DEPARTMENT OF COMPUTER SCIENCE

# CS/IT Honours Project Final Paper 2023

Title: Extending Defeasible Reasoning beyond Rational Closure

Author: David Pullinger

Project Abbreviation: EXTRC

Supervisor(s): 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* | 3 |
| Results, Findings and Conclusions | *10* | *20* | 17 |
| Aim Formulation and Background Work | *10* | *15* | 15 |
| Quality of Paper Writing and Presentation | *10* | | 10 |
| Quality of Deliverables | *10* | | 10 |
| Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*) | *0* | *10* | |
| **Total marks** | | **80** | **80** |

# Extending Defeasible Reasoning beyond Rational Closure

David Pullinger
plldav013@myuct.ac.za
University of Cape Town
Cape Town, South Africa

## ABSTRACT

In classical logic, entailment is concerned with computing what conclusions can be made from some predefined knowledge base. Defeasible entailment extends this idea by allowing the knowledge base to also model uncertain information, where necessary. The most notable method for computing defeasible entailment is rational closure, which utilizes the notion of a statement's base rank to partition the knowledge base into ranks. However, base rank may not fully capture our insights about the knowledge base. Rather, a $\mathcal{K}$-faithful[1], user-defined ranking of the statements within our knowledge base may be needed. This paper presents two algorithms for computing such a ranking, given some form of user preferences. Then, we prove that these algorithms respect the maximum number of user preferences, and that they generate a $\mathcal{K}$-faithful ranking. Finally, an implementation and algorithmic analysis is presented.

## 1 INTRODUCTION

### 1.1 Motivation and Context

Knowledge representation and reasoning [11] is a vital part of many intelligent systems, allowing information we have about the world to be represented in a form that computers can process, analyze and draw conclusions from. Propositional logic is one such foundational system, offering a structured framework where statements about the world can be expressed as simple, atomic propositions and combined using logical connectives to form more complex assertions. Then, using classical entailment, we can systematically deduce new propositions from a given set of premises, ensuring that the derived conclusions are logically consistent and valid within the established framework.

However, while classical entailment provides a robust mechanism for deduction in environments with clear and definite information, real-world scenarios often present us with some form of uncertainty. This leads to situations where strictly following the rules of classical entailment might not produce conclusions that align with our intuitive understanding of the situation at hand. To address these issues, the concept of defeasible entailment has been introduced.

Defeasible entailment operates on the principle of non-monotonic logic. Unlike classical entailment, which operates under monotonicity (where the addition of premises never reduces the set of conclusions), defeasible entailment allows for the possibility that new information can retract previous conclusions. Thus, it provides a mechanism to handle conflicting or incomplete information by deriving the most probable or justified conclusions based on the current knowledge base, acknowledging the potential for these conclusions to be revised as new data is incorporated.

[1]As defined in [5].

The most notable method of defeasible entailment explored in the literature is rational closure, which serves as a basis for a well-defined class of entailment relations. This project aims to explore these extensions of rational closure, using Bayesian inference and user preferences. Furthermore, we develop a tool for generating large, defeasible knowledge bases to facilitate testing and validation of such defeasible entailment relations.

### 1.2 Contribution

Rational closure has been defined both semantically and syntactically. The syntactic definition, which will be the focus of this paper, employs the notion of a statement's base rank to order statements by their defeasibility. However, since a statement's base rank is based purely on the syntax of the knowledge base, it may not capture our full understanding of the knowledge base and its context. This gap prompts the exploration of custom, user-defined rankings of statements as a solution to better reflect our comprehensive insights into the knowledge base.

The aim of this paper is to introduce and validate two new algorithms that allow for a more comprehensive and user-centric approach in computing defeasible entailment. These algorithms, `GuidedUserRank` and `UnguidedUserRank`, allow a user to impose any preference of statements on a knowledge base $\mathcal{K}$, and return a valid ranking of statements, which is shown to be $\mathcal{K}$-faithful. These algorithms therefore result in an entailment relation which is LM-rational and falls in the class of basic defeasible entailment relations, as defined in [5].

### 1.3 Structure of paper

The paper is structured as follows:

- **Section 2 - Background**: This section presents foundational theories and principles related to propositional logic, KLM-style defeasible reasoning, rational closure as a form of defeasible entailment, and various extensions to rational closure in the literature.
- **Section 3 - Algorithm Design**: This section introduces two algorithms, `GuidedUserRank` and `UnguidedUserRank`, designed to incorporate user preferences. Properties, motivations, and structural equivalences of the algorithms are discussed. The section also presents a significant result about the maximality of user preferences which are satisfied by the two algorithms.
- **Section 4 - Algorithm Validation**: This section provides theoretical validation of the proposed algorithms. It demonstrates that the rank function produced by `GuidedUserRank` and `UnguidedUserRank` is $\mathcal{K}$-faithful and minimal rank preserving.

- **Section 5 - Implementation and Analysis**: This section details the technical aspects of the algorithm implementation. It offers a performance analysis contrasting the proposed algorithms against `BaseRank`.
- **Section 6 - Conclusions**: The paper concludes with a reflection on the importance of user preferences in ranking statements. It summarizes the strengths and applications of the proposed algorithms and outlines potential areas for future research.

## 2 BACKGROUND

### 2.1 Propositional Logic

Propositional logic operates on simple, indivisible units of meaning called *atoms* or *propositional variables.* We denote the set of all such propositional variable with $\mathcal{P}$. To construct more complex statements, called *formulas*, we use *connectives.* The primary connectives in propositional logic, are conjunction ($\land$), disjunction ($\lor$), negation ($\neg$), implication ($\rightarrow$), and biconditional ($\leftrightarrow$)[2].

**Example 2.1.** Consider two atoms $p$ and $q$ representing the statements "It is raining" and "The ground is wet", respectively, Then, we have that: $p \land q$ is interpreted as "It is raining and the ground is wet", $\neg p$ is interpreted as "It is not raining", and $p \rightarrow q$ is interpreted as "If it is raining then the ground is wet".

Furthermore, we can connect formulas themselves using these connectives, resulting in a diverse and expressive language. We denote the set of all these formulas as $\mathcal{L}$.

**Example 2.2.** Consider the atoms $p$ and $q$ as before. Then, the formula $\neg p \rightarrow \neg q$ is interpreted as "If it is not raining, then the ground is not wet".

Now, the semantics of propositional logic focus on determining the truth values of formulas based on the truth values of their individual atoms. To begin, we define a *valuation* as a function which maps an atom $p \in \mathcal{P}$ to either $T$ or $F$, understood to be true or false, respectively [2]. We denote the set of all such valuations as $\mathcal{U}$.

**Example 2.3.** Consider the atoms $p, q, r \in \mathcal{P}$ and some valuation $u \in \mathcal{U}$. Under this valuation, the atoms might be assigned values as follows: $u(p) = T,\ u(q) = F,$ and $u(r) = F$. It is worth noting that a concise way to represent this valuation is $\mathrm{p\overline{q}\overline{r}}$. In this notation, an atom with a bar above it is deemed false, while one without a bar is true.

Given that the truth values of individual atoms have been established through a valuation, the truth value of a formula is determined using the truth-functional properties of its logical connectives, which take one or more truth values as input and produce a truth value as output. If a formula $\alpha \in \mathcal{L}$ is $T$ under some valuation $u \in \mathcal{U}$, we say that *u satisfies* $\alpha$, and write $u \models \alpha$. Conversely, if $\alpha$ is $F$ under $u$, we write $u \not\models \alpha$ [2]. Furthermore, if $u \models \alpha$, we say the $u$ is a *model* of $\alpha$ and write $u \in Mod(\alpha)$, where $Mod(\alpha)$ denotes the set of all models of $\alpha$.

**Example 2.4.** Consider the atoms $p, q, r$ and the valuation $u$ as before. Then, we have (for instance) that $\neg p$ is $F$, while $\neg q \land \neg r$ is $T$. Thus $u \not\models \neg p$ but $u \models \neg q \land \neg r$. In this case, $u \in Mod(\neg q \land \neg r)$.

We can extend the idea of satisfaction of a formula to multiple formulas. First, we define a *knowledge base* as a set of formulas and denote it as $\mathcal{K}$. Then, a valuation satisfies $\mathcal{K}$ if it satisfies all the formulas in $\mathcal{K}$. That is to say, given some $u \in \mathcal{U}$, $u \models \mathcal{K}$ if and only if $u \models \alpha, \forall \alpha \in \mathcal{K}$. We can then call $u$ a model of $\mathcal{K}$, and denote the set of all models of $\mathcal{K}$ as $Mod(\mathcal{K})$.

Finally, we consider what we can deduce from a knowledge base. We say that a formula $\alpha \in \mathcal{L}$ is a *logical consequence* of $\mathcal{K}$, denoted[2] $\mathcal{K} \models \alpha$, if and only if every valuation $u \in Mod(\mathcal{K})$ also satisfies $\alpha$. In other words, $\mathcal{K} \models \alpha$ if it is impossible for all the formulas in $\mathcal{K}$ to be true and $\alpha$ to be false simultaneously. This relationship captures the notion of *entailment* in propositional logic, where the truth of some formulas guarantees the truth of another [2].

**Example 2.5.** Let $\mathcal{K} = \{p, q \rightarrow \neg p\}$. Then, we can deduce that $\neg q$ is a logical consequence of (equivalently, entailed by) $\mathcal{K}$, because if both $p$ is true and $q \rightarrow \neg p$ is true, then $q$ must be false. Thus, $\mathcal{K} \models \neg q$.

### 2.2 KLM-style Defeasible Reasoning

Consider the statements "birds have hollow bones", "kiwis are birds" and "kiwis do not have hollow bones". We can use propositional atoms and formulas to represent this information in the following knowledge base: $\mathcal{K} = \{b \rightarrow h, k \rightarrow b, k \rightarrow \neg h\}$. From $k \rightarrow b$ and $b \rightarrow h$, one might infer that $k \rightarrow h$. However, given that $k \rightarrow \neg h$ is a formula in $\mathcal{K}$, this inference is contradicted. In fact, $\mathcal{K}$ is only satisfied by valuations in which $k$ is false, implying that kiwis do not exist. This limitation underscores a significant flaw in classic propositional logic. Fortunately, a concept known as defeasible reasoning offers a more effective approach for handling such scenarios [12].

