Paving the Way for Efficient Disjunctive Hybrid MKNF Knowledge Base Solvers

by

Spencer Killen

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Can efficient solvers be built for disjunctive hybrid MKNF knowledge bases? Recent breakthroughs in solver construction have proven Answer Set Programming (ASP) to be a fruitful medium for tackling problems that lie within the lower two levels of the polynomial hierarchy. The logic of hybrid MKNF presents a powerful knowledge representation language by elegantly pairing ASP with ontologies. It would seem that strong answer set solvers that incorporate the reasoning power of ontologies are within close grasp. Alas, some of the foundational work required to construct conflict-driven nogood learning (CDNL) solvers does not yet exist for hybrid MKNF. In this thesis, we build upon the existing foundation of hybrid MKNF to move towards the possibility of constructing a CDNL-based solver. We lift the existing definition of unfounded sets for knowledge bases with non-disjunctive rules to knowledge bases that contain disjunctive rules. Unfounded sets serve as the basis for building operators that can be used to propagate information during the solving process. Because our operators cannot be used to verify arbitrary interpretations, we provide a characterization of knowledge bases that leverages a family of fixpoint operators for model-checking. This characterization is reminiscent of prior techniques that leverage loop formulas to identify classes of knowledge bases that can be solved in NP-time. We identify an analogous case for disjunctive hybrid MKNF where interpretations can be verified in polynomial time instead of requiring an NP oracle. Finally, we outline an abstract solver that utilizes this technique.

Preface

This work shares content with two papers submitted to seperate conferences. Each paper shares the primary author with this thesis and was coauthored by Jia-Huai You. The content in Chapter 4 was submitted and accepted to the 18th International Conference on Principles of Knowledge Representation and Reasoning (KR 2021) under the title "Unfounded Sets for Disjunctive Hybrid MKNF Knowledge Bases". Some feedback from anonymous reviewers and editors was incorporated into the text. The content in Chapter 5 was submitted to the 14th Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2021) under the title "Fixpoint Characterizations of Disjunctive Hybrid MKNF Knowledge Bases". At the time of writing, the notification date for acceptance into this conference has yet not passed.

Acknowledgements

The work contained in this thesis was partly funded through an NSERC CGS M scholarship. I would like to acknowledge and appreciate the feedback from the thesis from Dr. Marek Reformat and Dr. Randy Goebel who served as examiners in my thesis defence. I was also would to extend enormous gratitude to Dr. Jia-Huai You whose supervisorship and guidance saw me through completing this work.

Contents

Chapter 1

Introduction

Combining answer set programming (ASP) with ontologies is of great theoretical and practical interest. While researchers have proposed a variety of hybrid semantics and have developed efficient CDNL (conflict-driven nogood learning) solvers for them [4], [5], these semantics trend towards ASP modulo theories, a class of semantics where stable models are validated by an external theory. Disjunctive rules are a powerful extension to answer set programming that increase the expressive power of programs in the polynomial complexity hierarchy [3]. Disjunctive hybrid MKNF knowledge bases, introduced by Motik and Rosati [19], can succinctly extend disjunctive ASP with ontologies without increasing the complexity of reasoning tasks. However, in many cases, solver development is lagging behind. As the result, the only known method of computing models for disjunctive hybrid MKNF knowledge bases is based on guess-and-verify, as formulated by Motik and Rosati in their original work. Proven solver techniques, such as constraint-driven nogood learning (CDNL) [7], are essential to constructing highly efficient ASP solvers.

Existing theoretical foundation that enables efficient answer set solvers can be transferred to disjunctive Hybrid MKNF Knowledge bases. In this work, we address some issues that must be solved before a CDNL-based (conflict-driven nogood learning) solver for hybrid MKNF knowledge bases can be developed. A main obstacle to constructing such solvers is understanding how constraint propagation may be performed by a solver, which, in the context of ASP, centers around the computation of unfounded atoms, the atoms that are false

given a partial interpretation. Unfounded sets are leveraged extensively in modern solving techniques and thus are an essential component of building a solver. We define unfounded sets for disjunctive hybrid MKNF knowledge bases, identify lower complexity bounds, demonstrate the role these developments play in constructing a solver, and compare our definition to prior work.

Another obstacle to developing an efficient CDNL-based solver for disjunctive hybrid MKNF is the unavailability of syntactic dependency; In general, a dependency graph can not be generated for a hybrid knowledge base without knowing the internal structure of the ontology. A framework that treats the ontology as a black box has a broader range of applications than a solver that must be tuned for a particular ontology. Atom dependency analysis is crucial for both model verification and conflict generation. Existing solvers are heavily reliant on syntactic dependency. We investigate constructions that fill the role of loop formulas for ASP: We characterize disjunctive hybrid MKNF semantics through a family of fixpoint operators, then contextualize this framework by demonstrating the role it would play in a solver. Crucially, our approach can be performed without relying on a dependency graph. Finally, we recognize a property of our characterization that is analogous to head-cycle free disjunctive logic programs and demonstrate how to exploit this property to reduce the complexity of model verification.

In Chapter 2 we give a loose overview of answer set programming; this is intended as an intuitive introduction to the language. A reader who is already familiar with answer-set programming may wish to skip this chapter and progress to Chapter 3 where we formally outline the preliminary material pertinent to the rest of the text. Chapters 4 and 5 are heavily reliant on this preliminary chapter. Chapter 4 defines unfounded sets for disjunctive hybrid MKNF knowledge bases and Chapter 5 introduces our fixpoint characterizations of these knowledge bases. Though discussed in separate chapters, the issue of unfounded sets and dependency turn out to be tightly intertwined. To conclude, we offer a deeper discussion of how the work in these chapters connects in Chapter 6.

1.1 Contributions

The major contributions of these work are as follows: In Chapter 4, We give a definition of unfounded sets for disjunctive hybrid MKNF knowledge bases, we provide lower-bound complexity analysis for computing the greatest unfounded set for the normal and disjunctive cases, we give well-founded operators that compute an approximation of the the greatest unfounded set, and we incorporate these operators into a DPLL-based solver. In Chapter 5, we give a fixedpoint characterization of disjunctive MKNF knowledge bases that can be used for model-checking and as an alternative to loop-formulas.

Chapter 2

Answer Set Programming

2.1 ASP Semantics

Answer set programming (ASP) is a class of logic programming focused on bottom-up reasoning to find a minimal model that satisfies all rules in a logic program. Established after Gelfond and Lifschitz defined stable model semantics [8], [10], ASP's underlining semantics, ASP has since been recast under a variety of different semantics. ASP succinctly expresses NP-complete problems, as well as more complex problems in the level above NP in the polynomial hierarchy. The class of NP-complete problems can be loosely described as the class of problems that have combinatorially large search spaces but also have the property that solutions can be verified in polynomial time. Problems in the level above NP, Σ_2^P, are defined in the same way as NP with the difference that a verification procedure has access to an NP oracle. Many practical problems such as graph colouring or optimization fall within the classes NP or Σ_2^P. While ASP shares its niche with SAT (boolean satisfiability problems), ASP is distinct from SAT in that programs can easily be written and read by humans. In ASP, logical consequences are directly encoded through rules and classical negation is absent; instead ASP uses the intuitionistic negation as failure. In this paradigm, negated facts are derived if there is no positive proof. For example, one might assume that pigs do not fly if they have not seen a pig fly.

In this section, we give a loose overview of answer set programming. Later, in Chapters 4 and 5 we supersede this overview with definitions that are more

rigorous. Below, we give an ASP program that contains a single rule.

$$a \leftarrow \textbf{not}\, b.$$

Under stable model semantics, the above program has a single stable model (also called an answer set). This model is the boolean assignment that assigns a to true and b to false. This assignment is a model because it satisfies two conditions: (1) Each rule in the program is satisfied: The rule's body ($\textbf{not}\, b$) is satisfied by the assignment, so the rule's head (a) must also be satisfied by the assignment (this works just like logical implication). (2) There is no smaller assignment that satisfies the GL-reduct (Gelfond-Lifschitz reduct) with respect to the initial assignment. Here, a "smaller assignment" is formed by changing some atoms that were assigned true to false instead. For the model of the above program ("a is true, b is false"), the only smaller assignment is the assignment that assigns false to both a and b. By "GL-reduct with respective to the initial assignment", we mean a new program obtained by removing negation from the program as follows: Rules that contain negated atoms in their bodies are removed entirely if these atoms are assigned true in the assignment. Otherwise, the rule is kept and the negated atoms are removed from its body. Below, we give the appropriate GL-reduct for the assignment "a is true, b is false":

$$a \leftarrow$$

The atom $\textbf{not}\, b$ has been removed from the rule because it is false with respect to the assignment. Because the body of the rule is now empty, it is automatically satisfied, thus a must be true. However, this is not the case for the assignment that assigns both a and b to be false. Because there is no smaller assignment, "a is true, b is false" is an answer set of the program.

Now for a program that has multiple rules.

$$a \leftarrow \textbf{not}\, b$$

$$b \leftarrow \textbf{not}\, a$$

Consider the assignment that assigns both a and b to true. To verify whether this assignment is an answer set, we first confirm that all rules in the program

are satisfied. Both rules are satisfied because their bodies are not satisfied by the assignment. The GL-reduct of this program with respect to this assignment is an empty program without any rules: because each rule contains negated atoms that are not satisfied by the assignment, we remove each rule entirely. We can generate a smaller assignment that assigns both a and b to be false which satisfies the empty program, thus the initial assignment "a is true, b is true" is not an answer set of the program. The assignments "a is true, b is false" and "a is false, b is true" are the only answer sets of the program.

To effectively encode problems as answer set programs, it is useful to have variables in programs and to add a grounding phase to the solving process. In this grounding pase, the program is replaced with an equivalent program that does not contain variables. Take the following program that contains the variables X and Y.

$$\begin{aligned} \mathit{letter}(a) &\leftarrow \mathbf{not}\, \mathit{letter}(b) \\ \mathit{letter}(b) &\leftarrow \mathbf{not}\, \mathit{letter}(a) \\ \mathit{predicate}(X) &\leftarrow \mathit{letter}(X), \mathbf{not}\, \mathit{letter}(Y) \end{aligned}$$

Various constraints such as requiring that variables appear in the body of the rule are commonplace and ensure that the program can be grounded and that the grounded program is meaningful. Below, we give an equivalent, grounded version of the program above:

$$\begin{aligned} \mathit{letter}(a) &\leftarrow \mathbf{not}\, \mathit{letter}(b) \\ \mathit{letter}(b) &\leftarrow \mathbf{not}\, \mathit{letter}(a) \\ \mathit{predicate}(a) &\leftarrow \mathit{letter}(a), \mathbf{not}\, \mathit{letter}(a) \\ \mathit{predicate}(a) &\leftarrow \mathit{letter}(a), \mathbf{not}\, \mathit{letter}(b) \\ \mathit{predicate}(b) &\leftarrow \mathit{letter}(b), \mathbf{not}\, \mathit{letter}(a) \\ \mathit{predicate}(b) &\leftarrow \mathit{letter}(b), \mathbf{not}\, \mathit{letter}(b) \end{aligned}$$

A variety of other language features that are syntactic sugar exist to make encoding problems easier. Programs that utilize these features can be efficiently converted to equivalent programs that do not use these features. One

such feature is so-called constraints which are rules that do not contain atoms in their head. If an assignment satisfies the body of a constraint, then it is not an answer set.

We do not dwell on syntactic sugar and instead focus on a particular language extension which, in general, can not be efficiently removed from a program. We call programs that allow for disjunction in the heads of rules *disjunctive programs*. Conversely, we refer to programs that do not contain disjunction in the heads of rules as *normal programs*. Under this extension, we allow for multiple atoms to appear in the head of a rule, for example:

$$a, b \leftarrow$$

Here, the program has two answer sets. Each assigns a single atom to be true and the other to be false. The assignment that assigns both atoms to true is not an answer set because of these smaller assignments that satisfy the program. The above program can easily be converted to an equivalent normal program with the same answer sets that does not contain disjunction in the heads of rules.

$$a \leftarrow \textbf{not}\, b$$
$$b \leftarrow \textbf{not}\, a$$

However, not all disjunctive programs can be efficiently converted to a normal program in this way. Take the following program for example.

$$\begin{aligned} a, b &\leftarrow \\ a &\leftarrow \textbf{not}\, a \\ b &\leftarrow a \end{aligned}$$

The only answer set to this program is the assignment that assigns both a and b to true. When we perform the process used before to convert this program

to a normal program, we result in the following program.

$$a \leftarrow \textbf{not}\, b$$

$$b \leftarrow \textbf{not}\, a$$

$$a \leftarrow \textbf{not}\, a$$

$$b \leftarrow a$$

The GL-reduct of this transformed program with respect to the assignment "a is true, b is true" is as follows.

$$b \leftarrow a$$

The assignment that assigns false to both a and b satisfies this new program, thus the assignment that assigns both a and b to true is not an answer set of the new, normal program above. It follows that this normal program is not equivalent to the initial disjunctive program.

While one may try to find an efficient procedure to convert any disjunctive program to an equivalent normal program, such a discovery would have radical complexity implications. It has been shown that, in general, determining whether a disjunctive program has an answer set has a complexity of Σ_2^P [3]. This complexity class is constructed similarly to the complexity class NP, however, it is thought to be much more difficult than NP; While NP stipulates that a solution must be verifiable in polynomial time, problems in the class Σ_2^P can invoke an NP oracle in their otherwise polynomial verification process. Thus, a polynomial algorithm that converts any disjunctive logic program to a normal logic program does not exist unless $\Sigma_2^P = NP$, which is still an open problem. There is a certain class of disjunctive programs (referred to as head-cycle free programs) that can be converted to equivalent normal programs in polynomial time using a known procedure [16]. In the previous example, we successfully converted the single-ruled disjunctive program, which is head-cycle free, to an equivalent normal program using this procedure.

2.2 Encoding Problems

We apply the semantics outlined in Subsection 2.1 to encode an instance of the graph colouring problem as an answer set program whose answer sets coincide with the solutions to the problem. In graph colouring, we are given a graph containing nodes connected via edges. In the following, we give a graph containing three nodes as an answer set program.

$$node(a) \leftarrow$$
$$node(b) \leftarrow$$
$$node(c) \leftarrow$$
$$edge(a, b) \leftarrow$$
$$edge(b, a) \leftarrow$$
$$edge(b, c) \leftarrow$$
$$edge(c, b) \leftarrow$$

Then we add a rule to ensure that each node is assigned some colour (either red, green, or blue).

$$coloured(N, red),\ coloured(N, green),\ coloured(N, blue) \leftarrow node(N)$$

Finally, we add a rule to ensure that adjacent nodes do not share the same color

$$invalid \leftarrow edge(X, Y),\ coloured(X, C),\ coloured(Y, C),\ \textbf{not}\, invalid$$

Note that because this is the only rule that derives the atom *invalid*, it is not possible for it to be true in any answer set. The final program has answer sets that coincide with a two-colour and three-colour solutions to the above problem. For example, the following is part of a three-coloured solution.

$$coloured(a, red)$$
$$coloured(b, green)$$
$$coloured(c, blue)$$

The following is part of a two-coloured solution.

$$coloured(a, red)$$
$$coloured(b, green)$$
$$coloured(c, red)$$

Colouring a graph with n colours is well known to be NP-complete [21]. The above solution can easily be adapted to solve any graph colouring problem.

2.3 MKNF

Lifschitz formulated minimal knowledge and negation as failure (MKNF) to unify several nonmonotonic logics with answer set programming [14]. This logic has its roots in autoepistemic logic; it extends first-order logic with the modal operators **K** and **not** which encode minimal knowledge and negation as failure respectively. An MKNF formula, a first-order formula that uses these operators, is evaluated against a set of "possible worlds". A formula within a **K** operator, e.g., $\mathbf{K}\,a$, holds if a is true in every possible world whereas $\mathbf{not}\,a$ holds if a is false in at least one possible world. To determine whether the knowledge in a set of possible worlds is minimal, this set is maximized by adding more possible worlds to it. After this enlargement, you have your initial set of possible worlds and an enlarged set of possible worlds. The MKNF formula is reevaluated using both of these sets of possible worlds: subformulas with a **K** operator are evaluated against the new, enlarged set of possible worlds whereas subformulas with a **not** operator are evaluated against the initial set of possible worlds. This splitting of the evaluation of **K** and **not** serves the same function as the GL-reduct described in Section 2.1. In the previous section, we gave the following answer set program:

$$a \leftarrow \mathbf{not}\,b$$
$$b \leftarrow \mathbf{not}\,a$$

One representation of the program as an MKNF formula reads as the fol-

lowing:

$$\mathbf{K}\, a \subset \mathbf{not}\, b \wedge$$
$$\mathbf{K}\, b \subset \mathbf{not}\, a$$

Where $\subset$ is the implication operator. The assignment that assigns a to true and b to false, is the set of possible worlds:

$$\{\{\text{“}a \text{ is true, } b \text{ is false”}\}, \{\text{“}a \text{ is true, } b \text{ is true”}\}\}$$

To test whether this set of possible worlds is an MKNF model (i.e., it satisfies all formulas with minimal knowledge) we enlarge the set above with the possible worlds where a is false:

$$\{\{\text{“}a \text{ is true, } b \text{ is false”}\}, \{\text{“}a \text{ is true, } b \text{ is true”}\},$$
$$\{\text{“}a \text{ is false, } b \text{ is false”}\}, \{\text{“}a \text{ is false, } b \text{ is true”}\}\}$$

We find that the first rule is not satisfied by the enlarged set of possible worlds: The formula $\mathbf{not}\, b$ holds because there is an assignment in the initial set of possible worlds where b is false and there is an assignment in the enlarged set of possible worlds where a is false, thus $\mathbf{K}\, a \subset \mathbf{not}\, b$ is not satisfied.

2.4 Hybrid MKNF

Hybrid MKNF, defined by Motik and Rosati [19], is a subset of MKNF designed to combine answer set programming with ontologies. Under this semantics, an ontology is treated as a decidable first-order formula. A Hybrid MKNF knowledge base consists of a program, a set of implications of the form shown in Section 2.3 (often called rules), and an ontology, a first-order formula nested inside a single $\mathbf{K}$ operator. A disjunctive (resp. normal) Hybrid MKNF Knowledge base contains (resp. does not contain) disjunction in the heads of rules.

Under the closed-world assumption, negation as failure is permitted. Ontologies are developed under the open-world assumption where classical negation is available and negated conclusions require proof. Hybrid MKNF

Knowledge bases allow for the utilization of closed-world and open-world reasoning in unison. As an example, suppose that a local theatre has a system that develops schedules for booking performances on their stage. This system can schedule a performance on the stage at a particular time if the stage not already been booked during that time. Here it does not make sense to require proof that the stage has not been booked, failure to find a booking for that particular time (negation as failure) is sufficient.

Under the open-world assumption, one assumes that truth is separate from knowledge and that the lack of knowledge about something does not imply falsity. Keeping with our example, after scheduling a performance on the stage, the theatre must also schedule one of their own sound engineers to work on the night of a performance if the booked company is not going to provide their own engineer. If this requirement is modeled under the open-world assumption, then the system can still rule out certain schedules or construct partial schedules even if it is not known whether a certain company will provide their own sound engineer.

2.5 Head-cycle Free

Every answer set program has a set of loop formulas associated with it. These first-order formulas encapsulate the additional complexity that disjunctive heads bring to answer set programming [12]. As such, a criterion was developed, dubbed "head-cycle free" for which if a program's loop formulas satisfy this property, then the program can be quickly converted to an equivalent normal logic program[1]. We briefly outline loop formulas as described by Linke et al. [12]. First, construct a graph where every atom in the program has a node associated with it. If an atom appears in the positive body of a rule, add edges to the graph that connect this atom to each atom in the head of the rule. A set of atoms L is a *loop* if for every pair of atoms $a, b \in L$, there exists a path of nonzero length connecting these atoms where every node along this path is in L. A program's (disjunctive) loop formulas is the first-order formulas equivalent to the following statement: "If for some loop L there is

an atom in L that is true, then there is a rule whose body does not share any atom with L and all of the true atoms in the head of this rule are are contained in L". A program is *head-cycle free*, a notion introduced by Ben-Eliyahu and Dechter [1], if, for every rule, no two atoms in the head of the rule are in a loop together.

