Model-based Defeasible Reasoning

Project Proposal

Carl Combrinck
University of Cape Town
Cape Town, South Africa
CMBCAR007@myuct.ac.za

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

ABSTRACT

Defeasible reasoning attempts to formalize aspects of human reasoning in which prior conclusions can be retracted with the addition of new information, thereby addressing limitations of classical reasoning. Algorithms for computing several forms of defeasible consequence based on the ranking of formulas in a knowledge base, have been defined. Recent work has shown that particular models of a knowledge base characterize reasonable forms of defeasible entailment. However, there is work to be done in constructing, representing and using such models for defeasible entailment checking. In this project, we aim to formulate and analyse model-based entailment algorithms for two principle patterns of reasoning (Rational and Lexicographic Closure).

CCS CONCEPTS

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

KEYWORDS

artificial intelligence, knowledge representation and reasoning, defeasible reasoning, Rational Closure, Lexicographic Closure

1 INTRODUCTION

Knowledge representation and reasoning (KRR) is a subfield of artificial intelligence that attempts to formalize the expression of information and philosophical patterns of reasoning. Knowledge is encoded symbolically and collated in a structure referred to as a *knowledge base*. Reasoning services are then defined to facilitate the drawing of reasonable 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 a number of desirable characteristics, propositional logic has two fundamental limitations in its ability to mimic human reasoning.

Propositional logic cannot explicitly express *typicality* whereby certain 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 [7]. Such retractions are crucial in formalizing 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. Unlike classical entailment, there is no obvious way in which defeasible entailment ought to behave.

Kraus, Lehmann and Magidor (KLM) [7], proposed a set of properties as a thesis for how to define a 'sensible' or 'rational' notion of defeasible entailment. Two such examples, which will be our primary focus in this project, are *Rational Closure* [9] and *Lexicographic Closure* [8], each representing distinct, valid patterns of human reasoning.

Computing entailment for a given knowledge base, in both cases, has been defined based on semantics involving the ranking of formulas in the knowledge base [9]. Giordano et. al [4] provide an alternative but equivalent semantic characterization of Rational Closure based on a form of defeasible entailment known *minimal ranked entailment*. Casini et. al [3] extend this characterization to Lexicographic Closure, a refinement of Rational Closure, noting that it too can be characterized by a specific ranked model. Constructing model-based representations of these forms of entailment, algorithmically, will be the focus of this project.

2 BACKGROUND

2.1 Propositional Logic

2.1.1 Language. 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 possible formulas is often referred to as $\mathcal{L}$ (the language of propositional logic).

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 A under a given interpretation $\mathcal{I}$ (of the atoms in $\mathcal{P}$) as $v_{\mathcal{I}}(A)$.

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 $v_{\mathcal{I}}(A) = T$ for every formula $A \in \mathcal{K}$.

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 A, written as $\mathcal{K} \models A$, if and only if $Mod(\mathcal{K}) \subseteq Mod(A)$.

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

Example 2.1. Consider a knowledge base $\mathcal{K} = \{\mathsf{p} \vee \mathsf{q}, \neg\mathsf{p}\}$. $Mod(\mathcal{K}) = \{\overline{\mathsf{p}}\mathsf{q}\}$ ($\overline{\mathsf{p}}\mathsf{q}$ is shorthand for an interpretation that maps p to false and q to true). Consequently, $\mathcal{K} \models \mathsf{p} \rightarrow \mathsf{q}$ since $\overline{\mathsf{p}}\mathsf{q} \Vdash \mathsf{p} \rightarrow \mathsf{q}$ (equivalently, $\overline{\mathsf{p}}\mathsf{q} \in Mod(\mathsf{p} \rightarrow \mathsf{q})$) and so every model of $\mathcal{K}$ is also a model of $\mathsf{p} \rightarrow \mathsf{q}$.

2.2 Defeasible Reasoning

2.3 The KLM Framework and Extensions

Initially, KLM [7] 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 $\mathcal{L}_P := \mathcal{L} \cup \{\alpha \mid\sim \beta \mid \alpha, \beta \in \mathcal{L}\}$ [6]. The semantics of $\mid\sim$ are then defined using *ranked interpretations* [9].

Definition 2.4. A ranked interpretation is a function $\mathcal{R} : \mathscr{U} \mapsto \mathcal{N} \cup \{\infty\}$, such that for every $i \in \mathcal{N}$, 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 [6].

Ranked interpretations, therefore, assign to each propositional interpretation, a rank (with lower ranks corresponding, semantically, with more typical interpretations and higher ranks with less typical "worlds"). Worlds with a rank of ∞, according to the ranked interpretation, are impossible, whereas worlds with finite ranks are possible.