In [9], the authors (commonly abbreviated to KLM) consider a form of defeasible reasoning known as preferential reasoning, where they introduce a binary relation over the propositional language. They term this relation a *preferential consequence relation*, and denote it by $\mid\!\sim$. Since $\mid\!\sim$ is a binary relation, it corresponds to a set of ordered pairs $(\alpha, \beta)$, where $\alpha, \beta \in \mathcal{L}$. In KLM's formulation, $\mid\!\sim$ is also conceptualized as a set of *defeasible implications*, denoted $\alpha \mid\!\sim \beta$. In this system, $\alpha \mid\!\sim \beta$ is read "typically, if $\alpha$ is true, we can conclude that $\beta$ is true, unless there is evidence to the contrary".

**Example 2.6.** Consider the formulas $\alpha, \beta, \gamma, \sigma \in \mathcal{L}$. Then the following are equivalent:

- $(\alpha, \beta), (\gamma, \sigma) \in \mid\!\sim$
- $\alpha \mid\!\sim \beta$ and $\gamma \mid\!\sim \sigma$
- typically, if $\alpha$, we can conclude $\beta$ and typically, if $\gamma$, we can conclude $\sigma$, unless there is information to the contrary

KLM define 6 properties that any acceptable $\mid\!\sim$ should follow. Later, in [10], another property is added, resulting in what we will refer to as an *LM-rational* consequence relation. It is shown in [10] that such a consequence relation can be represented by a *ranked interpretation*, defined as follows [5]: A ranked interpretation $\mathcal{R}$ is a function from $\mathcal{U}$ to $\mathbb{N} \cup \{\infty\}$ such that $\mathcal{R}(u) = 0$ for some $u \in \mathcal{U}$, and which satisfies the following convexity property: for all $i \in \mathbb{N}$,

[2]Note that if a valuation is on the left hand side of $\models$, it is understood to represent satisfaction, while a set of formulas on the left hand side of $\models$ is understood to represent entailment.

if there exists a valuation $v \in \mathcal{U}$ such that $\mathcal{R}(v) = i$, then there exists, for each $0 \leq j < i$, a valuation $w$ such that $\mathcal{R}(w) = j$. Then, given a ranked interpretation $\mathcal{R}$ and valuation $u \in \mathcal{U}$, we call $\mathcal{R}(u)$ the *rank* of $u$ with regards to $\mathcal{R}$ [5].

Since it is understood that valuations with a lower rank are more preferred (or more typical) than valuations with a higher rank, we would like to define satisfaction and entailment of formulas in terms of these valuations. To begin, we define a *minimal $\alpha$-world* as a valuation that, among all valuations satisfying $\alpha$, has the lowest rank in $\mathcal{R}$. That is to say, a valuation $u \in Mod(\alpha)$ is a minimal $\alpha$-world if and only if there exists no valuation $v \in Mod(\alpha)$ such that $\mathcal{R}(v) < \mathcal{R}(u)$ [5].

Then, we have that a ranked interpretation $\mathcal{R}$ satisfies a defeasible implication $\alpha \mid\sim \beta$, denoted $\mathcal{R} \models \alpha \mid\sim \beta$, if and only if all the minimal $\alpha$ worlds in $\mathcal{R}$ satisfy $\beta$. Furthermore, a ranked interpretation $\mathcal{R}$ satisfies a classical formula $\alpha$, denoted $\mathcal{R} \models \alpha$, if and only if all valuations with finite rank in $\mathcal{R}$ satisfy $\alpha$. Equivalently, $\mathcal{R} \models \alpha$ if and only if $\mathcal{R} \models \neg\alpha \mid\sim \bot$, where $\bot$ is the logical constant representing a formula which is always $F$ [3]. Therefore, we can represent any classical formula $\alpha$ as the defeasible implication $\neg\alpha \mid\sim \bot$. Finally, a ranked interpretation $\mathcal{R}$ satisfies a set of defeasible implications $\mathcal{K}$ (a conditional knowledge base), if and only if $\mathcal{R} \models \alpha \mid\sim \beta, \forall \alpha \mid\sim \beta \in \mathcal{K}$. In this case, we call $\mathcal{R}$ a *ranked model* of $\mathcal{K}$.

**Example 2.7.** Let $\mathcal{K} = \{k \rightarrow b, b \mid\sim h, k \mid\sim \neg h\}$ be a conditional knowledge base representing the statements that "kiwis are birds", "birds **typically** have hollow bones", and "kiwis **typically** do not have hollow bones". Then the following is a graphical representation of a ranked model of $\mathcal{K}$ (with the infinite rank omitted):

| | |
|---|---|
| 2 | kbh |
| 1 | $\bar{k}b\bar{h}$ $kb\bar{h}$ |
| 0 | $\bar{k}\bar{b}\bar{h}$ $\bar{k}\bar{b}h$ $\bar{k}bh$ |

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

This is a ranked model of $\mathcal{K}$ since all the valuations satisfy the classical statement $k \rightarrow b$, the minimal $b$-world ($\bar{k}bh$) satisfies $h$ and the minimal $k$-world ($kb\bar{h}$) satisfies $\neg h$.

Now that we have defined satisfaction of formulas by a ranked interpretation $\mathcal{R}$, we consider entailment of formulas by a ranked interpretation $\mathcal{R}$. We say that, for a given ranked interpretation $\mathcal{R}$, $\mathcal{R}$ generates a *defeasible entailment relation*, denoted $\approx_{\mathcal{R}}$, by setting $\mathcal{K} \approx_{\mathcal{R}} \alpha \mid\sim \beta$ if and only if $\mathcal{R} \models \alpha \mid\sim \beta$ [5]. Since $\mathcal{R}$ is a ranked interpretation, $\approx_{\mathcal{R}}$ corresponds to an *LM-rational entailment relation.* Note that $\approx_{\mathcal{R}}$ is syntactic sugar: if $\mathcal{K} \approx_{\mathcal{R}} \alpha \mid\sim \beta$, then $(\alpha, \beta) \in \mid\sim$, so a knowledge base together with all the defeasible implications it entails form an LM-rational consequence relation $\mid\sim$.

Having defined entailment for ranked interpretations, the authors in [10] introduce a specific entailment relation called *rank entailment*, denoted $\approx_R$. It is defined as follows: $\mathcal{K} \approx_R \alpha \mid\sim \beta$ if and only if $\alpha \mid\sim \beta$ is satisfied by **every** ranked model of $\mathcal{K}$. However, $\approx_R$ cannot in fact be represented by a ranked interpretation, so it is not LM-rational.

[3]This holds because if we want $\alpha$ to be true in all finite valuations in $\mathcal{R}$, we don't want any finite valuations to satisfy $\neg\alpha$. But no valuations satisfy $\bot$, so $\mathcal{R} \models \neg\alpha \mid\sim \bot$ if and only if no valuations satisfy $\neg\alpha$.

## 2.3 Rational Closure

The first entailment relation which was shown to be LM-rational is *rational closure.* We begin by defining an ordering $\prec$ over the ranked models of some knowledge base $\mathcal{K}$: $\mathcal{R}_1 \prec \mathcal{R}_2$ if and only if $\mathcal{R}_1(v) \leq \mathcal{R}_2(v), \forall v \in \mathcal{U}$, with the intuition that ranked models which are lower in the ordering are more typical, since the ranks they assign to valuations are lower [7]. Now, this ordering has a unique minimal element, denoted $\mathcal{R}^{RC}_{\mathcal{K}}$, which we use to define rational closure [7]. A defeasible implication $\alpha \mid\sim \beta$ is in the rational closure of $\mathcal{K}$, denoted $\mathcal{K} \approx_{RC} \alpha \mid\sim \beta$, if and only if $\mathcal{R}^{RC}_{\mathcal{K}} \models \alpha \mid\sim \beta$ [5]. Since $\approx_{RC}$ is defined by a single ranked interpretation $\mathcal{R}^{RC}_{\mathcal{K}}$, we conclude that is is LM-rational.

Given a knowledge base $\mathcal{K}$, one can also define rational closure in terms of the *base ranks* of the defeasible implications in $\mathcal{K}$. To begin, we define a formula $\alpha$ as *exceptional* with regards to $\mathcal{K}$ if and only if $\overrightarrow{\mathcal{K}} \models \neg\alpha$, where $\overrightarrow{\mathcal{K}}$ is called the *materialisation* of $\mathcal{K}$, and is defined as $\overrightarrow{\mathcal{K}} = \{\alpha \rightarrow \beta : \alpha \mid\sim \beta \in \mathcal{K}\}$.

Now let $\varepsilon(\mathcal{K}) := \{\alpha \mid\sim \beta \mid \overrightarrow{\mathcal{K}} \models \neg\alpha\}$. We can then define a non-increasing sequence of knowledge bases $\mathcal{E}^{\mathcal{K}}_0, \ldots, \mathcal{E}^{\mathcal{K}}_\infty$ as follows [5]:

$$\begin{aligned} \mathcal{E}^{\mathcal{K}}_0 &:= \mathcal{K} \\ \mathcal{E}^{\mathcal{K}}_i &:= \varepsilon(\mathcal{E}^{\mathcal{K}}_{i-1}) \ \text{ for } 0 < i < n \\ \mathcal{E}^{\mathcal{K}}_\infty &:= \mathcal{E}^{\mathcal{K}}_n \end{aligned}$$

where $n$ is the smallest $i$ such that $\mathcal{E}^{\mathcal{K}}_i = \mathcal{E}^{\mathcal{K}}_{i+1}$ (note that since $\mathcal{K}$ is finite, $n$ must exist [5]).

