EPiC Series in Computer Science
Volume 35, 2015, Pages 67–80
LPAR-20. 20th International Conferences on Logic for Programming, Artificial Intelligence and Reasoning - Short Presentations

Reasoning in the presence of inconsistency through Preferential $\mathcal{ALC}$

Graham Deane, Krysia Broda, and Alessandra Russo

Imperial College London {graham.deane10,k.broda,a.russo}@imperial.ac.uk

Abstract

This paper presents an inconsistency tolerant semantics for the Description Logic $\mathcal{ALC}$ called *Preferential* $\mathcal{ALC}$ (p-$\mathcal{ALC}$). A p-$\mathcal{ALC}$ knowledge base is comprised of *defeasible* and *non-defeasible* axioms. The defeasible ABox and TBox are labelled with *confidence weights* that could reflect an axiom's provenance. Entailment is defined through the notion of *preferred interpretations* which minimise the total weight of the inconsistent axioms. We introduce a modified $\mathcal{ALC}$ tableau algorithm in which the open branches give rise to the preferred interpretations, and show that it can compute p-$\mathcal{ALC}$ entailment by refutation. The modified algorithm is implemented as an incremental answer set program (ASP) that exploits *optimisation* to capture preferred interpretations of p-$\mathcal{ALC}$.

1 Introduction

Web Ontology Languages[25] (OWL) are based on Description Logics, a family of decidable logics that offer low complexity of reasoning. Popular reasoners such as Hermit[13] and Fact++[24] assume Tarskian model theoretic semantics and thus require a consistent Knowledge Base (KB). However, for many applications, such as multi-agent systems, knowledge is collated from multiple sources and inconsistencies inevitably arise. Inconsistencies within a KB may come about in any of the three basic settings: direct conflicts between ABox axioms (e.g. $Woman(alex)$ and $\neg Woman(alex)$); contradictions between TBox and Abox axioms (e.g. $Woman(alex)$, $\neg Female(alex)$ and $Woman \sqsubseteq Female$); and potentially conflicting TBox axioms (e.g. a KB including the TBox axioms $Woman \sqsubseteq Human$, $Human \sqsubseteq Animal$, $Woman \sqsubseteq \neg Animal$ can become inconsistent once an individual classified as a $Woman$ is added to it).

The inevitability of inconsistency in Knowledge Bases has inspired the search for alternative solutions that allow reasoning without having to locate and repair the inconsistencies (e.g. [17]). In the literature, two strands of work have been investigated. On one hand, a notion of "typicality" has been proposed for which KBs are designed under the assumption that certain knowledge is typically true but admits "exceptions". Languages are augmented with defeasible implication and supported by some form of non-classical semantics [3, 4, 5, 8, 9, 10, 12, 15, 22]. On the other hand, inconsistency tolerant semantics have been proposed for KBs assumed to be designed for and used by classical reasoners[16, 19, 20, 21, 23, 26]. An example is [19], where entailment is defined in terms of *ABox closed repair semantics*, which consider maximally consistent consequences derivable from subsets of ABox axioms that are consistent with

A.Fehnker, A.McIver, G.Sutcliffe and A.Voronkov (eds.), LPAR-20 (EPiC Series in Computer Science, vol. 35), pp. 67–80

TBox axioms. This method, however, does not provide for "arbitration" between conflicting consequences, limiting therefore the set of consequences that could be derived from an inconsistent KB. This limitation is even more evident when such a notion of entailment is extended to *TBox repairs* [20], because each TBox axiom is treated atomically, it is either included in the inference process or omitted.

This paper follows the second strand of research and proposes an alternative semantics to the problem of reasoning in the presence of inconsistent KBs that does not suffer from the above limitations. A KB is defined as comprising of *defeasible* and *non-defeasible* axioms. Each defeasible axiom is labelled with an integer *weight* that expresses the degree of confidence in that axiom. A *preferential semantics*, called *Preferential* $\mathcal{ALC}$ and denoted as *p*-$\mathcal{ALC}$, is proposed, which defines entailment through a notion of *preferred interpretations*. These are interpretations that preferentially retain the defeasible knowledge in which there is the greatest confidence and use the weights to inform the inference of consequences from conflicting axioms. A tableau-based algorithm, which combines techniques used in DL tableau methods, is also proposed for determining what is provable or not from a given inconsistent KB within our new *p*-$\mathcal{ALC}$ semantics. A prototype implementation of our tableau-based algorithm has been developed using a state-of-the-art incremental answer set solver [11], which exploits the optimisation features of Answer Set Programming (ASP) to incrementally compute preferred interpretations. Preliminary evaluations of our prototype have been conducted to demonstrate the applicability of our approach.

The paper is structured as follows. Section 3 introduces our new notion of preferential semantics. Section 4 presents our tableau-based algorithm for computing consequences in the presence of inconsistencies with respect to the preferential semantics, and Section 5 describes the implementation of our algorithm in Answer Set Programming (ASP). Evaluation and related work are discussed in Sections 6 and 7, while Section 8 concludes the paper and outlines future work.

2 Background

In this section we recall the syntax and semantics of the Description Logic $\mathcal{ALC}$. An $\mathcal{ALC}$ signature is a tuple $\langle N_I, N_R, N_C\rangle$ where N_I, N_R, N_C are finite sets of names that refer, respectively, to *individuals*, *roles* and *named concepts*. Given a signature, *concepts* are either named concepts or complex concepts defined inductively as follows: $C, D = \top|\bot|A|\neg C|C\sqcap D|C\sqcup D|\exists R.C|\forall R.C$, where $A \in N_C$, $R \in N_R$, and $\top$ and $\bot$ denote the top and bottom concept. An ABox (resp. TBox) is a finite set of axioms of the form $C(x)$ or $R(x, y)$ (resp. $C \sqsubseteq D$ or $C \equiv D$) where $x, y \in N_I$, $R \in N_R$, C, D are concepts and $C \equiv D$ abbreviates the set of axioms $C \sqsubseteq D$ and $D \sqsubseteq C$. A knowledge base $\mathcal{K}$ is defined as a tuple $\langle \mathcal{A}, \mathcal{T}\rangle$, where $\mathcal{A}$ is the ABox of $\mathcal{K}$ and $\mathcal{T}$ is the TBox of $\mathcal{K}$. The set of all concepts that can be formed using the signature of $\mathcal{K}$ is referred as the *language* of $\mathcal{K}$.

An interpretation of a knowledge base $\mathcal{K}$ is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, .^{\mathcal{I}}\rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set, called *domain* of the interpretation, and $.^{\mathcal{I}}$ is an interpretation function. The function $.^{\mathcal{I}}$ interprets each individual x, in the signature, as $x^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each named concept C, in N_C, as $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and each role R, in N_R, as $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. It is extended inductively over all concepts that can be formed from a given signature: $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$, $\bot^{\mathcal{I}} = \emptyset$, $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $(\forall R.C)^{\mathcal{I}} = \left\{u \in \Delta^{\mathcal{I}} | \forall v \left[(u, v) \in R^{\mathcal{I}} \rightarrow v \in C^{\mathcal{I}}\right]\right\}$, $(\exists R.C)^{\mathcal{I}} = \left\{u \in \Delta^{\mathcal{I}} | \exists v \left[(u, v) \in R^{\mathcal{I}} \wedge v \in C^{\mathcal{I}}\right]\right\}$. Given a knowledge base $\mathcal{K}$, an axiom Z in $\mathcal{K}$ is said to be *satisfied* by an interpretation $\mathcal{I}$ if and only if one of the following cases holds: (i) $Z = C(x)$ then $x^{\mathcal{I}} \in C^{\mathcal{I}}$; (ii) $Z = R(x, y)$ then $(x^{\mathcal{I}}, y^{\mathcal{I}}) \in R^{\mathcal{I}}$; (iii) $Z = C \sqsubseteq D$ then $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.