2.3.1 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 with respect to 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 $\mathcal{R}(v) < \mathcal{R}(u)$ [6].

This 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 $\alpha \mid\sim \beta$, $\mathcal{R}$ satisfies $\alpha \mid\sim \beta$, written $\mathcal{R} \Vdash \alpha \mid\sim \beta$ if and only if for every s minimal in $[\![\alpha]\!]^{\mathcal{R}}$, $s \Vdash \beta$. If $\mathcal{R} \Vdash \alpha \mid\sim \beta$ then $\mathcal{R}$ is said to be a *model* of $\alpha \mid\sim \beta$ [6].

This says that in order for a ranked interpretation $\mathcal{R}$ to satisfy a defeasible implication $\alpha \mid\sim \beta$, it need only satisfy $\alpha \rightarrow \beta$ in the most typical (lowest ranked) α worlds of $\mathcal{R}$.

In the case of a 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. This is consistent with idea that 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.

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

2.3.2 Entailment. 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 [7] which is extended to define more specific classes of entailment [3, 9]. We will look at two specific patterns of entailment, namely *Rational Closure* and *Lexicographic Closure* with a particular emphasis on their model-based semantics for the purposes of computing entailment.

2.4 Rational Closure

Rational Closure represents a prototypical pattern of defeasible reasoning (one that is extremely conservative in abnormal cases) in the KLM framework. Lehmann and Magidor [9] propose that any other reasonable form of entailment, while possibly being more "adventurous" in its conclusions, should endorse at least those assertions in the Rational Closure of the corresponding knowledge base.

There are 2 principle ways in which to compute the Rational Closure of a given knowledge base. The first is *minimal ranked entailment*. This approach defines Rational Closure and the semantics of the associated entailment relation using a unique ranked model for a given knowledge base. The second is an algorithmic approach involving the ranking of statements in the knowledge base [9]. Of these two methods, we will focus on the first due to its model-based approach.

2.4.1 Minimal Ranked Entailment. A partial order over all ranked models of a knowledge base $\mathcal{K}$, denoted $\preceq_{\mathcal{K}}$, is defined as follows [3]:

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 $u \in \mathscr{U}$, $\mathcal{R}_1^{\mathcal{K}}(u) \leq \mathcal{R}_2^{\mathcal{K}}(u)$.

Intuitively, this partial order favours ranked models that have their worlds "pushed down" as far as possible [6]. It has a unique minimal element, $\mathcal{R}_{RC}^{\mathcal{K}}$, as shown by Giordano et al. [4]. 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, $\mid\approx$, called minimal ranked entailment, such that for any

defeasible implication $\alpha \mid\sim \beta$, $\mathcal{K} \mid\approx \alpha \mid\sim \beta$ if and only if $\mathcal{R}^{\mathcal{K}}_{RC} \Vdash \alpha \mid\sim \beta$ [6].

Example 2.2. Consider the following knowledge base: $\mathcal{K} := \{\texttt{bird} \mid\sim \texttt{fly}, \texttt{bird} \mid\sim \texttt{wings}, \texttt{kiwi} \rightarrow \texttt{bird}\}$.

Intuitively, $\mathcal{K}$ suggests that birds usually fly, birds usually have wings, and kiwis are birds (kiwi here refers to the national bird of New Zealand). Using the partial order of ranked interpretations defined previously, the minimal ranked model, $\mathcal{R}^{\mathcal{K}}_{RC}$, of $\mathcal{K}$ is:

∞	$\overline{\texttt{bfkw}}\ \overline{\texttt{bf}}\texttt{kw}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{f}}\texttt{kw}\ \overline{\texttt{b}}\overline{\texttt{fkw}}$
1	$\texttt{b}\overline{\texttt{f}}\texttt{kw}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{b}\overline{\texttt{fkw}}$
0	$\texttt{bfkw}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{b}\overline{\texttt{f}}\texttt{kw}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}\ \overline{\texttt{b}}\texttt{fkw}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{f}}\texttt{kw}\ \overline{\texttt{b}}\overline{\texttt{fkw}}$

Figure 1: Minimal ranked model of $\mathcal{K}$

For brevity, each proposition is represented by a single letter.

We see that $\mathcal{R}^{\mathcal{K}}_{RC} \Vdash \texttt{kiwi} \mid\sim \texttt{wings}$ since the circled minimal kiwi world has that wings is true, i.e. it follows that kiwis typically have wings ($\mathcal{K} \mid\approx \texttt{kiwi} \mid\sim \texttt{wings}$).

