UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Project Final Paper 2022

Title: Model-Based Defeasible Reasoning

Author: Jaron Cohen

Project Abbreviation: MBDR

Supervisor(s): Professor Tommie Meyer

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	20
Experiment Design and Execution	*0*	*20*	4
System Development and Implementation	*0*	*20*	4
Results, Findings and Conclusions	*10*	*20*	17
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	
Total marks		**80**	

Model-Based Defeasible Reasoning

Jaron Cohen
University of Cape Town
Cape Town, South Africa
CHNJAR003@myuct.ac.za

ABSTRACT

Well-known forms of KLM-style defeasible entailment can be defined syntactically, via formula-based manipulations, and semantically, using ranked models. While entailment algorithms based on such syntactic characterisations have been developed, algorithms that directly manipulate the underlying models have not been explored. We present and analyse several algorithms based on ranked model semantics for computing a core form of rational defeasible entailment: rational closure. For each algorithm, we define an abstract representation of the ranked model, an algorithm for its construction, and a suitable adaptation of existing entailment algorithms compatible with the representation. We develop a software toolkit for their implementation and compare the run-time performance of these new approaches to their theoretical analysis and existing formula-based counterparts.

CCS CONCEPTS

• **Theory of computation → Automated reasoning**; • **Computing methodologies → Nonmonotonic, default reasoning and belief revision**.

KEYWORDS

Artificial Intelligence, Knowledge Representation and Reasoning, Defeasible Reasoning, Propositional Logic, Rational Closure, Lexicographic Closure

1 INTRODUCTION

Knowledge Representation and Reasoning (KRR) is a subfield of Artificial Intelligence (AI) that takes a formal and, at times, philosophical approach to the problem of simulating intelligence. At the heart of this subfield is the notion of a knowledge-based system, where information is encoded symbolically and collated in a structure referred to as a *knowledge base*. Reasoning services are then defined to facilitate drawing conclusions from such knowledge bases.

A simple, yet expressive logic-based approach to KRR is defined in *classical propositional logic* (or *propositional logic*). While exhibiting several desirable characteristics, propositional logic has two fundamental limitations in its ability to mimic human reasoning.

Classical logics, including propositional logic, cannot explicitly express *typicality* whereby specific implications usually hold but may have exceptions. It is also *monotonic*, meaning conclusions drawn from some knowledge base cannot be retracted with the addition of new knowledge [8]. Such retractions are crucial in formalising the idea that new knowledge may require a re-examination of past conclusions.

In order to address these shortcomings, defeasible approaches to reasoning have been proposed as non-monotonic alternatives to classical forms of entailment. However, unlike classical entailment, there is no obvious way defeasible entailment ought to behave.

Kraus, Lehmann and Magidor (KLM) [8], proposed a set of properties as a thesis for how to define a 'sensible' or 'rational' notion of defeasible entailment. Two such examples are *rational closure* [11] and *lexicographic closure* [10], each representing distinct, valid patterns of human reasoning. The primary focus of this paper being on rational closure.

Both rational and lexicographic closure are characterised semantically from several distinct but equivalent perspectives. Currently, approaches to computing entailment, i.e. answering entailment queries, are based on semantics that involves ranking knowledge base formulas and classical entailment checking [11].

However, alternative model-based semantic characterisations of rational and lexicographic closure exist for which corresponding algorithms have not yet been explored and formalised in KLM [8] extensions of propositional logic.

Regarding answering entailment queries, both the formula-based algorithms proposed by Casini et al. in [3] and the model-based algorithms proposed within this paper can be characterised as having two distinct phases. A primary, once-off phase constructs a representation of the underlying knowledge and a secondary, repeatable phase in which the representation produced by the initial phase is used to answer the given query. We thus classify these phases as 'construction' and 'entailment', respectively.

Centrally, due to how rational defeasible entailment [3] is defined, having a model-based representation of the underlying knowledge allows one to answer entailment queries straightforwardly as part of this second phase. Thus a motivation for this work is the expectation that the challenges of constructing model-based representations of the underlying knowledge will precipitate overarching efficiency benefits to be reaped by the algorithms that form part of the entailment phase.

Hence, this paper attempts to fill this gap in the literature by focusing on the problem of algorithmically constructing model-based representations as part of rational closure entailment checking.

Additionally, we compare these newly developed approaches' computational and implementation efficiency against the existing algorithms and their implementations in [6]. In doing so, we develop an extension to the *TweetyProject* library collection [13] that we claim will benefit future work in KLM-style defeasible reasoning.

In section 2, we cover the necessary background material for this work. In section 3, we present our newly developed algorithms inspired by the model-based semantics of rational closure. We motivate the choices for their design and discuss their characteristics. In section 4, we analyse our proposed algorithms. In section 5, we discuss our extensions to the TweetyProject that underpins the implementations we provide. In section 6, we present the experiments conducted to compare the performance of particular algorithms

from section 3 against the implementations from [3]. Finally, in section 7, we discuss the theoretical findings from section 4 and the results from the experiments in section 6.

2 BACKGROUND

2.1 Propositional Logic

2.1.1 Syntax. We define a set $\mathcal{P}$ containing all atomic propositions, representing the most basic units of knowledge [1]. Several connectives are defined to construct expressive formulas from these atoms.

Formulas can consist of a single atom, the negations ($\neg$) of other formulas, or the combination of two other formulas using one of the binary connectives $\{\wedge, \vee, \rightarrow, \leftrightarrow\}$. The set of all well-formed formulas is often referred to as $\mathcal{L}$ (the language of propositional logic). Note, we also include the constants $\top$ and $\bot$ in $\mathcal{L}$, which denote a formula that is always true and a formula that is always false, respectively.

2.1.2 Semantics. The formulas described assume truth values inductively through the assigning of truth values to the atoms in the formulas and via the semantics of the connectives. This assignment is fulfilled by interpretations.

Definition 2.1. An interpretation $\mathcal{I}$ is defined as a function $\mathcal{I} : \mathcal{P} \mapsto \{T, F\}$ which maps each propositional atom to a value of T or F (true and false respectively).

The values of a formula are derived using the usual semantics for propositional logic connectives [1]. We denote the value of a formula α under a given interpretation $\mathcal{I}$ (of the atoms in $\mathcal{P}$) as v I(α). In most cases, we are interested in interpretations that satisfy a particular formula or set of formulas (such a set is termed a *knowledge base*). We define satisfaction using the symbol $\Vdash$ as follows:

Definition 2.2. $\mathcal{I} \Vdash \mathcal{K}$ (where $\mathcal{I}$ is an interpretation of the formulas in the knowledge base $\mathcal{K}$) if and only if v I(α) = T for every formula $\alpha \in \mathcal{K}$.

The definition for satisfaction of a single formula corresponds to the case of a singleton knowledge base.

Interpretations that satisfy a knowledge base are referred to as models of that knowledge base. We use the notation $Mod(\mathcal{K})$ (or $[\![\mathcal{K}]\!]$) to refer to the set of models of a knowledge base $\mathcal{K}$ (similarly for a single formula).

2.1.3 Entailment. Using the above model-based semantics, entailment (or logical consequence), denoted using the $\models$ symbol, can be defined.

Definition 2.3. A knowledge base $\mathcal{K}$ entails a formula α, written as $\mathcal{K} \models \alpha$, if and only if $Mod(\mathcal{K}) \subseteq Mod(\alpha)$.

Intuitively, whenever all the formulas in $\mathcal{K}$ are true under a given interpretation, such will be the case for α and so we are able to conclude α whenever we have $\mathcal{K}$.

Example 2.1. We have the following meta-variables

$$\mathcal{P} = \{m, l, p\}$$

that represent the atomic propositions "being a mammal" , "giving birth to live young" and "being a platypus" respectively. We can encode the knowledge:

(1) Mammals give birth to live young ($m \rightarrow l$)
(2) Platypuses are mammals ($p \rightarrow m$)

in a knowledge base $\mathcal{K} = \{m \rightarrow l, p \rightarrow m\}$.

Using this knowledge, we can conclude using classical entailment that platypuses give birth to live young. However this is not an accurate reflection of reality since platypuses are exceptional mammals that lay eggs. Thus, we can add the formula

$$p \rightarrow \neg l$$

to $\mathcal{K}$ to reflect the fact they do not give birth to live young. Unfortunately, classical reasoning will force us to conclude that platypuses cannot exist, since any model of $\mathcal{K}$ will require the atom p to be false.

In theory, one can adjust the knowledge base to specifically address platypuses as being exceptional, but this quickly becomes impractical as each further exception, for example the echidna, would warrant such a remodelling of the knowledge base. There is no convenient mechanism for an agent using classical logic to reconcile its 'beliefs' about the world when it is presented with new information inconsistent with its prior 'beliefs' [7].

2.2 KLM-Style Defeasible Reasoning

Initially, KLM [8] extended propositional logic by defining a 'consequence relation' $\mid\sim$ representing defeasible implications in an attempt to reasonably represent *typicality*. Extensions of this framework instead define $\mid\sim$ as an additional connective (where $\alpha \mid\sim \beta$, with propositional formulas α, β, is read as 'typically, if α, then β' [3]). This extended language is defined as $\mathcal{L}_P := \mathcal{L} \cup \{\alpha \mid\sim \beta \mid \alpha, \beta \in \mathcal{L}\}$ [7].

Let us refer to the agent analogy, where a knowledge base represents an agent's explicit knowledge about how its world works. In the case of propositional logic, each atom, thought to represent some atomic fact about the world, can be in one of two states: *true* or *false*. The formulas constructed from these atoms using the various logical connectives are, in some sense, an encoding of the mechanics and relationships between these atomic facts.

Propositional interpretations, therefore, correspond to the 'global state' of the agent's world - a unique combination of assignments of true and false values to all the atomic facts. Thus from this point onwards, we refer to these classical propositional interpretations as *worlds*.

It is reasonable to think that an agent is likely to deem particular worlds more typical than others. An agent would likely prefer more typical worlds over others that are less so. This is exactly the intuition behind the structure referred to as a *ranked interpretation* [11].

Definition 2.4. A ranked interpretation is a function $\mathcal{R} : \mathscr{U} \mapsto \mathbb{N} \cup \{\infty\}$, such that for every $i \in \mathbb{N}$, if there exists a $u \in \mathscr{U}$ such that $\mathcal{R}(u) = i$, then there must be a $v \in \mathscr{U}$ such that $\mathcal{R}(v) = j$ with $0 \leq j < i$, where $\mathscr{U}$ is the set of all possible propositional interpretations [7].

Ranked interpretations, therefore, assign to each world a rank (with lower ranks corresponding, semantically, with more typical

worlds and higher ranks with less typical worlds). According to the ranked interpretation, worlds with a rank of ∞ are impossible, whereas worlds with finite ranks are possible.

2.2.1 Satisfaction. Given that ranked interpretations indicate the relative typicality of worlds, it makes sense to define whether a ranked interpretation satisfies a defeasible implication based on the most typical worlds in that interpretation. In order to define the 'most typical worlds', a definition of minimal worlds concerning a formula in $\mathcal{L}$ is required.

Definition 2.5. Given a ranked interpretation $\mathcal{R}$ and any formula $\alpha \in \mathcal{L}$, it holds that $u \in [\![\alpha]\!]^{\mathcal{R}}$ (the models of α in $\mathcal{R}$) is minimal if and only if there is no $v \in [\![\alpha]\!]^{\mathcal{R}}$ such that $\mathcal{R}(v) < \mathcal{R}(u)$ [7].