It is important to note that if $\alpha \in \mathcal{K}$ and $\alpha$ is a classical formula, then $\alpha$ will always be in $\mathcal{E}^{\mathcal{K}}_\infty$. Recall that a classical formula $\alpha \in \mathcal{L}$ can be expressed as the defeasible implication $\neg\alpha \mid\sim \bot$. Then, to check if $\alpha$ is exceptional, we have to check (by definition) if the negation of the antecedent of $\neg\alpha \mid\sim \bot$ is entailed by $\overrightarrow{\mathcal{K}}$. That is, we have to check if $\overrightarrow{\mathcal{K}} \models \alpha$, which is trivially true since $\alpha \in \mathcal{K}$.

Finally, the base rank of a formula $\alpha$ with regards to a knowledge base $\mathcal{K}$, denoted $br_{\mathcal{K}}(\alpha)$, is defined to be the smallest integer $r$ for which $\alpha$ is not exceptional with regards to the knowledge base $\mathcal{E}^{\mathcal{K}}_r$ [5]. That is to say, $br_{\mathcal{K}}(\alpha) := min\{r \mid \overrightarrow{\mathcal{E}^{\mathcal{K}}_r} \not\models \neg\alpha\}$. It is important to note that the base rank of a defeasible implication is defined to be equal to the base rank of its *antecedent*, so $br_{\mathcal{K}}(\alpha \mid\sim \beta) := br_{\mathcal{K}}(\alpha)$ [8].

We can now define rational closure in terms of base ranks [7]: $\mathcal{K} \approx_{RC} \alpha \mid\sim \beta$ if and only if $br_{\mathcal{K}}(\alpha) < br_{\mathcal{K}}(\alpha \wedge \neg\beta)$ or $br_{\mathcal{K}}(\alpha) = \infty$. Furthermore, there is a correlation between the base rank of a formula $\alpha$ in $\mathcal{K}$ and the rank of its models in $\mathcal{R}^{RC}_{\mathcal{K}}$ [7]: $br_{\mathcal{K}}(\alpha) = min\{\mathcal{R}^{RC}_{\mathcal{K}}(v) : v \in Mod(\alpha)\}$. In other words, the base rank of $\alpha$ is equal to the minimum rank of all the models of $\alpha$ with regards to the minimal ranked model $\mathcal{R}^{RC}_{\mathcal{K}}$ of $\mathcal{K}$.

Having defined the base rank of a formula, we consider two algorithms which have been developed in order to answer queries of the form "$\mathcal{K} \approx_{RC} \alpha \mid\sim \beta$?", and which follow directly from the definition of rational closure in terms of base ranks [5]. The first algorithm (and the one which we will focus on, as it pertains to the main ideas in this paper), `BaseRank`, takes a knowledge base $\mathcal{K}$ as input and returns a partition $(R_0, R_1, \ldots, R_{n-1}, R_\infty)$ of $\overrightarrow{\mathcal{K}}$ according to base rank, where $R_i = \{\alpha \rightarrow \beta : \alpha \mid\sim \beta \in \mathcal{K}, br_{\mathcal{K}}(\alpha) = i\}$. The algorithm initializes a variable $E_0$ with the materialisation of the knowledge base, $\overrightarrow{\mathcal{K}}$. It then repeatedly does the following, until $E_{i-1} = E_i$:

(1) It assigns all the formulas $\alpha \rightarrow \beta$ in $E_i$ which *are* exceptional to $E_{i+1}$.
(2) Then, it assigns the remaining formulas (those which *are not* exceptional) to $R_i$. This corresponds to the formulas which have $br_{\mathcal{K}}(\alpha \mid\sim \beta) = i$.
(3) Finally, it increases $i$ by 1.

Equivalently, and perhaps more intuitively, it does the following, until $\mathcal{K} = \emptyset$ or until only exceptional formulas remain:

(1) It assigns all the formulas in $\overrightarrow{\mathcal{K}}$ which *are not* exceptional to $R_i$.
(2) Then, it removes all the above-mentioned formulas from $\mathcal{K}$.
(3) Finally, it increases $i$ by 1.

The latter formulation of the algorithm has been adapted for the purposes of this paper, and is reflected in the design of `GuidedUserRank` and `UnguidedUserRank`.

**Algorithm 1:** `BaseRank`

```
Input: A knowledge base $\mathcal{K}$
Output: A ordered tuple $(R_0, \ldots, R_{n-1}, R_\infty, n)$
$i \leftarrow 0$;
$E_0 \leftarrow \overrightarrow{\mathcal{K}}$;
while $E_{i-1} \neq E_i$ do
    $E_{i+1} \leftarrow \{\alpha \rightarrow \beta \in E_i : E_i \models \neg\alpha\}$;
    $R_i \leftarrow E_i \setminus E_{i+1}$ ;
    $i \leftarrow i + 1$;
end
$R_\infty \leftarrow E_{i+1}$;
if $E_{i-1} \neq \emptyset$ then
    $n \leftarrow i - 1$;
else
    $n \leftarrow i$;
end
return $(R_0, \ldots, R_{n-1}, R_\infty, n)$
```

The second algorithm used for computing rational closure, called `RationalClosure`, takes as input a knowledge base $\mathcal{K}$, a defeasible implication $\alpha \mid\sim \beta$ and returns **true** if $\mathcal{K} \models\!\approx_{RC} \alpha \mid\sim \beta$ and **false** otherwise. It employs `BaseRank` to partition $\mathcal{K}$ into ranks according to base rank, and essentially checks if $br_{\mathcal{K}}(\alpha) < br_{\mathcal{K}}(\alpha \land \neg\beta)$, returning **true** if this is the case, and **false** otherwise.

## 2.4 Beyond Rational Closure

In [5], the authors introduce two new properties which an LM-rational entailment relation should follow:

- **Inclusion:** $\mathcal{K} \models\!\approx \alpha \mid\sim \beta$ for every $\alpha \mid\sim \beta \in \mathcal{K}$.
- **Classic Preservation:** $\mathcal{K} \models\!\approx \alpha \mid\sim \bot$ if and only if $\mathcal{K} \models\!\approx_R \alpha \mid\sim \bot$, where $\models\!\approx_R$ denotes rank entailment.

Then, an LM-rational entailment relation which also follows the properties of Inclusion and Classic Preservation is called a *basic defeasible entailment relation* [5].

The authors also generalize the notion of base rank by introducing a *$\mathcal{K}$-faithful rank function*, defined as follows [5]: let $r : \mathcal{L} \rightarrow \mathbb{N} \cup \{\infty\}$ be a rank function such that $r(\top) = 0$ satisfying the following convexity property: for every $i \in \mathbb{N}$, if $r(\alpha) = i$ then, for every $j$ such that $0 \leq j < i$, there exists a $\beta \in \mathcal{L}$ such that $r(\beta) = j$. Then, $r$ is $\mathcal{K}$-faithful if and only if it satisfies the following:

(1) It is *entailment preserving*. That is, $\alpha \models \beta \implies r(\alpha) \geq r(\beta)$.
(2) $r(\alpha) < r(\alpha \land \neg\beta)$ or $r(\alpha) = \infty$ for every $\alpha \mid\sim \beta \in \mathcal{K}$.
(3) $r(\alpha) = \infty$ if and only if $\mathcal{K} \models\!\approx_R \alpha \mid\sim \bot$.

Now, we have the following definition, reminiscent of base rank: $r$ generates a defeasible entailment relation $\models\!\approx$ whenever $\mathcal{K} \models\!\approx \alpha \mid\sim \beta$ if $r(\alpha) < r(\alpha \land \neg\beta)$ or $r(\alpha) = \infty$. Given that we can compute a defeasible entailment relation $\models\!\approx$ from a rank function $r$, we have the following important result [5]: if $r$ is a $\mathcal{K}$-faithful rank function, then the defeasible entailment relation it generates is a basic defeasible entailment relation.

The aim of this paper is to introduce algorithms which take a user-defined ranking of defeasible implications as input, and ensure that the corresponding rank function is $\mathcal{K}$-faithful and thus generates a basic defeasible entailment relation. We introduce two algorithms for doing so, and show that they do indeed result in a $\mathcal{K}$-faithful rank function. Furthermore, we show that they satisfy the maximum amount of user-defined preferences, and that they offer more than a refinement of base rank.

## 2.5 Related Work

The extensions of rational closure discussed were introduced in [5] and [4], and have set about a concrete framework for extending rational closure to the class of basic defeasible entailment relations (as well as *rational* entailment relations, though these are not considered here). Furthermore, the idea of computing entailment from an arbitrary ranking of defeasible implications has been explored in [3], albeit in the setting of description logic. We deviate from this paper in that we focus on ensuring that the resulting entailment relation is a basic defeasible entailment relation, rather than considering the various properties of the rankings themselves and the entailment relations which they generate.

## 3 ALGORITHM DESIGN

## 3.1 Valid Rank Functions

We begin with the following definition, which uses a generalization introduced in [5] of the observation that $br_{\mathcal{K}}(\alpha) = min\{\mathcal{R}^{RC}_{\mathcal{K}}(v) : v \in Mod(\alpha)\}$:

**Definition 1.** Given a knowledge base $\mathcal{K}$, a rank function $r$ is *valid* if and only if it corresponds to a ranked model $\mathcal{R}$ of $\mathcal{K}$. This correspondence is given by $r(\alpha) = min\{\mathcal{R}(v) : v \in Mod(\alpha)\}$, for any $\alpha \mid\sim \beta \in \mathcal{K}$.

Now, to understand why not all rank functions are valid, we consider the following example:

**Example 3.1.** Consider the knowledge base $\mathcal{K} = \{p \mid\!\sim b, b \mid\!\sim f, p \mid\!\sim \neg f\}$. Assume we (as the user) prefer the statement $b \mid\!\sim f$ over the statement $p \mid\!\sim \neg f$. That is to say, if our preferences were given by a valid rank function $r$ with corresponding ranked interpretation $\mathcal{R}$, then we would have $r(b \mid\!\sim f) > r(p \mid\!\sim \neg f)$, since a higher rank indicates a greater preference. This implies that $min\{\mathcal{R}(v) : v \in Mod(b)\} > min\{\mathcal{R}(v) : v \in Mod(p)\}$. That is to say, the minimal $b$ world has a higher rank in $\mathcal{R}$ than the minimal $p$ world.

