Measuring Incoherence in Description Logic-based Ontologies

Guilin Qi[1] and Anthony Hunter[2]

[1]Institute AIFB
Universität Karlsruhe
D-76128 Karlsruhe, Germany
gqi@aifb.uni-karlsruhe.de
[2]Department of Computer Science
University College London, Gower Street
London WC1E 6BT,UK
a.hunter@cs.ucl.ac.uk

Abstract. Ontologies play a core role for the success of the Semantic Web as they provide a shared vocabulary for different resources and applications. Developing an error-free ontology is a difficult task. A common kind of error for an ontology is logical contradiction or *incoherence*. In this paper, we propose some approaches to measuring incoherence in DL-based ontologies. These measures give an ontology engineer important information for maintaining and evaluating ontologies. We implement the proposed approaches using the KAON2 reasoner and provide some preliminary but encouraging empirical results.

1 Introduction

Ontologies play the core role for the success of the Semantic Web (SW) as they provide shared vocabularies for different domains. There are many representation languages for ontologies, such as Description Logics (DLs) [1]. High quality ontologies are important for SW technology. However, in practice, it is often difficult to construct an ontology which is error-free. A common error for an ontology is *incoherence*, i.e. whether there are *unsatisfiable* concepts which is interpreted as an empty set in all the *models* of its terminology. Incoherence can occur for several reasons, such as modeling errors when constructing an ontology and migration or merging of ontologies [16]. For example, when two upper ontologies SUMO and CYC are used in a single document, there exist over 1000 unsatisfiable concepts. Currently, there are many discussions on how to debug and diagnose terminologies in ontologies [9, 8, 17, 16]. Therefore, incoherence is often viewed as negative information in an ontology. However, by measuring incoherence, we can get some useful information for maintaining and evaluating an ontology. For example, by measuring the extent of incoherence of different ontologies, we can give a rank over them. That is, an ontology is more reliable than another one if it contains less incoherent information. Similarly, by measuring the extent of incoherence of different axioms, we get some ranking information on axioms which can be used to resolve incoherence [8]. Furthermore, there is a trade-off in the amount of useful information in an ontology and the amount of coherent information. For example, in the extreme, we could guarantee a coherent ontology by only having the empty ontology. Obviously, this would not be acceptable, and so we need to tolerate the possiblity of some incoherence in an ontology. Once we do tolerate this possibility, we need to consider keeping track of it, perhaps as part of a process of improvement, or as a way of isolating the problematical parts of the ontology until we decide to how to fix those parts. So for these tasks, it is helpful to know where and by how much there is incoherence. This may include consideration of whether it is widespread, or localized. So the size and overlaps of incoherent subsets can be useful diagnostic tools.

Incoherence in ontologies corresponds to *inconsistency* in *knowledge bases* in classical logic, where a knowledge base is a finite set of classical formulae. A knowledge base is inconsistent if and only if there is no model satisfying all its formulae. We can regard the measures of inconsistency for a knowledge base (proposed to date) as falling into one of the following three classes:(**Formula-centric measures**) These measures take into account the number of formulae required for inconsistency, and so fewer formulae in an inconsistency means a higher degree of inconsistency (e.g. [10]); (**Atom-centric measures**) These measures take into account the proportion of the language affected by inconsistency, and so more propositional atoms involved in inconsistency means a higher degree of inconsistency (e.g. [5, 11]); and (**Conflict-centric measures**) These measures take into the account the number of conflicts each formula is involved in, and so if each formula is involved in more conflicts, there is a higher degree of inconsistency (e.g. [6]). Furthermore, given a measure of inconsistency of a knowledgebase, using one of the above possibilities, we can ascribe the blame or responsibility that each formula has in the set by drawing on an approach from game theory, called Shapley values, that allows for a principled assignment [7].

In this paper, we propose some approaches for measuring incoherence in DL-based ontologies. *The approaches proposed in this paper are independent of specific DL languages.* Our approaches are based on the scoring function introduced by Hunter in [6]. There are two classes of measures of incoherence: measures of incoherence for unsatisfiable concepts and measures of incoherence for terminologies. First, we define the scoring function for an unsatisfiable concept and use it to define a score ordering on unsatisfiable concepts. Second, we define the scoring function for a TBox and use it to define an ordering on terminology axioms and an ordering on TBoxes. We implement the proposed approaches using KAON2 reasoner[1] and report preliminary but encouraging experimental results.

