UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Final Paper 2021

Title: Propositional Defeasible Explanation

Author: Emily Morris

Project Abbreviation: PDE

Supervisor(s): Professor Tommie Meyer

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	25
Experiment Design and Execution	*0*	*20*	0
System Development and Implementation	*0*	*20*	0
Results, Findings and Conclusions	*10*	*20*	20
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*	0
Total marks		**80**	

Propositional Defeasible Explanation

Emily Morris
mrremi007@myuct.ac.za
Department of Computer Science
University of Cape Town

ABSTRACT

Explanations provide a way of showing why an entailment holds in classical logics and are a crucial aspect of reasoning systems. However, they have not yet been explored in detail for forms of defeasible reasoning. The KLM framework presents a way of defining defeasible reasoning by providing a set of properties it should adhere to. We explore defining and computing defeasible explanations in the context of the KLM framework for propositional logic, focusing on two well-known forms of KLM defeasible entailment: Rational Closure and Lexicographic Closure. We utilize a previously defined algorithm adaption approach to define defeasible justifications for Lexicographic Closure and apply a general definition for defeasible explanation, termed strong explanation, to Rational Closure. We conclude by proposing an general definition for defeasible explanation in the context of an extension of the KLM framework.

CCS CONCEPTS

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

KEYWORDS

Artificial Intelligence, Knowledge Representation and Reasoning, Propositional Logic, Justifications, Explanations, Defeasible Reasoning, KLM Framework

1 INTRODUCTION

Knowledge Representation and Reasoning is a field within Artificial Intelligence in which information is modeled using formal logic, allowing one to apply a set of rules and manipulations related to a form of reasoning [2]. There are many different forms of logic, with different levels of expressiveness. Classical logics are unable to express the idea of *typicality*. This makes it difficult to represent exceptional knowledge in a succinct way. To see this, consider the following example.

Example 1.1. Suppose we know the following information:

(1) Birds fly
(2) Penguins are birds

From these statements we can conclude that 'penguins fly'. Now suppose we add a new statement 'penguins do not fly' i.e. that penguins are an exceptional type of bird that do not fly. We can now infer that penguins both fly and do not fly which removes our ability to reason about penguins altogether as they can no longer exist.

For this example to work, we should instead phrase the first fact as 'birds typically fly'. This captures an idea of uncertainty that allows us to later retract our conclusions if we learn information that contradicts them. This form of reasoning is referred to as *defeasible reasoning*. In this paper we focus on a particular approach to defining defeasible reasoning known as the KLM approach [10], using propositional logic as a basis.

An important aspect of reasoning is being able to infer new knowledge from existing information. In addition to being able to infer new knowledge, it is often useful to know why certain information is being inferred. In formal logic, this is given by explanations. Explanations are an integral part of reasoning services that support knowledge base comprehension and thus are a crucial aspect of using reasoning systems practically.

While explanations have been well explored for classical logics, little work has been done on extending this to forms of defeasible reasoning. In this paper, we explore and define defeasible explanation within the context of the KLM framework, focusing on two forms of defeasible entailment: Rational Closure and Lexicographic Closure. Sections 2 and 3 provide the relevant background for this paper. In Section 4, we extend the Lexicographic Closure algorithm to compute a version of a defeasible justification. In Sections 5 and 6 we explore more general definitions for defeasible explanation and apply these to Rational Closure.

2 BACKGROUND

2.1 Propositional Logic

The basis for propositional logic [1] is given by what are known as *propositional atoms*. A propositional atom represents a basic statement that can be assigned a value of true or false. We use $\bot$ to denote an atom that is always false and $\top$ to denote an atom that is always true. We can recursively build more complicated statements, called *propositional formulas*, from atoms using boolean operators such as $\wedge, \vee, \rightarrow$ and $\neg$. We use Greek letters $\alpha, \beta, \ldots$ to denote formulas. Formally, $\alpha := \top \mid \bot \mid p \mid \neg\alpha \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \alpha \rightarrow \alpha$. The semantics for the operators in the last four combinations are intuitively analogous to the natural language statements: '*not* α', 'α *and* α', 'α *or* α' and 'α *implies* α'. For statements of the form $\alpha \rightarrow \beta$, we refer to α as the antecedent.

We assign meaning to atoms using *interpretations*. An interpretation is a function that assigns a truth value to each atom; in this way it presents a specific version of the world represented by the atoms. If a formula α evaluates to true according to the truth values and semantics of the operators for an interpretation $\mathcal{I}$, we call $\mathcal{I}$ a *model* of α.

A finite set of propositional formulas is called a *knowledge base*. An interpretation is a model of a knowledge base $\mathcal{K}$ if it is a model of all the formulas in $\mathcal{K}$. We say that $\mathcal{K}$ entails a statement α, denoted $\mathcal{K} \models \alpha$, if every model of $\mathcal{K}$ is a model of α. This gives us the basis for a simple reasoning system which we demonstrate with the following example.

Example 2.1. The statements in Example 1.1 have the atoms 'bird' (b), 'penguin' (p): and 'fly' (f) and can be expressed as formulas, yielding the knowledge base $\mathcal{K}$:

(1) Birds fly ($b \rightarrow f$)
(2) Penguins are birds ($p \rightarrow b$)

Then to see if we can conclude that 'penguins fly' ($p \rightarrow f$), we check whether $\mathcal{K} \models p \rightarrow f$.

2.2 Classical Explanation

In reasoning systems, *explanations* tell us which statements in a knowledge base are relevant to the entailment between a knowledge base and an entailed statement [5]. Justifications are a well explored approach to explanations for classical logics. A *justification* for the entailment of a formula α is a minimal subset of a knowledge base that entails α. Note one formula may have many justifications in a knowledge base. Justifications provide a simple and intuitive way of summarizing why an entailment holds by showing the exact formulas in the knowledge base which are responsible for the entailment. The concept of a justification is best illustrated with an example.

Example 2.2. Suppose one has a knowledge base $\mathcal{K}$ containing the following information:

(1) Birds have feet ($b \rightarrow f$)
(2) Penguins are birds ($p \rightarrow b$)
(3) Birds have wings ($b \rightarrow w$)

Then $\mathcal{K} \models p \rightarrow f$, with the justification $\mathcal{J} = \{b \rightarrow f, p \rightarrow b\}$, since $\{b \rightarrow f, p \rightarrow b\} \models p \rightarrow f$ and if any statements are removed, the entailment no longer holds.

A variety of algorithms have been developed for computing classical justifications [7]. These algorithms can be classified as either glass-box or black-box algorithms; in this paper we utilize a black-box algorithm for computing all justifications for a classical knowledge base. In black-box algorithms, justifications are computed independently of the underlying reasoning process. This means they are not restricted to a single form of logic [9], which allows us to use this algorithm with propositional logic.

2.3 KLM Defeasible Reasoning

Although there are many approaches to defeasible reasoning, one approach that has been studied extensively in the literature is that proposed by Kraus, Lehmann and Magidor (KLM) [10]. This approach extends propositional logic, introducing a defeasible implication operator $\mid\sim$ which can be viewed as a defeasible analogue of $\rightarrow$. Defeasible implications (DI) of the form $\alpha \mid\sim \beta$, are read as '*α typically implies β*'. So for Example 1.1, we would express 'birds typically fly' as $b \mid\sim f$.

We must then define how entailment functions for knowledge bases that contain defeasible implications. Note we assume that knowledge bases only contain defeasible implications but can express any classical formula α using the defeasible representation $\neg\alpha \mid\sim \bot$ [4]. KLM propose restrictions on the definitions for defeasible entailment by giving a set of properties, known as the KLM properties [10, 12], that they should adhere to. Thus, KLM does not define a single notion of defeasible entailment but rather defines a class of defeasible entailment relations that have some interesting theoretical and computational properties [8]. We refer to definitions for defeasible entailment that satisfy the KLM properties as *LM-rational.* We denote defeasible entailment as $\mathcal{K} \approx \alpha \mid\sim \beta$ and use subscripts to denote a particular definition.

2.4 Rational Closure

Rational Closure (RC) is an LM-rational definition for defeasible reasoning, proposed by Lehmann and Magidor [12]. Rational Closure can be defined semantically, using what are known as *ranked interpretations*, as well as algorithmically. For the purposes of this paper, we present only the algorithmic definition.

Casini et al. [4] present an algorithm for Rational Closure which has two distinct phases. The first, given by BaseRank shown in Algorithm 1, creates a ranking of statements. This is done by defining a series of subsets $E_0, E_1, \ldots, E_n, E_\infty$. Our initial subset E_0 is defined to contain the classical version of each defeasible implication in $\mathcal{K}$ i.e. $E_0 = \{\alpha \rightarrow \beta \mid \alpha \mid\sim \beta \in \mathcal{K}\}$. Each subsequent subset E_i is then defined to contain all the statements in the previous subset E_{i-1} for which we can 'disprove' the antecedent using statements in E_{i-1}. We denote the first subset for which $E_{i-1} = E_i$ as E_∞. The differences between these subsets are then used to define a series of disjoint base ranks (or levels) $R_0, R_1, \ldots, R_\infty$. These ranks represent a specificity ranking for $\mathcal{K}$ where statements in R_i are 'more specific' than statements in R_{i-1}. Note that R_∞ contains all classical statements in defeasible form. We use $br_{\mathcal{K}}(\alpha)$ to denote the base rank of a formula in the knowledge base $\mathcal{K}$ and define $br_{\mathcal{K}}(\alpha \mid\sim \beta) = br_{\mathcal{K}}(\alpha)$.

Algorithm 1: BaseRank

```
Input: A knowledge base $\mathcal{K}$
Output: An ordered tuple ($\mathcal{R}_0, ..., \mathcal{R}_{n-1}, \mathcal{R}_\infty$, n)
i := 0;
$E_0 := \overrightarrow{\mathcal{K}}$;
repeat
    $E_{i+1} := \{\alpha \rightarrow \beta \in E_i \mid E_i \models \neg\alpha\}$;
    $\mathcal{R}_i := E_i \setminus E_{i+1}$;
    i := i+1;
until $E_{i-1} = E_i$;
$\mathcal{R}_\infty := E_{i-1}$;
n := i-1;
return n;
```

We demonstrate how BaseRank is used to create a ranking for a knowledge base $\mathcal{K}$ by means of an example.

Example 2.3. Suppose one has the following defeasible knowledge base $\mathcal{K}$:

(1) Birds typically fly ($b \mid\sim f$)
(2) Penguins typically do not fly ($p \mid\sim \neg f$)
(3) Penguins typically swim ($p \mid\sim s$)
(4) Penguins are birds ($p \rightarrow b$)
(5) Max is a penguin ($m \rightarrow p$)

Intuitively, since penguins are a specific type of bird, we would expect statements about penguins to rank higher than those about birds. The resulting ranking after applying BaseRank is shown in

Figure 1. As an example of the base rank notation, note that from this ranking we can see $br_{\mathcal{K}}(b) = br_{\mathcal{K}}(b \mid\sim f) = 0$.

0	$b \mid\sim f$
1	$p \mid\sim \neg f, p \mid\sim s$
∞	$p \rightarrow b, m \rightarrow p$

Figure 1: Base ranking of statements for Example 2.3

The second phase of the Rational Closure algorithm, given by `RationalClosure` shown in Algorithm 2, uses the ranking produced by `BaseRank` to compute the entailment of a defeasible implication $\alpha \mid\sim \beta$. To do this, we check whether we can 'disprove' the antecedent α. If we can, we remove the most general information from our knowledge base; the information in the lowest rank. We repeat this process until either we can no longer 'disprove' the antecedent, or only the infinite rank remains. At this point we compute the final defeasible entailment result using the remaining ranks, the classical version of our defeasible implication, $\alpha \rightarrow \beta$, and classical entailment.

```
Algorithm 2: RationalClosure
  Input: A knowledge base $K$ and a DI $\alpha \mid\sim \beta$
  Output: true, if $K \approx \alpha \mid\sim \beta$, otherwise false
1 $(R_0, R_1, ..., R_\infty$, n) := BaseRank($K$);
2 i := 0;
3 $R := \bigcup_{i=0}^{j<n} R_j$;
4 while $R_\infty \cup R \models \neg\alpha$ and $R \neq \emptyset$ do
5     $R := R \setminus R_i$;
6     i := i+1;
7 end
8 return $R_\infty \cup R \models \alpha \rightarrow \beta$;
```

Example 2.4. Suppose one has the knowledge base in Example 2.3 and consider the entailment of the statement 'Max typically does not fly' ($m \mid\sim \neg f$). Using `RationalClosure`, `BaseRank` is first used to compute the ranking in Figure 1. Then we start by considering all the ranks and check whether $R_0 \cup R_1 \cup R_\infty \models \neg m$. Since this holds, R_0 is removed. Then we check whether $R_1 \cup R_\infty \models \neg m$. Since this entailment does not hold, we stop removing ranks and check whether $R_1 \cup R_\infty \models m \rightarrow \neg f$. Since this entailment holds, `RationalClosure` will return **true**.

