Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference

A Neo-Topological Approach to Reasoning on Ontologies with Exceptions and Comparison with Defeasible Description Logics

Christophe Jouis, Mohammed Yasin Rahman and Jean-Gabriel Ganascia
LIP6 (CNRS and Universite Pierre et Marie Curie) - 4, place Jussieu - 75005 Paris, France

Abstract

This article compares Defeasible Description Logics (DDL) and Topological Approach to reason on Ontologies with exceptions. DDL is integration between Description Logics and Defeasible Logics to deal with monotonic and non-monotonic parts of the knowledge bases respectively. Topological approach tries to reason on inconsistent knowledge bases using the conventional topological operators e.g., interior, exterior, border and closure. We develop neo-Topology based on topological operators and we make major development and improvements of current Topological approach by properly introducing the "Thickness Border" with strong inference rules. We proof the validity of the inference rules using set operations. We demonstrate both approaches with appropriate example. We show the advantages and disadvantages of both approaches.

1. Introduction

Reasoning on ontologies having incomplete or inconsistent information is a major problem. Adding non-monotonicity on reasoning methods proved to be a good solution to deal with these types of inconsistencies. Defeasible Description logics (DDL) add non-monotonicity to Description logics (DLs). Adding non-monotonicity to Description logics can be done in many other ways (Bonatti, Faella, and Sauro 2009). DDL approach is one of the prominent among them. DDL is a combination between Description Logics and Defeasible Logics. Description Logics are monotonic formalism based on first order logic having well established semantics. On the contrary, Defeasible Logics give the flexibility of non-monotonic reasoning that allow to reason on the knowledge bases having inconsistent or incomplete information but have weak semantics unlike Description Logics. Therefore, the combination of these two logics becomes a strong method to deal with these types of ontologies. However, Topological approach tries to reason on inconsistent knowledge bases using the classical topological operators, e.g., interior, exterior, border and closure. It tries to place the elements of each class into the interior, closure, border and exterior of the class based on the information present in the knowledge base. In neo-Topology a "thickness border" is used inside each class to describe the "intermediate" entities, that is to say entities which are more or less typical related to a class. This article is organized as follows: Section 2 talks about related works. Section 3 makes a comprehensive analysis of DDL and how it handles ontologies with exceptions. Section 4 introduces the notion of General Topology. Section 5 talks about neo-Topology and how it handles exceptions in ontologies. Then we solve one classical exception using both approach. Section 7 makes a general comparison among both the approaches and section 8 concludes the article with indication to the future.

2. Related Works

There is ample amount of literatures on non-monotonic extensions of description logics (Bonatti, Faella, and Sauro 2009). The discussion began long ago (Brewka 1987) and went through developments in stages. The approach became popular during the 90s (1993-1995) (Straccia 1993; Baader 1995). Before we talk about different approaches to handle conflicts in knowledge bases, we will first discuss about non-monotonic reasoning to get the basic ideas. A non-monotonic logic is a formal logic whose consequence relation is not monotonic. It makes the knowledge base consistent. To define consistency, we can show that fact P is true by trying to prove $\neg P$. If we fail we may say that P is consistent (since $\neg P$ is false). Default logic introduces a new inference rule: $\frac{A \cdot B}{C}$, which states if A is deducible and it is consistent to assume B then conclude C. It is similar to Non-monotonic reasoning with the following distinctions: New inference rules are used for computing the set of plausible extensions. In Default logic any non-monotonic expressions are rules of inference rather than expressions.

In a very different way, Descles introduced the concept of Logic of Determination of Objects (LDO) (Descles and Pascu 2004). With every concept F the following are canonically associated : (1) An object called "typical object", τF which represents the concept F as an object. This object is completely undetermined, for instance τ[Motorcycle] = "a motorcycle"; (2) A function δF defined on objects : the image-object is more determined than the argument-object for this function, for instance δ[Motorcycle] δ[Which-is-red] = "a typical red motorcycle"; (3) The intension of the concept, $Int(F)$ conceived as the class of all concepts that the concept F includes", that is a semantic network of concepts structured by the relation "IS-A", for instance Int([Motorcycle]) = {[Object-which-has-an-engine], [Object-which-has-2-wheels], [Vehicle], ...}; (4) The expanse of the concept, $Exp(F)$ which contains all "more or less determined objects" such that the concept F applies to; (5) A part of the expanse is the extension of the concept, $Ext(F)$ which contains all completely determined objects such that the concept F applies to. LDO captures two kinds of objects: typical objects and atypical objects. Typical objects in $Exp(F)$ inherit all concepts of $Int(F)$; atypical objects in $Exp(F)$ inherit only some concepts of $Int(F)$.

