Extending Defeasible Reasoning beyond Rational Closure

Julia Cotterrell
cttjul001@myuct.ac.za
University of Cape Town
Cape Town, South Africa

ABSTRACT

Defeasible reasoning extends classical logic by enabling the retraction of conclusions in the presence of exceptions, thereby supporting more realistic, non-monotonic inference. This paper focuses on a central KLM-framework entailment method, Rational Closure, outlining its theoretical foundations. We present a web-based tool, implemented in Java and TypeScript using the TweetyProject framework, that computes and visualises these reasoning processes, including base-rank construction and entailment evaluation. The system integrates an interactive debugger that allows users to inspect exceptions, trace inference steps, and identify conflicts within knowledge bases. Designed for both research and education, the tool facilitates a deeper understanding of defeasible reasoning and supports practical debugging in complex reasoning tasks.

KEYWORDS

artificial intelligence, knowledge representation and reasoning, non-monotonic reasoning, defeasible reasoning, rational closure, lexicographic closure, knowledge base generator, user interface, debugger

1 INTRODUCTION

- Context of the project: AI, KRR, and classical vs defeasible reasoning
- Overview of the AI field and Knowledge Representation and Reasoning [16], contrasting classical logic with defeasible reasoning [14, 18].
- Problem statement: classical logic fails with exceptions Explanation of why classical logic struggles to handle exceptions [14, 15].
- Motivation for exploring Rational Closure [6, 7]
- Project aim: develop a tool and investigate extensions beyond Rational Closure that makes defeasible reasoning tools more accessible
  Project objectives and intended outcomes, drawing on existing reasoning tools [8, 12].
- Outline of contributions and structure of the paper

2 BACKGROUND

2.1 Propositional Logic

Propositional logic formalises knowledge representation and reasoning [13]. In order to do this, we need to first formalise the idea of truth as a concept by assigning one of two arbitrary values to a statement. This research paper will adhere to the conventional use of *true*, of *T*, and *false*, of *F*. These statements can be combined using connectives to create more complicated formulas that assess the truth of statements without the subtleties of natural language [13] . However, its monotonic nature restricts its ability to handle exceptions [16]. This limitation highlights the need for alternative reasoning mechanisms.[2], and how they underpin non-monotonic reasoning [10].

2.1.1 Syntax. *Atoms* are a set of unbounded symbols, denoted by lowercase Latin alphabet letters. *Atomics Propositions, $\mathcal{P}$*, consist of elements $p, q, r, s, \ldots$ constructed as $\mathcal{P} = \{p, q, r, s, \ldots\}$ [2].

Boolean Operators are connectives that combine atoms to create formulas [2]. These are shown by:

$\wedge$	and
$\vee$	or
$\rightarrow$	if then
$\leftrightarrow$	if, and only if

Figure 1: Set of binary boolean operators

2.1.2 Semantics. Semantics provides meaning to logic. It is defined by interpretations that assign a value, T or F, to each atom in a formula that allows semantic rules to determine the truth value of a formula [2]. In Propositional Logic, any part of the language that is used to model knowledge is considered object-level, while anything that operates over the object-level is considered meta-level. Valuations, atoms, and boolean operators would all be considered at the object-level while satisfiability and entailment would be considered meta-level [13].

2.2 Defeasible Reasoning

2.2.1 The KLM Framework.

2.2.2 Propositional logic and entailment review. A brief review of propositional logic [2] and entailment definitions [16].

2.2.3 Defeasible implications and preferential models. Discussion of defeasible implication syntax [14] and preferential model interpretations [9, 15].

2.3 Base Rank

2.3.1 Base Rank algorithm (Algorithm 1). Description and explanation of the Base Rank algorithm [7, 11].

2.3.2 Semantics and materialisation. Explanation of the semantic underpinnings, and materialisation process [7].

2.4 Rational Closure

2.4.1 Rational Closure algorithm (Algorithm 2). Detailed outline of the Rational Closure algorithm and its application [5, 6].

3 RELATED WORK

3.1 Defeasible Reasoning Debugger

Providing an accessible DR tool for debugging and understanding is important.

- The TweetyProject library provides a foundational backend for implementing reasoning systems, including support for propositional logic and defeasible entailments. This makes it a suitable foundation for building debuggers and interactive tools [20].
- Grosof et al. developed SILK UI, which visually traces reasoning paths, enhancing interpretability [12]. Schekotihin et al. [19] introduced OntoDebug, an interactive plug-in for Protégé that identifies missing information through user queries, while Coetzer et al. [8] improved upon this by addressing issues caused by multi-level exceptions in classical ontologies.
- We built on previous honours and masters project frameworks [17, 21]

4 SYSTEM DESIGN AND IMPLEMENTATION

4.1 Aim

4.1.1 Design and implement a web-based reasoning tool. An interactive, multi-modal defeasible UI will further improve transparency by visualising reasoning steps graphically, textually, and logically. If time permits, support for converting natural language into formal logic will be added, lowering the barrier for non-experts [4, 12].

4.1.2 Support debugging and educational use. Intended applications in debugging how a Rational Closure conclusion was reached [8, 19].