Chapter 3

Preliminaries

Minimal knowledge and negation as failure (MKNF) [14] extends first-order logic with two modal operators, **K** and **not** , for minimal knowledge and negation as failure respectively. MKNF formulas are constructed from first-order formulas using these two modal operators for closed-world reasoning. Intuitively, $\mathbf{K}\,\psi$ asks whether ψ is known w.r.t. a collection of "possible worlds" - the larger the set, the fewer facts are known - while $\mathbf{not}\,\psi$ checks whether ψ is not known, based on negation as failure. An MKNF structure is a triple (I, M, N) where I is a first-order interpretation and M and N are sets of first-order interpretations. Operators shared with first-order logic are defined as usual. An *MKNF interpretation* M is a set of first-order interpretations ("possible worlds"). Hybrid MKNF knowledge bases rely on the standard name assumption [19]. Under this assumption, every first-order interpretation in an MKNF interpretation is required to be a Herbrand interpretation with a countably infinite number of additional constants. We refer to these constants as names. The satisfiability relation under an MKNF structure is defined as:

- $(I, M, N) \models A$ if A is true in I where A is a first-order atom
- $(I, M, N) \models \neg F$ if $(I, M, N) \not\models F$
- $(I, M, N) \models F \wedge G$ if $(I, M, N) \models F$ and $(I, M, N) \models G$
- $(I, M, N) \models \exists x, F$ if $(I, M, N) \models F[\alpha/x]$ for some name α (where $F[\alpha/x]$ is obtained by replacing every occurrence of the variable x with α)

- $(I, M, N) \models \mathbf{K}\, F$ if $(J, M, N) \models F$ for each $J \in M$
- $(I, M, N) \models \mathbf{not}\, F$ if $(J, M, N) \not\models F$ for some $J \in N$

Other symbols such as $\vee$, $\forall$, and $\supset$ are interpreted in MKNF as they are in first-order logic. An MKNF interpretation M satisfies a formula F, written $M \models_{MKNF} F$, if $(I, M, M) \models F$ for each $I \in M$. An *MKNF model* M of a formula F is an MKNF interpretation such that $M \models_{MKNF} F$ and there does not exist an MKNF interpretation $M' \supset M$ such that $(I, M', M) \models F$ for each $I \in M'$. Following Motik and Rosati [19], a *hybrid MKNF knowledge base* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ consists of a decidable description logic (DL) knowledge base $\mathcal{O}$ (typically called an ontology) which is translatable to first-order logic and a set of MKNF rules $\mathcal{P}$. We denote this translation as $\pi(\mathcal{O})$. Rules in $\mathcal{P}$ are of the form:

$$\mathbf{K}\, a_1, \ldots, \mathbf{K}\, a_k \leftarrow \mathbf{K}\, a_{k+1}, \ldots, \mathbf{K}\, a_m, \mathbf{not}\, a_{m+1}, \ldots, \mathbf{not}\, a_n$$

In the above, $a_1, \ldots, a_n$ are function-free first-order atoms of the form $p(t_1, \ldots, t_i)$ where p is a predicate and $t_1, \ldots, t_i$ are either constants or variables, with $k \geq 1$ and $m, n, i \geq 0$. Given a rule $r \in \mathcal{P}$, we define the following abbreviations:

$$\begin{aligned}
\mathit{head}(r) &= \{\mathbf{K}\, a_1, \ldots, \mathbf{K}\, a_k\}, \\
\mathit{body}^+(r) &= \{\mathbf{K}\, a_{k+1}, \ldots, \mathbf{K}\, a_m\}, \\
\mathit{body}^-(r) &= \{\mathbf{not}\, a_{m+1}, \ldots, \mathbf{not}\, a_n\}, \\
\mathbf{K}\,(\mathit{body}^-(r)) &= \{\mathbf{K}\, a \mid \mathbf{not}\, a \in \mathit{body}^-(r)\}, \text{ and} \\
\mathit{body}(r) &= (\mathit{body}^+(r), \mathbf{K}\,(\mathit{body}^-(r)))
\end{aligned}$$

A rule r in $\mathcal{P}$ is *DL-safe* if for every variable present in r, there is an occurence of that variable in the rule's positive body inside a predicate that does not occur in $\mathcal{K}$'s description logic knowledge base.

A hybrid MKNF knowledge base $\mathcal{K}$ is DL-safe if every rule in $\mathcal{P}$ is DL-safe. A knowledge base that is not DL-safe may not be decidable [19]. This constraint restricts the assignment of variables in $\mathcal{P}$ to names explictly referenced in the grounded $\mathcal{P}$. Let $\pi(\mathcal{P})$ denote rule set $\mathcal{P}$'s corresponding

MKNF formula:

$$\pi(\mathcal{P}) = \bigwedge_{r \in \mathcal{P}} \pi(r), where$$
$$\pi(r) = \forall \vec{x} \left(\bigvee_{i=1}^{k} \mathbf{K}\, a_i \subset \bigwedge_{i=k+1}^{m} \mathbf{K}\, a_i \wedge \bigwedge_{i=m+1}^{n} \mathbf{not}\, a_i \right)$$

where $\vec{x}$ is the vector of free variables found in r.

The semantics of a hybrid MKNF knowledge base $\mathcal{K}$ is obtained by applying both transformations to $\mathcal{O}$ and $\mathcal{P}$ and placing $\mathcal{O}$ within a $\mathbf{K}$ operator, i.e. $\pi(\mathcal{K}) = \pi(\mathcal{P}) \wedge \mathbf{K}\, \pi(\mathcal{O})$. We use $\mathcal{P}$, $\mathcal{O}$, and $\mathcal{K}$ in place of $\pi(\mathcal{P})$, $\pi(\mathcal{O})$, and $\pi(\mathcal{K})$ respectively when it is clear from context that the respective translated variant is intended. We refer to formulas of the form $\mathbf{K}\, a$ and $\mathbf{not}\, a$, where a is a first-order atoms, as $\mathbf{K}$-atoms and $\mathbf{not}$ -atoms respectively, and we refer to them collectively as modal-atoms. When it is clear from context, we may write a bare atom a in place of a $\mathbf{K}$-atom $\mathbf{K}\, a$. In the rest of paper, we sometimes refer to disjunctive hybrid MKNF knowledge bases simply as knowledge bases for abbreviation, or normal knowledge bases if each rule in the knowledge base has exactly one atom in the head.

line some definitions and conventions. For a hybrid MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$, we denote the set of all $\mathbf{K}$-atoms found within $\mathcal{P}$ using $\mathsf{KA}(\mathcal{K})$ where

$$\mathsf{KA}(\mathcal{K}) = \{\mathbf{K}\, a \mid \text{either } \mathbf{K}\, a \text{ or } \mathbf{not}\, a \text{ occursin the head or body of a rule in } \mathcal{P}\}$$

The *objective knowledge* of a hybrid MKNF knowledge base $\mathcal{K}$ w.r.t. a set of $\mathbf{K}$-atoms $S \subseteq \mathsf{KA}(\mathcal{K})$ is the set of first-order formulas $\{\pi(\mathcal{O})\} \cup \{a \mid \mathbf{K}\, a \in S\}$. We denote this set as $\mathsf{OB}_{\mathcal{O},S}$.

A *(partial) partition* of $\mathsf{KA}(\mathcal{K})$ is a nonoverlapping pair of subsets of $\mathsf{KA}(\mathcal{K})$ usually denoted as (T, F). $\mathbf{K}$-atoms in T are said to be true and $\mathbf{K}$-atoms in F are said to be false. A partition is *total* if $T \cup F = \mathsf{KA}(\mathcal{K})$. *A dependable partition* is a partial partition (T, F) with the additional restriction that $\mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ is consistent for each $\mathbf{K}\, b \in F$ or $\mathsf{OB}_{\mathcal{O},T}$ is consistent if F is empty. We add this partition variant for convenience and note that a partial partition that is not dependable may not be extended to an MKNF model. In

practice, a solver will include direct consequences of $\mathsf{OB}_{\mathcal{O},T}$ in T and it will only operate on dependable partitions. We denote the partition induced by the body of a rule r with $\mathit{body}(r) = (\mathit{body}^+(r), \mathbf{K}\,(\mathit{body}^-(r)))$. A rule body is applicable w.r.t. a partition (T, F) if $\mathit{body}(r) \sqsubseteq (T, F)$, i.e., if $\mathit{body}^+(r) \subseteq T$ and $\mathbf{K}\,(\mathit{body}^-(r)) \subseteq F$. We say that an MKNF interpretation M of $\mathcal{K}$ induces a partition (T, F) if

$$\bigwedge_{\mathbf{K}\,a \in T} M \models_{MKNF} \mathbf{K}\,a \wedge \bigwedge_{\mathbf{K}\,a \in F} M \models_{MKNF} \neg\mathbf{K}\,a$$

Note that the partition (T^*, F^*) induced by an MKNF model M is unique and dependable. For a partition (T, F) that is a subset of this (T^*, F^*), i.e., $(T, F) \sqsubseteq (T^*, F^*)$, we say that (T, F) can be extended to an MKNF model; such a partition is also dependable. Throughout this work and without loss of generality we assume that $\mathcal{P}$ is ground[1], i.e., it does not contain variables.

[1]In their original work, Motik and Rosati [19] show that if a knowledge base is DL-safe, a grounded version with the same MKNF models exists.

Chapter 4

Unfounded Sets

4.1 Introduction

Minimal knowledge and negation as failure (MKNF) is a modal autoepistemic logic defined by Lifschitz [14] which extends first-order logic with two modal operators **K** and **not**. It was later built upon by Motik and Rosati [19] to define hybrid MKNF knowledge bases, where rule-based MKNF formulas along with a description logic (DL) knowledge base intuitively encapsulate the combined semantics of answer set programs and ontologies. One argument for using hybrid MKNF is the existence of a proof theory based on guess-and-verify - one can enumerate partitions (a term that corresponds to interpretation in first-order logic) and for each one check whether it is an MKNF model. Such an approach is not efficient enough to be practical.

To address the above issue, Ji et al. [11] give a definition of unfounded sets and an abstract DPLL-based solver [20] for normal hybrid MKNF knowledge bases, where rules are constrained to a single atom in the head.

Disjunctive rules are a powerful extension to answer set programming that increases the expressive power of programs in the polynomial complexity hierarchy [3]. In this chapter, we extend the work of Ji et al. [11] by defining unfounded sets for disjunctive hybrid MKNF knowledge bases and we investigate the properties of such sets. This task turns out to be substantially more challenging than the normal case. We show the following main results. First, we show that the problem of determining whether an atom is unfounded w.r.t. a given (partial) partition is coNP-hard. This result is somewhat sur-

prising in that the claim holds even for normal rules under the condition that the entailment relation in the underlying DL is polynomial. This shows that the polynomial construction of the greatest unfounded set as given by Ji et al. [11] for the normal case is only an approximation. Our proof relies on an encoding that takes care of several conditions simultaneously (the hardness in the presence of non-disjunctive rules and the entailment relation assuming the DL is polynomial). Then, we formulate a polynomial operator to approximate the greatest unfounded set of disjunctive hybrid MKNF knowledge bases. Unlike the conventional definition of unfounded sets for disjunctive logic program [13], greatest unfounded sets under our definition exist unconditionally. We identify the conditions under which our approximation becomes exact for normal as well as for disjunctive hybrid MKNF knowledge bases. These conditions are also the ones under which the coNP-hardness reduces to polynomial complexity for the normal and disjunctive cases respectively, thus these results pinpoint the sources that contribute to the hardness of computing greatest unfounded sets in general. Finally, based on these results, we formulate a DPLL-based solver, where the computation of unfounded sets becomes a process of constraint propagation for search space pruning.

4.2 Unfounded Sets

First defined for normal logic programs by van Gelder et al. [22], unfounded sets encapsulate the notion that some atoms can be inferred to be false w.r.t. a partial interpretation. That is to say that given a partial partition (T, F) of $\mathsf{KA}(\mathcal{K})$, an unfounded set of a knowledge base $\mathcal{K}$ w.r.t. (T, F) is a set where every atom is false in every instance where (T, F) can be extended to a model.

A *head-cut* $R \subseteq \mathcal{P} \times \mathsf{KA}(\mathcal{K})$ is a set of rule atom pairs such that a rule $r \in \mathcal{P}$ occurs in at most one pair in R and for every pair $(r, h) \in R$ we have $\mathbf{K}\, h \in \mathit{head}(r)$. We use $\mathit{head}(R)$ to denote the set $\{h \mid (r, h) \in R\}$ where R is a head-cut.

Definition 4.2.1. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be a disjunctive hybrid MKNF knowledge base and (T, F) a partial partition of $\mathsf{KA}(\mathcal{K})$. A set X of $\boldsymbol{K}$-atoms is an*

unfounded set of $\mathcal{K}$ *w.r.t.* (T, F) *if for each* $\boldsymbol{K}$*-atom* $\mathbf{K}\,a \in X$ *and each head-cut* R *such that:*

1. $\mathit{head}(R) \cup \pi(\mathcal{O}) \models a$ *(with* $\mathcal{O}$*,* R *can derive* $\mathbf{K}\,a$*), and*
2. $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ *is consistent for each* $\mathbf{K}\,b \in F$ *and* $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T}$ *is consistent if* F *is empty (the partition* $(T \cup \mathit{head}(R), F)$ *is dependable),*

there is a pair $(r, h) \in R$ *such that at least one of the following conditions hold:*

i. $\mathit{body}^{+}(r) \cap (F \cup X) \neq \emptyset$ *(*r *positively depends on false or unfounded atoms),*

ii. $\mathbf{K}\,(\mathit{body}^{-}(r)) \cap T \neq \emptyset$ *(*r *negatively depends on true atoms), or*

iii. $\mathit{head}(r) \cap T \neq \emptyset$ *(the head of* r *is already satisfied by* T*)*

$\boldsymbol{K}$*-atoms that are found in unfounded sets are called* unfounded atoms*.*

We illustrate some general characteristics of this definition of unfounded sets with the following example.

Example 1. *Let* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ *where*

$$
\begin{aligned}
\mathcal{O} &= \{(a \supset a') \wedge (b \supset b') \wedge \neg f\} \\
\mathcal{P} &= \{\mathbf{K}\,f \leftarrow \mathbf{K}\,b; \\
&\qquad \mathbf{K}\,a \leftarrow \mathbf{not}\,b; \\
&\qquad \mathbf{K}\,a, \mathbf{K}\,b, \mathbf{K}\,c \leftarrow; \\
&\qquad \mathbf{K}\,a' \leftarrow \mathbf{K}\,a';\ \mathbf{K}\,b' \leftarrow \mathbf{K}\,b'\}
\end{aligned}
$$

Let (T, F) *be the dependable partition* $(\{\mathbf{K}\,b\}, \emptyset)$*. The* $\boldsymbol{K}$*-atom* $\mathbf{K}\,f$ *is an unfounded atom w.r.t.* (T, F) *because* $\mathbf{K}\,f$ *creates an inconsistency in* $\mathcal{O}$*.* $\mathbf{K}\,a$ *is an unfounded atom because the only way of deriving* $\mathbf{K}\,a$ *relies on* $\neg\mathbf{K}\,b$ *which contradicts* T*. The* $\boldsymbol{K}$*-atom* $\mathbf{K}\,a'$ *is unfounded because* $\mathbf{K}\,a$ *is unfounded and* $\mathbf{K}\,b'$ *is not unfounded because* $\mathsf{OB}_{\mathcal{O},T} \models b$*. Lastly,* $\mathbf{K}\,c$ *is an unfounded atom because the only rule that can derive* $\mathbf{K}\,c$ *has another head-atom (*$\mathbf{K}\,b$*) in* T*.*

A head-cut R that satisfies the first two conditions of the definition above is a set of rules that may be used in conjunction with (T, F) and $\mathcal{O}$ to derive $\mathbf{K}\, a$. A $\mathbf{K}$-atom is unfounded only if every such head-cut has a pair in it that meets one of the conditions *i* through *iii*. Note that if (T, F) is dependable, then it is impossible to derive a $\mathbf{K}$-atom found in F without violating condition 2 because an empty head-cut can be used to derive any $\mathbf{K}$-atom found in T. We demonstrate this property in the following example.

Example 2. *Let* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ *where*

$$\mathcal{O} = \{a \supset b\}, \text{ and}$$
$$\mathcal{P} = \{\mathbf{K}\, a \leftarrow \mathbf{not}\, b;\ \mathbf{K}\, b \leftarrow \mathbf{not}\, a\}$$

The dependable partition $(\{\mathbf{K}\, b\}, \{\mathbf{K}\, a\})$ *is the only total dependable partition induced by an MKNF model of* $\mathcal{K}$. *Suppose we have the dependable partition* $(T, F) = (\{\mathbf{K}\, a\}, \emptyset)$. *Neither* $\mathbf{K}\, a$ *nor* $\mathbf{K}\, b$ *is an unfounded atom w.r.t.* (T, F)*: when* $R = \emptyset$ *we have* $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$ *and* $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \models b$.

Now suppose that $(T, F) = (\emptyset, \{\mathbf{K}\, a\})$*; The* $\boldsymbol{K}$*-atom* $\mathbf{K}\, a$ *is an unfounded atom w.r.t.* (T, F). *The only head-cut that can derive* $\mathbf{K}\, a$ *is the set* $R = \{(\mathbf{K}\, a \leftarrow \mathbf{not}\, b, a)\}$, *however,* $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \cup \{\neg a\}$ *can be rewritten as* $\{a\} \cup \mathsf{OB}_{\mathcal{O},T} \cup \{\neg a\}$ *which is inconsistent.*

Under Definition 4.2.1, atoms in T cannot be unfounded if (T, F) is dependable. Note that if (T, F) is not dependable, then every set $X \subseteq \mathsf{KA}(\mathcal{K})$ is an unfounded set w.r.t. (T, F). In the following, we formally establish that no $\mathbf{K}$-atom in T can be an unfounded atom w.r.t. (T, F).

Lemma 4.2.1 (T is disjoint from any unfounded set)**.** *Let* U *be an unfounded set of a disjunctive knowledge base* $\mathcal{K}$ *w.r.t. a dependable partition* (T, F) *of* $\mathsf{KA}(\mathcal{K})$. *We have* $T \cap U = \emptyset$.

Proof. Assume for the sake of contradiction that $U \cap T \neq \emptyset$, and let $\mathbf{K}\, a \in U \cap T$. Because U is an unfounded set w.r.t. (T, F) we have for every head-cut R such that $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$, $\mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ is consistent for each $\mathbf{K}$-atom $\mathbf{K}\, b \in F$, and $\mathsf{OB}_{\mathcal{O},T}$ is consistent, that there is a pair $(r, h) \in R$

such that one of the conditions *i*, *ii*, or *iii* is satisfied. Let $R = \emptyset$. We have $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$ because $\mathbf{K}\,a \in T$. Because (T, F) is dependable, $\mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ is consistent for each $\mathbf{K}$-atom $\mathbf{K}\,b \in F$, and $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T}$ is consistent. However, there does not exist a pair $(r, h) \in R$ because R is empty, a contradiction. □

The property demonstrated in Lemma 4.2.1 is inherited from the definition of unfounded sets for normal hybrid MKNF knowledge bases [11]. This is quite different from the definition of unfounded sets for disjunctive logic programs: Leone et al. [13] refer to (partial) partitions (called *interpretations* in their context) where no atom in T is unfounded (under their own definition of unfounded sets) as *unfounded-free*. In some respects, unfounded sets under Leone et al. [13] can doubt the truth of $\mathbf{K}$-atoms in T. Since unfounded atoms are assumed to be false, an unfounded set w.r.t. (T, F) that shares $\mathbf{K}$-atoms with T is proof that (T, F) cannot be extended to a model. As shown in Lemma 4.2.1, Definition 4.2.1 lacks this property. We illustrate this difference in the following example.

