Approaches to inconsistency handling in description-logic based ontologies

David Bell[1], Guilin Qi[2], Weiru Liu[1]

[1]School of Electronics, Electrical Engineering and Computer Science
Queen's University Belfast
Belfast, BT7 1NN, UK
{w.liu, da.bell}@qub.ac.uk
[2]Institute AIFB
Universität Karlsruhe
D-76128 Karlsruhe, Germany
gqi@aifb.uni-karlsruhe.de

Abstract. The problem of inconsistency handling in ontologies has recently been attracting a lot of attention. When inconsistency occurs in an ontology, there are mainly two ways to deal with it - we either resolve it or reason with the inconsistent ontology.
In this paper, we give a survey of the existing approaches for handling inconsistency in ontologies and analyze their usability. We focus on Description Logics. We give clear examples to illustrate how to use these approaches to deal with practical problems.

1 Introduction

Knowledge representation for the Semantic Web (SW) requires analysis of the universe of discourse for concepts, definitions, objects, roles, properties,etc, and then selecting a computer-usable version of the results. The sharing of heterogeneous information requires agreed carefully-specified terms to describe the dispersed resources. Ontologies play a core role for the success of the Semantic Web as they provide shared vocabularies for different domains, such as Medicine and Bioinformatics. More generic high-level ontologies are also required for general-purpose SW use (see below). There are many representation languages for ontologies, such as Description Logics (DLs) [1], which have clear semantics and formal properties. The quality of ontologies is important for SW technology. However, in practice, it is often difficult to construct an ontology which is error-free. Inconsistency can occur due to several reasons, such as modeling errors, migration or merging ontologies, and ontology evolution. For example, if ontologies such as such as DOLCE, SUMO and CYC are used in a single document, hundreds of mis-alignments of concepts can be detected, and contradictory statements ('unsatisfiable concepts') might be introduced in particular applications. This will cause logical inconsistency. For example, an unusual individual which does not satisfy some of initial assumptions might be encountered in a running application (see Example 2 below). Do we simply flag that individual as an exception, or do we remove some of the clauses that encode our assumptions?

Current DL reasoners, such as RACER [5] and FaCT [7], can detect logical inconsistency. However, they only provide lists of unsatisfiable classes. The process of *resolving* inconsistency is left to the user or ontology engineers. The need to improve DL reasoners to reason with inconsistency is becoming urgent to make them more applicable.

There are mainly two ways to deal with inconsistent ontologies [8]. One way is to simply avoid the inconsistency and to apply a non-standard reasoning method to obtain meaningful answers. A general framework for reasoning with inconsistent ontologies based on *concept relevance* was proposed in [8]. The idea is to select from an inconsistent ontology some consistent sub-theories based on a *selection function*, which is defined on the syntactic or semantic relevance. Then standard reasoning on the selected sub-theories is applied to find *meaningful* answers. In [18, 25, 14, 13], four-valued logics have been applied to reason with inconsistent ontologies.

The second way to deal with logical contradictions is to resolve logical modeling errors whenever a logical problem is encountered. For example, several methods have been proposed to debug erroneous terminologies and have them repaired when inconsistencies are detected [23, 22, 17, 4].

In this paper, we give a survey of the existing approaches for handling inconsistency in ontologies and analyze their usability. We focus on Description Logics. We give clear examples to illustrate how to use these approaches to deal with practical problems.

The paper is organized as follows. Section 2 provides some basic notions of terminology debugging. We then give an overview of approaches for handling inconsistency in Section 3. Finally, we conclude and discuss the paper in Section 4.

2 Preliminaries

2.1 Description Logics

We now give a brief introduction of Description Logics (DLs) and refer the reader to the DL handbook [1] for more details.

A DL-based ontology (or ontology) $O = (\mathcal{T}, \mathcal{A})$ consists of a set $\mathcal{T}$ of concept axioms (TBox) and role axioms, and a set $\mathcal{A}$ of assertional axioms (ABox). Concept axioms have the form $C \sqsubseteq D$ where C and D are (possibly complex) concept descriptions, and role axioms are expressions of the form $R \sqsubseteq S$, where R and S are (possibly complex) role descriptions. We call both concept axioms and role axioms as terminology axioms. The ABox contains *concept assertions* of the form $C(a)$ where C is a concept and a is an individual name, and *role assertions* of the form $R(a, b)$, where R is a role and a and b are individual names.