The previous definition defines the concept of the 'best α worlds' (i.e. the lowest ranked, or most typical, of the worlds in which α is true).

Definition 2.6. Given a ranked interpretation $\mathcal{R}$ and a defeasible implication $\alpha \mathrel{|\!\sim} \beta$, $\mathcal{R}$ satisfies $\alpha \mathrel{|\!\sim} \beta$, written $\mathcal{R} \Vdash \alpha \mathrel{|\!\sim} \beta$ if and only if for every v minimal in $[\![\alpha]\!]^{\mathcal{R}}$, $v \Vdash \beta$. If $\mathcal{R} \Vdash \alpha \mathrel{|\!\sim} \beta$ then $\mathcal{R}$ is said to be a *model* of $\alpha \mathrel{|\!\sim} \beta$ [7].

The previous definition says that for a ranked interpretation $\mathcal{R}$ to satisfy a defeasible implication $\alpha \mathrel{|\!\sim} \beta$, it need only satisfy $\alpha \rightarrow \beta$ in the most typical (lowest-ranked) α worlds of $\mathcal{R}$.

In the case of a classical propositional formula $\alpha \in \mathcal{L}$, it is required that every finitely-ranked world in $\mathcal{R}$ satisfies α in order for $\mathcal{R}$ to satisfy α [3]. This definition is consistent with the idea that classical propositional formulas, which do not permit exceptionality, should be satisfied in every plausible world of a ranked interpretation if such a ranking is to satisfy the formula.

We define the *materialisation* of a defeasible knowledge base, $\mathcal{K}$, as $\overrightarrow{\mathcal{K}} := \{\alpha \rightarrow \beta \mid \alpha \mathrel{|\!\sim} \beta \in \mathcal{K}\}$ [3]. With this definition, we say that a propositional formula $\alpha \in \mathcal{L}$ is exceptional w.r.t. $\mathcal{K}$ iff $\overrightarrow{\mathcal{K}} \models \neg\alpha$.

Example 2.2. Consider Example 2.1 from earlier, we are now able to express that mammals typically give live birth ($m \mathrel{|\!\sim} l$), that platypuses are mammals ($p \rightarrow m$) and that platypuses typically do not give birth to live young ($p \mathrel{|\!\sim} \neg l$) in the knowledge base $\mathcal{K} = \{m \mathrel{|\!\sim} l, p \rightarrow m, p \mathrel{|\!\sim} \neg l\}$.

Then we have that platypuses (p) are exceptional while mammals (m) are not. This aligns with our intuitions of platypuses being exceptional or atypical mammals. Another takeaway is that one can only reason about the implications of exceptional formulas by disregarding the formulas concerning more general knowledge.

It is now possible to model knowledge that expresses typicality and thus handles exceptional cases more reasonably.

Note that for the algorithms we define, we consider only *defeasible knowledge bases* in which all formulas are normalised in terms of defeasible implications and note that any classical statement, α, can be written as the defeasible implication $\neg\alpha \mathrel{|\!\sim} \bot$ (simply note that for a given ranked interpretation, $\mathcal{R}$, $\mathcal{R} \Vdash \alpha$ iff $\mathcal{R} \Vdash \neg\alpha \mathrel{|\!\sim} \bot$) [3].

2.2.2 Entailment. As mentioned, defeasible entailment is not unique. We seek reasonable forms of non-monotonic entailment that permit the retraction of conclusions in cases where knowledge is added that contradicts these conclusions. Such entailment relations are defined by a set of postulates [8] which is extended to define more specific classes of entailment [3, 11]. Defeasible entailment relations that satisfy the KLM postulates are said to be *LM-rational* [3].

Two notable forms of LM-rational defeasible entailment are the aforementioned *rational* and *lexicographic closure*. As mentioned, this paper focuses on rational closure, a prototypical pattern of defeasible reasoning (one that is extremely conservative in abnormal cases)[3, 10], which is said to be the non-monotonic core of rational defeasible entailment relations [3].

2.3 Rational Closure

Consider the agent analogy again; it is fair to assume that different agents might have different ideas of what worlds are more typical than others and produce differing ranked interpretations. This also captures the idea that some agents are likely less adventurous (or more conservative) than others. Hence, defining a partial ordering, $\preceq_{\mathcal{K}}$, over the ranked interpretations was reasonable, such that more 'typical' or conservative ranked interpretations are preferred over less typical ones.

Definition 2.7. Given a knowledge base, $\mathcal{K}$, and $\mathcal{R}^{\mathcal{K}}$ the set of all ranked models of $\mathcal{K}$ (those ranked interpretations which satisfy $\mathcal{K}$), it holds for every $\mathcal{R}_1^{\mathcal{K}}, \mathcal{R}_2^{\mathcal{K}} \in \mathcal{R}^{\mathcal{K}}$ that $\mathcal{R}_1^{\mathcal{K}} \preceq_{\mathcal{K}} \mathcal{R}_2^{\mathcal{K}}$ if and only if for every $u \in \mathscr{U}$, $\mathcal{R}_1^{\mathcal{K}}(u) \leq \mathcal{R}_2^{\mathcal{K}}(u)$. [7]

Intuitively, this partial order favours ranked models that have their worlds 'pushed down' as far as possible [7] - corresponding to the most preferred and most conservative ranked interpretation. It has a unique minimal element, $\mathcal{R}_{RC}^{\mathcal{K}}$, as shown by Giordano et al. [5]. We now define *minimal ranked entailment* using this minimal element as follows:

Definition 2.8. Given a defeasible knowledge base $\mathcal{K}$, the minimal ranked interpretation satisfying $\mathcal{K}$, $\mathcal{R}_{RC}^{\mathcal{K}}$, defines an entailment relation, $\mathrel{|\!\approx}_{RC}$, called minimal ranked entailment, such that for any defeasible implication $\alpha \mathrel{|\!\sim} \beta$, $\mathcal{K} \mathrel{|\!\approx}_{RC} \alpha \mathrel{|\!\sim} \beta$ if and only if $\mathcal{R}_{RC}^{\mathcal{K}} \Vdash \alpha \mathrel{|\!\sim} \beta$. [7]

We clarify that minimal ranked entailment is just another name for the rational closure of a knowledge base. Furthermore, any defeasible entailment relation, $\mathrel{|\!\approx}$, is said to be LM-rational iff it can be generated from a ranked interpretation, $\mathcal{R}$. In the sense that $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$ iff $\mathcal{R} \Vdash \alpha \mathrel{|\!\sim} \beta$ [3, 11].

Thus, our work focuses on designing algorithms to produce representations 'compatible' with minimal ranked entailment as defined above or more generally the mechanics of satisfaction for ranked interpretations.

Although the original focus of this paper was purely on the semantics of rational closure and its model-theoretic construction, it is necessary to address its syntactic/algorithmic characterisation as the two are closely related. Furthermore, the natural progression of this investigation arrives at an algorithm that bares strong connections to these pre-existing approaches.

As mentioned, the current algorithmic approach proposed by Casini et al. in [3] for answering rational closure entailment queries is split into two phases that take the form of two algorithms: `BaseRank`, included as Algorithm 1, and `RationalClosure`, included as Algorithm 2.

Algorithm 1 BaseRank

Input: A knowledge base $\mathcal{K}$
Output: An ordered tuple $(R_0, ..., R_{n-1}, R_\infty, n)$
i := 0;
$E_0 := \overrightarrow{\mathcal{K}}$;
while $E_{i-1} \neq E_i$ **do**
    $E_{i+1} := \{\alpha \rightarrow \beta \in E_i \mid E_i \models \neg\alpha\}$;
    $R_i := E_i \setminus E_{i+1}$;
    $i := i + 1$;
end while
$R_\infty := E_{i-1}$;
$n := i - 1$;
return $(R_0, ..., R_{n-1}, R_\infty, n)$

BaseRank takes as input a knowledge base $\mathcal{K}$ and defines a sequence of subsets, $E_0, \ldots, E_{n-1}, E_\infty$ that is used to partition $\overrightarrow{\mathcal{K}}$ into a sequence of levels $R_0, \ldots, R_{n-1}, R_\infty$. The intuition being that the levels represent a ranking of the materialised formulas of the knowledge base according to their 'specificity' with lower ranks corresponding to more general or defeasible information [7].

Algorithm 2 RationalClosure

Input: A knowledge base $\mathcal{K}$, and a defeasible implication $\alpha \mid\sim \beta$
Output: **true**, if $\mathcal{K} \mid\approx \alpha \mid\sim \beta$, and **false** otherwise
$(R_0, ..., R_{n-1}, R_\infty, n)$:= BaseRank($\mathcal{K}$);
$i := 0$
$R := \bigcup_{j=0}^{j<n} R_j$;
while $R_\infty \cup R \models \neg\alpha$ and $R \neq \emptyset$ **do**
    $R := R \setminus R_i$;
    $i := i + 1$;
end while
return $R_\infty \cup R \models \alpha \rightarrow \beta$;

RationalClosure, then uses the ranking provided by BaseRank to answer the given query. The algorithm does this by combining the sequence of ranks together and iteratively removing the most general information (i.e. lowest rank) until either the antecedent is no longer exceptional or there are no more finite ranks left to remove. At which point, the algorithm returns whether the resulting subset of the knowledge base classically entails the materialised query. A result by Freund in [4], guarantees that RationalClosure returns *true* iff $KB \mid\approx_{RC} \alpha \mid\sim \beta$.

2.4 Related Work

2.4.1 Model-based Representations. Booth et al. [2] provide an algorithm for constructing the ranked model for Rational Closure in the context of Propositional Typicality Logic (PTL), which serves as inspiration for our foundational approach for our model-based algorithms within KLM-style Propositional Logic [3].

2.4.2 KLM-Style Defeasible Reasoning Implementations. Few software logic systems exist that support defeasible reasoning services. In particular, until last year's Scalable Defeasible Reasoning (SCADR) project [6], there did not exist implementations of rational and lexicographic closure for propositional logic consistent with the extended framework proposed by Casini et al [3]. Furthermore, the performance and scalability of rational and lexicographic closure in a software setting were not known. Thus, SCADR conducted preliminary investigations into the scalability of rational and lexicographic closure through their implementation and optimisation. SCADR implemented rational and lexicographic closure in Java using the *TweetyProject* - a collection of various Java libraries that implement approaches to different areas of artificial intelligence [13]. Although SCADR laid the foundations for implementing these algorithms, there was still the need for a platform or extension of the TweetyProject that provided abstract representations of the logical structures, such as ranked interpretations and defeasible knowledge bases, consistent with the literature to streamline future work.

3 MODEL-BASED ALGORITHMS

As alluded to earlier, the motivation for this investigation into devising algorithms based on the model-based semantics of rational closure is that possession of the minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC}$, theoretically provides a straightforward and efficient means of answering entailment queries. As a reminder, answering whether $\mathcal{K}$ entails a query, $\alpha \mid\sim \beta$, requires finding the minimal (with respect to the ranks in the minimal ranked model) α-world and then checking only whether all the α-worlds on such rank are also β worlds.