An interpretation $\mathcal{I}$ is said to be a *model* of a knowledge base $\mathcal{K}$ if every axiom in $\mathcal{K}$ is satisfied by $\mathcal{I}$. We also say, in this case, that $\mathcal{K}$ is *satisfied* in $\mathcal{I}$. If there is no model of $\mathcal{K}$ then $\mathcal{K}$ is said to be *inconsistent*. Given a knowledge base $\mathcal{K}$ and an axiom Z, $\mathcal{K}$ is said to *entail* Z, written $\mathcal{K} \models Z$, if and only if Z is satisfied in every model of $\mathcal{K}$.

In this paper we assume *uniqueness of names*, namely for every interpretation $\mathcal{I}$ each pair of names $x, y \in N_I$ satisfies $x^{\mathcal{I}} \neq y^{\mathcal{I}}$. Without loss of generality, we assume that all axioms are given in *negation normal form* (NNF), i.e. negation appears only in front of named concepts. Following [2], we will use the notation $\dot{\neg}C$ to indicate that $\neg C$ is written in NNF. In addition, we also assume that in concepts of the form $\exists R.C$, and $\forall R.C$, C will be a named concept or the negation of a named concept. Any complex concept C within the scope of a quantifier, can be reduced to our assumed simpler form by substituting it with a new named concept, say C_n, and adding the axiom $C_n \equiv C$ to the TBox.

3 Preferential $\mathcal{ALC}$

We can now introduce the semantics of our preferential $\mathcal{ALC}$. To accommodate inconsistency we introduce a notion of *defeasible axioms*. Intuitively, these are axioms in the knowledge base that we are prepared to falsify during the reasoning process. A defeasible axiom is defined as an $\mathcal{ALC}$ axiom with an associated positive integer weight w that indicates a measure of confidence in the truth of the axiom. The higher the value the greater the level of confidence. The notation $Z^{[w]}$ denotes a defeasible axiom with weight w. Given a set $\mathcal{S}$ of defeasible axioms, the notation $\mathcal{S}^{-W}$ will be used to refer to $\mathcal{S}^{-W} = \{Z | Z^{[w]} \in \mathcal{S}\}$.

Definition 3.1. *A* p-$\mathcal{ALC}$ *knowledge base* $\mathcal{K}$ *is a tuple* $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$*, where* $\mathcal{A}$ *and* $\mathcal{T}$ *are, respectively, two sets of ABox and TBox* $\mathcal{ALC}$ *axioms, and* $\mathcal{A}_d$ *and* $\mathcal{T}_d$ *are, respectively, two sets of defeasible ABox and TBox axioms such that* $\mathcal{A}$ *and* $\mathcal{A}_d$ *are disjoint and* $\mathcal{T}$ *and* $\mathcal{T}_d$ *are also disjoint.*

In general, weights of the defeasible axioms do not have to be equal, but in the case where they have equal weights a *p-*$\mathcal{ALC}$ knowledge is said to be *uniform*. Furthermore, *p-*$\mathcal{ALC}$ knowledge bases for which $\langle \mathcal{A}, \mathcal{T} \rangle$ is satisfiable are said to be *credible*. From now on we will consider only credible *p-*$\mathcal{ALC}$ knowledge bases unless otherwise stated.

The semantics of *p-*$\mathcal{ALC}$ extends the notion of $\mathcal{ALC}$ interpretations to the defeasible axioms and by introducing a notion of distance of the interpretation.

Definition 3.2. *Let* $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ *be a* p-$\mathcal{ALC}$ *knowledge base. A (*p-$\mathcal{ALC}$*) interpretation of* $\mathcal{K}$ *is a classical* $\mathcal{ALC}$ *interpretation* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, .^{\mathcal{I}} \rangle$ *of* $\langle \mathcal{A} \cup \mathcal{A}_d^{-W}, \mathcal{T} \cup \mathcal{T}_d^{-W} \rangle$*. Let* Z *be a defeasible axiom in* $\mathcal{K}$*,* $Z \in \mathcal{A}_d \cup \mathcal{T}_d$*. The* set of unsatisfied instances *of* Z *with respect to* $\mathcal{I}$*, denoted as* $U(Z, \mathcal{I})$*, is defined as follows*[1]*:*

$$U(Z,\mathcal{I}) = \{\langle C(x)^{[w]}, x^{\mathcal{I}} \rangle\}, \text{ if } C(x)^{\mathcal{I}} \notin \mathcal{I} \qquad \text{where } Z \text{ is } C(x)^{[w]}$$
$$U(Z,\mathcal{I}) = \{\langle R(x,y)^{[w]}, x^{\mathcal{I}} \rangle\}, \text{ if } R(x,y)^{\mathcal{I}} \notin \mathcal{I} \qquad \text{where } Z \text{ is } R(x,y)^{[w]}$$
$$U(Z,\mathcal{I}) = \{\langle C \sqsubseteq D^{[w]}, u \rangle \mid (u \in C^{\mathcal{I}}) \wedge (u \notin D^{\mathcal{I}}) \} \qquad \text{where } Z \text{ is } C \sqsubseteq D^{[w]}$$

The set of unsatisfied instances of defeasible axioms *in* $\mathcal{K}$ *with respect to* $\mathcal{I}$*, denoted as* $\mathcal{U}(\mathcal{K}, \mathcal{I})$*, is defined as*

$$\mathcal{U}(\mathcal{K},\mathcal{I}) = \bigcup_{Z \in \mathcal{A}_d \cup \mathcal{T}_d} U(Z,\mathcal{I}).$$

[1] For the form $R(x,y)^{[w]}$, $y^{\mathcal{I}}$ is not required to define a unique axiom instance.

The distance *of the interpretation $\mathcal{I}$, denoted as $d(\mathcal{U}(\mathcal{K},\mathcal{I}))$, is then given by*

$$d(\mathcal{U}(\mathcal{K},\mathcal{I})) = \sum_{\langle Z^{[w]},u\rangle \in \mathcal{U}(\mathcal{K},\mathcal{I})} w.$$

Defeasible (instances of) axioms that are falsified by an interpretation are said be *defeated*. An interpretation $\mathcal{I}$ of $\mathcal{K}$ is said to be n-distant if $n = d(\mathcal{U}(\mathcal{K},\mathcal{I}))$. Using the notion of distance of interpretations, a partial ordering relation, $\prec$, can be defined over the set of all interpretations of a *p-$\mathcal{ALC}$* knowledge base $\mathcal{K}$. Given two interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$ of $\mathcal{K}$, $\mathcal{I}_1 \prec \mathcal{I}_2$ if and only if $d(\mathcal{U}(\mathcal{K},\mathcal{I}_1)) < d(\mathcal{U}(\mathcal{K},\mathcal{I}_2))$. Preferred interpretations are those interpretations with minimal distance value.

Definition 3.3. *Let $\mathcal{I}$ be an interpretation of a* p-$\mathcal{ALC}$ *knowledge base $\mathcal{K}$. $\mathcal{I}$ is said to be a* preferred *interpretation of $\mathcal{K}$ if and only if (i) $\mathcal{I}$ satisfies $\mathcal{A}\cup\mathcal{T}$ and (ii) there is no other interpretation $\mathcal{I}'$ of $\mathcal{K}$ such that $\mathcal{I}' \prec \mathcal{I}$. $\mathcal{K}$ is said to be n-inconsistent if the preferred interpretations of $\mathcal{K}$ are n-distant.*

The entailment relation of our *p-$\mathcal{ALC}$* is based on all preferred interpretations.

Definition 3.4. *Let $\mathcal{K}$ be a* p-$\mathcal{ALC}$ *knowledge base and let Z be an axiom written in the language of $\mathcal{K}$. Z is a preferred consequence of $\mathcal{K}$, written $\mathcal{K} \mid\!\approx Z$, if and only if Z is satisfied in every preferred interpretation $\mathcal{I}$ of $\mathcal{K}$.*

Using the *$\mathcal{ALC}$ finite model property* (i.e. every satisfiable axiom admits a *finite model*) it is possible to show that for any given *p-$\mathcal{ALC}$* knowledge base there exists an n-distant preferred interpretation, with $n \geq 0$.