The semantics of DLs is defined via a model-theoretic semantics, which explicates the relationship between the language syntax and the model of a domain: An interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain set $\triangle^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$, which maps from individuals, concepts and roles to elements of the domain, subsets of the domain and binary relations on the domain, respectively.

Given an interpretation $\mathcal{I}$, we say that $\mathcal{I}$ satisfies a concept axiom $C \sqsubseteq D$ (resp., a role inclusion axiom $R \sqsubseteq S$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ (resp., $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$). Furthermore, $\mathcal{I}$ satisfies a concept assertion $C(a)$ (resp., a role assertion $R(a,b)$) if $a^{\mathcal{I}}{\in}C^{\mathcal{I}}$ (resp., $(a^{\mathcal{I}},b^{\mathcal{I}}){\in}R^{\mathcal{I}}$). An interpretation $\mathcal{I}$ is called a *model* of an ontology ontology, iff it satisfies each axiom in the ontology.

2.2 Incoherence in DL-based ontologies

We introduce the notion of incoherence in DL-based ontologies defined in [3].

Definition 1 (Unsatisfiable Concept). *A concept name C in an ontology O, is unsatisfiable iff, for each interpretation $\mathcal{I}$ of O, $C^{\mathcal{I}} = \emptyset$.*

That is, a concept name is unsatisfiable in an ontology iff it is interpreted as an empty set by all models of O.

Definition 2 (Incoherent Ontology). *An ontology O is incoherent iff there exists an unsatisfiable concept name in O.*

For example, an ontology $O = \{A{\sqsubseteq}B, A{\sqsubseteq}\neg B\}$ is incoherent because A is unsatisfiable in O. As pointed out in [3], incoherence does not provide the classical sense of the inconsistency because there might exist a model for an incoherent ontology. We first introduce the definition of an inconsistent ontology.

Definition 3 (Inconsistent Ontology). *An ontology O is inconsistent iff it has no model.*

However, incoherence and inconsistency are related with each other. According to the discussion in [3], incoherence is a potential cause of inconsistency. That is, suppose C is an unsatisfiable concept in O, by adding an instance a to C will result in an inconsistent ontology. For example, the ontology $O = \{A{\sqsubseteq}B, A{\sqsubseteq}\neg B\}$ is incoherent but consistent (any interpretation which interprets A as an empty set and B as an nonempty set is a model of O). However, $O' = \{A(a), A{\sqsubseteq}B, A{\sqsubseteq}\neg B\}$ is both incoherent and inconsistent.

In most current work on debugging ontologies, the incoherence problem is discussed at the terminology level. That is, ABoxes are usually considered as irrelevant for incoherence. Therefore, when we talk about an axiom in an ontology, we mean only the terminology axiom.

In the following, we introduce some definitions which are useful to explain logical incoherence.

Definition 4. *[23] Let A be a concept name which is unsatisfiable in a TBox $\mathcal{T}$. A set $\mathcal{T}'{\subseteq}\mathcal{T}$ is a* minimal unsatisfiability-preserving sub-TBox (MUPS) *of $\mathcal{T}$ if A is unsatisfiable in $\mathcal{T}'$, and A is satisfiable in every sub-TBox $\mathcal{T}'' \subset \mathcal{T}'$.*

A MUPS of $\mathcal{T}$ and A is the minimal sub-TBox of $\mathcal{T}$ in which A is unsatisfiable. For example, given TBox $\mathcal{T} = \{C \sqsubseteq A, A{\sqsubseteq}B, A{\sqsubseteq}\neg B\}$. C is an unsatisfiable concept and it has one MUPS, i.e., $\mathcal{T}$. Based on MUPS, we can classify unsatisfiable concepts into derived unsatisfiable concepts and root unsatisfiable concepts as follows:

Definition 5. *[23] Let $\mathcal{T}$ be an incoherent TBox. A TBox $\mathcal{T}'\subseteq\mathcal{T}$ is a* minimal incoherence-preserving sub-TBox (MIPS) *of $\mathcal{T}$ if $\mathcal{T}'$ is incoherent, and every sub-TBox $\mathcal{T}''\subset\mathcal{T}'$ is coherent.*

A MIPS of $\mathcal{T}$ is the minimal sub-TBox of $\mathcal{T}$ which is incoherent. Let us consider the example used to illustrate Definition 4, there is only one MIPS of $\mathcal{T}$: $\{A\sqsubseteq B, A\sqsubseteq\neg B\}$. We say a terminology axiom is *in conflict* in $\mathcal{T}$ if there exists a MIPS of $\mathcal{T}$ containing it.