But, in order to satisfy $\mathcal{K}$ (in particular, the defeasible implication $p \mid\!\sim b$), any ranked model of $\mathcal{K}$ must have that the minimal $p$ world satisfies $b$. So, if the minimal $b$ world has a higher rank in $\mathcal{R}$ than the minimal $p$ world, the minimal $p$ world must have that $b$ is false. Thus, the minimal $p$ world will not satisfy $b$, and $\mathcal{R}$ cannot satisfy $\mathcal{K}$. Therefore, our preference cannot be fulfilled, as this rank function $r$ does not correspond to a ranked model of $\mathcal{K}$, and is thus not valid.

Clearly, our algorithms need to ensure that the user preferences over $\mathcal{K}$ generate a valid rank function $r$. Therefore, we would like to consider what properties $r$ should follow in order to be deemed valid, and we will use these properties to guide the design of our algorithms.

While we do so, we will assume that our rank function $r$ assigns ranks to defeasible implications of the form $\alpha \mid\!\sim \beta$, and that $r(\alpha)$ is not defined. While we would like to have that $r(\alpha) = r(\alpha \mid\!\sim \beta)$, as in the case of base rank, this cannot automatically be assumed to be the case. For instance, consider the rank function $r$ in Example 3.1 above. We had, per the user's preferences, that $r(p \mid\!\sim \neg f) < r(b \mid\!\sim f)$. Now, since $r(p \mid\!\sim \neg f) = r(p \mid\!\sim b) = r(p \wedge b)$, the minimal $b$ world actually has a lower rank than the minimal $b \mid\!\sim f$ world, so $r(b) \neq r(b \mid\!\sim f)$. However, we note that a valid rank function $r$ should obviously have $r(\alpha) = r(\alpha \mid\!\sim \beta)$, but the assumption of such a fact would be assuming that $r$ is already valid.

Next, we introduce another definition:

**Definition 2.** Given a knowledge base $\mathcal{K}$, a valid rank function $r$ and corresponding ranked model $\mathcal{R}$, we say that a defeasible implication $\alpha \mid\!\sim \beta$ with $r(\alpha \mid\!\sim \beta) = i$ *requires* a valuation $v$ if $\mathcal{R}(v) = i$ and $v \models \alpha \wedge \beta$. In this case, we also say that $v$ *permits* $\alpha \mid\!\sim \beta$ to be on rank $i$, and refer to $v$ as $\alpha \mid\!\sim \beta$'s *permitting valuation*.

Intuitively, this definition states that for a defeasible implication $\alpha \mid\!\sim \beta$ to be given a rank of $i$ by $r$, there needs to be a valuation $v$ which has a rank of $i$ in $\mathcal{R}$ such that the valuation satisfies $\alpha \wedge \beta$. With this definition in hand, and given two defeasible implications $\alpha \mid\!\sim \beta, \gamma \mid\!\sim \sigma \in \mathcal{K}$, we consider three possibilities for how they relate to each other according to our rank function $r$:

(1) $r(\alpha \mid\!\sim \beta) < r(\gamma \mid\!\sim \sigma)$. In this case, assume $r(\alpha \mid\!\sim \beta) = i$, and let $R_i$ be the set of all defeasible implications of rank $i$. Then we must have that $\alpha \mid\!\sim \beta$ has a permitting valuation of rank $i$, and that this permitting valuation also satisfies $\neg\gamma$. That is to say, we must have $\exists v \in Mod(\wedge\overrightarrow{R_i})$ such that $v \models \alpha \wedge \neg\gamma$. Equivalently, we must have that $\overrightarrow{R_i} \not\models \neg(\alpha \wedge \neg\gamma)$. But $\neg(\alpha \wedge \neg\gamma) \equiv \neg\alpha \vee \gamma \equiv \alpha \rightarrow \gamma$, so we must have that $\overrightarrow{R_i} \not\models \alpha \rightarrow \gamma$.

(2) $r(\alpha \mid\!\sim \beta) = r(\gamma \mid\!\sim \sigma)$. Again, assume $r(\alpha \mid\!\sim \beta) = r(\gamma \mid\!\sim \sigma) = i$, and let $R_i$ be the set of all defeasible implications of rank $i$. Then we must have that both $\alpha \mid\!\sim \beta$ and $\gamma \mid\!\sim \sigma$ have permitting valuations on rank $i$. That is to say, we must have $\exists v, w \in Mod(\wedge\overrightarrow{R_i})$ such that $v \models \alpha$ and $w \models \gamma$ ($v$ and $w$ are not necessarily distinct). Equivalently, we must have that $\overrightarrow{R_i} \not\models \neg\alpha$ and $\overrightarrow{R_i} \not\models \neg\gamma$.

(3) $r(\alpha \mid\!\sim \beta) > r(\gamma \mid\!\sim \sigma)$. This case is analogous to the first, so we must have that $\overrightarrow{R_i} \not\models \gamma \rightarrow \alpha$. Note that now $i = r(\gamma \mid\!\sim \sigma)$.

Considering these cases, we have two basic requirements for our rank function, which can be formalized as a set of satisfiability checks:

(1) All defeasible implications which are in the same rank, say $R_i$, are not exceptional in that rank.

(2) For a defeasible implication $\alpha \mid\!\sim \beta$ to have a lower rank than another defeasible implication $\gamma \mid\!\sim \sigma$, the set of defeasible implications with the same rank as $\alpha \mid\!\sim \beta$ (more specifically, their materialisation) can not entail $\alpha \rightarrow \gamma$.

## 3.2 User Rank Algorithms

**Algorithm 2:** `GuidedUserRank`

```
Input: A knowledge base $\mathcal{K}$
Output: A $\mathcal{K}$-faithful ranking tuple $(R_0, R_1, \dots, R_\infty)$
$i \leftarrow 0$;
while $\mathcal{K} \neq \emptyset$ do
    $A \leftarrow \{\alpha \mid\!\sim \beta \in \mathcal{K} : \overrightarrow{\mathcal{K}} \not\models \neg\alpha\}$;
    if $A = \emptyset$ then
        $R_\infty = \mathcal{K}$;
        break;
    end
    // returns a subset of $A$
    // which the user wants to place on rank $i$
    $R_i \leftarrow$ UserPreferences($A$);
    for $\alpha \mid\!\sim \beta \in R_i$ do
        for $\gamma \mid\!\sim \sigma \in \mathcal{K} \setminus R_i$ do
            if $\overrightarrow{\mathcal{K}} \models \alpha \rightarrow \gamma$ then
                $R_i = R_i \setminus \{\alpha \mid\!\sim \beta\}$;
                break;
            end
        end
    end
    if $R_i \neq \emptyset$ then
        $\mathcal{K} \leftarrow \mathcal{K} \setminus R_i$;
        $i \leftarrow i + 1$;
    end
end
return $(R_0, R_1, \dots, R_\infty)$
```

We now provide two algorithms which take in a knowledge base $\mathcal{K}$ and return a ranking $(R_0, R_1, \dots, R_\infty)$, and which are designed with the aforementioned requirements in mind. The first algorithm we consider is `GuidedUserRank`, which is more restrictive with what it allows a user to put on a specific rank. It begins by repeatedly doing the following, until $\mathcal{K} = \emptyset$:

(1) Assign all the defeasible implications $\alpha \mid\!\sim \beta$ in $\mathcal{K}$ which are not exceptional, to the set $A$ (these are **A**llowed to be

**Algorithm 3:** UnguidedUserRank

```
Input: A knowledge base $\mathcal{K}$
Output: A $\mathcal{K}$-faithful ranking tuple $(R_0, R_1, \ldots, R_\infty)$
// user splits $\mathcal{K}$ into their preferred ranks
$(R_0, R_1, \ldots, R_\infty) \leftarrow$ UserPreferences($\mathcal{K}$);
$i \leftarrow 0$;
while $\mathcal{K} \neq \emptyset$ do
    $A \leftarrow \{\alpha \mid\sim \beta \in \mathcal{K} : \overrightarrow{\mathcal{K}} \not\models \neg\alpha\}$;
    if $A = \emptyset$ then
        $R_\infty = \mathcal{K}$;
        break;
    end
    $T \leftarrow R_i$;
    $R_i \leftarrow R_i \cap A$;
    for $\alpha \mid\sim \beta \in R_i$ do
        for $\gamma \mid\sim \sigma \in \mathcal{K} \setminus R_i$ do
            if $\overrightarrow{\mathcal{K}} \models \alpha \rightarrow \gamma$ then
                $R_i = R_i \setminus \{\alpha \mid\sim \beta\}$;
                break;
            end
        end
    end
    // Lift statements up to next rank
    $R_{i+1} \leftarrow R_{i+1} \cup (T \setminus R_i)$;
    if $R_i \neq \emptyset$ then
        $\mathcal{K} \leftarrow \mathcal{K} \setminus R_i$;
        $i \leftarrow i + 1$;
    else
        // Ensure tuple has no empty ranks
        $(\ldots, R_i, R_{i+1}, \ldots, R_{n-1}, R_\infty) =$
          $(\ldots, R_{i+1}, R_{i+2}, \ldots, R_n, R_\infty)$;
    end
end
return $(R_0, R_1, \ldots, R_\infty)$
```

on rank $i$). If there are no such defeasible implications, then we assign the remaining defeasible implications in $\mathcal{K}$ to $R_\infty$, and break.