Example 2.3. Suppose the statement $\texttt{kiwi} \rightarrow \neg\texttt{fly}$ (that kiwis do not fly) was added to $\mathcal{K}$. The minimal ranked model $\mathcal{R}^{\mathcal{K}\cup\{\texttt{kiwi}\rightarrow\neg\texttt{fly}\}}_{RC}$ of $\mathcal{K} \cup \{\texttt{kiwi} \rightarrow \neg\texttt{fly}\}$, is:

∞	$\overline{\texttt{bf}}\texttt{kw}\ \overline{\texttt{bf}}\overline{\texttt{kw}}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{f}}\texttt{kw}\ \overline{\texttt{b}}\overline{\texttt{fkw}}\ \texttt{bfkw}\ \texttt{bfk}\overline{\texttt{w}}$
1	$\overline{\texttt{bf}}\texttt{kw}\ \overline{\texttt{bf}}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}$
0	$\texttt{bf}\overline{\texttt{kw}}\ \texttt{b}\overline{\texttt{f}}\texttt{kw}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}\ \overline{\texttt{b}}\texttt{fkw}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{fkw}}$

Figure 2: Minimal ranked model of $\mathcal{K} \cup \{\texttt{kiwi} \rightarrow \neg\texttt{fly}\}$

Now, notice that $\mathcal{R}^{\mathcal{K}\cup\{\texttt{kiwi}\rightarrow\neg\texttt{fly}\}}_{RC} \not\Vdash \texttt{kiwi} \mid\sim \texttt{wings}$, since the circled minimal kiwi worlds do not both have wings being true. This demonstrates the non-monotonicity of Rational Closure, as a previous conclusion was retracted with the addition of new information. Importantly, it also demonstrates the conservative nature of prototypical reasoning, formalized in Rational Closure. In $\mathcal{K} \cup \{\texttt{kiwi} \rightarrow \neg\texttt{fly}\}$, kiwis are atypical birds since they are birds that do not fly and hence don't conform to the prototype of a bird, warranting the retraction of the conclusion in example 2.2.

2.5 Lexicographic Closure

Lexicographic Closure is a formalism of the presumptive pattern of reasoning introduced by Reiter [10] in the context of default logics. Presumptive reasoning is more "adventurous" and willing to conclude statements so long as their is no evidence to the contrary (even in atypical cases). The semantics of Lexicographic Closure depends on a "seriousness" ordering that is defined based on two criteria: specificity and cardinality. Like Rational Closure, there is both a ranked formula and model-based description of Lexicographic Closure [3].

Constructing the ranked model corresponding to Lexicographic Closure can be done as follows [3, 8]:

Definition 2.9. $m <^{\mathcal{K}}_{LC} n$ if and only if $\mathcal{R}^{\mathcal{K}}_{RC}(n) = \infty$, or $\mathcal{R}^{\mathcal{K}}_{RC}(m) < \mathcal{R}^{\mathcal{K}}_{RC}(n)$, or $\mathcal{R}^{\mathcal{K}}_{RC}(m) = \mathcal{R}^{\mathcal{K}}_{RC}(n)$ and m satisfies more formulas than n in $\mathcal{K}$ [3].

This definition characterises Lexicographic Closure as a refinement of Rational Closure, in that its ranked model respects the rankings of that for Rational Closure (which encodes seriousness) but refines preference for worlds with the same rank (based on cardinality).

Example 2.4. Returning to our kiwi example where $\mathcal{K} := \{\texttt{bird} \mid\sim \texttt{fly}, \texttt{bird} \mid\sim \texttt{wings}, \texttt{kiwi} \rightarrow \texttt{bird}, \texttt{kiwi} \rightarrow \neg\texttt{fly}\}$, we can construct the model $\mathcal{R}^{\mathcal{K}}_{LC}$ corresponding to Lexicographic Closure by "lifting up" worlds that satisfy fewer statements while preserving the original Rational Closure ordering.