3 Overview of Approaches for Inconsistency Handling

3.1 Resolving inconsistency

Debug and Repair DL-based Ontologies The first approach for debugging of terminologies is originally proposed in [23]. Their debugging approach is restricted to *unfoldable $\mathcal{ALC}$ TBoxes*, i.e., the left-hand sides of the concept axioms (the defined concepts) are atomic and if the right-hand sides (the definitions) contain no direct or indirect reference to the defined concept. Suppose $\mathcal{T}$ is an incoherent unfoldable TBox and A is an unsatisfiable in it. To calculate a MUPS of $\mathcal{T}$ $w.r.t$ A, we can construct a tableaux from a branch B initially containing only *labelled formula* $(a : A)^{\emptyset}$ (for a new individual name a) by applying the tableaux rules as long as possible.

Terminological diagnosis, as defined in [22], is an instance Reiter's diagnosis from first principles. Therefore, we can use Reiter's algorithms to calculate terminological diagnoses. An important notion in diagnosis is called a *conflict set*, which is an incoherent subset of a TBox. Given a TBox $\mathcal{T}$, a subset $\mathcal{T}'$ of $\mathcal{T}$ is a diagnosis for an incoherent $\mathcal{T}$ if $\mathcal{T}'$ is a minimal set such that $\mathcal{T}\setminus\mathcal{T}'$ is not a conflict set for $\mathcal{T}$.

A drawback of the debugging approach in [23] is that it is restricted to unfoldable $\mathcal{ALC}$ TBoxes. Furthermore, it is based on the tableaux algorithms for DLs. Therefore, it is dependent on the tableaux reasoner. In [17, 9], two orthogonal debugging approaches are proposed to detect the clash/sets of support axioms responsible for unsatisfiable classes, and to identify root/derived unsatisfiable classes. The first one is a glass box approach which is based on description logic tableaux reasoner-Pellet. This approach is closely related to the top-down approach to explanation in [24]. However, the approach proposed in [17] is not limited to DL $\mathcal{ALC}$ and is designed for OWL DL. The second one is a black box approach [9] which is better suitable to identify dependencies in a large number of unsatisfiable classes. The approach is reasoner-independent, in the sense that the DL reasoner is solely used as an oracle to determine concept satisfiability with respect to a TBox. It consists of two main steps. In the first step, it computes a *single* MUPS of the concept and then it utilizes the Hitting Set algorithm to retrieve the remaining ones. This approach is closely related to the bottom up approach to explanation. Based on the debugging approach, in [10], the authors give a tool to repair unsatisfiable concepts in OWL ontologies. The basic idea is to rank erroneous axioms and then to generate a plan to resolve the errors in a given set of unsatisfiable concepts by taking into account the axiom ranks. In [19], a score ordering on terminology axioms is defined to help the user to select axioms to delete.

Example 1. Suppose that we have an ontology $DICE$[1] which contains the following terminologies:

$ax_1 = Brain \sqsubseteq CentralNervousSystem \sqcap BodyPart \sqcap \forall region.HeadAndNeck$

$ax_2 = CentralNervousSystem \sqsubseteq NervousSystem$

$ax_3 = Disjoint(NervousSystem, BodyPart).$

ax_1 says that Brain is part of Central Nervous System and part of Body and it is in the region of Head and Neck. ax_2 says that Central Nervous System is part of Nervous System and ax_3 tells us that Nervous System and Body Part are disjoint. The ontology is incoherent because $Brain$ is claimed to be a part of Nervous System and Body (according to ax_1 and ax_2), whilst Nervous System and Body Part are claimed to be disjoint (according to ax_3). To resolve the incoherence, we can delete any of the axioms ax_i ($i = 1, 2, 3$).

Maximal Satisfiable Terminologies Another way to resolve incoherence in an ontology is to find some subsets of the TBox which are consistent and are maximal w.r.t. set-inclusion. In [16], a tableau-like procedure was proposed to these subsets and some optimization techniques were given to improve the runtime behavior of the procedure. A drawback of the approach in [16] is that it removes an axiom when it is involved in conflict, even if only part of the axiom is responsible for the conflict. So a fine-grained approach was proposed in [12] to generalize the approach in [16]. This approach can not only pinpoint the axioms which are involved in conflict, but also trace which parts of the axioms are responsible for the conflict. Based on the algorithm, a tool was developed for debugging and repairing an incoherent ontology.