(2) Allow the user to select which defeasible implications in $A$ they would like to assign a rank of $i$ to. These defeasible implications are assigned to $R_i$.
(3) For each of the defeasible implications $\alpha \mid\sim \beta$ in $R_i$, we check if any of the defeasible implications $\gamma \mid\sim \sigma$ which have a greater rank (those which are in the set $\mathcal{K} \setminus R_i$) satisfy the condition $\overrightarrow{\mathcal{K}} \models \alpha \rightarrow \gamma$. If they do, we remove $\alpha \mid\sim \beta$ from $R_i$.
(4) Finally, if $R_i$ is not empty (this can occur if the user selects no elements from $A$ to put on rank $i$, or if all of their choices are invalid and get removed), we remove the defeasible implications in $R_i$ from $\mathcal{K}$ and increase $i$ by 1.

Then, it returns the tuple $(R_0, R_1, \ldots, R_\infty)$, which represents a partition of $\mathcal{K}$ into ranks.

The second algorithm we consider is UnguidedUserRank, which takes a once off ranking from the user, and ensures that this ranking is valid, modifying it where appropriate. This algorithm bears many resemblances to GuidedUserRank and is, in fact, trivially equivalent. It begins by initialising $(R_0, R_1, \ldots, R_\infty)$ with a user-defined partition of $\mathcal{K}$. Then, it repeatedly does the following, until $\mathcal{K} = \emptyset$:

(1) Assign all the defeasible implications $\alpha \mid\sim \beta$ in $\mathcal{K}$ which are not exceptional, to the set $A$. If there are no such defeasible implications, then we assign the remaining defeasible implications in $\mathcal{K}$ to $R_\infty$, and break.
(2) Assign $R_i$ to a temporary variable $T$.
(3) Filter $R_i$ to include only those defeasible implications which are in $A$.
(4) For each of the defeasible implications $\alpha \mid\sim \beta$ in $R_i$, we check if any of the defeasible implications $\gamma \mid\sim \sigma$ which have a greater rank (those which are in the set $\mathcal{K} \setminus R_i$) satisfy the condition $\overrightarrow{\mathcal{K}} \models \alpha \rightarrow \gamma$. If they do, we remove $\alpha \mid\sim \beta$ from $R_i$.
(5) Add all the elements in $T$ that are not in $R_i$ to $R_{i+1}$. Here we are essentially lifting up all the defeasible implications which could not be assigned a rank of $i$, to rank $i + 1$.
(6) Finally, check if $R_i$ is not empty (this can occur if all of the user's choices for $R_i$ are invalid and get removed). If it is not empty, we remove the defeasible implications in $R_i$ from $\mathcal{K}$, and increase $i$ by 1. Otherwise, we remove $R_i$ from our ranking tuple, essentially dropping all the higher ranks by 1. This is done because we can not have any empty ranks.

Then, it returns the tuple $(R_0, R_1, \ldots, R_\infty)$, which represents a partition of $\mathcal{K}$ into ranks.

Both of these algorithms adhere to the requirements we identified for a valid rank function. In GuidedUserRank, we only allow the user to rank defeasible implications in the set $A$ and in UnguidedUserRank we only allow the user's $R_i$ to contain defeasible implications which are also in $A$. Furthermore, in GuidedUserRank and UnguidedUserRank we ensure that if a defeasible implication $\alpha \mid\sim \beta$ has a lower rank than another defeasible implication $\gamma \mid\sim \sigma$, it cannot be the case that $\overrightarrow{\mathcal{K}} \models \alpha \rightarrow \gamma$. These two requirements being satisfied will allow us to prove that the provided algorithms result in a $\mathcal{K}$-faithful ranking.

However, there are two issues to address. Firstly, it may be the case that enforcing that the user selects only defeasible implications from $A$ is too restrictive. That is to say, there may be defeasible implications which can be put on rank $i$ that are not in $A$. Secondly, both algorithms share a resemblance to BaseRank (specifically, the set $A$ in GuidedUserRank and UnguidedUserRank is equivalent to $R_i$ in BaseRank), which may lead one to think that the resulting rankings are simply *refinements* of $br_{\mathcal{K}}$. We resolve both these issues by showing that the algorithms satisfy as many user preferences as possible (that is, $A$ is maximal) and by giving an example of how they extend $br_{\mathcal{K}}$, rather than refining it.

We begin by proving the following:

Theorem 1. *Let $\mathcal{K}$ be a knowledge base, $r$ a valid rank function, and $R$ a partition of $\mathcal{K}$ into ranks given by $R = (R_0, R_1, \ldots, R_n)$, where $R_i = \{\alpha \mid\sim \beta : r(\alpha \mid\sim \beta) = i\}$. Let the set $A$ be defined as $A = \{\alpha \mid\sim \beta : \bigcup_{k=i}^{n} \overrightarrow{R_k} \not\models \neg\alpha\}$. Then $A$ is the largest set which contains exactly all the defeasible implications which can be assigned a rank of $i$.*

Proof. Let $r$ be a valid rank function and let $\mathcal{R}$ represent it's associated ranked interpretation. Assume, for the sake of contradiction, that there exists a set $V \supset A$ such that $V$ contains exactly all the defeasible implications which can be assigned a rank of $i$.

From this assumption, it follows that there exists some $\alpha \mid\sim \beta \in V \setminus A$ with $r(\alpha \mid\sim \beta) = i$. Since $r$ is valid, there must exist a permitting valuation $v$ in rank $i$ of $\mathcal{R}$ such that $v \models \alpha$.

Given $\alpha \mid\sim \beta \notin A$, we must have that $\bigcup_{k=i}^{n} \overrightarrow{R_k} \models \neg\alpha$, by the definition of $A$. However, since $v \models \alpha$, we deduce that $v \not\models \bigwedge \bigcup_{k=i}^{n} \overrightarrow{R_k}$. Therefore, there exists a defeasible implication $\gamma \mid\sim \sigma \in \bigcup_{k=i}^{n} R_k$ for which $v \not\models \gamma \rightarrow \sigma$. That is to say, in the valuation $v$, $\gamma$ is true and $\sigma$ is false (this is the only assignment of $\gamma$ and $\sigma$ for which $\gamma \rightarrow \sigma$ is false).

Now, since $\gamma \mid\sim \sigma \in \bigcup_{k=i}^{n} R_k$, it follows that $r(\gamma \mid\sim \sigma) \geq i = \mathcal{R}(v)$, so $v$ is a minimal $\gamma$ world. But any minimal $\gamma$ world must have $\sigma$ true, which $v$ does not. Thus there exists no defeasible implication $\alpha \mid\sim \beta \in V \setminus A$, a contradiction. Thus, we conclude that $A$ is the largest set which contains exactly all the defeasible implications which can be assigned a rank of $i$. □

Next, we introduce the following definition:

**Definition 3.** We define a rank function $r$ as *refining* base rank (also called *rank preserving* in [5]) if, for all $\alpha, \beta \in \mathcal{L}$, whenever $br_{\mathcal{K}}(\alpha) < br_{\mathcal{K}}(\beta)$, we have $r(\alpha) < r(\beta)$.

Given this definition, we show that the ranking returned by `GuidedUserRank` and `UnguidedUserRank` is not necessarily a refinement of base rank, by considering the following example:

**Example 3.2.** Let $\mathcal{K} = \{q \mid\sim u, x \mid\sim q, l \mid\sim \neg u, i \mid\sim u \wedge l\}$ be a knowledge base. Given $\mathcal{K}$ as input, `GuidedUserRank` will allow us to rank $A = \{q \mid\sim u, x \mid\sim q, l \mid\sim \neg u\}$ on rank 0. Assume we select $l \mid\sim \neg u$ to be on rank 0. Then, when $i = 1$, $A = \{q \mid\sim u, x \mid\sim q, i \mid\sim u \wedge l\}$. Assume we select $i \mid\sim u \wedge l$ to be on rank 1. Finally, when $i = 2$, we will have that $A = \{q \mid\sim u, x \mid\sim q\}$. Assume we select both of these to be on rank 2. Then `GuidedUserRank` will terminate, returning the following ranking: $(\{l \mid\sim \neg u\}, \{i \mid\sim u \wedge l\}, \{q \mid\sim u, x \mid\sim q\})$. `BaseRank`, on the other hand, returns $(\{q \mid\sim u, x \mid\sim q, l \mid\sim \neg u\}, \{i \mid\sim u \wedge l\})$. Clearly, $br_{\mathcal{K}}(q \mid\sim u) < br_{\mathcal{K}}(i \mid\sim u \wedge l)$, while in `GuidedUserRank`, we can have the opposite, $r(q \mid\sim u) > r(i \mid\sim u \wedge l)$. Therefore, `GuidedUserRank` and `UnguidedUserRank` can offer more than a refinement of `BaseRank`.

For interest sake, the reason we can place $i \mid\sim u \wedge l$ below $q \mid\sim u$ is because when the user places $l \mid\sim \neg u$ on rank 0, this defeasible implication is removed from $\mathcal{K}$ and when $A$ is calculated for $i = 1$, $i \mid\sim u \wedge l$ is no longer exceptional in $\mathcal{K}$ and can now be placed on rank 1 (whereas it couldn't while $l \mid\sim \neg u$ was still in $\mathcal{K}$ because any valuation satisfying $l \mid\sim \neg u$ necessarily has $l$ true and $u$ false, so can not satisfy $u \wedge l$, and thus can not satisfy $i \mid\sim u \wedge l$).

We have now designed two algorithms which we believe to be valid, in terms of the requirements we set out earlier. We now validate two important results about both algorithms. First, we show that the ranking they generate satisfies *minimal rank preservation*. Secondly, we show that the ranking they generate is also $\mathcal{K}$-faithful.

## 4 ALGORITHM VALIDATION

For the purposes of this section, we will refer to `GuidedUserRank` and `UnguidedUserRank` collectively as `UserRank`, since they are equivalent. Also, we refer to the first for loop variable in `UserRank` ($\alpha \mid\sim \beta$) as *lo* and we refer to the second for loop variable in `UserRank` ($\gamma \mid\sim \sigma$) as *hi*. Finally, we denote the rank function returned by `UserRank` given $\mathcal{K}$ as input as $r_{\mathsf{UR}}^{\mathcal{K}}$ and define $r_{\mathsf{UR}}^{\mathcal{K}}(\alpha \mid\sim \beta) = i$ if and only if $\alpha \mid\sim \beta \in R_i$, where $(R_0, R_1, \ldots, R_i, \ldots, R_\infty)$ is the output of `UserRank`.