3. Defeasible Description Logic

Defeasible Description Logic is a combination of Description Logics and Defeasible Logics. It adds non-monotonicity to DLs (Governatori 2004). It is used to reason on ontologies having inconsistent or incomplete knowledge bases. Description Logics provide strong reasoning mechanism but it can not handle the conflicts in the knowledge domains. Defeasible Logics have non-monotonic reasoning ability which can be used in ontologies having inconsistent knowledge bases. The combination of these two types of logics give a very strong reasoning ability over inconsistent knowledge bases. In this section we introduce both logics and we show the combination of this two formalisms to reason on ontologies with exceptions.

3.1. Description Logics

Description logics are based upon first-order logic and it is a monotonic formalism (Baader et al. 2003; Governatori 2004). DLs consist of atomic concepts and atomic roles. A Description Logic (DL) models concepts, roles and individuals, and their relationships. Atomic concepts are entities in the knowledge base e.g, 'Man' is a concept and 'Mohammed' is an instance of the concept. Atomic roles are used to express binary relationships between individuals. In Description Logic, we can build complex concepts by concept conjunctions, concept disjunctions and concept negations (Baader et al. 2003; Governatori 2004). One or more concepts can be added to define a complex concept or its properties. An example of concept conjunction is: Father $\sqsubseteq$ Man $\sqcap$ Parent. Concept disjunction restricts the definition of a complex concept or at least its properties, to appearing in the set of one concept or the other. An example of concept disjunction is: Person $\sqsubseteq$ Man $\sqcup$ Woman. Concept negation gives us the opportunity to define complex concept with the negation of another concept. An example of concept negation is: Woman $\sqsubseteq$ $\neg$Man $\sqcap$ Person. In Description Logics we also have value of role restriction constructs (Baader et al. 2003; Governatori 2004). There is two types of role restrictions in Description Logics; Universal restrictions ($\forall R.C$) and Existential restrictions ($\exists R.C$). Existential restriction ($\exists R.C$) is the construct that requires at least one of the individuals that are in a specified relationship R belong to the concept C. Examples of both restrictions are given: $\forall$hasChild.female and $\exists$hasChild.female. In $\mathcal{AL}$, negation can only be applied to atomic concepts. For this reason we choose to extend $\mathcal{AL}$ with $\mathcal{C}^-$ ($\mathcal{C}$:Complex concept negation).Therefore we use $\mathcal{ALC}^-$ extension of description logic to combine with defeasibile logics.

3.2. Knowledge bases of DLs

DLs knowledge bases comprised of TBoxes and ABoxes. In TBoxes concept definition is given (Baader et al. 2003; Governatori 2004). Concept definitions define new concepts based on existing concepts. The following example illustrates concept definition Woman $\equiv$ Person $\sqcap$ Female. ABoxes contain assertional information. They define specific roles or concepts and change based on circumstances. The following example illustrates the type of information ABoxes have: (1) a concept instance such as Person(Mohammed) ; (2) a role instance such as Father(Mohammed, Yanis).

3.3. Reasoning in DLs

The basic reasoning method in DLs is subsumption. If we have two concepts C and D and a knowledge base $\sum$, then D subsumes C in $\sum$ is written as follows: $\sum \models C \sqsubseteq D$. For example: $Man \sqsubseteq Person$ implies that the concept Person is more general than the concept Man. We can also have concept equivalence. The reasoning can be written as follows: $\sum \models C \equiv D$. Another reasoning method is concept satisfiability. The reasoning can be written as follows: $\sum \not\models C \equiv \bot$. It means that concept C is not equivalent to empty set.

