \mathbf{K}\,s\}$. In ASP, the Herbrand base of an ASP program plays a role analogous to $\mathsf{HA}(\mathcal{K}_G)$ and $\mathsf{KA}(\mathcal{K}_G)$. The following definition shows how to obtain the desired representation of the MKNF models of $\mathcal{K}_G$.

Definition 4.2. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground MKNF knowledge base and let P_h be a subset of $\mathsf{HA}(\mathcal{K}_G)$. The *objective knowledge* of P_h w.r.t. $\mathcal{K}_G$ is the first-order theory $\mathsf{OB}_{\mathcal{O},P_h}$ defined as follows:

$$\mathsf{OB}_{\mathcal{O},P_h} = \{\pi(\mathcal{O})\} \cup \{\xi \mid \mathbf{K}\,\xi \in P_h\}$$

In our running example, the MKNF model M^{ex} corresponds to the objective knowledge $\mathsf{OB}_{\mathcal{O},P_h^{ex}}$ where $P_h^{ex} = \{\mathbf{K}\, q\}$—that is, $M^{ex} = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h^{ex}}\}$.

The following definition can be seen as being opposite to the previous one: it establishes a relationship between an MKNF interpretation M and a subset of the atoms of $\mathsf{HA}(\mathcal{K}_G)$ that are true in M.

Definition 4.3. For an MKNF interpretation M and a set of ground $\mathbf{K}$-atoms S, the subset of S *induced* by M is the set $\{\mathbf{K}\, \xi \in S \mid M \models \xi\}$.

The following key lemma shows that, for each MKNF model M of $\mathcal{K}_G$, the subset P_h of $\mathsf{HA}(\mathcal{K}_G)$ that is induced by M also characterizes M via $\mathsf{OB}_{\mathcal{O},P_h}$.

LEMMA 4.4. *For $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ a ground MKNF knowledge base, let M be an MKNF model of $\mathcal{K}_G$, and let P_h be the subset of $\mathsf{HA}(\mathcal{K}_G)$ induced by M. Then, M is equal to the set of first-order interpretations $M' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$.*

PROOF. Let M be an MKNF model of $\mathcal{K}$ and consider an arbitrary first-order interpretation $I \in M$. Since $M \models \mathbf{K}\, \mathcal{O}$, we have $I \models \mathcal{O}$. Furthermore, P_h is induced by M so, for each $\mathbf{K}\, \xi \in P_h$, we have $M \models \mathbf{K}\, \xi$, which implies $I \models \xi$. Thus, $I \models \mathsf{OB}_{\mathcal{O},P_h}$, which proves $M \subseteq M'$.

To show that $M' = M$, assume that $M' \supsetneq M$ and consider an arbitrary interpretation $I' \in M' \setminus M$. For each modal atom $\mathbf{not}\, \xi$, we have $(I', M', M) \models \mathbf{not}\, \xi$ iff $(I', M, M) \models \mathbf{not}\, \xi$ by the definition of satisfiability of $\mathbf{not}$-atoms in an MKNF triple. Furthermore, for each modal atom $\mathbf{K}\, \xi$ such that $M \not\models \mathbf{K}\, \xi$, since $M' \supsetneq M$, we have $M' \not\models \mathbf{K}\, \xi$ as well. Finally, for each modal atom $\mathbf{K}\, \xi \in P_h$, by the definition of $\mathsf{OB}_{\mathcal{O},P_h}$ we have $M' \models \mathbf{K}\, \xi$. To summarize, all $\mathbf{K}$-atoms in P_h and all $\mathbf{not}$-atoms have the same truth values in M and M', and if $M \not\models \mathbf{K}\, \xi$, then $M' \not\models \mathbf{K}\, \xi$ as well; we denote this property with (*). Consider now an arbitrary ground rule $r_G \in \mathcal{P}_G$. Since M is an MKNF model of $\mathcal{K}_G$, we have $M \models r_G$. If r_G contains a modal atom $\mathbf{K}\, \xi$ in the head such that $M \models \mathbf{K}\, \xi$, then $(I', M', M) \models r_G$ due to (*). If all head $\mathbf{K}$-atoms of r_G are false in M, then r_G contains a body atom that is false in M. If this is a $\mathbf{K}$-atom, by (*) this atom is false in M' as well, so $(I', M', M) \models r_G$; in case of a $\mathbf{not}$-atom, $(I', M', M) \models r_G$ holds trivially. Finally, $(I', M', M) \models \mathbf{K}\, \mathcal{O}$ by the definition of M'; hence, $(I', M', M) \models \mathcal{K}_G$, which contradicts the assumption that M is an MKNF model of $\mathcal{K}_G$. □

Thus, each MKNF model M of $\mathcal{K}_G$ is represented by a subset P_h of $\mathsf{HA}(\mathcal{K}_G)$ and, conversely, each subset P_h of $\mathsf{HA}(\mathcal{K}_G)$ corresponds to an MKNF interpretation M via $\mathsf{OB}_{\mathcal{O},P_h}$. Thus, to check whether $\mathcal{K}_G$ admits an MKNF model, one might guess P_h and then check, using the definitions from Section 2.3, whether the set of first-order interpretations of $\mathsf{OB}_{\mathcal{O},P_h}$ constitutes an MKNF model of $\mathcal{K}_G$. This requires determining the truth value of each atom $\mathbf{K}\, \xi \in \mathsf{KA}(\mathcal{K}) \setminus P_h$: if $\mathsf{OB}_{\mathcal{O},P_h} \models \xi$, then $\mathbf{K}\, \xi$ is true in the MKNF interpretation corresponding to P_h, and vice versa. Such a characterization, however, would not allow us to derive the optimal complexity results in Section 6, so we proceed in a slightly different way. We observe that each MKNF interpretation M partitions $\mathsf{KA}(\mathcal{K}_G)$ into (P, N), where P is a set of $\mathbf{K}$-atoms that are true in M, and N is a set of $\mathbf{K}$-atoms that are false in M. The relationship between (P, N) and M is captured by the following definition.

Definition 4.5. A partition (P, N) of $\mathsf{KA}(\mathcal{K}_G)$ is *consistent* with an MKNF interpretation M iff $\mathbf{K}\,\xi \in P$ implies $M \models \mathbf{K}\,\xi$, and $\mathbf{K}\,\xi \in N$ implies $M \not\models \mathbf{K}\,\xi$. Furthermore, (P, N) is *weakly consistent* with M iff $\mathbf{K}\,\xi \in P \cap \mathsf{HA}(\mathcal{K}_G)$ implies $M \models \mathbf{K}\,\xi$, and $\mathbf{K}\,\xi \in N$ implies $M \not\models \mathbf{K}\,\xi$.

In our running example, let $P^{ex} = \{\mathbf{K}\,q, \mathbf{K}\,r, \mathbf{K}\,s\}$ and $N^{ex} = \{\mathbf{K}\,p\}$. Partition (P^{ex}, N^{ex}) is consistent with M^{ex}: the set P^{ex} contains exactly the atoms that are true in M^{ex}, and the set N^{ex} contains exactly the atoms that are false in M^{ex}.

Instead of guessing P_h and then evaluating the atoms in $\mathsf{KA}(\mathcal{K}_G)$, we guess a partition (P, N) of $\mathsf{KA}(\mathcal{K}_G)$ and thus fix the value of all atoms in advance; then, for $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$, we check whether the initial guess was consistent with $\mathsf{OB}_{\mathcal{O},P_h}$.

In our running example, consider a partition (P, N) of $\mathsf{KA}(\mathcal{K}_G^{ex})$ where $P = \{\mathbf{K}\,q\}$ and $N = \{\mathbf{K}\,p, \mathbf{K}\,r, \mathbf{K}\,s\}$. Then, $P_h = P \cap \mathsf{HA}(\mathcal{K}_G) = \{\mathbf{K}\,q\}$. But then, axiom (42) implies that $\mathsf{OB}_{\mathcal{O},P_h} \models r$, so the choice $\mathbf{K}\,r \in N$ is not consistent with $\mathsf{OB}_{\mathcal{O},P_h}$.

The following lemma shows how to check whether a partition (P, N) is consistent with $\mathsf{OB}_{\mathcal{O},P_h}$.

Lemma 4.6. *Let $\mathcal{K}_G$ be a ground MKNF knowledge base, let $P_h \subseteq \mathsf{HA}(\mathcal{K}_G)$ be a set of $\mathbf{K}$-atoms, let $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$, and let (P, N) be a partition of $\mathsf{KA}(\mathcal{K}_G)$ such that $P_h \subseteq P$. Then, (P, N) is weakly consistent with M if and only if $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi$ for each $\mathbf{K}\,\xi \in N$. Furthermore, (P, N) is consistent with M if and only if, additionally, $\mathsf{OB}_{\mathcal{O},P_h} \models \xi$ for each $\mathbf{K}\,\xi \in P \setminus P_h$.*

Proof. Observe that $\mathsf{OB}_{\mathcal{O},P_h} \models \xi$ for each $\mathbf{K}\,\xi \in P_h$. Both claims now follow trivially from Definition 4.5. □

Thus, we can guess a partition (P, N) of $\mathsf{KA}(\mathcal{K}_G)$, take the set of head atoms $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$, and compute the formula $\mathsf{OB}_{\mathcal{O},P_h}$ that represents an MKNF interpretation M. To check whether $M \models \mathcal{P}_G$, since our partition (P, N) determines the value of all modal atoms in $\mathcal{P}_G$, we can simply replace each modal atom in $\mathcal{P}_G$ with its value as determined by (P, N) and simplify the result according to the rules for propositional logic. This is captured by the following lemma.

Lemma 4.7. *Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground MKNF knowledge base, let (P, N) be a partition of $\mathsf{KA}(\mathcal{K}_G)$, and let M be an MKNF interpretation.*

(1) *If (P, N) is consistent with M, then $\mathcal{P}_G[P, N] = \mathsf{true}$ iff $(I, M, M) \models \mathcal{P}_G$ for each $I \in M$.*
(2) *If (P, N) is weakly consistent with M and $\mathcal{P}_G$ is positive, then $\mathcal{P}_G[P, N] = \mathsf{true}$ implies $(I, M, M) \models \mathcal{P}_G$ for each $I \in M$.*

Proof. The first claim follows straightforwardly from Definition 4.5 and the definition of satisfaction of an MKNF theory in an MKNF triple. For the second claim, assume $\mathcal{P}_G[P, N] = \mathsf{true}$ and consider an arbitrary positive ground rule $r_G \in \mathcal{P}_G$ and each $I \in M$. Since $r_G[P, N] = \mathsf{true}$, either a head atom $\mathbf{K}\,H_i$ exists such that $\mathbf{K}\,H_i \in P$, or a body atom $\mathbf{K}\,B_j$ exists such that $\mathbf{K}\,B_j \notin P$. In the first case, since $\mathbf{K}\,H_i \in \mathsf{HA}(\mathcal{K}_G)$ and (P, N) is weakly consistent with M, we have $M \models \mathbf{K}\,H_i$, so $(I, M, M) \models r_G$. In the second case, $\mathbf{K}\,B_j \notin P$ implies $\mathbf{K}\,B_j \in N$; but then, since (P, N) is weakly consistent with M, we have $M \not\models \mathbf{K}\,B_j$, so $(I, M, M) \models r_G$ as well. □

In our running example, we have $\mathcal{P}_G^{ex}[P^{ex}, N^{ex}] = \mathsf{true}$; by Lemma 4.7, we then know that $M^{ex} \models \mathcal{P}_G^{ex}$.

Finally, to check whether $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$, we need to evaluate ψ in the MKNF interpretation represented by $\mathsf{OB}_{\mathcal{O},P_h}$. If ψ were to contain only atoms from $\mathsf{KA}(\mathcal{K}_G)$, we could do this as shown in Lemma 4.7. For the sake of generality, however, we allow ψ to contain arbitrarily nested modal operators, as well as modal atoms that do not occur in $\mathsf{KA}(\mathcal{K}_G)$. The truth value of ψ in the MKNF interpretation represented by $\mathsf{OB}_{\mathcal{O},P_h}$ is then determined inductively, as shown by the following lemma.

Definition 4.8. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground MKNF knowledge base, let P_h be a set of $\mathbf{K}$-atoms such that $P_h \subseteq \mathsf{HA}(\mathcal{K}_G)$, and let ψ be a modally closed MKNF formula. The *value* of ψ in P_h, written $\psi[P_h]$, is defined inductively as follows:

—if $\psi = \mathbf{K}\,\xi$, then $\psi[P_h] = \mathsf{true}$ if and only if $\mathsf{OB}_{\mathcal{O},P_h} \models \xi'$, where ξ' is the modally closed formula obtained from ξ by replacing each outermost modal atom in ξ with its value in P_h;

—if $\psi = \mathbf{not}\,\xi$, then $\psi[P_h] = \mathsf{true}$ if and only if $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi'$, where ξ' is the modally closed formula obtained from ξ by replacing each outermost modal atom in ξ with its value in P_h;

—in all other cases, $\psi[P_h] = \mathsf{true}$ if and only if $\mathsf{OB}_{\mathcal{O},P_h} \models \psi'[P_h]$, where ψ' is the modally closed formula obtained from ψ by replacing each outermost modal atom in ψ with its value in P_h.

LEMMA 4.9. *Let $\mathcal{K}_G$ be a ground MKNF knowledge base, let $P_h \subseteq \mathsf{HA}(\mathcal{K}_G)$ be a set of $\mathbf{K}$-atoms, let $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$, and let ψ be a modally closed MKNF formula. Then, $(I, M, M) \models \psi$ for each $I \in M$ if and only if $\psi[P_h] = \mathsf{true}$.*

PROOF. By induction on the structure of ψ one can show that, for each modally closed subformula ψ' of ψ, we have $(I, M, M) \models \psi'$ for each $I \in M$ iff $\psi'[P_h] = \mathsf{true}$; this property follows straightforwardly from the definition of evaluation of ψ' in (I, M, M) in Section 2.3, and it implies our claim. □

Based on these definitions, our characterization of MKNF entailments of the form $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$ is given by function $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$, defined in Table IV.

THEOREM 4.10. *For a ground MKNF knowledge base $\mathcal{K}_G$ and a modally closed MKNF formula ψ, $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ returns true if and only if $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$.*

PROOF. ($\Rightarrow$) If $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ returns true, then some $P \subseteq \mathsf{KA}(\mathcal{K}_G)$ satisfies Conditions (1)–(6). Let $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$ and $N = \mathsf{KA}(\mathcal{K}_G) \setminus P$. We show that $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$ is an MKNF model of $\mathcal{K}_G$. Since Condition (1) is satisfied, M is not empty. Since Conditions (2) and (3) are satisfied, (P, N) is consistent with M by Lemma 4.6. But then, since Condition (4) is satisfied, $(I, M, M) \models \mathcal{P}_G$ for each $I \in M$ by Lemma 4.7. Clearly $M \models \mathcal{O}$, so $(I, M, M) \models \mathcal{K}_G$ as well. Assume now that $M'' \supsetneq M$ exists such that $(I'', M'', M) \models \mathcal{K}_G$ for each $I'' \in M''$. Let P'_h be the subset of $\mathsf{HA}(\mathcal{K}_G)$ that is induced by M'', let $M' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P'_h}\}$, let P' be the subset of $\mathsf{KA}(\mathcal{K}_G)$ that is induced by M', and let $N' = \mathsf{KA}(\mathcal{K}_G) \setminus P'$. Clearly $M'' \subseteq M'$, so $M' \models \mathbf{K}\,\xi$ implies $M'' \models \mathbf{K}\,\xi$ for each $\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G)$; furthermore, $M'' \models \mathbf{K}\,\xi$ iff $M' \models \mathbf{K}\,\xi$ for each $\mathbf{K}\,\xi \in \mathsf{HA}(\mathcal{K})$, so $(I', M', M) \models \mathcal{K}_G$ for each $I' \in M'$. Consider now each $\mathbf{K}\,\xi \in P'$: since $M' \models \mathbf{K}\,\xi$ and $M \subseteq M'$, we

Table IV. The Definition of the Function $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$

```
Input:
  $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$: a ground MKNF knowledge base
  $\psi$ : a modally closed MKNF formula
Output:
  true if $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$; false otherwise

  if $P \subseteq \mathsf{KA}(\mathcal{K}_G)$ exists such that, with $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$ and $N = \mathsf{KA}(\mathcal{K}_G) \setminus P$,
  (1) $\mathsf{OB}_{\mathcal{O},P_h}$ is satisfiable, and
  (2) $\mathsf{OB}_{\mathcal{O},P_h} \models \xi$ for each $\mathbf{K}\,\xi \in P \setminus P_h$, and
  (3) $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi$ for each $\mathbf{K}\,\xi \in N$, and
  (4) $\mathcal{P}_G[P, N] = \mathsf{true}$, and
  (5) for each $P' \subsetneq P$, with $P'_h = P' \cap \mathsf{HA}(\mathcal{K}_G)$ and $N' = \mathsf{KA}(\mathcal{K}_G) \setminus P'$,
      (5a) $\mathsf{OB}_{\mathcal{O},P'_h} \models \xi$ for some $\mathbf{K}\,\xi \in N' \setminus N$, or
      (5b) $\mathcal{P}_G[\mathbf{not}, P, N][P', N'] = \mathsf{false}$
      and
  (6) $\psi[P_h] = \mathsf{false}$
  then return true; otherwise return false
```

have $M \models \mathbf{K}\,\xi$ as well, so $\mathbf{K}\,\xi \in P$. Furthermore, it is impossible that $P' = P$ and $M \neq M'$, so $P' \subsetneq P$. Since P' is induced by M', the partition (P', N') is consistent with M', so Condition (5a) does not hold by Lemma 4.6. Since $(I', M', M) \models \mathcal{K}_G$ for each $I' \in M'$ and because $\mathcal{P}_G[\mathbf{not}, P, N]$ does not contain **not**-atoms, we have $(I', M', M') \models \mathcal{P}_G[\mathbf{not}, P, N]$ as well. But then, $\mathcal{P}_G[\mathbf{not}, P, N][P', N'] = \mathsf{true}$ by Lemma 4.7, so Condition (5b) does not hold as well. This, however, contradicts the assumption that Condition (5) is satisfied. Hence, no such M' and M'' exist, and M is an MKNF model of $\mathcal{K}_G$. Due to Condition (6) and Lemma 4.9, we have $(I, M, M) \not\models \psi$ for some $I \in M$, so $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$.

($\Leftarrow$) If $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$, then an MKNF model M of $\mathcal{K}_G$ exists such that $M \not\models \psi$. We show that Conditions (1)–(6) are satisfied for the subset P of $\mathsf{KA}(\mathcal{K}_G)$ that is induced by M, $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$, and $N = \mathsf{KA}(\mathcal{K}_G) \setminus P$. By Lemma 4.4, we have $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$. Since M is not empty, $\mathsf{OB}_{\mathcal{O},P_h}$ is satisfiable, so Condition (1) holds. Since P is induced by M, the partition (P, N) is consistent with M, so Conditions (2) and (3) hold. Since $(I, M, M) \models \mathcal{P}_G$ for each $I \in M$, we have $\mathcal{P}_G[P, N] = \mathsf{true}$ by Lemma 4.7, so Condition (4) holds. Assume now that Condition (5) does not hold for some $P' \subsetneq P$, and let $P'_h = P' \cap \mathsf{HA}(\mathcal{K}_G)$, $N' = \mathsf{KA}(\mathcal{K}_G) \setminus P'$, and $M' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P'_h}\}$. Since $P'_h \subseteq P_h$ and $\mathsf{OB}_{\mathcal{O},P_h}$ is satisfiable by Condition (1), $\mathsf{OB}_{\mathcal{O},P'_h}$ is satisfiable as well, so M' is not empty. Since Condition (5a) does not hold, we have $\mathsf{OB}_{\mathcal{O},P'_h} \not\models \xi$ for each $\mathbf{K}\,\xi \in N'$, so (P', N') is weakly consistent with M' by Lemma 4.6. Since Condition (5b) does not hold, $\mathcal{P}_G[\mathbf{not}, P, N][P', N'] = \mathsf{true}$; furthermore, $\mathcal{P}_G[\mathbf{not}, P, N]$ is positive, so $(I', M', M') \models \mathcal{P}_G[\mathbf{not}, P, N]$ for each $I' \in M'$ by Lemma 4.7. Since $\mathcal{P}_G[\mathbf{not}, P, N]$ does not contain **not**-atoms, we have $(I', M', M) \models \mathcal{P}_G[\mathbf{not}, P, N]$ as well. Clearly, $M' \models \mathcal{O}$, so $(I', M', M) \models \mathcal{K}_G$. Finally, P is exactly the subset of $\mathsf{KA}(\mathcal{K}_G)$ that is true in M, so $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P}\}$. Since $P'_h \subseteq P' \subsetneq P$, we have $M \subsetneq M'$. This, however, contradicts the assumption that M is an MKNF model of $\mathcal{K}_G$. Hence, no such P' exists and Condition (5) is satisfied. Condition (6) is satisfied by Lemma

4.9, so $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ returns true. □

We invite the reader to verify that (P^{ex}, N^{ex}) is the only partition of $\mathsf{KA}(\mathcal{K}_G^{ex})$ that satisfies Conditions (1)–(5) in Table IV. Hence, our characterization correctly identifies $\mathsf{OB}_{\mathcal{O},P_h^{ex}}$ as the MKNF model of $\mathcal{K}_G^{ex}$.

Also, the reader should note a close correspondence between the characterization shown in Table IV and the procedure used to compute answer sets of an ASP program outlined at the beginning of this section: step 1 of the ASP algorithm corresponds to guessing a subset P of $\mathsf{KA}(\mathcal{K}_G)$; step 2 corresponds to Condition (4); and step 3 corresponds to guessing a subset $P' \subsetneq P$ and checking Condition (5b). Thus, the general structures of the two algorithms are the same, and Conditions (1)–(3) and (5a) in Table IV can be understood as additional filters that ensure the compatibility of an answer set with the DL knowledge base.

Due to this similarity, we believe that $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ provides a starting point for the implementation of practical reasoners for cases when $\mathcal{P}_G$ is finite and all first-order problems are decidable. ASP systems such as DLV and Smodels use advanced optimization strategies such as intelligent grounding [Eiter et al. 1997] to prune the number of candidate interpretations; however, the implemented algorithms follow roughly the algorithm outlined in the previous paragraph. These optimizations are equally applicable in the case of MKNF knowledge bases. Thus, a practical reasoner for MKNF knowledge bases can be obtained by integrating an ASP system with a DL reasoner and extending the basic ASP algorithm with Conditions (1)–(3) and (5a). While integrating an ASP solver with a DL reasoner might not be straightforward from an engineering point of view, we see no fundamental obstacles that would make our approach inappropriate for practical usage.

4.2 Positive Rules

The function from the previous section can be simplified if the rules in $\mathcal{P}_G$ do not contain **not**-atoms and ψ is of the form $\mathbf{K}\,\varphi$ with φ a closed first-order formula. To this end, we use the following property.