Proposition 1. *Let $\mathcal{K}$ be a credible* p-$\mathcal{ALC}$ *knowledge base. There exists a non-negative integer n and a preferred interpretation $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I}$ is n-distant.*

From Proposition 1 and Definition 3.3 it follows that any given credible knowledge base $\mathcal{K}$ is n-inconsistent, for some non negative integer n.

Example 1. *Let $\mathcal{K}_1 = \langle \emptyset, \emptyset, \{R(b,a)^{[1]}, \neg C(a)^{[1]}, (\forall R.C)(b)^{[1]}\}, \emptyset\rangle$. Every interpretation of $\mathcal{K}_1$ fails to satisfy at least one defeasible ABox axiom. If we order all its interpretations according to their distance value, the preferred interpretations are all* 1-distant*, i.e. those interpretations for which the sum of the weighted of unsatisfied defeasible axioms is equal to* 1*. $\mathcal{K}_1$ is therefore* 1*-inconsistent.*

When computing the distance of an interpretation, ABox axioms are treated *atomically* since each axiom is either satisfied or unsatisfied in an interpretation. Different is the case of defeasible TBox axioms. If TBox axioms were treated atomically, a single individual falsifying the axiom in a given interpretation would lead to all other instances of the TBox to be defeated even though the interpretation would not enforce it. In our notion of distance, and therefore preferred interpretations, we consider violations of *instances* of defeasible TBox axioms.

Example 2. *Let $\mathcal{K}_2 = \langle \emptyset, \emptyset, \{C(a)^{[1]}, \neg D(a)^{[1]}, C(b)^{[1]}\}, \{C \sqsubseteq D^{[1]}\}\rangle$. $\mathcal{K}_2$ includes an inconsistency associated with a. The distance of a* p-$\mathcal{ALC}$ *interpretation takes into account* each domain element *for which a TBox axiom is defeated. For instance, the interpretation $\mathcal{I}_1$ where $C^{\mathcal{I}_1} = \{a^{\mathcal{I}_1}, b^{\mathcal{I}_1}\}$ and $D^{\mathcal{I}_1} = \{a^{\mathcal{I}_1}, b^{\mathcal{I}_1}\}$ would be a* 1*-distant interpretation of $\mathcal{K}_2$, whereas the interpretation $\mathcal{I}_2$, where $C^{\mathcal{I}_2} = \{a^{\mathcal{I}_2}, b^{\mathcal{I}_2}\}$ and $D^{\mathcal{I}_2} = \{a^{\mathcal{I}_2}\}$, would be a* 2*-distant interpretation. $\mathcal{I}_1$ would therefore be a preferred interpretation for which $D(b)$ would be true. $\mathcal{K}_2$ is* 2*-inconsistent. $\mathcal{K}_2 \mid\!\approx C(b)$ and $\mathcal{K}_2 \mid\!\approx D(b)$.*

Example 3. *Let a, b, and c be individuals. Let's consider the following ABox axioms: "a belongs to a group of patients" ($P(a)$); "everyone that b referred is healthy ($(\forall R.H)(b)$)", "c is sick" ($S(c)$), "everybody that c referred is sick" ($(\forall R.S)(c)$), and "c referred at least one patient in the group ($(\exists R.P)(c)$)" and the TBox axiom "Healthy and sick are disjoint concepts" ($H \sqsubseteq \neg S$). The defeasible knowledge includes the defeasible ABox axiom "We believe that b referred c" ($R(b,c)^{[1]}$), and defeasible TBox axiom "patients in group P are healthy" ($P \sqsubseteq H^{[1]}$). The domain is represented by the credible knowledge base $\mathcal{K}_3 = \langle \mathcal{A}_3, \mathcal{T}_3, \mathcal{A}_{d3}, \mathcal{T}_{d3} \rangle$, where:*

$\mathcal{A}_3$	:	$P(a)$	***(1)***	$(\forall R.H)(b)$	***(2)***	$S(c)$	***(3)***	$(\forall R.S)(c)$	***(4)***	$(\exists R.P)(c)$	***(5)***
$\mathcal{T}_3$	:	$H \sqsubseteq \neg S$	***(6)***								
$\mathcal{A}_{d3}$	:	$R(b,c)^{[1]}$	***(7)***								
$\mathcal{T}_{d3}$	:	$P \sqsubseteq H^{[1]}$	***(8)***								

$\mathcal{K}_3$ includes two sets of axioms leading to inconsistencies: $\{(2),(3),(6),(7)\}$ and $\{(4),(5),(6),(8)\}$. Each preferred interpretation must satisfy the non defeasible axioms and hence must satisfy $(1)-(6)$. But in the first set, any preferred interpretation that satisfies (2), (3) and (6) must also satisfy $\neg H(c)$ and therefore defeats $R(b,c)^{[1]}$ (i.e. (7)). But any such preferred interpretation must also satisfy (5). So there is some (named or unnamed) individual x in the domain for which $R(c,x)$ and $P(x)$ are satisfied. By (4), $S(x)$ is also satisfied and since $(\neg H \sqcup \neg S)(x)$ is satisfied, so is $\neg H(x)$. Thus, $(\neg P \sqcup H)(x)$ is not satisfied (i.e. (8) is defeated). Hence, each preferred interpretation must be at least 2-distant. Hence, $\mathcal{K}_3$ is 2-inconsistent, and since in every preferred interpretation (8) is satisfied for a, we have that $\mathcal{K}_3 \mid\!\!\approx H(a)$. From the above argument we have $\mathcal{K}_3 \mid\!\!\approx S(c)$ and $\mathcal{K}_3 \mid\!\!\approx (\exists R.(P \sqcap S))(c)$. The latter follows because each preferred interpretation includes some individual reified as x for which $P(x)$ and $S(x)$ are satisfied. We can infer that a is healthy, because he/she belongs to the group, but there is at least one unhealthy individual within this group.

Considering (3) to be defeasible, i.e. $S(c)^{[1]}$, would allow for the possibility that c might not be sick. Now there are 2-distant preferred interpretations in which c is sick and others in which c is healthy. $\mathcal{K}_3 \not\mid\!\!\approx H(c)$ and $\mathcal{K}_3 \not\mid\!\!\approx S(c)$. The conflict can be arbitrated by choosing a higher weight for axiom (3) or (7). For example, considering (7) to be $R(b,c)^{[2]}$ leads to $\mathcal{K}_3 \mid\!\!\approx \neg S(c)$. In an interpretation satisfying $S(c)$, $\neg H(c)$ is also satisfied from (6), and (7) is defeated. But defeating (7) adds a weight of 2 and any such interpretation would be at least 3-distant and therefore not preferred. Having defined the semantics for our preferential $\mathcal{ALC}$, we are now interested in computing the preferred consequences of a given p-$\mathcal{ALC}$ knowledge base. Tableau algorithms underpin many modern description logic reasoners including [13] and [24]. Such algorithms incorporate *blocking strategies* that limit the size of the domains considered during reasoning, in the presence of cycles in the TBox in order to guarantee termination. Inspired by these existing techniques we have developed a modified tableau algorithm that accommodates our notion of preferential consequences (see Section 4). We have then encoded this algorithm (see Section 5) in Answer Set Programming (ASP) and used its *optimisation* features to target minimal (or maximal) satisfaction of constraints when searching for models (or solutions.).

