User-Informed Preferential Reasoning for Ontologies

Michael Harrison
Department of Computer Science
University of Cape Town
hrrmic014@myuct.ac.za

ABSTRACT

Human knowledge often is many domains often contains exceptions. The usefulness of Knowledge Representation and Reasoning in real world applications has been limited by the lack of a well-defined system to handle nonmonotonicity. This paper presents a method for implementing a preferential reasoning system that utilises a user defined ranking of defeasible statements. This tool provides proof of an effective form of defeasible reasoning that gives a user more control and expressivity.

CCS CONCEPTS

• **Computing methodologies → Knowledge representation and reasoning**; *Description logics*; *Semantic networks*; *Nonmonotonic, default reasoning and belief revision*; *Reasoning about belief and knowledge*; Ontology engineering;

KEYWORDS

Preferential Reasoning, Defeasible Reasoning, Non-monotonicity, Rational Closure, Description Logics, Knowledge Representation, OWL, Web Ontology Language, OWL API

1 INTRODUCTION

A knowledge representation is a contextual and domain-specific computable model [33]. Davis, Shrobe, and Szolovits [18] argue that a knowledge representation can be best understood in terms of five distinct roles that it plays: firstly, knowledge representation is a surrogate for actual things that allows reasoning about it instead of actual action; secondly, knowledge representation is a set of ontological commitments; thirdly, it is a fragmentary theory of intelligent reasoning; fourthly, it is a medium for efficient computation; fifthly, it is a medium of human expression.

Reasoning is the production of new propositions through the manipulation of symbols representing a set of believed propositions [8]. Another way of saying this is that new knowledge is inferred from asserted knowledge. Classical reasoning is *monotonic*, meaning that adding new knowledge does not remove conclusions that have already been drawn. This prevents the use of classical reasoning in a knowledge base where new information contradicts current information [31].

An ontology is a system that represents the entities or concepts of a subject domain and their relationships. In this sense, it is a form of knowledge representation. Ontologies were traditionally only present in the the field of Philosophy but, in recent years, the value of ontologies for computer information systems purposes has been discovered [20]. An ontology can be replicated as a software artifact. The ontology is simply the structure and needs to be represented by some sort of language. The chosen language or knowledge representation *formalism* should, ideally, capture all of the human knowledge in the chosen subject domain [2]. Using logic-based formalisms makes it possible to extract *implicit* information from the ontology. This is generally done through *classical reasoning* processes. A statement that is found to hold true in every interpretation of a knowledge base can be said to be *entailed* by the knowledge base.

Reasoning programs exist that can infer consequences from the knowledge contained in an ontology [1].

The standard ontology reasoning programs perform classical reasoning. This places a restriction on the ontology. The system is required to be monotonic for the reasoners to work correctly. This is inappropriate if a domain that contains exceptions needs to be modeled [15]. This brings about the need for *defeasible* reasoning. A defeasible fact is a non-strict fact [14]. Defeasible information can be represented through *conditional assertions* [25]. A defeasible knowledge base (KB) is composed of a classical TBox T and a DBox D that contains a set of defeasible conditional inclusions [30]. Defeasibility is introduced through a defeasible subsumption operator ($\sqsubsetsim$) [13] that states C is *usually* subsumed by D. Defeasible subsumption can also be read as: *typically*, an instance of C is also an instance of D [14]. There have been many attempts at performing defeasible reasoning. These attempts take different approaches. Some of the more prominent approaches include Circumscription [6], Default Logic [32], Negation as failure [23], Probabilistic logic [26], and Preferential reasoning [13].

Preferential reasoning is particularly appealing because it can be reduced to classical reasoning [15]. This is advantageous because preferential reasoning algorithms can make use of the performance of well-developed classical reasoners. Lehmann et al. [25] have motivated that a Tarskian style of entailment is not sufficient as a method of preferential reasoning due to it defining a monotonic consequence relation. A monotonic consequence relation cannot cope with the retraction of previously known knowledge. A preference relation is also too restrictive if it does not satisfy the seventh KLM postulate of Rational Monotonicity [24]. Rational Monotonicity expresses that previously drawn plausible conclusions should only be withdrawn is information that is the negation of what was expected is added to the knowledge base.

A method of performing preferential reasoning that has been proposed that is nonmonotonic and satisfies the Rational Monotonicity rule is Rational Closure [25]. The algorithm to perform rational closure is explained in detail in Section 3. Rational closure utilises a ranking of defeasible statements that is generated by an exceptionality algorithm. This paper is part of a joint project with Reid Swan that takes the Rational Closure approach to preferential reasoning and attempts to generalise it by using arbitrary user-defined rankings in place of the ranking defined by the exceptionality algorithm. Reid worked on the theoritical underpinnings [34] and I carried out the implementation. My implementation includes a standalone ontology editor with preferential reasoning capabilities. This paper gives the necessary background and then focuses on the implementation of this tool.