∞	$\overline{\texttt{bfkw}}\ \overline{\texttt{bf}}\texttt{kw}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{f}}\texttt{kw}\ \overline{\texttt{b}}\overline{\texttt{fkw}}\ \texttt{bfkw}\ \texttt{bfk}\overline{\texttt{w}}$
2	$\texttt{b}\overline{\texttt{f}}\texttt{k}\overline{\texttt{w}}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}$
1	$\texttt{b}\overline{\texttt{f}}\texttt{kw}\ \texttt{b}\overline{\texttt{f}}\texttt{k}\overline{\texttt{w}}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}$
0	$\texttt{bf}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{bf}\overline{\texttt{kw}}\ \texttt{b}\overline{\texttt{f}}\texttt{kw}\ \texttt{b}\overline{\texttt{f}}\overline{\texttt{kw}}\ \overline{\texttt{b}}\texttt{fkw}\ \overline{\texttt{b}}\texttt{f}\overline{\texttt{kw}}\ \overline{\texttt{b}}\overline{\texttt{fkw}}$

Figure 3: Ranked model for Lexicographic Closure of $\mathcal{K}$

Checking if the formula, $\texttt{kiwi} \mid\sim \texttt{wings}$, is satisfied by $\mathcal{R}^{\mathcal{K}}_{LC}$, and hence in the Lexicographic Closure of $\mathcal{K}$, we find that $\mathcal{R}^{\mathcal{K}}_{LC} \Vdash \texttt{kiwi} \mid\sim \texttt{wings}$ and hence that $\mathcal{K} \mid\approx_{LC} \texttt{kiwi} \mid\sim \texttt{wings}$.

Notice that Lexicographic Closure endorses that kiwis have wings whereas Rational Closure would not. This speaks to the presumptive nature of Lexicographic Closure, as it is willing to assert that kiwis have wings despite the fact that kiwis are atypical birds (since there is no knowledge to the contrary).

This also demonstrates that there are multiple valid solutions to the problem of defeasible entailment. We may prefer the behaviour of Rational Closure in this case (since we know kiwis don't have wings), but may prefer Lexicographic Closure if kiwis are substituted for penguins, which do have wings.

3 PROJECT DESCRIPTION

3.1 Overview

Defeasible approaches to reasoning have been developed to address the limitations of classical reasoning.

Prominent forms of defeasible entailment, specifically Rational and Lexicographic Closure, are characterised semantically from several distinct, but equivalent perspectives. Formal approaches to computing entailment are currently based on semantics that involve the ranking of knowledge base formulas, and classical entailment checking [9]. Algorithms based on alternative model-based semantics have not yet been explored and formalised in KLM [7] extensions of propositional logic.

The project will focus on devising algorithms for constructing such models and performing entailment checking. Additionally, the computational and implementation efficiency of such approaches with respect to existing algorithms and their implementations in [5], will be analysed.

3.2 Motivation

Defeasible reasoning approaches have produced more accurate formal characterisations of human reasoning.

In the cases of Rational and Lexicographic Closure, model-based semantics are defined but have not been used to formulate entailment algorithms. Exploring such may provide a more efficient means of computing entailment compared to traditional formula-based approaches.

Most notably, Lexicographic Closure's current rank-based entailment approach [3] is intractable, further motivating the exploration of alternative methods of computation.

4 PROBLEM STATEMENT

The project will attempt to address the gap in existing literature pertaining to the lack of algorithms, based on ranked-model semantics, for computing the Rational and Lexicographic Closure of knowledge bases.

4.1 Aims

In this project, we aim to:

- Propose an abstract representation of the ranked models compatible with the operations performed during entailment checking.
- Devise model-based algorithms for constructing the ranked models corresponding to both Rational and Lexicographic Closure.
- Provide an algorithmic specification for computing entailment using the constructed models.
- Improve the devised model-based algorithms in terms of computational complexity and run-time performance (execution time).
- Implement the proposed entailment checking algorithms for Rational and Lexicographic Closure.
- Compare the algorithms' theoretical and execution-time performance with that of the corresponding ranked-formula based approaches in [5].

4.2 Research Questions

Our work will aim to address the following research questions:

Jaron Cohen

(1) How can the ranked model, corresponding to Rational Closure, be represented abstractly and constructed algorithmically for the purposes of entailment checking?
(2) How can such a model-based algorithm for computing Rational Closure be optimized theoretically, in terms of its computational complexity, and in its implementation, measured experimentally using execution time?
(3) How does the performance of the foundational and optimised model-based algorithms for Rational Closure compare to the corresponding formula-based implementations in [5] 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)?

Carl Combrinck

(1) How can the ranked model, corresponding to Lexicographic Closure, be represented abstractly and constructed algorithmically for the purposes of entailment checking?
(2) How can such a model-based algorithm for computing Lexicographic Closure be optimized theoretically, in terms of its computational complexity, and in its implementation, measured experimentally using execution time?
(3) How does the performance of the foundational and optimised model-based algorithms for Lexicographic Closure compare to the corresponding formula-based implementations in [5] 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)?