Example 3. *Let* $\mathcal{K} = (\emptyset, \mathcal{P})$ *where* $\mathcal{P} = \{\mathbf{K}\,a, \mathbf{K}\,b \leftarrow\}$ *and construct the dependable partition* $(T, F) = (\{\mathbf{K}\,a, \mathbf{K}\,b\}, \emptyset)$. *Under Leone et al.'s definition, both* $\{\mathbf{K}\,a\}$ *and* $\{\mathbf{K}\,b\}$ *are unfounded sets w.r.t.* (T, F), *however, the set* $\{\mathbf{K}\,a, \mathbf{K}\,b\}$ *is not an unfounded set w.r.t.* (T, F). *Under Definition 4.2.1, none of the three aforementioned sets are unfounded sets w.r.t.* (T, F) *due to Lemma 4.2.1.*

Leone et al. show that the partial partitions that have the unfounded-free property and satisfy every rule in $\mathcal{P}$ are precisely the partial partitions that can be extended to stable models [13]. In the example above, the dependable partition $(T, F) = (\{\mathbf{K}\,a, \mathbf{K}\,b\}, \emptyset)$ cannot be extended to an MKNF model and neither $\mathbf{K}\,a$ nor $\mathbf{K}\,b$ is an unfounded atom w.r.t. (T, F). This indicates that unfounded sets under Definition 4.2.1 cannot be used to determine whether a partition can be extended to an MKNF model in the same way as Leone et al. We demonstrate that this is the case even for a normal knowledge base with an empty ontology.

Example 4. *Let $\mathcal{K} = (\emptyset, \mathcal{P})$ where $\mathcal{P} = \{\mathbf{K}\, a \leftarrow \mathbf{not}\, a\}$. Note that $\mathcal{K}$ does not have an MKNF model. The two possible total partitions are $(T_1, F_1) = (\emptyset, \{\mathbf{K}\, a\})$ and $(T_2, F_2) = (\{\mathbf{K}\, a\}, \emptyset)$. Under both Definition 4.2.1 and Leone et al.'s definition of unfounded sets, the only unfounded set w.r.t. (T_1, F_1) is $\emptyset$. Like Leone et al. , we can determine that (T_1, F_1) is not an MKNF model of $\mathcal{K}$ because there is a rule $r \in \mathcal{P}$ such that $body(r) \sqsubseteq (T_1, F_1)$ and $head(r) \cap T_1 = \emptyset$. Under Leone et al.'s definition, the set $\{\mathbf{K}\, a\}$ is an unfounded set of $\mathcal{P}$ w.r.t. (T_2, F_2), however, $\{\mathbf{K}\, a\}$ is not an unfounded set of $\mathcal{K}$ w.r.t. (T_2, F_2) under Definition 4.2.1. Critically, we cannot use Definition 4.2.1 to conclude that there is no MKNF model that induces (T_2, F_2).*

The above example demonstrates a limitation that prevents unfounded sets from being used as a mechanism for MKNF model checking. This limitation is also present in the unfounded sets defined by Ji et al. [11], however, it does not inhibit unfounded sets from being useful in a solver. Following Ji et al. [11] and Leone et al. [13], we show that unfounded sets in Definition 4.2.1 are closed under union. First we note that condition *iii* of Definition 4.2.1 ($head(r) \cap T \neq \emptyset$) does not depend on the unfounded set X like it does in Leone et al.'s definition (in this context, $(head(r) \setminus X) \cap T \neq \emptyset$). The property that all dependable partitions are unfounded-free (Lemma 4.2.1) removes the need for an additional restriction on partitions as is needed for disjunctive logic programs [13]. Applying Lemma 4.2.1, $(head(r) \setminus X) \cap T \neq \emptyset$ can be rewritten as $head(r) \cap T \neq \emptyset$. We formally demonstrate that unfounded sets by Definition 4.2.1 are closed under union and that there exists a greatest unfounded set in the following proposition.

Proposition 4.2.1 (Existence of a greatest unfounded set)**.** *Given a disjunctive hybrid MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ and a partial partition (T, F) of $\mathsf{KA}(\mathcal{K})$, there exists a greatest unfounded set $U_{\mathcal{K}}^{(T,F)}(T, F)$ such that for every unfounded set X of $\mathcal{K}$ w.r.t. (T, F) we have $U_{\mathcal{K}}^{(T,F)}(T, F) \supseteq X$.*

Proof. We show that unfounded sets are closed under union and the existence of a greatest unfounded set directly follows. Let X_a and X_b be unfounded sets of $\mathcal{K}$ w.r.t. a partial partition (T, F) of $\mathsf{KA}(\mathcal{K})$. We show that the set

$X_c = X_a \cup X_b$ is an unfounded set of $\mathcal{K}$ w.r.t. (T, F). If (T, F) is not dependable, then every set $X \subseteq \mathsf{KA}(\mathcal{K})$ is an unfounded set of $\mathcal{K}$ w.r.t. (T, F) including X_c. Assume that (T, F) is dependable and for the sake of contradiction, assume X_c is not an unfounded set. For some **K**-atom $\mathbf{K}\, a \in X_c$ we have a head-cut R s.t. conditions 1 ($\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$) and 2 ($\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ is consistent for each $\mathbf{K}\, b \in F$ or $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T}$ is consistent if F is empty) hold. In this head-cut, there is a pair (r, a) such that none of the conditions *i* ($\mathit{body}^+(r) \cap (X_c \cup F) \neq \emptyset$), *ii* ($\mathit{body}^-(r) \cap T \neq \emptyset$), or *iii* ($\mathit{head}(r) \cap T \neq \emptyset$) hold. For simplicity, assume $\mathbf{K}\, a \in X_a$ (proof is identical if $\mathbf{K}\, a \in X_b$). If $\mathit{body}^+(r) \cap (X_a \cup F) \neq \emptyset$ then we have $\mathit{body}^+(r) \cap (X_c \cup F) \neq \emptyset$ and it follows that X_c is an unfounded set. □

This property is a natural result of Lemma 4.2.1 and differs from Leone et al.'s unfounded sets, which are closed under union only if (T, F) is unfounded-free.

A solver can use any unfounded set to extend a dependable partition's false atoms without altering which models it finds. We now relate unfounded sets to MKNF models.

Proposition 4.2.2. *Let (T^*, F^*) be the partition induced by an MKNF model of a disjunctive hybrid MKNF knowledge base $\mathcal{K}$. For any dependable partition $(T, F) \sqsubseteq (T^*, F^*)$, $U_{\mathcal{K}}^{(T,F)}(T, F) \cap T^* = \emptyset$.*

Proof. Let M be the MKNF model that induces (T^*, F^*). Note that (T^*, F^*) is total and dependable. Let $(T, F) \sqsubseteq (T^*, F^*)$ and U be an unfounded set of $\mathcal{K}$ w.r.t. (T, F). We show that $U \cap T^* = \emptyset$ and it follows that $U_{\mathcal{K}}^{(T,F)}(T, F) \cap T^* = \emptyset$. Assume for the sake of contradiction that $U \cap T^* \neq \emptyset$. With $B = U \cap T^*$, let us construct an MKNF interpretation M' such that

$$M' = \{I \mid I \models \mathsf{OB}_{\mathcal{O},T} \text{ and } I \models t \text{ for each } \mathbf{K}\, t \in T^* \setminus B\ \}\}$$

The dependable partition induced by M' is $(T^* \setminus B, F^* \cup B)$. For each $b \in B$, $\mathsf{OB}_{\mathcal{O},T} \not\models b$, thus $M' \supset M$. We will derive a contradiction by showing U is not an unfounded set of $\mathcal{K}$ w.r.t. (T, F). By construction, $(I, M', M) \models_{MKNF} \pi(\mathcal{O})$

for each $I \in M'$. We now show $\forall I \in M', (I, M', M) \models_{MKNF} \pi(\mathcal{P})$, and it then follows that M is not an MKNF model which leads to a contradiction. Let $(T^* \setminus B, F)$ be the induced partition (from M' for true **K**-atoms in $T^* \setminus B$ and from M for false **K**-atoms in F). Observe that if a rule $r \in \mathcal{P}$ is not satisfied w.r.t. $(T^* \setminus B, F)$ then it must be the case that $body(r) \sqsubseteq (T^* \setminus B, F)$, $head(r) \cap T^* \neq \emptyset$, and $head(r) \cap (T^* \setminus B) = \emptyset$. That is, r is a rule whose body is satisfied by $(T^* \setminus B, F)$ but all true atoms in its head come from B, because M satisfies all rules in $\mathcal{P}$.

Let $R = \{(r, h)\}$ and b be some atom from $head(r) \cap B$. Conditions 1 and 2 of Definition 4.2.1 are met for R to test if U is an unfounded set of $\mathcal{K}$ w.r.t. (T, F). We show that none of the conditions i through iii are met by R, as such U cannot be an unfounded set w.r.t. (T, F), which leads to a contradiction. First, $body^+(r) \subseteq T^* \setminus B$ gives us $body^+(r) \cap (F \cup U) = \emptyset$ (which violates condition i). Then, from $\mathbf{K}\,(body^-(r)) \subseteq F$, we derive $\mathbf{K}\,(body^-(r)) \cap T = \emptyset$ (which violates condition ii). Finally, using $head(r) \cap T^* \subseteq B$ and $B \cap T = \emptyset$ (Lemma 4.2.1), we obtain $head(r) \cap T = \emptyset$ (which falsifies condition iii). We have shown $U \cap T^* = \emptyset$, as desired. □

We have shown that if a dependable partition (T, F) can be extended to an MKNF model, no unfounded set of $\mathcal{K}$ w.r.t. (T, F) may overlap with the true atoms in the model. It follows directly from Proposition 4.2.2 that the following analogous property holds for unfounded atoms w.r.t. (T, F).

Corollary 4.2.1. *Let (T^*, F^*) be the partition induced by an* MKNF *model M of a disjunctive hybrid MKNF knowledge base $\mathcal{K}$. Then, for any dependable partition $(T, F) \sqsubseteq (T^*, F^*)$, $M \models_{MKNF} \neg\mathbf{K}\,u$ for all $u \in U_{\mathcal{K}}^{(T,F)}(T, F)$.*

With these properties, we have shown that unfounded sets can be used to extend a partition without missing any models, i.e., if (T, F) can be extended to an MKNF model M then $(T, F \cup U)$ can be extended to the same model M for any unfounded set U w.r.t. (T, F).

4.3 Computing Unfounded Sets

Due to the inconsistencies that can arise in connection with $\mathcal{O}$, computing the greatest unfounded set w.r.t. a partial partition is intractable in general.

Example 5. *Let* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ *where* $\mathcal{O} = \neg(a \wedge b)$ *and*

$$\mathcal{P} = \{\mathbf{K}\, a \leftarrow \mathbf{not}\, b;\ \mathbf{K}\, b \leftarrow \mathbf{not}\, a;\ \mathbf{K}\, c \leftarrow \mathbf{K}\, c\}$$

Under Definition 4.2.1, $\mathbf{K}\, c$ *is an unfounded atom w.r.t.* $(\emptyset, \emptyset)$*, however, with the* $V_{\mathcal{K}}^{(\emptyset,\emptyset)}$ *operator defined by Ji et al. [11] we have* $\mathit{lfp}(V_{\mathcal{K}}^{(\emptyset,\emptyset)}) = \mathsf{KA}(\mathcal{K})$ *which misses* $\mathbf{K}\, c$ *as an unfounded atom.*[1] *It's clear that a similar operator for disjunctive knowledge bases would have the same limitation.*

In the following, we first give a formal proof of intractability and then we construct an operator for hybrid MKNF knowledge bases with disjunctive rules that adopts the same approximation technique used by Ji et al. in their $V_{\mathcal{K}}^{(T,F)}$ operator [11] for hybrid MKNF knowledge bases with normal rules.

We now show that deciding whether an atom of a normal hybrid MKNF knowledge base is unfounded is coNP-hard by comparing the head-cuts that need to be considered to determine unfoundedness with the SAT assignments that need to be considered to determine the satisfiability of a 3SAT problem.

Proposition 4.3.1. *Let* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ *be a normal hybrid MKNF knowledge base such that the entailment relation* $\mathsf{OB}_{\mathcal{O},S} \models a$ *can be checked in polynomial time for any set* $S \subseteq \mathsf{KA}(\mathcal{K})$ *and for any* $\boldsymbol{K}$*-atom* $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K})$*. Determining whether a* $\boldsymbol{K}$*-atom* $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K})$ *is an unfounded atom of* $\mathcal{K}$ *w.r.t. a dependable partition* (T, F) *of* $\mathsf{KA}(\mathcal{K})$ *is coNP-hard.*

Proof. We show that the described problem is coNP-hard. The 3SAT problem is well known to be NP-complete [21]. Let SAT be an instance of 3SAT in conjunctive normal form such that $CLAUSE = \{c_1, c_2, \ldots, c_n\}$ is the set of

[1] This is because in the least fixed point computations of the $V_{\mathcal{K}}^{(T,F)}$ operator, a default negation $\mathbf{not}\, q$ is true if $\mathbf{K}\, q$ is not known to be true, and as such, both $\mathbf{K}\, a$ and $\mathbf{K}\, b$ are derived in the first iteration which leads to inconsistency with $\mathcal{O}$.

clauses in SAT and $VAR = \{v_1, v_2, \ldots, v_n\}$ is the set of variables in SAT. Determining whether SAT is unsatisfiable is coNP-hard. We construct a normal hybrid MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ s.t.

$$
\begin{aligned}
\mathcal{O} =& \{v_i^u \oplus v_i^f \oplus v_i^t \mid \text{ for each } v_i \in VAR \text{ where } \oplus \text{ is exclusive-or } \} \cup \\
& \{(\bigwedge_{v_i \in VAR} \neg v_i^u \iff total), total \supset sat\} \cup \\
& \{clause_i \vee \neg total \mid \text{ for each clause } c_i \in CLAUSE \\
& \text{where } clause_i \text{ is a formula obtained by replacing all occurences} \\
& \text{of } v_i \text{ and } \neg v_i \text{ in } c_i \text{ with } v_i^t \text{ and } v_i^f \text{ respectively}\} \\
\mathcal{P} =& \{\mathbf{K}\, sat \leftarrow \mathbf{K}\, sat\} \cup \bigcup \{\{(\mathbf{K}\, v_i^t \leftarrow \mathbf{not}\, v_i^f), (\mathbf{K}\, v_i^f \leftarrow \mathbf{not}\, v_i^t)\} \mid v_i \in VAR\}
\end{aligned}
$$

Note that the rule $\mathbf{K}\, sat \leftarrow \mathbf{K}\, sat$ is only required to ensure that $\mathbf{K}\, sat$ is in $\mathsf{KA}(\mathcal{K})$. The time to construct the above knowledge base is linear in the number of clauses and variables in SAT. The first set of formulas in $\mathcal{O}$ requires exactly one of v_i^u, v_i^f, or v_i^t to be true. This constraint is analogous to a three-valued assignment for SAT where a variable $v_i \in VAR$ is unassigned if v_i^u is true, assigned false if v_i^f is true, and assigned true if v_i^t is true. The second set in $\mathcal{O}$ ensures that the atom $total$ is true if and only if no variable is unassigned. Finally, the third set of formulas ensure that $\pi(\mathcal{O})$ is inconsistent if the assignment is total and a clause in SAT is not satisfied. We show that (A) For any $\mathbf{K}$-atom $\mathbf{K}\, a$ and set of $\mathbf{K}$-atoms S, the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models a$ is computable in polynomial time and (B) that $\mathbf{K}\, sat$ is an unfounded atom of $\mathcal{K}$ w.r.t. $(\emptyset, \emptyset)$ if and only if SAT is unsatisfiable.

(A) We call a set of $\mathbf{K}$-atoms S total if it contains either $\mathbf{K}\, v_i^t$ or $\mathbf{K}\, v_i^f$ for each variable $v_i \in VAR$. Note that for a variable $v_i \in VAR$, the set $\mathsf{KA}(\mathcal{K})$ only contains $\mathbf{K}\, v_i^t$ and $\mathbf{K}\, v_i^f$; It does not contain $\mathbf{K}\, v_i^u$. Let $S \subseteq \mathsf{KA}(\mathcal{K})$. We show that we can, in polynomial time, determine whether $S \cup \pi(\mathcal{O})$ is consistent. We split cases where S is total and where it is not. First, assume S is not total: For some variable $v_i \in VAR$, neither $\mathbf{K}\, v_i^t$ nor $\mathbf{K}\, v_i^f$ is in S. By fixing v_i^u to be true in a consistent first-order interpretation I of $S \cup \pi(\mathcal{O})$, we ensure the atom $total$ is false. If the atom $total$ is false, we can

determine whether $\mathsf{OB}_{\mathcal{O},S}$ is consistent in polynomial time because we only need to consider the first two sets of formulas in $\mathcal{O}$. If S is total, we can, in polynomial time, verify that $S \cup \pi(\mathcal{O})$ is consistent by checking that only one of v_i^t or v_i^f is present in S and that every clause $clause_i$ is satisfied. After determining whether $S \cup \pi(\mathcal{O})$ is consistent, we can quickly check the relations $\mathsf{OB}_{\mathcal{O},S} \models v_i^t$ and $\mathsf{OB}_{\mathcal{O},S} \models v_i^f$ for any variable $v_i \in VAR$: Assuming $S \cup \pi(\mathcal{O})$ is consistent, the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models v_i^t$ (resp. $\mathsf{OB}_{\mathcal{O},S} \models v_i^f$) holds if and only if $\mathbf{K}\, v_i^t \in S$ (resp. $\mathbf{K}\, v_i^f \in S$). When $S \cup \pi(\mathcal{O})$ is consistent, the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models \mathit{total}$ holds if and only if S is total. Finally, we have $\mathsf{OB}_{\mathcal{O},S} \models \mathit{sat}$ if and only if $\mathbf{K}\, \mathit{sat} \in S$ or $\mathsf{OB}_{\mathcal{O},S} \models \mathit{total}$. If $S \cup \pi(\mathcal{O})$ is inconsistent, the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models \mathbf{K}\, a$ holds vacuously for any $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K})$.