3.1 ModelRank

Algorithm 3 ModelRank

Input: A defeasible knowledge base $\mathcal{K}$
Output: A ranked interpretation $(R_0, ..., R_{n-1}, R_\infty)$ and the number of ranks, n
$i := 0$;
$\mathcal{P}_\mathcal{K} := \{p \mid p$ is a propositional letter occurring in $\mathcal{K}\}$;
$\mathscr{U}_i$:= universe of worlds for vocabulary $\mathcal{P}_\mathcal{K}$;
$\mathcal{K}_i := \overrightarrow{\mathcal{K}}$;
repeat
    $R_i := \{v \in \mathscr{U}_i \mid v \Vdash \mathcal{K}_i\}$;
    $\mathscr{U}_{i+1} := \mathscr{U}_i \setminus R_i$;
    $\mathcal{K}_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid \nexists v \in R_i$ s.t. $v \Vdash \alpha\}$;
    $i := i + 1$;
until $R_{i-1} = \emptyset$
$n := i - 1$;
$R_\infty = \mathscr{U}_i$;
$\mathcal{R}^* = (R_0, ..., R_{n-1}, R_\infty)$
return $\mathcal{R}^*, n$

3.1.1 Motivation. The next question to ask is: how does one acquire or construct the minimal ranked model for any given defeasible knowledge base?

The preference ordering over ranked interpretations in definition 2.7 characterises the minimal model with respect to other models of the knowledge base. We seek to develop an algorithm that constructs a representation of the minimal model directly, without the need to compare models.

A way to view this problem is to consider starting with all the worlds as most preferred as possible and then performing only the

most necessary 'bumping up' of worlds. The intuition is to place as many worlds as possible on each rank to produce not only a model but the minimal ranked model with all the worlds as 'pushed down' as the knowledge permits [7].

3.1.2 Overview. We start with all the possible worlds for the propositional vocabulary of the knowledge base. Then, at each step of the algorithm, we place all the worlds that are models of the remaining materialised formulas from our knowledge base on the current rank. All such worlds are then removed from the collection of to-be-placed worlds to ensure they cannot be placed on more than one rank. Finally, we remove all the formulas whose antecedents are satisfied by a world we have just placed on the current rank from our collection of to-be-considered formulas. Together these two steps construct the minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC}$, defined in definition 2.7.

We show that: *(i)* the `ModelRank` algorithm terminates given a finite defeasible knowledge base, *(ii)* $\mathcal{R}^*$ returned by `ModelRank` is a ranked model of $\mathcal{K}$, and *(iii)* $\mathcal{R}^*$ is the minimal ranked model for $\mathcal{K}$, i.e. $\mathcal{R}^* = \mathcal{R}^{\mathcal{K}}_{RC}$.

The complete proofs of all the subsequently mentioned lemmas and propositions are included in appendix B.

We show *(i)* in terms of proposition B.1. Termination follows from the fact that the knowledge base is finite and that the sequence of remaining worlds is decreasing.

We show *(ii)* in terms of proposition B.2. This is done by showing that the representation the algorithm produces is both a ranked interpretation and a model of the given knowledge base.

The proof of *(iii)* was more challenging and required Lemmas B.3, B.4, B.5 and B.6.

In lemma B.3, we show that for all formulas in the set difference between any two sets of remaining formulas, $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$, that there exists a world on rank $i-1$ that satisfies both α and β. The intuition behind the set difference $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$ is that these are the formulas whose best α-worlds must have been placed on rank $i-1$. Lemma B.4 is related to the previous lemma in that we now show there cannot exist an α-world any lower than rank $i-1$. Together, these two lemmas allow us to conclude that for all the formulas in the previously mentioned set difference, there are no $\alpha \wedge \beta$-worlds any lower than rank $i-1$. Finally, with lemma B.6 we show that for any world on a given rank i, $i > 0$, it must violate some formula in $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$. Given that the sequence of remaining formula sets is decreasing by definition, we find that the formula violated in $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$ must be in all the previous (i.e. within $\mathcal{K}_j, \forall j < i$) remaining formula sets. Thus if one were to move any world to a lower rank than the one it has been placed on by `ModelRank`, it would produce a ranked interpretation that is no longer a model of the knowledge base.

Example 3.1. Consider the knowledge base from Example 2.2: $\mathcal{K} = \{\mathtt{m} \mid\sim \mathtt{l}, \mathtt{p} \rightarrow \mathtt{m}, \mathtt{p} \mid\sim \neg\mathtt{l}\} = \{\mathtt{m} \mid\sim \mathtt{l}, \neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot, \mathtt{p} \mid\sim \neg\mathtt{l}\}$.

The corresponding universe of worlds is:

$$\mathscr{U} = \{\mathtt{l}\overline{\mathtt{mp}}, \mathtt{l}\overline{\mathtt{m}}\mathtt{p}, \mathtt{l}\overline{\mathtt{mp}}, \overline{\mathtt{l}}\mathtt{mp}, \overline{\mathtt{l}}\mathtt{m}\overline{\mathtt{p}}, \overline{\mathtt{l}}\overline{\mathtt{mp}}, \overline{\mathtt{l}\mathtt{mp}}, \mathtt{lmp}\}.$$

We briefly overview how `ModelRank` constructs the minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC}$. First, we consider which worlds in $\mathscr{U}_0 := \mathscr{U}$ are models of $\mathcal{K}_0 := \overrightarrow{\mathcal{K}}$. These worlds are $\overline{\mathtt{p}}\mathtt{ml}, \overline{\mathtt{pml}}, \overline{\mathtt{pml}}$, which we now place on rank 0. Hence we remove them from $\mathscr{U}_0$ to produce $\mathscr{U}_1$. We then see that one of the worlds we have just placed on R_0 is a m-world. Thus we remove $\mathtt{m} \mid\sim \mathtt{l}$ from $\mathcal{K}_0$ to produce $\mathcal{K}_1$. We repeat this process by continuing to place worlds that are models of the remaining formulas, removing the worlds we place and the formulas for which the placed worlds are models of their antecedents. The resulting minimal ranked model is shown in figure 1 (we ignore the infinite rank for compactness).

R_2	`lmp`
R_1	$\overline{\mathtt{l}}\mathtt{m}\overline{\mathtt{p}}$, $\overline{\mathtt{l}}\mathtt{mp}$
R_0	$\mathtt{l}\overline{\mathtt{mp}}$, $\mathtt{l}\mathtt{m}\overline{\mathtt{p}}$, $\overline{\mathtt{lmp}}$

Figure 1: Representation of $\mathcal{R}^{\mathcal{K}}_{RC}$ for Example 3.1

3.1.3 Entailment. Given that the representation produced by `ModelRank` is exactly $\mathcal{R}^{\mathcal{K}}_{RC}$, answering entailment queries is carried out as defined for minimal ranked entailment. We do however formalise the mechanics of how this is done as algorithm 6 in appendix A, which we refer to as `ModelSatisfaction`.

3.2 FormulaRank

3.2.1 Motivation. The previous algorithm directly produces the minimal ranked model in a representation consistent with its abstract definition in the literature [2, 3, 7]. As we discuss in section 4, there is a clear exponential relationship between the cardinality of the propositional vocabulary of a given knowledge base and the cardinality of the corresponding universe of worlds. Thus the space-complexity of the `ModelRank` algorithm is also exponential.

This challenge precipitated further investigation into new ways of representing ranked interpretations that are still compatible with the model-theoretic properties of minimal-ranked entailment but are still tractable alternatives to the current formula-based approaches.

Our first approach is then to construct formulas in correspondence with the ranks of the minimal ranked model such that the models of each formula correspond exactly with the worlds situated on the corresponding level in the minimal ranked interpretation. That is, for a knowledge base, $\mathcal{K}$, and its corresponding minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC} = (R_0, \ldots, R_{n-1}, R_\infty)$, we seek to construct a representation of the form $(F_0, \ldots, F_{n-1}, F_\infty)$, where each F_i is a propositional formula satisfying the condition: $Mod(F_i) = R_i$.

Hence, instead of enumerating the entire universe of worlds for the propositional vocabulary of the knowledge base and then determining whether such worlds satisfy specific criteria to place them on ranks, we instead place the 'criteria' itself on the ranks of the new representation.

Algorithm 4 `FormulaRank`

Input: A defeasible knowledge base $\mathcal{K}$
Output: A ranked formula interpretation $(F_0, ..., F_{n-1}, F_\infty)$ and the number of ranks, n
$i := 0$;
$\mathcal{K}_i := \overrightarrow{\mathcal{K}}$;
repeat
  $F_i := (\bigwedge \mathcal{K}_i) \wedge \neg(\bigvee_{j<i} F_j)$;
  $\mathcal{K}_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid F_i \models \neg\alpha\}$;
  $i := i + 1$;
until $\mathcal{K}_i = \mathcal{K}_{i-1}$
$n := i$;
$F_\infty := F_i$;
$F^* := (F_0, ..., F_{n-1}, F_\infty)$
return F^*, n

3.2.2 Overview. `FormulaRank` produces what can be seen as a declarative ranked model in the sense that, during a given iteration of each algorithm, where `ModelRank` determines which of the remaining worlds, $\mathscr{U}_i$, are models of our remaining knowledge, $\mathcal{K}_i$, `FormulaRank` instead constructs a declarative formula, F_i, whose models are precisely those of the remaining knowledge and not of any of the previous formulas $(F_0, \ldots, F_{i-1})$.

Thus each `FormulaRank` representative rank formula is comprised of the conjunction of all the remaining formulas and the negation of the disjunction of all the previous representative rank formulas:

$$F_i := (\bigwedge \mathcal{K}_i) \wedge \neg(\bigvee_{j<i} F_j)$$

The negation of the disjunction of all the previous representative rank formulas,

$$\neg(\bigvee_{j<i} F_j),$$

essentially asserts that we wish to exclude worlds that are already associated with the previous representative rank formulas.

We prove that $\forall i, Mod(F_i) = R_i$ and that `FormulaRank` terminates at the same point as `ModelRank` in the proof of proposition C.2 in appendix C.

Example 3.2. Again, consider the knowledge base from Example 2.2: $\mathcal{K} = \{\mathtt{m} \mid\sim \mathtt{l}, \neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot, \mathtt{p} \mid\sim \neg\mathtt{l}\}$.

We include the corresponding ranked formula interpretation produced by `FormulaRank` for $\mathcal{K}$ as Figure 2 (we ignore the infinite rank for compactness).

F_2	$(\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot) \wedge \neg F_0 \wedge \neg F_1$
F_1	$((\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot) \wedge (\mathtt{p} \mid\sim \neg\mathtt{l})) \wedge \neg F_0$
F_0	$(\mathtt{m} \mid\sim \mathtt{l}) \wedge (\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot) \wedge (\mathtt{p} \mid\sim \neg\mathtt{l})$

Figure 2: Representation of F^* for Example 3.2

3.2.3 Entailment. The representation produced by `FormulaRank` is not immediately compatible with the mechanics of minimal ranked entailment, which is defined in terms of worlds and not formulas. Therefore, we provide a modified version of the entailment algorithm for `ModelRank` that formalises the mechanics of how entailment queries are answered using this new representation. The aforementioned algorithm is included as algorithm 7 in appendix A and we refer to this algorithm as `FormulaModelSatisfaction`.