5 METHODS AND PROCEDURES

5.1 Overview

The initial stages of this project will be largely theoretical and pertain to the design of aforementioned model-based algorithms for computing Rational and Lexicographic Closure. Thereafter, optimised versions of these algorithms will be produced with both initial and optimised versions being implemented in Java. Using the Knowledge Base Generation Tool (KBGT) from [5] and a custom bench-marking tool, we will analyse the performance characteristics of these new algorithms and compare them to the algorithms currently available in the literature [3, 5].

5.2 Foundational Algorithm Design

Though work has already been done to familiarise ourselves with the necessary material, a thorough review of the present algorithms and optimisation approaches will be conducted.

We will focus on developing the foundational mechanics of our approach for Rational Closure with the intention of extending this approach to Lexicographic Closure. Specifically, we will agree on an appropriate abstract representation of the underlying models and basic algorithms for obtaining the desired ranked models.

We will then design draft versions of our model-based algorithms for Rational and Lexicographic Closure and present them to our supervisor for review. If our drafts are deficient, we will correct them based on feedback received. Otherwise, we will move onto finalising both initial algorithms. Firstly, we will provide formal high-level descriptions of our algorithms and proofs of their correctness. Secondly, we will analyse their computational complexity - as this will serve as the baseline for future optimisations.

5.3 Implementation and Optimised Algorithm Design

Once serviceable algorithms have been described, we will develop prototype implementations in Java to corroborate their behaviour and performance characteristics prior to optimisation. These implementations will require integration with the logics library collection used in [5], referred to as the *TweetyProject*.

We will use the KBGT [5] to produce various datasets comprising knowledge bases and query sets that control for size in terms of

number of formulas, number of ranks and the distribution of the formulas across the ranks. Unit tests will be produced to ensure that the prototype implementations of our foundational model-based algorithms are consistent with their formula-based counterparts found in [5]. A basic automated testbed, that uses the Java Microbenchmark Harness (JMH), will be built to facilitate the capturing of benchmark results. We will initially use the testbed to corroborate the calculated computational complexity of our algorithms.

Our next stage of work will concern devising optimised algorithms with improved time and space complexity, using two main approaches. Firstly, by trying to exploit the underlying logical structures and semantics of Rational and Lexicographic Closure and, secondly, by applying several traditional optimisation heuristics. An agile/iterative approach to this stage of work will be adopted, testing various optimisation heuristics and ideas as modifications of our foundational algorithms. These will be validated by our unit tests and naively analysed using our bench-marking testbed.

Once several approaches have been explored, draft descriptions of the optimised algorithms will be presented to our supervisor for feedback and review. We will conclude this phase by formalising descriptions of our optimised algorithms, proving their correctness, and analysing their computational complexity.

5.4 Analysis of Algorithms

The final stage of our work will concern the complete analysis of our finalised foundational and optimised algorithm implementations, including comparing their performance with corresponding implementations in [5].

Once we have tested the implementations of our finalised algorithms, we will extend the automated testbed to perform a wider range of experiments. A variety of new datasets for these experiments will be produced using the KBGT [5] and will be designed to elicit the performance characteristics of the model-based algorithms. Experiments will be conducted on a fixed device and will involve performing and timing repeat runs of our implementations and those in [5] on the generated test cases - warmup and measurement being accurately handled by the JMH.

Such analysis may afford recommendations regarding circumstances in which our algorithms out-perform current approaches.

6 RELATED WORK

Booth et al. [2] provide an algorithm for constructing the ranked model for Rational Closure in the context of Propositional Typicality Logic (PTL). Our foundational approaches will likely adapt this algorithm for KLM-style Propositional Logic [3].

All works that relate to the KLM framework and in particular Rational Closure and Lexicographic Closure are also relevant.

The seminal KLM paper [7] introduces the eponymous framework, providing a number of postulates that characterise KLM-style approaches to defeasible reasoning. Subsequently, Lehmann and Magidor [9] introduced Rational Closure in model-theoretic terms as well as a formula-based algorithm to answer entailment queries. Lehmann [8] introduces Lexicographic Closure as a bolder, presumptive, form of rational defeasible entailment.

Casini et al. presented the systematic approach for extending the KLM framework for defeasible entailment in [3]. Our work falls under the umbrella of this extension to the KLM framework and we will be comparing the implementations of our model-based algorithms with implementations of Rational Closure and Lexicographic Closure found in [5] based on formula-ranking algorithms described in [3].

7 ETHICAL, PROFESSIONAL AND LEGAL ISSUES

This project comprises both theoretical and practical components.