Lemma 4.11. *For $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ a positive MKNF knowledge base and φ a closed first-order formula, $\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\,\varphi$ if and only if $\mathcal{K} \models_{\mathsf{S5}} \mathbf{K}\,\varphi$.*

Proof. The ($\Leftarrow$) direction is trivial because all S5 models are also MKNF models. For the ($\Rightarrow$) direction, assume that $\mathcal{K} \not\models_{\mathsf{S5}} \mathbf{K}\,\varphi$; hence, an S5 model M of $\mathcal{K}$ exists such that $I_1 \not\models \varphi$ for some $I_1 \in M$. Let M' be the maximal MKNF interpretation such that $M' \supseteq M$ and $(I', M', M') \models \mathcal{K}$ for each $I' \in M'$. Since $\mathcal{K}$ does not contain **not**-atoms, $(I', M', M) \models \mathcal{K}$ as well, so M' is an MKNF model of $\mathcal{K}$. Clearly, $I_1 \in M'$, so $M' \not\models \mathbf{K}\,\varphi$, which implies $\mathcal{K} \not\models_{\mathsf{MKNF}} \mathbf{K}\,\varphi$. □

Let $\mathsf{not\text{-}entails}^+(\mathcal{K}_G, \psi)$ be the function defined as shown in Table IV, but without Conditions (2) and (5). As we show next, such a function can be used for answering positive queries in a positive knowledge base.

Theorem 4.12. *For a positive ground MKNF knowledge base $\mathcal{K}_G$ and $\psi = \mathbf{K}\,\varphi$ with φ a closed first-order formula, the function $\mathsf{not\text{-}entails}^+(\mathcal{K}_G, \psi)$ returns true if and only if $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$.*

Proof. If $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$, then Conditions (1), (3), and (4) in Table IV hold in the same way as in Theorem 4.10, so $\mathsf{not\text{-}entails}^+(\mathcal{K}_G, \psi)$ returns true. Conversely, if the function not-entails$^+(\mathcal{K}_G, \psi)$ returns true, then a subset P of $\mathsf{KA}(\mathcal{K}_G)$ satisfying Conditions (1), (3), (4), and (6) exists. Let $P_h = P \cap \mathsf{HA}(\mathcal{K}_G)$, $N = \mathsf{KA}(\mathcal{K}_G) \setminus P$, and $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$. Condition (1) holds, so M is not empty. Condition (3) holds, so (P, N) is weakly consistent with M. Condition (4) holds and $\mathcal{P}_G$ is positive, so $(I, M, M) \models \mathcal{P}_G$ for each $I \in M$ by Lemma 4.7. Clearly, $M \models \mathcal{O}$, so M is an S5 model of $\mathcal{K}_G$. Condition (6) holds, so $M \not\models \psi$ by Lemma 4.9, which implies $\mathcal{K}_G \not\models_{\mathsf{S5}} \psi$. But then, $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$ as well by Lemma 4.11. □

4.3 Positive Nondisjunctive Rules

We now turn our attention to the case when $\mathcal{K}_G$ is a positive nondisjunctive MKNF knowledge base—that is, when the rules in $\mathcal{P}_G$ have the form

$$\mathbf{K}\, H \leftarrow \mathbf{K}\, B_1, \ldots, \mathbf{K}\, B_m. \tag{45}$$

We show that such programs are either unsatisfiable or they have a single MKNF model that corresponds to the least fixpoint of a certain operator.

Theorem 4.13. *If an MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ is positive and nondisjunctive, then $\mathcal{K}$ has at most one MKNF model.*

Proof. We first show the following property (*): if $\mathcal{M}$ is a set of MKNF interpretations such that $M \in \mathcal{M}$ implies $M \models \mathcal{K}$, then $\bigcup \mathcal{M} \models \mathcal{K}$ as well.[8] Since $M \in \mathcal{M}$ implies $M \models \mathcal{K}$, we have $I \models \mathcal{O}$ for each $I \in M$, so $\bigcup \mathcal{M} \models \mathcal{O}$ as well. Consider an arbitrary ground rule r_G obtained from an arbitrary rule $r \in \mathcal{P}$ by replacing all universally quantified variables with some elements from $\triangle$. The rule r_G is of the form (45). Assume that r_G is such that $\bigcup \mathcal{M} \models \mathbf{K}\, B_i$ for each $1 \leq i \leq n$. Consider an arbitrary $M \in \mathcal{M}$. Since $M \subseteq \bigcup \mathcal{M}$, we have $M \models \mathbf{K}\, B_i$ as well. Since $M \models r_G$, we have $M \models \mathbf{K}\, H$. But then, $\bigcup \mathcal{M} \models \mathbf{K}\, H$, so $\bigcup \mathcal{M} \models r_G$ as well.

Let $\mathcal{M}$ be the set of all MKNF models of $\mathcal{K}$. To prove this theorem, assume that $\mathcal{M}$ contains more than one element. Clearly, $M \subsetneq \bigcup \mathcal{M}$ for each $M \in \mathcal{M}$; furthermore, $\bigcup \mathcal{M} \models \mathcal{K}$ by (*), which contradicts the assumption that $\mathcal{M}$ contains all MKNF models of $\mathcal{K}$. □

By Lemma 4.4, the MKNF model of $\mathcal{K}_G$ can be represented by using a subset P_h of $\mathsf{HA}(\mathcal{K}_G)$. We now show how to compute this subset in a deterministic way.

Definition 4.14. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a positive ground nondisjunctive MKNF knowledge base. The *immediate consequence operator* $T_{\mathcal{K}_G} : 2^{\mathsf{HA}(\mathcal{K}_G)} \rightarrow 2^{\mathsf{HA}(\mathcal{K}_G)}$ is defined as follows:

$$\begin{aligned} T_{\mathcal{K}_G}(S) = \{\mathbf{K}\, H \mid & \text{ for each rule } r_G \in \mathcal{P}_G \text{ of the form (45) such that} \\ & \mathsf{OB}_{\mathcal{O},S} \models B_i \text{ for each } 1 \leq i \leq m\} \cup \\ & \{\mathbf{K}\, \xi \in \mathsf{HA}(\mathcal{K}_G) \mid \mathsf{OB}_{\mathcal{O},S} \models \xi\} \end{aligned}$$

Intuitively, the first part of $T_{\mathcal{K}_G}(S)$ computes the immediate consequences of $\mathcal{P}_G$ assuming that the atoms in S are known to hold, and the second part adds the atoms from $\mathsf{HA}(\mathcal{K}_G)$ that are entailed by $\mathsf{OB}_{\mathcal{O},S}$. Note that, if $\mathsf{OB}_{\mathcal{O},S}$ is unsatisfiable, then $T_{\mathcal{K}_G}(S) = \mathsf{HA}(\mathcal{K}_G)$. The following property of $T_{\mathcal{K}_G}$ is easy to prove.

[8] $\bigcup \mathcal{M}$ should be understood as $\bigcup_{M \in \mathcal{M}} M$.

LEMMA 4.15. *The operator $T_{\mathcal{K}_G}(S)$ is monotone on the lattice of the subsets of $\mathsf{HA}(\mathcal{K}_G)$—that is, for each $S, S' \subseteq \mathsf{HA}(\mathcal{K}_G)$, if $S \subseteq S'$, then $T_{\mathcal{K}_G}(S) \subseteq T_{\mathcal{K}_G}(S')$.*

PROOF. If $\mathbf{K}\,\xi \in T_{\mathcal{K}_G}(S)$ because $\mathsf{OB}_{\mathcal{O},S} \models \xi$, by monotonicity of first-order logic we have $\mathsf{OB}_{\mathcal{O},S'} \models \xi$ as well, so $\mathbf{K}\,\xi \in T_{\mathcal{K}_G}(S')$. If $\mathbf{K}\,\xi \in T_{\mathcal{K}_G}(S)$ because $\mathbf{K}\,\xi$ occurs in the head of some rule $r_G \in \mathcal{P}_G$ and $\mathsf{OB}_{\mathcal{O},S} \models B_i$ for each $\mathbf{K}\,B_i \in \mathsf{body}(r_G)$, then $\mathsf{OB}_{\mathcal{O},S'} \models B_i$ as well, so $\mathbf{K}\,\xi \in T_{\mathcal{K}_G}(S')$. □

By Lemma 4.15 and the well-known Knaster-Tarski theorem, the operator $T_{\mathcal{K}_G}$ has a unique least fixpoint $T^{\infty}_{\mathcal{K}_G}$. This fixpoint can be obtained by setting $S_0 = \emptyset$ and $S_i = T_{\mathcal{K}_G}(S_{i-1})$ for $i > 0$. If $\mathcal{P}_G$ is finite, then $S_n = S_{n+1} = \ldots = T^{\infty}_{\mathcal{K}_G}$ for some integer n; otherwise, $T^{\infty}_{\mathcal{K}_G} = \bigcup_{i \in \mathbb{N}} S_i$. We now show that, if it is consistent, the objective knowledge $\mathsf{OB}_{\mathcal{O},T^{\infty}_{\mathcal{K}_G}}$ defines an MKNF model of $\mathcal{K}_G$.

THEOREM 4.16. *Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a positive ground nondisjunctive MKNF knowledge base, and let $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},T^{\infty}_{\mathcal{K}_G}}\}$.*

(1) If $M \neq \emptyset$, then M is an MKNF model of $\mathcal{K}_G$.

(2) If $\mathcal{K}_G$ has an MKNF model, then this model is equal to M.

PROOF. (Claim 1) Assume that $M \neq \emptyset$. Clearly, $M \models \mathcal{O}$. Furthermore, for each rule of the form (45), if $M \models \mathbf{K}\,B_i$ for each $1 \leq i \leq m$, since $T^{\infty}_{\mathcal{K}_G}$ is a fixpoint of $T_{\mathcal{K}_G}$, then $\mathbf{K}\,H \in T^{\infty}_{\mathcal{K}_G}$ as well, so $M \models \mathbf{K}\,H$. Hence, $(I, M, M) \models \mathcal{K}_G$ for each $I \in M$. Assume now that some $M'' \supsetneq M$ exists such that $(I'', M'', M) \models \mathcal{K}_G$ for each $I'' \in M''$. Let P'_h be the subset of $\mathsf{HA}(\mathcal{K}_G)$ induced by M'', and let $M' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P'_h}\}$. As in the proof of Theorem 4.10, we have $M'' \subseteq M'$ and $(I', M', M) \models \mathcal{K}_G$ for each $I' \in M'$. Now consider an arbitrary ground rule $r_G \in \mathcal{P}_G$ of the form (45). If $\mathsf{OB}_{\mathcal{O},P'_h} \models B_i$ for each $1 \leq i \leq m$, then $M' \models \mathbf{K}\,B_i$ as well; but then, $M' \models r_G$ implies $M' \models \mathbf{K}\,H$, which implies $\mathbf{K}\,H \in P'_h$. Similarly, if $\mathsf{OB}_{\mathcal{O},P'_h} \models \xi$ for some $\mathbf{K}\,\xi \in \mathsf{HA}(\mathcal{K}_G)$, then $M' \models \mathbf{K}\,\xi$, so $\mathbf{K}\,\xi \in P'_h$. Hence, $T_{\mathcal{K}_G}(P'_h) = P'_h$—that is, P'_h is a fixpoint of $T_{\mathcal{K}_G}$. Consider now each $\mathbf{K}\,\xi \in P'_h$. Since $M' \supsetneq M$, we have $M \models \mathbf{K}\,\xi$, which implies $\mathsf{OB}_{\mathcal{O},T^{\infty}_{\mathcal{K}_G}} \models \xi$; but then, by Definition 4.14, $\mathbf{K}\,\xi \in T^{\infty}_{\mathcal{K}_G}$, so $P'_h \subseteq T^{\infty}_{\mathcal{K}_G}$. Finally, since $M \neq M'$, we have $P'_h \neq T^{\infty}_{\mathcal{K}_G}$, so $P'_h \subsetneq T^{\infty}_{\mathcal{K}_G}$. Hence, P'_h is a fixpoint of $T_{\mathcal{K}_G}$ that is strictly smaller than the least fixpoint $T^{\infty}_{\mathcal{K}_G}$, which is a contradiction. Thus, no such M' and M'' exist, and the claim holds.

(Claim 2) Assume that $\mathcal{K}_G$ has an MKNF model M', which induces a subset P'_h of $\mathsf{HA}(\mathcal{K}_G)$. As in the proof of Claim 1, $T_{\mathcal{K}_G}(P'_h) = P'_h$—that is, P'_h is a fixpoint of $T_{\mathcal{K}_G}$. Clearly, $\mathsf{OB}_{\mathcal{O},P'_h}$ is satisfiable. Furthermore, since $T^{\infty}_{\mathcal{K}_G}$ is the minimal fixpoint of $T_{\mathcal{K}_G}$, we have $T^{\infty}_{\mathcal{K}_G} \subseteq P'_h$; but then, $\mathsf{OB}_{\mathcal{O},T^{\infty}_{\mathcal{K}_G}}$ is satisfiable and $M \neq \emptyset$. By claim 1, M is an MKNF model of $\mathcal{K}_G$; furthermore, by Theorem 4.13, $M = M'$. □

By Lemma 4.9 and Theorem 4.16, $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$ for some modally closed MKNF formula ψ if and only if $\mathsf{OB}_{\mathcal{O},T^{\infty}_{\mathcal{K}_G}} \models \psi[T^{\infty}_{\mathcal{K}_G}]$.

4.4 Stratified Rules

We now define the class of *stratified* programs. These are nondisjunctive, but they can contain **not**-atoms—that is, they are of the form (46). The rules, however, can be separated in strata, each of which can be evaluated separately.

$$\mathbf{K}\, H \leftarrow \mathbf{K}\, B_1, \ldots, \mathbf{K}\, B_m, \mathbf{not}\, B_{m+1}, \ldots, \mathbf{not}\, B_n \tag{46}$$

Definition 4.17. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground nondisjunctive MKNF knowledge base, and let $\lambda : \mathsf{KA}(\mathcal{K}_G) \to \mathbb{N}^+$ be a function assigning a positive integer to each **K**-atom in $\mathsf{KA}(\mathcal{K}_G)$. For a modal atom $\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G)$, the sets of **K**-atoms $[\mathbf{K}\,\xi]^{\uparrow}$ and $[\mathbf{K}\,\xi]^{\downarrow}$ are defined as follows:

$$[\mathbf{K}\,\xi]^{\downarrow} = \{\mathbf{K}\,\varphi \mid \mathbf{K}\,\varphi \in \mathsf{HA}(\mathcal{K}_G) \textit{ such that } \lambda(\mathbf{K}\,\varphi) \leq \lambda(\mathbf{K}\,\xi)\}$$
$$[\mathbf{K}\,\xi]^{\uparrow} = \mathsf{HA}(\mathcal{K}_G) \setminus [\mathbf{K}\,\xi]^{\downarrow}$$

We say that λ is a *stratification* of $\mathcal{P}_G$ if the following conditions hold:

(1) For each rule $r_G \in \mathcal{P}_G$ of the form (46), $\lambda(\mathbf{K}\,H) \geq \lambda(\mathbf{K}\,B_i)$ for each $1 \leq i \leq m$, and $\lambda(\mathbf{K}\,H) > \lambda(\mathbf{K}\,B_j)$ for each $m+1 \leq j \leq n$.

(2) For each atom $\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G)$ and each subset $S_h \subseteq [\mathbf{K}\,\xi]^{\downarrow}$, if $\mathsf{OB}_{\mathcal{O},S_h} \not\models \xi$, then $\mathsf{OB}_{\mathcal{O},S_h \cup [\mathbf{K}\,\xi]^{\uparrow}} \not\models \xi$ as well.

The program $\mathcal{P}_G$ is *stratified* if a stratification λ exists. A stratification λ partitions $\mathcal{P}_G$ into *strata* $\mathcal{P}_G^1, \ldots, \mathcal{P}_G^\ell$ as follows:

$$\mathcal{P}_G^i = \{r_G \in \mathcal{P}_G \mid \lambda(\mathbf{K}\,H) = i \text{ where } \mathbf{K}\,H \text{ is the head atom of } r_G\}$$

This sequence is often identified with λ and is also called a stratification.

A nonground MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ is *stratified* if $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ is stratified, where $\mathcal{P}_G$ is the ground program obtained from $\mathcal{P}'_G = \mathsf{gr}(\mathcal{P}, \triangle)$ by removing each rule that contains a ground first-order atom $\mathbf{K}\,P(t_1, \ldots, t_n)$ in its body such that P occurs neither in $\mathcal{O}$ nor in the head of some rule in $\mathcal{P}'_G$.

Intuitively, if a ground program $\mathcal{P}_G$ is stratified, then the values of all **not**-atoms in a stratum $\mathcal{P}_G^i$ are fully defined by lower strata, and that no modal atom in $\mathcal{P}_G^i$ changes its value during an evaluation of a higher stratum. Condition (1) of Definition 4.17 corresponds to the case of ordinary programs, and it ensures that evaluating a rule from $\mathcal{P}_G^i$ does not derive a fact from a lower stratum. To understand the rationale behind Condition (2), consider the MKNF knowledge base $\mathcal{K}_G$ where $\mathcal{DL}$ is propositional logic, $\mathcal{O} = \{q \wedge r \supset p\}$, and $\mathcal{P}_G$ contains the rules $\mathbf{K}\,r \leftarrow \mathbf{not}\,p$ and $\mathbf{K}\,q \leftarrow \mathbf{K}\,r$. Without $\mathcal{O}$, the program $\mathcal{P}_G$ is stratified for $\lambda(\mathbf{K}\,p) = 1$, $\lambda(\mathbf{K}\,r) = 2$, and $\lambda(\mathbf{K}\,q) = 3$: we can first derive $\mathbf{K}\,r$ using the first rule, and then derive $\mathbf{K}\,q$ using the second rule. With $\mathcal{O}$, however, after we derive these facts, the atom $\mathbf{K}\,p$ becomes true, which invalidates the antecedent of the first rule. Note that λ does not satisfy Condition (2) of Definition 4.17: for $S = \emptyset \subseteq [\mathbf{K}\,p]^{\downarrow}$ we have $\mathsf{OB}_{\mathcal{O},S} \not\models p$, but $\mathsf{OB}_{\mathcal{O},S \cup [\mathbf{K}\,p]^{\uparrow}} \models p$. Clearly, this condition makes checking whether $\mathcal{K}_G$ is stratified difficult in general. Stratification can be checked in the usual way if (i) no predicate in the head of a rule in $\mathcal{P}_G$ occurs in $\mathcal{O}$ and (ii) if $\mathcal{O}$ or $\mathcal{P}_G$ use equality, then $\mathcal{K}_G$ axiomatizes the unique name assumption. This is an important case because it allows one to use rules to define constraints over a DL knowledge base.

A nonground program $\mathcal{P}$ is stratified if, roughly speaking, its grounding $\mathsf{gr}(\mathcal{P}, \triangle)$ is stratified as well. The condition in Definition 4.17 relaxes this rather strict notion of stratification by removing from $\mathsf{gr}(\mathcal{P}, \triangle)$ the ground rules that are obviously satisfied in each MKNF model of $\mathcal{K}$.

An MKNF model for a stratified MKNF knowledge base $\mathcal{K}_G$ can be computed by processing strata sequentially.

Definition 4.18. Let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be a ground MKNF knowledge base and let $\mathcal{P}_G^1, \ldots, \mathcal{P}_G^\ell$ be a stratification of $\mathcal{P}_G$. The sequence of subsets $U_0, \ldots, U_\ell$ of $\mathsf{HA}(\mathcal{K}_G)$ is inductively defined as

$$U_0 = \emptyset \text{ and}$$

$$\left.\begin{aligned} P_i &= \{\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G) \mid \lambda(\mathbf{K}\,\xi) < i \text{ and } \mathsf{OB}_{\mathcal{O},U_{i-1}} \models \xi\} \\ N_i &= \{\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G) \mid \lambda(\mathbf{K}\,\xi) < i \text{ and } \mathsf{OB}_{\mathcal{O},U_{i-1}} \not\models \xi\} \\ \chi_i &= \{\mathbf{K}\,\xi \leftarrow \quad \mid \mathbf{K}\,\xi \in U_{i-1}\} \cup \mathcal{P}_G^i[\mathbf{not}, P_i, N_i] \\ \mathcal{K}_G^i &= (\mathcal{O}, \chi_i) \\ U_i &= T_{\mathcal{K}_G^i}^\infty \end{aligned}\right\} \text{ for } 1 \leq i \leq \ell.$$

We define $U_{\mathcal{K}_G}^\infty = U_\ell$.

We now show that $U_{\mathcal{K}_G}^\infty$ defines an MKNF model of $\mathcal{K}_G$.

THEOREM 4.19. *For $\mathcal{K}_G$ a stratified ground MKNF knowledge base, let $U_{\mathcal{K}_G}^\infty$ be as specified in Definition 4.18 using any stratification and $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},U_{\mathcal{K}_G}^\infty}\}$.*

—If $M \neq \emptyset$, then M is an MKNF model of $\mathcal{K}_G$.

—If $\mathcal{K}_G$ has an MKNF model, then this model is equal to M.

PROOF. Let $\mathcal{P}_G^1, \ldots, \mathcal{P}_G^\ell$ be the stratification of $\mathcal{P}_G$ used to compute $U_{\mathcal{K}_G}^\infty$. Furthermore, let $\mathcal{P}_G^{\leq i} = \bigcup_{1 \leq j \leq i} \mathcal{P}_G^j$ and $\mathcal{K}_G^{\leq i} = (\mathcal{O}, \mathcal{P}_G^{\leq j})$ for $1 \leq i \leq \ell$, and let $M_i = \{I \mid I \models \mathsf{OB}_{\mathcal{O},U_i}\}$ for $0 \leq i \leq \ell$. Each χ_i is a positive nondisjunctive program so, by Theorem 4.13, it has at most one MKNF model that corresponds to M_i by Theorem 4.16.

Each χ_i contains the rule $\mathbf{K}\,\xi \leftarrow$ if $\mathbf{K}\,\xi \in U_{i-1}$, so $U_{i-1} \subseteq U_i$. We now prove property (*): for each $\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G)$, $i = \lambda(\mathbf{K}\,\xi)$, and each $j > i$, we have $M_i \models \mathbf{K}\,\xi$ if and only if $M_j \models \mathbf{K}\,\xi$, or, equivalently, $\mathsf{OB}_{\mathcal{O},U_i} \models \xi$ if and only if $\mathsf{OB}_{\mathcal{O},U_j} \models \xi$. If $\mathsf{OB}_{\mathcal{O},U_i} \models \xi$, since $U_i \subseteq U_j$, then $\mathsf{OB}_{\mathcal{O},U_j} \models \xi$ as well. Assume that $\mathsf{OB}_{\mathcal{O},U_i} \not\models \xi$, but $\mathsf{OB}_{\mathcal{O},U_j} \models \xi$. But then, since $U_j \setminus U_i \subseteq [\mathbf{K}\,\xi]^\uparrow$, we have $\mathsf{OB}_{\mathcal{O},U_i \cup [\mathbf{K}\,\xi]^\uparrow} \models \xi$, which contradicts Condition (2) of Definition 4.17, so $\mathcal{K}_G$ is not stratified.

The definitions of P_i, N_i, and M_i straightforwardly imply property (**): for each $1 \leq i \leq \ell$ and each $\mathbf{K}\,\xi \in \mathsf{KA}(\mathcal{K}_G)$ such that $\lambda(\mathbf{K}\,\xi) < i$, we have that $M_i \models \mathbf{K}\,\xi$ if and only if $\mathbf{K}\,\xi \in P_i$, and $M_i \not\models \mathbf{K}\,\xi$ if and only if $\mathbf{K}\,\xi \in N_i$.

We now prove by induction on $1 \leq i \leq \ell$ that both claims of this theorem hold for M_i and $\mathcal{K}_G^{\leq i}$. For the base case, note that $\mathcal{K}_G^{\leq 1}$ contains only positive rules, so the claim holds for $i = 1$ by Theorem 4.16. We next consider the induction step.