3.3 CumulativeFormulaRank

3.3.1 Motivation. After implementing the `FormulaRank` algorithm using our extension of the Tweety Project Library, we encountered severe performance issues. We determined the cause to be the interaction between the implementation of the Sat4j SAT solver [9] provided by the *TweetyProject* library [13] and the construction of the representative formulas on each rank (F_i).

When implementing `FormulaRank`, we reformulated the entailment query of step seven of the `FormulaRank` algorithm to a satisfiability query that we can present to the SAT solver. For example, $\alpha \models \beta \Leftrightarrow Mod(\alpha \wedge \neg\beta) \subseteq \emptyset$. Or in other words, iff $\alpha \wedge \neg\beta$ is unsatisfiable.

We discovered that the SAT solver then converts the given query to Conjunctive Normal Form (CNF) as part of its implementation. Due to the recurrence relation between the representative rank formulas, the number of nested negated disjunctions increases as the rank index increases. Consequently, when the SAT solver attempts to convert the representative formula to CNF, there is an exponential explosion in the length of the converted formula that results in heap memory issues within the runtime environment.

To solve this problem we had to determine whether it was possible to produce a more compact formula representation than the one produced by `FormulaRank`, that was still compatible with the mechanics of minimal ranked entailment.

Algorithm 5 `CumulativeFormulaRank`

Input: A defeasible knowledge base $\mathcal{K}$
Output: A cumulative ranked formula interpretation $(F'_0, ..., F'_{n-1}, F'_\infty)$ and the number of ranks, n
$i := 0$;
$\mathcal{K}'_i := \overrightarrow{\mathcal{K}}$;
repeat
  $F'_i := (\bigwedge \mathcal{K}'_i)$;
  $\mathcal{K}'_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}'_i \mid F'_i \models \neg\alpha\}$;
  $i := i + 1$;
until $\mathcal{K}'_i = \mathcal{K}'_{i-1}$
$n := i$
$F'_\infty := \top$
$F^{**} := (F'_0, ..., F'_{n-1}, F'_\infty)$
return F^{**}, n

3.3.2 Overview. On closer inspection of the properties of the representative formulas produced by `FormulaRank`, we realised that it is in fact possible to disregard the negation of the disjunction of all the previous representative rank formulas. The resulting sequence of formulas now represents an accumulation of worlds whereby the following property holds: $\forall i, Mod(F'_i) = \bigcup_{j=0}^{i} R_j$ where each F'_i is from the sequence of formulas produced by `CumulativeFormulaRank` and each R_i is a rank from $\mathcal{R}^{\mathcal{K}}_{RC}$ produced by `ModelRank`. We prove this result as part of proposition D.1 in appendix D.

We term this new representation the 'cumulative ranked formula model' of a knowledge base. Significantly, this new representation does not affect our ability to answer entailment queries using minimal ranked entailment since satisfaction only takes into account the minimal alpha worlds and this new representation neither affects the point at which they are found nor our ability to reason with them correctly. Furthermore, this new representation avoids the complexity issues relating to the conversion to CNF.

This new representation is intimately related to the original `BaseRank` and `RationalClosure` algorithms. The `BaseRank` ranks are constructed from the difference between successive sets of exceptional formulas. `RationalClosure` effectively reconstructs the sequence of exceptional sets initially produced by `BaseRank` from the `BaseRank` ranks to answer the entailment query.

We also note that the representative formula on a given finite rank of the cumulative ranked model is, in fact, the conjunction of the formulas in the exceptional set produced by `BaseRank` of the same index. Simply note the definition of E_i in `BaseRank` and the definitions of F'_i and $\mathcal{K}'_{i+1}$ in `CumulativeFormulaRank`.

Thus, not only does the cumulative ranked formula model provide a syntactic representation of the models of a given knowledge base in a cumulative sense, it functions as a cache of the information used by `RationalClosure` to answer entailment queries. Hence answering entailment queries using the cumulative ranked model is similar to the `RationalClosure` algorithm.

Example 3.3. Once more, consider the knowledge base from Example 2.2: $\mathcal{K} = \{\mathtt{m} \mid\sim \mathtt{l}, \neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot, \mathtt{p} \mid\sim \neg\mathtt{l}\}$.

For completeness, we include the corresponding cumulative ranked formula interpretation produced by `CumulativeFormulaRank` for $\mathcal{K}$ as Figure 3 (we ignore the infinite rank for compactness).

F_2	$(\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot)$
F_1	$(\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot) \wedge (\mathtt{p} \mid\sim \neg\mathtt{l})$
F_0	$(\mathtt{m} \mid\sim \mathtt{l}) \wedge (\neg(\mathtt{p} \rightarrow \mathtt{m}) \mid\sim \bot) \wedge (\mathtt{p} \mid\sim \neg\mathtt{l})$

Figure 3: Representation of F^{**} for Example 3.3

3.3.3 Entailment. `CumulativeFormulaRank` uses the same entailment query algorithm, termed `FormulaModelSatisfaction`, as `FormulaRank`, which again is included as algorithm 7 in appendix A.

4 MODEL-BASED ALGORITHMS ANALYSIS

4.1 Construction Algorithms

4.1.1 ModelRank. We present an estimation of the time and space complexity of the algorithm using Big O notation. We consider the number of classical satisfaction checks executed as our basic unit of computation for our time-complexity analysis of this algorithm.

The algorithm keeps track of the remaining worlds and formulas as part of two decreasing sequences, $\mathscr{U}_i$ and $\mathcal{K}_i$, respectively. The $\mathscr{U}_i$ sequence begins with the entire universe of worlds for the propositional vocabulary of the knowledge base. Thus if there are p atoms in the vocabulary, there will be a total of 2^p worlds. The $\mathcal{K}_i$ sequence begins with the materialisation of all the formulas in the knowledge base. Thus suppose there are n formulas in the knowledge base. Then, using a result by Giordano et al. [5] that links the base ranks of formulas concerning a given knowledge base, $\mathcal{K}$, to the ranks of worlds within the corresponding minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC}$, we can conclude there can be at most $n + 1$ ranks in the corresponding ranked model. Thus the repeat until loop will execute at most $n + 1$ times. Hence as a worst-case approximation, there will be $2^p \cdot (n + 1)$ satisfaction checks at line 8 of the algorithm. Furthermore, there will be, at most, $2^p \cdot (n + 1)$ satisfaction checks at line 10 of the algorithm. Thus in total, we have $n \cdot (2 \cdot 2^p(n + 1))$ satisfaction checks, which results in a time-complexity of $O(n^2 \cdot 2^p)$.

We clarify that these estimates are conservative upper bounds because they do not consider that the sequences of remaining formulas and remaining worlds are strictly decreasing. There are many factors to consider when estimating complexity due to the interactions between the varied characteristics of the particular knowledge base one is considering. For example, there can be situations where the knowledge base contains many defeasible formulas but has a relatively small vocabulary and vice versa.

We also claim that 2^p satisfaction checks can be considered to be approximately equivalent to a single entailment check in that, in the worst case, checking whether a particular entailment holds, say $\alpha \models \beta$, can be evaluated by determining whether $\alpha \wedge \neg\beta$ is unsatisfiable, as mentioned in section 3.3.1.

Thus, we can say that the worst case time complexity is $O(n^2)$, where n now represents classical entailment checks. We also note that this shows that computing the minimum ranked model via `ModelRank` is no harder than checking classical entailment since each classical entailment check reduces to the Boolean satisfiability problem, which is NP-complete.

Finally, the space complexity of the algorithm is primarily dictated by the number of worlds needed to be stored in memory, which is determined by the cardinality of the propositional vocabulary of the knowledge base, say p. Hence the space complexity is $O(2^p)$.

4.1.2 FormulaRank and CumulativeFormulaRank. Both algorithms keep track of the remaining formulas as part of the decreasing sequence of $\mathcal{K}_i$ sets. Again, for a knowledge base of size n formulas, they will perform at most $n + 1$ iterations. Each algorithm will perform a single entailment check for each remaining formula in each iteration's current $\mathcal{K}_i$ set. Thus the time complexity is $O(n \cdot (n + 1)) = O(n^2)$.

However, this does not give the complete picture of the algorithm's performance from an implementation perspective. The complexities or lengths of the formulas themselves have a significant impact on the complexity of the entailment check. This discrepancy is evident when we consider `FormulaRank` and `CumulativeFormulaRank` together. Both algorithms perform the same number of classical entailment checks. However, the representative formulas used by `FormulaRank` are far longer and more complex than the ones used by `CumulativeFormulaRank` and in practice, `FormulaRank` is entirely infeasible.

We estimate the space complexity of the `FormulaRank` and `CumulativeFormulaRank` algorithms regarding the number of syntactic knowledge base formulas their resulting representations comprise.

This decision is motivated by the fact that the representative formulas on each rank are entirely comprised of combinations of formulas from the original knowledge base.

Suppose there are n formulas in the knowledge base, $\mathcal{K}$. Then since $F_0 := \bigwedge \mathcal{K}_0 = \bigwedge \overrightarrow{\mathcal{K}}$, we have that F_0 contains n formulas. If we ignore that the sequence of $\mathcal{K}_i$s is decreasing, then we have an upper bound for each F_i of $2^i \cdot i$ formulas. Thus since there are at most $n+1$ ranks, we have a worst-case estimate for the space-complexity of $O(2^n \cdot n^2)$, where n represents the number of formulas in the knowledge base. On the other hand, for `CumulativeFormulaRank`, since we do not negate the disjunction of all the previous rank formulas, if we again make the simplifying assumption that the $\mathcal{K}_i$s are constant, we have that the space complexity is $O(n^2)$ in the number of formulas in $\mathcal{K}$.

In summary:

	Time	Space
ModelRank	$O(n^2)$	$O(2^p)$
FormulaRank	$O(n^2)$	$O(2^n \cdot n^2)$
CumulativeFormulaRank	$O(n^2)$	$O(n^2)$

4.2 Entailment Algorithms

`ModelSatisfaction` implicitly performs satisfaction checks to locate the minimum alpha world for a given query. Hence for a given knowledge base, $\mathcal{K}$, with a propositional vocabulary of cardinality, p, `ModelSatisfaction` has a worst-case time-complexity of $O(2^p)$ satisfaction checks. Using the same argument regarding the relationship between satisfaction and entailment checks, we can claim that worst-case time-complexity is $O(1)$ in the number of entailment checks.

On the other hand, `FormulaModelSatisfaction` performs at most n entailment checks resulting in a worst-case time-complexity is $O(n)$ in the number of entailment checks.

5 SYSTEM DEVELOPMENT AND IMPLEMENTATION

Although the contributions of this work are primarily theoretical, we felt there was merit in developing a cohesive minor software component that could aid not only the conceptual development of the algorithms presented within this paper but also assist future researchers within this area.

The software developed, therefore, primarily performed the role of a companion testing ground for the ideas we explored during the algorithms' development. The secondary role of the software developed was to acquire baseline comparisons between the newly developed algorithms' implementations and implementations of the pre-existing algorithms found in [3, 6].

5.1 Defeasible Extensions to the TweetyProject

The decision to utilise Java and the TweetyProject library was primarily motivated by the decision to build upon the work done in [6].

We provide parsing utilities for reading text files containing knowledge bases represented in a formula-per-line format. Furthermore, we implement class interfaces for all the relevant structures such as defeasible knowledge bases, ranked interpretations and ranked formula interpretations. In addition, all the model-based algorithms implement a generic or parameterised 'RankConstructor' interface for consistency and ease of use. Finally, we provide two minimal ranked entailment reasoners (one for ranked interpretations and one for ranked formula interpretations) that implement the 'DefeasibleReasoner' interface.