### 4.1 Minimal Rank Preservation

Given a rank function $r$ with corresponding ranked interpretation $\mathcal{R}$, we have mentioned that it can not be taken as fact that the rank assigned to a defeasible implication $\alpha \mid\sim \beta$ by $r$ necessarily corresponds to the rank of the minimal $\alpha$ world in $\mathcal{R}$, even though it should. Therefore, we define a rank function $r$ with corresponding ranked interpretation $\mathcal{R}$ as being *minimal rank preserving* if and only if $r(\alpha \mid\sim \beta) = min\{\mathcal{R}(v) : v \in Mod(\alpha)\}$. That is to say, the rank $r$ assigns to any defeasible implication with $\alpha$ as the antecedent corresponds to the rank of the minimal $\alpha$ world in $\mathcal{R}$. If $r$ is minimal rank preserving, then we have that $r(\alpha \mid\sim \beta) = r(\alpha)$, since $r(\alpha)$ is considered to be the rank of the minimal $\alpha$ world in $\mathcal{R}$. We would like to prove that $r_{\mathsf{UR}}^{\mathcal{K}}$ is minimal rank preserving.

We begin by proving that if a defeasible implication $\gamma \mid\sim \sigma$ requires $\alpha$ to be true, then it's rank is greater than or equal to the rank of any defeasible implication with $\alpha$ as the antecedent, according to `UserRank`.

Lemma 1. *Let $\mathcal{K}$ be a knowledge base and let $\gamma \mid\sim \sigma, \alpha \mid\sim \beta \in \mathcal{K}$ be defeasible implications such that $r_{\mathsf{UR}}^{\mathcal{K}}(\gamma \mid\sim \sigma) = j$ and $r_{\mathsf{UR}}^{\mathcal{K}}(\alpha \mid\sim \beta) = k$. Since $r_{\mathsf{UR}}^{\mathcal{K}}(\gamma \mid\sim \sigma) = j$, when the iteration variable $i = j$ in* `UserRank`, *there exist one or more valuations $v_1, v_2, \ldots, v_n \in Mod(\bigwedge \overrightarrow{\mathcal{K}})$ such that $v_i \models \gamma, 1 \leq i \leq n$. If all valuations $v_i, 1 \leq i \leq n$, also satisfy $\alpha$, then $k \leq j$.*

Proof. Assume, for the sake of contradiction, that all valuations $v_i, 1 \leq i \leq n$, satisfy $\alpha$, and $k > j$. Then when the iteration variable $i = j$ in `UserRank`, $\alpha \mid\sim \beta \in \mathcal{K} \setminus R_i$ and $\gamma \mid\sim \sigma \in R_i$, so we will eventually have that $lo = \gamma \mid\sim \sigma$ and $hi = \alpha \mid\sim \beta$, and we will evaluate if $\overrightarrow{\mathcal{K}} \models \gamma \rightarrow \alpha$ is true. Let $v \in Mod(\bigwedge \overrightarrow{\mathcal{K}})$ be a valuation. Then, there are two cases to consider.
**Case 1:** Assume $v \models \gamma$. We assumed that such valuations also satisfy $\alpha$, so $v \models \gamma \rightarrow \alpha$.
**Case 2:** Assume $v \models \neg\gamma$. Then $v \models \gamma \rightarrow \alpha$ trivially.
In both cases, $v \models \gamma \rightarrow \alpha$, so $\overrightarrow{\mathcal{K}} \models \gamma \rightarrow \alpha$ will evaluate to true. Thus, according to `UserRank`, $R_i = R_i \setminus \gamma \mid\sim \sigma \iff \gamma \mid\sim \sigma \notin R_i$, a contradiction. □

Clearly then, we have the following result:

Corollary 1. *Let $\mathcal{K}$ be a knowledge base and let $\alpha \mid\sim \beta, \gamma \mid\sim \sigma \in \mathcal{K}$ be defeasible implications. If $\gamma \models \alpha$, then $r_{\mathsf{UR}}^{\mathcal{K}}(\gamma \mid\sim \sigma) \geq r_{\mathsf{UR}}^{\mathcal{K}}(\alpha \mid\sim \beta)$. That is to say, $r_{\mathsf{UR}}^{\mathcal{K}}$ is entailment preserving.*

Proof. In Lemma 1, we showed that if $\gamma \models \alpha$ holds in valuations $v_1, v_2, \ldots, v_n \in Mod(\bigwedge \overrightarrow{\mathcal{K}})$, then the result follows. So, if $\gamma \models \alpha$, then $\gamma \models \alpha$ holds in all valuations, so the result follows. □

Now, we know that any defeasible implication $\gamma \mid\sim \sigma$ which requires $\alpha$ to be true will have a higher rank than any defeasible implication of the form $\alpha \mid\sim \beta$. Next, we would like to show that all defeasible implications of this form have the same rank.

Lemma 2. *Let $\mathcal{K}$ be a knowledge base and let $\alpha \mid\sim \beta, \alpha \mid\sim \gamma \in \mathcal{K}$ be defeasible implications with the same antecedent, $\alpha$. Then we have that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \gamma)$.*

Proof. Assume, without loss of generality, that $j = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) < r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \gamma)$. When the iteration variable $i = j$ in `UserRank`, we must have that $\alpha \mid\sim \beta \in A$. But $\alpha \mid\sim \beta$ and $\alpha \mid\sim \gamma$ have the same antecedent, so we must have $\alpha \mid\sim \gamma \in A$ as well. Therefore, we must have that $\alpha \mid\sim \beta \in R_i$ and $\alpha \mid\sim \gamma \notin R_i$ follow from the user preferences; so $\alpha \mid\sim \beta \in R_i$ and $\alpha \mid\sim \gamma \in \mathcal{K} \setminus R_i$. Thus, we will eventually have that $lo = \alpha \mid\sim \beta$ and $hi = \alpha \mid\sim \gamma$, and we will evaluate if $\overrightarrow{\mathcal{K}} \models \alpha \to \alpha$. This will evaluate to true since $(\alpha \to \alpha) \equiv \top$, so, trivially, $\overrightarrow{\mathcal{K}} \models \alpha \to \alpha$. Thus, according to `UserRank`, $R_i = R_i \setminus \alpha \mid\sim \beta \iff \alpha \mid\sim \beta \notin R_i$, a contradiction. □

Theorem 2. *Let $\mathcal{K}$ be a knowledge base and $\alpha \mid\sim \beta \in \mathcal{K}$ a defeasible implication. Then, $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta)$ is equal to the rank of the minimal $\alpha$ world in the corresponding ranked interpretation $\mathcal{R}$. That is to say, $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = min\{\mathcal{R}(v) : v \in Mod(\alpha)\}$. Therefore, we adopt the convention that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta)$, and conclude that $r^{\mathcal{K}}_{\mathsf{UR}}$ is minimal rank preserving.*

Proof. From Lemma 1, we have that any defeasible implication $\gamma \mid\sim \sigma$ which requires (as per Definition 2) a valuation $v$ such that $v \models \alpha$, has a rank greater than or equal to any defeasible implication $\alpha \mid\sim \beta$. Furthermore, from Lemma 2, we have that all defeasible implications which have the same antecedent $\alpha$ have the same rank. Therefore, if $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = i$, we cannot possibly have any defeasible implication with a rank lower than $i$ requiring $\alpha$ to be true. That is to say, the minimal $\alpha$ world has rank $i = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta)$. □

## 4.2 $\mathcal{K}$-faithfulness

$\mathcal{K}$-faithfulness is a reasonable property for a rank function to possess, so we would like to prove that $r^{\mathcal{K}}_{\mathsf{UR}}$ is $\mathcal{K}$-faithful. We begin by proving the following two lemmas:

Lemma 3. *Let $\mathcal{K}$ be a knowledge base, and let $\alpha \mid\sim \beta \in \mathcal{K}$ be a defeasible implication. Then, either $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) < r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta)$ or $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$.*

Proof. Let $\alpha \mid\sim \beta \in \mathcal{K}$ be a defeasible implication, and assume $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) \neq \infty$. Then we want to show that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) < r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta)$. Assume, for the sake of contradiction, that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = j$ and $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta) = r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta \mid\sim \gamma) = k$, for some $\gamma \in \mathcal{L}$, and that $j \geq k$.

Then, consider when the iteration variable $i = k$ in `UserRank`. We will have that $\alpha \wedge \neg\beta \mid\sim \gamma \in R_i$. Thus, it must be the case that $\overrightarrow{\mathcal{K}} \not\models \neg(\alpha \wedge \neg\beta)$. That is to say, $\overrightarrow{\mathcal{K}} \not\models \alpha \to \beta$.

But we assumed that $j \geq k$, so $\alpha \mid\sim \beta \in \mathcal{K}$ (if $\alpha \mid\sim \beta \notin \mathcal{K}$, this would imply $\alpha \mid\sim \beta$ has already been placed in a lower rank, so $j < k$). Therefore, $\alpha \to \beta \in \overrightarrow{\mathcal{K}}$ and, trivially, $\overrightarrow{\mathcal{K}} \models \alpha \to \beta$, a contradiction. Thus, we conclude that $j < k$. That is to say, $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) < r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta)$.

□

Lemma 4. *Let $\mathcal{K}$ be a knowledge base, and let $\alpha \mid\sim \beta \in \mathcal{K}$ be a defeasible implication. Then $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$ if and only if $\mathcal{K} \mathrel{|\!\!\approx_R} \alpha \mid\sim \bot$, where $\mathrel{|\!\!\approx_R}$ denotes rank entailment.*

Proof. We first prove the forward implication. Assume that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$, and let $n = max\{\{r^{\mathcal{K}}_{\mathsf{UR}}(\gamma \mid\sim \sigma) : \gamma \mid\sim \sigma \in \mathcal{K}\} \setminus \{\infty\}\}$. Let $V_k$ denote the set $Mod(\bigwedge \overrightarrow{\mathcal{K}})$ when the iteration variable $i = k$.