(Claim 1) Assume that $M_i \neq \emptyset$. Then $M_{i-1} \neq \emptyset$ as well so, by the induction assumption, M_{i-1} is an MKNF model of $\mathcal{K}_G^{\leq i-1}$. Consider now an arbitrary rule $r_G \in \mathcal{P}_G^{\leq i-1}$; the rule is of the form (46). For each atom $\mathbf{K}\,B_j \in \mathsf{body}(r_G)$ or $\mathbf{not}\,B_j \in \mathsf{body}(r_G)$, by Condition (1) of Definition 4.17 we have $\lambda(\mathbf{K}\,H) \geq \lambda(\mathbf{K}\,B_j)$, so $i > \lambda(\mathbf{K}\,B_j)$. Hence, by (*) we have $M_{i-1} \models \mathbf{K}\,B_j$ if and only if $M_i \models \mathbf{K}\,B_j$, which implies that $M_{i-1} \models \mathbf{not}\, B_j$ if and only if $M_i \models \mathbf{not}\, B_j$; but then, $M_{i-1} \models r_G$ implies $M_i \models r_G$, so $M_i \models \mathcal{P}_G^{\leq i-1}$. Consider now an arbitrary ground rule $r_G \in \mathcal{P}_G^i$ of the form (46). For each atom $\mathbf{not}\, B_j \in \mathsf{body}(r_G)$, by Condition (1) of Definition 4.17 we have $\lambda(\mathbf{K}\, H) > \lambda(\mathbf{K}\, B_j)$, so $i > \lambda(\mathbf{K}\, B_j)$; hence, by (*) we have $M_{i-1} \models \mathbf{not}\, B_j$ if and only if $M_i \models \mathbf{not}\, B_j$; together with (**), this implies that $M_i \models r_G$ if and only if $M_i \models r_G[\mathbf{not}, P_i, N_i]$. Thus, $M_i \models \mathcal{P}_G^{\leq i}$ and $M_i \models \mathcal{K}_G^{\leq i}$.

Assume now that some $M'_i \supsetneq M_i$ exists such that $(I', M'_i, M_i) \models \mathcal{K}_G^{\leq i}$ for each $I' \in M'_i$. By (*), then $(I', M'_i, M_{i-1}) \models \mathcal{K}_G^{\leq i-1}$ for each $I' \in M'_i$ as well. Since M_i is an MKNF model of $\mathcal{K}_G^i$, we have $(I', M'_i, M_i) \not\models \mathcal{K}_G^i$ for some $I' \in M'_i$; since $\mathcal{K}_G^i$ and $\mathcal{K}_G^{\leq i}$ coincide on $\mathcal{P}_G^i$ and $\mathcal{O}$, we have $(I', M'_i, M_i) \not\models \chi_i$—that is, $M'_i \not\models \mathbf{K}\, \xi$ but $M_i \models \mathbf{K}\, \xi$ for some $\mathbf{K}\, \xi \in \mathsf{HA}(\mathcal{K}_G)$ such that $i > \lambda(\mathbf{K}\, \xi)$. Since $i > \lambda(\mathbf{K}\, \xi)$ and $U_{i-1} \subseteq U_i$, we have $M_{i-1} \models \mathbf{K}\, \xi$. Now let (P'_i, N'_i) be the partition of $\mathsf{KA}(\mathcal{K}_G^{\leq i-1})$ induced by M'_i, and let $M'' = (I \mid I \models \mathsf{OB}_{\mathcal{O},P'_i})$. By (*), $M_i \models \mathbf{K}\, \xi'$ iff $M_{i-1} \models \mathbf{K}\, \xi'$ whenever $i > \lambda(\mathbf{K}\, \xi')$; furthermore, M'' can be seen as a "projection" of M'_i on the $\mathbf{K}$-atoms of level smaller than i so, since $M'_i \not\models \mathbf{K}\, \xi$, we conclude $M'' \supsetneq M_{i-1}$. Furthermore, by the definition of M'', for each $\mathbf{K}\, \xi' \in \mathsf{KA}(\mathcal{K}_G)$ with $i > \lambda(\mathbf{K}\, \xi')$, we have $M'' \models \mathbf{K}\, \xi'$ iff $M'_i \models \mathbf{K}\, \xi'$. But then $(I', M'_i, M_{i-1}) \models \mathcal{P}_G^{\leq i-1}$ for each $I' \in M'_i$ implies $(I'', M'', M_{i-1}) \models \mathcal{P}_G^{\leq i-1}$ for each $I'' \in M''$; hence, $(I'', M'', M_{i-1}) \models \mathcal{K}_G^{\leq i-1}$ for each $I'' \in M''$ as well, which contradicts the induction assumption that M_{i-1} is an MKNF model of $\mathcal{K}_G^{\leq i-1}$.

(Claim 2) Assume that M'_i is an MKNF model of $\mathcal{K}_G^{\leq i}$. Let (P'_i, N'_i) be a partition of $\mathsf{KA}(\mathcal{K}_G^{\leq i})$ induced by M'_i, let $P = P'_i \cap \mathsf{KA}(\mathcal{K}_G^{\leq i-1})$ and let $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P}\}$. Clearly, $M \models \mathcal{K}_G^{\leq i-1}$. Assume now that M is not an MKNF model of $\mathcal{K}_G^{\leq i-1}$—that is, that an MKNF interpretation $M' \supsetneq M$ exists such that $(I', M', M) \models \mathcal{K}_G^{\leq i-1}$ for each $I' \in M'$. Let (P', N') be the partition $\mathsf{KA}(\mathcal{K}_G^{\leq i-1})$ induced by M', and let $\mathcal{K}'_G = (\mathcal{O}, \mathcal{P}'_G)$ where $\mathcal{P}'_G$ is defined as

$$\mathcal{P}'_G = \{\mathbf{K}\, \xi \leftarrow \mid \mathbf{K}\, \xi \in P' \cap \mathsf{HA}(\mathcal{K}_G^{\leq i})\} \cup \mathcal{P}_G^i[\mathbf{not}, P, N].$$

Finally, let $U = T_{\mathcal{K}'_G}^{\infty}$, let $M'' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},U}\}$, and let (P'', N'') be the partition of $\mathsf{KA}(\mathcal{K}_G^{\leq i})$ induced by M''. Since $\mathcal{P}_G$ is stratified, by an argument analogous to (*) one can show that $P'' \subseteq P'_i$ and $P'' \cap \mathsf{KA}(\mathcal{K}_G^{\leq i-1}) = P'$; but then, since $P' \subsetneq P$, we have $P'' \subsetneq P'_i$, so $M'' \supsetneq M'_i$. Furthermore, $(I'', M'', M'_i) \models \mathcal{K}_G^{\leq i}$ for each $I'' \in M''$, which contradicts the assumption that M'_i is an MKNF model of $\mathcal{K}_G^{\leq i}$; thus, M is an MKNF model of $\mathcal{K}_G^{\leq i-1}$. By the induction assumption $M = M_{i-1}$, which implies that $M'_i \models \mathbf{K}\, \xi$ iff $M_{i-1} \models \mathbf{K}\, \xi$ for each $\mathbf{K}\, \xi \in \mathsf{KA}(\mathcal{K}_G)$ such that $i > \lambda(\mathbf{K}\, \xi)$.

Thus, $M'_i \models \chi_i$ and, consequently, $M'_i \models \mathcal{K}_G^i$. Assume now that some $M''_i \supsetneq M'_i$ exists such that $(I'', M''_i, M'_i) \models \mathcal{K}_G^i$ for each $I'' \in M''_i$. Clearly, $M''_i \models \mathbf{K}\, \xi$ iff $M'_i \models \mathbf{K}\, \xi$ for each $\mathbf{K}\, \xi \in \mathsf{KA}(\mathcal{K}_G)$ such that $i > \lambda(\mathbf{K}\, \xi)$; thus, $(I'', M''_i, M'_i) \models \mathcal{K}_G^{\leq i}$ for each $I'' \in M''_i$, which contradicts the fact that M'_i is an MKNF model of $\mathcal{K}_G^{\leq i}$. Hence, M'_i is an MKNF model of $\mathcal{K}_G^i$, so $M'_i = M_i$ by Theorem 4.13. □

By Lemma 4.9 and Theorem 4.19, $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$ for some modally closed MKNF formula ψ if and only if $\mathsf{OB}_{\mathcal{O},U_{\mathcal{K}_G}^{\infty}} \models \psi[U_{\mathcal{K}_G}^{\infty}]$. Furthermore, we have the following proposition for checking MKNF entailment of negative facts.

PROPOSITION 4.20. *Let $\mathcal{K}_G$ be a stratified ground MKNF knowledge base, and let ψ be of the form $\neg\,\mathbf{K}\,\varphi$ where φ is a closed first-order formula. Then, $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$ iff $\mathcal{K}_G$ is unsatisfiable or $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \mathbf{K}\,\varphi$.*

PROOF. Both directions are trivial if $\mathcal{K}_G$ is MKNF unsatisfiable, so assume that $\mathcal{K}_G$ has a (unique) MKNF model M. For the ($\Rightarrow$) direction, $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$ implies $M \models \psi$, so $M \not\models \neg\psi$ and $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \mathbf{K}\,\varphi$. For the ($\Leftarrow$) direction, $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \mathbf{K}\,\varphi$ implies $M \not\models \mathbf{K}\,\varphi$, which implies $M \models \psi$; since M is the only model of $\mathcal{K}_G$, we conclude that $\mathcal{K}_G \models_{\mathsf{MKNF}} \psi$. □

4.5 Nonstratified Nondisjunctive Programs

We finally consider the case when $\mathcal{K}_G$ is a nondisjunctive MKNF knowledge base for which a stratification does not exist. For such programs, we must guess the subset $P \subseteq \mathsf{KA}(\mathcal{K}_G)$; however, $\mathcal{P}_G[\mathbf{not}, P, N]$ is a positive nondisjunctive MKNF program, for which we can compute an MKNF model as explained in Section 4.3.

Let nondisjunctive-not-entails($\mathcal{K}_G, \psi$) be the function defined as shown in Table IV, but where Condition (5) is satisfied iff $P_h = T^{\infty}_{\mathcal{K}'_G}$, where $\mathcal{K}'_G = (\mathcal{O}, \mathcal{P}_G[\mathbf{not}, P, N])$. We now show that this function is sound and complete for nondisjunctive MKNF knowledge bases.

THEOREM 4.21. *For $\mathcal{K}_G$ a nonstratified nondisjunctive ground MKNF knowledge base and ψ a modally closed MKNF formula,* nondisjunctive-not-entails($\mathcal{K}_G, \psi$) *returns* true *if and only if $\mathcal{K}_G \not\models_{\mathsf{MKNF}} \psi$.*

PROOF. From the proof of Theorem 4.10, one can see that Condition (5) in Table IV ensures that $M = \{I \mid I \models \mathsf{OB}_{\mathcal{O},P_h}\}$ satisfies the preference on MKNF interpretations, which is the case iff M is the MKNF model of $\mathcal{K}'_G = (\mathcal{O}, \mathcal{P}_G[\mathbf{not}, P, N])$. Since $\mathcal{K}'_G$ is positive and nondisjunctive, by Theorem 4.16, its model is characterized by $T^{\infty}_{\mathcal{K}'_G}$. Thus, M satisfies the preference on MKNF interpretations if and only if $T^{\infty}_{\mathcal{K}'_G} = P_h$. □

5. UNDECIDABILITY OF REASONING IN MKNF KNOWLEDGE BASES

An important design requirement for most description logics is decidability of reasoning, as it has been empirically shown that, for decidable DLs, one can often develop optimizations that make DL reasoning suitable for practical usage [Horrocks 1998]. Consequently, decidability of reasoning has been an important requirement for most existing combinations of DLs and ASP as well.

To make reasoning with MKNF knowledge bases decidable, the first-order fragment $\mathcal{DL}$ must obviously be decidable. Furthermore, answer set programming is decidable: the semantics of nonground programs is defined through grounding, which reduces the reasoning problem to the propositional case.[9] We show, however, that, even with a very simple language $\mathcal{DL}$ and safe rules, reasoning with MKNF knowledge bases is undecidable. This may seem unsurprising, since even very simple DLs cannot be extended with safe first-order rules without losing decidability [Levy and Rousset 1998]. This well-known result is, roughly speaking, due to the

[9]In order to justify such a definition, rules are typically required to be safe, as this makes their interpretation independent of the universe $\triangle$.

fact that the existential quantification in the DL knowledge base can be used to axiomatize existence of an infinite chain of objects, which allows one to encode a range of undecidable problems using rules.

As explained in Section 3.4, however, the semantics of MKNF rules is quite different from the semantics of first-order rule formalisms such as CARIN [Levy and Rousset 1998] or SWRL [Horrocks et al. 2005] in the treatment of existential quantifiers. Therefore, the undecidability proofs for first-order formalisms do not carry over to MKNF rules. In fact, undecidability of reasoning with MKNF knowledge bases has different causes: $\mathcal{DL}$ can make some concept A equivalent to the infinite interpretation domain $\triangle$, which makes the rules applicable to unnamed individuals and can be exploited to encode undecidable problems.

Our proofs use reductions from the Domino Tiling problem [Börger et al. 1996]. A *domino system* is a triple $\mathcal{D} = (D, H, V)$, where $D = \{D_1, \ldots, D_n\}$ is a finite set of *domino types*, $H : D \to 2^D$ is the *horizontal compatibility condition*, and $V : D \to 2^D$ is the *vertical compatibility condition.* A *$\mathcal{D}$-tiling* of an infinite grid is a function $t : \mathbb{N} \times \mathbb{N} \to D$ such that $t(i, j+1) \in H(t(i, j))$ and $t(i+1, j) \in V(t(i, j))$ for all $i, j \in \mathbb{N}$. Checking whether a $\mathcal{D}$-tiling exists is undecidable in general [Börger et al. 1996]. In our proofs, we shall identify each domino type D_ℓ with an atomic concept of the same name.

We identify two different sources of undecidability. Theorem 5.1 shows that using conjunctive queries as generalized atoms in rule heads leads to undecidability. The theorem uses the DL $\mathcal{ALCN}$ [Baader et al. 2007], which allows for conjunction and disjunction of concepts, existential and universal quantifiers, and unqualified number restrictions (see Section 2.1).

Theorem 5.1. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be an MKNF knowledge base in which $\mathcal{O}$ is an $\mathcal{ALCN}$ knowledge base and $\mathcal{P}$ contains a single positive safe MKNF rule whose head is a generalized atom consisting of a conjunctive query. Then, checking whether $\mathcal{K}$ is satisfiable is undecidable.*

Proof. Let $\mathcal{D}$ be any domino system, and let $\mathcal{O}_\mathcal{D}$ be the $\mathcal{ALCN}$ knowledge base using the concepts $D_1, \ldots, D_n$ and G, and the roles *hor* and *ver*, containing the following TBox axioms:

$$\top \sqsubseteq\, \leq 1\, hor \tag{47}$$

$$\top \sqsubseteq\, \leq 1\, ver \tag{48}$$

$$\top \sqsubseteq G \tag{49}$$

$$G \sqsubseteq D_1 \sqcup \ldots \sqcup D_n \tag{50}$$

$$D_i \sqcap D_j \sqsubseteq \bot \qquad \text{for each } 1 \leq i < j \leq n \tag{51}$$

$$D_i \sqsubseteq \forall hor. \bigsqcup_{d \in H(D_i)} d \qquad \text{for each } 1 \leq i \leq n \tag{52}$$

$$D_i \sqsubseteq \forall ver. \bigsqcup_{d \in V(D_i)} d \qquad \text{for each } 1 \leq i \leq n \tag{53}$$

Furthermore, let $\mathcal{P}_\mathcal{D}$ be the program containing only the following safe MKNF rule:

$$\mathbf{K}[\exists y, z, w : hor(x, y) \land ver(x, z) \land hor(z, w) \land ver(y, w)] \leftarrow \mathbf{K}\, G(x) \tag{54}$$

Finally, let $\mathcal{K}_\mathcal{D} = (\mathcal{O}_\mathcal{D}, \mathcal{P}_\mathcal{D})$. We now show that $\mathcal{K}_\mathcal{D}$ is satisfiable if and only if a $\mathcal{D}$-tiling exists; this clearly implies the claim of this theorem.

($\Leftarrow$) Let t be a $\mathcal{D}$-tiling, let $\triangle$ be a countably infinite set, and let $\tau : \mathbb{N} \times \mathbb{N} \to \triangle$ be a bijective mapping; we write $\tau(i,j)$ as $\tau_{i,j}$. We define a first-order interpretation I over $\triangle$ as follows.

$$\begin{aligned} G^I &= \triangle \\ D_\ell^I &= \{\tau_{i,j} \mid t(i,j) = D_\ell\} \text{ for each domino tile } D_\ell \\ hor^I &= \{\langle \tau_{i,j}, \tau_{i+1,j}\rangle \mid i,j \in \mathbb{N}\} \\ ver^I &= \{\langle \tau_{i,j}, \tau_{i,j+1}\rangle \mid i,j \in \mathbb{N}\} \end{aligned}$$

It is clear that I satisfies axioms (47)–(50); furthermore, for each $\tau_{i,j} \in \triangle$, we have

$$I \models \exists y, z, w : hor(\tau_{i,j}, y) \wedge ver(\tau_{i,j}, z) \wedge hor(z, w) \wedge ver(y, w). \tag{55}$$

Let M be the maximal set of first-order interpretations satisfying $\mathcal{K}_\mathcal{D}$. Such a set exists because it contains at least I. Clearly, M is an MKNF model of $\mathcal{K}_\mathcal{D}$.

($\Rightarrow$) Let M be a model of $\mathcal{K}_\mathcal{D}$ with a domain $\triangle$, and let I be an arbitrarily chosen first-order interpretation from M (such I exists since $M \neq \emptyset$). We define inductively $\tau_{i,j}$, where i and j are nonnegative integers, as follows.

(1) Let $\tau_{0,0}$ be an arbitrarily chosen element of $\triangle$.

(2) Assume that $\tau_{i,j}$ has been defined. Because of (49), we have $G^I = \triangle$, so $M \models \mathbf{K}\, G(\tau_{i,j})$; furthermore, (54) implies that (55) holds for $\tau_{i,j}$. Thus, we can define $\tau_{i,j+1}$, $\tau_{i+1,j}$, and $\tau_{i+1,j+1}$ by choosing the elements of $\triangle$ for which the following holds:

$$I \models hor(\tau_{i,j}, \tau_{i+1,j}) \wedge ver(\tau_{i,j}, \tau_{i,j+1}) \wedge hor(\tau_{i,j+1}, \tau_{i+1,j+1}) \wedge ver(\tau_{i+1,j}, \tau_{i+1,j+1})$$

These elements in item (2) are uniquely defined because of axioms (47) and (48). Because of (50) and (51), for each $\tau_{i,j}$ there is exactly one domino tile D_ℓ such that $I \models D_\ell(\tau_{i,j})$, so we define a function $t : \mathbb{N} \times \mathbb{N} \to D$ as $t(i,j) = D_\ell$. Because of (52) and (53), it is clear that t is a $\mathcal{D}$-tiling. □

Theorem 5.1 may not come as a surprise: the rule (54) contains in the head a conjunctive query that connects objects in a grid-like manner. It might be therefore tempting to prohibit conjunctive queries in the head in hope of obtaining decidability. The following theorem, however, identifies another source of undecidability: negation as failure can be applied to the entire interpretation domain $\triangle$ even if the rules are safe.

Theorem 5.2. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be an MKNF knowledge base in which $\mathcal{O}$ contains an axiom of the form $\top \sqsubseteq C$ for C an atomic concept, and $\mathcal{P}$ contains safe (not necessarily positive) rules in which all generalized atoms are standard first-order atoms. Then, checking whether $\mathcal{K}$ is satisfiable is undecidable.*

Proof. Let $\mathcal{D}$ be any domino system, and let $\mathcal{K}_\mathcal{D} = (\mathcal{O}_\mathcal{D}, \mathcal{P}_\mathcal{D})$ be the MKNF knowledge base defined as follows. The DL knowledge base $\mathcal{O}_\mathcal{D}$ contains only the following TBox axiom:

$$\top \sqsubseteq G \tag{56}$$

The program $\mathcal{P}_\mathcal{D}$ contains five sets of rules. Rules (57)–(58) ensure that each two objects in $\triangle$ are connected either by *hor* or *not_hor*. Furthermore, rules (59)–(60) ensure that each element is connected to at least one other object by *hor*.

$$\mathbf{K}\,hor(x,y) \vee \mathbf{K}\,not\_hor(x,y) \leftarrow \mathbf{K}\,G(x), \mathbf{K}\,G(y) \tag{57}$$
$$\mathsf{false} \leftarrow \mathbf{K}\,hor(x,y), \mathbf{K}\,not\_hor(x,y) \tag{58}$$
$$\mathbf{K}\,has\_hor(x) \leftarrow \mathbf{K}\,hor(x,y) \tag{59}$$
$$\mathsf{false} \leftarrow \mathbf{K}\,G(x), \mathbf{not}\,has\_hor(x) \tag{60}$$

Similarly, rules (61)–(64) ensure that each object in $\triangle$ is connected to at least one other object by *ver*.

$$\mathbf{K}\,ver(x,y) \vee \mathbf{K}\,not\_ver(x,y) \leftarrow \mathbf{K}\,G(x), \mathbf{K}\,G(y) \tag{61}$$
$$\mathsf{false} \leftarrow \mathbf{K}\,ver(x,y), \mathbf{K}\,not\_ver(x,y) \tag{62}$$
$$\mathbf{K}\,has\_ver(x) \leftarrow \mathbf{K}\,ver(x,y) \tag{63}$$
$$\mathsf{false} \leftarrow \mathbf{K}\,G(x), \mathbf{not}\,has\_ver(x) \tag{64}$$

The following rule "closes" the grid.

$$\mathbf{K}\,hor(z,w) \leftarrow \mathbf{K}\,hor(x,y), \mathbf{K}\,ver(x,z), \mathbf{K}\,ver(y,w) \tag{65}$$

The following rules label each element of the grid with exactly one domino type.

$$\mathbf{K}\,D_1(x) \vee \ldots \vee \mathbf{K}\,D_n(x) \leftarrow \mathbf{K}\,G(x) \tag{66}$$
$$\mathsf{false} \leftarrow \mathbf{K}\,D_i(x), \mathbf{K}\,D_j(x) \qquad \text{for each } 1 \leq i < j \leq n \tag{67}$$

The following rules ensure horizontal and vertical compatibility between grid nodes.

$$\bigvee_{d \in H(D_i)} \mathbf{K}\,d(x) \leftarrow \mathbf{K}\,D_i(x), \mathbf{K}\,hor(x,y) \qquad \text{for each } 1 \leq i \leq n \tag{68}$$
$$\bigvee_{d \in V(D_i)} \mathbf{K}\,d(x) \leftarrow \mathbf{K}\,D_i(x), \mathbf{K}\,ver(x,y) \qquad \text{for each } 1 \leq i \leq n \tag{69}$$

We now show that $\mathcal{K}_\mathcal{D}$ is satisfiable if and only a $\mathcal{D}$-tiling exists.