Regarding the algorithms themselves, we provide implementations of the pre-existing algorithms found in [3, 6] as well as all the model-based algorithms presented within this paper.

6 EXPERIMENT METHODOLOGY

One of the initial aims of this project was to compare the algorithms' theoretical and execution-time performance with that of the corresponding ranked-formula-based approaches in [6].

Related to this aim was the research question: how does the performance of the foundational and optimised model-based algorithms for rational closure compare to the corresponding ranked-formula-based implementations in [6] with respect to computational complexity and execution time, measured across knowledge bases and query sets of varying size (number of formulas) and structure (number and distribution of ranks)?

We hypothesised that performing entailment query checks using minimal ranked entailment on model-based knowledge representations would perform better than the pre-existing approaches based on ranked-formula semantics in terms of execution time.

Due to time constraints and the significant theoretical component of this paper, we conducted experiments intending to get a preliminary answer to our research question, leaving a more profound investigation to a future experimental continuation of this work.

6.1 Evaluation metrics

We developed a benchmark testbed using the Java Microbenchmark Harness (JMH) to perform accurate millisecond benchmarks of the algorithms on the evaluation data. The JMH handles warmup and records precise measurements by interfacing with the Java Virtual Machine (JVM). The metric used for the experiments is the average execution time across the measurement iterations (after the warmup iterations).

6.2 Evaluation Data Generation

[6] developed a Knowledge Base Generation Tool (KBGT) to generate the knowledge base sets used in testing their implementations. The KGBT accepts several parameters that control the structure of the knowledge bases it produces. For example, such parameters control the number of base ranks, the total number of formulas and the distribution of those formulas across the ranks.

6.2.1 Knowledge Bases. We wrote a small script to use the KBGT to deterministically produce all possible combinations of knowledge bases whose number of ranks and formulas per rank ranged over set intervals. The belief is that these two parameters are likely to have the most significant impact on the performance of the algorithms, and by trying all combinations over a reasonable range of values, we will be able to elicit the performance characteristics of the algorithms.

6.2.2 Query Sets. We then wrote a second script to select ten formulas randomly from each knowledge base to produce a corresponding query set, which we used to evaluate the entailment performance of the algorithms.

6.3 Experiments

Given the previously discussed implementation inefficiencies of `FormulaRank`, only `ModelRank` and `CumulativeFormulaRank` lend themselves to further investigation from an experimental or implementation perspective.

A taxonomy of the experiments that were conducted is as follows:

(1) **Construction Experiments**
   (a) `ModelRank` **vs** `BaseRank`
   (b) `CumulativeFormulaRank` **vs** `BaseRank`
(2) **Entailment Experiments**
   (a) `ModelRank` + `ModelSatisfaction` **vs** `BaseRank` + `RationalClosure`
   (b) `CumumlativeFormulaRank` + `FormulaModelSatisfaction` **vs** `BaseRank` + `RationalClosure`

Each construction experiment has a corresponding test set of knowledge bases, generated as described in 6.2. For each knowledge base in the test set, we record the average execution time taken by each algorithm to construct its corresponding representation.

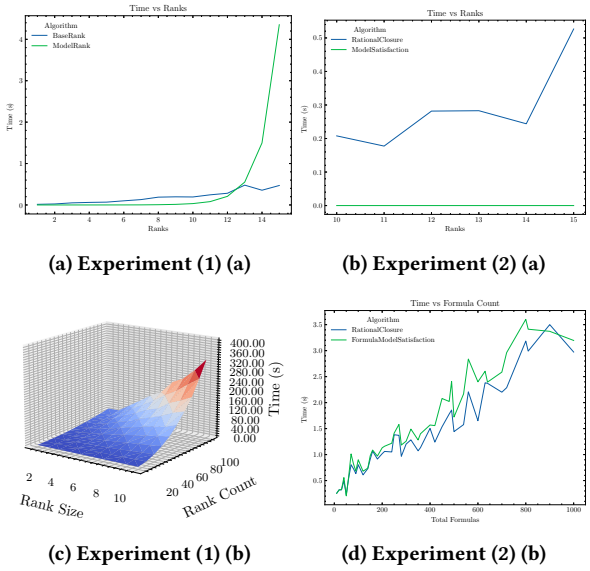
Each entailment experiment has a corresponding test set of knowledge bases in correspondence with companion query sets, again generated as described in 6.2. For each knowledge base and query set pair, we record the average execution time taken by each algorithm to answer all the queries in the test set using their respective representations (ignoring the time taken to construct those representations).

6.3.1 Experiments Involving ModelRank. The space-complexity issues with `ModelRank` mentioned in section 4 constrain the size of the propositional vocabulary we can use when generating knowledge bases. More specifically, large vocabulary sizes result in heap memory issues in the runtime environment. Thus, due to time and technical constraints, we limited our construction experiments for comparing `ModelRank` and `BaseRank` on knowledge bases whose number of base ranks ranges from 1 to 15 with only a single formula on each base rank. For entailment, we generate query sets as described in 6.2 for the knowledge bases whose number of ranks ranges from 10 to 15.

6.3.2 Experiments involving CumulativeFormulaRank. `CumulativeFormulaRank` does not suffer from such space complexity issues. Therefore we generated knowledge bases according to section 6.2 with the following characteristics: number of ranks from 10 to 100 in intervals of 10 and numbers of formulas per rank from 2 to 10, resulting in 90 distinct knowledge bases. For entailment, we used the script mentioned in section 6.2 to produce 90 corresponding query sets of 10 queries each.

6.3.3 Experiment Execution. All experiments were run on a machine with the following specifications:

- CPU: Apple M1 Pro, 10 cores
- Memory: 32GB



(a) Experiment (1) (a) **(b) Experiment (2) (a)**

(c) Experiment (1) (b) **(d) Experiment (2) (b)**

Figure 4: Experiment Results

As mentioned, the benchmarking testbed uses the JMH to record accurate benchmark results. However, due to the time constraints of performing 90 complete benchmarks for each algorithm, we set the JMH parameters as follows: *Fork*=1, *Measurement*=2, *Warmup*=1.

The *warmup* value controls the number of discarded executions performed to allow the JVM to perform any necessary procedures that may affect the measurement results. The *measurement* value determines how many benchmark iterations are measured and averaged together. Finally, *Fork* controls the number of times the complete warmup and measurement sequence is repeated.

6.3.4 Replicability. All the experiments conducted are easily reproducible. The code for all the implemented algorithms and KLM-style defeasible extensions to the TweetyProject will be open-sourced and made available on GitHub.

7 RESULTS AND DISCUSSION

Due to space limitations, we include a single figure for each experiment conducted to isolate a particular characteristic worth mentioning in the discussion. Additional results are included in Appendix E.

Figure 4a shows the results for comparing `ModelRank` and `BaseRank` in terms of execution time across the test set described in 6.3.1. Predictably, `BaseRank` is relatively efficient in producing its representation of the knowledge in terms of base ranks, given that the resulting representation is comprised of only the formulas from the knowledge base itself. Unfortunately, the theoretical analysis of `ModelRank` in section 4 does not give a complete picture of its performance in practice. For example, the graph shows an exponential relationship between execution time and the number of ranks of the knowledge bases for the `ModelRank` algorithm despite the claimed time complexity of $O(n^2)$. The cause of this discrepancy is likely the additional implementational overhead of enumerating the entire universe of worlds prior to the ranking process, which was

assumed to be immediately available during theoretical analysis. Hence, one should bare that in mind for future implementations.

On the other hand, if we now consider the results from Figure 4b, the benefits of using a ranked model representation with minimal ranked entailment are evident. As the number of ranks increases, the execution time of the implementation of `ModelSatisfaction` remains virtually constant. This result aligns with the computational complexity of the `ModelSatisfaction` algorithm ($O(1)$) and provides evidence that the theoretical performance benefits afforded by minimal ranked entailment carry over to implementation.

Figure 4c shows how the performance of the `CumulativeFormulaRank` algorithm scales across different combinations of numbers of ranks and formulas per rank. It is clear from the surface plot that rank size and rank count equally contribute to the execution times observed, which indicates that a better metric to consider is the total number of formulas within a given knowledge base. Although we do not plot the results for `BaseRank`, we note that its results produce a surface identical in shape to that of `CumulativeFormulaRank`, except achieving slightly lower execution times - this is expected given the similarities between the operations both algorithms perform.

Finally, Figure 4d compares the entailment performance of the implementation of `FormulaModelSatisfaction` using the representation produced by `CumulativeFormulaRank` against `BaseRank` and `RationalClosure`. It is clear that they achieve similar performance, but that `BaseRank` and `RationalClosure` generally outperform `FormulaModelSatisfaction` and `CumulativeFormulaRank`. The initial intuition was that the reverse would be true, given that the representation produced by `CumulativeFormulaRank` caches the information used by `RationalClosure`. This result is potentially a consequence of representing each rank as a conjunction of formulas instead of a set. We suspect that the Sat4j SAT solver implementation is more amenable to queries represented in terms of sets. We leave this as an investigation for future work.

Regarding model-based representations in general, one can view them from the perspective of a space-time trade-off because, at the cost of storing much more information, one can ideally answer entailment queries faster. `ModelRank` is an obvious example of this, in that the representation it constructs stores as much information as possible about the underlying knowledge and allows for almost constant time defeasible entailment query checking. We argue that it may be a worthwhile trade-off for situations where the propositional vocabulary size is relatively small, but the knowledge base contains many formulas.

8 CONCLUSIONS

This work represents an avenue largely unexplored in the literature: the design of model-based algorithms for computing forms of KLM-style defeasible entailment, namely rational closure.

We present three new algorithms for constructing representations of the rational closure ranked models of a given defeasible knowledge base. The first algorithm, `ModelRank`, constructs a representation consistent with its abstract definition in the literature. The second and third algorithms construct new compact representations for the ranked models using representative formulas. The final algorithm, `CumulativeFormulaRank`, produces a new class of representation that we term cumulative due to the cumulative nature in which worlds are associated with each rank. With all algorithms following the same bottom-up pattern of construction, based on the initial model ranking algorithm, we prove these produce the desired ranked models for computing the rational closure of a given knowledge base.

Finally, we implement and compare the performance of these proposed algorithms against existing approaches. In doing so, we provide a software foundation for future work.

9 FUTURE WORK

This work naturally leads to future theoretical as well as experimental work. For example, one direction is the extension of these model-based algorithms to general forms of rational defeasible entailment as defined in [3].

Another direction to consider is updating an already constructed representation of the underlying knowledge base if the knowledge base is updated itself. Furthermore, one can consider forms of entailment that allow for syntax-splitting, which refers to the restriction of attention to only the parts of the knowledge base that share atoms with a given query. One can also consider exploring these results for more expressive logics such as in [12].

Concerning the implementations of these algorithms, there are many optimisations and heuristics that can be explored. In particular, representing the ranks as sets of formulas instead of conjunctions is to be explored. Finally, we suggest that there is potentially pedagogical value in further extensions to the implementations we provide.