In Example 1, there are three maximal satisfiable subsets: $\{ax_1, ax_2\}$, $\{ax_2, ax_3\}$ and $\{ax_1, ax_3\}$. However, ax_1 can be split into three parts:

$Brain \sqsubseteq CentralNervousSystem$

$Brain \sqsubseteq BodyPart$ and

$Brain \sqsubseteq \forall region.HeadAndNeck.$

Only $Brain \sqsubseteq CentralNervousSystem$ and $Brain \sqsubseteq BodyPart$ are involved in the conflict. Therefore, by applying the approach in [12], we can repair the ontology to give the following one: $\{Brain \sqsubseteq \forall region.HeadAndNeck, ax_2, ax_3\}$.

Consistent Ontology Evolution The work in [6] describes a process to support the consistent evolution of OWL DL based ontologies, which ensures that the consistency of an ontology is preserved when changes are applied to the ontology. The process consists of two main phases: (1) *Inconsistency Detection*, which is responsible for checking the consistency of an ontology with the respect to the ontology consistency definition and identifying parts in the ontology that do not meet consistency conditions; (2) *Change Generation*, which is responsible for ensuring the consistency of the ontology by generating additional changes that resolve detected inconsistencies. The authors define methods for detecting and resolving inconsistencies in an OWL ontology after the application of a change. As for some changes there may be several different consistent states of the ontology, *resolution strategies* allow the user to control the evolution.

[1] The ontology $DICE$ is under development at the Academic Medical Center in Amsterdam.

The methods for detecting inconsistencies rely on the idea of a selection function are to identify the *relevant* axioms that contribute to the inconsistency. In the most simple case, syntactic relevance – considering how the axioms of the ontology are structurally connected with the change – is used. Based on the selection function, algorithms to find *minimal inconsistent subontologies* and *maximal consistent subontologies* are presented.

The approach only supports repairs by removing complete axioms from the ontology, a weakening based on a finer granularity is mentioned as an extension, but no algorithms are proposed. The approach does not make a distinction between ABox and TBox axioms, as such both ABox and TBox inconsistencies are trivially supported. Further, the approach does not provide any explicit support for dealing with networked ontologies.

Let us consider an example given in [6].

Example 2. Given a University ontology which contains the following terminology axioms and assertional axioms:

$Researcher \sqsubseteq Person$ (researchers are persons)

$PhDStudent \sqsubseteq Student$ (PhD students are students)

$Student \sqsubseteq \neg Researcher$ (students are not researchers)

$Article \sqsubseteq Publication$ (articles are publications)

$Researcher(Johanna)$ (Johanna is a researcher).

Suppose we now receive a new assertion which says that Johanna is a PhD student, i.e., PhDStudent(Johanna). This assertion is in conflict with the existing ontology because students cannot be researchers whilst Johanna is claimed to be both a PhD student thus a student and a researcher. According to the algorithm in [6], we can delete $Student \sqsubseteq \neg Researcher$ to restore consistency.

Knowledge base revision in description logics In [20], the revised AGM postulates for belief revision in [11] were generalized and two revision operators which satisfy the generalized postulates were given. One operator is the weakening-based revision operator which is defined by weakening of statements in a DL knowledge base. The idea of weakening a terminology axiom is similar to weakening an uncertain rule in [2]. That is, when a term is involved in conflict, instead of dropping it completely, we remove those individuals which cause the conflict. The weakening-based revision operator may result in counterintuitive results in some cases, so another operator was proposed to refine it. It was shown that both operators capture some notions of minimal change.

Let us go back to Example 2. To resolve inconsistency, we can now weaken the terminology axiom $Student \sqcap \neg\{Johanna\} \sqsubseteq \neg Researcher$, where $\{Johanna\}$ is a *nominal*. That is, all students except Johanna are not researchers. In this way, we can add PhDStudent(Johanna) to the ontology consistently.

The weakening-based approach is more fine-grained than the approach in [6]. However, it is computationally harder because we need to find the individuals which are responsible for the conflict.