($\Leftarrow$) Let t be a $\mathcal{D}$-tiling. Furthermore, let $\triangle = \{a_{i,j} \mid i,j \in \mathbb{N}\}$, and let M be the set of first-order interpretations such that each $I \in M$ satisfies the following properties (*), for each $i,j,k,l \in \mathbb{N}$.

$$\begin{array}{ll}
I \models hor(a_{i,j}, a_{k,l}) & \text{iff } k = i+1 \text{ and } l = j \\
I \models not\_hor(a_{i,j}, a_{k,l}) & \text{iff } k \neq i+1 \text{ or } l \neq j \\
I \models ver(a_{i,j}, a_{k,l}) & \text{iff } k = i \text{ and } l = j+1 \\
I \models not\_ver(a_{i,j}, a_{k,l}) & \text{iff } k \neq i \text{ or } l \neq j+1 \\
I \models D_\ell(a_{i,j}) & \text{iff } t(i,j) = D_\ell \\
I \models G(a_{i,j}) \wedge has\_hor(a_{i,j}) \wedge has\_ver(a_{i,j}) &
\end{array}$$

Clearly, $M \neq \emptyset$, $M \models \mathcal{O}_\mathcal{D}$, and $M \models \mathcal{P}_\mathcal{D}$. Assume now that some $M' \supsetneq M$ exists such that $(I', M', M) \models \mathcal{K}_\mathcal{D}$ for each $I' \in M'$. By (56), we have $G^{I'} = \triangle$, so $M' \models G(a_{i,j})$ for each $i,j \in \mathbb{N}$. Since M is the maximal set of first-order interpretations satisfying (*), there is at least one property ξ from the set of properties (*) such that $M \models \xi$, but $M' \not\models \xi$ for some $i,j \in \mathbb{N}$. If $\xi = hor(a_{i,j}, a_{i+1,j})$ or $\xi = not\_hor(a_{i,j}, a_{k,l})$, then (57) is not true in M'; similarly, if $\xi = ver(a_{i,j}, a_{i+1,j})$

or $\xi = \mathit{not\_ver}(a_{i,j}, a_{k,l})$, then (61) is not true in M'. If $\xi = \mathit{has\_hor}(a_{i,j})$, then (59) is not true in M'; similarly, if $\xi = \mathit{has\_ver}(a_{i,j})$, then (63) is not true in M'. Finally, if $\xi = D_\ell(a_{i,j})$, then (66) is not true in M'. Hence, such M' does not exist, so M is an MKNF model of $\mathcal{K}_\mathcal{D}$.

($\Rightarrow$) Let M be an MKNF model of $\mathcal{K}_\mathcal{D}$ with a domain set $\triangle$. By (56), we have $M \models G(\alpha)$ for each $\alpha \in \triangle$. We define $\tau_{i,j}$, where i and j are nonnegative integers, inductively as follows.

(1) Let $\tau_{0,0}$ be an arbitrarily chosen element of $\triangle$.

(2) Assume that $\tau_{i,j}$ has been defined. Because of (60), we have $M \models \mathit{has\_hor}(\tau_{i,j})$. The rule (59) is the only one that contains $\mathit{has\_hor}$ in the head, so $\mathit{has\_hor}(\tau_{i,j})$ must be derived through (59) for $x = \tau_{i,j}$. Hence, we can pick some $\tau_{i+1,j}$ such that $M \models \mathit{hor}(\tau_{i,j}, \tau_{i+1,j})$. Similarly, because of (63) and (64), we can pick some $\tau_{i,j+1}$ such that $M \models \mathit{ver}(\tau_{i,j}, \tau_{i,j+1})$. Finally, by repeating the argument for $\tau_{i+1,j}$, we can pick some $\tau_{i+1,j+1}$ such that $M \models \mathit{ver}(\tau_{i+1,j}, \tau_{i+1,j+1})$. By rule (65), $M \models \mathit{hor}(\tau_{i,j+1}, \tau_{i+1,j+1})$.

Because of (66) and (67), for each $\tau_{i,j}$ there is exactly one domino tile D_ℓ such that $M \models D_\ell(\tau_{i,j})$, so we define a function $t : \mathbb{N} \times \mathbb{N} \to D$ as $t(i,j) = D_\ell$. Because of (68) and (69), it is clear that t is a $\mathcal{D}$-tiling. $\square$

We used in the proof of Theorem 5.2 disjunctive rules for the sake of clarity. Eiter et al. [2004] showed that positive disjunctive rules can be transformed to nondisjunctive rules with negation as failure. By applying this transformation to (57), (61), (66), (68), and (69), Theorem 5.2 can be sharpened to the case when $\mathcal{P}$ is a nondisjunctive program.

We finish this section with a note that, in their unrestricted form, most nonmonotonic formalisms are not even semidecidable. We conjecture that the same is the case for MKNF knowledge bases; however, we do not have a formal proof yet.

6. DECIDABLE CASES AND COMPUTATIONAL COMPLEXITY

We now investigate the possibilities for obtaining a decidable formalism. In Section 6.1, we propose the notions of DL-safety and admissibility that are sufficient to make reasoning decidable. We then investigate the complexity of reasoning with DL-safe MKNF knowledge bases. In particular, we consider combined complexity in Section 6.2 and data complexity in Section 6.3.

6.1 DL-Safety and Admissibility

Undecidability of reasoning with MKNF knowledge bases arises because rules can be applied to all objects in the infinite domain $\triangle$. To the best of our knowledge, such problems are solved in all existing combinations of DLs and rules by employing some form of syntactic safety of the rules. The main idea is to restrict the applicability of the rules only to the explicitly named part of $\triangle$—that is, only to the individuals that are explicitly mentioned by name in the knowledge base. We now adapt this notion to MKNF knowledge bases.

Definition 6.1 *(DL-Safety).* We assume that the signature Σ contains a subset $\Sigma_{DL} \subseteq \Sigma$ such that $\approx \in \Sigma_{DL}$. We call the predicates in Σ_{DL} *DL-predicates* and assume that $\mathcal{DL}$ refers only to such predicates; furthermore, we call the predicates in $\Sigma \setminus \Sigma_{DL}$ *non-DL-predicates.*

A generalized atom ξ is a *DL-atom* if it contains only predicates of Σ_{DL}; furthermore, ξ is a *non-DL-atom* if it is of the form $(\neg)P(t_1, \ldots, t_n)$ where P is a non-DL-predicate.[10] A modal atom $\mathbf{K}\,\xi$ is a DL-**K**-atom or a non-DL-**K**-atom if ξ has the respective property. Similarly, a modal atom $\mathbf{not}\,\xi$ is a DL-**not**-atom or a non-DL-**not**-atom if ξ has the respective property.

An MKNF rule r is *DL-safe* if each modal atom in it is a DL-**K**-atom, a DL-**not**-atom, a non-DL-**K**-atom, or a non-DL-**not**-atom, and if each variable in r occurs in the body of r in some non-DL-**K**-atom. An MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ is *DL-safe* if each rule $r \in \mathcal{P}$ is DL-safe.

Consider again the MKNF knowledge base from Section 3.5. Rule (37) is not DL-safe: both *seasideCity* and *portCity* are DL-predicates, so the variable x does not occur in a body non-DL-**K**-atom. The rule can be made DL-safe if we introduce a special non-DL-predicate O, add the assertion $\mathbf{K}\,O(a)$ for each individual a, and modify the rule into

$$\mathbf{K}\,\mathit{seasideCity}(x) \leftarrow \mathbf{K}\,\mathit{portCity}(x), \mathbf{not}\,\neg\mathit{seasideCity}(x), \mathbf{K}\,O(x). \tag{70}$$

The atom $\mathbf{K}\,O(x)$ acts as a guard that makes the rule applicable only to the individuals explicitly mentioned by name in the knowledge base. Motik et al. [2005] have discussed in-depth the semantic effects of such a transformation in the case of first-order rules. In MKNF, the rules usually cannot be applied to the individuals that are not explicitly named (see the discussion in Section 3.4) so, for all practical intents and purposes, this transformation does not affect the semantics of MKNF rules significantly.

A DL-safe knowledge base can be grounded w.r.t. the set of constants occurring in it without affecting its semantics.

Proposition 6.2. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be a DL-safe MKNF knowledge base, and let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ be the ground MKNF knowledge base where $\mathcal{P}_G = \mathsf{gr}(\mathcal{P}, O_\mathcal{K})$ and $O_\mathcal{K}$ is as specified in Definition 3.3. Then, the MKNF models of $\mathcal{K}$ and $\mathcal{K}_G$ coincide.*

Proof. We first introduce several definitions. The $\triangle$-*equivalence* $\sim$ induced by an MKNF interpretation M is the relation on $\triangle$ such that, for each $a, b \in \triangle$, we have $a \sim b$ iff $M \models \mathbf{K}\,a \approx b$. Clearly, $\sim$ is an equivalence relation on $\triangle$. For each $a \in \triangle$, let $a_\sim$ be a constant chosen from the equivalence class of a in $\sim$ such that the constants from $O_\mathcal{K}$ are preferred over the constants from $\triangle \setminus O_\mathcal{K}$. For α a generalized atom or a rule, $\alpha_\sim$ is the result of replacing in α each constant a with $a_\sim$. Finally, $\Gamma_\sim$ is the set containing each ground generalized non-DL-atom ξ such that $\xi_\sim$ contains a constant from $\triangle \setminus O_\mathcal{K}$.

($\Rightarrow$) Let M be an MKNF model of $\mathcal{K}$, and let $\sim$ be the $\triangle$-equivalence induced by M. We next show the property (*): $M \not\models \mathbf{K}\,A$ for each $A \in \Gamma_\sim$. Assume that $M \models \mathbf{K}\,A$ for some $A \in \Gamma_\sim$ and consider $M' = M \cup \{I\}$ where I is an interpretation obtained by choosing any interpretation from M and flipping the truth value of each $B \in \Gamma_\sim$ such that $M \models \mathbf{K}\,B$. Now $M' \models \mathbf{K}\,\mathcal{O}$ since all $B \in \Gamma_\sim$ are non-DL-atoms. Furthermore, for each $B \in \Gamma_\sim$, since B is a possibly negated first-order

[10]Note that atoms containing both DL- and non-DL-predicates are neither DL- nor non-DL-atoms.

atom, $M' \not\models \mathbf{K}\, B$. Finally, for each ground generalized atom B such that $B \notin \Gamma_\sim$, we have $M \models \mathbf{K}\, B$ iff $M' \models \mathbf{K}\, B$. Consider now an arbitrary rule $r \in \mathcal{P}$ and its arbitrary ground instance r_G w.r.t. $\triangle$. If r_G does not contain an atom from $\Gamma_\sim$, then clearly $M' \models r_G$. Otherwise, since r is DL-safe, r_G contains an atom $\mathbf{K}\, B$ in the body such that $B \in \Gamma_\sim$; furthermore, $M' \not\models \mathbf{K}\, B$, so $M' \models r_G$ as well. Hence, $(I', M', M) \models \mathcal{K}$ for each $I' \in M'$, which contradicts our assumption that M is an MKNF model of $\mathcal{K}$. Thus, property (*) holds.

Clearly, $M \models \mathcal{K}_G$. Assume now that M is not an MKNF model of $\mathcal{K}_G$—that is, that an MKNF interpretation $M'' \supsetneq M$ exists such that $(I'', M'', M) \models \mathcal{K}_G$ for each $I'' \in M''$. Now $\mathcal{K}_G$ does not contain a constant from $\triangle \setminus O_\mathcal{K}$, so we can make the following assumption (**): if $M'' \models A$ for some generalized atom A, then $M'' \models B$ for each generalized atom B such that $A_\sim = B_\sim$. Due to (*) and $M'' \supsetneq M$, we have $M'' \not\models \mathbf{K}\, A$ for each $A \in \Gamma_\sim$. Consider now an arbitrary rule $r \in \mathcal{P}$ and its arbitrary ground instance r_G w.r.t. $\triangle$. We have the following possibilities.

—r_G contains an atom A such that $A \in \Gamma_\sim$. Since r is DL-safe, r_G contains an atom $\mathbf{K}\, B$ in the body such that $B \in \Gamma_\sim$. Since $M'' \not\models \mathbf{K}\, B$, we have $(I'', M'', M) \models r_G$ for each $I'' \in M''$.

—r_G does not contain an atom from $\Gamma_\sim$. Let $r'_G = (r_G)_\sim$; furthermore, let p and p' be the ground rules obtained from r_G and r'_G, respectively, by replacing each **not**-atom with its value in M. Since $(I'', M'', M) \models \mathcal{P}_G$ for each $I'' \in M''$, we have $(I'', M'', M) \models p'$ for each $I'' \in M''$ as well. Due to (**), we then have $(I'', M'', M) \models p$ for each $I'' \in M''$ as well.

Thus, $(I'', M'', M) \models \mathcal{K}$ for each $I'' \in M''$, which contradicts the assumption that M is an MKNF model of $\mathcal{K}$.

($\Leftarrow$) Let M be an MKNF model of $\mathcal{K}_G$, and let $\sim$ be the $\triangle$-equivalence induced by M. Since the constants from $\triangle$ do not occur in $\mathcal{K}_G$, we have $M \not\models \mathbf{K}\, A$ for each $A \in \Gamma_\sim$. Consider now an arbitrary rule $r \in \mathcal{P}$ and its ground instance r_G w.r.t. $\triangle$. We have the following possibilities.

—r_G contains an atom A such that $A \in \Gamma_\sim$. Since r is DL-safe, then r_G contains an atom $\mathbf{K}\, B$ in the body such that $B \in \Gamma_\sim$. Since $M \not\models \mathbf{K}\, B$, we have $M \models r_G$.

—r_G does not contain an atom from $\Gamma_\sim$. Let $r'_G = (r_G)_\sim$. Since $r'_G \in \mathcal{P}_G$ and $M \models \mathcal{P}_G$, we clearly have $M \models r_G$.

Thus, $(I, M, M) \models \mathcal{K}$ for each $I \in M$. Assume now that an MKNF interpretation $M' \supsetneq M$ exists such that $(I', M', M) \models \mathcal{K}$ for each $I' \in M'$. Then clearly $(I', M', M) \models \mathcal{K}_G$ as well, which contradicts the assumption that M is an MKNF model of $\mathcal{K}_G$. □

We now formulate precise conditions under which reasoning with MKNF knowledge bases is decidable.

Definition 6.3 *(Admissibility)*. Let $\mathcal{DL}$ be a description logic, let $\mathcal{B}$ be a generalized atom base, and let $\mathcal{H}$ be a subset of $\mathcal{B}$. Then, $\mathcal{DL}$, $\mathcal{B}$, and $\mathcal{H}$ are *admissible* if, for each $\mathcal{O} \in \mathcal{DL}$, each finite set $S \subseteq \mathcal{H}$ of ground generalized atoms, each finite set N of assertions of the form $a \not\approx b$, and each generalized atom $\xi \in \mathcal{B}$, checking whether $\mathcal{O} \cup S \cup N \models \xi$ is decidable. Let $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ denote the computational complexity of the latter problem.

An MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ over $\mathcal{DL}$ and $\mathcal{B}$ is *admissible* if $\mathcal{DL}$, $\mathcal{B}$, and $\mathcal{H}$ are admissible, $\mathcal{K}$ is DL-safe and finite, and each rule in $\mathcal{P}$ contains only the generalized atoms from $\mathcal{H}$ in its head.

We shall use the function $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ defined in Table IV to obtain a decision procedure for reasoning with admissible MKNF knowledge bases. In Conditions (2), (3), and (5a) we need to decide MKNF entailments of the form $\mathcal{O} \cup S \models \xi$, where S is a set of ground generalized atoms, and ξ is a ground generalized atom. A minor problem is that S can contain both DL- and non-DL-atoms, which is not directly supported by $\mathcal{DL}$. These problems, however, can be reduced to standard entailments supported by $\mathcal{DL}$. For an equivalence relation $\sim$ on the constants of the signature of $\mathcal{K}_G$ and a constant a, let $a_\sim$ be a uniquely chosen constant from the equivalence class of a. For a generalized atom α, let $\alpha_\sim$ be the result of replacing in α each constant a with $a_\sim$; furthermore, let $(\mathbf{K}\,\xi)_\sim = \mathbf{K}\,\xi_\sim$ and $(\mathbf{not}\,\xi)_\sim = \mathbf{not}\,\xi_\sim$. For S a set of generalized or modal atoms, let $S_\sim = \{\alpha_\sim \mid \alpha \in S\}$. Finally, for a DL knowledge base $\mathcal{O}$, let $\mathcal{O}_\sim$ be the result of (i) replacing in $\mathcal{O}$ each constant a with $a_\sim$ and (ii) appending an axiom $a_\sim \not\approx b_\sim$ for each $a_\sim \neq b_\sim$.

Proposition 6.4. *Let $\mathcal{O}$ be a DL knowledge base, let S be a set of ground generalized atoms, and let ξ be a ground generalized atom such that ξ and all atoms in S are either DL- or non-DL-atoms. Furthermore, let S' and S'' be the subsets of DL-atoms and non-DL-atoms of S, respectively. Then, $\mathcal{O} \cup S \not\models \xi$ iff an equivalence relation $\sim$ on the constants in $\mathcal{O} \cup S \cup \{\xi\}$ exists such that*

(1) $\mathcal{O}_\sim \cup S'_\sim$ is satisfiable, and
(2) $S''_\sim$ does not contain a complementary pair of non-DL-atoms, and
(3) $\mathcal{O}_\sim \cup S'_\sim \not\models \xi_\sim$ if ξ is a DL-atom, or $\xi_\sim \notin S''_\sim$ if ξ is a non-DL-atom.

Proof. It is trivial to see that $\mathcal{O} \cup S \not\models \xi$ if and only if an equivalence $\sim$ on the constants exists such that $\mathcal{O}_\sim \cup S_\sim \not\models \xi_\sim$. If the latter condition is true, then $\mathcal{O}_\sim \cup S'_\sim$ and $S''_\sim$ must be satisfiable, and $\xi_\sim$ follows from either the DL- or the non-DL-part of $\mathcal{O}_\sim \cup S_\sim$. If ξ is a DL-atom, then we must check $\mathcal{O}_\sim \cup S'_\sim \not\models \xi_\sim$; otherwise, if ξ is a non-DL-atom, we can simply check whether $\xi_\sim \notin S''_\sim$. The converse direction can be easily proved in the same manner. $\square$

Hence, if only positive generalized atoms are used in $\mathcal{P}$, then $\mathcal{DL}$ must support standard ABoxes; if $\mathcal{P}$ contains also negative generalized atoms, then $\mathcal{DL}$ must support negative ABox assertions; if $\mathcal{P}_G$ contains generalized atoms consisting of conjunctive queries in the body, then $\mathcal{DL}$ must additionally support answering conjunctive queries. Finally, if $\mathcal{P}_G$ contains generalized atoms consisting of conjunctive queries in the head, then the set S can contain Boolean conjunctive queries. We can decide $\mathcal{O} \cup S \models \xi$ by skolemizing the queries in S—that is, by replacing in each query each nondistinguished variable with a fresh individual and dropping the existential quantifier.

We are now ready to show that reasoning with admissible MKNF knowledge bases is decidable.

Theorem 6.5. *For an admissible MKNF knowledge base $\mathcal{K}$ and a ground generalized atom A, checking whether $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\,\mathbf{K}\,A$ is decidable.*

PROOF. Since $\mathcal{K}$ is DL-safe, by Proposition 6.2 $\mathcal{K}$ has the same MKNF models as its grounding $\mathcal{K}_G$ w.r.t. $O_\mathcal{K}$. Furthermore, $\mathcal{K}$ is finite, so $\mathcal{K}_G$ is finite as well, and so is the set $\mathsf{KA}(\mathcal{K}_G)$ (see Definition 4.1). Thus, we can decide $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ using the function $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ defined in Table IV, provided that the first-order problems in Conditions (1), (2), (3), and (5a) are decidable. All of these problems involve deciding entailments of the form $\mathcal{O} \cup S \cup N \models \xi$, where S is a set of ground generalized atoms from $\mathsf{HA}(\mathcal{K}_G)$, N is a set of ground inequalities, and ξ is a ground generalized atom from $\mathsf{KA}(\mathcal{K}_G)$. Since $\mathcal{K}$ is DL-safe, these problems can be solved using Proposition 6.4; furthermore, since $\mathcal{K}$ is admissible, these checks are decidable. □

6.2 Combined Complexity

In this section, we determine the *combined complexity* bounds of checking whether $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$, where $\mathcal{K}$ is an admissible MKNF knowledge base and $A \in \mathcal{B}$ is a ground generalized atom. Such complexity is measured in $|\mathcal{K}| + |A|$ (i.e., the size of both the knowledge base and the query), and it clearly depends on the complexity of reasoning in $\mathcal{DL}$. $\mathcal{SHOIN}$ is a widely used DL, mainly because it provides the formal underpinning of the OWL DL variant of the Web Ontology Language (OWL) [Patel-Schneider et al. 2004]. This DL is known to be NExpTime-complete [Tobies 2001], so we provide complexity estimates based on the assumption that $\mathcal{DL}$ is an NExpTime-complete language. Furthermore, we show that, even if $\mathcal{DL}$ is of lower complexity, the combined complexity of reasoning with MKNF knowledge bases stays the same, so we do not consider simpler DLs explicitly.

PROPOSITION 6.6. *For $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ an admissible MKNF knowledge base, deciding $\mathcal{K} \not\models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is $\text{NExpTime}^{\text{NP}}$-complete in $|\mathcal{K}| + |A|$, provided that the DL-predicates are bounded in arity and $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} \subseteq$ NExpTime.*

PROOF. By Proposition 6.2, the MKNF models of $\mathcal{K}$ and its grounding $\mathcal{K}_G$ w.r.t. $O_\mathcal{K}$ coincide. Thus, we first compute the grounding $\mathcal{P}_G$ of $\mathcal{P}$, which requires exponential time. By Theorem 4.10, we can check $\mathcal{K}_G \not\models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ using $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$. Thus, we guess a subset $P \subseteq \mathsf{KA}(\mathcal{K}_G)$, which requires nondeterministic exponential time. Condition (1) holds iff $\mathsf{OB}_{\mathcal{O},P_h} \not\models \mathsf{false}$, and Condition (3) holds iff $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi$ for exponentially many ξ. All of these checks can be solved using Proposition 6.4: we guess an equivalence $\sim$ on the constants in $O_\mathcal{K}$, we separate the atoms from P_h into DL-$\mathbf{K}$-atoms P'_h and non-DL-$\mathbf{K}$-atoms P''_h, and we then check whether $\mathsf{OB}_{\mathcal{O}_\sim,P'_{h\sim}}$ is satisfiable, whether $\{\mathbf{K}\, A, \mathbf{K}\, \neg A\} \not\subseteq P''_{h\sim}$ for each ground first-order non-DL-atom A, and whether $\mathsf{OB}_{\mathcal{O}_\sim,P'_{h\sim}} \not\models \xi_\sim$ or $\mathbf{K}\, \xi_\sim \notin P''_{h\sim}$. By the assumption on the complexity of reasoning in $\mathcal{DL}$, these checks can be performed in nondeterministic exponential time. Condition (4) can be checked in exponential time. Furthermore, we can check complements of Conditions (2) and (5) using an oracle that receives P_h as input. In the worst case, $P_h = \mathsf{HA}(\mathcal{K}_G)$, so P_h can be exponential in $|\mathcal{K}|$; furthermore, if $|P_h|$ is not exponential in $|\mathcal{K}|$, we can pad it to make it exponential. Thus, the input of the oracle is always exponential in $|\mathcal{K}|$. Since the arity of the predicates in $\mathcal{DL}$ is bounded, the number of such different ground atoms is polynomial in $|\mathcal{K}|$. Hence, the set P'_h of the predicates from $\mathcal{DL}$ is polynomial in $|\mathcal{K}|$. Thus, Conditions (2) and (5a) can be checked in nondeterministic, and Condition (5b) in polynomial time *in the size of the oracle* *input*, so the oracle runs in NP. Finally, if the query is of the form $\mathbf{K}\, A$, then Condition (6) holds if $\mathsf{OB}_{\mathcal{O},P_h} \not\models A$, which can be decided in the same way as Condition (3); in contrast, if the query is of the form $\neg\, \mathbf{K}\, A$, then Condition (6) holds if $\mathsf{OB}_{\mathcal{O},P_h} \models A$, which can be decided in the same way as Condition (2). Hence, our algorithm can be implemented in $\text{NExpTime}^{\text{NP}}$. Hardness is inherited from disjunctive datalog under stable model semantics [Eiter et al. 1997]. □

W3C has recently released a revision of OWL DL, called OWL 2. The formal underpinnings of OWL 2 are provided by the DL $\mathcal{SROIQ}$ [Kutz et al. 2006], which was recently shown to be N2ExpTime-complete [Kazakov 2008]. The upper bound of the complexity of reasoning with an MKNF knowledge base is then given by the following lemma. For the sake of brevity, we do not present the lower bound; however, we conjecture that the lower bound actually matches the upper one.