10 ACKNOWLEDGEMENTS

Thanks to our supervisor Tommie Meyer for his invaluable insights and guidance throughout this project and to my defeasible-reasoning comrade, Carl Combrinck, for his undying willingness to discuss the logical implications of egg-laying mammals.

REFERENCES

[1] Mordechai Ben-Ari. 2012. *Propositional Logic: Formulas, Models, Tableaux.* Springer London, London, 1, 7–47.

[2] Richard Booth, Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. 2015. On the Entailment Problem for a Logic of Typicality. In *Proceedings of the 24th International Conference on Artificial Intelligence* (Buenos Aires, Argentina) *(IJCAI'15)*. AAAI Press, 2805–2811.

[3] Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. 2019. Taking Defeasible Entailment Beyond Rational Closure. In *Logics in Artificial Intelligence*. Springer International Publishing, Cham, 182–197.

[4] Michael Freund. 1998. Preferential reasoning in the perspective of Poole default logic. *Artificial Intelligence* 98, 1 (1998), 209–235. https://doi.org/10.1016/S0004-3702(97)00053-2

[5] L. Giordano, V. Gliozzi, N. Olivetti, and G.L. Pozzato. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226 (2015), 1–33. https://doi.org/10.1016/j.artint.2015.05.001

[6] Joel Hamilton, Joonsoo Park, Aidan Bailey, and Thomas Meyer. 2022. *An Investigation into the Scalability of Defeasible Reasoning Algorithms.* SACAIR 2021 Organising Committee, Online, 235–251. https://protect-za.mimecast.com/s/OFYSCpgo02fL1l9gtDHUkY

[7] Adam Kaliski. 2020. *An Overview of KLM-Style Defeasible Entailment.* Master's thesis. Faculty of Science, University of Cape Town, Rondebosch, Cape Town, 7700.

[8] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence* 44, 1 (1990), 167–207. https://doi.org/10.1016/0004-3702(90)90101-5

[9] Daniel Le Berre and Anne Parrain. 2010. The SAT4J library, Release 2.2, System Description. *Journal on Satisfiability, Boolean Modeling and Computation* 7 (2010), 59–64. https://doi.org/10.3233/SAT190075

[10] Daniel Lehmann. 1999. Another perspective on Default Reasoning. *Annals of Mathematics and Artificial Intelligence* 15 (11 1999). https://doi.org/10.1007/BF01535841
[11] Daniel Lehmann and Menachem Magidor. 1992. What does a conditional knowledge base entail? *Artificial Intelligence* 55, 1 (1992), 1–60. https://doi.org/10.1016/0004-3702(92)90041-U
[12] Matthew Morris, Tala Ross, and Thomas Meyer. 2020. Algorithmic definitions for KLM-style defeasible disjunctive Datalog. *South African Computer Journal* 32, 2 (2020), 141–160.
[13] Matthias Thimm. 2017. The Tweety Library Collection for Logical Aspects of Artificial Intelligence and Knowledge Representation. *Künstliche Intelligenz* 31, 1 (March 2017), 93–97.

A ENTAILMENT ALGORITHMS

Algorithm 6 `ModelSatisfaction`

Input: A ranked interpretation $R^* := (R_0, ..., R_{n-1}, R_\infty)$, the number of ranks, n, and a query defeasible implication, $\alpha \mid\sim \beta$.
Output: **true** if $R^* \Vdash \alpha \mid\sim \beta$, otherwise **false**
$i := 0$;
while $R_i \cap [\![\alpha]\!] = \emptyset$ and $i < n$ **do** ▷ No α worlds found
    $i := i + 1$;
end while
return $R_i \cap [\![\alpha]\!] \subseteq [\![\beta]\!]$; ▷ All minimal α worlds are β worlds

Algorithm 7 `FormulaModelSatisfaction`

Input: A formula-based representation of a ranked interpretation $F^* := (F_0, ..., F_{n-1}, F_\infty)$, the number of ranks, n, and a query defeasible implication, $\alpha \mid\sim \beta$.
Output: **true** if $R^* \Vdash \alpha \mid\sim \beta$ where R is the ranked interpretation associated with F, otherwise **false**
$i := 0$;
while $F_i \wedge \alpha \models \bot$ and $i < n$ **do** ▷ No α worlds found
    $i := i + 1$;
end while
return $F_i \wedge \alpha \models \beta$; ▷ All minimal α worlds are β worlds

B MODELRANK ALGORITHM PROOFS

Proposition B.1. *The* `ModelRank` *algorithm terminates.*

Proof. We assume that $\overrightarrow{\mathcal{K}}$ is consistent. Thus, we want to show that $R_i = \emptyset$ for some i.

By definition, we have that $\forall j, R_j := \mathscr{U}_j \cap Mod(\mathcal{K}_j)$ and $\mathscr{U}_{j+1} \subseteq \mathscr{U}_j$.

Thus for arbitrary i, either:

(1) $\mathscr{U}_{i+1} \subset \mathscr{U}_i$
(2) $\mathscr{U}_{i+1} = \mathscr{U}_i$

Since $\mathscr{U}$ is finite, (1) can only occur a finite number of times. If $\mathscr{U}_{i+1} = \mathscr{U}_i$, then $R_i = \emptyset$ since $\mathscr{U}_{i+1} := \mathscr{U}_i \setminus R_i$ and $R_i \subseteq \mathscr{U}_i$.

□

Proposition B.2. *The* `ModelRank` *algorithm produces a ranked model of the given defeasible knowledge base,* $\mathcal{K}$.

Proof. Suppose `ModelRank` produces $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$. We therefore wish to show that $\mathcal{R}^*$ is a ranked model of $\mathcal{K}$.

We show this in two parts:

(1) **$\mathcal{R}^*$ a ranked interpretation:**
We show that $\mathcal{R}^*$ is a function from $\mathscr{U}$ to $\mathbb{N} \cup \{\infty\}$ such that $\mathcal{R}^*(u) = 0$ for some $u \in \mathscr{U}$, and satisfying the following convexity property: $\forall i \in \mathbb{N}$, if $\mathcal{R}^*(v) = i$, then, for $\forall j$ such that $0 \leq j < i$, $\exists u \in \mathscr{U}$ for which $\mathcal{R}^*(u) = j$.

We assume that $\overrightarrow{\mathcal{K}}$ is consistent.
Hence, $Mod(\overrightarrow{\mathcal{K}}) \neq \emptyset$. Thus $R_0 := \mathscr{U}_0 \cap Mod(\mathcal{K}_0) \neq \emptyset$.
Thus, $\exists u \in \mathscr{U}$ such that $\mathcal{R}^*(u) = 0$.
Take arbitrary $u \in \mathscr{U}$.

Either $u \in R_j$ or $u \notin R_j$ for some $j \in \mathbb{N}$.
If $u \in R_j$, then $u \in \mathscr{U}_j$ and $u \in Mod(\mathcal{K}_j)$.

$u \in \mathscr{U}_j$ and $u \in R_j$
$\Rightarrow u \notin \mathscr{U}_{j+1}$
$\Rightarrow \forall m > 0, u \notin \mathscr{U}_{j+1+m}$, since $\forall i > 0, \mathscr{U}_{i+1} \subset \mathscr{U}_i$
$\Rightarrow \forall m > 0, u \notin R_{j+1+m}$

If $u \notin R_j$ for some $j \in \mathbb{N}$, then $u \in \mathscr{U}_i$ for $i \leq n$. But $R_\infty := \mathscr{U}_n$, thus $u \in R_\infty$.
Thus, as soon as a world is placed on a rank, it can no longer be placed on any subsequent ranks. We note that the stopping condition of the algorithm is that the current rank is empty, and that this empty rank is excluded from the output. Thus, there can never be any empty ranks.

(2) $\mathcal{R}^*$ **is a model of** $\mathcal{K}$**:**
We want to show that $\forall \alpha \mid\sim \beta \in \mathcal{K}$, $min_{\prec}[\![\alpha]\!]^{\mathcal{R}^*} \subseteq [\![\beta]\!]^{\mathcal{R}^*}$.
We note that $min_{\prec}[\![\alpha]\!]^{\mathcal{R}^*}$ is just alternative notation for the minimal α-worlds with respect to the interpretation $\mathcal{R}^*$.
Take arbitrary $\alpha \mid\sim \beta \in \mathcal{K}$.
(a) If $[\![\alpha]\!]^{\mathcal{R}^*} = \emptyset$, we are done.
(b) If $[\![\alpha]\!]^{\mathcal{R}^*} \neq \emptyset$, then take arbitrary $v \in [\![\alpha]\!]^{\mathcal{R}^*}$.
Suppose $\mathcal{R}^*(v) = i$.
Note that $R_j := \mathscr{U}_j \cap Mod(\mathcal{K}_j)$ and

$$\mathcal{K}_j := \{\gamma \rightarrow \delta \in \mathcal{K}_{j-1} \mid \nexists v \in R_{j-1} \text{ s.t. } v \Vdash \gamma\}.$$

Since $v \in [\![\alpha]\!]^{\mathcal{R}^*}$ and $\mathcal{R}^*(v) = i$, we must have that $\forall j < i, \nexists u \in R_j$ such that $u \Vdash \alpha$.
Note that $\alpha \rightarrow \beta \in \mathcal{K}_0 := \overrightarrow{\mathcal{K}}$.
Hence, $\alpha \rightarrow \beta \in \mathcal{K}_j, \forall j \leq i$.
Since $v \in [\![\alpha]\!]^{\mathcal{R}^*}$ and $v \in R_i := \mathscr{U}_i \cap Mod(\mathcal{K}_i)$ and $\alpha \rightarrow \beta \in \mathcal{K}_i$, we have that $v \in [\![\beta]\!]^{\mathcal{R}^*}$.

□

Lemma B.3. *Suppose* ModelRank *produces* $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
Take arbitrary $v \in R_i$ *for* $i > 0$.
$\forall \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i, \exists w \in R_{i-1}$, *s.t.* $w \Vdash \alpha \wedge \beta$.

Proof. Take arbitrary $\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$

$$\begin{aligned}
\Rightarrow \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i &= \mathcal{K}_{i-1} \cap \overline{\mathcal{K}_i} \\
&= \mathcal{K}_{i-1} \cap \overline{\{\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \mid \nexists v \in R_i \text{ s.t. } v \Vdash \alpha\}} \\
&= \mathcal{K}_{i-1} \cap \overline{(\mathcal{K}_{i-1} \cap \{\alpha \rightarrow \beta \in \overrightarrow{\mathcal{K}} \mid \nexists v \in R_i \text{ s.t. } v \Vdash \alpha\})} \\
&= \mathcal{K}_{i-1} \cap (\overline{\mathcal{K}_{i-1}} \cup \overline{\{\alpha \rightarrow \beta \in \overrightarrow{\mathcal{K}} \mid \nexists v \in R_i \text{ s.t. } v \Vdash \alpha\}}) \\
&= \mathcal{K}_{i-1} \cap \{\alpha \rightarrow \beta \in \overrightarrow{\mathcal{K}} \mid \nexists v \in R_i \text{ s.t. } v \Vdash \alpha\} \\
&= \mathcal{K}_{i-1} \cap \{\alpha \rightarrow \beta \in \overrightarrow{\mathcal{K}} \mid \exists v \in R_i \text{ s.t. } v \Vdash \alpha\} \\
&= \{\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \mid \exists v \in R_i \text{ s.t. } v \Vdash \alpha\}
\end{aligned}$$

Since $\alpha \rightarrow \beta \in \{\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \mid \exists v \in R_i \text{ s.t. } v \Vdash \alpha\}$, take arbitrary $u \in R_{i-1}$ such that $u \Vdash \alpha$.
But $u \in R_i \Rightarrow u \in Mod(\mathcal{K}_{i-1}) \subseteq Mod(\mathcal{K}_{i-1} \setminus \mathcal{K}_i)$.
Hence, $u \Vdash \alpha$ and $u \in Mod(\mathcal{K}_{i-1} \setminus \mathcal{K}_i)$ and $\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i \Rightarrow u \Vdash \beta$. □

Lemma B.4. *Suppose* ModelRank *produces* $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
Let $i > 0$.
$\forall \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i, \forall j < i-1, \nexists w \in R_j$, *s.t.* $w \Vdash \alpha$.