Knowledge integration for description logics In [15], an algorithm, called *refined conjunctive maxi-adjustment* (RCMA for short) was proposed to weaken conflicting information in a *stratified* DL knowledge base and obtain some consistent DL knowledge bases. To weaken a terminological axiom, they introduced a DL expression, called *cardinality restrictions* on concepts. However, to weaken an assertional axiom, they simply delete it. In [21], the authors first define two revision operators in description logics, one is called a weakening-based revision operator and the other is its refinement. The revision operators are defined by introducing a DL constructor called *nominals*. The idea is that when a terminology axiom or a value restriction is in conflict, they simply add explicit exceptions to weaken it and assume that the number of exceptions is minimal. Based on the revision operators, they then propose an algorithm to handle inconsistency in a *stratified* description logic knowledge base. It was shown that when the weakening-based revision operator is chosen, the resulting knowledge base of their algorithm is semantically equivalent to that of the RCMA algorithm. However, their syntactical forms are different.

3.2 Reasoning with inconsistent ontologies

Coherence-based approaches This kind of approach is based on the idea of removing contradictory information before applying classical reasoning algorithms-especially for question-answering. This can be realized e.g. by starting with an empty (thus consistent) ontology and incrementally selecting and adding axioms to that ontology, which do not result in inconsistency. A general framework for reasoning with inconsistent ontologies based on *concept relevance* was proposed in [8]. The idea is to select from an inconsistent ontology some consistent sub-theories based on a *selection function*, which is defined on the syntactic or semantic relevance. Then standard reasoning on the selected sub-theories is applied to find *meaningful* answers.

Example 3. Given our updated University ontology which contains the following terminology axioms and assertional axioms:
$Researcher \sqsubseteq Person$ (researchers are persons)
$PhDStudent \sqsubseteq Student$ (PhD students are students)
$Student \sqsubseteq \neg Researcher$ (students are not researchers)
$Article \sqsubseteq Publication$ (articles are publications)
$Researcher(Johanna)$ (Johanna is a researcher)
$PhDStudent(Johanna)$ (Johanna is a PhD student).

As we have seen, this ontology is inconsistent. Suppose we want to query if Johanna is a person, i.e. $Person(Johanna)$. We first select the axioms which are directly relevant to the query $Person(Johanna)$, i.e., axioms where the concept $Person$ and/or individual $Johanna$ appear(s). They are $Researcher \sqsubseteq Person$, $Researcher(Johanna)$ and $PhDStudent(Johanna)$. From these axioms, we apply a DL reasoner and we can infer that $Person(Johanna)$. So the answer is ”yes” for the query.

Paraconsistent reasoning on inconsistent ontologies The second approach does not modify the knowledge base but changes the semantics under which it is reasoned with, employing a so-called *paraconsistent semantics*. Unlike the semantics of classical two-valued semantics, the semantics employed in this case uses four truth values, namely for *true* (t), *false* (f), *undetermined* (u) and *overdetermined* (o). The fourth truth value, *overdetermined*, stands for contradictory information. That is, if an assertion $C(a)$ gets assigned the truth value *overdetermined*, then this assertion is considered to be *true* and *false* at the same time.

Let us consider Example 3 again. By applying the four-valued semantics in [13], we can infer that $Student(Johanna)$, i.e. Johanna is a student. At the same time, we can infer that both $Researcher(Johanna)$ (Johanna is a researcher) and $\neg Researcher(Johanna)$ (Johanna is not a researcher). Therefore, by applying the paraconsistent semantics, we may infer contradictory conclusions.

4 Discussion and Conclusion

In this paper we give a brief survey of existing work on inconsistency handling in DL-based ontologies. We divide the existing approaches into two families: the approaches that resolve inconsistency and the approaches that reason with inconsistent ontologies.

When resolving inconsistency, we differentiate logical inconsistency from incoherence. The former is in the sense of inconsistency in classical logic and the latter is more DL-specific. The debugging and repair approaches are usually applied to resolve incoherence. When resolving inconsistency, we can either delete an axiom in the ontology or weaken it. The weakening-based approaches are usually more fine-grained than the deletion-based ones. However, the computational complexity of the former approaches are usually greater.

When reasoning with inconsistent ontologies, we either select some consistent sub-ontologies and apply a DL-reasoner to answer the query or change the semantics of the ontology languages. The first kind of approaches do not have good semantical explanation and are usually syntax-dependent. However, they are proven to be very efficient for reasoning with some real life ontologies. The second kind of approaches have semantical definitions. However, we may draw contradictory conclusions using paraconsistent semantics.

Acknowledgements: The second author was supported by the European Commission NeOn (IST-2006-027595, http://www.neon-project.org/). We also want to thank Yue Ma for her valuable comments on the draft paper.