This paper is organized as follows. We first give some preliminaries on DLs and related notions on incoherence in Section 2. We then discuss related work in Section 3. Afterwards, we propose some measures of incoherence in DL-based ontologies in Section 4. In Section 5, the applications of measures of incoherence are given. Finally, we report preliminary evaluation results in Section 6 and conclude this paper in Section 7.

[1] c.f. `http://kaon2.semanticweb.org`

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 O, iff it satisfies each axiom in O.

2.2 Incoherence in DL-based ontologies

We introduce the notion of incoherence in DL-based ontologies defined in [4].

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

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 [4], 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 [4], 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 of current work on debugging ontologies, the incoherence problem is often 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. *[17] 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 $\mathcal{T}'' \subset \mathcal{T}'$. The set of all MUPS of $\mathcal{T}$ with respect to A is denoted as $MU_A(\mathcal{T})$*

A MUPS of $\mathcal{T}$ and A is the minimal sub-TBox of $\mathcal{T}$ in which A is unsatisfiable. For example, given an 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 (Root and Derived). *[9] C is a derived unsatisfiable concept in $\mathcal{T}$ iff it satisfies the following condition: $\exists i, j$ such that $MUPS_i(C) \supseteq MUPS_j(D)$, for an unsatisfiable concept D. If C does not satisfy this condition then it is a root unsatisfiable concept.*

Definition 6. *[17] 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. The set of all MIPSs of $\mathcal{T}$ is denoted as $MI(\mathcal{T})$.*

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.

Algorithms have been given to calculate MUPS and MIPS in a given ontology (see [17, 9]). It was shown in [17] that calculating MUPS is the same as the satisfiability check if a glass-box algorithm is used, i.e., the algorithm is based on the description logic tableaux reasoner.

3 Related Work

This work is related to the work on debugging terminologies and resolve incoherence [9, 8, 17, 16]. The first work on debugging erroneous terminologies is reported in [17] where the authors provide a specialized algorithm for the DL $\mathcal{ALC}$. The notions of MUPS and MIPS are introduced to explain logical incoherences there. In [9], two orthogonal debugging approaches are proposed to detect the clash/sets of support axioms responsible for an unsatisfiable classes, and to identify root/derived unsatisfiable classes. Based on the debugging approach, in [8], 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. Although the above-mentioned work provides potential starting points for measuring incoherence, they are not explicitly used for this purpose. The approaches to measuring incoherence in ontologies are still underdeveloped.

Recently, some approaches for measuring inconsistency in an ontology have been proposed [3, 12]. In [3], some inconsistency measures are proposed by adapting the approach based on Shapley values in [7]. The approach given in [12] is defined by a four-valued semantics of DL $\mathcal{ALC}$. Our approaches differ from these approaches in that we consider measuring incoherence of an ontology instead of inconsistency, although the measures of incoherence implicitly provides information for degree of inconsistency of an ontology (recall that inconsistency and incoherence are two different but related notions in DLs).

4 Measures of Incoherence

4.1 Measures of incoherence for unsatisfiable concepts

If an ontology is incoherent, there is at least one unsatisfiable concept in its TBox. For these unsatisfiable concepts, some are more problematic than others. For example, given a TBox $\mathcal{T} = \{A\sqsubseteq B, B\sqsubseteq C, B\sqsubseteq\neg C\}$. A and B are both unsatisfiable concepts. However, A is unsatisfiable because of B. That is, if B becomes satisfiable, then A is also satisfiable. So in a sense, we may regard B as more incoherent than A. However, we may argue that B is less incoherent than A because the axioms involved in the conflict for concept B are a subset of those for concept A. We develop an alternative (conflict-centric) characterization here.