1.1 Significance

An algorithm to perform preferential reasoning will allow for nonmonotonic knowledge bases to be modeled and queried. This is hugely significant, due to most areas of knowledge containing exceptions of some sort. The applications of knowledge reperesentation and reasoning have traditionally been limited due to the lack of an appropriate way to handle nonmonotonicity. Allowing defeasibilty changes this and opens up a wealth of possibilities for the application of knowledge representation and reasoning. Using a user-provided ranking further adds value. A domain specialist can communicate more knowledge with greater detail by expressing exceptionality. In addition to being practically utilised, this tool serves great value as a proof of concept for a rational preference relation algorithm.

2 BACKGROUND

An ontology, in a philosophical sense, is a particular system of categories that account for a certain vision of the world. To make use of an ontology in the realm of Artificial Intelligence, we need to turn it into a engineering artifact. To do this, the ontology requires a specific vocabulary and a set of explicit assumptions to convey the intended meaning of the vocabulary words [20]. The vocabulary is the formalism for representing knowledge. Description Logics provide such a vocabulary.

Description Logics (DLs) are decideable fragments of first-order logic [3]. DLs are a family of logics that provide a good balance between expressive power and computational complexity of reasoning tasks [7]. Greater expressive power allows more knowledge of different types to be represented. Another appealing feature is that these languages have a precise syntax and semantics that allow an ontology to be represented purely as a set of logical statements about the domain [31]. Description Logics descend from *structured inheritance networks* [9]. This means that they start with atomic unary predicate *concepts* and binary predicate *roles*. Complex concepts and roles can then be built using a set of epistemologically adequate constructors. Reasoning is performed by drawing inferences from the explicit knowledge in the DL knowledge base. There are two typical forms of inference problems [2]. The first is the *subsumption* problem. This concerns checking if concept A is subsumed by concept B. This would be that A is a subconcept of B. The second is the *instance* problem. a in an instance of concept C only if a in an instance of C for every interpretation of the TBox. The TBox contains all the asserted terminological axioms. *Intensional knowledge* is contained in the form of a terminology in the TBox. *Extentional/assertational knowledge* is contained in the ABox [4].

To check for subsumption in a DL knowledge base, tableau algorithms are used. Tableau algorithms check for the unsatisfiability of concept descriptions derived from the subsumption statement using negation [3]. $C \sqsubseteq D$ if and only if $C \sqcap \neg D$ is unsatisfiable.

Description logics are central to many modern Artificial Intelligence applications because they provide the logical foundation of formal ontologies [11]. Description Logics are used for representing declarative knowledge about a chosen domain of interest, as well as revealing and representing implicit knowledge about the domain by enabling automated reasoning over the explicit declared knowledge [29]. Certain logics are appropriate languages to express ontologies with. This is due to their capacity to allow for the symbolic representation of all relevant knowledge in a domain [2]. There are two requirements for symbolic representation. The first is having declarative semantics to represent the knowledge. The second is some "intelligent" retrieval mechanism to obtain implicit information. DLs meets both of these criteria. Description Logics are particularly well-suited for representing and reasoning about terminological knowledge, and constitute the formal foundations of semantic web ontologies [10]. OWL is the standard formalism for software-based ontologies.

OWL is the Web Ontology Language. It is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called an ontology [27]. OWL DL provides maximum expressiveness for OWL and contains all OWL constructs. It retains computational completeness as all conclusions are guaranteed to be computable. It also retains decidability as all computations will finish in finite time. OWL DL is so named due to its correspondence with description logics. [27].

OWL 2 is an extension and revision of the OWL Web Ontology Language developed by the W3C Web Ontology Working Group. One of the new features in OWL 2 which makes it easy to express defeasible subsumption is the introduction of OWL annotations. One can therefore attach an annotation to a standard subsumption axiom in the ontology which allows a reasoning system to interpret this subsumption as a defeasible one [28]. OWL 2 DL is an extension and revision of OWL DL. OWL 2 DL has the full expressivity of the SROIQ description logic language. The primary exchange syntax for OWL 2 is the RDF Syntax (RDF/XML). OWL 2 also has a more syntax that is more readable for humans, called the Manchester Syntax. The Manchester Syntax is used in several ontology editing tools [35].

RDF is a datamodel for objects and the relations between them. It provides a simple semantics for this datamodel. These datamodels can be represented in an XML syntax [27]. RDF expresses meaning by encoding it in sets of triples, with each triple being rather like the subject, verb and object of an elementary sentence [5].

XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents [27]. XML allows users to add arbitrary structure to their documents but says nothing about what the structures mean [5].

The Semantic Web is a vision for the future of the Web in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web. The Semantic Web will build on XML's ability to define customized tagging schemes and RDF's flexible approach to representing data. The first level above RDF required for the Semantic

Web is an ontology language what can formally describe the meaning of terminology used in Web documents [27].The Semantic Web will bring structure to the meaningful content of Web pages [5].

Many ontology reasoners have been built that are capable of inferring logical consequences from OWL ontologies. OWL entailment checking can be reduced to Description Logic satisfiability [22]. This allows OWL reasoners to use tableaux algorithms derived from DL reasoning. Queries can be answered by OWL reasoners using forward chaining or backward chaining [1]. Forward chaining starts from the asserted facts and derives valid inferences. Backward chaining starts with the query and tries to find possible solutions for it.

The *OWL API* is a high level Application Programming Interface (API) that supports the creation and manipulation of OWL Ontologies. It is implemented in Java and is available as open source under an LGPL licence. [21]. OWL is very useful in that it supplies the semantics that define precisely what entailment means with respect to OWL ontologies. The OWLReasoner interface provides access to to all the functionality needed to reason with OWL ontologies [21].

3 RATIONAL CLOSURE

In order to reason about a defeasible ontology, inferences need to be derived from its conditional knowledge base. New information can be derived through a closure operation that returns a preferential consequence relation [25]. This method of preferential entailment, however, is too weak [17]. Preferential entailment satisfies six of the seven properties required for a rational preference relation, as defined by Kraus, Lehmann, and Magidor [24]. These properties are know as the KLM postulates. Preferential entailment does not satisfy Rational Monotonicity. Preferential entailment is, therefore, not rational. A rational relation containing p preferentially entailed by q, must contain p and r preferentially entailed by q unless it contains p and $\neg r$.

Rational closure is an approach to preferential reasoning. Rational closure satisfies all of the KLM postulates, including Rational Monotonicity. Rational closure, therefore, computes a rational consequence relation. The notion of a relation being preferable to another one can be captured by defining a partial ordering between rational relations. The ranking of a defeasible knowledge base can be determined using *exceptionality.* The rational closure of a knowledge base is then defined as the rational extension of a knowledge base that is preferable in the ordering to all other rational extensions [25]. Let D be a defeasible TBox. The rational closure of D is the (unique) rational subsumption relation which includes D and is preferable to all other rational subsumption relations including D [13].

Not every knowledge base has a rational closure, but any admissible knowledge base has a rational closure. A knowledge base K is said to be admissible if and only if all formulas that have no rank for K are inconsistent for K. Any finite knowledge base has a rational closure [25].

Rational closure has shown favourable performance over other closures [30]. Performing rational closure can be reduced to classical entailment checks, allowing the use of a developed classical reasoner [31]. A decidable algorithm can then be constructed to check whether a given defeasible subsumption statement is in the rational closure of a defeasible TBox D [13].

The first step in performing Rational Closure, as defined by Moodley [29], is computing the *ranking*. The ranking algorithm takes in all the defeasible statements in the defeasible knowledge box (DBox). The ranking algorithm returns a set of DBoxes [31]. Each subsequent DBox in the set contains more exceptional statements. All the statements are initially in the most general set (level) of the ranking. The *exceptional* algorithm is performed on each statement. A defeasible axiom $C \sqsubsetsim D \in D$ is considered exceptional if its antecedent C is exceptional. If the statement is exceptional amongst the other statements in that set, then it is moved to the next set. The higher the set or level, the more exceptional a statement is. Strict statements in the form of $C \sqsubseteq D$ will always be in the most exceptional set. In order to compute the rational closure for a query $C \sqsubsetsim D$, the portion of the computed ranking that is compatible with C is obtained. This is done using the *exceptional* algorithm for C. Once a set is found where C is no longer atypical, that set and all the more exceptional sets are retained. This retained subset of the ranking is used for the final step. The final step takes all models for the strict Tbox T which satisfy the defeasible subsumptions in the retained portion of the ranking and the antecedent of the query (C). If the consequent of the query (D) is satisfied by each of these models, then it can be inferred that $C \sqsubsetsim D$ is in the rational closure of the knowledge base.