Then, when the iteration variable $i = n + 1$, we must have that $A = \emptyset$ and $\alpha \mid\sim \beta \in \mathcal{K}$ (since $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = \infty$). Therefore, $\alpha \mid\sim \beta \notin A$, implying that $\overrightarrow{\mathcal{K}} \models \neg\alpha$, by the definition of $A$. Thus, for all valuations $v \in V_{n+1}$, we have that $v \models \neg\alpha$. But $V_0 \subset \cdots \subset V_{n+1}$, so $v$ satisfies $\neg\alpha$, for all $v \in V_i, 0 \leq i \leq n + 1$. Therefore, when a defeasible implication is placed on rank $i$, the valuation $w$ which permits it to be on rank $i$ (as per Definition 2) satisfies $\neg\alpha$. That is to say, in any ranked model $\mathcal{R}$ of $\mathcal{K}$, any valuation $v$ with $\mathcal{R}(v) \leq n$ (that is, $\mathcal{R}(v) \neq \infty$) satisfies $\neg\alpha$. Thus, $\mathcal{R} \models \alpha \mid\sim \bot$ for any ranked model of $\mathcal{K}$, so $\mathcal{K} \mathrel{|\!\!\approx_R} \alpha \mid\sim \bot$.

Now, to prove the reverse implication, assume that $\mathcal{K} \mathrel{|\!\!\approx_R} \alpha \mid\sim \bot$, and that $n = max\{\{r^{\mathcal{K}}_{\mathsf{UR}}(\gamma \mid\sim \sigma) : \gamma \mid\sim \sigma \in \mathcal{K}\} \setminus \{\infty\}\}$. Let $V_k$ denote the set $Mod(\bigwedge \overrightarrow{\mathcal{K}})$ when the iteration variable $i = k$.

Then, in any ranked model $\mathcal{R}$ of $\mathcal{K}$, all valuations $v$ with $\mathcal{R}(v) \neq \infty$ satisfy $\neg\alpha$. Thus, for all valuations $v \in V_i, 0 \leq i \leq n + 1, v \models \neg\alpha$. Thus, $\alpha \mid\sim \beta \notin A$ for all iterations. Thus, when $i = n + 1$, $\alpha \mid\sim \beta \in \mathcal{K}$ and $A = \emptyset$, so $\alpha \mid\sim \beta \in R_\infty$. That is to say, $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$. □

Theorem 3. *$r^{\mathcal{K}}_{\mathsf{UR}}$, as defined by `UserRank`, is $\mathcal{K}$-faithful.*

Proof. From Corollary 1, we have that $r^{\mathcal{K}}_{\mathsf{UR}}$ is entailment preserving. Furthermore, we have from Lemma 3 that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) < r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \wedge \neg\beta)$ or $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$ for every $\alpha \mid\sim \beta \in \mathcal{K}$. Also, from Lemma 4, we have that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = \infty$ if and only if $\mathcal{K} \mathrel{|\!\!\approx_R} \alpha \mid\sim \bot$. Thus, it remains to show that $r^{\mathcal{K}}_{\mathsf{UR}}(\top) = 0$ and that $r^{\mathcal{K}}_{\mathsf{UR}}$ satisfies the following convexity property: for every $i \in \mathbb{N}$, if $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha) = i$ then, for every $j$ such that $0 \leq j < i$, there exists a $\beta \in \mathcal{L}$ such that $r^{\mathcal{K}}_{\mathsf{UR}}(\beta) = j$.

Let $\top \mid\sim \alpha \in \mathcal{K}$. Then when the iteration variable $i = 0$, $\top \mid\sim \alpha \in A$ since $\overrightarrow{\mathcal{K}} \not\models \bot$ trivially. Assume that $\top \mid\sim \alpha \notin R_i$. So, given any defeasible implication $lo = \gamma \mid\sim \sigma$ and $hi = \top \mid\sim \alpha$, we will check if $\gamma \to \top$, which is trivially true. Therefore, $\gamma \mid\sim \sigma$ (any defeasible implication $\gamma \mid\sim \sigma \in R_i$) will be removed from $R_i$, so $R_i = \emptyset$ at the end of the iteration. Therefore, $i$ will not be incremented. Furthermore, $i$ will not be incremented until $\top \mid\sim \alpha$ is added to $R_i = R_0$ by the user. Therefore, $r^{\mathcal{K}}_{\mathsf{UR}}(\top \mid\sim \alpha) = 0$.

Lastly, note that we check if $R_i = \emptyset$ for each value of $i$. Therefore, $\exists\alpha \mid\sim \beta \in \mathcal{K}$ such that $r^{\mathcal{K}}_{\mathsf{UR}}(\alpha \mid\sim \beta) = i, 0 \leq i \leq n$ where $n = max\{\{r^{\mathcal{K}}_{\mathsf{UR}}(\gamma \mid\sim \sigma) : \gamma \mid\sim \sigma \in \mathcal{K}\} \setminus \{\infty\}\}$. Therefore, $r^{\mathcal{K}}_{\mathsf{UR}}$ satisfies the stated convexity property, and is thus $\mathcal{K}$-faithful. □

## 5 IMPLEMENTATION AND ANALYSIS

### 5.1 Implementation Details

Although the focus of this paper lies predominantly in theoretical contributions, we recognize the value in constructing a small-scale software component. This software serves as a practical testing environment for validating the theoretical constructs surrounding the algorithms discussed in this paper. Furthermore, it serves as a benchmarking platform by which the new algorithms can be compared with existing methods.

The software is written in Python and leverages the PyEDA [6] library for handling Boolean algebra and logical expressions. This choice was motivated by Python's wide use in academic research and the comprehensive capabilities of the PyEDA library for Boolean operations, which are essential for the algorithms presented.

In terms of functionality, the software provides a command line interface for ranking defeasible implication within a knowledge base. It reads (via standard input or a file) these defeasible implications and allows the user to rank them according to their preferences. For this, we provide two separate but similar ranking algorithms.

The first algorithm, `UnguidedUserRank`, provides the user with a list of all defeasible implications in the knowledge base and asks them to assign a rank to each in one go. This allows for an open-ended ranking process but may require the user to have a comprehensive understanding of the entire knowledge base.

The second algorithm, `GuidedUserRank`, iteratively presents a subset of defeasible implications to the user (those which are logically allowed to be placed on that rank), asking them to choose which defeasible implications to include in the current rank. This step-by-step approach provides a more guided experience and could be helpful for users who are not familiar with the entire knowledge base.

Both algorithms employ logical entailment checks to validate the user's ranking choices, and are performed using an internal utility function for entailment. If a user attempts to rank a defeasible implication higher than another defeasible implication that logically entails it, the software will flag this as an invalid choice.

In this case, both algorithms will remove the lower ranked defeasible implication from the current rank. In the unguided ranking algorithm, the lower ranked defeasible implication will simply be lifted up to the next rank, while in the guided ranking algorithm, the user will be prompted to revise their choices for that rank, and the rank will not be incremented. The guided ranking algorithm will also inform the user which lower ranked defeasible implication could not be ranked, and which higher ranked defeasible implication was prohibiting the ranking (these correspond to the variables $lo$ and $hi$ in `UserRank`, respectively).

In summary, the software acts as a companion testing ground for the theoretical concepts explored in this paper and offers a baseline for comparing the performance and utility of the newly developed ranking algorithms against existing methods, such as `BaseRank`.

## 5.2 Theoretical Analysis

The theoretical analysis focuses on comparing the developed algorithm, `UserRank`, with `BaseRank`. To this end, we gauge the performance of `UserRank` based on the `UnguidedUserRank` algorithm, as it does not require continuous user input; while this is not necessary for the theoretical analysis, it is necessary for the experimental analysis.

We begin by noting that the critical operation in both algorithms is a subroutine called `entails`, which performs a single satisfiability check. We denote its worst-case time complexity with $T_{SAT}(n)$, where $n$ is the number of variables in the SAT problem.

Now, in `BaseRank`, the outer while loop runs at most $N$ times, and within each iteration, the `entails` subroutine is called at most $N$ times, once for each formula in $E_i$. Thus, the time complexity for each iteration becomes $O(N \times T_{SAT}(n))$. Therefore, the overall time complexity of `BaseRank` is $O(N^2 \times T_{SAT}(n))$.

`UserRank` is similar but includes additional nested for loops, which are necessary to validate the user's preferences. The outer while loop runs $N$ times, and the first set of `entails` operations contributes $O(N \times T_{SAT}(n))$ to the time complexity. The nested for loops run $k \times (N - k)$ times, where $k$ denotes the size of $R_i$, and $0 < k < N$, and call the `entails` subroutine once for each iteration. This results in a worst-case time complexity of $O(N^2 \times T_{SAT}(n))$ for the nested loops. Combining these, the overall time complexity for `UserRank` is $O(N^3 \times T_{SAT}(n))$.

Space complexity primarily depends on the storage of intermediate results and data structures. For `BaseRank`, two lists $E$ and $R$ are maintained, both of which can grow to size $N$. Thus, the space complexity is $O(N)$. In `UserRank`, aside from the user-defined ranking $R$, temporary lists are also used, but the size remains proportional to $N$. Therefore, the space complexity is also $O(N)$.

In summary, `BaseRank` is computationally more efficient with a time complexity of $O(N^2 \times T_{SAT}(n))$ compared to `UserRank`'s $O(N^3 \times T_{SAT}(n))$. However, `UserRank` may offer more intuitive insights by allowing for user-defined preferences, albeit at the cost of increased computational complexity. Furthermore, it can be argued that the nested for loops are absolutely necessary for validating the user preferences, thus a more efficient algorithm is not immediately obvious.