We define an ordering between two unsatisfiable concepts based on the *scoring function.*

Definition 7. *Let $\mathcal{T}$ be an incoherent TBox, and A be an unsatisfiable concept name in $\mathcal{T}$ and $MU_A(\mathcal{T})$ be the set of all MUPSs of $\mathcal{T}$ with respect to A. The scoring function for A is a function $S_{\mathcal{T},A} : \wp(\mathcal{T}) \mapsto \mathbb{N}$ ($\wp(\mathcal{T})$ denotes the power set of $\mathcal{T}$) such that for all $\mathcal{T}'\in\wp(\mathcal{T})$*

$$S_{\mathcal{T},A}(\mathcal{T}') = |\{\mathcal{T}_i\in MU_A(\mathcal{T}) : \mathcal{T}_i\cap\mathcal{T}'\neq\emptyset\}|.$$

The scoring function $S_{\mathcal{T},A}$ for A returns for each subset $\mathcal{T}'$ of $\mathcal{T}$ the number of MUPS of $\mathcal{T}$ with respect to A that have overlap with $\mathcal{T}'$. The scoring function is originally defined in [6] to compare two logical inconsistent sets of propositional formulae. It is clear that we have the following proposition.

Proposition 1. *Let $\mathcal{T}$ be an incoherent TBox, and A be an unsatisfiable concept names in $\mathcal{T}$ and $MU_A(\mathcal{T})$ be the set of all MUPSs of $\mathcal{T}$ with respect to A. Suppose $S_{\mathcal{T},A}$ is the scoring function for A, then for all $\mathcal{T}'\in\wp(\mathcal{T})$*

$$S_{\mathcal{T},A}(\mathcal{T}') = |MU_A(\mathcal{T})| - |MU_A(\mathcal{T}\setminus\mathcal{T}')|.$$

According to Proposition 1, the scoring function for $\mathcal{T}'$ gives the number of MUPS that would be eliminated from $\mathcal{T}$ if $\mathcal{T}'$ were substracted from $\mathcal{T}$.

Let $\mathcal{T}$ be an incoherent TBox, and A and B be two unsatisfiable concept names in $\mathcal{T}$. Let $M_A = \cup_{\mathcal{T}_i\in MU_A(\mathcal{T})}\mathcal{T}_i$ and $M_B = \cup_{\mathcal{T}_j\in MU_B(\mathcal{T})}\mathcal{T}_j$. Suppose $|M_A|<|M_B|$, then we add some dummy axioms to M_A such that $|M_A| = |M_B|$. The dummy axioms can be constructed by some fresh concept names. We can define a score ordering as follows:

Definition 8. *Assume that $S_{\mathcal{T},A}$ and $S_{\mathcal{T},B}$ are the scoring functions for A and B respectively. $S_{\mathcal{T},A}\leq_S S_{\mathcal{T},B}$ iff there is a bijection $f : \wp(M_A) \rightarrow \wp(M_B)$ such that the following condition is satisfied:*

$$\forall\mathcal{T}' \in \wp(M_A),\ S_{\mathcal{T},A}(\mathcal{T}')\leq S_{\mathcal{T},B}(f(\mathcal{T}')).$$

As usual, $S_{\mathcal{T},A}<_S S_{\mathcal{T},B}$ denotes $S_{\mathcal{T},A}\leq_S S_{\mathcal{T},B}$ and $S_{\mathcal{T},B}\not\leq_S S_{\mathcal{T},A}$, and $S_{\mathcal{T},A}\simeq_S S_{\mathcal{T},B}$ denotes $S_{\mathcal{T},A}\leq_S S_{\mathcal{T},B}$ and $S_{\mathcal{T},B}\leq_S S_{\mathcal{T},A}$. The score ordering, denoted $\leq$, is defined as: for any two unsatisfiable concepts A and B,

$$A\leq B\quad iff\quad S_{\mathcal{T},A}\leq_S S_{\mathcal{T},B}.$$

Intuitively, $S_{\mathcal{T},A}\leq_S S_{\mathcal{T},B}$ means that the MUPSs in $MU_A(\mathcal{T})$ are less overlapping than those in $MU_B(\mathcal{T})$. So A is less incoherent than B with respect to the score ordering iff the MUPSs in A is less overlapping than those in B.

We illustrate the score ordering by the following example.