(B) When determining whether the $\mathbf{K}$-atom $\mathbf{K}\, \mathit{sat}$ is unfounded w.r.t. $(\emptyset, \emptyset)$, we must consider each way to select a head-cut R. We show that there is a correspondence between the head-cuts that can disprove the unfoundedness of $\mathbf{K}\, \mathit{sat}$ w.r.t. $(\emptyset, \emptyset)$ and total sat assignments for SAT. Let $X = \{\mathbf{K}\, \mathit{sat}\}$ be a set that is possibly unfounded w.r.t. $(\emptyset, \emptyset)$. Observe that a larger unfounded set $X' \supset X$ w.r.t. $(\emptyset, \emptyset)$ cannot exist unless X is an unfounded set w.r.t. $(\emptyset, \emptyset)$. A head-cut R cannot be used to disprove the unfoundedness of $\mathbf{K}\, \mathit{sat}$ if either condition 1 or 2 of Definition 4.2.1 do not hold. Before creating a mapping between head-cuts and sat assignments for SAT, we exclude head-cuts that cannot be used to disprove the unfoundedness of $\mathbf{K}\, \mathit{sat}$, i.e., conditions 1 and 2 of Definition 4.2.1 are met and *i*, *ii*, and *iii* do not hold. Firstly, we exclude head-cuts that contain the pair (r, sat) because $\mathit{body}^+(r) \cap X \neq \emptyset$. We further exclude any head-cut R containing a pair of pairs (r_0, v_i^t) and (r_1, v_i^f)[2] because $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset}$ is inconsistent. Thirdly, we exclude any head-cuts that do not contain either (r_0, v_i^t) or (r_1, v_i^f) for each variable $v_i \in VAR$ noting that if such a head-cut R also meets the previous two conditions we have $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset} \not\models \mathit{sat}$ (See part (A) of this proof for details). The remaining head-cuts have a one to one correspondence with

[2]Due to the uniqueness of the second component in such a pair, there should be no confusion about which rule the first component refers to.

total assignments for SAT: if a head-cut contains a pair with v_i^t (resp. v_i^f) the corresponding assignment for SAT assigns v_i to be true (resp. false). We have for every such head-cut R that $head(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset} \models sat$ and that for every pair in $(r, h) \in R$ we have $head(r) \cap T = \emptyset$, $body^+(r) \cap (F \cup X) = \emptyset$, and $body^-(r) \cap T = \emptyset$. If $head(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset}$ is consistent, then every clause is satisfied by the corresponding sat assignment, otherwise, the inconsistency is caused by an unsatisfied clause $\neg clause_i$, thus the assignment does not satisfy SAT. If no such head-cut R exists such that $head(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset}$ is consistent, then $\mathbf{K}\, sat$ is unfounded w.r.t. $(\emptyset, \emptyset)$ and SAT is unsatisfiable. Conversely, if SAT is unsatisfiable, a head-cut R such that $head(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset}$ is consistent and $head(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset} \models sat$ does not exist, thus $\{\mathbf{K}\, sat\}$ is an unfounded set w.r.t. $(\emptyset, \emptyset)$. We have shown that deciding whether an $\mathbf{K}$-atom is unfounded is coNP-hard. □

It follows that computing the greatest unfounded set of a disjunctive hybrid MKNF knowledge base is coNP-hard. Since we are unlikely to find a way to compute $U_{\mathcal{K}}^{(T,F)}(T, F)$ in polynomial time, we are motivated to construct a polynomial operator that computes an approximation (a subset) of the greatest unfounded set. We define a family of operators $Z_{\mathcal{K}}^{(T,F)}$ where each operator induced by a dependable partition (T, F) computes an approximation of the greatest unfounded set of $\mathcal{K}$ w.r.t. (T, F)

$$\begin{aligned} Z_{\mathcal{K}}^{(T,F)}(X) = \{&\mathbf{K}\, a \mid \mathsf{OB}_{\mathcal{O},X} \models a \text{ for each } \mathbf{K}\, a \in \mathsf{KA}(\mathcal{K})\} \cup \\ &\{\mathbf{K}\, a \mid \exists r \in \mathcal{P} \text{ with } \mathbf{K}\, a \in head(r) \text{ s.t.} \\ &body^+(r) \subseteq X \ \wedge body^+(r) \cap F = \emptyset \wedge \\ &\mathbf{K}\,(body^-(r)) \cap T = \emptyset \ \wedge head(r) \cap T = \emptyset \wedge \\ &\{a, \neg b\} \cup \mathsf{OB}_{\mathcal{O},T} \text{ is consistent for each } \mathbf{K}\, b \in F\} \end{aligned}$$

This operator is the direct result of combining the $V_{\mathcal{K}}^{(T,F)}$ operator for normal hybrid MKNF knowledge bases [11] with the Φ operator for disjunctive logic programs [13]. It is easy to see that the $Z_{\mathcal{K}}^{(T,F)}$ operator is monotonic, and let us use $Atmost_{\mathcal{K}}(T, F)$ to denote its least fixed point. This operator computes a subset of $\mathsf{KA}(\mathcal{K}) \setminus U_{\mathcal{K}}^{(T,F)}(T, F)$. In particular, if $Atmost_{\mathcal{K}}(T, F) \cup$

$\pi(\mathcal{O})$ is inconsistent, we have $\mathsf{KA}(\mathcal{K}) \setminus \mathit{Atmost}_{\mathcal{K}}(T, F) = \emptyset$, a compromise to keep the operator computable in polynomial time.

To determine whether an atom is unfounded when there are disjunctive rules, we must consider an exponential number of head-cuts. The $Z_{\mathcal{K}}^{(T,F)}$ operator instead considers the heads of rules all at once and this can result in $\mathit{Atmost}_{\mathcal{K}}(T, F)$ missing some unfounded atoms even if $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent.

Example 6. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be a disjunctive hybrid MKNF knowledge base where $\mathcal{P} = \{\mathbf{K}\, a, \mathbf{K}\, b \leftarrow;\ \mathbf{K}\, c \leftarrow \mathbf{K}\, c\}$ and $\mathcal{O} = (a \wedge b) \supset c$. We have that $\{\mathbf{K}\, c\}$ is an unfounded set of $\mathcal{K}$ w.r.t. $(\emptyset, \emptyset)$. However, $\mathit{Atmost}_{\mathcal{K}}(T, F)$ is $\{a, b, c\}$ and $\mathsf{KA}(\mathcal{K}) \setminus \{a, b, c\} \neq U_{\mathcal{K}}^{(T,F)}(\emptyset, \emptyset)$.*

We intend to identify the class of knowledge bases for which the $Z_{\mathcal{K}}^{(T,F)}$ operator does not miss unfounded atoms as a result of disjunctive heads. First we define a *weak head-cut* to be a set of rule atom pairs R^w such that $R^w \subseteq \mathcal{P} \times \mathsf{KA}(\mathcal{K})$ and $h \in \mathit{head}(r)$ for each pair $(r, h) \in R$. Note that this definition is identical to the definition of head-cuts without the constraint that a rule can appear in at most one pair in R^w; within a weak head-cut, there may be two pairs (r, h_0) and (r, h_1) such that $h_0 \neq h_1$. In the following, we define a property that captures a subset of knowledge bases where $\mathit{Atmost}_{\mathcal{K}}(T, F)$ computes $U_{\mathcal{K}}^{(T,F)}(T, F)$ if $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent.

Definition 4.3.1. *A hybrid MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ is* head-independent *w.r.t. a dependable partition (T, F) if for every $\boldsymbol{K}$-atom $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K})$ and every weak head-cut R^w such that $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$, there exists a head-cut R such that $R \subseteq R^w$ and $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$.*

Head-independence means that we cannot derive atoms that we would not be able to derive using only a single atom from each rule head by using multiple atoms in the head of a rule in conjunction with the ontology. The head-independence property is violated by the knowledge base in Example 6 and it ensures that $\mathit{Atmost}_{\mathcal{K}}(T, F) \neq U_{\mathcal{K}}^{(T,F)}(T, F)$. Were we to alter the knowledge base in Example 6 such that the rule $\mathbf{K}\, a, \mathbf{K}\, b \leftarrow$ were changed to

the pair of rules $\mathbf{K}\,a \leftarrow \mathbf{not}\,b$ and $\mathbf{K}\,b \leftarrow \mathbf{not}\,a$ then $\mathcal{K}$ would have head-independence. We show formally that for a head-independent knowledge base $\mathcal{K}$, the $Z_{\mathcal{K}}^{(T,F)}$ operator computes the greatest unfounded set w.r.t. (T, F) if $Atmost_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent.

Proposition 4.3.2. *If $\mathcal{K}$ is head-independent w.r.t. a dependable partition (T, F) and $Atmost_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent, then $U_{\mathcal{K}}^{(T,F)}(T, F) = \mathsf{KA}(\mathcal{K}) \setminus Atmost_{\mathcal{K}}(T, F)$.*

Proof. First we show (1) that no $\mathbf{K}$-atom computed by $Atmost_{\mathcal{K}}(T, F)$ is unfounded w.r.t. (T, F) and then we show (2) that every atom that is not unfounded w.r.t. (T, F) is computed by $Atmost_{\mathcal{K}}(T, F)$.

(1) We first show no $\mathbf{K}$-atom in $Z_{\mathcal{K}}^{(T,F)}(\emptyset)$ is unfounded. Let $\mathbf{K}\,a \in Z_{\mathcal{K}}^{(T,F)}(\emptyset)$. Construct a weak head-cut R^w that contains a pair (r, h) for each head $\mathbf{K}$-atom $\mathbf{K}\,h \in head(r)$ and rule $r \in \mathcal{P}$ where $body^+(r) \subseteq \emptyset$, $\mathbf{K}\,(body^-(r)) \cap T = \emptyset$, and $head(r) \cap T = \emptyset$. The weak head-cut R^w contains every rule that was applied in the computation of $Z_{\mathcal{K}}^{(T,F)}(\emptyset)$. We have $head(R^w) \cup \mathsf{OB}_{\mathcal{O},T} \models a$. Applying the head-independence condition, we obtain a head-cut R such that $R \subseteq R^w$ and $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$. For every pair $(r, h) \in R$, $body^+(r) \subseteq T$, $\mathbf{K}\,(body^-(r)) \cap T = \emptyset$, and $head(r) \cap T$. The head-cut R shows that $\mathbf{K}\,a$ is not an unfounded atom w.r.t. (T, F), thus it is not a member of any unfounded set. We show that no atom computed by a successive application of $Z_{\mathcal{K}}^{(T,F)}$, e.g., $Z_{\mathcal{K}}^{(T,F)}(Z_{\mathcal{K}}^{(T,F)}(\emptyset))$, is unfounded w.r.t. (T, F). Let Z_i be result of applying the $Z_{\mathcal{K}}^{(T,F)}$ operator i times where $Z_0 = \emptyset$. We assume that no atom in Z_i is unfounded w.r.t. (T, F) and show the same for Z_{i+1}. Construct a weak head-cut R^w that contains a pair (r, h) for each head a $\mathbf{K}$-atom $\mathbf{K}\,h \in head(r)$ and rule $r \in \mathcal{P}$ where $body^+(r) \subseteq Z_i$, $body^-(r) \cap T = \emptyset$, and $head(r) \cap T = \emptyset$. Let $\mathbf{K}\,a \in Z_{\mathcal{K}}^{(T,F)}(Z_i)$. We have $head(R^w) \cup \mathsf{OB}_{\mathcal{O},T} \models a$. Applying the head-independence condition, we obtain a head-cut R such that $R \subseteq R^w$ and $head(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$. Now we have for each pair $(r, h) \in R$, $body^+(r) \subseteq Z_i$. Knowing that no $\mathbf{K}$-atom in Z_i is a member of an unfounded set, we conclude that a is not an unfounded atom w.r.t. (T, F).

(2) We show that if a $\mathbf{K}$-atom $\mathbf{K}\,a$ is not computed by $Atmost_{\mathcal{K}}(T, F)$

and it is not an unfounded atom w.r.t. (T, F), we can derive a contradiction. Let $U = \mathsf{KA}(\mathcal{K}) \setminus \mathit{Atmost}_{\mathcal{K}}(T, F)$. Let $\mathbf{K}\, a \in U$ be an $\mathbf{K}$-atom such that there exists a head-cut R where $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \models a$, $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T}$ is consistent and $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},T} \cup \{\neg b\}$ is consistent for each $\mathbf{K}\, b \in F$ and for each pair $(r, h) \in R$, $\mathit{head}(r) \cap T = \emptyset$ and $\mathit{body}^-(r) \cap T = \emptyset$. If for each pair $(r, h) \in R$ we have $\mathit{body}^+(r) \not\subseteq \mathit{Atmost}_{\mathcal{K}}(T, F)$ then U is an unfounded set w.r.t. (T, F), otherwise $\mathbf{K}\, a \in \mathit{Atmost}_{\mathcal{K}}(T, F)$. Both cases contradict the initial assumptions. □

For normal knowledge bases, i.e., where each rule contains only a single head-atom, the head-independence condition is satisfied automatically. If a knowledge base $\mathcal{K}$ is not head-independent, the $Z_{\mathcal{K}}^{(T,F)}$ operator computes a subset of $U_{\mathcal{K}}^{(T,F)}(T, F)$. Therefore, for a normal knowledge base and dependable partition (T, F) s.t. $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent, we have $U_{\mathcal{K}}^{(T,F)}(T, F) = \mathsf{KA}(\mathcal{K}) \setminus \mathit{Atmost}_{\mathcal{K}}(T, F)$. The following corollary follows directly from Proposition 4.3.2.

Corollary 4.3.1. *If a knowledge base $\mathcal{K}$ is head-independent w.r.t. a dependable partition (T, F) and $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent, then the greatest unfounded set of $\mathcal{K}$ w.r.t. (T, F) is computable in polynomial time.*

We have shown that computing the greatest unfounded set of a normal knowledge base is coNP-hard (Proposition 4.3.1). Because $\mathit{Atmost}_{\mathcal{K}}(T, F)$ can be computed in polynomial time, we conclude that the greatest unfounded set of a normal knowledge base $\mathcal{K}$ can be computed in polynomial time if $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent and the greatest unfounded set of a disjunctive knowledge base $\mathcal{K}$ can be computed in polynomial time if $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent and $\mathcal{K}$ is head-independent. Observe that for the knowledge base constructed in our proof of Proposition 4.3.1, $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is inconsistent. We formally demonstrate the intractability of computing $U_{\mathcal{K}}^{(T,F)}(T, F)$ for a disjunctive knowledge base when $\mathit{Atmost}_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent but the head-independence condition is not met.

Proposition 4.3.3. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ be a disjunctive hybrid MKNF knowledge base such that the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models a$ can be checked in polynomial time for any set $S \subseteq \mathsf{KA}(\mathcal{K})$ and for any $\boldsymbol{K}$-atom $\mathbf{K}\,a \in \mathsf{KA}(\mathcal{K})$. Let (T, F) be a dependable partition of $\mathsf{KA}(\mathcal{K})$ such that $Atmost_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent. Determining whether a $\boldsymbol{K}$-atom $\mathbf{K}\,a \in \mathsf{KA}(\mathcal{K})$ is an unfounded atom of $\mathcal{K}$ w.r.t. (T, F) is coNP-hard.*

Proof. Let SAT be an instance of 3SAT in conjunctive normal form such that $CLAUSE = \{c_1, c_2, ..., c_n\}$ is the set of clauses in SAT, and $VAR = \{v_1, v_2, ..., v_n\}$ is the set of variables in SAT. We construct a disjunctive hybrid MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ s.t.

$$
\begin{aligned}
\mathcal{O} =& \{(v_i^f \vee v_i^t) \oplus v_i^u \mid \text{ for each } v_i \in VAR \text{ where } \oplus \text{ is exclusive-or } \} \cup \\
& \{(v_i^f \wedge v_i^t) \implies sat \mid \text{ for each } v_i \in VAR\} \cup \\
& \{\left((\bigwedge_{c_i \in CLAUSE} clause_i) \wedge (\bigwedge_{v_i \in VAR} \neg v_i^u) \right) \implies sat \mid \\
& \quad \text{where } clause_i \text{ is a formula obtained by replacing all occurrences of} \\
& \quad v_i \text{ and } \neg v_i \text{ in } c_i \text{ with } v_i^t \text{ and } v_i^f \text{ respectively}\} \\
\mathcal{P} =& \{\mathbf{K}\,sat \leftarrow \mathbf{K}\,sat\} \cup \bigcup \{\{(\mathbf{K}\,v_i^t, \mathbf{K}\,v_i^f \leftarrow)\} \mid v_i \in VAR\}
\end{aligned}
$$

Let $(T, F) = (\emptyset, \emptyset)$ and observe that $Atmost_{\mathcal{K}}(T, F) \cup \pi(\mathcal{O})$ is consistent ($Atmost_{\mathcal{K}}(T, F) = \mathsf{KA}(\mathcal{K}) \setminus \{\mathbf{K}\,sat\}$). We show that (A) For any $\mathbf{K}$-atom $\mathbf{K}\,a$ and set of $\mathbf{K}$-atoms S, the entailment relation $\mathsf{OB}_{\mathcal{O},S} \models a$ is computable in polynomial time and (B) that $\mathbf{K}\,sat$ is an unfounded atom of $\mathcal{K}$ w.r.t. $(\emptyset, \emptyset)$ if and only if SAT is unsatisfiable.

(A) Observe that $\mathsf{KA}(\mathcal{K}) \cup \pi(\mathcal{O})$ is consistent, therefore, $S \cup \pi(\mathcal{O})$ is consistent for any set of $\mathbf{K}$-atoms $S \subseteq \mathsf{KA}(\mathcal{K})$. The entailment relation $\mathsf{OB}_{\mathcal{O},S} \models v_i^t$ (resp. $\mathsf{OB}_{\mathcal{O},S} \models v_i^f$) holds if and only if $v_i^t \in S$ (resp. $v_i^f \in S$). What remains to show is that $\mathsf{OB}_{\mathcal{O},S} \models sat$ can be checked in polynomial time when $\mathbf{K}\,sat \notin S$. We call a set of $\mathbf{K}$-atoms S *consistent* if it does not contain both $\mathbf{K}\,v_i^t$ and $\mathbf{K}\,v_i^f$ for every variable $v_i \in VAR$. If S is not consistent, then we have $\mathsf{OB}_{\mathcal{O},S} \models sat$ due to the second set of formulas in $\mathcal{O}$. We assume that S is consistent. We call a set of $\mathbf{K}$-atoms S *total* if it contains either $\mathbf{K}\,v_i^t$

or $\mathbf{K}\, v_i^f$ for each variable $v_i \in VAR$. We consider the cases where S is total and where S is not total. If S is not total, we can construct a consistent first-order interpretation of $S \cup \pi(\mathcal{O})$ such that v_i^u is true for some $v_i \in VAR$, thus $\mathsf{OB}_{\mathcal{O},S} \not\models \mathit{sat}$ if S is consistent and not total. Now we assume that S is total and it follows that $\bigwedge_{v_i \in VAR} \neg v_i^u$ is satisfied in the third set of formulas in $\mathcal{O}$. We refer to a model M of $S \cup \pi(\mathcal{O})$ as a *proper model* if for every $v_i \in VAR$ we have v_i^f (resp. v_i^t) to be false in M if $v_i^f \notin S$ (resp. $v_i^f \notin S$). Observe that for all models of $S \cup \pi(\mathcal{O})$ modulo proper models, *sat* is true because of the second set of formulas in $\mathcal{O}$ (recall that S is total and consistent). Note that for each proper model M we have $M \models v_i^f \oplus v_i^t$ (where $\oplus$ is exclusive-or) because S is consistent. The only case where $\mathsf{OB}_{\mathcal{O},S} \not\models \mathit{sat}$ is if we have $\mathsf{OB}_{\mathcal{O},S} \models \mathit{sat}$ if and only if S satisfies every formula clause_i. This can easily be checked in polynomial time.

(B) When determining whether the $\mathbf{K}$-atom $\mathbf{K}\, \mathit{sat}$ is unfounded w.r.t. $(\emptyset, \emptyset)$, we must consider each way to select a head-cut R. This part of the proof carries out almost identically to part 2 of our proof of Proposition 4.3.1. We only outline the key differences: Rather than relying on $\mathit{head}(R) \cup \pi(\mathcal{O})$ to be inconsistent if R does not correspond to a satisfying assignment of SAT like in our proof of Proposition 4.3.1, we rely on there being a single model of $\mathit{head}(R) \cup \pi(\mathcal{O})$ where *sat* is false (See part (A) of this proof for details on proper models). This is enough to show that $\mathit{head}(R) \cup \mathsf{OB}_{\mathcal{O},\emptyset} \not\models \mathit{sat}$. When only considering proper models of $\mathit{head}(R) \cup \pi(\mathcal{O})$, we can ignore the second set of formulas in $\mathcal{O}$ because a set of rule atom pairs R containing both (r, v_i^f) and (r, v_i^t) is not a valid head-cut. In order to determine whether a $\mathbf{K}$-atom $\mathbf{K}\, a$ is unfounded w.r.t. $(\emptyset, \emptyset)$, we must exhaustively check $\mathit{head}(R) \cup \pi(\mathcal{O})$ for every head-cut R and can conclude that SAT is unsatisfiable. If we know that SAT is unsatisfiable, there cannot exist a head-cut R, which proves that $\mathbf{K}\, a$ is not an unfounded atom. □

4.4 A DPLL-Based Solver

In this section we formulate a DPLL-based solver. First, we construct a well-founded operator $W_{\mathcal{K}}^{(T,F)}$ using the greatest unfounded set approximator from the previous section:

$$\begin{aligned} T_{\mathcal{K}}^{(T,F)}(X,Y) = \{&\mathbf{K}\,a \mid \text{ where } \mathsf{OB}_{\mathcal{O},T\cup X} \models a \text{ for some } \mathbf{K}\,a \in \mathsf{KA}(\mathcal{K})\} \cup \\ &\{\mathbf{K}\,a \mid \text{ where } \mathit{head}(r) \setminus F = \{\mathbf{K}\,a\} \text{ and} \\ &\mathit{body}(r) \sqsubseteq (T \cup X, F \cup Y) \text{ for some } r \in \mathcal{P}\}) \\ W_{\mathcal{K}}^{(T,F)}(X,Y) = &\left(T_{\mathcal{K}}^{(T,F)}(X,Y) \cup T, (\mathsf{KA}(\mathcal{K}) \setminus Z_{\mathcal{K}}^{(T,F)}(X,Y)) \cup F\right) \end{aligned}$$

We show that this operator maintains the property shown in Proposition 4.2.2.