## 5.3 Experimental Analysis

The primary objective of the experimental analysis was to assess and compare the real-world performance of `BaseRank` and `UserRank`, both of which aim to compute a ranking for a given knowledge base $K$ of size $N$. The investigation primarily aims to investigate how these algorithms perform in terms of execution time and whether the theoretical time complexities hold in practice.

To this end, the experiments were designed to capture the average execution time of both algorithms across various sizes of knowledge bases with different structures (that is, various numbers of ranks), which were generated using a team member's Defeasible Knowledge Base Generator. This tool offered an easy way to specify how many formulas should be in the knowledge base, as well as how many ranks the knowledge base should be partitioned into, according to `BaseRank`.

To evaluate the performance of `UserRank`, we used the Python implementation of `UnguidedUserRank`, as it does not require continuous user input, and can in fact be instantiated with a ranking beforehand to ensure a fair comparison. Therefore, we evaluated `UserRank` with two initialisations of the ranking $R$:

(1) $R = (\mathcal{K})$. This initialisation places all formulas of $\mathcal{K}$ on to the first rank (rank 0). In fact, this is equivalent to `BaseRank`, as each iteration of the while loop places all non-exceptional

formulas on that rank, while leaving the exceptional formulas for the next iteration. This initialisation was chosen to directly compare UserRank and BaseRank. In the results, this initialisation of the algorithm is referred to simply as User Rank.

(2) $R$ = RandomPartition($\mathcal{K}$). This initialisation randomly partitioned $\mathcal{K}$ into various ranks, and was chosen to represent a more realistic scenario of the algorithm's use. In the results, this initialisation of the algorithm is referred to as Random User Rank.

Both of these initialisations were compared to BaseRank, and all three were run with knowledge bases of the following sizes (denoted N in the results): 50,100,150,200,250,300 and the following number of ranks (denoted R in the results): 10,20,50. The mean execution time, computed over multiple experimental runs, served as the principal metric for assessing algorithmic performance. Each experiment was run on an Apple MacBook Pro with an M1 chip (8 cores) and 8GB RAM.

First, we consider how BaseRank compares to User Rank, the results of which are presented in Figure 2. Consistent with the theoretical time complexities, BaseRank outperformed User Rank by a significant margin across all data points. While R made a slight difference in execution time, it was not significant, especially for User Rank where the execution time of the knowledge base with N = 300, R = 20 was actually slightly higher than the knowledge base with N = 300, R = 50. Across the various values for N, however, we observed that BaseRank outperformed User Rank by an average factor of 10.59 (and an average factor of 0.06×N).

Next, we consider how BaseRank compares to Random User Rank, the results of which are presented in Figure 3. Again, BaseRank outperformed Random User Rank by a significant margin across all data points. This time, R also made a larger difference. Random User Rank, when given a knowledge base with 50 ranks, ran on average 1.81 times slower than when it was given a knowledge base with 10 ranks (in contrast to only 1.33 times slower with User Rank). Furthermore, across the various values for N, we observed that BaseRank outperformed Random User Rank by an average factor of 12.54 (and an average factor of 0.07×N).

Clearly then, User Rank outperformed Random User Rank. However, most of the difference in their performance is owed specifically to the case when R = 50, as one can see in Figure 4. Besides this case, User Rank and Random User Rank perform quite similarly (the latter was on average 1.20 times slower when R = 10 and 20, while it was 1.55 times slower when R = 50). 4

In summary, the experimental analysis substantiated the theoretical time complexities and offered valuable insights into the practical performance of BaseRank and UserRank. While BaseRank proved to be computationally more efficient, it is important to note that UserRank provides the flexibility of accommodating user-defined rankings. This trade-off between efficiency and flexibility becomes crucial depending on the specific application and computational resources available.

## 6 CONCLUSIONS

In this paper, we have introduced two novel algorithms designed to incorporate user preferences into knowledge base ranking. The primary goal of these algorithms, collectively referred to as UserRank, is to ensure the resulting rank function is both valid and $\mathcal{K}$-faithful. Our motivation stems from the recognized limitations of existing methods, particularly BaseRank, which may fall short in capturing a user's nuanced insights about a knowledge base. By integrating user preferences, we aim to offer a more comprehensive representation of these valuable insights.

Our research demonstrates that these algorithms effectively honor a maximum number of user preferences, resulting in rank functions that are not only minimal rank preserving but also $\mathcal{K}$-faithful. Furthermore, we have provided a proof of concept through a small-scale practical implementation, illustrating how user preferences can be gathered and integrated into the ranking process.

In a comparative analysis with BaseRank, UserRank exhibits a noticeable computational overhead, both theoretically and practically. However, it's important to highlight that UserRank offers a critical advantage in terms of flexibility by accommodating user-defined rankings. Particularly, for small knowledge bases where user-defined rankings are most likely to be useful, the difference in execution time between BaseRank and UserRank becomes negligible.

## 7 FUTURE WORK

Looking ahead, future work could focus on optimizing the computational efficiency of these algorithms. One avenue of exploration could involve the optimization of the nested for loop within UserRank, potentially incorporating memoization for entailment checks. Additionally, the practical implementation could find valuable applications in cognitive defeasible reasoning, following the approach outlined in [1]. Finally, the development of more intuitive methods for ranking defeasible implication remains an intriguing direction for further research, given that only two basic approaches have been introduced in this paper.

In conclusion, our work not only introduces innovative algorithms for user preference-driven knowledge base ranking but also sheds light on the trade-offs between computational efficiency and user-defined flexibility. These findings open up exciting possibilities for future research and applications in the field of knowledge representation and reasoning.

## 8 ACKNOWLEDGEMENTS

I wish to extend my sincere thanks to my supervisor, Tommie Meyer, for his invaluable guidance and support throughout this project. His expertise has been instrumental in shaping this research, and has sparked a strong personal interest in the field of Knowledge Representation and Reasoning.

## REFERENCES

[1] Baker, C. K., Denny, C., Freund, P., and Meyer, T. Cognitive defeasible reasoning: the extent to which forms of defeasible reasoning correspond with human reasoning. In *Southern African Conference for Artificial Intelligence Research* (2020), Springer, pp. 199–219.

[2] Ben-Ari, M. *Mathematical logic for computer science.* Springer Science & Business Media, 2012.

[3] Casini, G., Harrison, M., Meyer, T., and Swan, R. Arbitrary ranking of defeasible subsumption.

[4] Casini, G., Meyer, T., and Varzinczak, I. Defeasible entailment: From rational closure to lexicographic closure and beyond. In *Proceeding of the 17th International Workshop on Non-Monotonic Reasoning (NMR 2018)* (2018), pp. 109–118.

[5] Casini, G., Meyer, T., and Varzinczak, I. Taking defeasible entailment beyond rational closure. In *Logics in Artificial Intelligence: 16th European Conference, JELIA 2019, Rende, Italy, May 7–11, 2019, Proceedings 16* (2019), Springer, pp. 182–197.

[6] Contributors, P. Pyeda: Python electronic design automation, 2023. Accessed: 27 August 2023.

[7] Giordano, L., Gliozzi, V., Olivetti, N., and Pozzato, G. L. Semantic characterization of rational closure: From propositional logic to description logics. *Artificial Intelligence 226* (2015), 1–33.

[8] Kaliski, A. An overview of klm-style defeasible entailment. *NA* (2020).

[9] Kraus, S., Lehmann, D., and Magidor, M. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence 44*, 1-2 (1990), 167–207.

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

[11] Levesque, H. J. Knowledge representation and reasoning. *Annual review of computer science 1*, 1 (1986), 255–287.

[12] Pollock, J. L. Defeasible reasoning. *Cognitive science 11*, 4 (1987), 481–518.

## A FIGURES FOR EXPERIMENTAL ANALYSIS

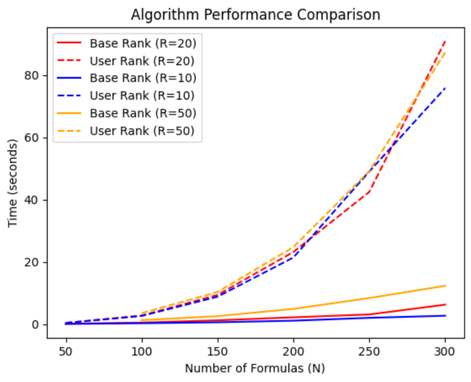

**Figure 2: Performance comparison of `BaseRank` and `User Rank` across various knowledge bases.**

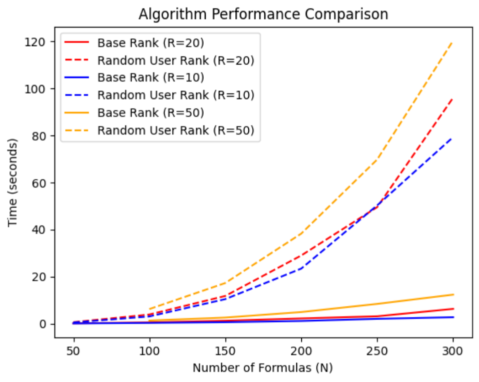

**Figure 3: Performance comparison of `BaseRank` and `Random User Rank` across various knowledge bases.**

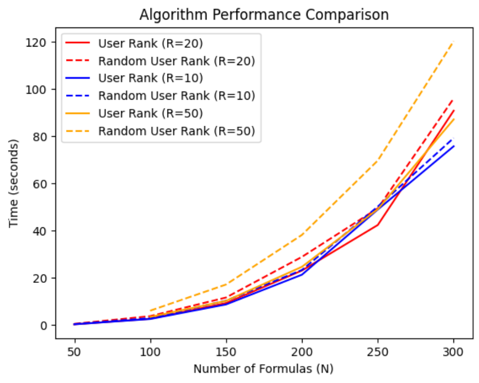

**Figure 4: Performance comparison of `User Rank` and `Random User Rank` across various knowledge bases.**