Example 1. Given a TBox $\mathcal{T} = \{A\sqsubseteq B, A\sqsubseteq C, B\sqsubseteq D, C\sqsubseteq\neg D, E\sqsubseteq F, E\sqsubseteq\neg F, F\sqsubseteq D, E\sqsubseteq\neg D\}$, where A, B, C, D, E, F are concept names. Clearly, A and E are two root unsatisfiable concept in $\mathcal{T}$, and $MU_A = \{\mathcal{T}_1\}$, where $\mathcal{T}_1 = \{A\sqsubseteq B, A\sqsubseteq C, B\sqsubseteq D, C\sqsubseteq\neg D\}$ and $MU_E = \{\mathcal{T}_2, \mathcal{T}_3\}$, where $\mathcal{T}_2 = \{E\sqsubseteq F, E\sqsubseteq\neg F\}$ and $\mathcal{T}_3 = \{E\sqsubseteq F, F\sqsubseteq D, E\sqsubseteq\neg D\}$. So $M_A = \{A\sqsubseteq B, A\sqsubseteq C, B\sqsubseteq D, C\sqsubseteq\neg D\}$ and $M_E = \{E\sqsubseteq F, E\sqsubseteq\neg F, F\sqsubseteq D, E\sqsubseteq\neg D\}$. Let $S_{\mathcal{T},A}$ and $S_{\mathcal{T},E}$ be the scoring function for A and E respectively, then $S_{\mathcal{T},A}(\mathcal{T}') = 1$, for all $\mathcal{T}' \in \wp(M_A)$. However, $S_{\mathcal{T},E}(\{E\sqsubseteq F\}) = 2$ and $S_{\mathcal{T},E}(\mathcal{T}')\geq 1$ for all other $\mathcal{T}'\in\wp(M_E)$. So $S_{\mathcal{T},A}<_S S_{\mathcal{T},E}$ and we have $A<E$.

When defining the score ordering, we need to find a bijection f mapping every subset of M_A to a subset of M_B. In the following, we provide a procedure to find the bijection f. Let $|\wp(M_A)| = |\wp(M_B)| = n$.

Step 1: for each $\mathcal{T}_i\in\wp(M_A)$ and each $\mathcal{T}'_j\in\wp(M_B)$, compute $S_{\mathcal{T},A}(\mathcal{T}_i)$ and $S_{\mathcal{T},B}(\mathcal{T}'_j)$,
Step 2: rearrange $\mathcal{T}_1$,...,$\mathcal{T}_n$ as $\mathcal{T}_{i_1}$,...,$\mathcal{T}_{i_n}$ ($i_k\in\{1, ..., n\}$) such that $S_{\mathcal{T},A}(\mathcal{T}_{i_1})\geq S_{\mathcal{T},A}(\mathcal{T}_{i_2})\geq...\geq S_{\mathcal{T},A}(\mathcal{T}_{i_n})$, and rearrange $\mathcal{T}'_1$,...,$\mathcal{T}'_n$ as $\mathcal{T}'_{j_1}$,...,$\mathcal{T}'_{j_n}$ ($j_k\in\{1, ..., n\}$) such that $S_{\mathcal{T},B}(\mathcal{T}'_{j_1})\geq S_{\mathcal{T},B}(\mathcal{T}'_{j_2})\geq...\geq S_{\mathcal{T},B}(\mathcal{T}'_{j_n})$,

Step 3: a mapping $f_S : \wp(M_A) \rightarrow \wp(M_B)$ is defined as follows: for each $\mathcal{T}_{i_k} \in \wp(M_A)$, $f_S(\mathcal{T}_{i_k}) = \mathcal{T}'_{j_k}$.

It is clear that f_S is a bijection. The following proposition shows that f_S is the bijection which is used to define the score ordering.