3.4. Defeasible Logics

Defeasible Logic is a non-monotonic reasoning proposed by Donald Nute (Governatori 2004). It has less computational complexity and it is easy to implement. Defeasible Logic is flexible enough to deal with many intuitions of non-monotonic reasoning (Governatori 2004). It has many application such as legal reasoning, automated negotiation, contracts, business rules, and multi agent systems. 3.4.1. Preliminaries. A defeasible theory contains five different kinds of knowledge: facts, strict rules, defeasible rules, defeaters, and a superiority relation (Governatori 2004): (1) Facts: These are statements which are indisputable. Such as, "Mohammed is a lecturer" is a fact which we can't dispute. In the form of logic we can write Lecturer(Mohammed); (2) Strict rules: These are rules which holds, whenever the premises are indisputable (e.g., facts) then so is the conclusion. Such as, Lecturers are faculty member"" is a strict rule. It means that if somebody e.g., 'x' is a Lecturer then we can conclude that 'x' is a faculty member. We can write formally: Lecturer(x) $\rightarrow$ FacultyMember(x); (3) Defeasible rules: These are rules which can be defeated by contrary evidence. Such as, "people giving lectures are faculty members" is defeasible rule because we can provide contrary evidence that is to show that some people who are not faculty members can also give lectures. We can write formally: GiveLecturers(x) $\Rightarrow$ FacultyMember(x). (4) Defeaters: They are used to defeat the defeasible rules by providing contrary evidence. They prevent conclusions. Such as, "tutors might not be faculty members" statement can defeat the previous statement "people giving lectures are faculty members". We can write formally: Tutor(x) $\rightsquigarrow$ ¬FacultyMember(x).The main thing here is, if somebody is tutor, it is not enough to conclude that he/she is a faculty member. It does not say that, if Tutor(x) then ¬FacultyMember(x), rather it just prevents the conclusion due to the lack of information ; (5) Superiority relation: It defines priorities among rules. It is used when one rule overrides the conclusion of another rule . If we have the following two rules: r: GivesLectures(x)⇒ FacultyMember(x): r': GuestLecturer(x)⇒ ¬ FacultyMember(x). The above mentioned rules contradict each other and we can't draw any conclusion whether a guest lecturer is faculty member or not a faculty member. In this situation, a superiority relation "<" is introduced to give one rule more priority than the other. Therefore if we add $r' < r$, then we can conclude that a guest lecturer is not a faculty member. We only use this superiority relation in case of contradictory rules.

3.5. Defeasible Description Logics

We combine the Description Logic ($\mathcal{ALC}^-$) with defeasible logic. In defeasible description logic we consider that the monotonic part is handled by description logic and the non-monotonic part is handled by defeasible logic (Governatori 2004). In defeasible description logic, the ABox contains the set of facts, and the TBox contains the monotonic part of the rules in a defeasible theory. Strict rules in TBox can be characterized precisely (Antoniou and Wagner 2003). Specifically, given an inclusion axiom: $\sqcap_{i=1}^{n} C_i \sqsubseteq \sqcap_{j=1}^{m} D_j$. This inclusion axiom is equivalent to the following set of strict rules (in TBox): $C_1, ...C_n \rightarrow D_1$. $C_1, ...C_n \rightarrow D_m$. C_i and D_j are atomic concepts when $n = m = 1$. The contrapositive of the inclusion axiom is also included: $\neg D_j \rightarrow \neg C_i$.Now we can deal with the monotonic part of the defeasible description logic knowledge base by using structural subsumption of $\mathcal{ALC}^-$ or derivability of defeasible logic. Non-monotonicity is added by introducing defeasible logic rules, and defeasible logic proof theory to a knowledge base in $\mathcal{ALC}^-$. Defeasible description logic has the following structure: $(\mathcal{A}, \mathcal{T}, R, <)$. Here, $\mathcal{A}$ is the ABox, $\mathcal{T}$ is the TBox, R is a set of rules (strict rules, defeasible rules and defeaters), and $<$ is the superiority binary relation defined on rules in R.

4. Topological Approach: General Topology

Topology may be defined as the study of places in a space, their characteristics and their properties (Jouis et al. 2012). Let E be any set and let T be a family of sub-sets of E. T is a topology on E if: (1) both the empty set and E are elements of T; (2) any union of elements of T is an element of T; any intersection a finite number of elements of T is an element of T. If T is a topology on E, then E together with T is called a topological space. All sub-sets in T are called open. Note that not all sub-sets of E are in T: a sub-set of E is said to be closed if its complement is in T (i.e. it is open). A sub-set of E may be open, closed, both or neither. The empty set is open; the union of any number of open sub-sets is open; the intersection of a finite set of open sub-sets is open. In this topological space we can represent a network of concepts and of semantic relationships between these concepts: (1) Entities: are points or elements in the space; (2) Classes: clusters or groups of entities in the space.For each class we can apply the following classical topological operators:(1) Interior (i): The interior of a class 'A', marked $i(A)$, consists of all the elements which satisfy all the properties of that class; (2) Border (bo): The border of a class 'A', marked $bo(A)$, consists of all the elements which does not satisfies all the properties of that class; and (3) Closure (cl):The closure of a class 'A', marked $cl(A)$, consists of all the elements which are intuitively "close to A". It means the elements can reside in the area between the interior and the border of its class. : The exterior of a class 'A', marked $e(A)$, consists of all the elements which does not satisfy any of properties of that class. We demonstrate the concept of general topology by the following example:

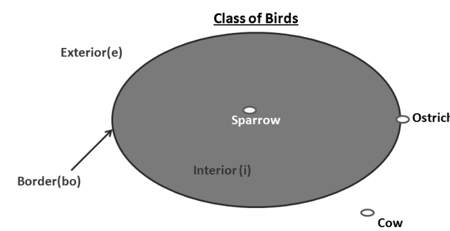

Figure 1: Class of birds in General Topology

In Fig. 1 we have demonstrated the class of 'Birds' using General topology. We say that 'sparrow' is a typical element of the class 'Birds' because it satisfies all the properties of the class as a result 'Sparrow' is at the interior of the class. On the other hand we consider 'Ostrich' as an atypical element of the class 'Birds' because it does not satisfy one of the major properties 'To Fly' of the class as a result 'Ostrich' is at the border of the class. We consider 'Cow' at the exterior of the class because it does not satisfy any properties of the class 'Birds'.

5. Development of neo-Topology

5.1. Degree of Typicality

In this section, we use the concept of Typicality Degree (Jouis et al. 2012) to assign an element 'L' (in a class) which does not match all the properties of a class e.g., 'A'. This concept may be connected to the border of 'A' and enables a model of divergences to be constituted. We create a "thickness" (see Fig. 2)within the border which defines a topological area where we can assign those elements of the class which are neither typical nor atypical rather falls in between. We call this model "neo-Topology".

5.2. Neo-Topology

A neo-Topology can be used to describe the "intermediate" entities (elements and classes), that is to say entities which are more or less typical related to a class. In order to do this,

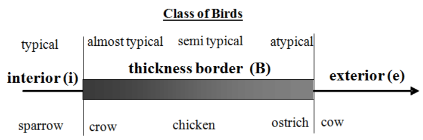

Figure 2: Concept of Thickness Border

we divide a class e.g., 'A' (see Fig. 3), into the following different areas:

Figure 3: neo-Topology with Thickness Border

(1) The interior of class 'A', denoted by "iA", contains all the elements of 'A' that satisfy all the properties of 'A';

(2) An interior border of 'A' is denoted by "L_iA" and an exterior border is denoted by "L_eA". A thickness border of 'A' is denoted by "BA" which is the area between "L_iA" and "L_eA". "L_iA" and "L_eA" borders are "strict" in the sense of general topology, while "BA" can contain both elements and classes. The thickness border contains entities (classes or elements) that are "more or less typical" related to class 'A'. For example, in Fig. 3, 'X' is an "almost typical", while 'Y', which is located in the thickness border is "semi-typical", and 'Z', which is located in "L_eA", is "atypical";

(3)The exterior of class 'A', denoted by "eA", contains all entities that are unrelated to the class 'A'. This new class representation with a thickness border is called "neo-Topology".

5.3. Neo-Topological relations

In neo-Topology, we use the following relationships for elements: (1) Membership of an entity E at the interior of a class A: $E \in_i A$, it implies, E and its neighbourhood are typical relative to A; (2) Membership of an entity E at the interior border of a class A: $E \in_{L_i} A$, it implies, E and its neighborhood are at the interior border of A; (3) Membership of an entity E at the thickness border of a class A: $E \in_B A$, it implies,E and its neighborhood are at the thickness border of A; (4) Membership of an entity E at the exterior border of a class A': $E \in_{L_e} A$, it implies, E and its neighborhood are at the exterior border of A. In neo-topology, we use the following relationships for classes: (1) Inclusion of the class B at the interior of the class A: $B \subset_i A$, it implies, all the elements of class B are at the interior of class A ; (2) Inclusion of the class B at the interior border of the class A: $B \subset_{L_i} A$, it implies, all the elements of class B are at the interior border of class A; (3) Inclusion of the class B at the thickness border of the class$A$: $B \subset_B A$, it implies, class B is at the thickness border of class A; (4) Inclusion of the class B at the exterior border of the category A: $B \subset_{L_e} A$, it implies, all the elements of class B are at the exterior border of class A.

