Extending Defeasible Reasoning Beyond Rational Closure

Background

Propositional Logic

Propositional logic uses Boolean connectives to combine propositions, resulting in more complex statements.

A knowledge base is a set of such statements. The following knowledge base contains the information "**Penguins are birds**", "**Birds fly**", and "**Penguins don't fly**":

$$K = \{p \rightarrow b, b \rightarrow f, p \rightarrow \neg f\}$$

Classical Reasoning

Using logical rules, we can infer new statements from other statements. For example, from the above knowledge base, we can infer the statement that **penguins are not possible:**

$$K \models \neg p$$

Defeasible Reasoning

Classical propositional logic cannot express the concept of typicality. As a result, penguins, being atypical birds, are deemed to be impossible.

Defeasible reasoning solves this by allowing statements and inferences that usually hold but can be retracted given new information.

For example, the modified knowledge base contains "**Penguins are birds**", "**Birds typically fly**", and "**Penguins typically don't fly**":

$$K = \{p \rightarrow b, b \mathrel{|\!\sim} f, p \mathrel{|\!\sim} \neg f\}$$

$$K \not\models \neg p$$

Rational Closure

Defeasible reasoning is formalized by rational closure, which captures a reasoning pattern known as **presumption of typicality.**

Rational closure works by ranking information according to its level of typicality and assumes information to be as typical as possible.

There are two ways to rank information:

(1) Ranking possible "worlds" that align with the statements.

(2) Ranking individual statements themselves.

Assuming information to be as typical as possible gives it the characteristic of being the most conservative pattern of defeasible reasoning.

Our aim is to investigate the development of extensions of rational closure, from both a theoretical and practical standpoint.

Defeasible KB Generator

Aim

We create a non-deterministic algorithm called KBGenerator and an optimised variant called KBGeneratorThreaded, capable of generating defeasible knowledge bases of varying complexities and structures, organized in a manner aligned with Base Rank's ranking of a knowledge base.

Features of KBGenerator/Threaded

- Number defeasible implications (DIs)
- Number of ranks
- Distribution of DIs
- Simple/Mixed DI generation
- Adj. Antecedent and Consequent complexity
- Adj. DI connective types

0	$v \mathrel{|\!\sim} s,\ l \wedge f \mathrel{|\!\sim} v$
1	$a \mathrel{|\!\sim} \neg s,\ a \mathrel{|\!\sim} v,\ w \mathrel{|\!\sim} a \vee d$
2	$n \mathrel{|\!\sim} s,\ n \mathrel{|\!\sim} a,\ b \mathrel{|\!\sim} n \leftrightarrow j,\ o \rightarrow t \mathrel{|\!\sim} n$

(1) A Defeasible KB with a growth distribution, mixed statements and varied connective types.

Future Work

Further improvements can be made to the pseudo-random recycling of atoms in DIs, along with more sophisticated thread management for the optimised generator. These modifications have the potential to yield significant performance improvements in generating complex defeasible implications

User-Defined Rankings

Invalid Rankings

Some rankings of statements are invalid. For example, a ranking where $b \mathrel{|\!\sim} f$ is higher than $p \mathrel{|\!\sim} \neg f$ does not have a **corresponding** ranking of worlds.

K-faithful Rankings

To ensure that the ranking is valid, we want to ensure that it is a **K-faithful** ranking. That is to say, for a given knowledge base K, our ranking will entail all the defeasible statements in K.

UserRank

We develop an algorithm called UserRank which allows a user to input any ranking of statements they wish, and ensures that the ranking is K-faithful, making adjustments to the ranking as necessary

Properties of UserRank

We show that our algorithm, UserRank, satisfies various important properties:

1. It satisfies the maximum possible number of the user's original rankings.
2. It withholds **rank minimality**. That is to say, it conforms with the correspondence function between rankings of statements and rankings of worlds.
3. It is K-faithful.

Future Work

The UserRank algorithm and its practical implementation can be used for research in cognitive reasoning. Furthermore, there may be performance improvements in the design of UserRank; specifically, optimizing a double-nested for loop which is required to do various validity checks on the user's rankings.

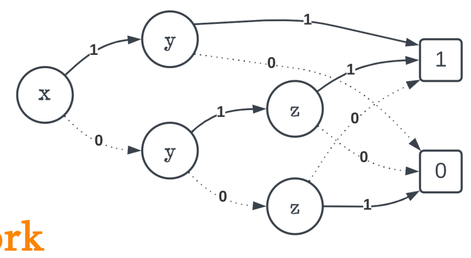
Reduced Model Ranking

Problem

The number of potential models increases exponentially with each new proposition. Leading to infeasible storage requirements.

Solution

Use ROBDDs to remove redundant information.



Future Work

Empirically assess the computational benefits of using the reduced model ranking approach compared to the statement ranking approach.

Naïve Bayes Entailment

Probabilistic Rankings

Treat the antecedent as evidence and refine the rankings of worlds in rational closure according to their probability using naïve bayes calculations.

Example

$\mathcal{K} = \{\texttt{emperor} \rightarrow \texttt{penguin}, \texttt{emperor} \mathrel{|\!\sim} \texttt{wings},$
$\texttt{king} \rightarrow \texttt{penguin}, \texttt{king} \mathrel{|\!\sim} \texttt{wings},$
$\texttt{chinstrap} \rightarrow \texttt{penguin}, \texttt{chinstrap} \mathrel{|\!\sim} \neg\texttt{wings}\}$

$$\mathcal{K} \mathrel{|\!\approx}_{NB} \texttt{penguin} \mathrel{|\!\sim} \texttt{wings}$$

If two out of three penguins have wings, then penguins typically have wings.

Future Work

Empirically assess the complexity of the naïve bayes algorithm, and further explore its properties.

Authors

lngale007@myuct.ac.za Alec Lang

plldav013@myuct.ac.za David Pullinger

sltluk001@myuct.ac.za Luke Slater

Supervisor

Professor Tommie Meyer

tmeyer@cair.org.za

IT | SCHOOL OF IT

www.sit.uct.ac.za

dept@cs.uct.ac.za

021 650 2663