Proof. Take arbitrary $\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$.
Suppose for the sake of contradiction that $\exists w \in R_j$ with $j < i-1$ and that $w \Vdash \alpha$.
By definition, $R_j = \mathscr{U}_j \cap Mod(\mathcal{K}_j)$.
$\Rightarrow w \in Mod(\mathcal{K}_j)$
By definition, $\forall m > 0, \mathcal{K}_{m+1} \subset \mathcal{K}_m$.
By assumption, $\alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i \subseteq \mathcal{K}_{i-1}$
$\Rightarrow \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \subset \cdots \subset \mathcal{K}_j \subset \mathcal{K}_{j-1} \subset \cdots \subset \mathcal{K}_1 \subset \mathcal{K}_0 := \overrightarrow{\mathcal{K}}$.
Note that $\mathcal{K}_{j+1} := \{\alpha \rightarrow \beta \in \mathcal{K}_j \mid \nexists v \in R_j \text{ s.t. } v \Vdash \alpha\}$.
Now, $\alpha \rightarrow \beta \in \mathcal{K}_j$ and $w \in R_j$ and $w \Vdash \alpha$.
Hence, $\alpha \rightarrow \beta \notin \mathcal{K}_{j+1} \Rightarrow \alpha \rightarrow \beta \notin \mathcal{K}_{i-1} \Rightarrow \alpha \rightarrow \beta \notin \mathcal{K}_{i-1} \setminus \mathcal{K}_i$.
Which is clearly a contradiction. Thus no such w exists. □

Lemma B.5. *Suppose* ModelRank *produces* $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
Take arbitrary $v \in R_i$ *for* $i > 0$.
$\forall \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$, $min_{\prec}[\![\alpha \wedge \beta]\!]^{\mathcal{R}^*} \subseteq R_{i-1}$.

Proof. This follows from Lemma B.3 and Lemma B.4 since Lemma B.3 shows that there exists a world with the specified property and Lemma B.4 shows that there does not exist a world with such property on any lower rank.

□

Lemma B.6. *Suppose* ModelRank *produces* $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
Take arbitrary $v \in R_i$ *for* $i > 0$.
$\exists \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$ *such that* $v \nVdash \alpha \rightarrow \beta$

Proof. Suppose for the sake of contradiction that $v \in R_i$ and $\forall \alpha \rightarrow \beta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i, v \Vdash \alpha \rightarrow \beta$.
Hence, $v \in Mod(\mathcal{K}_{i-1} \setminus \mathcal{K}_i)$.
By definition, $R_i = \mathscr{U}_i \cap Mod(\mathcal{K}_i) \subseteq Mod(\mathcal{K}_i)$.
$\Rightarrow v \in \mathscr{U}_i$ and $v \in Mod(\mathcal{K}_i)$.
Take arbitrary $x \in \mathcal{K}_{i-1}$.
Since $\mathcal{K}_i \subset \mathcal{K}_{i-1}$, we have that $\mathcal{K}_{i-1} = \mathcal{K}_i \cup (\mathcal{K}_{i-1} \setminus \mathcal{K}_i)$.
Hence, either $x \in \mathcal{K}_i$ or $x \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$.
If $x \in \mathcal{K}_i$, then since $v \in Mod(\mathcal{K}_i)$, $v \Vdash x$.
If $x \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i$, then since $v \in Mod(\mathcal{K}_{i-1} \setminus \mathcal{K}_i)$, $v \Vdash x$.
Thus $v \in Mod(\mathcal{K}_{i-1})$.
Hence, $v \in \mathscr{U}_{i-1}$ and $v \in Mod(\mathcal{K}_{i-1}) \Rightarrow v \in R_{i-1}$.
This is a contradiction since we assumed that $v \in R_i$ and we have shown that $\mathcal{R}^*$ is a ranked interpretation.

□

Consider the ordering $\preceq_{\mathcal{K}}$ on all ranked models of a knowledge base $\mathcal{K}$, which is defined as follows: $\mathcal{R}_1 \preceq_{\mathcal{K}} \mathcal{R}_2$ if for every $v \in \mathscr{U}$, $\mathcal{R}_1(v) \leq \mathcal{R}_2(v)$.

Proposition B.7. *Suppose* ModelRank *produces* $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
$\mathcal{R}^*$ *is the minimal ranked model of* $\mathcal{K}$ *with respect to* $\preceq_{\mathcal{K}}$.

Proof. Suppose ModelRank produces $\mathcal{R}^* = (R_0, \cdots, R_{n-1}, R_\infty)$.
Take arbitrary $v \in R_i$ for $i > 0$.

We want to show that if we remove v and place it on any rank lower than i, that the resulting ranked interpretation, is no longer a model of $\mathcal{K}$.

To do this, we use Lemma B.5 and Lemma B.6.

Lemma B.5 shows that all the best alpha worlds, that are also beta worlds, for any formula in $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$, are located on rank $i-1$.

Lemma B.6 then shows that there must be at least one formula, say $\gamma \rightarrow \delta$, in $\mathcal{K}_{i-1} \setminus \mathcal{K}_i$ that v violates.

Hence, $\gamma \rightarrow \delta \in \mathcal{K}_{i-1} \setminus \mathcal{K}_i \subset \mathcal{K}_{i-1} \subset \cdots \subset \mathcal{K}_0$.

Thus, we can conclude that

$$\mathcal{R}^{*'} := (R_0, \cdots, R_{i-k} \cup \{v\}, \cdots, R_i \setminus \{v\}, \cdots, R_{n-1}, R_\infty)$$

for some $0 < k \leq i$ is not a model of $\mathcal{K}$. □

C FORMULARANK ALGORITHM PROOFS

Lemma C.1. *Consider each set of remaining worlds, $\mathscr{U}_i$, as defined in the* ModelRank *algorithm as $\mathscr{U}_i := \mathscr{U}_{i-1} \setminus R_{i-1}, \forall i > 0$. One can write, $\forall i > 0, \mathscr{U}_i = \mathscr{U} \setminus \bigcup_{j=0}^{i-1} R_j$.*

Proof. We use induction.

- Base Case:

$$\begin{aligned} \mathscr{U}_1 &= \mathscr{U}_0 \setminus R_0 && \text{(by definition)} \\ &= \mathscr{U}_0 \setminus \bigcup_{j=0}^{0} R_j \end{aligned}$$

- Induction Step: suppose for some $k > 0, k \in \mathbb{N}$ that

$$\mathscr{U}_k = \mathscr{U} \setminus \bigcup_{j=0}^{k-1} R_j \text{ holds.}$$

We wish to show that

$$\mathscr{U}_{k+1} = \mathscr{U} \setminus \bigcup_{j=0}^{k} R_j$$

$$\begin{aligned} \mathscr{U}_{k+1} &= \mathscr{U}_k \setminus R_k && \text{(by definition)} \\ &= (\mathscr{U} \setminus \bigcup_{j=0}^{k-1} R_j) \setminus R_k \\ &= \mathscr{U} \setminus ((\bigcup_{j=0}^{k-1} R_j) \cup R_k) \\ &= \mathscr{U} \setminus \bigcup_{j=0}^{k} R_j \end{aligned}$$

□

Proposition C.2. *With respect to the* ModelRank *and* FormulaRank *algorithms, the representative formula, F_i, on each rank of the* FormulaRank *model, is related to the worlds on each rank, R_i, of the* ModelRank *model by the following property: $\forall i, Mod(F_i) = R_i$ and $\mathcal{K}'_i = \mathcal{K}_i$. Additionally, both algorithms terminate at the same point.*

Proof. Base Case:

We assume that $\mathcal{K}$ is consistent.

Thus we have that $R_0 := \mathscr{U}_0 \cap Mod(\mathcal{K}_0) = Mod(\overrightarrow{\mathcal{K}})$ is not empty.

Furthermore, for both ModelRank and FormulaRank, $\mathcal{K}_0 := \overrightarrow{\mathcal{K}}$ and $\mathcal{K}'_0 := \overrightarrow{\mathcal{K}}$. Thus $\mathcal{K}_0 = \mathcal{K}'_0$.

$$\begin{aligned} F_0 &:= \bigwedge \mathcal{K}'_0 \wedge \neg(\bigvee_{j<0} F_j) \\ &= \bigwedge \mathcal{K}'_0 \wedge \neg\bot \\ &= \bigwedge \mathcal{K}'_0 \wedge \top \\ &= \bigwedge \mathcal{K}'_0 \\ &= \bigwedge \mathcal{K}_0 && \text{(by definition)} \\ \Rightarrow Mod(F_0) &= Mod(\bigwedge \mathcal{K}_0) \\ &= Mod(\mathcal{K}_0) \\ &= \mathscr{U}_0 \cap Mod(\mathcal{K}_0) && \text{(since } \mathscr{U}_0 := \mathscr{U} \text{)} \\ &= R_0 \end{aligned}$$

We also know that both $\mathcal{K}_1$ and $\mathcal{K}'_1$ exist.

'Repeating Base Case':

Suppose that for some $i > 0$, $R_i \neq \emptyset$, that $\mathcal{K}_i = \mathcal{K}'_i$ and that $\forall k \leq i, Mod(F_k) = R_k$.

We first show that $\mathcal{K}_{i+1} = \mathcal{K}'_{i+1}$.

Note that

$$\mathcal{K}_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid \nexists v \in R_i s.t. v \Vdash \alpha\}$$

and

$$\mathcal{K}'_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}'_i \mid F_i \models \neg\alpha\}.$$

Since $\mathcal{K}_i = \mathcal{K}'_i$ by our induction hypothesis,

$$\mathcal{K}'_{i+1} = \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid F_i \models \neg\alpha\}.$$

Now,

$$\begin{aligned} F_i \models \neg\alpha &\Leftrightarrow Mod(F_i) \subseteq Mod(\neg\alpha) \\ &\Leftrightarrow R_i \subseteq Mod(\neg\alpha) && \text{(by induction hypothesis)} \\ &\Leftrightarrow \forall u \in R_i, u \in Mod(\neg\alpha) \\ &\Leftrightarrow \forall u \in R_i, u \Vdash \neg\alpha \\ &\Leftrightarrow \nexists u \in R_i, u \Vdash \alpha \end{aligned}$$

Thus, since $\mathcal{K}_i = \mathcal{K}'_i$ (by induction hypothesis), and $F_i \models \neg\alpha \Leftrightarrow \nexists u \in R_i$ s.t. $u \Vdash \alpha$, we have that $\mathcal{K}_{i+1} = \mathcal{K}'_{i+1}$.