4 PREFERENTIAL REASONING

Preferential reasoning was first proposed by Kraus, Lehmann, and Magidor [24]. A preference relation can only be considered *rational* if it satisfies a set of all of the KLM inference rules. A rational preference relation is a preference relation that satisfies Rational Monotonicity. For the theoretical underpinnings of this project, Swan was able to prove that preferential reasoning performed with a defeasible knowledge base that has an arbitrary ranking of defeasible statements satisfies Rational Monotonicity and, therefore, defines a rational preference relation [34]. This method of preferential reasoning can be seen as a generalisation of Rational Closure. This arbitrary ranking could replace the ranking computed by the *exceptionality* algorithm for Rational Closure. The remainder of the Rational Closure algorithm could then be computed as normal. This allows a user to have greater control over the meaning of the defeasible knowledge base. A user could theoretically specify the ranking of defeasible statements based on the user's own notion of exceptionality. Once a ranking has been defined by the user, the preferential reasoning algorithm would replicate the Rational Closure algorithm as carried out by Moodley [29]. A user would be able to query if the knowledge base entails $C \sqsubsetsim D$. To answer this query, the preferential reasoning algorithm would first have to find the portion of the user-defined ranking where C is not exceptional. To do this, the algorithm first needs to obtain the *internalisation* of each subset of the ranking. The internalisation of a set of axioms is the conjuction of the *materialisation* of every axiom within the set. The materialisation of $C \sqsubseteq D$ is $\neg C \sqcup D$ [29]. The notions of materialisation and internalisation are well known in the context of propositional logic. To obtain the portion of the ranking that is compatible with C (the subset of the ranking where C is no longer exceptional), the algorithm iteratively checks if the internalisation of a subset of the ranking $\sqsubseteq \neg C$ is entailed by the strict portion of the knowledge base (T). The algorithm starts with the most general subset of the ranking, which includes all defeasible axioms. If T entails the internalisation of all of these axioms $\sqsubseteq \neg C$, then C is exceptional with regards to the subset. The most general axioms in the set (the ones with in the lowest level) are then discarded and the process is repeated with the more exceptional subset of axioms. When the algorithm reaches a subset of the ranking that has a materialisation $\sqsubseteq \neg C$ not entailed by T, then that subset of the portion of the ranking where C is not exceptional. The final portion of the algorithm checks if $C \sqcap$ internalisation of this portion of the ranking $\sqsubseteq D$ is entailed by T. If *true* is returned, $C \mathrel{\underset{\sim}{\sqsubset}} D$ is in the rational closure of the knowledge base. This means that yes can be answered to the query of $C \mathrel{\underset{\sim}{\sqsubset}} D$ being inferred by the knowledge base. If the user enters a strict query ($C \sqsubseteq D$), entailment is checked in the normal classical sense.

5 IMPLEMENTATION

In order to create a tool that is tailored for preferential reasoning with user rankings, a standalone tool was built rather than a plugin for an existing tool. The tool was built in Java using the OWL API to interact with the OWL ontology and HermiT reasoner to perform classical entailment checks. It was built into a runnable executable file using Gradle. The tool implements Preferential Reasoning as described in Section 4. The major implementation steps are explained:

5.1 Ontology creation

The knowledge base is stored as an ontology. This ontology is represented using the OWL web ontology language, which is equivalent to Description Logics. A user can create new classes, relations, as well as enter subsumption axioms. This are entered using the Manchester Syntax [35], which is very similar to natural language and more readable to humans than other syntaxes. The class expressions and axioms are added to the ontology and represented in OWL and XML. The user can open an existing OWL ontology file or create a new one. Due to the nature of the OWL API, the user must first enter concept names and relations before they can be used in an axiom. The user is then able to enter axioms to the ontology as seen in Figure 1. SubClassOf is the Manchester Syntax equivalent of $\sqsubseteq$. Complex class expressions including operators, negations, and relations may be entered on either side of the subsumption statemnt (SubClassOf). The defined axiom will then be added to the ontology.

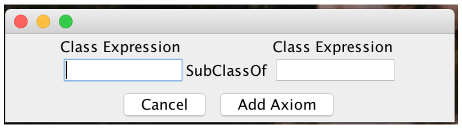
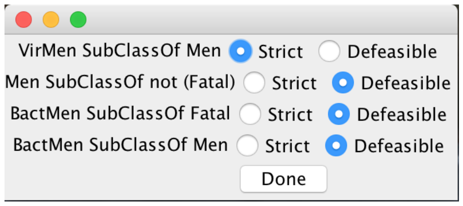


Figure 1: Adding an axiom

5.2 Denoting defeasibility

Defeasibility is represented in OWL using an annotation, such as in DIP [29]. A boolean value is used to denote whether the axiom is defeasible or not. A user will be able to toggle the defeasibilty of all axioms in the ontology, as well as declare an axiom as defeasible when entering it. This replicates a defeasible subsumption statement. The options to change an axiom between strict and defeasible is shown in Figure 2. This is performed in the OWL ontology by assigning the Defeasible annotation that I created to true or false.