Proposition 4.4.1. *If a dependable partition (T,F) can be extended to an MKNF model M, then the dependable partition* ***lfp*** *$(W_{\mathcal{K}}^{(T,F)})$ can also be extended to M.*

Proof. It follows from Corollary 4.2.1 that if (T,F) can be extended an MKNF model M, then $(T, F \cup Z_{\mathcal{K}}^{(T,F)}(T,F))$ can be extended to M. What's left to show is that if (T,F) can be extended to an MKNF model M, then $(T \cup T_{\mathcal{K}}^{(T,F)}(T,F)), F)$ can be extended to M. Suppose that there is some $\mathbf{K}$-atom $\mathbf{K}\,a$ in $T \cap T_{\mathcal{K}}^{(T,F)}(T,F))$ such that $M \not\models_{MKNF} \mathbf{K}\,a$. Then we either have that $\mathsf{OB}_{\mathcal{O},T} \models a$, and thus $M \not\models_{MKNF} \pi(\mathcal{O})$ or that $M \not\models_{MKNF} \mathbf{K}\,h$ for each $\mathbf{K}\,h \in \mathit{head}(r)$ and thus $M \not\models_{MKNF} \pi(\mathcal{P})$. Either case contradicts the assumption that M is an MKNF model of $\mathcal{K}$. □

Following Ji et al. [11], we construct an abstract solver in Algorithm 1 that prunes the search space for solving by using the $W_{\mathcal{K}}^{(T,F)}$ operator. The *CHECK-MODEL* procedure checks whether the MKNF interpretation

$$\{I \mid \text{where } I \models \pi(\mathcal{O}),\ I \models t \text{ for each } \mathbf{K}\,t \in T, \text{ and } I \not\models f \text{ for each } \mathbf{K}\,f \in F\}$$

is an MKNF model of $\mathcal{K}$ whenever the solver reaches a total dependable partition. This procedure is analogous to the NP-oracle required to check a model

Algorithm 1: $solver(\mathcal{K}, (T, F))$

```
1  (T, F) ← W_K(T, F) ⊔ (T, F);
2  if T ∩ F ≠ ∅ then
3      return false;
4  else if T ∪ F = KA(K) then
5      if CHECK-MODEL(T, F) then
6          return true;
7      else
8          return false;
9  else
10     choose a K-atom K a from KA(K) \ (T ∪ F);
11     if solver(K, (T ∪ {K a}, F)) then
12         return true;
13     else
14         return solver(K, (T, F ∪ {K a}));
```

of a disjunctive logic program [1]. Further developments are required for a more precise definition of this procedure.

Proposition 4.4.2. *Given a partial partition* (T, F) *of* $\mathsf{KA}(\mathcal{K})$*, the invocation of Algorithm 1* $solver(\mathcal{K}, (\emptyset, \emptyset))$ *will return true if* (T, F) *can be extended to an MKNF model of* $\mathcal{K}$*.*

Proof. It follows from Proposition 4.4.1 that the extension of (T, F) on the first line of the algorithm, $(T, F) \leftarrow W_{\mathcal{K}}(T, F) \sqcup (T, F)$, does not miss any models. A model that induces a partition (T, F) s.t. $T \cap F \neq \emptyset$ does not exist. Without the use of the $W_{\mathcal{K}}(T, F)$ operator, the solver algorithm will explore every partition $(T, F) \subseteq \mathsf{KA}(\mathcal{K}) \times \mathsf{KA}(\mathcal{K})$ where $T \cap F = \emptyset$. Thus, the usage of the $W_{\mathcal{K}}(T, F)$ operator simply prunes the search space. □

Given Proposition 4.4.2, it is easy to modify Algorithm 1 to report models instead of returning a boolean value.

We have identified some fundamental challenges in computing unfounded sets for hybrid MKNF knowledge bases that make the problem intractable.

The operator constructed by Ji et al. [11] computes a subset of the greatest unfounded set and we build on this approximation with an extension for programs with rules with disjunctive heads.

4.5 Related Work

Ji et al. establish a definition of unfounded sets for normal hybrid MKNF knowledge bases and construct well-founded operators that can be directly embedded in a solver [11]. We extend their work by introducing a definition of unfounded sets that handles disjunctive rules, rules that have multiple **K**-atoms in their heads. Our extension borrows from the unfounded-set techniques outlined by Leone et al. [13] for disjunctive logic programs but with a few noteworthy differences. Namely, our definition cannot be used directly for model-checking. If the ontology in $\mathcal{K}$ is empty, our definition is equivalent to Leon et al.'s for unfounded-free partitions. Similarly, if $\mathcal{K}$ is a normal knowledge base, our definition is equivalent to Ji et al.'s definition.

Both Ji et al. and Leone et al. outline abstract solvers for finding models of their respective languages. These solvers follow the DPLL paradigm of exploring the search space for a model. Both solvers substantially prune their search space using unfounded sets. Because the complexity of model-checking a disjunctive hybrid MKNF knowledge base is greater than that of normal hybrid MKNF knowledge bases [19], our abstract solver in this work consults a model checker once it reaches a total partition. This differs from the solver described by Ji et al. which does not rely on a model checker [11]. Leone et al.'s solver does not deepen its search on partial interpretations that assign unfounded atoms as true (partitions that cannot be extended to models) [13]. This aggressive pruning strategy requires, at each step of the solver, an invocation of an algorithm with a complexity of $\Delta_2^P[O(log\ n)]$ [13]. Industry-grade solvers, such as Clingo [6] or HEX [4], recognize the impracticality of enumerating all unfounded sets many times during the solving process and these solvers introduce approximations techniques. As a caveat of using approximations of unfounded sets, a solver may deepen its search on partial interpretations that

cannot be extended to models. Because we rely on approximations of greatest unfounded sets, we think it is reasonable for our solver to employ similar strategies used by practical solvers and include some partitions that cannot be extended to models in its search.

Both Clingo and HEX have additional support for external atoms, atoms whose truth is dependant on external sources. Clingo 5 defines $\mathbb{T}$-stable semantics [5] to reason about external atoms via external theories. HEX defines semantics for external atoms using boolean functions that take a total interpretation as input [3]. For any hybrid MKNF knowledge base, models of the accompanying ontology must be monotonic [19]. While it may be possible to encode the semantics of hybrid MKNF knowledge bases using either the HEX or Clingo extensions, neither solution exploits the monotonicity of external sources and both support nonmonotonic models of the external theories.

4.6 Conclusion

We have provided a definition of unfounded sets for disjunctive hybrid MKNF knowledge bases, studied its properties, and formulated an operator to compute a subset of the greatest unfounded set of a knowledge base. This leads to a DPLL-based solver where after each decision constraint propagation is carried out by computing additional true and false atoms on top of the current partial partition. Our methods can be directly embedded into a solver for a drastic increase in efficiency when compared to a guess-and-verify solver, the current state of art for reasoning with disjunctive hybrid MKNF knowledge bases. The addition of ontologies to answer set programs brings new challenges, namely, there is a complexity increase in computing unfounded sets even in the case of normal hybrid MKNF knowledge bases. We leave computing unfounded sets in light of inconsistencies that arise because of $\mathcal{O}$ to future work.

Chapter 5

Fixpoint Characterizations

5.1 Introduction

The operators introduced in the previous chapter extend partial partitions with atoms that exist in all models that induce the partition. A limitation of these operators is that for a dependable partition (T, F) that cannot be extended to an MKNF model, this property can only be observed if **lfp** $W_{\mathcal{K}}^{(T,F)}$ is not dependable. Critically, a total partition may be dependable when there does not exist a model that induces it; Unlike Leone et al. [13], we have no way to use unfounded sets to perform model checking. As such, additional constructions are required to show that atoms in a partition are justified.

For showing an atom is justified, we mimic the approach of loop formulas using fixpoint operators instead of a dependency graph. Using these operators, we can determine whether a partial partition can be extended to a model. While in this chapter we limit ourselves to showing that atoms in T are justified, the work in the previous section on propagating false atoms can be applied here. A major difference with our approach here and the approach of using loop formulas is that loop formulas require a dependency graph. In general, a dependency graph can not be generated for a hybrid knowledge base without knowing the internal structure of the ontology. A framework that treats the ontology as a black box has a broader range of applications than a solver that must be tuned for a particular ontology. Atom dependency analysis is crucial for both model verification and conflict generation. In this chapter, we present a framework for disjunctive hybrid MKNF knowledge bases

that utilize fixpoint operators. This framework does not rely on dependency graphs and the only restriction it imposes on the description logic is that the entailment relation can be checked in polynomial time. The framework can perform model-checking, rule out some partial partitions as potential models, and can function similarly to head-cycle free checking. Another limitation of the operators discussed in the previous chapter is that they can not extend a partial partition to a total partition if it can be extended to multiple models. The framework presented in this chapter also provides a way of exploring the search space in a meaningful way that does not miss models.

5.2 Related Work

Numerous accounts on the complexity of disjunctive ASP [3], [12], [13], [15] agree that the complexity of computing answer sets of disjunctive logic programs resides in the class Σ_2^P. Leone et al. show that answer sets can be computed by generating unfounded-free interpretations [13], while Lee and Lifschitz show that loop formulas in conjunction with a program's Clark completion can be used for model-checking [12]. Both unfounded sets and loop formulas rely heavily on syntactic dependencies between atoms: One cannot generate a set of loop formulas for a program without knowing its structure and an atom is not deemed unfounded unless it is known to be underivable. The semantics of the well-known ASP solver, Clingo [6], is defined in terms of loop formulas and in terms of unfounded sets. Due to the complexity of model checking, there is an intractable number of loop formulas; Because static dependencies between atoms are easy to establish, this complexity can be handled lazily [2]. Unlike ASP, its ontology-free counterpart, hybrid MKNF knowledge bases do not lend themselves well to dependency graph generation. If a knowledge base's ontology is left unrestricted, generating a static dependency graph for a hybrid MKNF knowledge base would require testing the ontology's entailment relation for all subsets of atoms. In the remainder of this paper, we describe a framework that establishes dependencies between atoms through fixpoint operators and thus does not require static dependency analysis.

5.3 Headcut Semantics

In this section, we formulate a framework that relies on fixpoint operators to represent MKNF models. First consider the MKNF knowledge base $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ where $\pi(\mathcal{O}) = \{(a \vee b) \supset c\}$ and $\mathcal{P}$ only contains the rule $\mathbf{K}\, a, \mathbf{K}\, b \leftarrow \mathbf{not}\, c$. This knowledge base has no MKNF models. Intuitively, we can verify that $\mathcal{K}$ does not have an MKNF model by recognizing that there is no rule to derive $\mathbf{K}\, c$ and thus some $\mathbf{K}$-atom in the head of this rule must be true in an MKNF model of $\mathcal{K}$. With either $\mathbf{K}\, a$ or $\mathbf{K}\, b$ true, an inconsistency is created if $\mathbf{K}\, c$ is false.

We generalize and formally express these intuitive semantics by considering head-cuts of $\mathcal{P}$. We define a head-cut R of $\mathcal{K}$ to be a set $R \subseteq \mathcal{P} \times \mathsf{KA}(\mathcal{K})$ where each rule $r \in \mathcal{P}$ occurs in no more than one pair $(r, h) \in R$ and $\mathbf{K}\, h \in \mathit{head}(r)$. For example, the program $\mathcal{P} = \{\mathbf{K}\, a, \mathbf{K}\, b \leftarrow\}$, $\mathcal{P}$ has exactly two head-cuts, $\{(r, a)\}$ and $\{(r, b)\}$ where r refers to the only rule in $\mathcal{P}$ (Note that we omit "$\mathbf{K}$" when describing head-cuts). Given a head-cut R, we use $\mathit{head}(R)$ and $\mathit{rule}(R)$ to denote the sets $\{h \mid (r, h) \in R\}$ and $\{r \mid (r, h) \in R\}$ respectively.

Definition 5.3.1. *For a total partition (T, F), we define $H_{\mathcal{K}}^{(T,F)}$ to be the set containing every head-cut R of $\mathcal{K}$ such that $\mathit{head}(R) \subseteq T$ and for each rule $r \in \mathcal{P}$, $r \in \mathit{rule}(R)$ if and only if $\mathit{body}(r) \sqsubseteq (T, F)$.*

Intuitively, there is a head-cut in $H_{\mathcal{K}}^{(T,F)}$ for each way of selecting a single head-atom for every satisfied rule in $\mathcal{P}$. In essence, $H_{\mathcal{K}}^{(T,F)}$, gives us a way to avoid dealing with negation or disjunction. We use this set to show atoms are justified by defining a family of operators induced by a head-cut R:

$$\begin{aligned} Q^R(X) = &\{\mathbf{K}\, h \mid \text{ where } (r, h) \in R \text{ and } \mathit{body}^+(r) \subseteq X \text{ for each } r \in \mathcal{P} \} \\ &\cup \{\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K}) \mid \mathsf{OB}_{\mathcal{O},X} \models a\} \end{aligned}$$

We denote the least fixpoint of an operator Q^R as $\mathbf{lfp}\ Q^R$ and use Q_i^R to denote applying the Q^R operator i times on the empty set (e.g., $Q_2^R = Q^R(Q^R(\emptyset))$). This operator simply takes a set of $\mathbf{K}$-atoms X and extends it with immediate consequences under R.

Revisiting the initial example where $\mathcal{K} = (\mathcal{O}, \mathcal{P})$, $\pi(\mathcal{O}) = \{(a \vee b) \supset c\}$, and $\mathcal{P} = \{\mathbf{K}\, a, \mathbf{K}\, b \leftarrow \mathbf{not}\, c\}$, consider any total dependable partition (T, F) where $\mathbf{K}\, c \in F$. Observe that for each head-cut $R \in H_{\mathcal{K}}^{(T,F)}$, we have $c \in \mathbf{lfp}\ Q^R$. For example, let $(T, F) = (\{\mathbf{K}\, a\}, \{\mathbf{K}\, c, \mathbf{K}\, b\})$. Every head-cut R in $H_{\mathcal{K}}^{(T,F)}$ contains the rule from $\mathcal{P}$, thus the Q^R operator will compute either a or b on the first iteration and then c on the second iteration. Conversely, if we consider a dependable partition (T, F) such that $\mathbf{K}\, c \in T$, then for each head-cut $R \in H_{\mathcal{K}}^{(T,F)}$, we have $c \notin \mathbf{lfp}\ Q^R$. For example, let $(T, F) = (\{\mathbf{K}\, a, \mathbf{K}\, c\}, \{\mathbf{K}\, b\})$. No rule in $\mathcal{P}$ is applicable w.r.t. (T, F), thus $H_{\mathcal{K}}^{(T,F)}$ contains a single head-cut, $\emptyset$. We have $Q_0^{\emptyset} = Q_1^{\emptyset} = \emptyset$, thus $c \notin \mathbf{lfp}\ Q^R$.

We now formally show that sets of the form $H_{\mathcal{K}}^{(T,F)}$ coincide with an MKNF models of hybrid knowledge bases. First, we connect family of sets $H_{\mathcal{K}}^{(T,F)}$ to total partitions that satisfy all rules in $\mathcal{P}$ but are not necessarily induced by an MKNF model.

Lemma 5.3.1. *For a total partition (T, F), the set $H_{\mathcal{K}}^{(T,F)}$ is empty if and only if there exists a rule $r \in \mathcal{P}$ where $body(r) \sqsubseteq (T, F)$ and $head(r) \cap T = \emptyset$.*

Proof. ($\Rightarrow$) We prove the contrapositive by constructing a head-cut R: For each rule $r \in \mathcal{P}$ where $body(r) \sqsubseteq (T, F)$, include a pair (r, h) in R where h is selected arbitrarily from $head(r) \cap T$. If $body(r) \not\sqsubseteq (T, F)$, then $r \notin rule(R)$, thus $R \in H_{\mathcal{K}}^{(T,F)}$; $H_{\mathcal{K}}^{(T,F)}$ is nonempty.

($\Leftarrow$) Let $r \in \mathcal{P}$ be a rule where $body(r) \sqsubseteq (T, F)$ and $head(r) \cap T = \emptyset$. Any head-cut $R \in H_{\mathcal{K}}^{(T,F)}$ must contain a pair with r. For R to be a head-cut, h must come from $head(r)$, and $H_{\mathcal{K}}^{(T,F)}$ requires that $h \in T$. No head-cut can satisfy both of these requirements, thus, $H_{\mathcal{K}}^{(T,F)}$ is empty. □

Definition 5.3.2. *A set of head-cuts H is a supporting set for a total dependable partition (T, F) if: An MKNF model M of $\mathcal{K}$ that induces (T, F) exists if and only if H is nonempty and for each $R \in H$ the set computed by **lfp** Q^R is precisely T.*

Proposition 5.3.1. *The set $H_{\mathcal{K}}^{(T,F)}$ is a supporting set of the dependable partition (T, F).*

Proof. Let (T, F) be a total dependable partition. We show that an MKNF model M that induces (T, F) exists if and only if $H_{\mathcal{K}}^{(T,F)}$ is nonempty and for each $R \in H_{\mathcal{K}}^{(T,F)}$, $\mathbf{lfp}\ Q^R = T$.

($\Leftarrow$) Assume $H_{\mathcal{K}}^{(T,F)}$ is nonempty and for each $R \in H_{\mathcal{K}}^{(T,F)}$, $\mathbf{lfp}\ Q^R$ is precisely T. We construct an MKNF model of $\mathcal{K}$ that induces (T, F). Let M be an MKNF interpretation containing all first-order interpretations that satisfy both $\mathcal{O}$ and T, i.e.,

$$M = \{I \mid I \models \pi(\mathcal{O})\} \cap \{I \mid I \models t \text{ for each } \mathbf{K}\, t \in T\}$$

We have $M \models_{MKNF} \neg\mathbf{K}\, a$ for each $\mathbf{K}\, a \in F$ and $M \models_{MKNF} \mathbf{K}\, a$ for each $\mathbf{K}\, a \in T$, thus M induces the dependable partition (T, F). By construction, we have $M \models_{MKNF} \mathbf{K}\, \pi(\mathcal{O})$; Applying Lemma 5.3.1 while knowing $H_{\mathcal{K}}^{(T,F)} \neq \emptyset$, we get $M \models_{MKNF} \pi(\mathcal{P})$. It remains to be shown that M is maximal, i.e., there does not exist a larger MKNF interpretation M' such that $M' \supset M$ and $(I, M', M) \models \pi(\mathcal{K})$ for each $I \in M'$.