4 Tableau for Preferential $\mathcal{ALC}$

In this section we present an $\mathcal{ALC}$ satisfiability tableau algorithm [2] for the p-$\mathcal{ALC}$ semantics. A satisfiability tableau algorithm is a proof procedure that when applied to a knowledge base generates some partial model of the knowledge base if this is consistent, or a closed tableau otherwise. Many common inference tasks for description logics are implemented using satisfiability tableau algorithms. These are proof by *refutation* algorithms used to check whether

$\mathcal{K} \models C(x)$ by checking whether $\mathcal{K} \cup \neg C(x)$ is satisfiable (i.e. leads to a closed tableau). Before embarking on the presentation of our tableau method it is important to show that the concept of proof by refutation is also true in the context of our *p-$\mathcal{ALC}$* semantics.

Theorem 1. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ be an n-inconsistent* p-$\mathcal{ALC}$ *knowledge base and let $C(x)$ be an ABox axiom in the language of $\mathcal{K}$. Then $\mathcal{K} \mid\!\!\approx C(x)$ if and only if either* (i)$\langle \mathcal{A} \cup \{\neg C(x)\}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ *is inconsistent or* (ii) $\langle \mathcal{A} \cup \{\neg C(x)\}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ *is m-inconsistent, for some $m > n$.*

We now introduce a modified $\mathcal{ALC}$ tableau algorithm for the *p-$\mathcal{ALC}$* semantics. A satisfiability $\mathcal{ALC}$ tableau algorithm begins with the given ABox and applies a set of *rules* to develop all possible expansions (branches). The TBox are expanded by including suitable ABox axioms. Expansions continue until each possible expansion either a) leads to a contradiction, called a *clash*, or (b) no further expansion is possible. In the latter case the branch constructed so far represents a (partial) model of $\mathcal{K}$. Expansions for existential quantifiers introduce names called *parameters* that do not appear in the knowledge base signature. These individuals serve as *witnesses* to the expansions. Each name introduced is required to be *fresh* to the ABox being expanded, meaning it does not appear within this ABox. To ensure termination in the presence of a cyclic TBox, a *blocking* strategy is used. Informally, the idea is to detect when expanding an axiom would provide no new information, i.e. there is some other individual within the ABox that is already required to satisfy the same set of concepts.

The technique is adapted for *p-$\mathcal{ALC}$* by considering all possible valid expansions using the non-defeasible axioms, omitting subsets of the defeasible ABox axioms and TBox axioms from defeasible TBox rule applications. The intuition is that an interpretation based on an open branch which minimises the omissions is maximally consistent and coincides with a preferred interpretation. Algorithm 1 constructs a *p-$\mathcal{ALC}$* tableau branch using the tableau rules defined in Table 1 and the expansions formalised in Definitions 4.1 - 4.2. Note that a branch will only be composed of (non-defeasible) ABox instances and we will sometimes refer to a branch as a sequence of ABox axioms.

Algorithm 1 starts with the non-defeasible ABox axioms and a chosen subset of the defeasible ABox axioms. The remaining (those not chosen) defeasible ABox axioms are added to an *omitted* set $\mathcal{O}_b$. As the ABox axioms are "expanded", all possible applications of $\rightarrow_{\mathcal{T}}$ rule to the non-defeasible TBox axioms in $\mathcal{T}$ are performed. For the defeasible TBox axioms, the application of the $\rightarrow_{\mathcal{T}_d}$ rule may or may not lead to the addition of an ABox axiom to the branch. The set S in line 9 of the algorithm keeps track of the defeasible TBox axiom instances to which $\rightarrow_{\mathcal{T}_d}$ rule is applied. The defeasible TBox axiom instances for which an ABox axiom was not added to the branch are at the end added to the omitted set $\mathcal{O}_b$ (see line 12). However, new individuals may be introduced during the tableau expansion (i.e. see rule $\rightarrow_{\exists\forall}$). To guarantee termination, a blocking strategy is used (see Definition 4.1). This assumes an ordering in which individuals are *introduced*, during the construction of a branch, for which some individual are defined to be *older* than others, so preventing mutual blocking (x blocks y and y blocks x). Once an individual is blocked in a branch, it will never become unblocked[2]. When no further expansions can be applied to a branch, the omitted set corresponds to the defeasible axioms instances that will be assumed to be false (i.e. defeated) in the interpretation constructed from that branch, and the sum of their weights defines the distance of the interpretation. Note that by changing the initial choice of defeasible ABox axioms and/or choosing different additions to a branch of defeasible TBox related axiom instance, Algorithm 1 can generate many possible

[2]Our notion of blocking is an example of *static subset blocking* in [18].

branches. As shown later, those open branches with minimal sum of weights of the defeated axiom instances in the omitted set correspond to preferred interpretations.

Definition 4.1. *Let $\mathcal{A}_0 \subseteq \mathcal{A}_1 \ldots \subseteq \mathcal{A}_n$ denote the sequence of (not closed) ABoxes of a branch in a sequence of applications of tableau rules. An individual x is older than an individual y if x is introduced in $\mathcal{A}_i$ and y is introduced in $\mathcal{A}_j$ where $0 \leq i < j \leq n$. An individual y is blocked by individual x at step $\mathcal{A}_j$ if (i) x is older than y and (ii) $\{C|C(y) \in \mathcal{A}_j\} \subseteq \{C|C(x) \in \mathcal{A}_j\}$. If y is blocked by x we say y is blocked.*

Algorithm 1: A branch generating tableau algorithm

Input: $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$, a credible knowledge base
Output: $\mathcal{A}_b$, an expanded ABox
Output: $\mathcal{O}_b$, the *omitted set*
Choose $\mathcal{A}_o$ a subset of $\mathcal{A}_d$;
$\mathcal{O}_{init} = \{\langle C(x)^{[w]}, x\rangle \mid C(x)^{[w]} \in \mathcal{A}_o\} \cup \{\langle R(x,y)^{[w]}, x\rangle \mid R(x,y)^{[w]} \in \mathcal{A}_o\}$;
$\mathcal{A}_b := \mathcal{A} \cup (\mathcal{A}_d - \mathcal{A}_o)^{-W}$;
$\mathcal{S} := \{\}$;
while $\mathcal{A}_b$ *is open and not complete* **do**
  Choose r an instance of a rule that applies to $\mathcal{A}_b$;
  $\mathcal{A}_b :=$ the expansion of $\mathcal{A}_b$ by r ;
  if r *is* $\rightarrow_{\mathcal{T}_d}$ *for unblocked* x *and* $C \sqsubseteq D^{[w]}$ **then** $\mathcal{S} := \mathcal{S} \cup \{\langle C \sqsubseteq D^{[w]}, x\rangle\}$;
end
$\mathcal{O}_b := \mathcal{O}_{init} \cup \{\langle C \sqsubseteq D^{[w]}, x\rangle | \langle C \sqsubseteq D^{[w]}, x\rangle \in \mathcal{S},\ (\dot{\neg}C \sqcup D)(x) \notin \mathcal{A}_b)\}$;
if $\mathcal{A}_b$ *is open* **then return** $\langle \mathcal{A}_b, \mathcal{O}_b \rangle$;
else return $\bot$;