0	~~$b \mid\sim f$~~
1	$p \mid\sim \neg f, p \mid\sim s$
∞	$p \rightarrow b, m \rightarrow p$

Figure 2: Top level of ranking is removed in Example 2.4

3 LEXICOGRAPHIC CLOSURE

As a continuation of Background, in this section we present a description of Lexicographic Closure (LC), a more permissive LM-rational definition for defeasible entailment proposed by Lehmann [11]. Rational Closure is the most conservative form of KLM defeasible entailment. Removing entire ranks can lead to statements that are not related to 'disproving' the antecedent also being removed. Generally, only some statements in the rank are responsible for 'disproving' the antecedent. Furthermore, if there are multiple statements within the rank responsible for 'disproving' the antecedent, sometimes only one of these statements needs to be removed. In Lexicographic Closure, we remove single statements instead of entire ranks. Like Rational Closure, Lexicographic Closure can be defined both semantically and algorithmically; we once again present the algorithmic definition.

3.1 Sub-Knowledge Base Approach

The Lexicographic Closure algorithm [13] can be seen as a refinement of the Rational Closure algorithm, where we either refine the ranking or the removal process. For the purposes of this paper, we rank statements in the same manner as in Rational Closure but refine the removal of statements. Instead of removing the entire rank when we can 'disprove' the antecedent, we first try removing statements within the rank with the aim being to remove exactly the statements responsible for 'disproving' the antecedent. Since we do not initially know which statements these might be and all the statements have the same rank, there is no obvious way to choose which statements to remove or in what order to do so. Instead, we non-deterministically remove statements from the level, considering all possible ways of removing statements simultaneously, until either we remove the statements that allow us to 'disprove' the antecedent or the level is empty.

We can do this by simply creating sub-knowledge bases by considering all subsets of the rank R_i of a fixed size and then combining each of these with the remaining ranks $R_{i+1}, \ldots, R_\infty$. If n is the number of statements in a rank, then if we consider all subsets of size $n - x$ we are considering all ways of removing x statements from the rank. We then check if we can 'disprove' the antecedent in these sub-knowledge bases. We start with $x = 1$, retaining as much information as possible. If we can 'disprove' the antecedent in all the sub-knowledge bases, each sub-knowledge base still contains the statements required to 'disprove' the antecedent and thus we need to remove more statements. We repeat this process, incrementing x by 1 each time until either we find an x for which some sub-knowledge base does not 'disprove' the antecedent, or $x = n$ in which case we remove the level and repeat the process with the next level.

Example 3.1. Suppose one has the following defeasible knowledge base $\mathcal{K}$:

(1) Birds typically have wings ($b \mid\sim w$)
(2) Birds typically have four toes ($b \mid\sim 4t$)
(3) Birds typically have eyes ($b \mid\sim e$)
(4) Penguins typically do not have both wings and four toes ($p \mid\sim \neg(w \wedge 4t)$)
(5) Penguins are birds ($p \rightarrow b$)

If `BaseRank` is applied, we achieve the ranking shown in Figure 3.

Consider the entailment of the statement, 'penguins typically have eyes' ($p \mid\sim e$). Taking the ranked knowledge base, we check whether we can 'disprove' p. Since we can, instead of removing R_0, we consider all ways of removing 1 statement from R_0. This yields the sub-knowledge bases shown in Figure 4 (we show R_1 and R_∞ once to emphasize their inclusion in all sub-knowledge

0	$b \mid\!\sim w, b \mid\!\sim 4t, b \mid\!\sim e$
1	$p \mid\!\sim \neg(w \wedge 4t)$
∞	$p \to b$

Figure 3: Base ranking of statements for Example 3.1

bases). Since we can no longer 'disprove' p in the second and third sub-knowledge bases, we stop removing statements.

0	$b \mid\!\sim w, b \mid\!\sim 4t$

0	$b \mid\!\sim w, b \mid\!\sim e$

0	$b \mid\!\sim 4t, b \mid\!\sim e$

1	$p \mid\!\sim \neg(w \wedge 4t)$
∞	$p \to b$

Figure 4: Sub-knowledge bases for Example 3.1

As in Rational Closure, when we can no longer 'disprove' the antecedent, we compute the final entailment result using classical entailment. However, since we are now working with a collection of sub-knowledge bases we need to define how we do this. We define the final entailment result to be true if it is true in *all* of the sub-knowledge bases.

3.2 Combined Formula Approach

We can perform the checking of all sub-knowledge bases for a fixed subset size in a single step by combining the various statements in these sub-knowledge bases in an appropriate way. The method for formula combination if based on the following two facts:

(1) $\mathcal{K} \cup \{\alpha \vee \beta\} \models \gamma$ iff $\mathcal{K} \cup \{\alpha\} \models \gamma$ *and* $\mathcal{K} \cup \{\beta\} \models \gamma$
(2) $\mathcal{K} \cup \{\alpha \wedge \beta\} \models \gamma$ iff $\mathcal{K} \cup \{\alpha, \beta\} \models \gamma$

The first fact tells us that if we combine the formulas that differ between knowledge bases using the 'or' binary connective, we can compute exactly those statements entailed by all the knowledge bases. The second fact tells us that, in the context of classical entailment, we can treat multiple statements within a knowledge base as a single statement created by combining the individual statements using the 'and' binary connective. Thus we can combine the multiple sub-knowledge base checks as follows:

(1) When considering a subset $\{\alpha_1, \alpha_2, \ldots, \alpha_y\}$ of a rank R_i of size y, instead consider $S = \alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_y$.
(2) When considering all possible subsets $S_1, S_2, \ldots, S_z$ of R_i of size y separately, instead consider $S_1 \vee S_2 \vee \ldots \vee S_z$.

Since the rest of the knowledge base ranks are included in all the sub-knowledge bases, we simply replace the rank R_i we are considering with the combined formula we create. This combined formula approach provides a way of presenting a succinct algorithmic definition, given by `LexicographicClosure` shown in Algorithm 3.

As a full demonstration of `LexicographicClosure`, consider the following example.

Example 3.2. Suppose one has the knowledge base $\mathcal{K}$ presented in Example 3.1 and consider once again the entailment of the statement $p \mid\!\sim e$. Using `LexicographicClosure`, we start by ranking $\mathcal{K}$ using `BaseRank` to achieve the ranking in Figure 3. We then check whether $R_0 \cup R_1 \cup R_\infty \models \neg p$. Since this holds, we consider all ways of removing 1 formula from R_0, replacing R_0 with the appropriate combined formula as shown in Figure 5 which is denoted as $R_{0,2}$. Now we check whether $R_{0,2} \cup R_1 \cup R_\infty \models \neg p$. This does not hold, so we stop removing statements and check whether $R_{0,2} \cup R_1 \cup R_\infty \models p \to e$. Since this entailment holds, `LexicographicClosure` returns **true**.

Algorithm 3: LexicographicClosure

```
Input: A knowledge base $\mathcal{K}$ and DI $\alpha \mid\!\sim \beta$
Output: true, if $K \mid\!\approx_{LC} \alpha \mid\!\sim \beta$, otherwise false
1  $(\mathcal{R}_0, ..., \mathcal{R}_{n-1}, \mathcal{R}_\infty$, n) := BaseRank($\mathcal{K}$);
2  i := 0;
3  $\mathcal{R} := \bigcup_{j=0}^{j<n} \mathcal{R}_j$;
4  while $\mathcal{R}_\infty \cup \mathcal{R} \models \neg\alpha$ and $\mathcal{R} \neq \emptyset$ do
5      $\mathcal{R} := \mathcal{R} \setminus \mathcal{R}_i$;
6      m := | $\mathcal{R}_i$ |-1;
7      $\mathcal{R}_{i,m} := \bigvee_{X \in \text{Subsets}(\mathcal{R}_i, m)} \bigwedge_{x \in X} x$;
8      while $\mathcal{R}_\infty \cup \mathcal{R} \cup \{\mathcal{R}_{i,m}\} \models \neg\alpha$ and $m > 0$ do
9          m := m-1;
10         $\mathcal{R}_{i,m} := \bigvee_{X \in \text{Subsets}(\mathcal{R}_i, m)} \bigwedge_{x \in X} x$;
11     end
12     $\mathcal{R} := \mathcal{R} \cup \{\mathcal{R}_{i,m}\}$;
13     i := i+1;
14 end
15 return $R_\infty \cup R \models \alpha \to \beta$;
```

0	~~$b \mid\!\sim w, b \mid\!\sim 4t, b \mid\!\sim e$~~
0, 2	$(b \mid\!\sim w \wedge b \mid\!\sim 4t) \vee (b \mid\!\sim w \wedge b \mid\!\sim e) \vee (b \mid\!\sim 4t \wedge b \mid\!\sim e)$
1	$p \mid\!\sim \neg(w \wedge 4t)$
∞	$p \to b$

Figure 5: Top level of ranking is replaced for Example 3.2

4 WEAK JUSTIFICATIONS FOR LEXICOGRAPHIC CLOSURE

We begin our work in this paper by considering an adaption of the Lexicographic Closure algorithm to allow for the computation of a form of defeasible justification. While justifications and their computation are clearly defined for classical logics, little work has been done on defining and computing justifications for forms of defeasible reasoning. The adaption approach we follow in this section is proposed by Chama [5] as a method for computing and defining defeasible justifications for Rational Closure. We refer to justifications produced in this algorithm adaption approach as *weak justifications.*

4.1 Motivation for the Approach

Whether a statement is defeasibly entailed by a knowledge base when using Lexicographic or Rational Closure is determined by whether it is classically entailed by the remaining ranks after the required removal of statements. It thus makes sense to provide classical justifications from these remaining ranks as our overall defeasible justifications.

We demonstrated in Section 3 how the Lexicographic Closure algorithm can be viewed as performing classical entailment checks in a series of sub-knowledge bases. For the final entailment to hold, the classical entailment of the classical version of the defeasible implication must hold in each sub-knowledge base. Thus there must exist classical justifications for the classical entailment in each of these sub-knowledge bases. These are the justifications we wish to compute as our defeasible justifications.

When computing these justifications, it is more appropriate to use the idea of subsets and sub-knowledge bases rather than our combined formula representation. This allows us to deal with individual formulas and pinpoint exactly those required for the entailment instead of a larger group of formulas.

4.2 Algorithm

The basic idea behind the algorithm we present is to reconstruct the sub-knowledge bases used in the final entailment computation, and then apply an algorithm for computing all justification for classical logics to each of these sub-knowledge bases. To do this, we first modify `LexicographicClosure` to create `LexicographicClosureForJustifications` shown in Algorithm 7 in Appendix A. This modification is done by introducing a new variable *m*, which represents the subset size that is used in the final classical entailment computation, and returning *m* and *i*, the lowest complete level used in the computation, as an ordered pair along with the final defeasible entailment result. This provides us with all the information required to reconstruct the final sub-knowledge bases.

Example 4.1. Given the knowledge base and query in Example 3.2, `LexicographicClosureForJustifications` returns **true**, along with the ordered pair (1, 2).

Since we require our entailment to hold in all sub-knowledge bases, it does not necessarily make sense to present a single set of statements as a final justification. If we computed and combined our various sets of justifications, we could lose information on the structure of the sub-knowledge bases and thus only provide partial information as to why the final entailment holds. For example, some sub-knowledge bases may contain justifications that are not contained in other sub-knowledge bases, and thus providing one of these justifications as the final defeasible justification would be incorrect. We instead want to retain a structure that refers to which statements are responsible for the entailment in which sub-knowledge bases. Thus we present a tuple of justifications, where the *i*'th element in the tuple is a justification for the entailment in the *i*'th sub-knowledge base.