Assume for the sake of contradiction, that such an interpretation M' exists and let (T', F') be the dependable partition induced by M' (Note that because we use M' to evaluate positive rule bodies only, we rely on the dependable partition (T', F) rather than (T', F')). We will locate a head-cut $R \in H_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^R \subset T$ and contradict the assumption that $\mathbf{lfp}\ Q^R = T$. With $M' \models_{MKNF} \pi(\mathcal{P})$, we have for each rule $r \in \mathcal{P}$ where $\mathit{body}(r) \sqsubseteq (T, F)$ (satisfied by M) if $\mathit{head}(r) \cap T' = \emptyset$, then $\mathit{body}^+(r) \subseteq T \setminus T'$. Let R be a head-cut $R \in H_{\mathcal{K}}^{(T,F)}$ where for each $(r, h) \in R$, either $h \in T'$ or $\mathit{body}^+(r) \subseteq T \setminus T'$. Clearly, such a head-cut exists. Intuitively, we only constrain the pairs whose positive rule bodies are satisfied by M'; A pair $(r, h) \in R$ containing a rule r whose positive body is not satisfied by M' may use any $h \in \mathit{head}(r) \cap T$. Suppose for the sake of contradiction that $\mathbf{lfp}\ Q^R \not\subset T$. Because $T' \subset T$, the Q^R operator has computed atoms in $T \setminus T'$ through either the ontology or a pair in R. If it was through a pair (r, h) in R, then we have $\mathit{body}^+(r) \subseteq T \setminus T'$. We must have that $\mathsf{OB}_{\mathcal{O},T'} \models t$ for some $t \in T \setminus T'$ which contradicts the assumption that $(I, M', M) \models \pi(\mathcal{O})$ for each $I \in M'$. This gives us that $\mathbf{lfp}\ Q^R \subset T$ which contradicts the assumption that $\mathbf{lfp}\ Q^R = T$. We conclude that M is

an MKNF model of $\mathcal{K}$.

($\Rightarrow$) Assume that either $H_{\mathcal{K}}^{(T,F)}$ is empty or there exists a head-cut $R \in H_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^R \neq T$. Let M be an MKNF interpretation induced by (T, F). We show that either $M \not\models_{MKNF} \pi(\mathcal{K})$ or there exists an interpretation $M' \supset M$ such that $(I, M', M) \models \pi(\mathcal{K})$ for each $I \in M'$. Clearly if $H_{\mathcal{K}}^{(T,F)} = \emptyset$, then $M \not\models_{MKNF} \pi(\mathcal{K})$ because there exists a rule $r \in \mathcal{P}$ where $M \not\models_{MKNF} \pi(r)$ (Lemma 5.3.1). Assume there exists a head-cut $R \in H_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^R \neq T$. Clearly if $\mathbf{lfp}\ Q^R \setminus T \neq \emptyset$, then T is missing $\mathbf{K}$-atoms derived from either a rule or the ontology and thus $M \not\models_{MKNF} \pi(\mathcal{K})$. Assuming $\mathbf{lfp}\ Q^R \subset T$, let T' be the set of $\mathbf{K}$-atoms that R fails to compute, i.e., $T' = T \setminus \mathbf{lfp}\ Q^R$. Construct the MKNF interpretation M' that both induces $(T', \mathsf{KA}(\mathcal{K}) \setminus T')$ and satisfies $\mathcal{O}$.

$$M' = \{I \mid I \models \pi(\mathcal{O})\} \cap \{I \mid I \models t \text{ for each } \mathbf{K}\, t \in T'\}$$

We have $M' \supset M$ and $M' \models_{MKNF} \pi(\mathcal{K})$, thus M is not an MKNF model. We conclude that if given an MKNF model M that induces a total dependable partition (T, F), then $\mathbf{lfp}\ Q^R = T$ for each $R \in H_{\mathcal{K}}^{(T,F)}$. □

In the following example, we demonstrate how the set $H_{\mathcal{K}}^{(T,F)}$ can be used to verify that a partition can be extended to a model.

Example 7. *Let $\mathcal{K} = (\emptyset, \mathcal{P})$ where $\mathcal{P}$ is defined as*

$$1 : \mathbf{K}\, a, \mathbf{K}\, b \leftarrow \mathbf{K}\, c \qquad\qquad 2 : \mathbf{K}\, b, \mathbf{K}\, c \leftarrow$$

Let $(T, F) = (\{\mathbf{K}\, b, \mathbf{K}\, c\}, \{\mathbf{K}\, a\})$. By definition, the set $H_{\mathcal{K}}^{(T,F)}$ contains two head-cuts: $R_0 = \{(1, b), (2, b)\}$ and $R_1 = \{(1, b), (2, c)\}$. When we repeatedly apply the Q operator on each of these head-cuts we obtain the following sets.

$$Q_0^{R_0} = \emptyset \qquad Q_1^{R_0} = \{b\} \qquad Q_2^{R_0} = Q_1^{R_0} \qquad Q_3^{R_0} = Q_1^{R_0}$$
$$Q_0^{R_1} = \emptyset \qquad Q_1^{R_1} = \{c\} \qquad Q_2^{R_1} = \{b, c\} \qquad Q_3^{R_1} = Q_2^{R_1}$$

*Using Proposition 5.3.1, it is easy to confirm that $(T, F) = (\{\mathbf{K}\, a, \mathbf{K}\, b, \mathbf{K}\, c\}, \emptyset)$ can not be extended to a model of $\mathcal{K}$ by observing that neither **lfp** Q^{R_0} nor Q^{R_1} computes T. If we instead use $(T, F) =$*

$(\{\mathbf{K}\, a, \mathbf{K}\, c\}, \{\mathbf{K}\, b\})$, *then the set* $H_{\mathcal{K}}^{(T,F)}$ *contains the just a single head-cut,* $R_0 = \{(1, a), (2, c)\}$. *We have* ***lfp*** $Q^{R_0} = T$, *thus* (T, F) *can be extended to an MKNF model.*

Now that we have established a semantics in terms of a family of fixpoint operators, we discuss some optimizations that, when applied to $H_{\mathcal{K}}^{(T,F)}$, greatly reduce the number of head-cuts in the set. This is a crucial step that enables the set to be used in a solver. For a dependable partition (T, F), that cannot be extended to a model, there are many head-cuts in $H_{\mathcal{K}}^{(T,F)}$ that contain rules whose bodies are never satisfied by iterative construction or do not derive anything new. Our first optimization step is to remove such rules from head-cuts by defining a new set $HM_{\mathcal{K}}^{(T,F)}$ based on $H_{\mathcal{K}}^{(T,F)}$. A head-cut R is in $HM_{\mathcal{K}}^{(T,F)}$ if and only if there is a head-cut $R' \in H_{\mathcal{K}}^{(T,F)}$ such that $R = R' \setminus RM$, where

$$RM = \{(r, h) \in R \mid \forall i, \mathit{body}(r) \subseteq Q_i^R \implies \mathit{head}(r) \subseteq Q_i^R\}$$

This set removes some pairs in each head-cut from $H_{\mathcal{K}}^{(T,F)}$. For each head-cut $R \in HM_{\mathcal{K}}^{(T,F)}$ there is some head-cut $R' \in H_{\mathcal{K}}^{(T,F)}$ for which $R \subseteq R'$. However, R' is not unique in general. The set $HM_{\mathcal{K}}^{(T,F)}$ also has the convenient property of only including pairs that contribute to the computation of Q^R, thus $\mathit{head}(R) \subseteq \mathbf{lfp}\ Q^R$.

In the example following example, we demonstrate that $HM_{\mathcal{K}}^{(T,F)}$ can be used as a supporting set.

Example 8. *Let* $\mathcal{K} = (\emptyset, \mathcal{P})$ *where* $\mathcal{P}$ *contains the following rules*

$$\begin{aligned}
0: &\quad \mathbf{K}\, a, \mathbf{K}\, b \leftarrow \\
1: &\quad \mathbf{K}\, c \leftarrow \mathbf{K}\, a \\
2: &\quad \mathbf{K}\, c \leftarrow \mathbf{K}\, b \\
3: &\quad \mathbf{K}\, a, \mathbf{K}\, d \leftarrow \mathbf{K}\, c
\end{aligned}$$

Let $(T, F) = (\mathsf{KA}(\mathcal{K}), \emptyset)$ *and* $R = \{(0, a), (1, c), (3, d)\}$. Q_1^R *computes "a" via rule* 0, *then* Q_2^R *computes "c" via rule* 2. *On the third iteration,* Q_3^R *computes*

"d" via rule 3, however, we already have head$(3) \subseteq Q_2^R$*, thus* R *is not in* $HM_{\mathcal{K}}^{(T,F)}$*. Instead, the head-cut* $R = \{(0, a), (1, c)\}$ *is in* $HM_{\mathcal{K}}^{(T,F)}$*.*

Like $H_{\mathcal{K}}^{(T,F)}$, the set $HM_{\mathcal{K}}^{(T,F)}$ is a supporting set of (T, F).

Proposition 5.3.2. $HM_{\mathcal{K}}^{(T,F)}$ *is a supporting set of* (T, F)*.*

Proof. It is trivial to show that the property demonstrated in Lemma 5.3.1 also applies to $HM_{\mathcal{K}}^{(T,F)}$, thus $HM_{\mathcal{K}}^{(T,F)} = H_{\mathcal{K}}^{(T,F)}$ if and only if either $H_{\mathcal{K}}^{(T,F)}$ or $HM_{\mathcal{K}}^{(T,F)}$ is empty. We build upon the proof in Proposition 5.3.1 by showing $\mathbf{lfp}\ Q^{R'} = T$ for each $R' \in HM_{\mathcal{K}}^{(T,F)}$ if and only if $\mathbf{lfp}\ Q^R = T$ for each $R \in H_{\mathcal{K}}^{(T,F)}$. It follows that $HM_{\mathcal{K}}^{(T,F)}$ is a supporting set of (T, F).

($\Rightarrow$) Let $R \in H_{\mathcal{K}}^{(T,F)}$ and have $R' \in HM_{\mathcal{K}}^{(T,F)}$ be the head-cut where $R' \subseteq R$. We assume $\mathbf{lfp}\ Q^{R'} = T$ and show $\mathbf{lfp}\ Q^R = T$. Apply Lemma 5.3.2 to obtain $\mathbf{lfp}\ Q^R \supseteq T$. By $\mathbf{lfp}\ Q^{R'} = T$, we have $\mathsf{OB}_{\mathcal{O},T} \not\models a$ for each $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K}) \setminus T$. We have $head(R) \subseteq T$, thus it would be absurd for a $\mathbf{K}$-atom $\mathbf{K}\, a \in \mathsf{KA}(\mathcal{K}) \setminus T$ to be in $\mathbf{lfp}\ Q^R$ because it would imply $\mathsf{OB}_{\mathcal{O},head(R)} \models a$.

($\Leftarrow$) Let $R' \in HM_{\mathcal{K}}^{(T,F)}$ and assume for each $R \in H_{\mathcal{K}}^{(T,F)}$, $\mathbf{lfp}\ Q^R = T$. We show $\mathbf{lfp}\ Q^{R'} = T$ by adding a set of rule pairs S to R' allowing $(R' \cup S)$ to be a head-cut in $H_{\mathcal{K}}^{(T,F)}$ while keeping $\mathbf{lfp}\ Q^{R' \cup S} = \mathbf{lfp}\ Q^{R'}$. For any rule $R \in H_{\mathcal{K}}^{(T,F)}$, the set $rule(R)$ is the same, i.e. $\{rule(R)\} = \bigcap\{rule(R) \mid R \in H_{\mathcal{K}}^{(T,F)}\}$. The set $rule(R) \setminus rule(R')$ contains precisely the rules that need to be added to R' via S to have $(R' \cup S) \in H_{\mathcal{K}}^{(T,F)}$. From the construction of $HM_{\mathcal{K}}^{(T,F)}$: For each rule r in $rule(R) \setminus rule(R')$, we have a $\mathbf{K}$-atom $\mathbf{K}\, h \in head(r)$ such that $h \in Q_i^{R'}$ where $i+1$ is the smallest integer such that $Q_{i+1}^{R'} \supseteq body^+(r)$. Have S be the set of pairs (r, h) where $r \in rule(R) \setminus rule(R')$ and $\mathbf{K}\, h$ is such an $\mathbf{K}$-atom from $head(r)$. It follows that $\mathbf{lfp}\ Q^{R'} = \mathbf{lfp}\ Q^{R' \cup S}$, thus $\mathbf{lfp}\ Q^{R'} = T$. □

Lemma 5.3.2. *Given two head-cuts* $R, R' \in H_{\mathcal{K}}^{(T,F)}$ *where* $R \subset R'$ *we have* $\mathbf{lfp}\ Q^R \subseteq \mathbf{lfp}\ Q^{R'}$

Proof. Assume the contrary and let i be the smallest integer such that $Q_{i+1}^R \not\subseteq Q_{i+1}^{R'}$. Let $X = Q_i^R = Q_i^{R'}$ and let a be an atom in Q_{i+1}^R but not in $Q_{i+1}^{R'}$. From $a \notin Q_{i+1}^{R'}$, we have $\mathsf{OB}_{\mathcal{O},X} \not\models a$ (otherwise $Q_{i+1}^{R'}$ would compute a). Therefore there must exist a pair $(r, a) \in R$ where $body(r) \subseteq X$ for Q_{i+1}^R to

compute a. However, this pair also exists in R' ($R \subseteq R'$), thus $a \in Q^R_{i+1}$ a contradiction. □

We define more optimizations that further reduce the number of head-cuts that need to be tested to verify a model. A head-cut R is *branch-minimal* w.r.t. a set of head-cuts S if for each $R' \in S$ such that $\mathit{head}(R) \subseteq \mathit{head}(R')$ or $\mathit{head}(R') \subseteq \mathit{head}(R)$, we have $\mathit{head}(R) \subseteq \mathit{head}(R')$. It can be easily shown that this relation is a partial order between head-cuts.

We formulate a further optimization of $HM^{(T,F)}_{\mathcal{K}}$ based on this notion of minimality.

$$HP^{(T,F)}_{\mathcal{K}} = \{R \in HM^{(T,F)}_{\mathcal{K}} \mid R \text{ is branch-minimal w.r.t. } HM^{(T,F)}_{\mathcal{K}}\}$$

Proposition 5.3.3. *$HP^{(T,F)}_{\mathcal{K}}$ is a supporting set of (T, F).*

Proof. We have $HP^{(T,F)}_{\mathcal{K}} \subseteq HM^{(T,F)}_{\mathcal{K}}$, and if $HM^{(T,F)}_{\mathcal{K}}$ is nonempty, then there exists a least head-cut w.r.t. the branch-minimal relation. Thus, $HP^{(T,F)}_{\mathcal{K}}$ is empty if and only if $HM^{(T,F)}_{\mathcal{K}}$ is empty. We show

$$(\forall R \in HM^{(T,F)}_{\mathcal{K}}, \mathbf{lfp}\ Q^R = T) \iff (\forall R' \in HP^{(T,F)}_{\mathcal{K}}, \mathbf{lfp}\ Q^{R'} = T)$$

and it follows from Proposition 5.3.2 that $HP^{(T,F)}_{\mathcal{K}}$ is a supporting set of (T, F).

($\Rightarrow$) Trivial because $HP^{(T,F)}_{\mathcal{K}} \subseteq HM^{(T,F)}_{\mathcal{K}}$ by definition.

($\Leftarrow$) Assume that for each $R' \in HP^{(T,F)}_{\mathcal{K}}$, we have $\mathbf{lfp}\ Q^{R'} = T$. Let $R \in HM^{(T,F)}_{\mathcal{K}} \setminus HP^{(T,F)}_{\mathcal{K}}$ We know $\mathbf{lfp}\ Q^R \subseteq T$ and show $\mathbf{lfp}\ Q^R \subseteq T$. Because $R \notin HP^{(T,F)}_{\mathcal{K}}$, we have an $R' \in HP^{(T,F)}_{\mathcal{K}}$ such that $\mathit{head}(R) \supset \mathit{head}(R')$. By the initial assumption, $\mathsf{OB}_{\mathcal{O},\mathit{head}(R')} \models t$ for each $\mathbf{K}\, t \in T$, thus $\mathbf{lfp}\ Q^R \supseteq \mathbf{lfp}\ Q^{R'}$ because $\mathcal{O}$ is monotonic. It follows that $\mathbf{lfp}\ Q^{R'} = T$. □

The set $HP^{(T,F)}_{\mathcal{K}}$ is not practical for use in solver: Because of the complexity of determining whether a head-cut is branch-minimal, the set cannot be efficiently enumerated. We develop a supporting set that is a compromise of $HP^{(T,F)}_{\mathcal{K}}$.

Given a head-cut R, we define $R[i]$ to be the subset of R that contains only the rules in $\mathit{rule}(R)$ whose bodies are satisfied after i iterations of the Q^R

operator, that is,

$$R[i] = \{(r, h) \in R \mid h \notin Q_i^R, body^+(r) \subseteq Q_i^R\}$$

Intuitively, $R[i]$ contains the atoms newly derived on iteration i. Likewise, we define a range $R[0..j]$ where $0 \leq j$

$$R[0..j] = \bigcup_{k=0}^{j} R[k]$$

A head-cut R is *semi-branch-minimal* w.r.t. a head-cut R' if for the largest i such that

$$R[0..(i-1)] = R'[0..(i-1)]$$

we have $head(R[i]) \subseteq head(R'[i])$.

We define $HG_{\mathcal{K}}^{(T,F)}$ to be the set

$HG_{\mathcal{K}}^{(T,F)} = \{R \in HM_{\mathcal{K}}^{(T,F)} \mid$

R is semi-branch-minimal w.r.t. every other head-cut $R' \in HM_{\mathcal{K}}^{(T,F)}\}$

We give an example of a head-cut from $HM_{\mathcal{K}}^{(T,F)}$ that is also in $HG_{\mathcal{K}}^{(T,F)}$

Example 9. *Let $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ where $\mathcal{O} = \emptyset$ and*

$$1 : \mathbf{K}\, a, \mathbf{K}\, b \leftarrow \mathbf{K}\, c$$

$$2 : \mathbf{K}\, a, \mathbf{K}\, b \leftarrow \mathbf{K}\, c$$

$$3 : \mathbf{K}\, c \leftarrow$$

Let $(T, F) = (\mathsf{KA}(\mathcal{K}), \emptyset)$. Define the following head-cuts:

$$R_0 = \{(1, a), (2, a), (3, c)\},$$

$$R_1 = \{(1, a), (2, b), (3, c)\},$$

$$R_2 = \{(1, b), (2, a), (3, c)\},$$

$$R_3 = \{(1, b), (2, b), (3, c)\}$$

We have $HM_{\mathcal{K}}^{(T,F)} = \{R_0, R_1, R_2, R_3\}$. For each pair of head-cuts R_i and R_j, we have $R_i[0..1] = R_j[0..1]$. However, we have $head(R_0[2]) \subset head(R_1[2])$ and $head(R_0[2]) \subset head(R_2[2])$, thus neither R_1 nor R_2 is semi-branch-minimal. This gives us $HG_{\mathcal{K}}^{(T,F)} = \{R_0, R_2\}$.

Both the sets $HP_{\mathcal{K}}^{(T,F)}$ and $HG_{\mathcal{K}}^{(T,F)}$ are subsets of $HM_{\mathcal{K}}^{(T,F)}$, however, in general, neither set is a subset of the other. The set $HG_{\mathcal{K}}^{(T,F)}$ is easier to enumerate than $HP_{\mathcal{K}}^{(T,F)}$ because head-cuts can be constructed iteratively whereas to enumerate head-cuts in $HP_{\mathcal{K}}^{(T,F)}$, one must test whether each head-cut in $HM_{\mathcal{K}}^{(T,F)}$ is branch-minimal. We demonstrate how $HG_{\mathcal{K}}^{(T,F)}$ may be iterated later on when we construct an abstract solver.