Rule	Valid expansion of $\mathcal{A}_b$
$\rightarrow_{\sqcap}$	**If** $(C \sqcap D)(x) \in \mathcal{A}_b$, x is not blocked and ($C(x) \notin \mathcal{A}_b$ or $D(x) \notin \mathcal{A}_b$) **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{C(x), D(x)\}$
$\rightarrow_{\sqcup}$	**If** $(C \sqcup D)(x) \in \mathcal{A}_b$, x is not blocked and ($C(x) \notin \mathcal{A}_b$ and $D(x) \notin \mathcal{A}_b$) **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{C(x)\}$ or $\mathcal{A}_e = \mathcal{A}_b \cup \{D(x)\}$
$\rightarrow_{\forall}$	**If** $(\forall R.C)(x) \in \mathcal{A}_b$, x is not blocked and $R(x,y) \in \mathcal{A}_b$ and $C(y) \notin \mathcal{A}_b$ **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{C(y)\}$
$\rightarrow_{\mathcal{T}}$	**If** $(C \sqsubseteq D) \in \mathcal{T}$ and $\exists$ an unblocked individual x in $\mathcal{A}_b$ and $(\dot{\neg}C \sqcup D)(x) \notin \mathcal{A}_b$ **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{(\dot{\neg}C \sqcup D)(x)\}$
$\rightarrow_{\mathcal{T}_d}$	**If** $C \sqsubseteq D^{[w]} \in \mathcal{T}_d$ and there exists an unblocked individual x in $\mathcal{A}_b$ and $\langle C \sqsubseteq D^{[w]}, x\rangle \notin \mathcal{S}$ **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{(\dot{\neg}C \sqcup D)(x)\}$ or $\mathcal{A}_e = \mathcal{A}_b$
$\rightarrow_{\exists\forall}$	**If** $(\exists R.C)(x) \in \mathcal{A}_b$, x is not blocked and $\neg\exists y\,[R(x,y) \in \mathcal{A}_b$ and $C(y) \in \mathcal{A}_b]$ and no other rule applies to $\mathcal{A}_b$ **then** $\mathcal{A}_e = \mathcal{A}_b \cup \{R(x,z), C(z)\} \cup \{D(z) \mid (\forall R.D)(x) \in \mathcal{A}_b\}$ where z is **fresh**

Table 1: *p*-$\mathcal{ALC}$ blocking tableau rules

Definition 4.2. *Let $\mathcal{A}_b$ be a current (open) branch generated starting from a given knowledge base $\langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$. $\mathcal{A}_e$ is a valid expansion of $\mathcal{A}_b$ with respect to $\mathcal{T}$ and $\mathcal{T}_d$ if and only if $\mathcal{A}_e$ is*

generated from $\mathcal{A}_b$ applying an instance of a p-$\mathcal{ALC}$ *tableau rule in $\mathcal{R}$ defined in Table 1, with blocking conditions in Def. 4.1. $\mathcal{A}_b$ is said to be* complete *if there are no valid expansions of $\mathcal{A}_b$ under the set of rules $\mathcal{R}$; $\mathcal{A}_b$ it is said to be* closed *if $\mathcal{A}_b$ includes $C(x)$ and $\neg C(x)$ for some concept C and* open *otherwise.*

Given a *p-$\mathcal{ALC}$* knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$, the application of Algorithm 1 to $\mathcal{K}$ generates possible closed or open branches. Each branch has an associated omitted set $\mathcal{O}_b$ and we call the *distance* of a branch the sum of the weights of the defeasible axiom instances in the omitted set. An open branch is called an *m-minimal* branch, if there is no other open branch whose distance is strictly smaller than m.

Definition 4.3. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ be a credible knowledge base and $\mathcal{B}$ be the set of m-minimal branches generated for $\mathcal{K}$ by Algorithm 1. Let $b = \langle \mathcal{A}_b, \mathcal{O}_b \rangle \in \mathcal{B}$, $P(b)$ denote the set of unblocked parameters introduced in the branch and $\mathcal{A}'_b$ be the set of axioms in $\mathcal{A}_b$ not involving blocked individuals. Let Δ_b be the Herbrand domain based on the signature of $\mathcal{K}$ in which N_I is augmented by the unblocked parameters $P(b)$. The Herbrand interpretation $\mathcal{I}_b$ of $\mathcal{K}$ based on Δ_b is defined as follows. For every $x, y \in \Delta_b$, $A \in N_C$ and $R \in N_R$: $A(x)$ is true iff $A(x) \in \mathcal{A}'_b$; $R(x, y)$ is true if either $R(x, y) \in \mathcal{A}'_b$ or $R(x, z) \in \mathcal{A}_b$ and z is blocked by y and is false otherwise.*

Proposition 2. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ be a credible n-inconsistent knowledge base. Let $\mathcal{B}$ be the set of m-minimal branches generated by Algorithm 1 applied to $\mathcal{K}$. Then $m \geq n$ and for each branch $b \in \mathcal{B}$ the interpretation $\mathcal{I}_b$ constructed from b is an m-distant interpretation of $\mathcal{K}$.*

Proposition 3. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ be a credible n-inconsistent knowledge base. Algorithm 1, applied to $\mathcal{K}$, generates at least one n-minimal open branch.*

Given the above results, to check whether $\mathcal{K} \mid\!\approx C(x)$ the idea would be first enumerate all the branches generated by applying Algorithm 1 to $\mathcal{K}$ to identify the distance m of the minimal branches, this can be done by using a branch-and-bound search (or similar). Then Algorithm 1 can be applied to $\mathcal{K}' = \langle \mathcal{A} \cup \neg C(x), \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ to search for exactly an m-distant branch. If such a branch is found then we now conclude $\mathcal{K} \not\mid\!\approx C(x)$. Otherwise $\mathcal{K}'$ was inconsistent or $\mathcal{K}'$ is n-inconsistent for some $n > m$. In either case, we can conclude that $\mathcal{K} \mid\!\approx C(x)$. In the next section we show how the computation of such branches can be done in Answer Set Programming (ASP), exploiting the optimisation features of (ASP).

5 Implementation in ASP

In this section we present the ASP implementation of our tableau method for computing the entailment relation of our *p-$\mathcal{ALC}$* . It is implemented using Clingo (version 4.5.0)[11]. To make the paper self-contained we briefly summarise the features of the ASP fragment that we use in our implementation. For a full specification of the *ASP-Core*-2 syntax and semantics, the reader is referred to [7].

Terms are constants (integers or strings starting with a lower case letter), variables, represented as strings starting with an upper case letter, or functional terms of the form $f(t_1, ..., t_n)$ where f is a functor, t_i, for $0 \leq i \leq n$, are terms. Atoms are of the form $p(t_1, ..., t_n)$ where p is a predicate name, $n > 0$, and $t_1, ..., t_n$ are terms. Built-in comparison atoms $==, !=, <, >, <=, >=$ follow infix notation. A literal is an atom b or a negated atom *not* b where *not* denotes negation by failure. A rule r takes the form $h_1|...|h_m \leftarrow b_1, ..., b_n$ where $m \geq 0$, $n \geq 0$, h_i are atoms, the symbol $|$ denotes disjunction, $b_1, ..., b_n$, are literals where ,

denotes conjunction. $\{h_1, ..., h_m\}$ ($\{b_1, ..., b_n\}$) is the head (resp. body) of r. A fact is a rule with an empty body, an integrity constraint is a rule with an empty head. A *weak constraint* takes the form $:\sim b_1, ..., b_n[w, t_1, ..., t_m]$ where $n \geq 1$, $b_1, ..., b_n$ are literals, w is an integer, $m \geq 1$ and $t_1, ...t_m$ are terms. A *program* P is a finite set of rules and weak constraints.

HU_P (and HB_P) denotes the Herbrand Universe (and Herbrand Base) of P. A ground instance of P, $gnd(P)$ is obtained by substituting each variable appearing in a rule or weak constraint with an element from HU_P and evaluating built in predicates. Given a program P, $I \subseteq HB_P$ is a (Herbrand) interpretation of P; a rule $r \in gnd(P)$ is *satisfied* by I iff some $h \in \{h_1, ..., h_m\}$ is true w.r.t I when $b_1, ..., b_n$ are true w.r.t I; I is a model of P if every rule in $gnd(P)$ is satisfied by I. The *reduct* of P w.r.t I, denoted P^I, is the set of rules from $gnd(P)$ for which $b_1, ..., b_n$ are true w.r.t I; I is an answer set of P if I is a $\subseteq$-minimal model of the reduct P^I. Weak constraints identify optimal answer sets. The optimality of an answer set S of a program P is the weighted sum of w for each unsatisfied ground instance of weak constraints having a unique set of terms $t_1, ...t_m$. S is optimal for P if no other answer set of P has a smaller optimality.