References

1. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.
2. Salem Benferhat, Souhila Kaci, Daniel Le Berre, and Mary-Anne Williams. Weakening conflicting information for iterated revision and knowledge integration. *Artif. Intell.*, 153(1-2):339–371, 2004.
3. Giorgos Flouris, Zhisheng Huang, Jeff Z. Pan, Dimitris Plexousakis, and Holger Wache. Inconsistencies, negations and changes in ontologies. In *Proc. of AAAI'06*, 2006.

4. Gerhard Friedrich and Kostyantyn M. Shchekotykhin. A general diagnosis method for ontologies. In *Proc. of 4th International Conference on Semantic Web (ISWC'05)*, pages 232–246, 2005.
5. Volker Haarslev and Ralf Möller. RACER system description. In *IJCAR'01*, pages 701–706, 2001.
6. Peter Haase and Ljiljana Stojanovic. Consistent evolution of owl ontologies. In *In Proc. of ESWC'05*, pages 182–197, 2005.
7. Ian Horrocks. The fact system. In *Proc. of International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, pages 307–312. Springer, 1998.
8. Zhisheng Huang, Frank van Harmelen, and Annette ten Teije. Reasoning with inconsistent ontologies. In *Proc. of 19th International Joint Conference on Artificial Intelligence(IJCAI'05)*, pages 254–259. Morgan Kaufmann, 2005.
9. Aditya Kalyanpur, Bijan Parsia, Bernardo Cuenca Grau, and Evren Sirin. Justifications for entailments in expressive description logics. Technical report, University of Maryland Institute for Advanced Computer Studies (UMIACS), 2006.
10. Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and Bernardo Cuenca Grau. Repairing unsatisfiable concepts in owl ontologies. In *ESWC'06*, pages 170–184, 2006.
11. Hirofumi Katsuno and Alberto O. Mendelzon. Propositional knowledge base revision and minimal change. *Artif. Intell.*, 52(3):263–294, 1992.
12. Joey Lam, Jeff Z. Pan, Derek Seeman, and Wamberto Vasconcelos. A fine-grained approach to resolving unsatisfiable ontologies. In *Proc. of WI'06*, 2006.
13. Yue Ma, Pascal Hitzler, and Zuoquan Lin. Algorithms for Paraconsistent Reasoning with OWL. In *Proceedings of ESWC2007*, 2007. To appear.
14. Yue Ma, Zuoquan Lin, and Zhangang Lin. Inferring with inconsistent OWL DL ontology: A multi-valued logic approach. In *Proc. of EDBT Workshops*, pages 535–553, 2006.
15. Thomas Meyer, Kevin Lee, and Richard Booth. Knowledge integration for description logics. In *Proc. of 20th National Conference on Artificial Intelligence (AAAI'05)*, pages 645–650. AAAI Press, 2005.
16. Thomas Meyer, Kevin Lee, Richard Booth, and Jeff Z. Pan. Finding maximally satisfiable terminologies for the description logic alc. In *Proc. of AAAI'06*, 2006.
17. Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Debugging OWL ontologies. In *Proc. of WWW'05*, pages 633–640, 2005.
18. Peter F. Patel-Schneider. A four-valued semantics for terminological logics. *Artificial Intelligence*, 38:319–351, 1989.
19. Guilin Qi and Anthony Hunter. Measuring incoherence in description logic-based ontologies. In *Proc. of ISWC'06*. Springer Verlag, 2007, to appear.
20. Guilin Qi, Weiru Liu, and David A. Bell. Knowledge base revision in description logics. In *Proc. of JELIA'06*, pages 386–398. Springer Verlag, 2006.
21. Guilin Qi, Weiru Liu, and David A. Bell. A revision-based algorithm for handling inconsistency in description logics. In *Proc. of NMR'06*, 2006.
22. Stefan Schlobach. Diagnosing terminologies. In *Proc. of AAAI'05*, pages 670–675, 2005.
23. Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *IJCAI'03*, pages 355–362, 2003.
24. Stefan Schlobach, Zhisheng Huang, and Ronald Cornet. Inconsistent ontology diagnosis: Evaluation. Technical report, Department of Artificial Intelligence, Vrije University Amsterdam; SEKT Deliverable D3.6.2, 2006.
25. Umberto Straccia. A sequent calculus for reasoning in four-valued description logics. In *Proc. of TABLEAUX'97*, pages 343–357, 1997.