Proposition 6.7. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be an admissible MKNF knowledge base such that the DL-predicates are bounded in arity and $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} \subseteq$ N2ExpTime. Then, $\mathcal{K} \not\models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ can be decided by solving a problem in* N2ExpTime *and another problem in co*N2ExpTime.

Proof. As in the proof of Proposition 6.6, $\mathsf{HA}(\mathcal{K}_G)$ is exponential in $|\mathcal{K}|$, so the number of different sets P and P' that $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ might need to examine is doubly exponential. We can check Conditions (1) and (3) by solving a doubly exponential number of problems in N2ExpTime, which is equivalent to solving just one problem in N2ExpTime. Similarly, we can check Conditions (2) and (5a) by solving a doubly exponential number of problems in coN2ExpTime, which is equivalent to solving just one problem in coN2ExpTime. Condition (6) can be added to one of these two groups, depending on the polarity of the query. Finally, Conditions (4) and (5b) can be checked in doubly exponential time. □

As we discuss in Section 6.3, in typical applications of MKNF knowledge bases, we expect the data complexity to provide a more useful complexity estimate. Hence, for the sake of brevity, we do not provide combined complexity estimates for the cases when MKNF rules have simpler structure.

6.3 Data Complexity

As we discussed in Section 6.1, DL-safety makes the rules of an MKNF knowledge base applicable only to the explicitly named individuals. Therefore, the primary applications of MKNF rules are in answering queries over individuals, rather than modeling the conceptual part of a knowledge base. In such applications, the size of the data is usually several orders of magnitude larger than the size of the conceptual part, so *data complexity*—a complexity estimate under the assumption that the rules are fixed but the data varies—may be a much better indicator of how an algorithm scales to large amounts of data [Vardi 1982]. Therefore, in this section, we conduct a detailed data complexity analysis of reasoning with admissible MKNF knowledge bases for different types of MKNF rules.

For a ground generalized atom $A \in \mathcal{B}$ and an admissible MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ where $\mathcal{O}$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$, data complexity of $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is the complexity of the problem measured under the assumption that (i) $\mathcal{A}$ is *extensionally reduced*—that is, it contains only atomic formulae,

Table V. Data Complexity of Entailment Checking in Admissible MKNF KBs

	$\vee$	**not**	no $\mathcal{DL}$	$\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ = PTime	$\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ = *co*NP
1	no	no	PTime	PTime	coNP/coDP
2	no	stratified	PTime	PTime	Δ_2^p
3	no	yes	coNP	coNP	Π_2^p
4	yes	no	coNP/Π_2^p	coNP/Π_2^p	coNP/Π_2^p
5	yes	yes	Π_2^p	Π_2^p	Π_2^p

Note: The second column and third column determine the expressivity of the rule component (i.e., whether the rules are disjunctive and whether they contain **not**), and the fourth, fifth, and sixth column show the data complexity of MKNF entailment checking when the DL component is absent, and when it is complete for PTime and *co*NP w.r.t. data complexity, respectively.

(*ii*) the sizes of the TBox $\mathcal{T}$, the query A, and the rules in $\mathcal{P}$ that are not facts are bounded, and (*iii*) the size of A is bounded.

Data complexity of reasoning with admissible MKNF knowledge bases clearly depends on the data complexity $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ of the first-order problems $\mathcal{O} \cup S \cup N \models \xi$, where $\mathcal{O} \in \mathcal{DL}$, $S \subseteq \mathcal{H}$, N is a set of ground inequalities, and $\xi \in \mathcal{B}$. Data complexity of answering conjunctive queries in the DL $\mathcal{SHOIQ}$ is coNP-complete [Ortiz et al. 2008]; furthermore, expressive DL fragments (e.g., Horn-$\mathcal{SHIQ}$ [Hustadt et al. 2007], $\mathcal{EL}^{++}$ [Baader et al. 2005]) have been proposed for which the assertion checking problem can be solved in polynomial time. Most DLs used nowadays can be classified into one of these two categories, so in this section we consider the cases when $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ is PTime and coNP. There are DLs that have even lower data complexity; for example, answering conjunctive queries in DL-Lite is data complete for LogSpace [Calvanese et al. 2007]. For MKNF knowledge bases with such DLs, the data complexity of reasoning is dominated by the rules, so the overall complexity then coincides with the case when $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ is PTime.

Table V summarizes the data complexity of checking whether $\mathcal{K} \models_{\mathsf{MKNF}} \psi$, where $\psi = (\neg)\,\mathbf{K}\,A$ and $A \in \mathcal{B}$ is a ground generalized atom. If the complexity differs for $\psi = \mathbf{K}\,A$ and $\psi = \neg\,\mathbf{K}\,A$, the table contains two entries. All results are completeness results. For comparison, the table also shows the well-known complexity results for ordinary answer set programming, cf. [Eiter et al. 1997; Dantsin et al. 2001].

In our hardness proofs for $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = co\text{NP}$, we present reductions where $\mathcal{DL}$ is the logic of function-free universally-quantified rules interpreted in first-order logic. It is well known that checking entailment of ground facts for such a $\mathcal{DL}$ is data complete for coNP. Furthermore, we often use a simple encoding of a propositional formula φ into a set of universally-quantified first-order implications $\Upsilon \cup \mu(\varphi)$ as defined in Table VI, which exhibits the following useful property.

Lemma 6.8. *A propositional formula φ is satisfiable if and only if*

$$\mathbf{K}\,[\Upsilon \cup \mu(\varphi)] \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,F(u_\varphi).$$

Proof. Each subformula ψ of φ is represented in $\mu(\varphi)$ by a distinct constant u_ψ. The implications in Υ ensure that, whenever $I \models \mu(\varphi) \cup \Upsilon$ for some first-order interpretation I, either $I \models T(u_\psi)$ or $I \models F(u_\psi)$ but not both, and that $T(u_\psi)$ and $F(u_\psi)$ are defined according to the semantics of Boolean connectives.

Table VI. Definitions of μ and Υ

$\mu(x)$	$= \{var(u_x)\}$
$\mu(\psi_1 \land \psi_2)$	$= \{and(u_\varphi, u_{\psi_1}, u_{\psi_2}), var(u_\varphi), var(u_{\psi_1}), var(u_{\psi_2})\} \cup \mu(\psi_1) \cup \mu(\psi_2)$
$\mu(\psi_1 \lor \psi_2)$	$= \{or(u_\varphi, u_{\psi_1}, u_{\psi_2}), var(u_\varphi), var(u_{\psi_1}), var(u_{\psi_2})\} \cup \mu(\psi_1) \cup \mu(\psi_2)$
$\mu(\neg\psi)$	$= \{not(u_\varphi, u_\psi), var(u_\varphi), var(u_\psi)\} \cup \mu(\psi)$

Note: u_ψ is a constant that is unique and distinct for each formula ψ.

Υ	
$T(x) \lor F(x) \subset var(x)$	$F(y) \subset not(x,y) \land T(x)$
$\mathsf{false} \subset T(x) \land F(x)$	$T(x) \subset not(x,y) \land F(y)$
$T(x) \subset and(x,y,z) \land T(y) \land T(z)$	$T(x) \subset or(x,y,z) \land T(y)$
$T(y) \subset and(x,y,z) \land T(x)$	$T(x) \subset or(x,y,z) \land T(z)$
$T(z) \subset and(x,y,z) \land T(x)$	$T(y) \lor T(z) \subset or(x,y,z) \land T(x)$

Note: All implications are implicity universally quantified.

Assume now that φ is satisfiable, and consider an arbitrary assignment for the propositional variables of φ that satisfies φ. Let I be the first-order interpretation such that $I \models T(u_\psi)$ iff the subformula ψ of φ is true in the assignment. It is straightforward to see that $I \models \mu(\varphi) \cup \Upsilon$ and $I \models T(u_\varphi)$, so $I \not\models F(u_\varphi)$. Let $M = \{I \mid I \models \mu(\varphi) \cup \Upsilon\}$. Since $I \in M$, we have $M \neq \emptyset$, so M is the only MKNF model of $\mathbf{K}\,[\Upsilon \cup \mu(\varphi)]$; furthermore, $M \models \neg\,\mathbf{K}\,F(u_\varphi)$.

Conversely, if $\Upsilon \cup \mu(\varphi) \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,F(u_\varphi)$, then an MKNF model M of $\Upsilon \cup \mu(\varphi)$ and an interpretation $I \in M$ exist such that $I \not\models F(u_\varphi)$. But then, I defines an assignment for the propositional variables of φ that satisfies φ. $\square$

We now prove the data complexity claims from Table V for different types of MKNF knowledge bases. For easier reference, we label each proposition with the row of the table that the proposition pertains to. For example, $(1 + \text{PTime})$ means that the proposition refers to row 1 of the table and the case $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = \text{PTime}$. In all membership results, the first step is to compute the grounding $\mathcal{P}_G$ of $\mathcal{P}$ w.r.t. $O_\mathcal{K}$; since the nonground rules in $\mathcal{P}$ are bounded in size, $|\mathcal{P}_G|$ is polynomial in $|\mathcal{P}|$.

Proposition 6.9 $(1 + \text{PTime})$. *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = \text{PTime}$ and $\mathcal{P}$ a nondisjunctive positive program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\,\mathbf{K}\,A$ is* PTime*-complete in data complexity.*

Proof. By Theorem 4.16, we can perform the check by computing $\psi[T^\infty_{\mathcal{K}_G}]$, where $\psi = (\neg)\,\mathbf{K}\,A$. The computation of $T^\infty_{\mathcal{K}_G}$ requires at most $|\mathcal{P}_G|$ calls to an oracle running in PTime, and to compute $\psi[T^\infty_{\mathcal{K}_G}]$ we need another call to the oracle. Hence, the algorithm can be implemented in PTime. Hardness carries over from positive datalog programs, cf. [Dantsin et al. 2001]. $\square$

Proposition 6.10 $(1 + co\text{NP})$. *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = co\text{NP}$ and $\mathcal{P}$ a nondisjunctive positive program, deciding whether $\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\,A$ is co*NP*-complete, and deciding whether $\mathcal{K} \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,A$ is co*DP*-complete in data complexity.*

Proof. $(\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\,A)$ Membership in coNP carries over from the case of positive programs, so we defer it to Proposition 6.15. Hardness is inherited from $\mathcal{DL}$.

$(\mathcal{K} \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,A)$ By Proposition 4.20, $\mathcal{K} \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,A$ iff $\mathcal{K} \models_{\mathsf{MKNF}} \mathsf{false}$ or $\mathcal{K} \not\models_{\mathsf{MKNF}} \mathbf{K}\,A$. By the first claim of this proposition, the first check is in coNP and the second check is in NP, so the algorithm is in coDP.

For hardness, we use the DP-complete SAT-UNSAT problem [Papadimitriou 1993]: given two propositional formulae φ and ψ, the problem is to decide whether φ is satisfiable and ψ is unsatisfiable. In our proof, we use the coDP-hard complement of this problem: decide whether φ is unsatisfiable or ψ is satisfiable. Without loss of generality, we can assume that φ and ψ do not share propositional variables. For an arbitrary such pair of formulae, let $\mathcal{K}_{\varphi,\psi} = (\mathcal{O}_{\varphi,\psi}, \mathcal{P}_{\varphi,\psi})$ be the MKNF knowledge base where $\mathcal{O}_{\varphi,\psi}$ contains $\Upsilon \cup \mu(\varphi) \cup \mu(\psi)$ (see Table VI for the definitions of Υ and μ) and $\mathcal{P}_{\varphi,\psi}$ contains the MKNF rules (71)–(72), and the facts (73)–(74).

(71) $$\mathsf{false} \leftarrow \mathbf{K}\, F(x), \mathbf{K}\, N_\varphi(x)$$

(72) $$\mathbf{K}\, q \leftarrow \mathbf{K}\, F(x), \mathbf{K}\, N_\psi(x)$$

(73) $$\mathbf{K}\, N_\varphi(u_\varphi) \leftarrow$$

(74) $$\mathbf{K}\, N_\psi(u_\psi) \leftarrow$$

The formulae φ and ψ do not share variables. Therefore, by Lemma 6.8 we have that $\mathbf{K}\, \mathcal{O}_{\varphi,\psi} \models_{\mathsf{MKNF}} \mathbf{K}\, F(u_\varphi)$ iff φ is unsatisfiable, and $\mathbf{K}\, \mathcal{O}_{\varphi,\psi} \models_{\mathsf{MKNF}} \mathbf{K}\, F(u_\psi)$ iff ψ is unsatisfiable; we denote this property as (*). Also, $\mathcal{K}_{\varphi,\psi}$ is nondisjunctive and positive so it has at most one MKNF model. We now show that φ is unsatisfiable or ψ is satisfiable iff $\mathcal{K}_{\varphi,\psi} \models_{\mathsf{MKNF}} \neg\, \mathbf{K}\, q$.

($\Rightarrow$) Assume that φ is unsatisfiable. Then, due to (*), (71), and (73), the knowledge base $\mathcal{K}_{\varphi,\psi}$ is MKNF unsatisfiable, so $\mathcal{K}_{\varphi,\psi} \models_{\mathsf{MKNF}} \neg\, \mathbf{K}\, q$. Furthermore, assume that φ is satisfiable and ψ is satisfiable. Then, due to (*), $\mathcal{K}_{\varphi,\psi}$ has a single MKNF model M such that $M \not\models \mathbf{K}\, F(u_\varphi)$ and $M \not\models \mathbf{K}\, F(u_\psi)$; furthermore, $M \not\models \mathbf{K}\, q$ as well, so $M \models \neg\, \mathbf{K}\, q$ and $\mathcal{K}_{\varphi,\psi} \models_{\mathsf{MKNF}} \neg\, \mathbf{K}\, q$.

($\Leftarrow$) We show the contrapositive. Assume that φ is satisfiable and ψ is unsatisfiable. Then, due to (*), $\mathcal{K}_{\varphi,\psi}$ has a single MKNF model M such that $M \not\models \mathbf{K}\, F(u_\varphi)$ and $M \models \mathbf{K}\, F(u_\psi)$. By (72) and (74), then we have $M \models \mathbf{K}\, q$, so $M \not\models \neg\, \mathbf{K}\, q$ and $\mathcal{K}_{\varphi,\psi} \not\models_{\mathsf{MKNF}} \neg\, \mathbf{K}\, q$. □

PROPOSITION 6.11 ($2 + \text{PTime}$). *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = \text{PTime}$ and $\mathcal{P}$ a stratified program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is PTime-complete in data complexity.*

PROOF. By Theorem 4.19, we can perform the check by computing $\psi[U^\infty_{\mathcal{K}_G}]$, where $\psi = (\neg)\, \mathbf{K}\, A$. Thus, the computation of $U^\infty_{\mathcal{K}_G}$ requires a polynomial number of computations of $T^\infty_{\mathcal{K}^i_G}$, each of which can be performed in PTime by Proposition 6.9. The computation of $\psi[U^\infty_{\mathcal{K}_G}]$ requires another call to an oracle in PTime. Hence, the algorithm can be implemented in PTime. Hardness carries over from positive datalog programs, cf. [Dantsin et al. 2001]. □

PROPOSITION 6.12 ($2 + co\text{NP}$). *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = co\text{NP}$ and $\mathcal{P}$ a stratified program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is Δ^p_2-complete in data complexity.*

PROOF. By Theorem 4.19, we can perform the check by computing $\psi[U^\infty_{\mathcal{K}_G}]$, where $\psi = (\neg)\, \mathbf{K}\, A$. Thus, the computation of $U^\infty_{\mathcal{K}_G}$ requires a polynomial number of calls to an oracle running in NP and the computation of $\psi[U^\infty_{\mathcal{K}_G}]$ requires another call to the oracle. Hence, the algorithm can be implemented in Δ^p_2.

For hardness, we present a reduction from the Δ^p_2-complete DAGS(SAT) problem [Gottlob 1995]. An instance of the problem is a triple $D = \langle L, G, \varphi_R \rangle$ where L is the set of propositional *linking variables*, $G = \langle V, E \rangle$ is a directed acyclic graph, and $\varphi_R \in V$ is a distinguished *result node.* The vertices in G are propositional formulae—that is, $V = \{\varphi_1, \ldots, \varphi_n\}$. Each φ_i is associated with a distinct linking variable x_i, so L and V have the same number of elements; furthermore, in addition to *private variables* not occurring in any other formula, a formula $\varphi_i \in V$ contains all linking variables x_j that correspond to some φ_j such that $(\varphi_j, \varphi_i) \in E$. A valuation $\nu_D : L \to \{\mathsf{true}, \mathsf{false}\}$ is defined inductively as follows: $\nu_D(x_i) = \mathsf{true}$ if and only if the propositional formula φ'_i, obtained by replacing in φ_i the linking variables with their values under ν_D, is satisfiable (since G is a direct acyclic graph, this induction is correctly defined). The problem is to decide whether $\nu_D(x_R) = \mathsf{true}$, where x_R is the linking variable corresponding to φ_R. Without loss of generality, we assume that $j < i$ for each $(\varphi_j, \varphi_i) \in E$—that is, the index of each node in G is larger than the indices of all of its predecessors.

Given an instance D of DAGS(SAT), let $\mathcal{K}_D = (\mathcal{O}_D, \mathcal{P}_D)$ be the MKNF knowledge base such that $\mathcal{O}_D = \Upsilon \cup \bigcup_{\varphi \in V} \mu(\varphi)$ (see Table VI for the definitions of Υ and μ), and $\mathcal{P}_D$ contains the following rules.

$$\mathbf{K}\, T(y) \leftarrow \mathbf{K}\, \mathit{formula}(x, y), \mathbf{not}\, F(x) \tag{75}$$

$$\mathbf{K}\, F(y) \leftarrow \mathbf{K}\, \mathit{formula}(x, y), \mathbf{K}\, F(x) \tag{76}$$

$$\mathbf{K}\, \mathit{formula}(u_{x_i}, u_{\varphi_i}) \leftarrow \quad \text{for each } \varphi_i \in V \text{ and } x_i \text{ the linking variable of } \varphi_i \tag{77}$$

The graph G is acyclic, and so are the assertions of the form (77). By Definition 4.17, all ground rules in $\mathsf{gr}(\mathcal{P}_D, O_{\mathcal{K}})$ where x is mapped to a constant other than u_{x_i} or y is mapped to a constant other than u_{φ_i} are not relevant. $\mathcal{K}_D$ is therefore a stratified MKNF knowledge base so, by considering each $1 \leq i \leq n$ in succession, we can determine the truth values of $\mathbf{K}\, T(u_{\varphi_i})$ and $\mathbf{K}\, F(u_{\varphi_i})$, and then determine the truth values of $\mathbf{K}\, T(u_{x_i})$ and $\mathbf{K}\, F(u_{x_i})$ through rules (75)–(77). We now prove by induction on i that $\nu_D(x_i) = \mathsf{true}$ iff $\mathcal{K}_D \models_{\mathsf{MKNF}} \mathbf{K}\, T(u_{x_i})$ for each linking variable x_i; for $x_i = x_R$, this implies that $\mathcal{K}_D$ exactly encodes DAGS(SAT).

For $i = 1$, the formula φ_1 contains no linking variables. Since the formulae in V do not share private variables, $\mathcal{K}_D \models_{\mathsf{MKNF}} \mathbf{K}\, F(u_{\varphi_1})$ iff $\nu_D(x_1) = \mathsf{false}$ by Lemma 6.8. But then, (75) ensures that $\mathcal{K}_D \models_{\mathsf{MKNF}} \mathbf{K}\, T(u_{x_1})$ iff $\nu_D(x_1) = \mathsf{true}$, and (76) ensures that $\mathcal{K}_D \models_{\mathsf{MKNF}} \mathbf{K}\, F(u_{x_1})$ iff $\nu_D(x_1) = \mathsf{false}$. Furthermore, for some $i \geq 1$, the rules (75)–(76) determine the values of the linking variables for $i + 1$, so the inductive step holds in exactly the same way. □

Proposition 6.13 (3+PTime). *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = \mathrm{PTime}$ and $\mathcal{P}$ a nondisjunctive program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is co*NP*-complete in data complexity.*

Proof. By Theorem 4.21, we can compute $\mathsf{nondisjunctive\text{-}not\text{-}entails}(\mathcal{K}_G, \psi)$ with $\psi = (\neg)\, \mathbf{K}\, A$ and complement the result. We guess a subset $P \subseteq \mathsf{KA}(\mathcal{K}_G)$. Since $\mathcal{K}'_G = (\mathcal{O}, \mathcal{P}_G[\mathbf{not}, P, N])$ is a nondisjunctive positive program, the computation of $T^{\infty}_{\mathcal{K}'_G}$ can be performed in polynomial time. Hence, all conditions can be verified in polynomial time, so the algorithm runs in NP. Hardness carries over from nondisjunctive datalog programs under stable model semantics, cf. [Dantsin et al. 2001]. □

Proposition 6.14 $(3 + co\mathrm{NP})$. *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = co\mathrm{NP}$ and $\mathcal{P}$ a nondisjunctive program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\, \mathbf{K}\, A$ is Π^p_2-complete in data complexity.*

Proof. Membership carries over from the case of general programs and is deferred to Proposition 6.16. To prove hardness, we use the Σ_2^p-complete 2-QBF problem [Stockmeyer 1976]: decide whether a second-order propositional formula $\varphi = \exists x_1, \ldots, x_n \forall y_1, \ldots, y_m : \psi$ is satisfiable, where ψ is a quantifier-free propositional formula over the variables $x_1, \ldots, x_n, y_1, \ldots, y_m$.

For an arbitrary formula φ of the mentioned form, let $\mathcal{K}_\varphi = (\mathcal{O}_\varphi, \mathcal{P}_\varphi)$ be the MKNF knowledge base where $\mathcal{O}_\varphi = \Upsilon \cup \mu(\psi)$ (see Table VI for the definitions of Υ and μ) and $\mathcal{P}_\varphi$ contains the following rules.