Since neither component will include user testing, nor any privacy-breaching experiments or data collection, we do not foresee any such associated ethics issues.

In terms of intellectual property rights, the Oracle OpenJDK releases for Java utilise the GNU General Public License (GPL), version 2, with the Classpath Exception - hence there are no legal concerns for the use of such software in this project. The main external library we intend to make use of for our implementations, the *TweetyProject*, utilises the GNU General Public License, thus we are free to use the software for our research purposes.

The KBGT [5] is published under the MIT license, allowing the free use and modification of the tool subject to the inclusion of the original copyright and license permissions in derivatives.

We will, therefore, be licensing the software produced by this project under an open source software license (likely MIT) and will heed all necessary open source software guidelines during the development of our source code.

8 ANTICIPATED OUTCOMES

8.1 Expected Impact

The work of this project represents an avenue largely unexplored in the literature in terms of the design of algorithms that compute defeasible entailment. We expect that we will gain insights into the algorithmic complexity of such algorithms. It is likely that model-based algorithms will scale differently to formula-based ranking algorithms and may be most suited to certain situations or applications. Furthermore, efficiency improvements in entailment checking may increase the feasibility of using larger and more complex knowledge bases in practice. In particular, we expect to see the greatest efficiency improvements for Lexicographic Closure. Importantly, determining whether model-based algorithms are a feasible alternative to the current formula-based ranking algorithms will likely encourage further investigation into their design and utilisation.

8.2 Key Success Factors

The project will be considered a success if:

(1) Correct foundational model-based Rational Closure and Lexicographic Closure algorithms have been produced.
(2) Foundational algorithms were optimised to produce new versions that have significantly improved time and space complexity.
(3) Working implementations of both the foundational and optimised model-based algorithms have been developed.

(4) These implementations allowed us to adequately assess the feasibility of these new algorithms with respect to existing approaches.

8.3 System

Java-based implementations of both the foundational and optimised model-based algorithms for checking defeasible entailment are to be produced. These will likely remain command-line executable and will integrate with pre-existing tools from the SCADR project [5] and *TweetyProject* library collection. Hence, the major design challenges will likely concern integration with these code bases. Additionally, a custom benchmarking testbed will be developed for the algorithms produced in this project and will be used to automate the acquisition of performance data for comparison with pre-existing implementations. Design challenges for the testbed will relate to the utilisation of the JMH.

9 PROJECT PLAN

9.1 Risks and Contingencies

A number of risks have been enumerated in Table 3 in Appendix A alongside their associated probabilities of occurrence and the impact on the project. Mitigation, monitoring and management strategies are then provided for each risk in the corresponding Table 4 also in Appendix A. This risk assessment indicates that there is an overall low-risk of project failure.

9.2 Resources Required

In terms of the theoretical aspects of the project, access to the relevant literature such as textbooks and research journals will be important. For the practical components, we will need computers to develop and implement the algorithms capable of running software such as the IntelliJ and VSCode IDEs. Furthermore, we will require access to open source software such as Java (OpenJDK), the *TweetyProject* and the source code for the implementation of the tools and algorithms in [5].

9.3 Deliverables

Deliverable	Due Date
Literature Reviews	4 May
Project Proposal Draft	22 May
Project Proposal Final	27 May
Project Proposal Presentation	24 May
Software Feasibility Demonstration	25 July
Final Paper Draft	23 August
Final Project Papers	2 September
Final Project Code	5 September
Final Project Presentation	19 September
Project Poster	3 October
Project Webpage	10 October