5.4. Inference rules in neo-Topology

Using the relationships of neo-Topology, we deduce inference rules for elements and classes using all possible combinations (See Table 1 and 2) (Jouis et al. 2012).

	$B \subset_{L_i} C$	$B \subset_B C$	$B \subset_{L_e} C$
$E \in_i B$	1	5	9
$E \in_{L_i} B$	2	6	10
$E \in_B B$	3	7	11
$E \in_{L_e} B$	4	8	12

Table 1: Inference rules for elements

In Table 1, E is an element and B and C are classes. From Table 2 we demonstrate rule no 1.
1. $(E \in_i B) \wedge (B \subset_{L_i} C) \rightarrow (E \in_{L_i} C)$

	$B \subset_{L_i} C$	$B \subset_B C$	$B \subset_{L_e} C$
$A \subset_{L_i} B$	13	16	19
$A \subset_B B$	14	17	20
$A \subset_{L_e} B$	15	18	21

Table 2: Inference rules for classes

In Table 2, A, B, and C are classes. From Table 3 we demonstrate rule no 18.
18. $(A \subset_{L_e} B) \wedge (B \subset_B C) \Rightarrow (A \subset_B C)$

5.5. Mathematical Proofs of the inference rules

In this section we proof the validity of the inference rules using the set operations. From the Fig. 4 we can write:

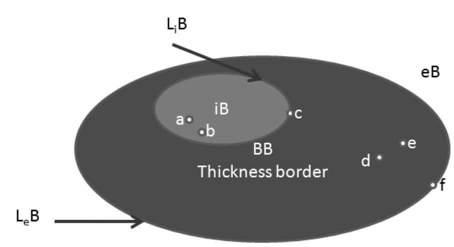

Figure 4: A class with thickness border

$iB = \{a, b\}; L_iB = \{c\}; BB = \{d, e\}; L_eB = f; eB = \bar{A}$
Proof: Rule 1. $(E \in_i B) \wedge (B \subset_{L_i} C) \rightarrow (E \in_{L_i} C)$ and $(E \in_i B)$ means $iB = \{a, b, E\}$ and $B = \{a, b, E, c, d, e, f\}$ (see Fig. 5)
We assume, $L_iC = \{g, h, i\}$. So, $(B \subset_{L_i} C)$ means $LiC = B \cup \{g, h, i\} = \{a, b, \mathbf{E}, c, d, e, f, g, h, i\}$ (see Fig. 6). We can observer that, $(E \in_{L_i} C)$. So, the inference rule $(E \in_i B) \wedge (B \subset_{L_i} C) \rightarrow (E \in_{L_i} C)$ is true. In the same way we can proof all the 12 inference rules that we extracted from table 1.

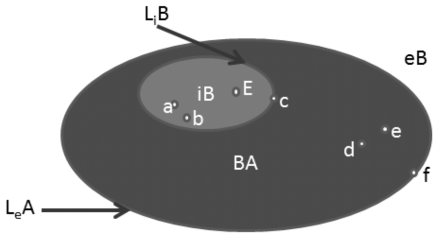

Figure 5: Class B and its elements

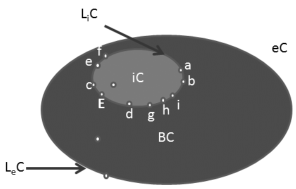

Figure 6: Class C and its elements

Proof: Rule 18. $(A \subset_{L_e} B) \wedge (B \subset_B C) \to (A \subset_B C)$

We assume, $A = \{a, b, c\}, L_eB = \{d, e\}, and\ B = \{d, e, f\}$. $(A \subset_{L_e} B)$ implies $L_eB = A \cup \{d, e\} = \{a, b, c, d, e\}$. Now, $B = \{a, b, c, d, e, f\}$. We assume, $BC = \{g, h\}$, So, $(B \subset_B C)$ implies, $BC = B \cup \{g, h\} = \{$**a,b,c**$, d, e, f, g, h\}$ So, $(A \subset_B C)$. The inference Rule $(A \subset_{L_e} B) \wedge (B \subset_B C) \to (A \subset_B C)$ is true. In the same way we can proof all the 9 inference rules that we extracted from table 2.

6. Problem solving

In this section, we solve the classical example demonstrating conflicts in the knowledge bases using both approaches. We propose our solution with neo-Topological approach. We also discuss the advantages and disadvantages of both approaches.