Example 10. *Let* $\mathcal{K} = (\mathcal{O}, \mathcal{P})$ *such that* $\mathcal{O} = \emptyset$ *and*

$$\begin{aligned}\mathcal{P} = \{&1 : \mathbf{K}\, a, \mathbf{K}\, b, \mathbf{K}\, c, \mathbf{K}\, d \leftarrow;\\ &2 : \mathbf{K}\, a, \mathbf{K}\, b, \mathbf{K}\, d \leftarrow;\\ &\mathbf{K}\, c \leftarrow a; \mathbf{K}\, c \leftarrow b;\\ &\mathbf{K}\, a \leftarrow b; \mathbf{K}\, a \leftarrow d;\\ &\mathbf{K}\, b \leftarrow a; \mathbf{K}\, b \leftarrow d; \}\end{aligned}$$

We use the total partition $(T, F) = (\mathsf{KA}(\mathcal{K}), \emptyset)$ *to consider various head-cuts in the sets* $HG_{\mathcal{K}}^{(T,F)}$ *and* $HP_{\mathcal{K}}^{(T,F)}$*. For brevity, we omit pairs in head-cuts that contain normal rules. First, we give a head-cut* R *found in* $HG_{\mathcal{K}}^{(T,F)}$ *but not* $HP_{\mathcal{K}}^{(T,F)}$*.*

$$\begin{aligned}R &= \{(1, d), (2, d)\}\\ Q_1^R &= \{d\}\\ \textbf{\textit{lfp}}\ Q^R &= \{a, b, c, d\}\end{aligned}$$

R is semi-branch-minimal w.r.t. every head-cut from $HM_{\mathcal{K}}^{(T,F)}$ *because head*(R) *contains a single atom and there is no head-cut* R' *in* $HM_{\mathcal{K}}^{(T,F)}$ *such that head*$(R') = \emptyset$*. However, the selection of* d *in* R *results in the atoms* a*,* b*, and* c*, also being derived; For the head* $R' = \{(1, a), (2, a)\}$*,* $\textbf{\textit{lfp}}\ Q^{R'} = \{a, b, c\}$*, thus* R *is not branch-minimal. We give a head-cut* R *that is in* $HP_{\mathcal{K}}^{(T,F)}$ *and not* $HG_{\mathcal{K}}^{(T,F)}$*:*

$$\begin{aligned}R &= \{(1, c), (2, b)\}\\ Q_1^R &= \{b, c\}\\ \textbf{\textit{lfp}}\ Q^R &= \{a, b, c\}\end{aligned}$$

R is branch-minimal because every head-cut in $HM_{\mathcal{K}}^{(T,F)}$ computes at least a, b, and c, however, R is not semi-branch-minimal because $c \in head(R)$; A head-cut R in $HG_{\mathcal{K}}^{(T,F)}$ cannot contain a pair with c in it because there is always a head-cut R' that is semi-branch-minimal w.r.t. R.

We give an exhaustive account of the head-cuts that are in both $HG_{\mathcal{K}}^{(T,F)}$ and $HP_{\mathcal{K}}^{(T,F)}$:

$$R_0 = \{(1, a), (2, a)\}$$
$$R_1 = \{(1, b), (2, b)\}$$
$$\textbf{\textit{lfp}}\ Q^{R_0} = \textbf{\textit{lfp}}\ Q^{R_1} = \{a, b\}$$

As mentioned above, no head-cut R in $HP_{\mathcal{K}}^{(T,F)}$ can have $d \in head(R)$ and no head-cut R in $HG_{\mathcal{K}}^{(T,F)}$ can have $c \in head(R)$, thus no head-cut in $HG_{\mathcal{K}}^{(T,F)} \cap HP_{\mathcal{K}}^{(T,F)}$ can contain either atom. Finally, we give an example of a head-cut in $HM_{\mathcal{K}}^{(T,F)}$ that is neither in $HM_{\mathcal{K}}^{(T,F)}$ nor $HG_{\mathcal{K}}^{(T,F)}$:

$$R = \{(1, a), (2, d)\}$$
$$Q_1^R = \{a, d\}$$
$$\textbf{\textit{lfp}}\ Q^R = \{a, b, c, d\}$$

The head-cut demonstrated above is neither branch-minimal nor semi-branch-minimal.

Even though $HP_{\mathcal{K}}^{(T,F)}$ and $HG_{\mathcal{K}}^{(T,F)}$ are disjoint, they are related in a crucial way that allows us to show that $HG_{\mathcal{K}}^{(T,F)}$ is a supporting set. Intuitively, a head-cut R from $HP_{\mathcal{K}}^{(T,F)}$ can be thought of as having a minimal set $head(R)$ that is globally minimal w.r.t. iterations of Q_i^R whereas a head-cut R from $HM_{\mathcal{K}}^{(T,F)}$'s set $head(R)$ is only locally minimal w.r.t. an iteration of Q_i^R. As it turns out, $HP_{\mathcal{K}}^{(T,F)}$ is more precise and if there is a head-cut in $HP_{\mathcal{K}}^{(T,F)}$ that fails to compute T, we can guarantee that such a head-cut exists in $HG_{\mathcal{K}}^{(T,F)}$ as well. We demonstrate this property formally.

Lemma 5.3.3. *If there exists a head-cut in $R \in HP_{\mathcal{K}}^{(T,F)}$ such that $\textbf{\textit{lfp}}\ Q^R \subset T$, then there is a head-cut $R' \in HP_{\mathcal{K}}^{(T,F)} \cap HG_{\mathcal{K}}^{(T,F)}$ such that $\textbf{\textit{lfp}}\ Q^{R'} \subset T$*

Proof. We demonstrate this property by describing an algorithm to convert an arbitrary head-cut R from $HP_{\mathcal{K}}^{(T,F)}$ into a head-cut R' in $HP_{\mathcal{K}}^{(T,F)} \cap HG_{\mathcal{K}}^{(T,F)}$. This new head-cut has the property that **lfp** $Q^{R'} \subseteq$ **lfp** Q^{R}. Thus, if there exists a head-cut $R \in HP_{\mathcal{K}}^{(T,F)}$ such that **lfp** $Q^R \subset T$, then we can apply this algorithm to obtain a head-cut $R' \in HP_{\mathcal{K}}^{(T,F)} \cap HG_{\mathcal{K}}^{(T,F)}$ such that **lfp** $Q^{R'} \subset T$.

Algorithm 2: hp-to-hm($R_{initial}$)

1 $R \leftarrow R_{initial}$;
2 $S_1 \leftarrow \{(r, \textbf{select } h \in \textbf{lfp } Q^{R_{initial}} \cap \mathit{head}(r)) \mid r \in \mathcal{P} \setminus \mathit{rule}(R_{initial}), \mathit{body}^+(r) \subseteq \textbf{lfp } Q^{R_{initial}}\}$;
3 $S_2 \leftarrow \{(r, \textbf{select } h \in \mathit{head}(r)) \mid r \in \mathcal{P} \setminus \mathit{rule}(R \cup S_1)\}$;
4 $R \leftarrow R \cup S_1 \cup S_2$;
5 **for** $i \leftarrow 0$; $i = 0$ *or* $Q_{i-1}^R \neq Q_i^R$; $i \leftarrow i + 1$ **do**
6 $\quad R^* \leftarrow R \setminus \{(r, h) \mid \neg\exists j, Q_j^R \supseteq \mathit{body}(r), Q_j^R \cap \mathit{head}(r) = \emptyset\}$;
7 $\quad$ **if** $\exists R' \in HG_{\mathcal{K}}^{(T,F)} \setminus HP_{\mathcal{K}}^{(T,F)}$ *s.t.* $R'[0..(i-1)] = R^*[0..(i-1)]$ *and* $\mathit{head}(R'[i]) \subset \mathit{head}(R^*[i])$ **then**
8 $\quad\quad R \leftarrow (R \setminus R^*[i]) \cup R'[i]$;
9 $R^* \leftarrow R \setminus \{(r, h) \mid \neg\exists j, Q_j^R \supseteq \mathit{body}(r), Q_j^R \cap \mathit{head}(r) = \emptyset\}$;
10 **return** R^*;

We use **select** $h \in S$ to denote that an atom h may be selected from a set S arbitrarily. We give an informal overview of the above algorithm and follow it with a formal proof. The algorithm begins by converting the head-cut R into a head-cut in $H_{\mathcal{K}}^{(T,F)}$ by adding missing rules appropriately. At the beginning of each iteration of the loop, we create a copy of R, named R^*, that has had these additions removed so that $R^* \in HG_{\mathcal{K}}^{(T,F)}$. We check whether R^* is semi-branch-minimal, and if it is not, we make it so. The body of the loop is repeated for each iteration of the Q^R operator. The end result is a head-cut $R \in HP_{\mathcal{K}}^{(T,F)} \setminus HM_{\mathcal{K}}^{(T,F)}$ such that **lfp** Q^R comptues fewer atoms than **lfp** $Q^{R_{initial}}$

We show formally that the head-cut returned by the algorithm has this property. Assume that Algorithm 5.3.3 is invoked with a head-cut $R_{initial} \in HP_{\mathcal{K}}^{(T,F)}$. Let $R^* = \{(r, h) \mid \neg\exists j, Q_j^R \supseteq \mathit{body}(r), Q_j^R \cap \mathit{head}(r) = \emptyset\}$ (the expression at line 9). The following invariants hold at the beginning of every iteration of the loop on line 5:

1. $R \in H_{\mathcal{K}}^{(T,F)}$
2. **lfp** $Q^{R^*} \subseteq$ **lfp** $Q^{R_{initial}}$
3. $R^* \in HM_{\mathcal{K}}^{(T,F)} \cap HP_{\mathcal{K}}^{(T,F)}$
4. For each j such that $0 \leq j < i$ there does not exist a head-cut $R' \in HG_{\mathcal{K}}^{(T,F)} \backslash HP_{\mathcal{K}}^{(T,F)}$ such that $R'[0..(i-1)] = R[0..(i-1)]$ and $\mathit{head}(R'[i]) \subset \mathit{head}(R[i])$ (the condition on line 7) is satisfied.

We first show that the invariants 1 through 4 hold for the first iteration of the loop. Let $R = R_{initial} \cup S_1 \cup S_2$.

(1) We have $R_{initial} \in HM_{\mathcal{K}}^{(T,F)}$, thus there is some head-cut $J \in H_{\mathcal{K}}^{(T,F)}$ such that $R = J \setminus (S_1 \cup S_2)$. Note that for every pair $(r, h) \in S_1$, there exists a head atom $h \in \mathit{head}(r) \cap$ **lfp** $Q^{R_{initial}}$ by $R_{initial} \in HM_{\mathcal{K}}^{(T,F)}$.

(2) We have **lfp** $Q^R \subseteq$ **lfp** $Q^{R_{initial}}$ because for every pair (r, h) added to $R_{initial}$ via S_1 and S_2 we have either $(r, h) \in S_2$ and $\mathit{body}^+(r) \not\subseteq$ **lfp** $Q^{R_{initial}}$ (thus $h \in$ **lfp** Q^R only if h is computed via a different pair) or $(r, h) \in S_1$ and $h \in$ **lfp** $Q^{R_{initial}}$. Because R^* is obtained by removing pairs from $R \cup S_1 \cup S_2$, we have **lfp** $Q^{R^*} \subseteq$ **lfp** $Q^{R_{initial}}$.

(3) From (1) we have $R \in H_{\mathcal{K}}^{(T,F)}$. The pairs R^* removes from R are precisely the pairs to have $R^* \in HM_{\mathcal{K}}^{(T,F)}$. From $R_{initial} \in HP_{\mathcal{K}}^{(T,F)}$ and 2, we have $R^* \in HP_{\mathcal{K}}^{(T,F)}$.

(4) We have $i = 0$, thus there does not exist an integer j, $0 \leq j < 0$.

Assuming each invariant holds at the beginning of an iteration, we show they each hold at the end of an iteration.

(1) We have $\mathit{rule}(R'[i]) = \mathit{rule}(R^*[i])$, thus line 8 does not affect the set $\mathit{rule}(R)$ and we still have $R \in H_{\mathcal{K}}^{(T,F)}$.

(2) Under the condition on line 7, we have $\mathit{head}(R'[i]) \subseteq \mathit{head}(R^*[i])$, thus line 8 can only shrink the set **lfp** Q^{R^*}.

(3) From (1) we have $R \in H_{\mathcal{K}}^{(T,F)}$. The pairs R^* removes from R are precisely the pairs to have $R^* \in HM_{\mathcal{K}}^{(T,F)}$. From $R_{initial} \in HP_{\mathcal{K}}^{(T,F)}$ and 2, we have $R^* \in HP_{\mathcal{K}}^{(T,F)}$.

(4) Because $R' \in HG_{\mathcal{K}}^{(T,F)}$, there does not exist another set $R'' \in HG$ such that $R'[0..(i-1)] = R''[0..(i-1)]$ and $head(R''[i]) \subset head(R'[i])$. It is minimal in this respect. Because line 8 only effects pairs in $R[i]$, this invariant continues to hold after i is incremented.

Finally, we return a head-cut which is in $HM_{\mathcal{K}}^{(T,F)} \cap HP_{\mathcal{K}}^{(T,F)} \cap HG_{\mathcal{K}}^{(T,F)}$ where $\mathbf{lfp}\ Q^{R^*} \subseteq \mathbf{lfp}\ Q^{R_{initial}}$. □

Finally, we can use the property demonstrated above to show that $HG_{\mathcal{K}}^{(T,F)}$ is a supporting set of (T, F).

Proposition 5.3.4. *$HG_{\mathcal{K}}^{(T,F)}$ is a supporting set of (T, F).*

Proof. $HG_{\mathcal{K}}^{(T,F)}$ is empty if and only if $HP_{\mathcal{K}}^{(T,F)}$ is empty. It is sufficient to show

$$(\forall R \in HM_{\mathcal{K}}^{(T,F)}, \mathbf{lfp}\ Q^R = T) \iff (\forall R' \in HG_{\mathcal{K}}^{(T,F)}, \mathbf{lfp}\ Q^{R'} = T)$$

. (⇒) Trivial since $HG_{\mathcal{K}}^{(T,F)} \subseteq HM_{\mathcal{K}}^{(T,F)}$.

(⇐) We show the contrapositive. Let $R \in HM_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^R \subset T$. By Proposition 5.3.3, we have a head-cut $R^p \in HP_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^{R^p} \subset T$. Apply Lemma 5.3.3 to R^p to obtain a head-cut $R' \in HG_{\mathcal{K}}^{(T,F)}$ such that $\mathbf{lfp}\ Q^{R'} \subset T$. We conclude

$$\neg(\forall R \in HM_{\mathcal{K}}^{(T,F)}, \mathbf{lfp}\ Q^{R'} = T) \implies \neg(\forall R' \in HG_{\mathcal{K}}^{(T,F)}, \mathbf{lfp}\ Q^{R'} = T)$$

.

It follows that $HG_{\mathcal{K}}^{(T,F)}$ is a supporting set of (T, F). □

5.4 An Abstract Solver

Up until now, we have dealt exclusively with total partitions. We now discuss how the techniques described in the previous section can be applied to partial partitions and in turn be used to develop an abstract solver.

We define the set $D_{\mathcal{K}}^R$ under a head-cut R.

$$D_{\mathcal{K}}^R = \{HG_{\mathcal{K}}^{(T^*,F^*)} \mid \text{ where } R = R^*[0..i]$$
$$\text{for some } i \text{ and } R^* \in HG_{\mathcal{K}}^{(T^*,F^*)} \text{ and total partition } (T^*, F^*)\}$$

Given a head-cut R, we can extract an appropriate partition from R:

$$\mathcal{S}(R) = (\textbf{lfp } Q^R, \bigcup\{body^-(r) \mid (r,h) \in R\})$$

Note that for each $HG_{\mathcal{K}}^{(T^*,F^*)} \in D_{\mathcal{K}}^R$, we have $\mathcal{S}(R) \sqsubseteq (T^*, F^*)$. Intuitively, the set $D_{\mathcal{K}}^R$ holds every possible set $HG_{\mathcal{K}}^{(T^*,F^*)}$ if $\mathcal{S}(R)$ were extended to a total partition (T^*, F^*). Note that (T^*, F^*) may not be dependable. We also define a total variant of $\mathcal{S}(R)$:

$$\mathcal{S}^*(R) = (\textbf{lfp } Q^R, \mathsf{KA}(\mathcal{K}) \setminus \textbf{lfp } Q^R)$$

We recursively define a subclass of head-cuts to limit the use of $D_{\mathcal{K}}^R$.

Definition 5.4.1. *Given, a head-cut R where $\mathcal{S}(R)$ is a dependable partition, we call R a head-cut state if either $R = \emptyset$ or there is another head-cut state R' such that either $R = R'[0..i]$ or $R \in \bigcup D_{\mathcal{K}}^{R'}$.*

Intuitively, a head-cut state is a head-cut that can be extended to some a head-cut in $R \in H$ for every $H \in D_{\mathcal{K}}^R$. Note that $\mathcal{S}(R) \sqsubseteq \mathcal{S}^*(R)$ for a head-cut state R.

We now define an abstract solver that operates on head-cut states. Supporting can assist a solver in several key ways: Conflict propagation and immediate propagation that adds positive direct consequences to a head-cut state R (See $T(R)$ in Algorithm 3), Guiding solver decisions at points where the current head-cut state cannot be extended through means of well-founded propagation (See $decisions(R)$ in Algorithm 4) and as already shown, supporting sets can be used for model verification. We join these roles together in an abstract solver outlined in Algorithm 6. Finally, we show how supporting sets can be used to reason about head-cut states that can be verified in polynomial time (similar to head-cycle free). We leave certain details unspecified such as how to enumerate the set $D_{\mathcal{K}}^R$ in an efficient way in the context of each

of the algorithms and the overall complexity of the algorithms we provide.

Algorithm 3: $T(R)$

1 $(T, F) \leftarrow \mathcal{S}(R)$;
2 $B \leftarrow \{R'[i+1] \mid R' \in \bigcup D^R_{\mathcal{K}}, R'[0..i] = R\}$;
3 **if** $\exists (r, h) \in \bigcap B,\ |\mathit{head}(r) \setminus F| \neq 1$ *or* $\mathit{body}(r) \not\sqsubseteq (T, F)$ **then**
4 $\quad$ **return** R;
5 **assert** $|B| \leq 1$;
6 **return** $R \cup \bigcap B$;

Intuitively, Algorithm 3 adds information to R only if there are only rules whose bodies are satisfied w.r.t. $\mathcal{S}(R)$ and the head of the rule contains no true atoms and only a single atom that is not false. We demonstrate that a solver cannot miss any models by applying $T(R)$ to a head-cut state.

Lemma 5.4.1. *For a head-cut state R and any head-cut $R^* \in \bigcup D^R_{\mathcal{K}}$ where $R = R^*[0..i]$, we either have $R^*[0..(i+1)] = T(R)$ or $R = T(R)$.*

Proof. We show that the assertion on line 5 holds and it directly follows that either $R = T(R)$ (when B is empty or the assert was not reached) or $T(R) = R^*[0..(i+1)]$ because $R^*[0..(i+1)]$ is the same for any R^*. Consider for the sake of contradiction that $|B| > 1$ at line 5. We have two head-cuts $R^*_1, R^*_2 \in \bigcup D^R_{\mathcal{K}}$ such that $R^*_1[0..i] = R^*_2[0..i] = R$ and $R^*_1[i+1] \neq R^*_2[i+1]$. Let $(r, h_1) \in R^*_1[i+1] \setminus R^*_2[i+1]$. If there is a pair $(r, h_2) \in R^*_2[i+1]$ such that $h_1 \neq h_2$, then $|\mathit{head}(r) \setminus F| \neq 1$, otherwise, $\mathit{body}(r) \not\sqsubseteq (T, F)$. In either case, the algorithm would have met the condition on line 3 and returned before reaching line 5, which contradictions the assumption that $|B| > 1$ at line 5. □

With $T(R)$, we only propagate information if it holds in all models that $\mathcal{S}(R)$ can be extended to. At some point in the solving process, we must add information for which this does not hold. We describe a process for extending

head-cut states with decision atoms.

Algorithm 4: *decisions*(R)

```
1 Decisions ← ∅;
2 if R ∈ ⋃ D^R_K then
3     return ∅;
4 for R′ ∈ ⋃ D^R_K where ∃i, R′[0..i] = R do
5     if S(R′[0..(i + 1)]) is dependable then
6         Decisions ← Decisions ∪ {R′[0..(i + 1)]};
7 return Decisions;
```

We guarantee very little about the atoms added by *decisions*(R) and in general, the solver will be forced to backtrack because of the atoms that were added by this procedure. However, extensions made by *decision*(R) will maintain the property that R is a head-cut state. Furthermore, if $\mathcal{S}(R)$ can be extended to an MKNF model of $\mathcal{K}$, then a head-cut in *decisions*(R) also has this property. This ensures that a solver that uses *decisions*(R) will not miss any models.