A given knowledge base $\mathcal{K}$ is represented in ASP as a set of facts denoted $\mathcal{K}^\tau$. Each name N in the signature is translated into an ASP constant N^τ by mapping the first letter to its lower case. Concepts are translated to ground ASP terms using the unary or binary function symbols *neg*, *and*, *or*, *oSome* and *oAll* to denote the constructors $\neg, \sqcap, \sqcup, \exists, \forall$ (resp.). Table 2 defines the inductive mappings used to translate concepts and axioms to ASP terms.

C	**Term C$^\tau$**	**C**	**Term C^τ**	**Z**	**Term Z$^\tau$**
$\top$	$thing$	$C_1 \sqcap C_2 ... \sqcap C_n$	$and(C_1^\tau, (C_2 ... \sqcap C_n)^\tau)$	$C(x)$	$ca(C^\tau, x^\tau)$
$\bot$	$neg(thing)$	$C_1 \sqcup C_2 ... \sqcup C_n$	$or(C_1^\tau, (C_2 ... \sqcup C_n)^\tau)$	$R(x, y)$	$ra(R^\tau, x^\tau, y^\tau)$
$\neg C$	$neg(C^\tau)$	$\exists R.C$	$oSome(R^\tau, C^\tau)$	$C \sqsubseteq D$	$sc(C^\tau, D^\tau)$
(C)	(C^τ)	$\forall R.C$	$oAll(R^\tau, C^\tau)$		

Table 2: Mapping concepts C and axioms Z to ASP terms

Definition 5.1. *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{A}_d, \mathcal{T}_d \rangle$ be a knowledge base with signature $\langle N_C, N_R, N_C \rangle$. The encoding in ASP of $\mathcal{K}$ and $sig(\mathcal{K})$ is defined as follows:*

$K^\tau = \{ax(Z^\tau, 0).|Z \in \mathcal{A} \cup \mathcal{T}\} \cup \{ax(Z^\tau, w).|Z^{[w]} \in \mathcal{A}_d \cup \mathcal{T}_d\}$

$sig(\mathcal{K})^\tau = \{i(x^\tau). \mid x \in N_I\} \cup \{r(R^\tau). \mid R \in N_R\} \cup \{c(C^\tau). \mid C \in N_C\}$

where the translations by τ are given in Table 2.

Considering Example 3, $\mathcal{K}_3$ is encoded by the ASP facts:

$ax(ca(p, a), 0).$ $ax(ca(oAll(r, h), b), 0).$ $ax(ca(s, c), 0).$
$ax(ca(oAll(r, s), c), 0).$ $ax(ca(oSome(r, p), c), 0).$ $ax(sc(h, neg(s)), 0).$
$ax(ra(r, b, c), 1).$ $ax(sc(p, h), 1).$
$i(a).\ i(b).\ i(c).$ $r(r).\ c(h).\ c(p).\ c(s).$

The full ASP encoding of a defeasible knowledge base is given by $ASP(\mathcal{K}) = \mathcal{K}^\tau \cup sig(\mathcal{K})^\tau \cup P_{base} \cup P_{cum}$ where P_{base} and P_{cum} are defined below.

The program P_{base} uses predicates $isa/2$ and $hasa/3$ to represent ABox axioms within a tableau expansion. The supported inference of ground atoms representing non-defeasible ABox axioms is unconditionally, whereas that of atom representing defeasible ABox axiom instances is subject to choice (see rule (a3)):

$$isa(X, C) \leftarrow ax(ca(C, X), 0) \quad \text{(a1)}$$

$$hasa(X,R,Y) \leftarrow ax(ra(R,X,Y),0) \tag{a2}$$
$$isa(X,C) \mid u(ca(C,X),X,W) \leftarrow ax(ca(C,X),W), W > 0 \tag{a3}$$
$$:\sim u(ca(C,X),X,W) \quad [W,ca,C,X] \tag{a4}$$
$$hasa(X,R,Y) \mid u(ra(R,X,Y),X,W) \leftarrow ax(ra(R,X,Y),W), W > 0 \tag{a5}$$
$$:\sim u(ra(R,X,Y),X,W) \quad [W,ra,R,X,Y] \tag{a6}$$

For example, given a non-defeasible ABox $(C \sqcap D)(a)$ in $\mathcal{K}$, answer sets of $ASP(\mathcal{K})$ will include $ax(ca(and(c,d),a),0)$. The atom $isa(and(c,d),a)$ will be included in every answer set by (a1). If the non-defeasible ABox $R(a,b)^{[2]}$ is in $\mathcal{K}$, the ASP program $\mathcal{K}^\tau$ will include $ax(ra(a,b),2)$. By rule (a5), either $hasa(a,r,b)$ or $u(ra(r,a,b),a,2)$ will be in each answer set. The weak constraint (a6) increases the total weight of the answer set by 2 (meaning the answer set is less optimal) when $u(ra(r,a,b),a,2)$ is added to it.

$$isa(X,C) \leftarrow isa(X,and(C,D)) \tag{e1}$$
$$isa(X,D) \leftarrow isa(X,and(C,D)) \tag{e2}$$
$$isa(X,C) \mid isa(X,D) \leftarrow isa(X,or(C,D)) \tag{e3}$$
$$isa(Y,C) \leftarrow isa(X,oAll(R,C)), hasa(X,R,Y) \tag{e4}$$
$$isa(X,or(@neg(C),D)) \leftarrow ax(sc(C,D),0), i(X) \tag{e5}$$
$$isa(X,or(@neg(C),D)) \mid u(sc(C,D),X,W) \leftarrow ax(sc(C,D),W), i(X), W > 0 \tag{e6}$$
$$:\sim u(sc(C,D),X,W) \quad [W,sc,C,D,X] \tag{e7}$$

Rules (e1-e7) capture the expansion rules. The symbol $@neg$ is a custom function implemented in the Lua language[3] and ensures concepts are expressed in NNF; if X is an ASP term representing concept C, $@neg(X) = (\dot{\neg}C)^\tau$.

$$isa(X,thing) \leftarrow i(X) \tag{e8}$$
$$\leftarrow isa(X,neg(thing)) \tag{e9}$$
$$\leftarrow isa(X,C), isa(X,neg(C)), c(C) \tag{e10}$$
$$hw(X,oSome(R,C)) \leftarrow isa(X,oSome(R,C)), hasa(X,R,Y), isa(Y,C) \tag{e11}$$
$$need(X,oSome(R,C)) \leftarrow isa(X,oSome(R,C)), not\ hw(X,oSome(R,C)) \tag{e12}$$
$$used(X) \leftarrow i(X) \tag{e13}$$

Rule (e8) captures the property that every named individual has to belong to the "top" concept, rules (e9, e10) guarantee that answer sets include only consistent expansions. Rules (e11-e13) capture the $\rightarrow_{\exists\forall}$ tableau rule, where the atom $hw(X,oSome(R,C))$ means that "X has a witness to the concept $\exists R.C$". Where no such witness exists, $need/2$ labels that a parameter must be introduced. However, since we do not know a priori how many parameters are required, new parameters and their associated rules are introduced on an "as-needed basis". Grounding and solving is implemented iteratively under script control.

Initially, the program $\mathcal{K}^\tau \cup sig(\mathcal{K})^\tau \cup P_{base}$ is grounded, denoted P_g^0, and solved returning some answer set S^0 or `unsatisfiable`. In the former case a sequence of one or more iterations is carried out, each generating an extension to the initial grounding P_g^0 and then re-solving the extended ground program. Each $need(X,oSome(R,C))$ atom instance in S^0 indicates that a parameter is needed to serve as a witness to the individual X for the concept represented as $oSome(R,C)$. The program P_g^0 is cumulatively extended with the program $gnd(P_{cum})$ giving

[3]Lua functions are distinguished by a leading @ symbol.

P_g^1 (see below) and solved again, either returning an answer set S^1 or `unsatisfiable`. Subsequent iterations are similarly carried out and terminate either when no further parameters are needed or the solver returns `unsatisfiable`. We call the final generated answer sets the *optimal solutions* of program $ASP(\mathcal{K})$.