If we are working with a complete level i.e. $m = 0$, we have a single knowledge base and thus compute a set of 1-tuples. This is simply the set of classical justifications in the remaining full levels and is the set that would have been computed by the adapted algorithm for Rational Closure. Note that if we do not require a level of structure we can extract single justifications from the tuples, but it would be difficult to produce these tuples just given a set of justifications. We present `ComputeDefeasibleJustifications`, given by Algorithm 4, as a formal algorithm for computing defeasible justifications for Lexicographic Closure.

Algorithm 4: ComputeDefeasibleJustifications

```
Input: A knowledge base $\mathcal{K}$ and DI $\alpha \mid\sim \beta$
Output: Set of all justifications for $K \mid\approx_{LC} \alpha \mid\sim \beta$
$(\mathcal{R}_0, ..., \mathcal{R}_{n-1}, \mathcal{R}_\infty, n) := BaseRank(\mathcal{K})$;
$(i, m) := LexicographicClosureForJustifications(\mathcal{K}, \alpha \mid\sim \beta)$;
$\mathcal{R} := \bigcup_{j=i}^{j<n} \mathcal{R}_j$;
$\mathcal{J} := \varnothing$;
if $m > 0$ then
    subsets := $Subsets(R_{i-1}, m)$;
    x := 0;
    for $S$ in subsets do
        $\mathcal{F} := S \cup \mathcal{R} \cup \mathcal{R}_\infty$;
        $\mathcal{J}_i := ComputeAllJustifications(\mathcal{F}, \alpha \rightarrow \beta)$;
        x := x+1;
    end
    $\mathcal{J} := \mathcal{J}_0 \times \mathcal{J}_1 \times \ldots \times \mathcal{J}_{x-1}$;
else
    $\mathcal{J} := ComputeAllJustifications(\mathcal{R} \cup \mathcal{R}_\infty)$;
return $\mathcal{J}$;
```

Example 4.2. Suppose one has the knowledge base from Example 3.2. Consider the defeasible justifications for the entailment of the statement 'penguins typically have eyes' ($p \mid\sim e$). This entailment has the sub-knowledge bases shown in Figure 4. Thus the justifications for the entailment in each sub-knowledge base are as follows:

$$\mathcal{J}_0 = \{\{b \mid\sim w, p \mid\sim \neg(w \wedge 4t), b \mid\sim 4t, p \rightarrow b\}\}$$
$$\mathcal{J}_1 = \{\{p \rightarrow b, b \mid\sim e\}\}$$
$$\mathcal{J}_2 = \{\{p \rightarrow b, b \mid\sim e\}\}$$

The set of all justifications is then the cross product of these sets. Therefor there is a single defeasible justification:

$$(\{b \mid\sim w, p \mid\sim \neg(w \wedge 4t), b \mid\sim 4t, p \rightarrow b\},$$
$$\{p \rightarrow b, b \mid\sim e\}, \{p \rightarrow b, b \mid\sim e\})$$

This example also serves to illustrate that, unlike in Rational Closure, we might still be able to 'disprove' our antecedent α in some of the sub-knowledge bases we reconstruct. Thus some of our justifications for the final entailment may include justifications for $\neg\alpha$. For example, the justification in $\mathcal{J}_0$ in Example 4.2 is a justification for $\neg p$. In a justification for $\neg\alpha$, we have essentially lost our ability to reason about α and thus these justifications are more informative about the sub-knowledge base structure than the actual formula entailment.

While it is important to maintain the sub-knowledge base structure in our defeasible justifications, this approach is only effective when there are justifications that contain statements in the subset rank level. Otherwise, if all the justifications are contained in higher ranks, our tuple may have the same justification for each element and thus will have the same level of description as a single set. To potentially avoid this, we can use the combined formula approach along with the algorithm for defeasible justifications for Rational Closure to verify whether information in the subset rank is a part of the final entailment, before using this more refined algorithm.

5 STRONG JUSTIFICATIONS FOR RATIONAL CLOSURE

While weak justifications present an intuitive and simple approach to defining defeasible explanation, their level of description is limited. We are only presenting information as to why the final classical entailment holds and are thus disregarding the rest of the reasoning process. In this section, we apply a more comprehensive definition for defeasible explanation, referred to as strong explanation, proposed by Brewka et al. [3] to KLM style reasoning, to produce what we refer to as *strong justifications*. In particular, we look at defining what constitutes a strong justification for Rational Closure and explore an algorithm for computing these strong justifications.

5.1 Motivation

One of the reasons why justifications work well in classical reasoning is due to the property of monotonicity, which means that anything entailed by a justification is entailed by the knowledge base. However, in the defeasible case, the rest of the knowledge base might contain information contradictory to the information in our justification. Brewka et al. [3] address this by extending the definition of a justification to include an extra property that states that if we add any statements to our justification, the justification should still defeasibly entail our statement.

This strengthening of the classical justification definition presents an interesting approach to defining defeasible explanation by viewing it as extension of a well established form of classical explanation. Thus, since this definition is proposed as a general definition for all forms of defeasible reasoning, we wish to assess the appropriateness of its application as a definition for defeasible explanation within the KLM framework. In this initial assessment we restrict ourselves to just considering Rational Closure due to its level of simplicity.

5.2 Overview of the Approach

Since weak justifications are a subset of our knowledge base that entail our statement, they seem to present a good basis for defining strong justifications. However, not all weak justifications fulfil the extended criteria of strong justifications. As an example of such a weak justification, consider the following defeasible entailment.

Example 5.1. Suppose one has a knowledge base $\mathcal{K}$ containing the following information:

(1) If something walks it typically does not fly ($w \mid\sim \neg f$)
(2) Pigeons typically fly ($p \mid\sim f$)
(3) Pigeons typically walk ($p \mid\sim w$)

This knowledge base has the ranking shown in Figure 6. Consider the entailment of the statement 'if something is a pigeon and it walks then it typically flies' ($p \wedge w \mid\sim f$). The weak justification for this statement is $\mathcal{W} = \{p \mid\sim f\}$.

0	$w \mid\sim \neg f$
1	$p \mid\sim w, p \mid\sim f$
∞	

Figure 6: Base ranking of statements for Example 5.1

Now consider what happens when the statement $w \mid\sim \neg f$ is added to $\mathcal{W}$. Ranking $\mathcal{W} \cup \{w \mid\sim \neg f\}$ yields the ranking shown in Figure 7. However, now when computing the entailment of $p \wedge w \mid\sim f$, $R_0 \vDash \neg(p \wedge w)$ and so R_0 is removed. But then $\varnothing \vDash (p \wedge w) \rightarrow f$ is computed as the final entailment result, which does not hold. Thus $\mathcal{W}$ is not a strong justification since $\mathcal{W} \cup \{w \mid\sim \neg f\}$ does not entail our statement.

0	$w \mid\sim \neg f, p \mid\sim f$
∞	

Figure 7: Base ranking of subset for Example 5.1

The issue that arises in Example 5.1 is that we can add information to our weak justification that allows us to 'disprove' the antecedent α of our entailed statement but not 'disprove' all the antecedents of our statements in our weak justification. This leads to some of the statements in our weak justification getting 'mixed into' lower ranks in which we can still 'disprove' α. These statements are then removed during the process of the algorithm, causing our final entailment computation to return false since $\mathcal{W}$ contains exactly those statements required to ensure the final entailment holds.

However, we can take our weak justification $\mathcal{W}$ in Example 5.1 and extend it to create the set $\mathcal{S} = \{p \mid\sim f, p \mid\sim w\}$ which is a strong justification. In $\mathcal{S}$, we have addressed our previous issue by adding statements to $\mathcal{W}$ to ensure that whenever we can 'disprove' the antecedent of our entailed statement α we can also 'disprove' all the antecedents of the statements in $\mathcal{W}$. Essentially, to create a strong justification from a weak justification $\mathcal{W}$, we are looking to extend $\mathcal{W}$ to create a set $\mathcal{S}$ such that, no matter what statements we add to $\mathcal{S}$ from our original knowledge base $\mathcal{K}$, the following partition is maintained when we rank our statements.

0		can 'disprove' α in these ranks
$\vdots$		
x		
$x+1$		cannot 'disprove' α and $\mathcal{W}$ is contained in these ranks
$\vdots$		
∞		

We also need to perform this extension in a manner that ensures the final set is minimal and thus adheres to the original justification criterion of minimality.

5.3 Algorithm

To create the partition described in the previous section, we need to look at all the ways of 'disproving' our various antecedents in the context of the ranking of the original knowledge base. We need to ensure a weak justification is always ranked above our ways of 'disproving' α, which are all the justifications for $\neg\alpha$ in $\mathcal{K}$. We denote a justification for $\neg\alpha$ as $\mathcal{J}_{\neg\alpha}$. To do this, we start by first defining the rank of a subset $\mathcal{K}'$ of a knowledge base $\mathcal{K}$ to be the rank of the lowest ranked statement in $\mathcal{K}'$ in the ranking of $\mathcal{K}$. This provides a useful way of defining the ranking of our justifications by classifying them by the minimum rank we have to remove in order for the justification to no longer hold in $\mathcal{K}$.

Example 5.2. Consider the knowledge base $\mathcal{K}$ in Figure 8 and consider the subset $\mathcal{K}' = \{4t \mid\sim \neg f, sp \mid\sim f\}$. The statement $4t \mid\sim \neg f$ has rank 2 in $\mathcal{K}$ and the statement $sp \mid\sim f$ has rank 3 in $\mathcal{K}$. Thus $\mathcal{K}'$ has rank 2 in $\mathcal{K}$.

0	$a \mid\sim \neg w$
1	$b \mid\sim w, b \mid\sim f, b \mid\sim j$
2	$p \mid\sim \neg f, 4t \mid\sim \neg f, 4t \mid\sim b, 4t \mid\sim \neg j$
3	$sp \mid\sim f$
∞	$p \to b, sp \to p, b \to a$

Figure 8: Base ranking of statements for Example 5.2

We restrict ourselves to considering strong justifications for entailments where all justifications $\mathcal{J}_{\neg\alpha}$ and weak justifications have finite rank. This means we can build up a sequences of subsets $(K_0, K_1, \ldots, K_x)$ where:

(1) $br_{K_i}(\gamma) = i$
(2) For all $K' \subset K_i$, $br_{K'}(\gamma) < br_{K_i}(\gamma)$

We create such sequences for our weak justification $\mathcal{W}$ and for all $\mathcal{J}_{\neg\alpha}$ with $\gamma = \alpha$ for $\mathcal{J}_{\neg\alpha}$ and $\gamma = \alpha \wedge \neg\beta$ for $\mathcal{W}$. To create these sequences, we choose an appropriate initial subset to be K_1 (either $\mathcal{W}$ or $\mathcal{J}_{\neg\alpha}$) and then define each successive set K_i to be a minimal set which contains the preceding set K_{i-1} and simultaneously 'disproves' all the antecedents of the statements in K_{i-1}, checking to ensure criteria (1) and (2) are fulfilled by K_i.

Essentially, these sequences represent the minimal ways of 'pushing up' our justifications $\mathcal{J}_{\neg\alpha}$ and our weak justification, where each subsequent set pushes the justification up by one rank. However, while we wish to consider all the possible ways of 'pushing up' our justifications $\mathcal{J}_{\neg\alpha}$, we only wish to consider some of the ways of 'pushing up' the weak justification. These are the ways in which the weak justification is ranked higher than the rank of the highest ranked justification $\mathcal{J}_{\neg\alpha}$ in $\mathcal{K}$ in the final set of the sequence. As an illustration of why we require this, consider the following example.

Example 5.3. Suppose one has the knowledge base in Figure 8 and consider the entailment of the statement $(sp \wedge 4t) \mid\sim f$, which has the weak justification $\mathcal{W} = \{sp \mid\sim f\}$. Now consider the justification $\mathcal{J}_{\neg\alpha} = \{b \mid\sim f, 4t \mid\sim \neg f, 4t \mid\sim b\}$ and two possible ways of 'pushing up' $\mathcal{W}$ and $\mathcal{J}_{\neg\alpha}$, shown in Figure 9.