Lemma 5.4.2. *Given a head-cut state R and an MKNF model M that induces $\mathcal{S}(R)$, there is either a head-cut state $R' \in$ decisions(R) such that M induces $\mathcal{S}(R')$ or decisions(R) is empty.*

Proof. Let (T^*, F^*) be the total dependable partial induced by M. If the condition does not hold on line 2, i.e. $R \notin \bigcup D^R_{\mathcal{K}}$, then there is at least one head-cut R' that meets the condition of the loop. Because M is a model, for at least one R', $\mathcal{S}(R)$ is dependable. The set $HG^{(T^*,F^*)}_{\mathcal{K}}$ is in $D^R_{\mathcal{K}}$, and there is a head-cut $R^* \in HG^{(T^*,F^*)}_{\mathcal{K}}$ such that $R = R^*[0..i]$. Thus, the head-cut $R^*[0..(i+1)]$ is in *decisions*(R). We have $\mathcal{S}(R^*[0..(i+1)]) \sqsubseteq (T^*, F^*)$, it follows that M induces $\mathcal{S}(R^*[0..(i+1)])$. □

Definition 5.4.2. *The definition of check-model(R):*

Algorithm 5: *check-model*(R)

```
if HG_K^{S*(R)} = ∅ then
    return false;
if S*(R) is dependable and for each R ∈ HG_K^{S*(R)}, lfp Q^R = T where
  (T, F) = S*(R) then
    return S*(R);
else
    return false;
```

We define *check-model*(R) to be a procedure that simply enumerates all head-cuts in $HG_{\mathcal{K}}^{\mathcal{S}^*(R)}$ to verify that $\mathcal{S}^*(R)$ can be extended to a model by applying the Q^R until a fixed point is reached. The correctness of such an algorithm follows directly from Proposition 5.3.4.

We now integrate all preceding algorithms into an abstract solver.

Algorithm 6: *solver*(R)

```
R ← lfp T(R);
for R' ∈ decisions(R) do
    if solver(R') then
        return solver(R');
if decisions(R) = ∅ and check-model(R) then
    return S*(R)
return false;
```

While we do not specify which head-cut from *decisions*(R) should be selected to minimize backtracking, our algorithm can locate a model if one exists.

Lemma 5.4.3. *Given a head-cut state R, if there exists a model that induces $\mathcal{S}(R)$, then solver(R) will return a total partition induced by a model.*

Proof. Let (T^*, F^*) be the total dependable partition induced by M. The solver algorithm is simply repeated application of $T(R)$ and *decisions*(R). From Lemma 5.4.1 we know that $T(R)$ will preserve $\mathcal{S}(R) \sqsubseteq (T^*, F^*)$. There is either a head-cut $R' \in$ *decisions*(R) such that $\mathcal{S}(R') \sqsubseteq (T^*, F^*)$ or *decisions*(R) is empty (Lemma 5.4.2). In the case that *decisions*(R) is empty, we check whether $\mathcal{S}^*(R)$, a total partition, can be extended to a model and return it. Clearly, if this is the case then $\mathcal{S}^*(R) = (T^*, F^*)$. □

While more efficient than a guess-and-verify solver, Algorithm 6 does not efficiently verify total partitions. It is well known that head-cycle free disjunctive logic programs can be solved in NP [1]. We demonstrate an analogous subclass of disjunctive MKNF knowledge bases that can be verified in polynomial time.

First, we identify a property of head-cut states that enables polynomial verification. We show that this property holds if and only if a head-cut state coincides with an MKNF model.

Definition 5.4.3. *A head-cut state R is P-verifiable if the following holds. For every head-cut $R^* \in \bigcup D^R_{\mathcal{K}}$, where $R^*[0..i] = R[0..i]$ and $R^*[i+1] \neq R[i+1]$, we have $\mathit{lfp}\ Q^{R^*} \supseteq \mathit{head}(R[i+1])$.*

Lemma 5.4.4. *Given a dependable partition (T, F), let $R_1, R_2 \in HG^{(T,F)}_{\mathcal{K}}$ s.t. for some i, $\mathit{lfp}\ Q^{R_1} \supseteq Q^{R_2}_i$. There exists a head-cut $R_3 \in HG^{(T,F)}_{\mathcal{K}}$ such that $R_3[0..i] = R_2[0..i]$ and $\mathit{lfp}\ Q^{R_1} \supseteq \mathit{lfp}\ Q^{R_3}$.*

Proof. We construct R_3 by iteratively adding pairs its base, $R_2[0..i]$. For each $r \in \mathcal{P}$ s.t. $\mathit{body}(r) \sqsubseteq (Q^{R_3}_j, F)$ and $\mathit{head}(r) \cap Q^{R_3}_j = \emptyset$, we have $\mathit{body}(r) \sqsubseteq (\mathbf{lfp}\ Q^{R_1}, F)$ and thus some $h \in \mathit{head}(r) \cap \mathbf{lfp}\ Q^{R_1}$. We add these pairs to R_3 so that $\mathbf{lfp}\ Q^{R_1} \supseteq Q^{R_3}_{i+1}$ is maintained. Let $R[i+1] = \{(r, h) \mid \mathit{body}(r) \sqsubseteq (Q^{R_3}_j, F)$ and $h \in \mathit{head}(r) \cap \mathbf{lfp}\ Q^{R_1}\}$. [1]. This process can be repeated until there are no pairs to add and the result is a head-cut $R_3 \in HG^{(T,F)}_{\mathcal{K}}$ such that $\mathbf{lfp}\ Q^{R_3} \subseteq \mathbf{lfp}\ Q^{R_1}$. □

Proposition 5.4.1. *If a head-cut state R is P-verifiable and $\mathcal{S}^*(R)$ is dependable, then $\mathit{lfp}\ Q^R = T$ if and only if $\mathit{lfp}\ Q^{R'} = T$ for each $R' \in HG^{\mathcal{S}^*(R)}_{\mathcal{K}}$.*

Proof. Because R is a head-cut state, $HG^{(T,F)}_{\mathcal{K}}$ is nonempty. Given that $R \in HG^{\mathcal{S}^*(R)}_{\mathcal{K}}$, we need only show that $\mathbf{lfp}\ Q^R = T$ implies $\forall R^* \in HG^{\mathcal{S}^* R}_{\mathcal{K}}, \mathbf{lfp}\ Q^{R^*}$.

[1] The pairs in $R[i+1]$ must also be minimal w.r.t. $\mathit{head}(R[i+1])$ but this does not affect the proof.

We begin by defining a mapping $m(x) : y$:

$$\begin{cases} \text{the smallest positive integer } y \text{ s.t. } Q^R_{y+1} = Q^R_y & x = 0 \\ \text{the largest positive integer } y \text{ s.t. } y < m(x-1) & x > 0 \\ \text{and } \exists R^* \in HG^{(T,F)}_{\mathcal{K}}, R^*[0..y] = R[0..y], \text{ and} & \\ R^*[y+1] \neq R[y+1] & \\ 0 & \text{there does not exist such an integer } y \end{cases}$$

We show that for any $R^* \in HG^{(T,F)}_{\mathcal{K}}$, if for some i, $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i)}$, then $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i+1)}$.

Let $(r, h) \in R[m(i)+1]$. From $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i)}$, we have $\mathit{body}^+(r) \subseteq \mathbf{lfp}\ Q^{R^*}$, thus we have $\mathbf{K}\, h' \in \mathit{head}(r)$ such that either $(r, h) \in R^*$ or we have that if $Q^{R^*}_j \supseteq \mathit{body}^+(r)$, then $h' \in Q^{R^*}_j$. In either case, we have $h' \in \mathbf{lfp}\ Q^{R^*}$ and with the definition of m, we have a head-cut R' such that

$$R[0..m(i)] = R'[0..m(i)]$$
$$R[m(i)+1] \neq R'[m(i)+1]$$
$$\mathbf{lfp}\ Q^{R^*} \supseteq R'[m(i)+1]$$

(Although the definition of m does not say that R' with $\mathbf{lfp}\ Q^{R^*} \supseteq R[m(i)+1]$ exists, it is clear that it must by the definition of $HG^{(T,F)}_{\mathcal{K}}$). Because R is P-verifiable and $R[0..m(i)] = R'[0..m(i)]$, we have for each $R'' \in HG^{(T,F)}_{\mathcal{K}}, R''[m(i)+1] = R'[m(i)+1]$ that $\mathbf{lfp}\ Q^{R''} \supseteq \mathit{head}(R[m(i)+1])$. Applying Lemma 5.4.4 with R^* and R', we obtain a head-cut R'' such that $R''[m(i)+1] = R'[m(i)+1]$ and $\mathbf{lfp}\ Q^{R^*} \supseteq \mathbf{lfp}\ Q^{R''}$. Thus, we have $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i)+1}$.

For each j s.t. $m(i-1)+1 < j < m(i)+1$, we have $\mathbf{lfp}\ Q^{R^*} \supseteq \mathit{head}(R[j])$. Note that if this were not the case, there would be multiple true atoms in the head of some $r \in \mathit{rule}(R[j])$ and thus $m(i-1)$ would be equal to $j+1$.

We have shown that for any $R^* \in HG^{(T,F)}_{\mathcal{K}}$, if for some i, $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i)}$, then $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i+1)}$. We now show inductively that $\mathbf{lfp}\ Q^{R^*} \supseteq T$. Let e be the integer such that $m(e-1) \neq 0$ and $m(e) = 0$. Clearly, $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(e)}$, since $Q^R_{m(e)} = \emptyset$. Assume for some k, $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(i)}$, we have $\mathbf{lfp}\ Q^{R^*} \supseteq Q^R_{m(k+1)}$ (shown above).

By the initial assumption, we have $Q^R_{m(0)} = T$, thus $\mathbf{lfp}\ Q^{R^*} \subseteq T$. We can not have $\mathbf{lfp}\ Q^{R^*} \subset T$, thus $\mathbf{lfp}\ Q^{R^*} = T$ □

Corollary 5.4.1. *A head-cut state R is P-verifiable if and only if $\mathcal{S}^*(R)$ can be extended to an MKNF model of $\mathcal{K}$.*

Proof. ($\Rightarrow$) See Proposition 5.4.1. ($\Leftarrow$) Under the definition of head-cut states, we have that $HG_{\mathcal{K}}^{\mathcal{S}^*(R)}$ is nonempty. If R were not P-verifiable, then we have a head-cut $R^* \in HG_{\mathcal{K}}^{\mathcal{S}^*(R)}$ such that **lfp** $Q^{R^*} \neq T$ where $(T, F) = \mathcal{S}^*(R)$. With Proposition 5.3.4, $\mathcal{S}^*(R)$ can not be extended to an MKNF model of $\mathcal{K}$. □

Using the above property, we construct a verification algorithm that is more efficient than *check-model*(R).

Algorithm 7: *check-model*$_2(R)$

```
1 if HG_K^{S*(R)} = ∅ then
2     return false;
3 for i ← 0; Q_i^R ≠ Q_{i-1}^R; i ← i + 1 do
4     for R' ∈ HG_K^{S*(R)} where R[0..i] = R'[0..i] and R[i+1] ≠ R'[i+1]
        do
5         if head(R[i+1]) ⊈ lfp Q^{R'} then
6             return false;
7 return true;
```

When *check-model*(R) is replaced with Algorithm 7 in the solver (Algorithm 6), P-verifiable head-cut states can be quickly verified. We feel strongly that with further complexity analysis we will be able conclude that our abstract solver algorithm, when used with an empty ontology and head-cycle free disjunctive logic program, can verify any enumerated partition in polynomial time.

5.5 Conclusion

We have provided a new way of characterizing disjunctive MKNF models through supporting sets. The largest set we defined, $H_{\mathcal{K}}^{(T,F)}$, contains many redundant head-cuts and is not practical for use in a solver. We defined a smaller set, $HM_{\mathcal{K}}^{(T,F)}$, where each head-cut in $HM_{\mathcal{K}}^{(T,F)}$ is limited to rules that contribute to the fixpoint computation. Next we refined $HM_{\mathcal{K}}^{(T,F)}$ further to obtain $HP_{\mathcal{K}}^{(T,F)}$ and the less precise but more tractable set $HG_{\mathcal{K}}^{(T,F)}$. We pro-

vided an abstract solver that utilizes supporting sets to enumerate partitions and to verify models. Finally, we characterized P-verifiable head-cut states, a property of head-cut states that is comparable to head-cycle free disjunctive logic programs, and we give a more efficient model verification procedure that leverages this property.

We speculate that the complexity of our abstract solver algorithm is no worse than a guess-and-verify solver if the entailment relation of the accompanying ontology can be computed in polynomial time. We also speculate that if the ontology is empty and $\mathcal{P}$ is a head-cycle free disjunctive logic program that the complexity of finding a model lies in NP. However, we leave a full analysis of the complexity of our algorithm and the complexity of recognizing P-verifiable head-cut states to future work. In this work, we introduce many new structures to characterize our semantics; It would be interesting to recast this work using more familiar fixpoint structures such as approximators in approximation fixpoint theory [18]. In the future, we would also like to leverage this framework to generate conflicts so that a CDNL-based solver may be constructed.

Chapter 6

Conclusion

The pairing of open-world reasoning with closed-world reasoning is highly sought after in the area of knowledge representation and reasoning. Hybrid MKNF matches the two very elegantly but solvers that adopt modern techniques, such as CDNL, have yet to be developed.

In this work, we defined unfounded sets for disjunctive hybrid MKNF knowledge bases.. We established a definition of unfounded sets that joins the definition from Ji et al. [11] with Leone et al.'s definition [13]. After identifying that the definitions are incompatible in certain respects, we opted for our definition to remain more faithful to Ji et al's definition. We obtained similar properties as Ji et al., while compromising some of the properties exhibited by Leone et al's definition. Namely, our definition of unfounded sets cannot be applied to arbitary partitions in order to determine whether they can be extended to an MKNF model. We have proven upper complexity bounds for computing our unfounded sets and have proven a new upper bound for computing Ji et al.'s unfounded sets. Finally, we constructed well-founded operators based on this definition and showed how these operators may be incorporating into the solving process.

We give a fixpoint characterization for disjunctive hybrid MKNF knowledge bases that can be applied to determine whether arbitrary partial partitions can be extended to MKNF models. This directly tackles the limitations of our definition of unfounded sets and also tackles the issue of a lack of syntactic dependency between atoms for an ontology. We provide several optimizations

to our construction that can reduce the search space required in the solving process. We also show that from within our characterizations we can identify a class of solver states that is analogous to head-cycle free disjunctive logic programs. Finally, we the role our constructions can play in a solver by constructing an abstract solver that uses them.

One of the shortcomings of our definition of unfounded sets is that atoms in T in a dependable partition cannot be unfounded atoms. As a result, unfounded sets cannot be used to identify unjustified atoms in a dependable partition. Our fixpoint developments remedy this by providing an alternative model-checking approach to unfounded-free checking and can identify some unjustified atoms for partial partitions.

The theoretical foundation we've provided in the thesis solves some of the issues that must be addressed before a more efficient disjunctive hybrid MKNF knowledge base solver can be constructed. With it, we are closer to building more efficient solvers for disjunctive hybrid MKNFknowledge bases.

References

[1] R. Ben-Eliyahu and R. Dechter, "Propositional semantics for disjunctive logic programs," *Annals of Mathematics and Artificial Intelligence*, vol. 12, no. 1, pp. 53–87, Mar. 1994.

[2] C. Drescher and T. Walsh, "Answer set solving with lazy nogood generation," in *Technical Communications of the 28th International Conference on Logic Programming, ICLP 2012, September 4-8, 2012, Budapest, Hungary*, A. Dovier and V. S. Costa, Eds., ser. LIPIcs, vol. 17, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012, pp. 188–200.

[3] T. Eiter and G. Gottlob, "On the computational cost of disjunctive logic programming: Propositional case," *Annals of Mathematics and Artificial Intelligence*, vol. 15, no. 3, pp. 289–323, Sep. 1995.

[4] T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits, "A uniform integration of higher-order reasoning and external evaluations in answer-set programming," p. 7, 2005.

[5] M. Gebser, R. Kaminski, B. Kaufmann, M. Ostrowski, T. Schaub, and P. Wanko, "Theory solving made easy with clingo 5," in *Technical Communications of the 32nd International Conference on Logic Programming, ICLP 2016 TCs, October 16-21, 2016, New York City, USA*, M. Carro, A. King, N. Saeedloei, and M. D. Vos, Eds., ser. OASICS, vol. 52, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, 2:1–2:15.

[6] M. Gebser, B. Kaufmann, and T. Schaub, "Conflict-driven answer set solving: From theory to practice," *Artificial Intelligence*, vol. 187-188, pp. 52–89, Aug. 2012.

[7] ——, "Advanced conflict-driven disjunctive answer set solving," in *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, F. Rossi, Ed., IJCAI/AAAI, 2013, pp. 912–918.

[8] M. Gelfond and V. Lifschitz, "The stable model semantics for logic programming," in *Proceedings of the Fifth International Conference and Symposium*, R. A. Kowalski and K. A. Bowen, Eds., MIT Press, 1988, pp. 1070–1080.

[9] ——, "Classical negation in logic programs and disjunctive databases," *New Gener. Comput.*, vol. 9, no. 3/4, pp. 365–386, 1991.

[10] ——, "The stable model semantics for logic programming," *Logic Programming*, vol. 2, Dec. 14, 2000.

[11] J. Ji, F. Liu, and J.-H. You, "Well-founded operators for normal hybrid MKNF knowledge bases," *Theory Pract. Log. Program.*, vol. 17, no. 5-6, pp. 889–905, 2017.

[12] J. Lee and V. Lifschitz, "Loop formulas for disjunctive logic programs," in *Logic Programming, 19th International Conference, ICLP 2003, Mumbai, India, December 9-13, 2003, Proceedings*, C. Palamidessi, Ed., ser. Lecture Notes in Computer Science, vol. 2916, Springer, 2003, pp. 451–465.

[13] N. Leone, P. Rullo, and F. Scarcello, "Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation," *Information and Computation*, vol. 135, no. 2, pp. 69–112, Jun. 1997.

[14] V. Lifschitz, "Nonmonotonic databases and epistemic queries," in *Proceedings of the 12th International Joint Conference on Artificial Intelligence. Sydney, Australia, August 24-30, 1991*, J. Mylopoulos and R. Reiter, Eds., Morgan Kaufmann, 1991, pp. 381–386.

[15] V. Lifschitz and A. A. Razborov, "Why are there so many loop formulas?" *ACM Trans. Comput. Log.*, vol. 7, no. 2, pp. 261–268, 2006.

[16] T. Linke, H. Tompits, and S. Woltran, "On acyclic and head-cycle free nested logic programs," in *Logic Programming*, B. Demoen and V. Lifschitz, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 225–239.

[17] F. Liu and J. You, "Three-valued semantics for hybrid MKNF knowledge bases revisited," *Artif. Intell.*, vol. 252, pp. 123–138, 2017.

[18] ——, "Alternating fixpoint operator for hybrid MKNF knowledge bases as an approximator of AFT," in *Rules and Reasoning - Third International Joint Conference, RuleML+RR 2019, Bolzano, Italy, September 16-19, 2019, Proceedings*, P. Fodor, M. Montali, D. Calvanese, and D. Roman, Eds., ser. Lecture Notes in Computer Science, vol. 11784, Springer, 2019, pp. 113–127.

[19] B. Motik and R. Rosati, "Reconciling description logics and rules," *Journal of the ACM*, vol. 57, no. 5, pp. 1–62, Jun. 1, 2010.

[20] R. Nieuwenhuis, A. Oliveras, and C. Tinelli, "Solving SAT and SAT Modulo Theories: From an abstract davis-putnam-logemann-loveland procedure to DPLL(T)," *Journal of the ACM*, vol. 53, no. 6, pp. 937–977, 2006.

[21] M. Sipser, *Introduction to the Theory of Computation*, First edition. International Thomson Publishing, 1996.

[22] A. Van Gelder, K. A. Ross, and J. S. Schlipf, "The well-founded semantics for general logic programs," *Journal of the ACM*, vol. 38, no. 3, pp. 619–649, Jul. 1991.