(78) $\mathbf{K}\, T(x) \leftarrow \mathbf{K}\, exvar(x), \mathbf{not}\, F(x)$

(79) $\mathbf{K}\, F(x) \leftarrow \mathbf{K}\, exvar(x), \mathbf{not}\, T(x)$

(80) $\mathbf{K}\, exvar(u_{x_i}) \leftarrow$ for each existentially quantified variable x_i in φ

The knowledge base $\mathcal{K}_\varphi$ is nondisjunctive. We now prove that φ is satisfiable if and only if $\mathcal{K}_\varphi \not\models_{\mathsf{MKNF}} \neg \mathbf{K}\, T(u_\psi)$; this implies Σ_2^p-hardness of nonentailment for nonstratified MKNF knowledge bases and, consequently, the claim of this proposition.

If φ is satisfiable, then a valuation ν for the variables x_i exists such that ψ is satisfied for each valuation of the variables y_i. Let M_ν be a set of first-order interpretations I satisfying the axioms from $\mathcal{O}_\varphi$ such that $I \models T(u_{x_i})$ if $\nu(x_i) = \mathsf{true}$ and $I \models F(u_{x_i})$ if $\nu(x_i) = \mathsf{false}$. Clearly, M_φ satisfies all the rules from $\mathcal{K}_\varphi$; furthermore, for each $M'_\varphi \supsetneq M_\varphi$, the head of either (78) or (79) is false. Hence, M_φ is an MKNF model of $\mathcal{K}_\varphi$. Since ψ is true for each value of the variables y_i, $M_\varphi \models \mathbf{K}\, T(u_\psi)$ by Lemma 6.8. Hence, $M_\varphi \not\models \neg \mathbf{K}\, T(u_\psi)$, so $\mathcal{K}_\varphi \not\models_{\mathsf{MKNF}} \neg \mathbf{K}\, T(u_\psi)$.

Conversely, if $\mathcal{K}_\varphi \not\models_{\mathsf{MKNF}} \neg \mathbf{K}\, T(u_\psi)$ holds, then an MKNF model M of $\mathcal{K}_\varphi$ exists such that $M \not\models \neg \mathbf{K}\, T(u_\psi)$—that is, $M \models \mathbf{K}\, T(u_\psi)$. Due to rules (78)–(79), either $M \models \mathbf{K}\, T(u_{x_i})$ or $M \models \mathbf{K}\, F(u_{x_i})$ for each variable x_i. Since M is the maximal set of first-order interpretations that satisfies $\mathcal{K}_\varphi$, for each valuation of the universally quantified variables y_i, a first-order interpretation $I \in M$ exists that corresponds to the valuation of y_i. Hence, ψ is true for each valuation of y_i by Lemma 6.8, so φ is satisfiable. □

Proposition 6.15 (4). *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ in co*NP *and $\mathcal{P}$ a positive program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\, A$ is co*NP*-complete, and deciding $\mathcal{K} \models_{\mathsf{MKNF}} \neg \mathbf{K}\, A$ is Π_2^p-complete in data complexity.*

Proof. ($\mathcal{K} \models_{\mathsf{MKNF}} \mathbf{K}\, A$) By Theorem 4.12, we compute $\mathsf{not\text{-}entails}^+(\mathcal{K}_G, \mathbf{K}\, A)$ and complement the result. A subset $P \subseteq \mathsf{KA}(\mathcal{K}_G)$ can be guessed in nondeterministic polynomial time. Condition (1) holds iff $\mathsf{OB}_{\mathcal{O},P_h} \not\models \mathsf{false}$; Condition (3) holds iff $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi$ for polynomially many ξ; and Condition (6) holds iff $\mathsf{OB}_{\mathcal{O},P_h} \not\models A$. All of these checks can be implemented using Proposition 6.4: we guess an equivalence $\sim$ on the constants, we separate the atoms from P_h into DL-$\mathbf{K}$-atoms P'_h and non-DL-$\mathbf{K}$-atoms P''_h, and we then check whether $\mathsf{OB}_{\mathcal{O}_\sim,P'_{h\sim}}$ is satisfiable, whether $\{\mathbf{K}\, A, \mathbf{K}\, \neg A\} \not\subseteq P''_{h\sim}$ for each ground first-order non-DL-atom A, and whether $\mathsf{OB}_{\mathcal{O}_\sim,P'_{h\sim}} \not\models \xi_\sim$ or $\mathbf{K}\, \xi_\sim \notin P''_{h\sim}$. Since $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}} = co$NP, this requires nondeterministic polynomial time. Condition (4) can clearly be verified in polynomial time, so the algorithm runs in NP. Hardness is inherited from $\mathcal{DL}$.

($\mathcal{K} \models_{\mathsf{MKNF}} \neg\,\mathbf{K}\,A$) Membership follows from the case of general programs and is proved in Proposition 6.16. Hardness follows from hardness of answering queries in positive datalog programs under stable model semantics [Eiter et al. 1997]. □

Proposition 6.16 (5). *For $\mathcal{C}_{\mathcal{DL},\mathcal{B},\mathcal{H}}$ in co*NP *and $\mathcal{P}$ a general program, deciding $\mathcal{K} \models_{\mathsf{MKNF}} (\neg)\,\mathbf{K}\,A$ is Π_2^p-complete in data complexity.*

Proof. By Theorem 4.10, we compute $\mathsf{not\text{-}entails}(\mathcal{K}_G, \psi)$ with $\psi = (\neg)\,\mathbf{K}\,A$ and complement the result. A subset $P \subseteq \mathsf{KA}(\mathcal{K}_G)$ can be guessed in nondeterministic polynomial time. Conditions (1) and (3) can be checked as discussed in Proposition 6.15, and require additional polynomial guessing. Condition (4) can be checked in polynomial time. The complement of Conditions (2) and (5) can be checked using an oracle. The oracle first checks $\mathsf{OB}_{\mathcal{O},P_h} \not\models \xi$ for each $\mathbf{K}\,\xi \in P \setminus P_h$ by using Proposition 6.4. If all checks are positive, the oracle guesses $P' \subsetneq P$, and then checks conditions (5a) and (5b). Finally, if the query is of the form $\mathbf{K}\,A$, then Condition (6) holds if $\mathsf{OB}_{\mathcal{O},P_h} \not\models A$, which can be decided in the same way as Condition (3); in contrast, if the query is of the form $\neg\,\mathbf{K}\,A$, then Condition (6) holds if $\mathsf{OB}_{\mathcal{O},P_h} \models A$, which can be decided in the same way as Condition (2). Hence, the oracle runs in NP. Hence, our algorithm runs in Π_2^p. Hardness is inherited from disjunctive datalog under stable model semantics [Eiter et al. 1997]. □

7. COMPARISON WITH RELATED FORMALISMS

In this section, we show that MKNF knowledge bases provide a very general framework that can capture many existing combinations of DLs and rules.

7.1 Extensions of DLs with First-Order Rules

Investigation of first-order combinations of a DL knowledge base $\mathcal{O}$ with a set of first-order rules $\mathcal{P}$ has a long history. To the best of our knowledge, $\mathcal{AL}$-log [Donini et al. 1998] was the first such hybrid system, and systems such as CARIN [Levy and Rousset 1998] and SWRL [Horrocks et al. 2005] followed. The resulting formalisms are quite expressive, which comes at a price: if $\mathcal{O}$ is allowed to contain existential quantifiers and $\mathcal{P}$ is allowed to contain recursive rules, the formalism is undecidable. Decidability can be ensured by either making the rules in $\mathcal{P}$ nonrecursive, or by restricting the rules in $\mathcal{P}$ to be *role-safe*: at least one of the arguments of each role atom (i.e., an atom with a binary DL-predicate) in a rule must also occur in a non-DL-atom in the body of the rule. This restriction makes the rules applicable only to the explicitly named individuals and their immediate successors. Motik et al. [2005] proposed a somewhat simpler concept of *DL-safety*, which generalizes a similar notion used in $\mathcal{AL}$-log: each variable in a rule must occur in a body non-DL-atom. Similarly as in Section 6.1, this makes the rules applicable only to the individuals that are known in $\mathcal{O}$ by name. Recently, Krötzsch et al. [2008b] have proposed an approach in which the rules are syntactically restricted such that they can be transformed to equivalent DL axioms.

All first-order combinations of DLs and rules can be captured straightforwardly using MKNF$^+$ knowledge bases. Furthermore, as we discussed in Section 3.3, each MKNF$^+$ knowledge base can be reduced to an MKNF knowledge base if we extend the generalized atom base. In fact, from a conceptual point of view, first-order rules are best thought as part of the first-order fragment $\mathcal{DL}$; the main idea of MKNF knowledge bases is to enable nonmonotonic reasoning over first-order logic.

7.2 $\mathcal{DL}$+log

Rosati [2005] proposed *r-hybrid knowledge bases* as one of the first formalisms that integrates open-world DLs with nonmonotonic rules, and de Bruijn et al. [2007] have shown that r-hybrid knowledge bases can be embedded into quantified equilibrium logic (QEL) [Pearce and Valverde 2005]. Rosati [2006] has later generalized r-hybrid KBs to $\mathcal{DL}$+log.

A $\mathcal{DL}$+log knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ consists of a DL knowledge base $\mathcal{O}$ and disjunctive rules $\mathcal{P}$ of the form (6) in which all literals are positive. The signature Σ is separated into DL- and non-DL-predicates, and DL-atoms cannot occur under *not* in the rules. The nonmonotonic semantics of $\mathcal{DL}$+log employs the standard name assumption, and is defined w.r.t. some fixed countably infinite universe $\triangle$. Let $\mathcal{P}_G = \mathsf{gr}(\mathcal{P}, \triangle)$. Furthermore, for a first-order interpretation I over $\triangle$, let I_{DL} and $I_{\overline{DL}}$ be the projection of I to the DL- and the non-DL-predicates, respectively. Finally, let $\Pi(\mathcal{P}_G, I)$ be the program obtained by replacing in $\mathcal{P}_G$ each ground DL-atom A with true if $I \models A$ and with false if $I \not\models A$. Then, I is a model of $\mathcal{K}$ if (i) $I_{DL} \models \mathcal{O}$ in the standard first-order sense, and (ii) $I_{\overline{DL}}$ is an answer set of $\Pi(\mathcal{P}_G, I)$. To achieve decidability, $\mathcal{DL}$+log employs a notion of *weak DL-safety*: each variable in a head DL-atom must occur in a body non-DL-atom. Under this restriction, variables in the bodies of a rule can be matched to unnamed individuals; however, the rules can derive facts only about the explicitly named individuals.

We next show that each $\mathcal{DL}$+log knowledge base can be encoded as an equivalent MKNF$^+$ knowledge base.

Definition 7.1. For a $\mathcal{DL}$+log knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$, the MKNF$^+$ encoding of $\mathcal{K}$ is the knowledge base $\mu(\mathcal{K}) = (\mathcal{O}, \mu(\mathcal{P}))$ where each $r \in \mathcal{P}$ is transformed into a rule $\mu(r) \in \mu(\mathcal{P})$ by transforming the literals according to the following table.

not A	$\rightsquigarrow$	**not** A	
A	$\rightsquigarrow$	A	if A is a DL-atom
A	$\rightsquigarrow$	$\mathbf{K}\, A$	if A is a non-DL-atom

THEOREM 7.2. *A $\mathcal{DL}$+log knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ is satisfiable if and only if its MKNF$^+$ encoding $\mu(\mathcal{K}) = (\mathcal{O}, \mu(\mathcal{P}))$ is MKNF satisfiable.*

PROOF. Let $\mathcal{P}_G = \mathsf{gr}(\mathcal{P}, \triangle)$ and $\mu(\mathcal{P}_G) = \mathsf{gr}(\mu(\mathcal{P}), \triangle)$. By the definition of the semantics of $\mathcal{DL}$+log and MKNF$^+$ knowledge bases, it suffices to show that the knowledge base $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ is satisfiable if and only if $\mu(\mathcal{K}) = (\mathcal{O}, \mu(\mathcal{P}_G))$ is MKNF satisfiable. For a ground rule $r \in \mathcal{P}_G$, let r_{DL} be the rule obtained from r by deleting all non-DL-atoms.

($\Rightarrow$) Assume that $\mathcal{K}$ is satisfiable in some model I. Let M be the maximal MKNF interpretation such that (i) for each $r \in \mathcal{P}_G$, if $I \models r_{DL}$, then $J \models r_{DL}$ for each $J \in M$, (ii) for each $r \in \mathcal{P}_G$, if $I \not\models r_{DL}$, then $J \not\models r_{DL}$ for at least one $J \in M$, (iii) for each non-DL-atom A, if $I \models A$, then $J \models A$ for each $J \in M$, (iv) for each non-DL-atom A, if $I \not\models A$, then $J \not\models A$ for at least one $J \in M$, and (v) $J \models \mathcal{O}$ for each $J \in M$. It should be clear that M is uniquely defined; furthermore, M is not empty since it contains I. For each ground non-DL-atom A, we have $I \models A$ if and only if $M \models \mathbf{K}\, A$ if and only if $M \not\models \mathbf{not}\, A$. Similarly, for each $r \in \mathcal{P}_G$, we have $M \models \mu(r_{DL})$ if and only if $I \models r_{DL}$. Clearly, $M \models \mu(r)$ for each $r \in \mathcal{P}_G$. By the definition of M, we have $M \models \mathcal{O}$, so $M \models \mu(\mathcal{K}_G)$. To show that M is an MKNF model of $\mu(KB)$, assume that an MKNF model $M' \supsetneq M$ exists such that $(J', M', M) \models \mu(\mathcal{K}_G)$ for some $J' \in M'$. Let J be an interpretation that coincides with J' on the DL-atoms but, for each non-DL-atom A, we have $J \models A$ if and only if $M' \models \mathbf{K}\, A$. Since $M' \supsetneq M$, we have $J \subsetneq I$. Clearly, $J \models \mathcal{O}$ and $J \models \Pi(\mathcal{P}_G, I)^I$, which contradicts the assumption that I is an answer set of $\Pi(\mathcal{P}_G, I)^I$.

($\Leftarrow$) Assume that $\mu(\mathcal{K}_G)$ has an MKNF model M. Let I' be any interpretation from M (note that M is not empty), and let I be an interpretation that coincides with I' on the DL-atoms but, for each non-DL-atom A, we have $I \models A$ if and only if $M \models \mathbf{K}\, A$. Since $(I, M, M) \models \mu(r)$, we have $I \models r$ for each $r \in \mathcal{P}_G$ and, by the definition of M, we have $I \models \mathcal{O}$. Assume now that some $J \subsetneq I$ exists such that $J \models \Pi(\mathcal{P}_G, I)^I$. Exactly as it was done for M and I in the ($\Rightarrow$) direction, we can construct an MKNF interpretation M' from J and show that $(J, M', M) \models \mu(\mathcal{K}_G)$. Since $J \subsetneq I$, we have $M' \supsetneq M$, which contradicts the assumption that M is an MKNF model of $\mu(\mathcal{K}_G)$. □

In $\mathcal{DL}$+log, the DL-predicates are interpreted always under open-world assumption, whereas non-DL-predicates are interpreted always under closed-world semantics. Thus, $\mathcal{DL}$+log knowledge bases are inflexible (in the sense from the introduction), and they cannot be used to provide semantics for constructs such as integrity constraints or default reasoning, which require nonmonotonic reasoning over DL-predicates. In MKNF$^+$ knowledge bases, however, one can freely choose an open- or closed-world interpretation of a predicate by using the predicate in either a non-modal or a modal atom. Thus, unlike $\mathcal{DL}$+log, MKNF$^+$ knowledge bases allow for nonmonotonic closed-world reasoning over DL-predicates.

Furthermore, DL-safety and the reasoning procedure from Table IV generalize weak DL-safety and the reasoning algorithm by Rosati [2006]. The MKNF$^+$ knowledge base $\mu(\mathcal{K})$ can be transformed into an MKNF knowledge base $\mathcal{K}'$ as shown in Proposition 3.6; the resulting knowledge base then contains the atoms $\mathbf{K}\, \xi$ in the rule heads where ξ is of the form (81) and it contains only DL-atoms.

$$A_1 \vee \ldots \vee A_n \subset \exists \overline{y} : B_1 \wedge \ldots \wedge B_m \tag{81}$$

If $\mathcal{K}$ is weakly DL-safe, $\mathcal{K}'$ is DL-safe, and grounding $\mathcal{K}'$ replaces all free variables in A_i by constants. The first-order problems in the algorithm shown in Table IV then involve implications of the form (81) where A_i are ground, which can be solved by answering unions of conjunctive queries over $\mathcal{O}$.

7.3 Disjunctive DL-Programs by Lukasiewicz

Lukasiewicz [2007] proposed *disjunctive dl-programs* as a simple hybrid formalism that integrates DLs and nonmonotonic rules. A disjunctive dl-program $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ consists of a DL knowledge base $\mathcal{O}$ and a set of disjunctive rules $\mathcal{P}$ of the form (6) in which no literal contains classical negation—that is, in which all H_i with $1 \leq i \leq k$ and all B_j with $1 \leq j \leq n$ are function-free first-order atoms. The semantics of $\mathcal{K}$ is defined by grounding $\mathcal{P}$ w.r.t. a finite set $\triangle$ of constants that contains at least all constants occurring in $\mathcal{O}$ and $\mathcal{P}$; let $\mathcal{K}_G = (\mathcal{O}, \mathcal{P}_G)$ where $\mathcal{P}_G = \mathsf{gr}(\mathcal{P}, \triangle)$. An interpretation I is a subset of the Herbrand base $HB_{\mathcal{P}_G}$, and I is a *model* of $\mathcal{O}$ if $\Pi(\mathcal{O}, I) = \mathcal{O} \cup I \cup \{\neg A \mid A \notin HB_{\mathcal{P}_G} \setminus I\}$ is satisfiable. Furthermore, I is a *model* of $\mathcal{K}_G$ if I is a model of $\mathcal{O}$ and $I \models \mathcal{P}_G$. Finally, I is an *answer set* of $\mathcal{K}_G$ if I is a model of $\mathcal{K}_G$ and no interpretation $I' \subsetneq I$ exists that is a model of $(\mathcal{O}, \mathcal{P}_G^I)$. Without loss of generality, one can assume that $\triangle = O_{\mathcal{K}}$; that is, the semantics can equivalently be considered w.r.t. the set of constants explicitly occurring in $\mathcal{K}$.

The semantics of disjunctive dl-programs is faithful w.r.t. the standard stable model semantics of disjunctive datalog programs with negation as failure [Gelfond and Lifschitz 1988]: for each ground atom A, a disjunctive dl-program $(\emptyset, \mathcal{P})$ entails A if and only if $\mathcal{P}$ entails A under the stable model semantics. The entailment relation is, however, undefined for classically negated ground atoms. The obvious extension would be to say that $(\mathcal{O}, \mathcal{P}) \models \neg A$ if and only if $A \notin I$ for each answer set I of $(\mathcal{O}, \mathcal{P})$. But then $(\emptyset, \emptyset) \models \neg A$ for each ground atom A, so entailment is not faithful w.r.t. the standard first-order semantics of DLs. For similar reasons, it is unclear how to extend disjunctive dl-programs to ASP rules. Thus, disjunctive dl-programs use a much weaker notion of faithfulness than MKNF knowledge bases, and they seem to be capable of reasoning only about positive atoms.

We next show that each disjunctive dl-program can be encoded as an equivalent MKNF knowledge base.

Definition 7.3. Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be a disjunctive dl-program and Λ the set of predicates occurring in $\mathcal{K}$. Furthermore, let O be a fresh unary predicate and w a fresh proposition, and, for each n-ary predicate $R \in \Lambda$, let R^{1+}, R^{1-}, R^{2+}, and R^{2-} be fresh n-ary predicates uniquely associated with R. Finally, let $a_1, \ldots, a_\ell$ be all constants occurring in $\mathcal{K}$. The *MKNF encoding* of $\mathcal{K}$ is an MKNF knowledge base $\mathcal{K}' = (\emptyset, \mathcal{P}')$ where the program $\mathcal{P}'$ contains (i) a rule of the form (82) for each a_i, $1 \leq i \leq \ell$, (ii) rule (83), (iii) rules of the forms (84)–(90) for each n-ary predicate $R \in \Lambda$, (iv) rules (91)–(94), and (v) two rules of the form (95)–(96) for each rule $r \in \mathcal{P}$ of the form (6), where H^{i+} is the atom obtained from H by replacing the predicate R of H by R^{i+} and similarly for B_j^{i+} and B_j^{i-}.

$$\mathbf{K}\, O(a_i) \leftarrow \tag{82}$$

$$\mathbf{K}\, \psi_O \leftarrow \tag{83}$$

$$\mathbf{K}\, R^{1+}(\overline{x}) \vee \mathbf{K}\, R^{1-}(\overline{x}) \leftarrow \mathbf{K}\, O(x_1), \ldots, \mathbf{K}\, O(x_n) \tag{84}$$

$$\mathsf{false} \leftarrow \mathbf{K}\, R^{1+}(\overline{x}), \mathbf{K}\, R^{1-}(\overline{x}) \tag{85}$$

$$\mathbf{K}\, R^{2+}(\overline{x}) \vee \mathbf{K}\, R^{2-}(\overline{x}) \leftarrow \mathbf{K}\, O(x_1), \ldots, \mathbf{K}\, O(x_n) \tag{86}$$

$$\mathbf{K}\, w \leftarrow \mathbf{K}\, R^{2+}(\overline{x}), \mathbf{K}\, R^{2-}(\overline{x}) \tag{87}$$

$$\mathbf{K}\, R^{2+}(\overline{x}) \leftarrow \mathbf{K}\, w, \mathbf{K}\, O(x_1), \ldots, \mathbf{K}\, O(x_n) \tag{88}$$

$$\mathbf{K}\, R^{2-}(\overline{x}) \leftarrow \mathbf{K}\, w, \mathbf{K}\, O(x_1), \ldots, \mathbf{K}\, O(x_n) \tag{89}$$

$$\mathbf{K}\, R^{2-}(\overline{x}) \leftarrow \mathbf{K}\, R^{1-}(\overline{x}) \tag{90}$$

$$\mathbf{K}\, w \leftarrow \mathbf{K}\, \psi_{1 \supset 2} \tag{91}$$

$$\mathsf{false} \leftarrow \mathbf{K}\, \Xi(\mathcal{O}, 1) \tag{92}$$

$$\mathbf{K}\, w \leftarrow \mathbf{K}\, \Xi(\mathcal{O}, 2) \tag{93}$$

$$\mathbf{K}\, w \leftarrow \mathbf{not}\, w \tag{94}$$

$$\mathbf{K}\, H_1^{1+} \vee \ldots \vee \mathbf{K}\, H_k^{1+} \leftarrow \mathbf{K}\, B_1^{1+}, \ldots, \mathbf{K}\, B_m^{1+}, \mathbf{K}\, B_{m+1}^{1-}, \ldots, \mathbf{K}\, B_n^{1-} \tag{95}$$

$$\mathbf{K}\, w \vee \mathbf{K}\, H_1^{2+} \vee \ldots \vee \mathbf{K}\, H_k^{2+} \leftarrow \mathbf{K}\, B_1^{2+}, \ldots, \mathbf{K}\, B_m^{2+}, \mathbf{K}\, B_{m+1}^{1-}, \ldots, \mathbf{K}\, B_n^{1-} \tag{96}$$

The first-order formulae ψ_O, $\psi_{1\supset 2}$, $\Xi(\mathcal{O},1)$, and $\Xi(\mathcal{O},2)$ are defined as follows.

$$\psi_O = \forall x : [O(x) \supset x \approx a_1 \vee \ldots \vee x \approx a_\ell] \tag{97}$$

$$\psi_{1\supset 2} = \bigwedge_{R \in \Lambda} \forall \overline{x} : \left[R^{1+}(\overline{x}) \wedge \bigwedge_{x \in \overline{x}} O(x) \supset R^{2+}(\overline{x}) \right] \tag{98}$$

$$\Xi(\mathcal{O},i) = \neg \left[\pi(\mathcal{O}) \wedge \bigwedge_{R \in \Lambda} \forall \overline{x} : R^{i+}(\overline{x}) \supset R(\overline{x}) \wedge \bigwedge_{R \in \Lambda} \forall \overline{x} : R^{i-}(\overline{x}) \supset \neg R(\overline{x}) \right] \tag{99}$$

Theorem 7.4. *A disjunctive dl-program $\mathcal{K} = (\mathcal{O},\mathcal{P})$ has an answer set if and only if its MKNF encoding $\mathcal{K}' = (\mathcal{O},\mathcal{P}')$ is MKNF satisfiable.*

Proof. Predicates R^{1+}, R^{1-}, R^{2+}, and R^{2-} can be considered as non-DL-predicates. Furthermore, while predicate O does not strictly satisfy the definition of non-DL-predicates due to (83), this fact merely "closes" the extension of O to all named constants; hence, O behaves as a non-DL-predicate. Thus, $\mathcal{K}'$ has the same semantic properties as if it were DL-safe. Let $\mathcal{K}'_G$ is the grounding of $\mathcal{K}'$ w.r.t. $O_{\mathcal{K}'}$; then $\mathcal{K}'$ and $\mathcal{K}'_G$ have the same MKNF models. To show the claim, it suffices to show that the grounding $\mathcal{K}_G$ of $\mathcal{K}$ has an answer set iff $\mathcal{K}'_G$ is MKNF satisfiable. By Theorem 4.10, $\mathcal{K}'_G$ is MKNF satisfiable iff $\textsf{not-entails}(\mathcal{K}'_G, \mathbf{K}\,\textsf{false})$ returns $\textsf{true}$.