0	$a \mid\sim \neg w$
1	$b \mid\sim w, sp \mid\sim f$
∞	$sp \to p, b \to a, p \to b$

(a) 'Pushing up' $\mathcal{W}$

0	$a \mid\sim \neg w$
1	$b \mid\sim w, b \mid\sim f$
2	$4t \mid\sim \neg f, 4t \mid\sim b$
∞	$b \to a$

(b) 'Pushing up' $\mathcal{J}_{\neg\alpha}$

Figure 9: Base ranking of statements for Example 5.3

There is no way of adding statements from the ranking in (a) to $\mathcal{W}$ to ensure that $\mathcal{W}$ is 'pushed above' $\mathcal{J}_{\neg\alpha}$ in the ranking shown in (b), since the statements in ranking (a) can give $\mathcal{W}$ a maximum ranking of 1, while $\mathcal{J}_{\neg\alpha}$ also has a ranking of 1.

To ensure we are constructing only the appropriate ways of 'pushing up' our weak justification, we use the E subsets described for `BaseRank` in Section 2.4. By choosing statements from the set E_i, we are ensuring that these statements can be 'pushed up' to a rank of at least i. Thus we initially select statements for simultaneously 'disproving' the antecedents of our statements in $\mathcal{W}$ from the set E_n, where n is the rank of the highest ranked justification $\mathcal{J}_{\neg\alpha}$ in $\mathcal{K}$. We then work backwards through the E sets, continuously choosing the minimum number of statements from E_i required to 'disprove' the statements we previously added from E_{i+1}, until we have a sequence in which the rank of the weak justification in the final set of the sequence is higher than n. This will be the minimum number of statements we need to 'push' $\mathcal{W}$ above any justification $\mathcal{J}_{\neg\alpha}$ in $\mathcal{K}$.

We present `ComputeSubsetSequences`, shown in Algorithm 5, as a formal algorithm for computing the appropriate sequences for weak justifications. This uses the recursive algorithm `Sequences`, shown in Algorithm 6, as a sub-process. Proposition 5.4, which is proven in Appendix B, states that `ComputeSubsetSequences` will always compute an appropriate sequence for some weak justification.

Proposition 5.4. *Let $\mathcal{K}$ be a knowledge base and γ a formula. Provided $br_{\mathcal{K}}(\gamma) \neq \infty$,* `ComputeSubsetSequences` *returns at least one sequence of sets.*

Algorithm 5: ComputeSubsetSequences

```
Input: A knowledge base $\mathcal{K}$, a formula $\alpha$, the rank n
       $\leq br_{\mathcal{K}}(\alpha)$ required for $\alpha$ in the final subset of a
       sequence
Output: Set of all sequences $(K_1, K_2, ..., K_n)$, where $\alpha$ has
        $br_{K_i}(\alpha) = i$, and each $K_i$ is minimal
1 i := 0;
2 $E_0 := \overrightarrow{\mathcal{K}}$;
3 repeat
4     $E_{i+1} := \{\alpha \to \beta \in E_i \mid E_i \models \neg\alpha\}$;
5     i := i+1;
6 until $E_{i-1} = E_i$;
7 $E_\infty := E_{i-1}$;
8 sequences := $Sequences((E_0, \ldots, E_\infty), \alpha, (\emptyset), n, 1)$;
9 return sequences;
```

Note in `Sequences`, *MinimalExtension*(α, A, B) computes the set of all sets M where M is attained by adding the minimum number of statements from A to B so that B entails α and *Minimize*(A, α) returns true if we can remove statements from A and maintain the ranking of α in A. As a demonstration of how `ComputeSubsetSequences` derives a sequence for a weak justification, consider the following example.

Example 5.5. Consider the knowledge base $\mathcal{K}$ in Figure 8 and the weak justification $\mathcal{W} = \{sp \mid\sim f\}$ for the entailment of $(sp \wedge 4t) \mid\sim f$. Since the highest ranked justification for $\neg(sp \wedge 4t)$ in $\mathcal{K}$ has rank 2, $\mathcal{W}$ must have a rank of 3. Thus the algorithm computes the

Algorithm 6: Sequences

```
Input: A knowledge base $\mathcal{K}$= $(E_0, E_1, ...E_\infty)$, a formula $\alpha$,
       the current sequence of subsets $(K_0, \ldots, K_{i-1})$, the
       rank n of $\alpha$ in the final subset of the sequence, the
       index i of the current subset
Output: A set of sequences of length n
if $i > n$ then
    return $\{(K_0, \ldots, K_{i-1})\}$;
$\mathcal{A} := \neg\alpha \wedge (\bigwedge_{\beta \in Antecedents(K_{i-1})} \neg\beta)$;
S := $MinimalExtension(\mathcal{A}, E_{n-i}, K_{i-1})$;
$\mathcal{F} := \varnothing$;
if $S \neq \varnothing$ then
    for $K_i$ in $S$ do
        if $br_{K_i}(\alpha) = i$ and $Minimize(K_i, \alpha)$ is False then
            $\mathcal{F} :=$
              $\mathcal{F} \cup Sequences(\mathcal{K}, \alpha, (K_0, \ldots, K_{i-1}, K_i), n, i+1)$;
        end
    end
return $\mathcal{F}$;
```

sequences of subsets $(K_0, K_1, K_2, K_3, K_4)$ where:

$$K_0 = \varnothing$$
$$K_1 = \{sp \mathrel{|\!\sim} f\}$$
$$K_2 = K_1 \cup \{p \mathrel{|\!\sim} \neg f, sp \rightarrow p\}$$
$$K_3 = K_2 \cup \{b \mathrel{|\!\sim} f, p \rightarrow b\}$$
$$K_4 = K_3 \cup \{b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \rightarrow a\}$$

We define a similar algorithm `ComputeGeneralSubsetSequences` with a similar sub-process `GeneralSequences`, shown in Algorithms 8 and 9 respectively in Appendix B, for computing all the 'pushing up' sequences for the justifications $\mathcal{J}_{\neg\alpha}$. However, since we do not require these justifications to achieve a particular rank, we do not have to compute the E subsets as in lines 1-7 of `ComputeSubsetSequences`. Instead we can work with the classical version of $\mathcal{K}$. In our sub-process algorithm, instead of checking if we have reached a required n value to return our sequence, we now return our sequence when we can no longer 'disprove' our antecedents in a minimal manner that adheres to our required criteria i.e. when $\mathcal{F}$ is empty. Proposition 5.6 states the soundness and completeness of `ComputeSubsetSequences` and is proven in Appendix B.

Proposition 5.6. *Let $\mathcal{K}$ be a knowledge base and γ a formula. Provided $br_{\mathcal{K}}(\gamma) \neq \infty$,* `ComputeGeneralSubsetSequences` *computes exactly $M \subseteq \mathcal{K}$ such that for all $M' \subset M, br_{M'}(\gamma) < br_M(\gamma)$ i.e. M is minimal in terms of the ranking of γ.*

Given the ability to compute these sequences, we now define our method for extending weak justifications to strong justifications. If we choose a specific sequence for a weak justification, we can use the subsets in the sequence to ensure that whenever one of our justifications $\mathcal{J}_{\neg\alpha}$ is 'pushed up' the ranks in some subset of $\mathcal{K}$, so is our weak justification. To do this, we start by considering each sequence $\mathcal{J}$ for each justification $\mathcal{J}_{\neg\alpha}$ individually and look at all the ways of adding the minimum number of statements from our weak justification 'pushing up' sequence to ensure that the weak justification is 'pushed above' $\mathcal{J}_{\neg\alpha}$ in each subset in the sequence $\mathcal{J}$. We start with $\mathcal{S}$ containing our weak justification and then iterate through each subset $K_i^{\mathcal{J}}$ in $\mathcal{J}$, checking whether we can 'disprove' the antecedents of our statements in $\mathcal{S}$ using $K_i^{\mathcal{J}}$ and $\mathcal{S}$. If we cannot, we use $K_{i+1}^{\mathcal{W}}$ from our weak justification sequence and consider all the ways of adding the minimum number of required statements to $\mathcal{S}$ so that together $K_i^{\mathcal{J}}$ and $\mathcal{S}$ 'disprove' all the antecedents of the statements previously in $\mathcal{S}$. We repeat this process until we have considered all sets in the sequence. Essentially, what we have done is create all the minimal sets S that ensure that whenever $\mathcal{J}_{\neg\alpha}$ has rank j in our sequence, our weak justifications has at least rank $j+1$. This process is formally defined by `StrongSequences` shown in Algorithm 10 in Appendix B. Proving soundness and completeness for this algorithm is left as future work.

Having constructed all these minimal sets S for each sequence for our justifications $\mathcal{J}_{\neg\alpha}$, we now consider all possible ways of taking an S set for each sequence and combining them. We then take the smallest such set as our strong justification $\mathcal{S}$. Minimizing in this manner ensures that the final set created is minimal and thus is a strong justification. This full process is defined by `StrongJustification`, shown in Algorithm 11. Proposition 5.7 states the correctness of `StrongJustification`. Both the algorithm and the proof are presented in Appendix B

Proposition 5.7. *Let $\mathcal{K}$ be a knowledge base and $\alpha \mathrel{|\!\sim} \beta$ a defeasible implication. Provided $br_{\mathcal{K}}(\alpha) \neq \infty$,* `StrongJustification` *returns a strong justification for the entailment $\mathcal{K} \mathrel{|\!\approx} \alpha \mathrel{|\!\sim} \beta$.*

Example 5.8. Consider once again the knowledge base $\mathcal{K}$ in Figure 8 and the entailment of the statement $(sp \wedge 4t) \mathrel{|\!\sim} f$. This has the weak justification and 'pushing up' sequence shown in Example 5.5. $\mathcal{K}$ has the justification 'pushing up' sets shown in Appendix B. Using `StrongJustification` we compute:

$$\mathcal{S} = \{sp \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f, b \mathrel{|\!\sim} f, sp \rightarrow p, p \rightarrow b\}$$

5.4 Limitations of the Approach

While we have defined strong justification as an extension of weak justifications, not every weak justification can be extended to create a strong justification. As an illustration of this, consider the following example.

Example 5.9. Suppose one has the knowledge base $\mathcal{K}$ shown in Figure 10 and consider the entailment of the statement $sp \wedge p \mathrel{|\!\sim} m \vee w$. This entailment has two weak justifications:

(1) $\mathcal{W}_1 = \{sp \mathrel{|\!\sim} f, f \mathrel{|\!\sim} m\}$
(2) $\mathcal{W}_2 = \{sp \mathrel{|\!\sim} f, f \mathrel{|\!\sim} w\}$

$\mathcal{W}_1$ cannot be extended to create a strong justification. If initially $\mathcal{S} = \mathcal{W}_1$, then to ensure $\mathcal{S} \cup \{p \mathrel{|\!\sim} \neg f\} \mathrel{|\!\approx}_{RC} sp \wedge p \mathrel{|\!\sim} m \vee w$, the statements $\{l \mathrel{|\!\sim} \neg x, f \mathrel{|\!\sim} w, w \mathrel{|\!\sim} x, w \rightarrow l\}$ need to be added to $\mathcal{S}$. Then $\mathcal{S} = \{l \mathrel{|\!\sim} \neg x, f \mathrel{|\!\sim} w, w \mathrel{|\!\sim} x, w \rightarrow l, sp \mathrel{|\!\sim} f, f \mathrel{|\!\sim} m\}$. But now if $f \mathrel{|\!\sim} m$ is removed from $\mathcal{S}$, we still have $\mathcal{S}' \mathrel{|\!\approx} sp \wedge p \mathrel{|\!\sim} m \vee w$ for all $\mathcal{S}' \subseteq \mathcal{S} \subseteq \mathcal{K}$. Thus due to the minimality property of strong justifications, $\mathcal{W}_1$ cannot be extended to create a strong justification.

0	$p \mid\sim \neg f, l \mid\sim \neg x$
1	$sp \mid\sim f, f \mid\sim m, f \mid\sim w, w \mid\sim x$
∞	$w \to l$

Figure 10: Base ranking for statements in Example 5.9

`Sequences` ensures that we only ever use those weak justifications that can be extended to form strong justifications; we are unable to create a minimal 'pushing up' sequence for $\mathcal{W}_1$ and thus it is not suitable for extension.