6.1. Problem Definition

We know that birds (B) can fly (F). And its an important property of birds. So, we say that generally all birds can fly. Now, we observe that penguin(P) is a bird but it can't fly.
In this example we have a clear contradiction in the knowledge base. We will see how both approaches deals with the conflict.

6.2. DDL solution

The knowledge base for the example is given below
$\sum = \{B \sqsubseteq F, P \sqsubseteq \neg F, P \sqsubseteq B, \}$. We can see here that there is a conflict with the statement $B \sqsubseteq F$ and $P \sqsubseteq \neg F$ because we also state that $P \sqsubseteq B$. To solve this conflict we take help of defeasible description logic (Heymans and Vermeir 2002). Here, to establish that penguin can't fly we use the superiority relation. We solve the conflict of the knowledge base by stating: $\{P \sqsubseteq \neg F < B \sqsubseteq F\}$. In this situation, penguin can't fly will have higher priority than the statement birds can fly. So, we can conclude that, penguin can't fly. Now, if we observe that, tweety is a penguin and it can fly. So, it is an exception over exception. So, the new modified knowledge base will be: $\sum = \{B \sqsubseteq F, P \sqsubseteq \neg F, P \sqsubseteq B, \{tweety\} \sqsubseteq P, \{tweety\} \sqsubseteq F\}$. In this case, we can use the properties of defeasible rules (see Section 3.4) and we have the following fact and set of rules R: (1) F is a set of Facts: penguin{tweety}; (2) R is a set of rules ; (3) Strict Rules: $penguin(X) \to bird(X)$; (4) Defeasible Rules: $bird(X) \Rightarrow flies(X)$ and $penguin(X) \Rightarrow \neg flies(X)$; (5)Defeater: $genetically modified Penguin(X) \leadsto flies(X)$; (6) < is a superiority relation on R; (7) $r : bird(X) \Rightarrow flies(X)$; $r' : penguin(X) \Rightarrow \neg flies(X)$.
So, the defeater defeats the statement "penguin can't fly". On the other hand, the superiority relation $r' < r$ gives higher priority to the statement "penguin can't fly". This is how defeasible description logic deals with exceptions in ontologies.

6.3. neo-Topology solution

In neo topology we handle this conflict with the neo-topological operators. We construct the conflicts with the notion of atypicality. As we know, flying is a typical property of birds. So, those birds which can't fly we call them atypical birds. In this case, penguin is an atypical birds. But how much atypical is penguin? Will we place it in thickness border, limit interior or in limit exterior? In this example, we see that, some penguin can also fly. Based on this, we put the class of penguin in the thickness border of the class of birds. In this example we also have an exception of an exception. Tweety which is a penguin but it can fly. To resolve this problem, we say that, tweety is an atypical penguin and we assign tweety in the limit exterior of the penguin class. The proposed solution model is given below

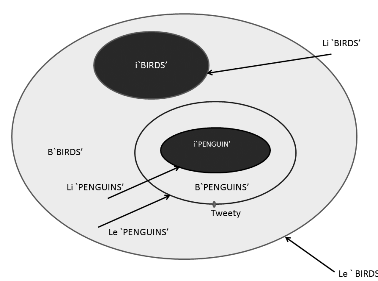

Figure 7: neo-Topological solution model

In the solution model we can see that, the conflicts have been resolved. Tweety is an atypical penguin therefore, it can fly even if the typical penguins don't fly. Penguin class is in the thickness border of the class birds so, penguins are not typical birds which in general can't fly.

7. Comparison: DDL vs neo-Topology

In this section we analyze the solution methods of the example by both approaches. We try to find out the advantages and disadvantages of both approaches. Also we try to see the compatibility of both approaches. We see common in DDL approach is the use of superiority relation (<) to solve conflicts in the knowledge bases. Whenever there is a conflict we need to set priorities among the rules. The higher priority rules gets more preferences about the lower priority rules. Thus the inconsistencies of the knowledge bases are removed. How the priority is set? It is actually based on experience, intuition, credibility (Governatori 2004). Contrary to DDL, neo-Topology solves the conflicts by placing the exceptions outside of the interior of the class and regards the exceptions as atypical entities relative to the corresponding class. In this case we need to identify the necessary properties of typical elements and if any entity (class and/ elements) lacks one of the necessary properties, we call that entity atypical. Based on the numbers of missing necessary properties the entity is placed at L_i, B, L_e and e of the class. If we can properly define the necessary properties of each class, it is easy to solve the conflicts in the knowledge bases. We also try to see the compatibility of both approaches. Are both approaches compatible? In order to see the compatibility we have to see whether everything of topological approach can be expressed by DDL and vise versa. Atomic roles can't be transformed into neo-Topology. The reason behind this inability is, neo-Topology is designed only to handle exceptions in ontology whereas Description Logics are used in many domains where Atomic roles are needed. In case of neo-Topology, the need to transform Atomic roles are unnecessary. As a result we can't find complete compatibility between these two approaches. Apart from atomic roles, other domains are compatible.We are trying to compare the complexity of reasoning of both approaches.