($\Leftarrow$) Let (P,N) be a partition of $\textsf{KA}(\mathcal{K}'_G)$ and $P_h = P \cap \textsf{HA}(\mathcal{K}'_G)$ such that Conditions (1)–(6) in Table IV are satisfied. Since (82)–(96) are satisfied in (P,N), we have $\mathbf{K}\, w \in P$, $\mathbf{K}\, O(a_i) \in P$ for each $1 \leq i \leq \ell$, $\mathbf{K}\, \psi_O \in P$, either $\mathbf{K}\, R^{1+}(\overline{a}) \in P$ or $\mathbf{K}\, R^{1-}(\overline{a}) \in P$ but not both for each $R \in \Lambda$ and each tuple of constants $\overline{a}$, and both $\mathbf{K}\, R^{2+}(\overline{a}) \in P$ and $\mathbf{K}\, R^{2-}(\overline{a}) \in P$ for each $R \in \Lambda$ and each tuple of constants $\overline{a}$. The formula $\textsf{OB}_{\emptyset,P_h}$ thus contains both $R^{2+}(\overline{a})$ and $R^{2-}(\overline{a})$ for each $R \in \Lambda$, so $\textsf{OB}_{\emptyset,P_h} \models \psi_{1\supset 2}$ and $\textsf{OB}_{\emptyset,P_h} \wedge \neg\Xi(\mathcal{O},2)$ is unsatisfiable; hence, $\mathbf{K}\, \psi_{1\supset 2} \in P$ and $\mathbf{K}\, \Xi(\mathcal{O},2) \in P$. Finally, by (92), $\mathbf{K}\, \Xi(\mathcal{O},1) \in N$. We now show that the interpretation $I = \{R(\overline{a}) \mid \mathbf{K}\, R^{1+}(\overline{a}) \in P_h\}$ is an answer set of $\mathcal{K}_G$. Since (P,N) satisfies each rule of the form (95), we have $I \models \mathcal{P}_G$. Furthermore, $\mathbf{K}\, \Xi(\mathcal{O},1) \in N$ implies $\textsf{OB}_{\emptyset,P_h} \wedge \neg\Xi(\mathcal{O},1)$ is satisfiable, so $\Pi(\mathcal{O},I)$ is satisfiable as well. Thus, I is a model of $\mathcal{K}_G$.

Assume now that I is not an answer set of $\mathcal{K}_G$ because a model $I' \subsetneq I$ of $(\mathcal{O},\mathcal{P}_G^I)$ exists. Let P' be a subset of $\textsf{KA}(\mathcal{K}'_G)$ such that $\mathbf{K}\, R^{1+}(\overline{a}) \in P'$ iff $\mathbf{K}\, R^{1+}(\overline{a}) \in P$ and $\mathbf{K}\, R^{1-}(\overline{a}) \in P'$ iff $\mathbf{K}\, R^{1-}(\overline{a}) \in P$, $\mathbf{K}\, R^{2+}(\overline{a}) \in P'$ iff $R(\overline{a}) \in I'$, $\mathbf{K}\, R^{2-}(\overline{a}) \in P'$ iff $R(\overline{a}) \notin I'$, $\mathbf{K}\, O(a_i) \in P$ for each $1 \leq i \leq \ell$, and $\mathbf{K}\, \psi_O \in P$; furthermore, let $P'_h = P' \cap \textsf{HA}(\mathcal{K}'_G)$ and $N' = \textsf{KA}(\mathcal{K}'_G) \setminus P'$. Clearly, (P,N) satisfies rules (82)–(95); furthermore, $I' \models \mathcal{P}_G^I$, so (P,N) satisfies (96); thus, Condition (5b) in Table IV is not satisfied. Furthermore, $I' \subsetneq I$ implies $\textsf{OB}_{\emptyset,P'_h} \not\models \psi_{1\supset 2}$; since $\Pi(\mathcal{O},I')$ is satisfiable, $\textsf{OB}_{\emptyset,P'_h} \wedge \neg\Xi(\mathcal{O},2)$ is satisfiable as well; finally, $\textsf{OB}_{\emptyset,P'_h} \not\models R^{2+}(\overline{a})$ for all $\mathbf{K}\, R^{2+}(\overline{a}) \in N' \setminus N$ and $\textsf{OB}_{\emptyset,P'_h} \not\models R^{2-}(\overline{a})$ for all $\mathbf{K}\, R^{2-}(\overline{a}) \in N' \setminus N$. Thus, Condition (5a) in Table IV is not satisfied, which contradicts our assumption that (P,N) satisfies Conditions (1)–(6) in Table IV.

($\Rightarrow$) Let I be an answer set of $\mathcal{K}_G$. Furthermore, let P be a subset of $\mathsf{KA}(\mathcal{K}'_G)$ such that $\mathbf{K}\, R^{1+}(\overline{a}) \in P$ iff $R(\overline{a}) \in I$, $\mathbf{K}\, R^{1-}(\overline{a}) \in P$ iff $R(\overline{a}) \notin I$, $\mathbf{K}\, R^{2+}(\overline{a}) \in P$ and $\mathbf{K}\, R^{2-}(\overline{a}) \in P$ for each $R \in \Lambda$ and each tuple of constants $\overline{a}$, $\mathbf{K}\, O(a_i) \in P$ for each $1 \leq i \leq \ell$, $\mathbf{K}\, \psi_O \in P$, $\mathbf{K}\, w \in P$, and $\mathbf{K}\, \Xi(\mathcal{O}, 2) \in P$; finally, let $P_h = P \cap \mathsf{HA}(\mathcal{K}'_G)$ and $N = \mathsf{KA}(\mathcal{K}'_G) \setminus P$. We now show that (P, N) satisfies Conditions (1)–(6) in Table IV. The set P_h contains only $\mathbf{K}$-atoms of the form $\mathbf{K}\, w$, $\mathbf{K}\, O(a_i)$, $\mathbf{K}\, \psi_O$, and $\mathbf{K}\, S(\overline{a})$ where S is of the form R^{1+}, R^{1-}, R^{2+}, and R^{2+}, so Condition 1 in Table IV is clearly satisfied. The formula $\mathsf{OB}_{\emptyset,P_h}$ contains both $R^{2+}(\overline{a})$ and $R^{2-}(\overline{a})$ for each $R \in \Lambda$, so $\mathsf{OB}_{\emptyset,P_h} \models \psi_{1\supset 2}$ and $\mathsf{OB}_{\emptyset,P_h} \wedge \neg\Xi(\mathcal{O}, 2)$ is unsatisfiable; thus, Condition 2 in Table IV is satisfied. For $\mathbf{K}\, \xi \in N$ of the form $R^{1+}(\overline{a})$ or $R^{1-}(\overline{a})$, clearly $\mathsf{OB}_{\emptyset,P_h} \not\models \xi$; furthermore, since $\Pi(\mathcal{O}, I)$ is satisfiable, $\mathsf{OB}_{\emptyset,P_h} \wedge \neg\Xi(\mathcal{O}, 1)$ is satisfiable as well; thus, Condition 3 in Table IV is satisfied. Rules (82)–(94) are clearly satisfied in (P, N). Furthermore, since $I \models \mathcal{P}_G$, all rules of the form (95) are satisfied in (P, N). Finally, since $\mathbf{K}\, w \in P$, all rules of the form (96) are satisfied in (P, N) as well. Thus, Condition (4) in Table IV is satisfied.

Assume that Condition (5) in Table IV is not satisfied for some (P', N'). By (84)–(85), $\mathbf{K}\, R^{1+}(\overline{a}) \in P'$ iff $\mathbf{K}\, R^{1+}(\overline{a}) \in P$ and $\mathbf{K}\, R^{1-}(\overline{a}) \in P'$ iff $\mathbf{K}\, R^{1-}(\overline{a}) \in P$. Furthermore, if $\mathbf{K}\, w \in P'$, by (88)–(89) then $\mathbf{K}\, R^{2+}(\overline{a}) \in P'$ and $\mathbf{K}\, R^{2-}(\overline{a}) \in P'$ for each $R \in \Lambda$ and each tuple of constants $\overline{a}$; but then $\mathbf{K}\, \Xi(\mathcal{O}, 2) \in P'$, so $P' = P$, which is a contradiction, so $\mathbf{K}\, w \notin P'$. Let $I' = \{R(\overline{a}) \mid \mathbf{K}\, R^{2+}(\overline{a}) \in P'_h\}$. Condition (5b) in Table IV is not satisfied for (P', N'), so all rules of the form (96) are true in (P', N'); since $\mathbf{K}\, w \notin P'$, we have $I' \models \mathcal{P}_G^I$. Furthermore, $\mathbf{K}\, w \notin P'$ and rule (93) is satisfied in (P', N'), so $\mathbf{K}\, \Xi(\mathcal{O}, 2) \in N'$; since Condition (5b) in Table IV is not satisfied, $\mathsf{OB}_{\emptyset,P'_h} \wedge \neg\Xi(\mathcal{O}, 2)$ is satisfiable, and so is $\Pi(\mathcal{O}, I')$. Thus, I' is a model of $(\mathcal{O}, \mathcal{P}_G^I)$. Finally, $\mathbf{K}\, w \notin P'$ and rule (91) is satisfied in (P', N'), so $\mathbf{K}\, \psi_{1\supset 2} \in N'$; since Condition (5b) in Table IV is not satisfied, $\mathsf{OB}_{\emptyset,P'_h} \not\models \psi_{1\supset 2}$, so $I' \subsetneq I$. But then, I is not an answer set of $\mathcal{K}_G$, which is a contradiction. $\square$

7.4 DL-Programs by Eiter et al.

All rule formalisms discussed thus far are examples of a tight integration between DLs and rules. Eiter et al. [2008] proposed a completely different integration strategy known as *dl-programs*. Despite the similarity in the name, this formalism is quite different from the one described in Section 7.3.

The atoms in a dl-program are divided into DL- and non-DL-atoms. A dl-program $\mathcal{K}$ consists of a DL knowledge base $\mathcal{O}$ and a set of nondisjunctive rules $\mathcal{P}$ of the form (6) (note that $k = 1$). Furthermore, each literal in the head must be a non-DL-atom, and each literal in the body can be either a non-DL-atom or a *dl-query* of the form

$$DL[S_1 \; op_1 \; p_1, \ldots, S_\ell \; op_\ell \; p_\ell; Q](\overline{t}), \tag{100}$$

where S_i are DL-predicates, p_i are non-DL-predicates, $op_i \in \{\uplus, \cup\!\!\!-, \cap\!\!\!-\}$, Q is an n-ary DL-predicate, and $\overline{t}$ is a vector of n terms. The semantics of $\mathcal{K}$ is defined for the grounding $\mathcal{K}_G$ w.r.t. the set of all constants $O_\mathcal{K}$. The Herbrand base $HB_{\mathcal{P}_G}$ is defined as the set of all ground literals that can be built using the non-DL-predicates and the constants from $\mathcal{K}$.[11] An *interpretation* I is a consistent subset of $HB_{\mathcal{P}_G}$. A

[11]Thus, $HB_{\mathcal{P}_G}$ does not contain dl-query atoms.

literal A containing a non-DL-predicate is true in I if and only if $A \in I$. A ground DL-query of the form (100)—that is, a query where $\bar{t}$ is a vector of constants—is satisfied in I if and only if $\mathcal{O} \cup \bigcup_{1 \leq i \leq \ell} A_i(I) \models Q(\bar{t})$, where

$$A_i(I) = \begin{cases} \{S_i(\bar{e}) \mid p_i(\bar{e}) \in I\} & \text{if } op_i = \uplus \\ \{\neg S_i(\bar{e}) \mid p_i(\bar{e}) \in I\} & \text{if } op_i = \cupdot \\ \{\neg S_i(\bar{e}) \mid p_i(\bar{e}) \notin I\} & \text{if } op_i = \cap\!\!\!\!- \end{cases}.$$

Eiter et al. [2008] defined strong and weak answer sets of dl-programs. For the sake of simplicity, we consider here only the strong answer set semantics of dl-programs and assume that no dl-query in the program contains $\cap\!\!\!\!-$. The *strong dl-transform* of a ground set of dl-rules $\mathcal{P}_G$ w.r.t. an interpretation I is the set of dl-rules $s\mathcal{P}_G^I$ obtained from $\mathcal{P}_G$ by deleting every rule containing an atom *not* A such that A is true in I, and deleting all atoms of the form *not* B in the remaining rules. An interpretation I is a *strong answer set* of $\mathcal{K}_G$ if $I \models s\mathcal{P}_G^I$ and, for each $I' \subsetneq I$, we have $I' \not\models s\mathcal{P}_G^I$.

In this style of integration of DLs and ASP, the rules in dl-programs cannot derive new facts about DL-predicates; they can only pose conditional queries to the DL knowledge base and then use the results in further computation. Thus, this style of integration of DLs and rules is not tight.

We now show that it is possible to encode dl-programs not containing $\cap\!\!\!\!-$ in dl-queries into MKNF knowledge bases.

Definition 7.5. Let $\mathcal{K}$ be a dl-program with a DL knowledge base $\mathcal{O}$ and a set of rules $\mathcal{P}$ such that no dl-query in $\mathcal{P}$ contains $\cap\!\!\!\!-$. For a literal A with a non-DL-predicate, let $\tau(A) = A$; furthermore, for a dl-query A of the form (100), let

$$\tau(A) = \left[\pi(\mathcal{O}) \wedge \bigwedge_{1 \leq i \leq \ell} \tau(S_i \ op_i \ p_i)\right] \supset Q(\bar{t}) \qquad \text{where}$$

$$\tau(S_i \ op_i \ p_i) = \begin{cases} \forall \bar{x} : p_i(\bar{x}) \supset S_i(\bar{x}) & \text{if } op_i = \uplus \\ \forall \bar{x} : p_i(\bar{x}) \supset \neg S_i(\bar{x}) & \text{if } op_i = \cupdot \end{cases}$$

The MKNF encoding of $\mathcal{K}$ is the MKNF knowledge base $\mathcal{K}' = (\emptyset, \mathcal{P}')$, where the set of rules $\mathcal{P}'$ is obtained from $\mathcal{P}$ be transforming each literal A into $\mathbf{K}\,\tau(A)$ and each literal *not* A into $\mathbf{not}\,\tau(A)$.

Theorem 7.6. *For $\mathcal{K}$ and $\mathcal{K}'$ as specified in Definition 7.5, $\mathcal{K}$ has a strong answer set if and only if $\mathcal{K}'$ has an MKNF model.*

Proof. Since $\mathcal{K}'$ is DL-safe, it suffices to prove that the grounding $\mathcal{K}_G$ of $\mathcal{K}$ has a strong answer set if and only if $\mathcal{K}'_G$ is MKNF satisfiable, where $\mathcal{K}'_G$ is the grounding of $\mathcal{K}'$ w.r.t. $\mathcal{O}_{\mathcal{K}'}$. By Theorem 4.10, $\mathcal{K}'_G$ is MKNF satisfiable if and only if $\mathsf{not\text{-}entails}(\mathcal{K}_G, \mathbf{K}\,\mathsf{false})$ returns true.

($\Leftarrow$) Let (P, N) be a partition of $\mathsf{KA}(\mathcal{K}'_G)$ and let $P_h = P \cap \mathsf{HA}(\mathcal{K}'_G)$ such that Conditions (1)–(6) in Table IV are satisfied. We show that $I = \{A \mid \mathbf{K}\,A \in P_h\}$ is a strong answer set of $\mathcal{K}_G$. Let A be some literal with a non-DL-predicate. By Conditions (2) and (3), clearly $\mathbf{K}\,A \in P$ if and only if $A \in I$. Furthermore, let A

be a dl-query of the form (100). Because

$$P_h \models [\pi(\mathcal{O}) \wedge \bigwedge_{1 \leq i \leq \ell} \tau(S_i \ op_i \ p_i)] \supset Q(\bar{t}) \quad \text{if and only if}$$
$$P_h \wedge [\pi(\mathcal{O}) \wedge \bigwedge_{1 \leq i \leq \ell} \tau(S_i \ op_i \ p_i)] \models Q(\bar{t})$$

holds in first-order logic, by Conditions (2) and (3) we have $\mathbf{K}\,\tau(A) \in P$ if and only if A is true in I. But then, Condition (4) implies that $I \models \mathcal{P}_G$. Assume that I is not a strong answer set of $\mathcal{K}_G$, so an interpretation $I' \subsetneq I$ exists such that $I' \models s\mathcal{P}_G^I$. Let $P' = \{\mathbf{K}\,\tau(A) \in \mathsf{KA}(\mathcal{K}'_G) \mid A \text{ is true in } I'\}$, $N' = \mathsf{HA}(\mathcal{K}'_G) \setminus P'$, and $P'_h = P' \cap \mathsf{HA}(\mathcal{K}'_G)$. Condition (5a) does not hold for (P', N'), so we have $A \in I'$ if and only if $\mathbf{K}\,\tau(A) \in P'$. But then, $I' \models s\mathcal{P}_G^I$ implies that Condition (5b) does not hold, which is a contradiction.

($\Rightarrow$) Let I be a strong answer set of $\mathcal{K}_G$. We show that the conditions in Table IV are satisfied for $P = \{\mathbf{K}\,\tau(A) \in \mathsf{KA}(\mathcal{K}'_G) \mid A \text{ is true in } I\}$, $N = \mathsf{KA}(\mathcal{K}'_G) \setminus P$, and $P_h = P \cap \mathsf{HA}(\mathcal{K}'_G)$. As in the previous paragraph, we can show that (P, N) satisfies Conditions (1)–(4). Assume now that Conditions (5a) and (5b) do not hold for some partition (P', N') of $\mathsf{KA}(\mathcal{K}'_G)$ and $P'_h = P' \cap \mathsf{HA}(\mathcal{K}'_G)$. Let $I' = \{A \mid \mathbf{K}\,A \in P'_h\}$. Partition (P', N') is not necessarily consistent; it is only weakly consistent: it is possible that $\mathbf{K}\,\tau(A) \in P'$, but A is false in I'. All such atoms $\mathbf{K}\,\tau(A)$, however, occur in the body of some rule in the positive program $\mathcal{P}'_G[P, N]$; hence, if $\mathcal{P}'_G[P, N][P', N'] = \mathsf{true}$, then $\mathcal{P}'_G[P, N]$ evaluates to true even if we make such $\mathbf{K}\,\tau(A)$ false. Hence, $I' \models s\mathcal{P}_G^I$, which is a contradiction. $\square$

Although we do not have a formal proof, we believe it is possible to extend this encoding to handle DL-atoms that contain $\Cap$, as well as to weak answer sets.

7.5 A Framework Based on Autoepistemic Logic

An approach related to ours has been pursued by de Bruijn et al. [2007], who have investigated the possibilities of using first-order autoepistemic logic (AEL) [Konolige 1991; Moore 1985] as a semantic framework for an integration of first-order logic and logic programming. They have presented several ways for embedding nonground logic programs into AEL, and they have extended this embedding to hybrid knowledge bases consisting of a first-order knowledge base and a logic program. No reasoning algorithm has been provided.

Rosati [1999] has shown that propositional AEL can be encoded into propositional MBNF in a simple way, and it is also easy to see that the same encoding is applicable in MKNF. No such result is known for first-order AEL by Konolige [1991]; however, we conjecture that a close relationship exists in this case as well. This could be used to establish a relationship between the hybrid KBs by de Bruijn et al. [2007] and MKNF$^+$ knowledge bases and thus obtain a reasoning algorithm.

7.6 *EQL-Lite*$(\mathcal{Q})$

Calvanese et al. [2007] introduced *EQL-Lite*$(\mathcal{Q})$ as a very general and powerful query language for description logics. This language is parameterized with a first-order DL query language $\mathcal{Q}$. An *EQL-Lite*$(\mathcal{Q})$ query is then defined as a first-order formula whose atoms are of the form $\mathbf{K}\,\xi$ with $\xi \in \mathcal{Q}$. Let φ be an *EQL-Lite*$(\mathcal{Q})$ query with $\bar{x}$ the set of free variables. A tuple of constants $\bar{a}$ is an *answer* to φ

over a DL knowledge base $\mathcal{O}$ iff $\mathcal{O} \models \varphi[\overline{a}/\overline{x}]$. If φ is a *domain independent formula* (Abiteboul et al. [1995] provide a formal definition) and answering queries of $\mathcal{Q}$ over $\mathcal{O}$ is decidable, then answering *EQL-Lite*$(\mathcal{Q})$ queries is decidable as well.

To transform an *EQL-Lite*$(\mathcal{Q})$ query φ with $\overline{x}$ the set of free variables into a set of MKNF rules, we start with the formula $\mathbf{K}\, Q(\overline{x}) \subset \varphi$ which we then transform using the well-known Lloyd-Topor transformation into rules [Lloyd and Topor 1984]. The correctness of this transformation immediately follows from the correctness of the Lloyd-Topor transformation. The resulting set of rules need not be DL-safe; however, we can make it DL-safe using the transformation presented after Definition 6.1. Since φ is domain-independent, it is not difficult to see that making the rules DL-safe does not change the semantics of the rules.

7.7 Extension of DLs with Default Logic

Default logic [Reiter 1980] is one of the basic nonmonotonic knowledge representation formalisms. A default theory (D, W) consists of a set of *world* formulae W, and a set of *default rules* D of the form $\alpha : \beta_1, \ldots, \beta_n / \gamma$, where α is the *premise* formula, $\beta_1, \ldots, \beta_n$ are the *justification* formulae, and γ is the *conclusion* formula; intuitively, this means "derive γ from α if assuming $\beta_1, \ldots, \beta_n$ does not lead to a contradiction." Baader and Hollunder [1995] explored the possibilities of extending DLs with default rules. They showed that adding open defaults (i.e., default rules with free variables) to DLs leads to undecidability of the basic reasoning problems. Therefore, they proposed an alternative semantics in which default rules are applicable only to the known individuals, for which they presented a decision procedure.