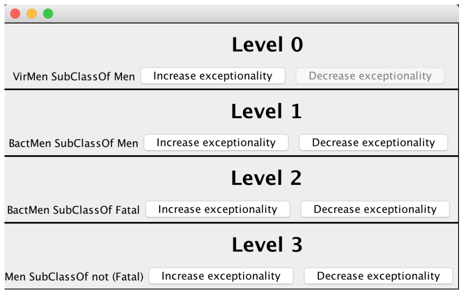


VirMen SubClassOf Men [✓ Strict] Defeasible
Men SubClassOf not (Fatal) Strict [✓ Defeasible]
BactMen SubClassOf Fatal Strict [✓ Defeasible]
BactMen SubClassOf Men Strict [✓ Defeasible]

Figure 2: Adding defeasible axioms to the ontology

5.3 Denoting and obtaining rankings

All defeasible axioms are given a default rank value of 0. 0 is the most general rank for defeasible axioms. The higher the rank number, the more exceptional the axiom. Ranks are represented as specified integer value annotations in the OWL ontology language. Users are able to change the ranking of all defeasible axioms via an interface in the tool. By clicking "Increase exceptionality" the rank annotation value of the axiom is increased by 1 and the axiom is moved to the next, more exceptional level in the ranking. By clicking "Decrease exceptionality" the rank annotation value of the axiom is decreased by 1 and the axiom is moved to the previous, more general level in the ranking.



Level 0
VirMen SubClassOf Men Increase exceptionality Decrease exceptionality

Level 1
BactMen SubClassOf Men Increase exceptionality Decrease exceptionality

Level 2
BactMen SubClassOf Fatal Increase exceptionality Decrease exceptionality

Level 3
Men SubClassOf not (Fatal) Increase exceptionality Decrease exceptionality

Figure 3: Changing the exceptionality ranking of axioms

5.4 Getting materialisations

Materialisations are classical representations of defeasible axioms. These materialisations are obtained by manipulating all the defeasible axioms using the OWL API. The materialisation of $C \mathrel{\underset{\sim}{\sqsubset}} D$ is $\neg C \sqcup D$ [10]. Materialisations are obtained as part of internalisations. Using the OWL API, the *complement* of the *superclass* of $C \mathrel{\underset{\sim}{\sqsubset}} D$ returns $\neg C$. The *subclass* of $C \mathrel{\underset{\sim}{\sqsubset}} D$ returns D. The *union* of $\neg C$ and D returns $\neg C \sqcup D$.

5.5 Getting internalisations

An *internalisation* is the intersection of the materialisation of every axiom in a ranking level and of every axiom in all of the more exceptional levels. For example: if there are 4 levels of exceptionality in a defeasible knowledge base, the materialisation of rank 2 is the intersection of the materialisation of every axiom in rank levels 2, 3, and 4 (the level at hand and all of the more exceptional levels). In order to obtain the internalisations, all of the defeasible axioms are added to a Java TreeMap according to their ranking. The keys in the TreeMap are the ranking integers and value assigned to each key is a Java Set of every axiom with that rank. By added the axioms to a TreeMap according to their rank, they are automatically put in the correct position. This saves the computational power of later having to sort the ranking. Each axioms materialisation is computed as it is being inserted into the TreeMap and the materialisation is what is stored. To create the internalisations from this data structure, the algorithm iterates from the most exceptional ranking. The internalisation for each ranking is computed by returning the *intersection* of every axiom in the current as well as all of the axioms in more exceptional rankings. A new TreeMap of internalisations is created with the keys being rank levels and the values being the internalisation of that rank. This is performed using Java 8's new lambda calculus functionality to make it less computationally expensive.

5.6 Performing Preferential Reasoning

Preferential reasoning is performed when a user queries if a statement is entailed by the knowledge base. If the user enters a strict query in the form $C \sqsubseteq D$, classical reasoning is performed to check if $C \sqsubseteq D$ is entailed by the strict axioms T. If the user enters a defeasible query in the form $C \mathrel{\underset{\sim}{\sqsubset}} D$, the generalisation of the computation of Rational Closure as described in Section 4 is carried out. In order to do this, the antecedent of the query needs to first be obtained. The antecedent is the subclass of the query in OWL terms. A complex OWL *SubClassOfAxiom*, which is equivalent to a subsumption statement, is created. The subclass of this axiom is the complement of the antecedent of the query ($\neg C$). The superclass of this axiom is the internalisation of the most general rank. Level 0 is the most general rank and its internalisation includes every defeasible axiom in the knowledge base. The TBox T is then obtained by getting all the statements that aren't denoted as defeasible. The algorithm checks if the complex SubClassOfAxiom is entailed by T. If the axiom is entailed by T, the algorithm creates a new axiom to check using the internalisation of rank 1, which excludes the axioms in rank 0. If this axiom is entailed by T, then the algorithm continues until there is a rank level where T does not entail the SubClassOfAxiom or the algorithm reaches the final ranking in the set. If T no longer entails the SubClassOfAxiom then C is not exceptional at the current rank level. If the algorithm reaches the end of the set, then a classical entailment check is performed to obtain the answer to the query.