7.1. Advantages and Disadvantages: DDL

The major advantages are : (1) Description Logics have clear semantics. That's why the expressive power of DLs are very high; (2) Flexibility of Defeasible Logics add flexibility to knowledge bases that have partial knowledge; (3) Description logics have strong and conclusive reasoning mechanisms ; (4) Conflicts are handled by prioritizing the rules; (5) Atomic roles make it possible to have relationships among individuals. The major disadvantages are : (1) Trade offs between expressive power and complexity ; (2) Priority settings among conflicts are based on intuition or random most of the time (3) DDL just avoids conflicts using superiority relation but does not give a permanent solution; (4) Time complexity on reasoning is very high.

7.2. Advantages and Disadvantages: neo-Topology

The major advantages are : (1) Strong graphical representation (2) Clear semantics of Topological operators; (3) Exceptions are placed in the proper place to have a complete ontology ; (4) Solves the conflicts and gives a permanent solution to the problem. The major disadvantages are: (1) lacks clear formalisms ; (2) Inference rules are based on conjunctions only ; (3) There is no defined way to know necessary properties ; (4) It is difficult to understand based on what criteria the atypical elements are placed at L_i, B, L_e, and e ; (5) Relationships between elements are not defined.

8. Conclusion

The conflicts in knowledge bases make Description Logics ineffective at reasoning whereas Defeasible Logic can have some useful solution to these conflicts through nonmonotonic reasoning. The combination of defeasible rules with a description logic knowledge base allows to derive defeasible derivability. It's significant to current reasoning methods and widely employed in the field of ontologies. Alternatively, Topology approach solves these conflicts using the general topological operators such as interior, exterior, border, and closure. In neo-topology we have "thickness border" to measure the appropriate degree of typicality. It has strong graphical representations for formalizing ontologies and its main purpose is to deal with exceptions in knowledge bases. In some cases DDL is better than neo-Topology and vise versa. Both approaches can handle conflicts in knowledge bases efficiently. DDL is strong in semantic representation and neo-Topology is strong in graphical representation. However, Description Logics have some limitations when it comes to complexity on reasoning. Future works will focus on comparing the complexity of both approaches.

References

Antoniou, G., and Wagner, G. 2003. Rules and defeasible reasoning on the semantic web. *RuleML, in LNCS,Berlin* 2876:111–120.

Baader, F.; Calvanes, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds. 2003. *The Description Logic Handbook*. Cambridge University Press.

Baader, F.; Hollunder, B. 1995. Embedding defaults into terminological knowledge representation formalisms. *Journal of Automated Reasoning* 14(1):149–180.

Bonatti, P. A.; Faella, M.; and Sauro, L. 2009. Defeasible inclusions in low-complexity dls: Preliminary notes. *In 21st IJCAI* 696–701.

Brewka, G. 1987. The logic of inheritance in frame systems. *In IJCAI* 483–488.

Descles, J.-P., and Pascu, A. 2004. Logic of determination of objects: the meaning of variable in quantification. *Flairs 2005* 491–495.

Governatori, G. 2004. Defeasible description logics. *Rules and Rule Markup Languages for the Semantic Web* 98–112.

Heymans, S., and Vermeir, D. 2002. A defeasible ontology language. *On the Move to Meaningful Internet Systems* (D):1033–1046.

Jouis, C.; Rahman, M.; Ganascia, J.; Jouis, C.; and Guy, F. 2012. Dealing with atypicality using neo-topology: Application to paleontology. *Terminology and Ontology: Theories and Applications* 6.

Straccia, U. 1993. Default inheritance reasoning in hybrid kl-one-style logics. *In IJCAI* 676–681.