It is also not the case that all strong justifications can be defined as an extension of a weak justification. As a demonstration of this fact, consider the following example.

Example 5.10. Suppose one has a knowledge base $\mathcal{K}$ shown in Figure 11 and consider the entailment of the statement $p \mid\sim s$. This has the weak justification $\mathcal{W} = \{p \mid\sim \neg f, \neg f \mid\sim s\}$. However, consider rather $\mathcal{B} = \{b \mid\sim s, p \to b\}$ as the base set for extension. $\mathcal{B}$ is not a weak justification since it contains information removed during the Rational Closure algorithm. However, $\mathcal{B}$ can be extended to form the strong justification

$$\mathcal{S} = \{b \mid\sim s, \neg f \mid\sim s, r \mid\sim w, \neg f \mid\sim \neg w, \neg f \mid\sim r, p \to b\}.$$

First notice $\mathcal{S} \models_{RC} p \mid\sim s$. Now consider what happens if any statements are added to $\mathcal{S}$. If we add any statements that cause $\mathcal{B}$ to be thrown away, namely $p \mid\sim \neg f$, the set then contains $\mathcal{W}$ and so the entailment still holds.

0	$b \mid\sim f, b \mid\sim s, r \mid\sim w$
1	$p \mid\sim \neg f, \neg f \mid\sim \neg w, \neg f \mid\sim r, \neg f \mid\sim s$
∞	$p \to b$

Figure 11: Base ranking of statements for Example 5.10

Strong justifications that are not derived from weak justifications lack intuitive similarity to the reasoning process since they seem to present the information required to 'disprove' the actual information we used as part of our conclusion.

6 GENERAL KLM DEFEASIBLE EXPLANATION

Strong justifications may not provide the level of description that we require from a defeasible explanation for KLM style reasoning. For Rational Closure, due to the minimality requirement, if information is included in all the ways of 'pushing up' the justifications $\mathcal{J}_{\neg\alpha}$ it will not be included in the final strong justification. Thus, a strong justification may only present partial information about the ranking involved in the final entailment. However, since the ranking of statements is an integral part of the KLM reasoning process, depending on the application of the explanation, we may want to fully capture the ranking in our explanation. In this section we present a general definition for defeasible explanation in the context of an extension of the KLM framework that aims to convey the notion of the ranking involved in the entailment. Intuitively, along with presenting the information that influences the final entailment, we also want to provide reasoning as to why this was the most 'specific' information used.

6.1 Definition

Casini et al. [4] present an extension of the KLM framework, motivating for additional properties that any reasonable definition for defeasible entailment should adhere to. This form of defeasible reasoning is termed *rational defeasible entailment.* Both Rational Closure and Lexicographic Closure are forms of rational defeasible entailment. In fact, any form of rational defeasible entailment can be expressed as a refinement of Rational Closure. A rational defeasible entailment relation can be defined by a *base rank preserving rank function*, which ranks all the formulas that can generated from a particular set of atoms. Essentially the knowledge base defines a particular ranked model, the definition of which depends on the version of rational defeasible entailment we are using, which in turn defines a base rank preserving rank function r. This can be used in a generalized version of the rational closure algorithm [4] to compute defeasible entailment. We can also say $\mathcal{K} \approx \alpha \mid\sim \beta$ iff r(α) = ∞ or $r(\alpha) < r(\alpha \wedge \neg\beta)$.

Unlike in Rational Closure, the statements in the ranks of the general rational defeasible entailment algorithm are not those from our original knowledge base. So to provide explanations that relate to our original knowledge base, we instead revert to considering how our statements influence the ranked model, which in turn influences the ranking function. We propose a general definition for explanations for rational defeasible entailment which we refer to as *Minimal Rank Establishing Sets* (MRES).

Definition 6.1. For a knowledge base $\mathcal{K}$, base rank preserving rank function r and defeasible implication $\alpha \mid\sim \beta$, we define a *Minimal Rank Establishing Set* to be a subset M of $\mathcal{K}$ such that:

(1) If $r_{\mathcal{K}}(\alpha) = \infty, r_M(\alpha) = \infty$ and for all $M' \subset M, r_M(\alpha) \neq \infty$.
(2) If $r_{\mathcal{K}}(\alpha) \neq \infty$, $r_M(\alpha \wedge \neg\beta) > r_{\mathcal{K}}(\alpha)$ and for all $M' \subset M, r_{M'}(\alpha \wedge \neg\beta) < r_M(\alpha \wedge \neg\beta)$.

This definition explicitly captures the ranking involved in the final entailment.

6.2 Algorithm for Rational Closure

The base rank preserving rank function associated with Rational Closure is the base rank function $br_{\mathcal{K}}$. Thus for Rational Closure the idea of an MRES relates to the 'pushing up' sequences for weak justifications we defined in Section 5. In fact, we can adapt `ComputeSubsetSequences` and `Sequences` to create `ComputeMRES` and `MRESSequences` shown in Algorithms 12 and 13 in Appendix C respectively. These algorithms compute all the MRESs for a knowledge base $\mathcal{K}$ and a defeasible implication $\alpha \mid\sim \beta$.

If the antecedent α has finite rank, the main change required for `ComputeSequence` is the length of the sequence it computes. Currently it only computes n sets, meaning that the formula γ passed in will have a rank of at most n in the sets in the sequence. However, for an MRES we want to consider all sets in which γ has rank at least n. Thus instead of immediately returning the sequence when we have n sets, we want to continue iterating until no more sets can be created. At this point we can return the sequence. Then if we take all the sets with an index greater than or equal to n in a sequence, we will have exactly the MRESs for the entailment. We also need to account for MRESs when α has infinite rank. This can be done by computing all the justification $\mathcal{J}_{\neg\alpha}$ with infinite rank and then iterating to 'disprove' all of the antecedents, adding the minimal number of statements in each iteration. Note this will only involve using statements with infinite rank and thus we do not have to construct a sequence as with the finite case. Proposition 6.2, which is proven in Appendix C, states that `ComputeMRES` is sound and complete.

Proposition 6.2. *Let $\mathcal{K}$ be a knowledge base and $\alpha \mid\sim \beta$ a defeasible implication.*

1. *If $br_K(\alpha) \neq \infty$,* `ComputeMRES`*($\mathcal{K}, \alpha \wedge \neg\beta, br_K(\alpha)+1$) returns exactly the MRESs for $\mathcal{K} \approx_{RC} \alpha \mid\sim \beta$.*
2. *If $br_K(\alpha) = \infty$,* `ComputeMRES`*($\mathcal{K}, \alpha, \infty$) returns exactly the MRESs for $\mathcal{K} \approx_{RC} \alpha \mid\sim \beta$*

As a demonstration of `ComputeMRES` and MRESs for Rational Closure in general, consider the following example.

Example 6.3. Suppose one has a knowledge base $\mathcal{K}$ containing the following statements:

1. Birds typically fly ($b \mid\sim f$)
2. Birds typically eat bugs ($b \mid\sim e$)
3. Penguins typically do not fly ($p \mid\sim \neg f$)
4. Penguins typically swim ($p \mid\sim s$)
5. Special penguins typically fly ($sp \mid\sim f$)
6. Special penguins are penguins ($sp \rightarrow b$)
7. Penguins are birds ($p \rightarrow b$)

This has the ranking shown in Figure 12 (a). Consider the entailment of the statement 'if something is a penguin or a special penguin, it typically swims' ($p \vee sp \mid\sim s$). The MRES for this entailment is shown in Figure 12 (b), with a more detailed explanation shown in Appendix C.

0	$b \mid\sim f, b \mid\sim e$
1	$p \mid\sim \neg f, p \mid\sim s$
2	$sp \mid\sim f$
∞	$sp \rightarrow p, p \rightarrow b$

(a) Original knowledge base

0	$b \mid\sim f$
1	$p \mid\sim \neg f$, $p \mid\sim s$
∞	$sp \rightarrow p$, $p \rightarrow b$

(b) MRES for the entailment

Figure 12: Base ranking of statements for Example 6.3

If we refer to the natural language sentences, we can see how this explanation can be partitioned into the information we used to make the final conclusion (shown in blue) and the reason why this was the most specific information used.

Given that all forms of rational defeasible entailment can be defined as a refinement of Rational Closure, we propose that one could define a general algorithm for computing MRESs by refining the algorithm used for Rational Closure in an appropriate manner.

7 RELATED WORK

Horridge [7] provides and investigates a variety of algorithms for computing classical justifications. Horridge also presents an in-depth analysis of the efficiency of these various algorithms.

Chama [5] presents an adaption of the Rational Closure algorithm for the computation of justifications for Rational Closure defeasible entailment. Chama uses algorithms presented by Horridge as a basis for computing justifications. The approach here resembles the reasoning process for Rational Closure: after eliminating less specific ranks, we rely on classical tools to reason about the knowledge base, only in this case we use classical justification instead of classical entailment.

Brewka et al. [3] take a different approach for defining defeasible justifications. Brewka et al. present an abstract idea of a defeasible justification that is claimed to work for all forms of defeasible reasoning, called a strong explanation.

8 CONCLUSIONS

In this paper, we explored a variety of approaches for defining and computing explanations for defeasible entailment in the context of the KLM framework for propositional logic. We found that what we term weak justifications provided an intuitive way of defining simple explanations for the algorithmic definition of Lexicographic Closure, as they have previously done for Rational Closure.

We investigated applying strong explanations as a definition for defeasible explanation in the KLM framework, focusing on Rational Closure. We found, that for Rational Closure, weak justifications provided a basis for extension to this more expressive form of explanation and proposed and partially proved the correctness of an algorithm for extending weak justifications to form what we referred to as strong justifications. However, we also found that not all strong justifications can be defined as an extension of a weak justification and that not every weak justification can be extended to create a strong justification.

Finally, we presented a general definition for defeasible explanation, Minimal Rank Establishing Sets, in the context of an extension of the KLM framework presented by Casini et al. [4], and suggested and proved an algorithm for Rational Closure that computes such explanations. This form of defeasible explanation aims to explicitly capture the ranking involved in the reasoning process.

9 FUTURE WORK

Since weak justification algorithms have been developed and seem implementable for Lexicographic Closure and Rational Closure, future work could involve optimizing and implementing these algorithms as a part of a basic reasoning system to provide justifications for these forms of defeasible entailment.

Future work could also involve further investigating strong justifications for KLM style defeasible reasoning. This could involve proving and then refining the current approach for Rational Closure to present a more efficient algorithm that also accounts for the creation of strong justifications when the antecedent has an infinite ranking, or disproving the current approach and proposing an alternative one. Having accomplished this, one could also generalize the approach to allow for the computation of strong justifications for other forms of KLM defeasible entailment such as Lexicographic Closure.

Finally, one can also consider defining a general way of computing Minimal Ranking Establishing Sets for other forms of rational defeasible entailment, building on the algorithm suggested for Rational Closure. This could also involve defining a general definition for weak justifications within the extended framework proposed by Casini et al.

REFERENCES

[1] Mordechai Ben-Ari. 2012. *Propositional Logic: Formulas, Models, Tableaux*. Springer London, London, 7–47. https://doi.org/10.1007/978-1-4471-4129-7_2

[2] Ronald J. Brachman and Hector J. Levesque. 2004. *Knowledge representation and reasoning*. Elsevier.

[3] Gerhard Brewka, Matthias Thimm, and Markus Ulbricht. 2019. Strong inconsistency. *Artificial Intelligence* 267 (2019), 78–117.

[4] Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. 2019. Taking defeasible entailment beyond rational closure. In *European Conference on Logics in Artificial Intelligence*. Springer, 182–197.

[5] Victoria Chama. 2020. *Explanation for defeasible entailment*. Master's thesis. University of Cape Town.

[6] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Gian Luca Pozzato. 2015. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence* 226 (2015), 1–33.

[7] Matthew Horridge. 2011. *Justification based explanation in ontologies*. Ph. D. Dissertation. University of Manchester.

[8] Adam Kaliski. 2020. *An overview of KLM-style defeasible entailment*. Master's thesis. University of Cape Town.

[9] Aditya Anand Kalyanpur. 2006. *Debugging and repair of OWL ontologies*. Ph. D. Dissertation.

[10] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. 1990. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence* 44, 1-2 (1990), 167–207.