Now,

$$\begin{aligned} F_{i+1} &:= \bigwedge \mathcal{K}'_{i+1} \wedge \neg(\bigvee_{j<i+1} F_j) \\ &= \bigwedge \mathcal{K}_{i+1} \wedge \neg(\bigvee_{j<i+1} F_j) \\ \Rightarrow Mod(F_{i+1}) &= Mod(\bigwedge \mathcal{K}_{i+1}) \cap Mod(\neg(\bigvee_{j<i+1} F_j)) \end{aligned}$$

Now,

$$\begin{aligned} Mod(\neg(\bigvee_{j<i+1} F_j)) &= \mathscr{U} \setminus Mod(\bigvee_{j<i+1} F_j) \\ &= \mathscr{U} \setminus \bigcup_{j=0}^{i} Mod(F_j) \\ &= \mathscr{U} \setminus \bigcup_{j=0}^{i} R_j && \text{(by induction hypothesis)} \\ &= \mathscr{U}_{i+1} && \text{(by lemma C.1)} \end{aligned}$$

Thus,

$$\begin{aligned} Mod(F_{i+1}) &= Mod(\bigwedge \mathcal{K}_{i+1}) \cap \mathscr{U}_{i+1} \\ &= \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\ &= R_{i+1} && \text{(by definition)} \end{aligned}$$

Now, if $\mathcal{K}'_{i+1} = \mathcal{K}'_i$, then we have that `FormulaRank` terminates. We must now show that `ModelRank` terminates at the same index $(i+1)$.

$$\begin{aligned} R_{i+1} :&= \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\ &= \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_i) \\ &= (\mathscr{U}_i \setminus R_i) \cap Mod(\mathcal{K}_i) \\ &= (\mathscr{U}_i \cap Mod(\mathcal{K}_i)) \setminus (R_i \cap Mod(\mathcal{K}_i)) \\ &= R_i \setminus R_i \\ &= \emptyset \end{aligned}$$

Thus `ModelRank` terminates at the same index.

□

D CUMULATIVEFORMULARANK ALGORITHM PROOFS

Proposition D.1. *With respect to the* `ModelRank` *and* `CumulativeFormulaRank` *algorithms, the representative formula, F'_i, on each rank of the* `CumulativeFormulaRank` *model, is related to the worlds on each rank, R_i, of the* `ModelRank` *model by the following property: $\forall i$, $Mod(F'_i) = \bigcup_{j=0}^{i} R_j$ and $\mathcal{K}'_i = \mathcal{K}_i$. Additionally, both algorithms terminate at the same point.*

Proof. Base Case:

We assume that $\mathcal{K}$ is consistent.
Thus we have that $R_0 := \mathscr{U}_0 \cap Mod(\mathcal{K}_0) = Mod(\overrightarrow{\mathcal{K}})$ is not empty.
Furthermore, for both ModelRank and FormulaRank, $\mathcal{K}_0 := \overrightarrow{\mathcal{K}}$ and $\mathcal{K}'_0 := \overrightarrow{\mathcal{K}}$. Thus $\mathcal{K}_0 = \mathcal{K}'_0$.

$$\begin{aligned} F'_0 &:= \bigwedge \mathcal{K}'_0 \\ &= \bigwedge \mathcal{K}_0 && \text{(by definition)} \\ \Rightarrow Mod(F_0) &= Mod(\bigwedge \mathcal{K}_0) \\ &= Mod(\mathcal{K}_0) \\ &= \mathscr{U}_0 \cap Mod(\mathcal{K}_0) && \text{(since } \mathscr{U}_0 := \mathscr{U} \text{)} \\ &= R_0 \\ &= \bigcup_{j=0}^{0} R_j \end{aligned}$$

We also know that both $\mathcal{K}_1$ and $\mathcal{K}'_1$ exist.

'Repeating Base Case':

Suppose for that for some $i > 0$, $R_i \neq \emptyset$, that $\mathcal{K}_i = \mathcal{K}'_i$ and that $\forall k \leq i$, $Mod(F'_k) = \bigcup_{j=0}^{k} R_j$.

We first show that $\mathcal{K}_{i+1} = \mathcal{K}'_{i+1}$.

Note that

$$\mathcal{K}_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid \nexists v \in R_i s.t. v \Vdash \alpha\}$$

and

$$\mathcal{K}'_{i+1} := \{\alpha \rightarrow \beta \in \mathcal{K}'_i \mid F'_i \models \neg\alpha\}.$$

Since $\mathcal{K}_i = \mathcal{K}'_i$ by our induction hypothesis,

$$\mathcal{K}'_{i+1} = \{\alpha \rightarrow \beta \in \mathcal{K}_i \mid F_i \models \neg\alpha\}.$$

We show that $\mathcal{K}_{i+1} \subseteq \mathcal{K}'_{i+1}$ and $\mathcal{K}'_{i+1} \subseteq \mathcal{K}_{i+1}$.
If $\mathcal{K}'_{i+1} \neq \emptyset$, then take arbitrary $\alpha \rightarrow \beta \in \mathcal{K}'_{i+1}$. Thus, we have that

$$\begin{aligned} F'_i \models \neg\alpha &\Leftrightarrow Mod(F'_i) \subseteq Mod(\neg\alpha) \\ &\Leftrightarrow \bigcup_{j=0}^{i} R_j \subseteq Mod(\neg\alpha) \\ &\Rightarrow R_i \subseteq Mod(\neg\alpha) \\ &\Rightarrow \alpha \rightarrow \beta \in \mathcal{K}_{i+1} \end{aligned}$$

If $\mathcal{K}_{i+1} \neq \emptyset$, then take arbitrary $\alpha \rightarrow \beta \in \mathcal{K}_{i+1}$. Thus, we have that

$$\begin{aligned} \nexists v \in R_i, \text{ s.t. } v \Vdash \alpha \Rightarrow \nexists v \in R_j, \forall j \leq i \text{ s.t. } v \Vdash \alpha \\ \Rightarrow \bigcup_{j=0}^{i} R_j \subseteq Mod(\neg\alpha) \\ \Rightarrow Mod(F'_i) \subseteq Mod(\neg\alpha) \\ \Leftrightarrow F'_i \models \neg\alpha \\ \Rightarrow \alpha \rightarrow \beta \in \mathcal{K}'_{i+1} \end{aligned}$$

Thus $\mathcal{K}_{i+1} = \mathcal{K}'_{i+1}$.

We now need to show that $Mod(F'_{i+1}) = \bigcup_{j=0}^{i+1} R_j$.

We first show that $\bigcup_{j=0}^{i+1} R_j \subseteq Mod(F'_{i+1})$.

$$\begin{aligned}
\bigcup_{j=0}^{i+1} R_j &= \bigcup_{j=0}^{i} R_j \cup R_{i+1} \\
&= Mod(F_i') \cup R_{i+1} \\
&= Mod(F_i') \cup (\mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\
&= Mod(\mathcal{K}_i) \cup (\mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\
&= (Mod(\mathcal{K}_i) \cup \mathscr{U}_{i+1}) \cap (Mod(\mathcal{K}_i) \cup Mod(\mathcal{K}_{i+1})) \\
&= (Mod(\mathcal{K}_i) \cup \mathscr{U}_{i+1}) \cap Mod(\mathcal{K}_{i+1}) \\
&\subseteq Mod(\mathcal{K}_{i+1}) \\
&= Mod(F_{i+1}')
\end{aligned}$$

Next, we show that $Mod(F_{i+1}') \subseteq \bigcup_{j=0}^{i+1} R_j$.

Suppose for the sake of contradiction that $Mod(F_{i+1}') \nsubseteq \bigcup_{j=0}^{i+1} R_j$.

$$\begin{aligned}
Mod(F_{i+1}') \nsubseteq \bigcup_{j=0}^{i+1} R_j &\Leftrightarrow Mod(\mathcal{K}_{i+1}') \nsubseteq \bigcup_{j=0}^{i+1} R_j \\
&\Leftrightarrow \exists v \in Mod(\mathcal{K}_{i+1}') \text{ s.t. } v \notin \bigcup_{j=0}^{i+1} R_j
\end{aligned}$$

Now,

$$\begin{aligned}
v \notin \bigcup_{j=0}^{i+1} R_j &\Leftrightarrow v \notin R_j, \forall j \leq i+1 \\
&\Rightarrow v \notin R_{i+1} = \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\
&\Rightarrow v \notin Mod(\mathcal{K}_{i+1}) = Mod(\mathcal{K}_{i+1}')
\end{aligned}$$

Thus, we have that $Mod(F_{i+1}') \subseteq \bigcup_{j=0}^{i+1} R_j$ and consequently, $Mod(F_{i+1}') = \bigcup_{j=0}^{i+1} R_j$.

Now, if $\mathcal{K}_{i+1}' = \mathcal{K}_i'$, then we have that `CumulativeFormulaRank` terminates.

We must now show that `ModelRank` terminates at the same index $(i+1)$.

$$\begin{aligned}
R_{i+1} :&= \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_{i+1}) \\
&= \mathscr{U}_{i+1} \cap Mod(\mathcal{K}_i) \\
&= (\mathscr{U}_i \setminus R_i) \cap Mod(\mathcal{K}_i) \\
&= (\mathscr{U}_i \cap Mod(\mathcal{K}_i)) \setminus (R_i \cap Mod(\mathcal{K}_i)) \\
&= R_i \setminus R_i \\
&= \emptyset
\end{aligned}$$

Thus `ModelRank` terminates at the same index.

□

E SUPPLEMENTARY INFORMATION

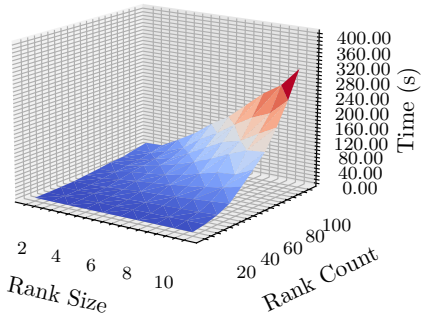


Figure 5: Experiment (1)(a) - BaseRank Execution Time Compared Against Number of Ranks and Rank Size

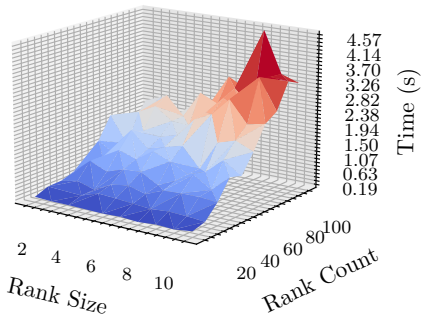


Figure 6: Experiment (2)(b) - FormulaModelSatisfaction Execution Time Compared Against Number of Ranks and Rank Size

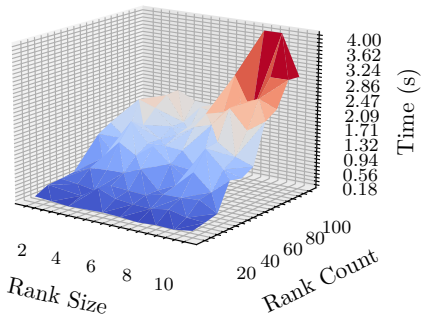


Figure 7: Experiment (2)(b) - RationalClosure Execution Time Compared Against Number of Ranks and Rank Size