The program P_{cum} implements the rules required to introduce and expand concepts for a parameter. It begins with the *#program* directive which instructs the grounder to postpone grounding the subsequent rules until requested. Each $need(X, oSome(R, C))$ atom instance is associated with a unique fresh parameter identifier(PID), and used to assign the three arguments in P_{cum}: $\_\_p$ (a PID), $\_\_i$ (an individual X), and $\_\_c$ (a concept $oSome(R, C)$), each triple of arguments leading to a set of ground instances of P_{cum}.

$$\#program\, cum(\_\_p, \_\_i, \_\_c) \tag{c1}$$
$$hasa(\_\_i, R, \_\_p) \leftarrow isa(\_\_i, \_\_c), oSome(R, C) = \_\_c \tag{c2}$$
$$isa(\_\_p, C) \leftarrow isa(\_\_i, \_\_c), oSome(R, C) = \_\_c \tag{c3}$$
$$isa(\_\_p, C) \leftarrow hasa(\_\_i, R, \_\_p), isa(\_\_i, oAll(R, C)) \tag{c4}$$
$$cea(\_\_p, C, \_\_i) \leftarrow oSome(R, C) = \_\_c, isa(\_\_i, \_\_c) \tag{c5}$$
$$cea(\_\_p, C, \_\_i) \leftarrow isa(\_\_i, oAll(R, C)), hasa(\_\_i, R, \_\_p) \tag{c6}$$
$$used(\_\_p) \leftarrow cea(\_\_p, C, \_\_i) \tag{c7}$$
$$isa(\_\_p, thing) \leftarrow used(\_\_p) \tag{c8}$$
$$dnb(Y, \_\_p) \leftarrow used(Y), Y != \_\_p, cea(\_\_p, C, \_\_i), not\, isa(Y, C) \tag{c9}$$
$$b(\_\_p) \leftarrow used(\_\_p), used(Y), Y != \_\_p, not\, dnb(Y, \_\_p) \tag{c10}$$

Rules (c2-c6) capture the $\rightarrow_{\exists\forall}$ rule with respect to fresh parameters. Concepts from the $\rightarrow_{\exists\forall}$-rule are recorded (c5,c6). Parameters serving as a witness are labelled as used (c7) and added to the top concept (c8). Rules (c9-c10) keep track of the blocking mechanism. Atom $dnb(Y, \_\_p)$ states that "Y does not block $\_\_p$", and atom $b(\_\_p)$ that "$\_\_p$ is blocked". Since the grounding calls to P_{cum} are sequential, each used Y was introduced within an earlier grounding step and represents an *older* individual within an expansion. Mutual blocking is prevented by enforcing $Y != \_\_p$. Rules (c11-c20) expand used, unblocked parameters:

$$isa(\_\_p, C) \leftarrow isa(\_\_p, and(C, D)), used(\_\_p), not\, b(\_\_p) \tag{c11}$$
$$isa(\_\_p, D) \leftarrow isa(\_\_p, and(C, D)), used(\_\_p), not\, b(\_\_p) \tag{c12}$$
$$isa(\_\_p, C) \mid isa(\_\_p, D) \leftarrow isa(\_\_p, or(C, D)), used(\_\_p), not\, b(\_\_p) \tag{c13}$$
$$isa(\_\_p, or(@neg(C), D)) \leftarrow ax(sc(C, D), 0), used(\_\_p), not\, b(\_\_p) \tag{c15}$$
$$isa(\_\_p, or(@neg(C), D)) \mid u(sc(C, D), \_\_p, W) \leftarrow ax(sc(C, D), W), W > 0, used(\_\_p), not\, b(\_\_p) \tag{c16}$$
$$:\sim u(sc(C, D), \_\_p, W), used(\_\_p), not\, b(\_\_p) \quad [W, sc, C, D, \_\_p] \tag{c17}$$
$$need(\_\_p, oSome(R, C)) \leftarrow isa(\_\_p, oSome(R, C)), used(\_\_p), not\, b(\_\_p) \tag{c18}$$
$$\leftarrow isa(\_\_p, neg(thing)) \tag{c19}$$
$$\leftarrow isa(\_\_p, C), isa(\_\_p, neg(C)), c(C) \tag{c20}$$

Recall from Section 4 that the query $\mathcal{K} \mid\!\!\approx C(x)?$ can be answered by refutation in two steps. Our implementation of Algorithm 1 in ASP also follows the same approach. We first find an optimal answer set with optimality o of $ASP(\mathcal{K})$, called S_o, which will indicate that $\mathcal{K}$ is o-inconsistent. Then we consider $ASP(\mathcal{K}')$, where $\mathcal{K}'$ is given by $\mathcal{K}$ augmented with $\neg C(x)$. In

KB		T1			T2 (PE)			T2 (DE)			T3		
Name	**n**	**t**	**m**	**i**	**t**	**m**	**i**	**t**	**m**	**i**	**t**	**m**	**i**
`amino-acid0`	0	13:15	836	7.3	0:13	0.5	1.5	2:37	7.3	7.3	28.19	2016	4
`amino-acid1`	1	7:56	319	4.3	0:03	0.2	1.2	5:23	7.9	7.9	50.16	1767	4
`amino-acid5`	5	42.15	985	8.0	3.45	3.5	2	17:50	12.1	9.9	`t-out`	`t-out`	

Table 3: Computation of answer sets from n-inconsistent knowledge bases

this second step we look for an o-optimal answer set. If such an answer set is not found then $\mathcal{K}'$ is either unsatisfiable or m-optimal for $m > o$, and in both cases we can conclude $\mathcal{K} \not\mid\!\approx C(x)$. This is guaranteed by Theorem 1 and the following properties of $ASP(\mathcal{K})$: $ASP(\mathcal{K})$ belongs to the class of Finitely Ground Programs [6], its rules (excluding constraints) are locally stratified, and interpretations constructed from n-minimal branches obtained from Algorithm 1 correspond to n-optimal solutions of $ASP(\mathcal{K})$.

6 Evaluation

We have applied our approach to the `amino-acid` knowledge base[4], used as a benchmark in [21]. This knowledge base includes 46 named concepts, 5 roles and 1 individual, as signature; and, as axioms, 1 concept assertion, 0 role assertions, 238 concept inclusions, 199 concept equivalences and 12 disjoint concepts. We have generated from it `amino-acid0`, a (uniform) 0-inconsistent defeasible p-$\mathcal{ALC}$ knowledge base with 20 individuals[5] in which every axiom is defeasibly asserted with a weight of 1. In addition, two inconsistent uniform knowledge bases were generated by adding further defeasible concept assertions: `amino-acid1` is 1-inconsistent and `amino-acid5` is 5-inconsistent. Three tasks were evaluated for each of these three knowledge bases $\mathcal{K}$: T1) Establish the n-inconsistency of $\mathcal{K}$ by finding one optimal answer set of $ASP(\mathcal{K})$; T2) Prove or disprove entailment for a randomly selected concept assertion $C(x)$ where $C \in N_C$ and $x \in N_I$ by searching for an n-optimal answer set of $ASP(\mathcal{K} \cup \{\neg C(x)\})$, finding such a model proves entailment (PE) and failing to find one disproves entailment (DE); T3) Concept retrieval, i.e. find the concepts to which each individual belongs by computing intersection of all n-optimal answer sets of $ASP(\mathcal{K})$.

Table 3 summarises, for each of these experiments, the computational time (t as minutes:seconds), the total number of models found (m) and the number of grounding iterations of P_{cum} (i). Values are averages of 10 runs using a platform based on an Intel(R) Core(TM) i7-2600 CPU @3.40GHz with 16G of RAM. `t-out` denotes the test was aborted after 60 minutes. The (non) entailments for `amino-acid0` were verified using HermiT[13] and for the others was checked manually. The results show that for each knowledge base proving entailment(PE) is faster than disproving entailment(DE), as is typical for tableau refutation. Increasing the numbers of inconsistencies leads to increased computation time.