Table 1: Deliverables

9.4 Milestones and Tasks

Tasks	Start Date	End Date
(1) Literature Review	28/03	4/5
(2) Project Proposal	10/5	27/5
(a) Draft	10/5	22/5
(b) Presentation	22/5	24/5
(c) Final	25/5	27/5
(3) Preliminary Work	28/5	9/6
(a) Refinement of overall approach	28/5	7/6
(b) Review with supervisor	8/6	9/6
(4) Foundational Algorithm Development (Rational and Lexicographic Closure)	10/6	26/6
(a) Draft Development	10/6	17/6
(b) Feedback and Review	16/6	17/6
(c) Algorithm Finalisation	18/6	23/6
(d) Theoretical Analysis	22/6	26/6
(5) Implementation and Optimised Algorithm Design	27/6	27/7
(a) Prototype Foundational Implementation Development	27/6	6/7
(b) Unit Testing and Data-set Generation	4/7	6/7
(c) Basic Testbed Development	4/7	10/7
(d) Preliminary Analysis	9/7	11/7
(e) Optimisation Approach Exploration	4/7	17/7
(f) Prototype Optimisation Implementation Development	9/7	17/7
(g) Draft Optimisation Description	16/7	22/7
(h) Feedback and Review	20/7	22/7
(i) Optimised Algorithm Finalisation	22/7	27/7
(j) Theoretical Analysis	25/7	27/7
(6) Analysis of Algorithms	27/7	10/8
(a) Finalise Implementations	27/7	1/8
(b) Extend Benchmark Testbed	30/7	1/8
(c) Data-set Generation for Experiments	30/7	1/8
(d) Performance Testing and Data Collection	1/8	4/8
(e) Compilation and Analysis of Results	4/8	10/8
(7) Software Feasibility Demonstration	20/7	25/7
(8) Final Paper	26/7	2/9
(a) Scaffold	26/7	29/7
(b) Draft	29/7	23/8
(c) Final	23/8	2/9
(9) Final Project Demonstration	5/9	19/9
(10) Project Poster	19/9	3/10
(11) Project Webpage	3/10	10/10

Table 2: Milestones and Tasks

9.5 Timeline

We have included a Gantt chart to illustrate the project milestones and task in Appendix B as Figure 4.

9.6 Work Allocation

Both project partners will work together during the *Preliminary Work* phase of the project in order to develop the overall model-based approach and foundational ideas that will inform the design of the Rational and Lexicographic Closure algorithms. After which, Jaron Cohen will work on Rational Closure and Carl Combrinck will work on Lexicographic Closure for Task (4). Both partners will collaborate on Tasks (5)(a)-(c), but will complete the remainder of Task (5) with respect to one's assigned form of entailment. Task (6)(b) will be collaborative, but the remaining parts of Task (6) will be completed individually. All tasks from (7) onwards, except for Task (8) (the final paper) will be done collaboratively.

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] 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

[5] 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

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

[7] 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

[8] Daniel Lehmann. 1999. Another perspective on Default Reasoning. *Annals of Mathematics and Artificial Intelligence* 15 (11 1999). https://doi.org/10.1007/BF01535841

[9] 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

[10] Raymond Reiter. 1980. A logic for default reasoning. *Artificial Intelligence* 13, 1 (1980), 81–132. https://doi.org/10.1016/0004-3702(80)90014-4 Special Issue on Non-Monotonic Logic.

Appendix A RISKS APPENDICES

ID	Risk	Probability	Impact
1	Supervisor is unavailable	Low	Medium
2	Project partner leaves	Low	Medium
3	Conflict/disagreement in terms of algorithm design	Low	Low
4	Poor time management	Medium	High
5	Scope creep during the implementation phase or attempted generalisation of algorithms	Medium	Medium
6	Difficulty optimising developed algorithms	Medium	Medium
7	Team member falls ill (e.g COVID-19) or is unavailable during the collaborative phase of algorithm development	Medium	Medium
8	Loss or corruption of source code	Low	High
9	Difficulty integrating code with tools developed for SCADR project	Medium	Medium
10	Initial scope encompassing theoretical algorithm development, implementation and testing proves to be too large	Low	Medium

Table 3: Risk Identification

ID	Mitigation	Monitoring	Management
1	Check the supervisor's availability in advance and establish a meeting schedule.	Maintain regular contact with the supervisor, confirming upcoming meetings and rescheduling if necessary.	Continue working on aspects of the project that do not need immediate feedback.
2	Ensure an open dialogue exists between project partners and project supervisor.	Maintain regular assessments of mental and physical health.	Reassess project scope and deliverables to ensure that all core project deliverables and outcomes are achieved.
3	Ensure communication is consistent. Each member must voice any issues as they arise.	Confirm that both partners are satisfied with any decisions that are made prior to moving forward.	Request assistance from supervisor to resolve dispute.
4	Implement the "deep work" technique and utilise the pomodoro method.	Regularly assess our progress against the project timeline and Gantt chart.	Adjust project timeline where possible utilising any float/slack time available.
5	Focus on our research questions and core deliverables found in the project timeline.	Record all instances of scope creep and ensure that the agreed- upon scope is not modified without consultation	Halt work on non-core scope components and redistribute time to essential components.
6	Review material on optimisation methods.	Run a consistent performance test between optimisation changes to assess improvement.	Assess the unoptimised implementation of model-based algorithms against pre-existing algorithms.
7	Reduce social contact and adhere to any social distancing guidelines in place.	Be cognisant of potential symptoms.	Distribute short-term work to project partner for the duration of recovery.
8	Regularly back-up work and make use of cloud services to limit the risk of personal device malfunctions.	Check that back-ups are up to date and monitor load-shedding schedules.	Restore work to last available backup and continue working.
9	Familiarise ourselves with the source code and experiment with its capabilities in the vacation.	Confirm ability to implement all necessary features for later comparison in terms of results.	Contact authors for clarification.
10	Ensure layers of scope are well-defined, utilising "onion approach".	Regularly assess our progress against the project timeline and Gantt chart.	Reassess remaining scope components and remove "nice to have" elements if necessary.