When C is no longer exceptional, the remaining portion of the DBox (the current ranking and all the more exceptional rankings) is used to check for the entailment of the query. A new SubClassOfAxiom is created with the intersection of the internalisation of the remaining portion of the DBox and the antecedent (C) being the subclass and the consequent (D) being the superclass. If this new SubClassOfAxiom is entailed by T, then $C \mathrel{\underset{\sim}{\sqsubset}} D$ is in the Rational Closure of the knowledge base. HermiT [19] classical OWL reasoner is used to perform the classical entailment checks for T. HermiT is based on a hypertableau system that generally makes it quicker than other classical reasoners.

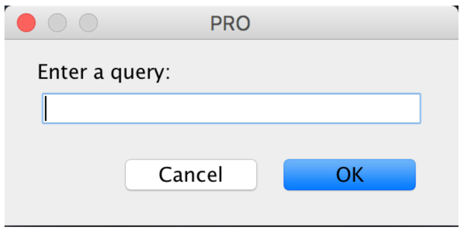


Figure 4: Entering a query

6 RELATED WORK

6.1 Alternative Defeasible Approaches

There are other approaches to preferential reasoning that are worth mentioning: *Circumscription* is a class of non-monotonic logics proposed by John McCarthy in 1980. The basic aim is to be able to represent, in KR formalisms, the presumption that something is "normal" and behaves as expected unless otherwise specified [29] *Default Reasoning* is formalism for encoding so-called defaults presented by Raymond Reiter in 1980. Defaults enable the representation of the plausible conditions under which a conclusion (first order sentence) could be entailed by a first order theory (KB) [29]. *Minimal Knowledge and Negation as Failure (MKNF)* uses epistemic querying to be able to pose queries about the external world that the KB is representing, as well as about what the KB itself knows about the external world, by viewing a KB as a set of statements about an external world, one should. This method can be very complex.[29]. *Lexicographic Closure* is an inferential extension of Rational Closure. Rational Closure defines a cautious inference mechanism. Rational Closure will only infer something if there is evidence to prove it. Lexicographic Closure defines the credulous counterpart to Rational Closure. Lexicographic Closure will infer something if there is no evidence to prove otherwise. Lexicographic Closure may be seen to be more suitable for real-world applications than Rational Closure due to the inference relation of Rational Closure sometimes being considered too cautious. There are issues with Lexicographic Closure, though. Lexicographic Closure is syntax-dependent. This means that, performing Lexicographic Closure inference on two logically equivalent KBs with differing syntaxes may yield different inferences. There are also cases where Lexicographic Closure may not perform very efficiently in practice due to some of its brute force computations [29].

Preferential reasoning has advantages over all of these other approaches. Many other approaches have never been implemented. The worst-case complexity of other methods is generally very poor. Methods such as circumscription require so many parameters to be set that it is not worth the effort. Once a user has defined preferences for preferential reasoning, the computation is automatic and relatively efficient.

6.2 Defeasible-Inference Platform

The Defeasible-Inference Platform (DIP) was created by Kody Moodley as part of his PhD dissertation[29]. DIP is a plugin for the Protégé ontology editor. DIP allows a user to get the defeasible subclasses or defeasible superclasses of an entity. His plugin also provides the functionality to change an axiom from strict to defeasible and back to strict. New annotations are created by the plugin to denote defeasibility and the rank of axioms in the OWL ontology representation syntax. The rankings of defeasible axioms are computed using an exceptionality algorithm. Experiments were run with DIP to compare the performance of checking for defeasible subclasses between rational closure, lexicographic closure, basic, minimal, and lexicographically relevant closure. Rational closure performed better than all the others, computing at only 3.5 times the speed of a classic entailment check[29].