[11] Daniel Lehmann. 1995. Another perspective on default reasoning. *Annals of mathematics and artificial intelligence* 15, 1 (1995), 61–82.

[12] Daniel Lehmann and Menachem Magidor. 1992. What does a conditional knowledge base entail? *Artificial intelligence* 55, 1 (1992), 1–60.

[13] 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.

A WEAK JUSTIFICATIONS FOR LEXICOGRAPHIC CLOSURE

Algorithm 7: LexicographicClosureForJustifications

```
Input: A knowledge base $\mathcal{K}$ and DI $\alpha \mathrel{|\!\sim} \beta$
Output: true, if $K \approx_{LC} \alpha \mathrel{|\!\sim} \beta$, otherwise false, and an
        ordered pair $(i, m)$ describing a rank and subset size
$(\mathcal{R}_0, ..., \mathcal{R}_{n-1}, \mathcal{R}_\infty$, n) := BaseRank($\mathcal{K}$);
i := 0;
m := 0;
$\mathcal{R} := \bigcup_{j=0}^{j<n} \mathcal{R}_j$;
while $\mathcal{R}_\infty \cup \mathcal{R} \vDash \neg\alpha$ and $\mathcal{R} \neq \emptyset$ do
    $\mathcal{R} := \mathcal{R} \setminus \mathcal{R}_i$;
    m := | $\mathcal{R}_i$ |-1;
    $\mathcal{R}_{i,m} := \bigvee_{X \in \text{Subsets}(\mathcal{R}_i, m)} \bigwedge_{x \in X} x$;
    while $\mathcal{R}_\infty \cup \mathcal{R} \cup \{\mathcal{R}_{i,m}\} \vDash \neg\alpha$ and $m > 0$ do
        m := m-1;
        $\mathcal{R}_{i,m} := \bigvee_{X \in \text{Subsets}(\mathcal{R}_i, m)} \bigwedge_{x \in X} x$;
    end
    $\mathcal{R} := \mathcal{R} \cup \{\mathcal{R}_{i,m}\}$;
    i := i+1;
end
return $R_\infty \cup R \vDash \alpha \to \beta$, (i, m);
```

B STRONG JUSTIFICATIONS FOR RATIONAL CLOSURE

Algorithm 8: ComputeGeneralSubsetSequences

```
Input: A knowledge base $\mathcal{K}$, a formula $\alpha$
Output: Set of all sequences $(K_0, K_1, ..., K_n)$, where $\alpha$ has
        $br_{K_i}(\alpha) = i$, and each $K_i$ is minimal
sequences := GeneralSequences($\overrightarrow{\mathcal{K}}, \alpha, (\emptyset), 1$);
return sequences;
```

Proposition B.1. *Let $\mathcal{K}$ be a knowledge base, γ a formula and $M \subseteq \mathcal{K}$ such that for all $M' \subseteq M, br_{M'}(\gamma) < br_M(\gamma)$ i.e. M is minimal in terms of the ranking of γ. Let $i = br_{\mathcal{K}}(\gamma)$. Provided $i \neq \infty$, we can express M as a sequence of set $(K_0, K_1, \ldots, K_i)$ such that*

(1) $br_{K_j}(\gamma) = j$
(2) For all $K' \subset K_j, br_{K'}(\gamma) < br_{K_j}(\gamma)$

Proof. Let M, $\mathcal{K}$ and γ be as described in Proposition B.1. We will construct a sequence of sets and prove that this sequence adheres to conditions (1) and (2).

Firstly $i = br_M(\gamma)$. We recursively construct a sequence of sets as follows:

- $M_0 = M$
- $M_{x+1} = MinimizeSet(M_x \setminus R_0^{M_x}, \gamma)$

Algorithm 9: GeneralSequences

```
Input: A materialized knowledge base $\mathcal{K}$, a formula $\alpha$, the
       current sequence of subsets $(K_0, \ldots, K_{i-1})$, the index
       $i$ of the current subset
Output: A set of sequences of varying lengths
$\mathcal{A} := \neg\alpha \wedge (\bigwedge_{\beta \in Antecedents(K_{i-1})} \neg\beta)$;
S := MinimalExtension($\mathcal{A}, \mathcal{K}, K_{i-1}$);
$\mathcal{F} := \emptyset$;
if $S \neq \emptyset$ then
    for $K_i$ in S do
        if $br_{K_i}(\alpha) = i$ and Minimize($K_i, \alpha$) is False then
            $\mathcal{F} := \mathcal{F} \cup$
              GeneralSequences($\mathcal{K}, \alpha, (K_0, \ldots, K_{i-1}, K_i), i+1$);

        end
    end
if $\mathcal{F} = \emptyset$ then
    $\mathcal{F} := \{(K_0, \ldots, K_{i-1})\}$;
return $\mathcal{F}$;
```

MinimizeSet(A, γ) returns the smallest subset A' of A such that $br_{A'}(\gamma) = br_A(\gamma)$. If we set

$$K_0 = \emptyset, K_1 = M_{i-1}, \ldots, K_j = M_{i-j}, \ldots, K_i = M_{i-i} = M_0$$

we claim we have constructed an appropriate sequence. We will prove this is the case by first proving the statement '*For M_y, M_y is minimal and $br_{M_y}(\gamma) = i - y$*' using induction on y.

(BC) Let y = 0. Thus $M_0 = M$. By assumption M is minimal and $br_M(\gamma) = i = i - 0 = i - y$. Our statement is true for our base case.

(IH) Let $n \in \mathbb{N}$ with $0 \leq n < i$ and suppose our statements is true for M_n. We wish to show the statement is true for M_{n+1}. We derive M_{n+1} from M_n by removing all statements with rank 0 in M_n from M_n and minimizing the resulting set. It thus follows from the construction that M_{n+1} is minimal and so we must just prove that $br_{M_{n+1}}(\gamma) = i - (n+1)$. Let

$$E_0^{M_n}, E_1^{M_n}, \ldots, E_{i-n-1}^{M_n}, E_{i-n}^{M_n}, \ldots, E_\infty^{M_n}$$

be the E sets for M_n. If we define $M' = M_n \setminus R_0^{M_n}$ we have $M' = E_1^{M_n}$. Thus

$$E_0^{M'} = E_1^{M_n}, E_1^{M'} = E_2^{M_n}, \ldots, E_{i-n-1}^{M'} = E_{i-n}^{M_n}, E_n^{M'} = E_{i-n+1}^{M_n}$$

Since $br_{M_n}(\alpha) = i - n$, we have that $E_{i-n}^{M_n}$ is the first E_x such that $E_x^{M_n} \nvDash \neg\alpha$. Thus $E_{i-n-1}^{M'}$ is the first E_x for M' for which $E_x^{M'} \nvDash \neg\alpha$ and so $br_{M'}(\alpha) = i - n - 1 = i - (n+1)$. Finally note that $br_{M'}(\gamma) = br_{M_{n+1}}(\gamma)$ since $M_{n+1} = MinimizeSet(M')$. So we have $br_{M_{n+1}}(\gamma) = i - (n+1)$ and thus are done.

Finally, since $br_{M_y}(\gamma) = i - y$ and $K_j = M_{i-j}, br_{K_j}(\gamma) = br_{M_{i-j}}(\gamma) = i - (i - j) = j$ and K_j is minimal. Thus our sequence satisfies both criteria and so we are done. □

Proposition B.2. *Let $\mathcal{K}$ be a knowledge base and γ a formula. Provided $br_{\mathcal{K}}(\gamma) \neq \infty$,* `ComputeGeneralSubsetSequences` *computes exactly $M \subseteq \mathcal{K}$ such that for all $M' \subset M, br_{M'}(\gamma) < br_M(\gamma)$ i.e. M is minimal in terms of the ranking of γ.*

Proof. Firstly note that since $\mathcal{K}$ is finite, *MinimalExtension* will eventually return an empty set and thus the algorithm will always terminate. To prove Proposition B.2, we need to show that:

(1) Every K_i in every sequence is minimal i.e. if $K' \subset K_i, br_{K'}(\gamma) < br_{K_i}(\gamma)$
(2) Every $M \subseteq \mathcal{K}$ such that for all $M' \subseteq M, br_{M'}(\gamma) < br_M(\gamma)$ appears in a sequence.

The proof of (1) is trivial, since as part of the computation we check if each K_i is minimal before adding it to a sequence.

The proof of (2) follows from Proposition B.1. We can express M as a sequence $(K_0, K_1, \dots, K_i)$ by using the construction presented in the proof for Proposition B.1. We need to show we can construct this sequence using the approach given in `GeneralSequences`. We have $K_0 = \emptyset$ by construction so trivially K_0 will be in a sequence. We will once again use induction to show that each K_j for $j > 0$ is a part of a sequence.

(BC) Let j = 1. Then since $br_{K_1}(\gamma) = 1$, $K_1 \subset \mathcal{K}$ such that $K_1 \vDash \neg\gamma$ and K_1 is minimal. Thus K_1 will be a part of S = *MinimalExtension*$(\mathcal{A} = \neg\gamma, \mathcal{K}, \emptyset)$ and since $br_{K_1}(\alpha) = 1$ and K_1 is minimal, K_1 will be added to a sequence $(K_0, K_1, \dots)$.

(IH) Let $1 \le j < i$ and suppose K_n is part of a sequence. We wish to show K_{n+1} will be added to a sequence. To do this we need to show:

(a) $K_{n+1} \setminus K_n$ is a minimal extension of K_n that allows us to entail $\mathcal{A} = \neg\gamma \wedge (\bigwedge_{\beta \in Antecedents(K_n)} \neg\beta)$
(b) K_{n+1} is minimal
(c) $br_{K_{n+1}}(\alpha) = n + 1$

For the proof of (a), note that $K_{n+1} = M_{i-n-1}$ and $K_n = M_{i-n}$. Since we get M_{i-n} by removing $R_0^{M_{i-n-1}}$ from M_{i-n-1} and minimizing, we have $M_{i-n} \subseteq E_1^{M_{i-n-1}}$. Thus $M_{i-n-1} \vDash \neg\beta_1 \wedge \neg\beta_2 \wedge \dots$ for $\beta_x \in Antecedents(M_{i-n})$. Let $X = M_{i-n-1} \setminus M_{i-n}$. Suppose X is not minimal i.e. we can disprove all the antecedents in M_{i-n} using some subset X' of X. But then $br_{M_{i-n} \cup X'}(\gamma) \ge n + 1 = br_{M_{i-n-1}}(\gamma)$ with $M_{i-n} \cup X' \subset M_{i-n-1}$, which contradicts the minimality of M_{i-n-1}. Thus X must be minimal. Also note since $M_{i-n} \subseteq M_{i-n-1}$ and $M_{i-n} = K_n \vDash \neg\gamma$ since it is part of the sequence, we must have $M_{i-n-1} \vDash \neg\gamma$. So X will be returned by S = *MinimalExtension*$(\mathcal{A}, \mathcal{K}, K_n)$ when $\mathcal{A} = \neg\gamma \wedge (\bigwedge_{\beta \in Antecedents(K_n)} \neg\beta)$, allowing as to create K_{n+1}. Conditions (2) and (3) are satisfied based on our definition of the sequence in the proof for Proposition B.1. So K_{n+1} will be added to a sequence and thus we are done.

Since each set in the sequence for M appears in a sequence, this means M itself appears in a sequence since $M = K_i$. ◻

Proposition B.3. *Let $\mathcal{K}$ be a knowledge base and γ a formula. Provided $br_{\mathcal{K}}(\gamma) \neq \infty$,* `ComputeSubsetSequences` *returns at least one sequence of sets.*

Proof. Let $\mathcal{K}$ and γ be as described in Proposition B.3. Firstly note that since `Sequences` recurses to a maximum depth of n+1 for each sequence, the algorithm must always terminate. Now consider $M \subseteq \mathcal{K}$ such that $br_M(\gamma) = br_{\mathcal{K}}(\gamma) = j$ and for any $M' \subset M$, $br_{M'}(\gamma) < br_M(\gamma)$ and let $n \in \mathbb{N}$ with $n < j$. By Proposition B.1, we can compute a sequence $(K_0, K_1, \dots, K_n, \dots, K_j)$ using the approach given in the proof. We claim if we take $(K_0, K_1, \dots, K_n)$, we will have a sequence computed by `ComputeSubsetSequences`.