4.2 Requirements

4.2.1 Functional Requirements. List of functional requirements for the UI.

4.2.2 Non-Functional Requirements. List of non-functional requirements for the UI.

4.3 Architecture

4.3.1 Software stack: Java, Spring-boot, React, TypeScript, TweetyProject. Description of technology choices, including TweetyProject for reasoning [20]. Insert a *Layered Architecture Diagram* here

4.4 System Implementation

4.4.1 Back-end Implementation.

- Use of Spring Boot for RESTful APIs
- Justification for layered approach

4.4.2 Front-end Implementation.

- Use of Typescript
- Use of react
- Justification of layout

5 RESULTS

5.0.1 Output examples: screenshots and UI tabs. Presentation of system outputs ([8, 12]).

5.0.2 Algorithm outputs: base rank, rational closure. Display of outputs generated by each reasoning algorithm [6].

6 TESTING AND EVALUATION

6.1 Correctness

Use of JUnit testing

6.2 Performance

- Execution time
- Memory Usage
- Scalability

6.3 Usability

Assessed by getting feedback from expert including but not limited to our supervisor and other colleagues.

Integration Testing

7 DISCUSSION

7.0.1 Insights from results. Observations on the Base Rank and Rational Closed Method Algorithms [6].

7.0.2 Modifications made for performance. Changes for efficiency [5]. Integration with partners in project.

7.0.3 How the tool supports learning and debugging. Educational and debugging value [8, 12].

7.0.4 Comparison to related systems. Comparison to prior reasoning tools with justifications for changes [4, 12].

7.1 Limitations

Limitations in the current approach.

8 CONCLUSIONS

- Summary of findings and contributions
- Tool's impact on visualising defeasible reasoning

8.1 Future work

Inclusion of other Defeasible Reasoning frameworks:

- Relevant closure
- Lexicographic closure

Possible extensions [1, 3].

REFERENCES

[1] Clayton Kevin Baker, Claire Denny, Paul Freund, and Thomas Meyer. Cognitive defeasible reasoning: the extent to which forms of defeasible reasoning correspond with human reasoning. In *Southern African Conference for Artificial Intelligence Research*, pages 199–219. Springer, 2020.

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

[3] Giovanni Casini, Michael Harrison, Thomas Meyer, and Reid Swan. Arbitrary ranking of defeasible subsumption. 2019.

[4] Giovanni Casini, Thomas Meyer, Kody Moodley, Uli Sattler, and Ivan Varzinczak. Introducing defeasibility into owl ontologies. In *The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II 14*, pages 409–426. Springer, 2015.

[5] Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. Defeasible entailment: From rational closure to lexicographic closure and beyond. In *Proceeding of the 17th International Workshop on Non-Monotonic Reasoning (NMR 2018)*, pages 109–118, 2018.

[6] Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. Taking defeasible entailment beyond rational closure. In *Logics in Artificial Intelligence: 16th European Conference, JELIA 2019, Rende, Italy, May 7–11, 2019, Proceedings 16*, pages 182–197. Springer, 2019.

[7] Giovanni Casini and Umberto Straccia. Rational closure for defeasible description logics. In *Logics in Artificial Intelligence: 12th European Conference, JELIA 2010, Helsinki, Finland, September 13-15, 2010. Proceedings 12*, pages 77–90. Springer, 2010.

[8] Simone Coetzer and Katarina Britz. Debugging classical ontologies using defeasible reasoning tools. In *Formal Ontology in Information Systems (FOIS 2022)*, volume 344, pages 97–106. IOS Press, 2022.

[9] Michael Freund. Preferential reasoning in the perspective of poole default logic. *Artificial Intelligence*, 98(1-2):209–235, 1998.

[10] Dov M Gabbay. *Theoretical foundations for non-monotonic reasoning in expert systems.* Springer, 1985.

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

[12] Benjamin Grosof, Mark Burstein, Mike Dean, Carl Andersen, Brett Benyo, William Ferguson, Daniela Inclezan, and Richard Shapiro. A silk graphical ui for defeasible reasoning, with a biology causal process example. In *9th International Semantic Web Conference ISWC 2010*, page 113. Citeseer, 2010.

[13] Adam Kaliski. An overview of klm-style defeasible entailment. *NA*, 2020.

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

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

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

[17] Thabo Vincent Moloi. Extending defeasible reasoning beyond rational closure. CS Honours Project, Department of Computer Science, 2024. Supervisor: Thomas Meyer.

[18] John L Pollock. Defeasible reasoning. *Cognitive science*, 11(4):481–518, 1987.

[19] Konstantin Schekotihin, Patrick Rodler, and Wolfgang Schmid. Ontodebug: Interactive ontology debugging plug-in for protégé. In *Foundations of Information and Knowledge Systems: 10th International Symposium, FoIKS 2018, Budapest, Hungary, May 14–18, 2018, Proceedings 10*, pages 340–359. Springer, 2018.

[20] Matthias Thimm. The tweety library collection for logical aspects of artificial intelligence and knowledge representation. *KI-Künstliche Intelligenz*, 31(1):93–97, 2017.

[21] Steve Wang. Defeasible justification for the klm framework. Master's thesis, Faculty of Science, University of Cape Town, 2022.

A ALGORITHMS

- Algorithm 1: BaseRank
- Algorithm 2: RationalClosure
- Modified versions (Appendix A in paper)

B SUPPLEMENTARY INFORMATION

B.1 Screenshots

- Knowledge base form input
- Rendered formula view
- Reasoning outputs: Base Rank and Rational Closure