7 CONCLUSIONS

In order to check the accuracy of rational closure computations, user rankings and defeasible ontologies were created to replicate those used as examples in Moodley's DIP tool [29]. The user supplied ranking was set to replicate the exceptionality ranking of Moodley to observe if the two tools behaved the same with the same ranking. The preferential reasoning tool successfully returned the correct answers for all standard defeasible subsumption queries. The tool did not, however, behave as expected when encountering a complex defeasible subsumption query. Union and intersection operators on the left hand side of the query resulted in inaccurate answers for rational closure checking. This is due to the nature with which OWL axioms are manipulated with the OWL API. Additional checks need to be put in place to handle complex queries and transform them appropriately when creating the SubClassOfAxiom that is used when performing the classical entailment portion of the algorithm. Only the subsumption type of inference was attempted in this project. The tool is a successful proof of concept for preferential reasoning with arbitrary rankings. Computation of classical entailment is more efficient than rational entailment, but the cost of the implementation of rational closure is not at all excessive [15]. The most computationally intensive components of the Rational closure procedure are its classical entailment checks [29]. Its computational complexity compares favorably with that of most well-established systems [25]. Therefore, the implementation of the additional inferential capabilities of Rational Closure is justified. By being able to use a generalisation of rational closure, preferential reasoning shares rational closures favourable computational efficiency. Preferential Reasoning implemented in this way satisfies the KLM requirements for a rational preference relation and allows users increased control. This work paves the way for nonmonotonic knowledge representation and reasoning applications.

8 FUTURE WORK

8.1 Role Restrictions

Defeasible role restrictions have not been the subject of much focus in defeasible reasoning research. A preferential universal restriction introduces an aspect of defeasibility to the base concept constructor of universal restriction by allowing exceptions. A preferential existential restriction, on the other hand, introduces an aspect of strictness to the base concept constructor of existential restriction. A tableau-based system can be used for reasoning with preferential role restrictions [12]

8.2 ABox Reasoning

Defeasibility in the ABox can be seen as the conclusion that an individual a presumably falls under the concept C. The knowledge base will have a classical ABox, composed of concept and role assertions, but, using the defeasible inclusion axioms in D, defeasible information about the individuals will be derived [16]. Extensions are made to the original ABox A that associates, with every individual, the defeasible information that is consistent with the rest of the knowledge base. The default information still respects the exceptionality ranking and each individual is considered as much typical as possible, preserving the general consistency. This approach to dealing with individuals remains consistent with the idea behind rational closure [10].

8.3 Handling Complex Subsumption Queries

Additional work is required to manipulate complex subsumption queries when performing preferential reasoning with OWL ontologies.

9 ACKNOWLEDGEMENTS

Thank you to my project partner, Reid Swan, for proving that preferential reasoning satisfies the KLM postulates. Thank you to Kody Moodley for your assistance and access to your code. Thank you to my supervisor, Tommie Meyer, for your patience and guidance.

REFERENCES

[1] Sunitha Abburu. 2012. A survey on ontology reasoners and comparison. *International Journal of Computer Applications* 57, 17 (2012).

[2] Franz Baader. 1999. Logic-Based Knowledge Representation. In *Artificial Intelligence Today*. 13–41. https://doi.org/10.1007/3-540-48317-9_2

[3] Franz Baader. 2003. *The description logic handbook: Theory, implementation and applications*. Cambridge university press.

[4] Franz Baader, Diego Calvanese, Deborah L McGuinness, Daniele Nardi, and Peter F Patel-Schneider. 2003. *The Description Logic Handbook: Theory, Implementation, and Applications*. Vol. 32. 622 pages.

[5] Tim Berners-Lee, James Hendler, and Ora Lassila. 2001. The Semantic Web. *Scientific American* 284, 5 (2001), 34–43.

[6] Piero A. Bonatti, Carsten Lutz, and Frank Wolter. 2006. Description logics with circumscription. In *Tenth International Conference on Principles of Knowledge Representation and Reasoning (KR)*. 400–410.

[7] Ronald J Brachman and Hector J Levesque. 1984. The tractability of subsumption in frame-based description languages. In *AAAI*, Vol. 84. 34–37.

[8] Ronald J. Brachman and Hector J. Levesque. 2004. *Knowledge Representation and Reasoning*. 1–381 pages. https://doi.org/10.1016/B978-1-55860-932-7.X5083-3

[9] Ronald J Brachman and James G Schmolze. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive science* 9, 2 (1985), 171–216.

[10] Katarina Britz, Giovanni Casini, Thomas Meyer, Kody Moodley, Uli Sattler, and Ivan Varzinczak. 2017. Rational Defeasible Reasoning for Description Logics. *Journal of Artificial Intelligence Research* (2017).

[11] Katarina Britz, Giovanni Casini, Thomas Meyer, Kody Moodley, and Ivan Varzinczak. 2013. *Ordered Interpretations and Entailment for Defeasible Description Logics*. Technical Report. UKZN CSIR Meraka Centre for Artificial Intelligence Research.