First we will prove that for each $M_s = K_{j-s}$, we have that $M_s \subseteq E_s^M$. We will prove this by induction on s.

(BC) Let s = 1. M_1 is constructed by taking $K' = M_0 \setminus R_0^{M_0} = M \setminus R_0^M$ and minimizing it. Thus $M_1 \subseteq K' \subseteq E_1^M$.

(IH) Let $1 \le s < j$ and suppose $M_s \subseteq E_s^M$. We wish to show that $M_{s+1} \subseteq E_{s+1}^M$. We construct M_{s+1} by taking $K' = M_s \setminus R_0^{M_s}$ and minimizing it. Since $M_s \subseteq E_s^M$, $E_0^{M_s} \subseteq E_s^M$ and thus $E_1^{M_s} \subseteq E_{s+1}^M$. But $M_{s+1} \subseteq K' \subseteq E_1^{M_s} \subseteq E_{s+1}^M$ and so we are done.

Thus we have that $K_x = M_{j-x} \subseteq E_{j-x}^M$. Since $j > n$, we have $j-1 \ge n$ and thus $j-x \ge n-(x-1)$. So $K_x \subseteq E_{j-x}^M \subseteq E_{n-(x-1)}^M \subseteq E_{n-(x-1)}^{\mathcal{K}} \subseteq E_{n-x}^{\mathcal{K}}$. So K_x is contained in $E_{n-x}^{\mathcal{K}}$ for each x and thus can be used to create an extension for K_{x-1}, meaning we are not limited by this restriction. So we can follow the same proof structure as the second half of the proof for Proposition 5.6 to prove that $(K_0, K_1, \dots, K_n)$ is in fact computed and returned by `ComputeSubsetSequence`. ◻

Algorithm 10: StrongSequences

```
Input: A sequence $\mathcal{N} = (S_0, S_1, \dots, S_n)$ for a formula $\alpha$, the
       formula $\alpha$, a sequence $\mathcal{M} = (J_0, J_1, \dots, J_m)$ for a
       formula $\beta$ where $m < n$, the formula $\beta$, the index $i$ of
       the current subset, the current minimal entailing set
       $\mathcal{S}$
Output: A set of sets $\mathcal{S}$ such that $br_{\mathcal{S} \cup J_i}(\alpha) > br_{\mathcal{S} \cup J_i}(\beta)$
        for all $i \le m$ and $\mathcal{S}$ is minimal
1  if $i > m$ then
2      return $\{\mathcal{S}\}$;
3  $\mathcal{A} := \neg\alpha \wedge (\bigwedge_{\beta \in Antecedents(\mathcal{S})} \neg\beta)$;
4  $\mathcal{B} := MinimalExtension(\mathcal{A}, S_{i+(n-m)}, J_i \cup \mathcal{S})$;
5  $\mathcal{F} := \emptyset$;
6  for $x$ in $\mathcal{B}$ do
7      S' = $(x \setminus J_i) \cup \mathcal{S}$;
8      if $br_x(\alpha) = br_x(\beta) + 1$ and Minimize($S, \alpha, S'$) is False
        then
9          $\mathcal{F} := \mathcal{F} \cup StrongSequences(\mathcal{N}, \alpha, \mathcal{M}, i + 1, S')$;
10     end
11 end
12 $\mathcal{S} := \mathcal{F}$;
13 return $\mathcal{S}$;
```

Note that in the `StrongSequences` algorithm we use a different version of *Minimize*(A, α, B) where it returns true if we can remove statements $x \in B$ from A and maintain the same ranking for α in A.

Proposition B.4. *Given a sequence $\mathcal{N} = (S_0, S_1, \dots, S_n)$ for a formula γ where $br_{S_i} = i$, a formula γ, a sequence $\mathcal{M} = (J_0, J_1, \dots, J_m)$ for a formula β where $br_{J_i}(\beta) = i$ and $m < n$, the formula β, the index 0 and the empty set as input,* `StrongSequences` *computes all minimal sets S such that:*

(1) $br_{S \cup J_i}(\gamma) > br_{S \cup J_i}(\beta)$ for all $0 \le i \le m$.
(2) Condition (1) does not hold for any $S' \subset S$.

Proposition B.4 is yet to be proven.

Algorithm 11: StrongJustification

```
Input: A knowledge base $\mathcal{K}$, the DI $\alpha \mathrel{|\!\sim} \beta$ for the
       entailment
Output: A strong justification $\mathcal{S}$
1  $\mathcal{J}$ := ComputeGeneralSubsetSequences($\mathcal{K}, \alpha$);
2  m := max$\{br_{\mathcal{K}}(j) : j \in \mathcal{J}\}$;
3  sequences = ComputeSubsetSequences($\mathcal{K}, \alpha \wedge \neg\beta, m+1$);
4  Choose K' = $(K_0, K_1, \ldots, K_{m+1})$ from sequences;
5  n := $|\mathcal{J}|$;
6  Let $S_1, S_2, \ldots, S_n$ be the set of minimal entailing sets
     associated with each sequence;
7  for $J_i$ in $\mathcal{J}$ do
8      $S_i$ := StrongSequences($K', \alpha \wedge \neg\beta, J_i, \beta, 0, \varnothing$);
9  end
10 $\mathcal{B} = \{s_1 \cup s_2 \cup \ldots \cup s_n : s_i \in S_i\}$;
11 x = min$\{|X| : X \in \mathcal{B}\}$;
12 Choose $\mathcal{S}$ from $\mathcal{B}$ such that $|\mathcal{S}| = x$;
13 return $S$;
```

Observation 1. *Giordano et al. [6] $\mathcal{K} \approx_{RC} \alpha \mathrel{|\!\sim} \beta$ iff $br_{\mathcal{K}}(\alpha) < br_{\mathcal{K}}(\alpha \wedge \neg\beta)$ or $br_{\mathcal{K}}(\alpha) = \infty$*

Proposition B.5. *Let $\mathcal{K}$ be a knowledge base and $\alpha \mathrel{|\!\sim} \beta$ a defeasible implication. Provided $br_{\mathcal{K}}(\alpha) \neq \infty$,* `StrongJustification` *returns a strong justification for the entailment $\mathcal{K} \approx_{RC} \alpha \mathrel{|\!\sim} \beta$.*

Proof. Let $\mathcal{K}$ and $\alpha \mathrel{|\!\sim} \beta$ be as described in Proposition B.5. Firstly note that if all the sub-processes of `StrongJustification` terminate, so will `StrongJustification`. Also note that by Proposition B.3, sequences is non-empty and thus we can always choose a K' to use for strengthening. Finally, note $\mathcal{J}$ will always be non-empty. In the case where $br_{\mathcal{K}}(\alpha) = 0$, it will just contain $\varnothing$. So `StrongJustification` will always return some set $\mathcal{S}$.

To show that $\mathcal{S}$ returned by `StrongJustification` is a strong justification, we must show that:

(1) For any $\mathcal{S} \subseteq S' \subseteq K$, $S' \approx_{RC} \alpha \mathrel{|\!\sim} \beta$
(2) For any $S'' \subset \mathcal{S}$, condition (1) does not hold.

For the proof of (1), suppose $\mathcal{S} \subseteq S' \subseteq K$ and let M be such that $br_M(\alpha) = br_{S'}(\alpha)$ and for all $M' \subset M, br_{M'}(\alpha) < br_M(\alpha)$. By Proposition B.2, M is part of a sequence computed by `ComputeGeneralSubsetSequences`. Let $\mathcal{S}_M$ be the minimal entailing set for this sequence included in $\mathcal{S}$. So $\mathcal{S}_M \cup M \subseteq \mathcal{S} \cup M \subseteq S'$ and thus $br_{S'}(\alpha \wedge \neg\beta) \geq br_{\mathcal{S} \cup M}(\alpha \wedge \neg\beta) \geq br_{\mathcal{S}_M \cup M}(\alpha \wedge \neg\beta) > br_{\mathcal{S}_M \cup M}(\alpha)$. But $br_{\mathcal{S}_M \cup M}(\alpha) = br_M(\alpha) = br_{S'}(\alpha)$ since $S_M \cup M \subseteq S'$ and M is minimal. Thus $br_{S'}(\alpha \wedge \neg\beta) > br_{S'}(\alpha)$ and so $S' \approx_{RC} \alpha \mathrel{|\!\sim} \beta$.

For the proof of (2), let $S'' \subset \mathcal{S}$ and let $z \in \mathcal{S} \setminus S''$. Let $S_1, S_2, \ldots, S_m$ be the minimal entailing sets that contained x and $X_1, X_2, \ldots, X_n$ be their associated sequences. If $br_{x_i \cup S''}(\alpha \wedge \neg\beta) > br_{x_i \cup S''}(\alpha)$ for all x_i in each X_j, then there is some minimal subset J_j of S'' that ensures this condition for each j. But then, by Proposition B.4, each of there J_j would have been computed for each sequence and thus $S'' \subseteq \mathcal{B}$. Since $S'' \subset \mathcal{S}$, $|S''| < |\mathcal{S}|$. But this contradicts the fact that $\mathcal{S}$ is chosen to be a set of minimum size in $\mathcal{B}$. Thus S' cannot exist and so $\mathcal{S}$ is in fact minimal. □

Example B.6. We provide a full demonstration of the algorithm from Example 5.8 which uses the knowledge base in Figure 8 and the entailment query $(sp \wedge 4t) \mathrel{|\!\sim} f$. We have only one weak justification $\mathcal{W} = \{sp \mathrel{|\!\sim} f\}$. Thus we have a single sequence returned by `ComputeSubsetSequences`, namely $(K_0, K_1, K_2, K_3, K_4)$ with:

$$K_0 = \varnothing$$
$$K_1 = \{sp \mathrel{|\!\sim} f\}$$
$$K_2 = K_0 \cup \{p \mathrel{|\!\sim} \neg f, sp \rightarrow p\}$$
$$K_3 = K_1 \cup \{b \mathrel{|\!\sim} f, p \rightarrow b\}$$
$$K_4 = K_2 \cup \{b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \rightarrow a\}$$

There are 9 justifications for $\neg(sp \wedge 4t)$.