To define the semantics of open defaults, Reiter [1980] employed skolemization and grounded the default rules w.r.t. the Herbrand universe. Lifschitz [1990] proposed an alternative semantics that does not employ skolemization, but in which defaults are interpreted over a fixed universe. The two versions of default logic coincide only under certain conditions. Furthermore, Lifschitz [1991] presented a relationship between defaults and MKNF: every default rule $\alpha : \beta_1, \ldots, \beta_n / \gamma$ can be encoded using the following MKNF formula.

$$\mathbf{K}\, \alpha \wedge \mathbf{not}\, \neg\beta_1 \wedge \ldots \wedge \mathbf{not}\, \neg\beta_n \supset \mathbf{K}\, \gamma \tag{101}$$

This equivalence holds only for the version of the default logic by Lifschitz [1990], and not by Reiter [1980].

Using the encoding (101), MKNF knowledge bases can be used to extend DL knowledge bases. In fact, the DL-safety restriction makes such defaults applicable only to the known individuals. Under such an assumption, the logics by Reiter [1980] and Lifschitz [1990] coincide [Lifschitz 1990, Propositions 4 and 5], so the results from this paper provide an alternative characterization and a proof procedure for the formalism by Baader and Hollunder [1995].

7.8 DLs of Minimal Knowledge and Negation as Failure

Because it provides a unifying framework for many different nonmonotonic formalisms, MKNF naturally lends itself as a basis for incorporating nonmonotonic features into first-order formalisms. Following this idea, Donini et al. [2002] proposed $\mathcal{ALCK}_{\mathcal{NF}}$—an extension of the basic description logic $\mathcal{ALC}$ with nonmonotonic operators $\mathbf{K}$ and $\mathbf{A}$. The first operator is the standard $\mathbf{K}$ operator of MKNF, and **A** is semantically equivalent to $\neg$ **not**. These extensions can be put to numerous uses, such as modeling default rules, expressing integrity constraints, and formalizing role and concept closure. For example, Grimm and Hitzler [2008] applied such extensions to the problem of semantic matchmaking of Web resources.

$\mathcal{ALCK}_{\mathcal{NF}}$ theories and MKNF KBs have incomparable expressivity: the former does not provide for arbitrary rules; however, it does allow the application of modal operators to open formulae. The latter is important as it allows one to perform nonmonotonic reasoning with unnamed individuals. To achieve decidability, the authors impose a range of restrictions on the places where modal operators can occur in axioms. The resulting logic has a relatively complex proof theory. In contrast, the proof theory for MKNF knowledge bases can be seen as an extension of the proof theory for answer set programs.

7.9 Extensions of DLs with Circumscription

The formalization of circumscription in second-order logic [McCarthy 1980] provides a powerful framework for nonmonotonic reasoning in first-order logic. Since DLs are first-order fragments, the semantics of circumscription can be straightforwardly applied to description logics. Recently, Bonatti et al. [2009] have studied the computational complexity of reasoning in DLs with circumscription, and have presented a number of undecidability results, as well as restrictions that can be used to make reasoning decidable.

DLs with circumscription and MKNF KBs have incomparable expressivity: the former does not provide for arbitrary rules, but they allow for intensional reasoning and can therefore be used, for example, to model exceptions in a DL TBox.

7.10 Well-Founded Semantics for MKNF Rules

Knorr et al. [2008] have recently proposed an extension of MKNF knowledge bases to the well-founded semantics [van Gelder et al. 1991]. MKNF is a two-valued logic which cannot directly capture the well-founded models, which are three-valued. Therefore, the authors have first extended MKNF to a three-valued logic, which can then be applied to MKNF knowledge bases in a standard way. Furthermore, the authors have presented an alternating fixpoint algorithm for computing well-founded MKNF models, as well as the corresponding upper complexity bound.

8. CONCLUSION

Using the logic MKNF by Lifschitz [1991] as foundation, in this paper we presented the formalism of MKNF knowledge bases, which seamlessly integrates DLs with answer set programming. This gives us a powerful hybrid formalism that combines the best features of both worlds: on the one hand, it provides DL-style modeling of taxonomic knowledge, and on the other hand, it provides ASP-style constructs, such as negation as failure. We have shown that MKNF knowledge bases provide us with the first faithful, tight, and flexible formalism that integrates description logics and ASP. To ensure decidability, we apply the well-known DL-safety restriction that makes the rules applicable only to explicitly known individuals, thus trading some expressivity for decidability.

We presented several reasoning algorithms for different fragments of our logic. Furthermore, we analyzed the combined complexity of the general formalism and the data complexities of different fragments. Our results show that, in many cases, reasoning with MKNF knowledge bases is not harder than reasoning in the corresponding fragment of logic programming.

There are many challenging problems left for future work, such as to study reasoning in classes of MKNF KBs in the absence of the DL-safety assumption, or to develop the optimization of the reasoning algorithms for more specific DLs and classes of ASP rules. On the practical side, the main problem is to implement a system supporting MKNF KBs and apply it in practical applications. As we discussed in Section 4.1, we believe that the algorithms presented in this paper provide a good starting point for the development of practical systems for reasoning with MKNF knowledge bases.

ACKNOWLEDGMENTS

The authors thank Jos de Bruijn, Thomas Lukasiewicz, and the anonymous referees for their numerous comments, which have significantly contributed to the quality of this paper.

REFERENCES

Abiteboul, S., Hull, R., and Vianu, V. 1995. *Foundations of Databases.* Addison Wesley.

Antoniou, G. 1997. *Nonmonotonic Reasoning.* MIT Press.

Baader, F., Brandt, S., and Lutz, C. 2005. Pushing the $\mathcal{EL}$ Envelope. In *Proc. of the 19th Int. Joint Confön Artificial Intelligence (IJCAI-05).* Morgan-Kaufmann Publishers, Edinburgh, UK, 364–369.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F., Eds. 2007. *The Description Logic Handbook: Theory, Implementation and Applications*, 2nd ed. Cambridge University Press.

Baader, F. and Hollunder, B. 1995. Embedding Defaults into Terminological Knowledge Representation Formalisms. *Journal of Automated Reasoning 14,* 1, 149–180.

Bonatti, P., Lutz, C., and Wolter, F. 2009. The Complexity of Circumscription in Description Logic. *Journal of Artificial Intelligence Research 35*, 717–773.

Börger, E., Grädel, E., and Gurevich, Y. 1996. *The Classical Decision Problem.* Springer.

Cadoli, M., Palopoli, L., and Lenzerini, M. 1997. Datalog and Description Logics: Expressive Power. In *Proc. of the 6th Int. Workshop on Database Programming Languages (DBLP-6).* LNCS, vol. 1369. Springer, Estes Park, CO, USA, 281–298.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. *Journal of Automated Reasoning 9*, 385–429.

Calvanese, D., Lenzerini, M., and Nardi, D. 1998. Description Logics for Conceptual Data Modeling. In *Logics for Databases and Information Systems*, J. Chomicki and G. Saake, Eds. Kluwer Academic Publishers, Chapter 8, 229–263.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. 2006. Data Complexity of Query Answering in Description Logics. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, P. Doherty, J. Mylopoulos, and C. A. Welty, Eds. AAAI Press, Lake District, UK.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. 2007. EQL-Lite: Effective First-Order Query Processing in Description Logics. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*, M. M. Veloso, Ed. Morgan Kaufmann Publishers, Hyderabad, India, 274–279.

Calvanese, D., De Giacomo, G., and Lenzerini, M. 1998. On the Decidability of Query Containment under Constraints. In *Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '98).* ACM Press, Seattle, WA, USA, 149–158.

Dantsin, E., Eiter, T., Gottlob, G., and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Computing Surveys 33,* 3, 374–425.

de Bruijn, J., Eiter, T., Polleres, A., and Tompits, H. 2007. Embedding Non-Ground Logic Programs into Autoepistemic Logic for Knowledge-Base Combination. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*, M. M. Veloso, Ed. Morgan Kaufmann Publishers, Hyderabad, India, 304–309.

de Bruijn, J., Lausen, H., Polleres, A., and Fensel, D. 2006. The Web Service Modeling Language: An Overview. In *Proc. of the 3rd European Semantic Web Conference (ESWC2006).* LNCS, vol. 4011. Springer, Budva, Serbia and Montenegro. 590–604.

de Bruijn, J., Pearce, D., Polleres, A., and Valverde, A. 2007. Quantified Equilibrium Logic and Hybrid Rules. In *Proc. of the 1st Int. Conf. on Web Reasoning and Rule Systems (RR 2007)*, M. Marchiori, J. Z. Pan, and C. de Sainte Marie, Eds. Innsbruck , Austria, 58–72.

de Bruijn, J., Polleres, A., Lara, R., and Fensel, D. 2005. OWL DL vs. OWL Flight: Conceptual Modeling and Reasoning on the Semantic Web. In *Proc. of the 14th Int. World Wide Web Conference (WWW2005).* ACM, Chiba, Japan, 623–632.

Derriere, S., Richard, A., and Preite-Martinez, A. 2006. An Ontology of Astronomical Object Types for the Virtual Observatory. In *Proc. of the 26th meeting of the IAU: Virtual Observatory in Action: New Science, New Technology, and Next Generation Facilities.* Prague, Czech Republic, 17–18.

Donini, F. M., Lenzerini, M., Nardi, D., and Schaerf, A. 1998. AL-log: Integrating Datalog and Description Logics. *Journal of Intelligent Information Systems 10,* 3, 227–252.

Donini, F. M., Nardi, D., and Rosati, R. 2002. Description Logics of Minimal Knowledge and Negation as Failure. *ACM Transactions on Computational Logic 3,* 2, 177–225.

Eiter, T., Fink, M., Tompits, H., and Woltran, S. 2004. On Eliminating Disjunctions in Stable Logic Programming. In *Proc. of the 9th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004)*, D. Dubois, C. A. Welty, and M.-A. Williams, Eds. AAAI Press, Whistler, Canada, 447–458.

Eiter, T., Gottlob, G., and Mannila, H. 1997. Disjunctive Datalog. *ACM Transactions on Database Systems 22,* 3, 364–418.

Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., and Tompits, H. 2008. Combining answer set programming with description logics for the Semantic Web. *Artificial Intelligence 172,* 12–13, 1495–1539.

Eiter, T., Leone, N., Mateis, C., Pfeifer, G., and Scarcello, F. 1997. A Deductive System for Non-Monotonic Reasoning. In *Proc. of the 4th Int. Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR '97)*, J. Dix, U. Furbach, and A. Nerode, Eds. LNAI, vol. 1265. Springer, Dagstuhl, Germany, 364–375.

Fikes, R. and Kehler, T. 1985. The Role of Frame-based Representation in Reasoning. *Communications of the ACM 28,* 9, 904–920.

Fitting, M. 1996. *First-Order Logic and Automated Theorem Proving, 2nd Edition.* Texts in Computer Science. Springer.

Gelfond, M. and Lifschitz, V. 1988. The Stable Model Semantics for Logic Programming. In *Proc. of the 5th Int. Conf. on Logic Programming (ICLP '88).* MIT Press, Seattler, WA, USA, 1070–1080.

Gelfond, M. and Lifschitz, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing 9,* 3–4, 365–386.

Glimm, B., Horrocks, I., Lutz, C., and Sattler, U. 2007. Conjunctive Query Answering for the Description Logic $\mathcal{SHIQ}$. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007).* Morgan Kaufmann Publishers, India, 399–404.

Golbreich, C., Zhang, S., and Bodenreider, O. 2006. The Foundational Model of Anatomy in OWL: Experience and Perspectives. *Journal of Web Semantics 4,* 3, 181–195.

Goodwin, J. 2005. Experiences of using OWL at the Ordnance Survey. In *Proc. of the OWL: Expreiences and Directions Workshop (OWLED 2005).* CEUR WS Proceedings, vol. 188. Galway, Ireland.

Gottlob, G. 1995. NP Trees and Carnap's Modal Logic. *Journal of the ACM 42,* 2, 421–457.

Grimm, S. and Hitzler, P. 2008. Semantic Matchmaking of Web Resources with Local Closed-World Reasoning. *International Journal of e-Commerce 12,* 2, 89–126.

Grosof, B. N., Horrocks, I., Volz, R., and Decker, S. 2003. Description Logic Programs: Combining Logic Programs with Description Logic. In *Proc. of the 12th Int. World Wide Web Conference (WWW 2003).* ACM Press, Budapest, Hungary, 48–57.

Haarslev, V. and Möller, R. 2001. RACER System Description. In *Proc. of the 1st Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, R. Goré, A. Leitsch, and T. Nipkow, Eds. LNAI, vol. 2083. Springer, Siena, Italy, 701–706.

Hayes, P. J. 1979. The Logic of Frames. In *Frame Conceptions and Text Understanding*, D. Metzing, Ed. Walter de Gruyter and Co., 46–61.

Horrocks, I. 1998. Using an Expressive Description Logic: FaCT or Fiction? In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR '98)*, A. G. Cohn, L. Schubert, and S. C. Shapiro, Eds. Morgan Kaufmann Publishers, Trento, Italy, 636–647.

Horrocks, I., Parsia, B., Patel-Schneider, P. F., and Hendler, J. 2005. Semantic web architecture: Stack or two towers? In *Proc. of the 3rd Int. Workshop on Principles and Practice of Semantic Web Reasoning (PPSWR 2005)*, F. Fages and S. Soliman, Eds. LNCS, vol. 3703. Springer, Dagstuhl Castle, Germany, 37–41.

Horrocks, I., Patel-Schneider, P. F., Bechhofer, S., and Tsarkov, D. 2005. OWL rules: A proposal and prototype implementation. *Journal of Web Semantics 3,* 1, 23–40.

Hustadt, U., Motik, B., and Sattler, U. 2007. Reasoning in Description Logics by a Reduction to Disjunctive Datalog. *Journal of Automated Reasoning 39,* 3, 351–384.

Kazakov, Y. 2008. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are Harder than $\mathcal{SHOIQ}$. In *Proc. of the 11th Int. Joint Conf. on Principles of Knowledge Representation and Reasoning (KR 2008)*, G. Brewka and J. Lang, Eds. AAAI Press, Sydney, NSW, Australia, 274–284.

Kifer, M., de Bruijn, J., Boley, H., and Fensel, D. 2005. A Realistic Architecture for the Semantic Web. In *Proc. of the Int. Conference on Rules and Rule Markup Languages for the Semantic Web (RuleML 2005)*, A. Adi, S. Stoutenburg, and S. Tabet, Eds. LNCS, vol. 3791. Springer, Galway, Ireland, 17–29.

Kifer, M., Lausen, G., and Wu, J. 1995. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM 42,* 4, 741–843.

Knorr, M., Alferes, J. J., and Hitzler, P. 2008. A Coherent Well-founded Model for Hybrid MKNF Knowledge Bases. In *Proc. of the 18th European Conf. on Artificial Intelligence (ECAI 2008)*, M. Ghallab, C. D. Spyropoulos, N. Fakotakis, and N. M. Avouris, Eds. Frontiers in Artificial Intelligence and Applications, vol. 178. IOS Press, Patras, Greece, 99–103.

Konolige, K. 1991. Quantification in Autoepistemic Logic. *Fundamenta Informaticae 15,* 3–4, 275–300.

Krötzsch, M., Rudolph, S., and Hitzler, P. 2008a. Conjunctive Queries for a Tractable Fragment of OWL 1.1. In *Proc. of the 7th Int. Semantic Web Conference (ISWC 2008)*, A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. Finin, and K. Thirunarayan, Eds. LNCS, vol. 5318. Springer, Karlsruhe, Germany, 310–323.

Krötzsch, M., Rudolph, S., and Hitzler, P. 2008b. ELP: Tractable Rules for OWL 2. In *Proc. of the 7th Int. Semantic Web Conference (ISWC 2008)*, A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. Finin, and K. Thirunarayan, Eds. LNCS, vol. 5318. Springer, Karlsruhe, Germany, 649–664.

Kutz, O., Horrocks, I., and Sattler, U. 2006. The Even More Irresistible SROIQ. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, P. Doherty, J. Mylopoulos, and C. A. Welty, Eds. AAAI Press, Lake District, UK, 68–78.

Lacy, L., Aviles, G., Fraser, K., Gerber, W., Mulvehill, A., and Gaskill, R. 2005. Experiences Using OWL in Military Applications. In *Proc. of the OWL: Expreiences and Directions Workshop (OWLED 2005).* CEUR WS Proceedings, vol. 188. Galway, Ireland.

Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., and Scarcello, F. 2006. The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic 7,* 3, 499–562.

Levesque, H. J. 1984. Foundations of a Functional Approach to Knowledge Representation. *Artificial Intelligence 23,* 2, 155–212.

Levy, A. Y. and Rousset, M.-C. 1998. Combining Horn Rules and Description Logics in CARIN. *Artificial Intelligence 104,* 1–2, 165–209.

Lifschitz, V. 1990. On Open Defaults. In *Proc. of the Symposium on Computational Logic.* ESPRIT Basic Research Series. Springer, Bruxelles, Belgium, 80–95.

Lifschitz, V. 1991. Nonmonotonic Databases and Epistemic Queries. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI '91)*, J. Mylopoulos and R. Reiter, Eds. Morgan Kaufmann Publishers, Sydney, Australia, 381–386.

Lifschitz, V. 1994. Minimal Belief and Negation as Failure. *Artificial Intelligence 70,* 1–2, 53–72.

Lloyd, J. W. and Topor, R. W. 1984. Making Prolog More Expressive. *Journal of Logic Programming 1,* 3, 225–240.

Lukasiewicz, T. 2007. A Novel Combination of Answer Set Programming with Description Logics for the Semantic Web. In *Proc. of the 4th European Semantic Web Conference (ESWC 2007)*, E. Franconi, M. Kifer, and W. May, Eds. LNCS, vol. 4519. Springer, Innsbruck, Austria, 384–398.

MacGregor, R. 1991. Inside the LOOM Description Classifier. *SIGART Bulletin 2,* 3, 88–92.

Marek, W. and Truszczyński, M. 1993. *Nonmonotonic Logics: Context-Dependent Reasoning.* Springer.

McCarthy, J. 1980. Circumscription—A Form of Nonmonotonic Reasoning. *Artificial Intelligence* 13, 27–39.

Moore, R. C. 1985. Semantical considerations on nonmonotonic logic. *Artificial Intelligence 25*, 75–94.

Motik, B., Sattler, U., and Studer, R. 2005. Query Answering for OWL-DL with rules. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web 3,* 1, 41–60.

Ortiz, M., Calvanese, D., and Eiter, T. 2008. Data Complexity of Query Answering in Expressive Description Logics via Tableaux. *Journal of Automated Reasoning 41,* 1, 61–98.

Papadimitriou, C. H. 1993. *Computational Complexity.* Addison Wesley.

Parsia, B. and Sirin, E. 2004. Pellet: An OWL-DL Reasoner. In *Poster at the 3rd Int. Semantic Web Conference (ISWC 2004).* Hiroshima, Japan.

Patel-Schneider, P. F., Hayes, P., and Horrocks, I. 2004. OWL Web Ontology Language: Semantics and Abstract Syntax, W3C Recommendation. `http://www.w3.org/TR/owl-semantics/`.

Patel-Schneider, P. F., McGuinness, D. L., Brachman, R. J., Resnick, L. A., and Borgida, A. 1991. The CLASSIC Knowledge Representation System: Guiding Principles and Implementation Rational. *SIGART Bulletin 2,* 3, 108–113.

Pearce, D. and Valverde, A. 2005. A First Order Nonmonotonic Extension of Constructive Logic. *Studia Logica 80,* 2–3, 321–346.

Reiter, R. 1980. A Logic for Default Reasoning. *Artificial Intelligence 13,* 1–2, 81–132.

Reiter, R. 1992. What Should a Database Know? *Journal of Logic Programming 14,* 1–2, 127–153.

Rosati, R. 1999. Reasoning about Minimal Belief and Negation as Failure. *Journal of Artificial Intelligence Research 11*, 277–300.

Rosati, R. 2003. Minimal Belief and Negation as Failure in Multi-Agent Systems. *Annals of Mathematics and Artificial Intelligence 37,* 1–2, 5–32.

Rosati, R. 2005. On the decidability and complexity of integrating ontologies and rules. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web 3,* 1, 61–73.

Rosati, R. 2006. $\mathcal{DL}+log$: A Tight Integration of Description Logics and Disjunctive Datalog. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, P. Doherty, J. Mylopoulos, and C. A. Welty, Eds. AAAI Press, Lake District, UK, 68–78.

SIDHU, A., DILLON, T., CHANG, E., AND SIDHU, B. S. 2005. Protein Ontology Development using OWL. In *Proc. of the OWL: Expreiences and Directions Workshop (OWLED 2005)*. CEUR WS Proceedings, vol. 188. Galway, Ireland.

SOERGEL, D., LAUSER, B., LIANG, A., FISSEHA, F., KEIZER, J., AND KATZ, S. 2004. Reengineering Thesauri for New Applications: The AGROVOC Example. *Journal of Digital Information 4,* 4.

STOCKMEYER, L. 1976. The polynomial-time hierarchy. *Theoretical Computer Science 3*, 1–22.

SYRJÄNEN, T. AND NIEMELÄ, I. 2001. The Smodels System. In *Proc. 6th Int. Conf. on Logic Programming and Nonmonotonic Reasoning (LPNMR 2001)*, T. Eiter, W. Faber, and M. Truszczynski, Eds. LNAI, vol. 2173. Springer, Vienna, Austria, 434–438.

TOBIES, S. 2001. Complexity Results and Practical Algorithms for Logics in Knowledge Representation. Ph.D. thesis, RWTH Aachen, Germany.

TSARKOV, D. AND HORROCKS, I. 2006. FaCT++ Description Logic Reasoner: System Description. In *Proc. of the 3rd Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*. LNAI, vol. 4130. Springer, Seattle, WA, USA, 292–297.

VAN GELDER, A., ROSS, K., AND SCHLIPF, J. S. 1991. The Well-Founded Semantics for General Logic Programs. *Journal of the ACM 38,* 3, 620–650.

VARDI, M. 1982. The Complexity of Relational Query Languages (Extended Abstract). In *Proc. of the 14th annual ACM Symposium on Theory of Computing (STOC '82)*, H. R. Lewis, B. B. Simons, W. A. Burkhard, and L. Landweber, Eds. ACM Press, San Francisco, CA, USA, 137–146.

VARDI, M. Y. 1996. Why Is Modal Logic So Robustly Decidable? In *Proc. of a DIMACS Workshop on Descriptive Complexity and Finite Models*, N. Immerman and P. Kolaitis, Eds. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 31. American Mathematical Society, Princeton University, USA, 149–184.

WOODS, W. A. AND SCHMOLZE, J. G. 1992. The KL-ONE Family. *Computers & Mathematics with Applications 23,* 2–5, 133–177.

YEN, J., NECHES, R., AND MACGREGOR, R. 1991. CLASP: Integrating Term Subsumption Systems and Production Systems. *IEEE Transactions on Knowledge and Data Engineering 3,* 1, 25–31.

Received Month Year; revised Month Year; accepted Month Year