[12] Katarina Britz, Giovanni Casini, Thomas Meyer, and Ivan Varzinczak. 2013. Preferential Role Restrictions. In *Description Logic Workshop*. Centre for Artificial Intelligence Research, 93–106.

[13] Katarina Britz, Thomas Meyer, and Ivan Varzinczak. 2011. Semantic foundation for preferential description logics. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 7106 LNAI. 491–500.

[14] Giovanni Casini, Thomas Meyer, Kody Moodley, Uli Sattler, and Ivan Varzinczak. 2015. Introducing Defeasibility into OWL Ontologies. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 9367. 409–426.

[15] Giovanni Casini, Thomas Meyer, Kody Moodley, and Ivan Varzinczak. 2013. Towards Practical Defeasible Reasoning for Description Logics. In *Proceedings of the 26th International Workshop on Description Logics*. 587–599.

[16] Giovanni Casini, Thomas Meyer, Ivan Varzinczak, and Kody Moodley. 2013. Nonmonotonic Reasoning in Description Logics: Rational Closure for the ABox. In *Proceedings of the 26th International Workshop on Description Logics*. 600–615.

[17] Giovanni Casini and Umberto Straccia. 2010. Rational Closure for Defeasible Description Logics. In *European Workshop on Logics in Artificial Intelligence*. Springer, 77–90.

[18] Randall Davis, Howard Shrobe, and Peter Szolovits. 1993. What Is a Knowledge Representation? *AI Magazine* 14, 1 (1993), 17. https://doi.org/10.1609/AIMAG.V14I1.1029 arXiv:arXiv:1011.1669v3

[19] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. 2014. HermiT: An OWL 2 Reasoner. *Journal of Automated Reasoning* 53, 3 (2014), 245–269. https://doi.org/10.1007/s10817-014-9305-1

[20] Nicola Guarino. 1998. Formal Ontology and Information Systems. *Proceedings of the first international conference* June (1998), 3–15. https://doi.org/10.1.1.29.1776 arXiv:NIHMS150003

[21] Matthew Horridge and Sean Bechhofer. 2011. The OWL API: A Java API for OWL ontologies. *Semantic Web* 2, 1 (2011), 11–21.

[22] Ian Horrocks and Peter Patel-Schneider. 2004. Reducing OWL entailment to description logic satisfiability. In *Web Semantics*, Vol. 1. 345–357. https://doi.org/10.1016/j.websem.2004.06.003

[23] Peihong Ke and Ulrike Sattler. 2008. Next steps for description logics of minimal knowledge and negation as failure. In *CEUR Workshop Proceedings*, Vol. 353. https://doi.org/10.1145/505372.505373

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

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

[26] Thomas Lukasiewicz. 2008. Expressive probabilistic description logics. *Artificial Intelligence* 172, 6-7 (2008), 852–883. https://doi.org/10.1016/j.artint.2007.10.017

[27] Deborah L McGuinness and Frank van Harmelen. 2004. OWL Web Ontology Language Overview. *W3C recommendation 10.2004-03* 2004, February (2004), 1–12.

[28] Thomas Meyer, Kody Moodley, and Uli Sattler. 2014. DIP: A defeasible-inference platform for OWL ontologies. In *CEUR Workshop Proceedings*.

[29] Kody Moodley. 2015. *Practical Reasoning for Defeasible Description Logics*. Ph.D. Dissertation. University of KwaZulu-Natal.

[30] Kody Moodley, Thomas Meyer, and Uli Sattler. 2014. Practical Defeasible Reasoning for Description Logics. In *Frontiers in Artificial Intelligence and Applications*, Vol. 264. 191–200.

[31] Kody Moodley, Thomas Meyer, and Ivan Varzinczak. 2012. A Defeasible Reasoning Approach for Description Logic Ontologies. In *South African Institute for Computer Scientists and Information Technologists Conference, SAICSIT 2012*. 69–78.

[32] R. Reiter. 1980. A logic for default reasoning. *Artificial Intelligence* 13, 1-2 (1980), 81–132. https://doi.org/10.1016/0004-3702(80)90014-4

[33] F. Sowa, John. 2000. *Knowledge representation: Logical, Philosophical, and Computational Foundations*. 594 pages. https://doi.org/10.1162/089120101750300544

[34] Reid Swan. 2017. Preferential Reasoning for Ontologies. (2017). https://people.cs.uct.ac.za/~SWNREI001

[35] W3C OWL Working Group. 2012. OWL 2 Web Ontology Language Document Overview. *OWL 2 Web Ontology Language* December (2012), 1–7.