(1) $J_1 = \{a \mathrel{|\!\sim} \neg w, b \mathrel{|\!\sim} w, 4t \mathrel{|\!\sim} b, b \rightarrow a\}$. This has the associated sequences:
   (a) $K_0^{J_1} = \varnothing, K_1^{J_1} = J_1$
(2) $J_2 = \{b \mathrel{|\!\sim} f, 4t \mathrel{|\!\sim} \neg f, 4t \mathrel{|\!\sim} b\}$. This has the associated sequences:
   (a) $K_0^{J_2} = \varnothing, K_1^{J_2} = J_2, K_2^{J_2} = K_1^{J_2} \cup \{b \mathrel{|\!\sim} w, b \rightarrow a, a \mathrel{|\!\sim} \neg w\}$
(3) $J_3 = \{b \mathrel{|\!\sim} j, 4t \mathrel{|\!\sim} \neg j, 4t \mathrel{|\!\sim} b\}$. This has the associated sequences:
   (a) $K_0^{J_3} = \varnothing, K_1^{J_3} = J_3, K_2^{J_3} = K_1^{J_3} \cup \{b \mathrel{|\!\sim} w, b \rightarrow a, a \mathrel{|\!\sim} \neg w\}$
(4) $J_4 = \{sp \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f, sp \mathrel{|\!\sim} p\}$. This has the associated sequences:
   (a) $K_0^{J_4} = \varnothing, K_1^{J_4} = J_4, K_2^{J_4} = K_1^{J_4} \cup \{b \mathrel{|\!\sim} f, p \rightarrow b\}, K_3^{J_4} = K_2^{J_4} \cup \{b \mathrel{|\!\sim} w, b \rightarrow a, a \mathrel{|\!\sim} \neg w\}$
(5) $J_5 = \{b \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f, p \rightarrow b, sp \rightarrow p\}$. This has the associated sequences:
   (a) $K_0^{J_5} = \varnothing, K_1^{J_5} = J_5, K_2^{J_5} = K_1^{J_5} \cup \{b \mathrel{|\!\sim} w, b \rightarrow a, a \mathrel{|\!\sim} \neg w\}$.
(6) $J_6 = \{a \mathrel{|\!\sim} \neg w, b \mathrel{|\!\sim} w, p \rightarrow b, sp \rightarrow p, b \rightarrow a\}$. This has the associated sequence:
   (a) $K_0^{J_6} = \varnothing, K_1^{J_6} = J_6$.
(7) $J_7 = \{4t \mathrel{|\!\sim} \neg f, sp \mathrel{|\!\sim} f\}$. This has the associated sequences:
   (a) $K_0^{J_7} = \varnothing, K_1^{J_7} = J_7, K_2^{J_7} = K_1^{J_7} \cup \{b \mathrel{|\!\sim} f, 4t \mathrel{|\!\sim} b, p \mathrel{|\!\sim} \neg f, sp \rightarrow p\}, K_3^{J_7} = K_2^{J_7} \cup \{p \rightarrow b, b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \rightarrow a\}$.
   (b) $K_0^{J_7} = \varnothing, K_1^{J_7} = J_7, K_2^{J_7} = K_1^{J_7} \cup \{b \mathrel{|\!\sim} j, 4t \mathrel{|\!\sim} b, 4t \mathrel{|\!\sim} \neg j, sp \rightarrow p\}, K_3^{J_7} = K_2^{J_7} \cup \{p \rightarrow b, b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \rightarrow a\}$.
(8) $J_8 = \{b \mathrel{|\!\sim} f, 4t \mathrel{|\!\sim} \neg f, sp \rightarrow p, p \rightarrow b\}$. This has the associated sequences:
   (a) $K_0^{J_8} = \varnothing, K_0^{J_8} = J_8, K_1^{J_8} = K_1^{J_8} \cup \{b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \mathrel{|\!\sim} a\}$
(9) $J_9 = \{b \mathrel{|\!\sim} j, 4t \mathrel{|\!\sim} \neg j, sp \rightarrow p, p \rightarrow b\}$. This has the associated sequences:
   (a) $K_0^{J_9} = \varnothing, K_1^{J_9} = J_9, K_2^{J_9} = K_1^{J_9} \cup \{b \mathrel{|\!\sim} w, a \mathrel{|\!\sim} \neg w, b \rightarrow a\}$.

We use `StrongSequences` to compute the set of all minimum sets which ensure that $br_{S \cup J_i}(\alpha \wedge \beta) > br_{S \cup J_i}(\alpha)$ for each of these sequences above. The largest set required is that for the sequences for J_7, these both return the set $\mathcal{S} = \{sp \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f, b \mathrel{|\!\sim} f, sp \rightarrow p, p \rightarrow b\}$. All the other sets returned for the other sequences are subsets of this set, and thus taking the minimum combination of these sets we get we will always get the strong justification:

$$\mathcal{S} = \{sp \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f, b \mathrel{|\!\sim} f, sp \to p, p \to b\}$$

C GENERAL KLM DEFEASIBLE EXPLANATION

Algorithm 12: ComputeMRES

```
Input: A knowledge base $\mathcal{K}$, a formula $\alpha$, the minimum
       rank $n \leq br_{\mathcal{K}}(\alpha)$ required for $\alpha$ in the final subset of
       a sequence
Output: The set of all MRESs for the entailment
i := 0;
$E_0 := \overrightarrow{\mathcal{K}}$;
repeat
    $E_{i+1} := \{\alpha \to \beta \in E_i \mid E_i \models \neg\alpha\}$;
    i := i+1;
until $E_{i-1} = E_i$;
$E_\infty := E_{i-1}$;
if $n = \infty$ then
    MRES := ComputeAllJustifications($E_\infty, \neg\alpha$);
    previous := $\varnothing$;
    repeat
        previous := MRES;
        for S in MRES do
            $\mathcal{A} := \neg\alpha \wedge (\bigwedge_{\beta \in Antecedents(S)} \neg\beta)$;
            S' := ComputeAllJustifications($E_\infty, \mathcal{A}$);
            MRES = (MRES$\setminus S) \cup S'$
        end
    until MRES = previous;
else
    sequences := MRESSequences($(E_0, \ldots, E_\infty), \alpha, (\varnothing), n, 1$);
    MRES := $\{ K_i \in X : X \in sequences$ and $i \geq n\}$
return MRES;
```

Observation 2. *Giordano et al. [6] For every knowledge base $\mathcal{K}$ and α a formula, $br_K(\alpha) = min\{i :$ there is a$v \in [[\alpha]] s.t. R^{RC}_{\mathcal{K}}(v) = i\}$*

Proposition C.1. *Let $\mathcal{K}$ be a knowledge base and $\alpha \mathrel{|\!\sim} \beta$ a defeasible implication.*

(1) If $br_K(\alpha) \neq \infty$, `ComputeMRES`($\mathcal{K}, \alpha \wedge \neg\beta, br_K(\alpha)+1$) returns exactly the MRESs for $\mathcal{K} \approx_{RC} \alpha \mathrel{|\!\sim} \beta$.

(2) If $br_K(\alpha) = \infty$, `ComputeMRES`($\mathcal{K}, \alpha, \infty$) returns exactly the MRESs for $\mathcal{K} \approx_{RC} \alpha \mathrel{|\!\sim} \beta$

Proof. Let $\mathcal{K}$, $\alpha \mathrel{|\!\sim} \beta$ be as described in Proposition C.1. There are two separate courses for the algorithm, determined by $br_K(\alpha) \neq \infty$ or $br_K(\alpha) = \infty$. For the former, note since $\mathcal{K}$ is finite, *MinimalExtension*$(\mathcal{A}, E_{n-i}, K_{i-1})$ will eventually return an empty set and thus `MRESSequences` will terminate and so `ComputeMRES` will terminate. For the latter, note since E_∞ is finite, `ComputeMRES` will terminate.

For the proof of (1), first note $br_M(\alpha) = \infty$ iff $R_\infty \models \neg\alpha$. To see this first note if $R_\infty \models \neg\alpha$, since $R_\infty \subseteq E_i$ and $br_M(\alpha) = min\{i :$

Algorithm 13: MRESSequences

```
Input: A knowledge base $\mathcal{K}$= $(E_0, E_1, ...E_\infty)$, a formula $\alpha$,
       the current sequence of subsets $(K_0, \ldots, K_{i-1})$, the
       rank $n$ that is the minimum rank $\alpha$ can have in the
       final subset of the sequence, the index i of the
       current subset
Output: A set of sequences of length at least $n$
$\mathcal{A} := \neg\alpha \wedge (\bigwedge_{\beta \in Antecedents(K_{i-1})} \neg\beta)$;
S := MinimalExtension($\mathcal{A}, E_{n-i}, K_{i-1}$);
$\mathcal{F} := \varnothing$;
if $S \neq \varnothing$ then
    for $K_i$ in S do
        if $br_{K_i}(\alpha) = i$ and Minimize($K_i$) is False then
            $\mathcal{F} := \mathcal{F} \cup$
              MRESSequences($\mathcal{K}, \alpha, (K_0, \ldots, K_{i-1}, K_i), n, i+1$);
        end
    end
if $\mathcal{F} := \varnothing$ and $i > n$ then
    $\mathcal{F} = \{(K_0, \ldots, K_{i-1})\}$;
return $\mathcal{F}$;
```

$E_i \not\models \neg\alpha\}$, $E_i \models \neg\alpha$ for all i and thus $br_M(\alpha) = \infty$. If $br_M(\alpha) = \infty$, for all $\alpha \mathrel{|\!\sim} \beta \in M, \alpha \mathrel{|\!\sim} \beta \in R_\infty$. But $R_\infty \models \neg\gamma$ for all $\gamma \mathrel{|\!\sim} \zeta \in R_\infty$. Thus $R_\infty \models \neg\alpha$[1]. Now to show that any S in MRES given by the algorithm, note that $S \models \neg\beta$ for all $\beta \in Antecedents(S)$. Thus S = R^S_∞ and $S \models \neg\alpha$, so $br_S(\alpha) = \infty$. Finally, since S is a justification, it is minimal and thus S is an MRES. Too see that any MRES M will be computed by this algorithm, note that $M = R^M_\infty$ and $br_M(\alpha) = \infty$. Thus $M \models (\neg\alpha \wedge \beta_1 \wedge \ldots)$ for $\beta_i \in Antecedents(M)$ and since M is an MRES, M is minimal. This means M is a justification for $(\neg\alpha \wedge \beta_1 \wedge \ldots)$ and so it will be computed by the algorithm.

For the proof of (2) we need to show:

(a) Each set in a sequence produced by `ComputeMRES`($\mathcal{K}, \alpha \wedge \neg\beta$, $n = br_K(\alpha) + 1$) with index greater than or equal to n is an MRES.

(b) If M is a MRES for $\mathcal{K}$ and $\alpha \mathrel{|\!\sim} \beta$, M is a part of a sequence in `ComputeMRES`($\mathcal{K}, \alpha \wedge \neg\beta$, $n = br_K(\alpha) + 1$) and has an index greater than or equal to n.

For the proof of (a), note that in a sequence $(K_0, K_1, \ldots, K_n, \ldots)$, each K_i is minimal and $br_{K_i}(\alpha \wedge \neg\beta) = i$ by construction. Thus if we take any set K_i with $i \geq n$, we have $br_{K_i} = i \geq n = br_{\mathcal{K}}(\alpha) + 1 > br_{\mathcal{K}}(\alpha)$. Thus K_i is a MRES.

For the proof of (b), let M be a MRES. Note since M is minimal and $br_M(\alpha \wedge \neg\beta) \neq \infty$, by Proposition B.1 we can construct a sequence $(K_0, K_1, \ldots, K_j)$ where $j = br_M(\alpha \wedge \neg\beta)$. Following a similar proof to Proposition B.3, we note that taking extensions from E_{n-i} does not affect us choosing extensions for creating K_{x+1} from K_x. Thus following the same proof as the second half of the proof for Proposition B.2, we have that each K_i will be a part of a sequence and since M is a MRES, $j = br_M(\alpha \wedge \neg\beta) > br_{\mathcal{K}}(\alpha)$ and so $j \geq br_{\mathcal{K}}(\alpha) + 1 = n$. Thus K_j will be part of a sequence returned

[1]Proof base on correspondence with Tommie Meyer

by `ComputeMRES`($\mathcal{K}$, $\alpha \wedge \neg\beta$, $n = br_K(\alpha) + 1$) and will have an index $j \geq n$. □

Example C.2. Suppose one has a knowledge base $\mathcal{K}$ containing the following statements:

(1) Birds typically fly ($b \mid\!\sim f$)
(2) Birds typically eat bugs ($b \mid\!\sim e$)
(3) Penguins typically do not fly ($p \mid\!\sim \neg f$)
(4) Penguins typically swim ($p \mid\!\sim s$)
(5) Special penguins typically fly ($sp \mid\!\sim f$)
(6) Special penguins are penguins ($sp \rightarrow b$)
(7) Penguins are birds ($p \rightarrow b$)

This has the ranking shown in Figure 13 (a). Consider the entailment of the statement 'if something is a penguin or a special penguin, it typically swims' ($p \vee sp \mid\!\sim s$). We have $br_{\mathcal{K}}(p \vee sp) = 1$, thus `ComputeMRES`($\mathcal{K}$, $(p \vee sp) \wedge \neg s$, 2) uses `MRESSequences` to compute the sequence (K_0, K_1, K_2) where:

$$K_0 = \varnothing$$
$$K_1 = \{p \mid\!\sim s, sp \rightarrow p\}$$
$$K_2 = K_1 \cup \{b \mid\!\sim f, p \mid\!\sim \neg f, p \rightarrow b\}$$

Taking K_2 then gives us the MRES for this entailment. The MRES for this entailment is shown in Figure 13 (b).

0	$b \mid\!\sim f, b \mid\!\sim e$
1	$p \mid\!\sim \neg f, p \mid\!\sim s$
2	$sp \mid\!\sim f$
∞	$sp \rightarrow p, p \rightarrow b$

(a) Original knowledge base

0	$b \mid\!\sim f$
1	$p \mid\!\sim \neg f$, $p \mid\!\sim s$
∞	$sp \rightarrow p$, $p \rightarrow b$

(b) MRES for the entailment

Figure 13: Base ranking of statements for Example C.2