7 Related Work

p-$\mathcal{ALC}$ is most closely related to the ICAR repair semantics [19] defined for *DL-Lite*$_\mathcal{A}$. However, the p-$\mathcal{ALC}$ semantics are less cautious than the ICAR semantics. To illustrate this, consider a knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ in *DL-Lite*$_\mathcal{A}$ that has an equivalent formulation in $\mathcal{ALC}$. $\mathcal{K}$ can

[4] Available in the TONES repository `http://www.tonesproject.org/`.
[5] 1 asserted individual for each amino acid.

be expressed as a uniform p-$\mathcal{ALC}$ knowledge $\mathcal{K}' = \langle \emptyset, \mathcal{T}, \mathcal{A}, \emptyset \rangle$, where each defeasible axiom in $\mathcal{A}$ is assigned a weight of 1. For example, consider $\mathcal{K}_4 = \langle \{C(a), R(a,b), C(b)\}, \{C \sqsubseteq \neg\exists R\} \rangle$ $\mathcal{K}_4 \models_{ICAR} C(b)$ and $\mathcal{K}'_4 \not\approx C(b)$. For $\mathcal{K}_5 = \langle \{R(a,b), R(a,c), D(a)\}, \{\exists R \sqsubseteq \neg D\} \rangle$ there are two possible CAR-repairs, the sets $\{R(a,b), R(a,c), \neg D(b)\}$, and $\{D(a)\}$ which have an empty intersection. In contrast, $\mathcal{K}'_5 \not\approx R(a,b), R(a,c), \neg D(a)$ because an interpretation that corresponds to the first CAR-repair is 1-distant and for the second is 2-distant. Error tolerant TBox reasoning in $\mathcal{EL}^*$ using repair semantics was investigated in [20], but focused on a different task. They targeted the removal of unwanted consequences by removing subsets of TBox axioms (treated atomically). The work in [16] and [23] is based on applying Possibilistic logic to $\mathcal{ALC}$ in order to deal with uncertainty. TBoxes are considered atomically and the weights are compared over an absolute scale rather than cumulatively. The semantics can provide a measure of certainty when computing entailment, a feature that is not provided in p-$\mathcal{ALC}$, but doesn't take into account derived consequences. Another approach to inconsistent tolerant reasoning is using 4-valued paraconsistent semantics ([21] for a review). An advantage of the approach is that implementation is possible using existing reasoners. However, inferences that rely on disjunctive syllogism are not possible leading to more cautious semantics than p-$\mathcal{ALC}$.

8 Conclusions and Future work

In this paper we have introduced p-$\mathcal{ALC}$, a preferential $\mathcal{ALC}$, proposed a modified tableau algorithm, together with its ASP implementation, and demonstrated the feasibility of approach using (modified) existing knowledge bases. Future directions include, conduct a more substantial evaluation aimed at assessing the impact on computational time of different ratios between defeasible and non-defeasible axioms, and of considering different weights. We will also assess the impact of incorporating tableau optimisations such as lazy unfolding and early clash detection mechanisms [1]. Confidence weights were used to arbitrate inconsistency and determine a measure of inconsistency for a given knowledge base. We will investigate the relationship between this and other inconsistency measures (e.g. Grant et al. [14]) and identify if such measures can be used to inform the assignment of weights to axioms.

References

[1] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, 2003.

[2] Franz Baader and Ulrike Sattler. An Overview of Tableau Algorithms for Description Logics. *Studia Logica*, 69(1):5–40, 2001.

[3] Katarina Britz, Giovanni Casini, Thomas Meyer, and Kody Moodley. Ordered Interpretations and Entailment for Defeasible DL. Technical report, CAIR,UKZN/CSIR-Meraka, SA, 2013.

[4] Katarina Britz, Johannes Heidema, and Thomas Meyer. Semantic Preferential Subsumption. In *Proc. KR-2008*, pages 476–484, 2008.

[5] Katarina Britz, Thomas Meyer, and Ivan Varzinczak. Semantic Foundation for Preferential Description Logics. *AI 2011: Advances in Artificial Intelligence*, 7106:491–500, 2011.

[6] Francesco Calimeri, Susanna Cozza, Giovambattista Ianni, and Nicola Leone. Computable functions in ASP: Theory and implementation. In *Logic Programming*, volume 5366 LNCS, pages 407–424, 2008.

[7] Francesco Calimeri, Wolfgang Faber, Martin Gebser, Giovambattista Ianni, Roland Kaminski, Thomas Krennwallner, Nicola Leone, Francesco Ricca, and Torsten Schaub. ASP-Core-2 Input language format, 2012.

[8] Giovanni Casini, Thomas Meyer, Kody Moodley, and Ivan Varzinczak. Nonmonotonic reasoning in DLs: Rational Closure for the ABox. *CEUR Workshop Proc.*, 1014:600–615, 2013.

[9] Giovanni Casini, Thomas Meyer, Kodylan Moodley, and Riku Nortjé. Relevant Closure: A New Form of Defeasible Reasoning for DLs. *JELIA*, pages 92–106, 2014.

[10] Giovanni Casini and Umberto Straccia. Rational Closure for Defeasible Description Logics. *Lecture Notes in Computer Science*, 6341:77–90, 2010.

[11] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Clingo = ASP + Control : Preliminary Report. *arXiv:1405.3694v1*, 2014.

[12] L Giordano, V Gliozzi, N Olivetti, and G L Pozzato. A non-monotonic Description Logic for reasoning about typicality. *Artificial Intelligence*, 195:165–202, 2013.

[13] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: An OWL 2 Reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[14] John Grant and Anthony Hunter. Distance-based measures of inconsistency. In *Lecture Notes in Artificial Intelligence*, volume 7958 LNAI, pages 230–241, 2013.

[15] Stephan Grimm and Pascal Hitzler. Defeasible Inference with Circumscribed OWL Ontologies. *ARea:Scalability and Commonsense*, 2008.

[16] Bernhard Hollunder. An Alternative Proof Method for Possibilistic Logic and its Application to Terminological Logics. *Proc. Uncertainty in AI*, pages 327–335, 1994.

[17] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Laconic and Precise Justifications in OWL. *The Semantic Web ISWC 2008*, 5318:323–338, 2008.

[18] I Horrocks and U Sattler. A description logic with transitive and inverse roles and role hierarchies. *Journal of Logic and Computation*, 9(3):385–410, 1999.

[19] Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant Semantics for Description Logics. *RR*, 6333:103–117, 2010.

[20] Michel Ludwig and Rafael Penaloza. Error-Tolerant Reasoning in the Description Logic EL. In *LAI*, volume 8761 of *LNCS*, pages 107–121. Springer International Publishing, 2014.

[21] F Maier, Y Ma, and P Hitzler. Paraconsistent OWL and related logics. *Semantic Web*, 4:395–427, 2013.

[22] Kody Moodley, Thomas Meyer, and Ivan José Varzinczak. A defeasible reasoning approach for description logic ontologies. In *SAICSIT Proc.*, SAICSIT, pages 69–78. ACM, 2012.

[23] Guilin Qi and Jeff Z. Pan. A tableau algorithm for possibilistic description logic ALC. In *The Semantic Web*, volume 5367 LNCS, pages 61–75, 2008.

[24] Dmitry Tsarkov and Ian Horrocks. FaCT++ DL Reasoner: System Description. In *IJCAR*, pages 292–297. Springer Berlin / Heidelberg, 2006.

[25] W3C. OWL 2 Web Ontology Language Document Overview, 2012. (accessed 30-May-2015).

[26] Jinfan Zhu, Guilin Qi, and Boontawee Suntisrivaraporn. Tableaux Algorithms for Expressive Possibilistic Description Logics. In *WI-IAT*, volume 1, pages 227–232. IEEE, 2013.