Table 4: Risk Mitigation, Monitoring, and Management

Appendix B GANTT CHART

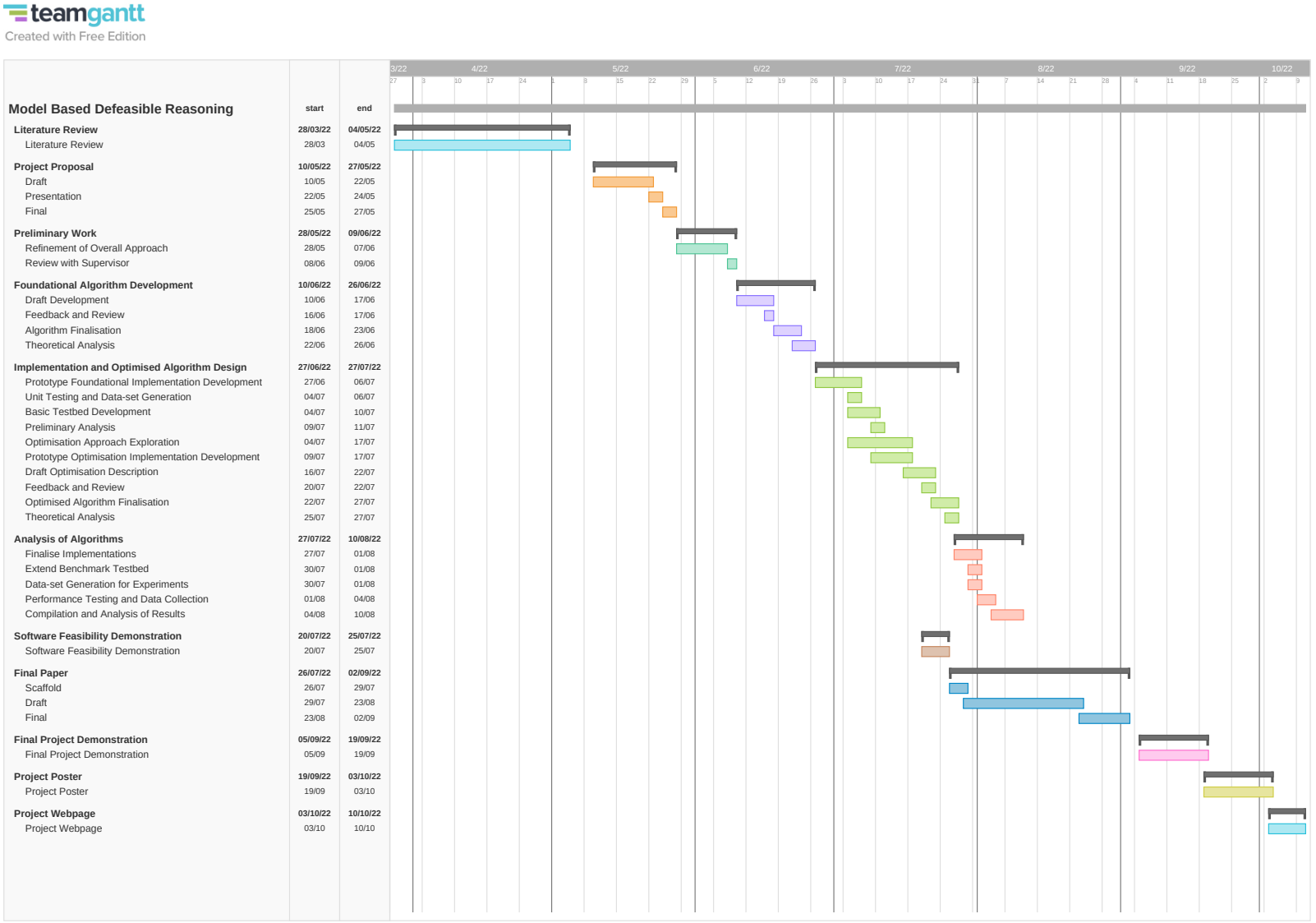

Figure 4: Gantt Chart