Proposition 2. *Assume that $S_{\mathcal{T},A}$ and $S_{\mathcal{T},B}$ are the scoring functions for A and B respectively. Then $S_{\mathcal{T},A} \leq_S S_{\mathcal{T},B}$ iff*

$$\forall \mathcal{T}' \in \wp(M_A),\ S_{\mathcal{T},A}(\mathcal{T}') \leq S_{\mathcal{T},B}(f_S(\mathcal{T}')).$$

Proof: *"If" part is clear by the definition of score ordering. We show "only if" part.*

Suppose $S_{\mathcal{T},A} \leq_S S_{\mathcal{T},B}$, then there exists a bijection f such that for all $\mathcal{T}' \in \wp(M_A)$, $S_{\mathcal{T},A}(\mathcal{T}') \leq S_{\mathcal{T},B}(f(\mathcal{T}'))$. We shown that $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_k})$ for all $k = 1, ..., n$ by induction over the index k.

Suppose $k = 1$. Then $S_{\mathcal{T},A}(\mathcal{T}_{i_1}) \leq S_{\mathcal{T},B}(f(\mathcal{T}_{i_1})) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_1})$.

Assume that $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_k})$ for all $k < m$. Suppose that $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$. Then $S_{\mathcal{T},B}(\mathcal{T}'_{j_m}) < S_{\mathcal{T},B}(f(\mathcal{T}_{i_m}))$. This means that there exists $j_l < j_m$ such that $f(\mathcal{T}_{i_m}) = \mathcal{T}'_{j_l}$. However, since $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$, we have that $S_{\mathcal{T},A}(\mathcal{T}_{i_k}) > S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$ for all $k < m$. Therefore, for any $k < m$, there exists $k' < m$ such that $f(\mathcal{T}_{i_k}) = \mathcal{T}'_{j_{k'}}$. Therefore, it is impossible that there exists $j_l < j_m$ such that $f(\mathcal{T}_{i_m}) = \mathcal{T}'_{j_l}$ (every such $\mathcal{T}'_{j_l}$ corresponds to a $\mathcal{T}_{i_k}$ with $k < m$). This is a contradiction. So $S_{\mathcal{T},A}(\mathcal{T}_{i_m}) \leq S_{\mathcal{T},B}(\mathcal{T}'_{j_m})$.

Proposition 3. *Let $\mathcal{T}$ be an incoherent TBox, and let A and B be two unsatisfiable concepts in it. If $A \sqsubseteq B \in \mathcal{T}$, then $B \leq_S A$.*

The proof of Proposition 3 is easy to establish. Proposition 3 tells us that if A is subsumed by B then A is more incoherent than B with respect to the score ordering. If we consider the example in the beginning of this section, B is more incoherent than A with respect to the score ordering. Therefore, our score ordering provides a different view on the extent of incoherence of a concept from the subsumption relation. Indeed, scoring ordering gives a conflict-centric view since the axioms involved in the conflict for concept B are a subset of those for concept A. Proposition 3 also provides us a way to improve the performance of our approach for generating the score ordering. That is, before comparing the score functions of two unsatisfiable concepts, we can first check if they have a subsumption relation in the ontology.

4.2 Measures of incoherence for terminologies

Given a TBox which may be incoherent, we propose three approaches to measuring its degree of incoherence. The first measure is defined by the ratio of number of unsatisfiable concepts and that of all the concepts in $\mathcal{T}$.

Definition 9. *Let $\mathcal{T}$ be a TBox. Suppose $Con(\mathcal{T})$ is the set of all concept names and $US(\mathcal{T})$ be the set of all unsatisfiable concept names in $\mathcal{T}$ respectively, the unsatisfiability ratio for $\mathcal{T}$, denoted d_{UR}, is defined as follows:*

$$d_{UR}(\mathcal{T}) = \frac{|US(\mathcal{T})|}{|Con(\mathcal{T})|}.$$

The unsatisfiability ratio gives us a simple view on the incoherence of a TBox. That is, if most of concept names are unsatisfiable in a TBox, the TBox is problematic. However, the unsatisfiability ration is misleading in some cases. For example, in an ontology such as Tambis[2] where there are large number of unsatisfiable concept names, many of the unsatisfiable concept names depend on other unsatisfiable concept names. The root unsatisfiable concept names are relative few (according to [9], in Tambis, 33 concepts names out of 144 unsatisfiable concept names are root unsatisfiable concept names) and by repairing these concept names we can get a coherent ontology. Therefore, this ontology is not "strongly" incoherent. To overcome the problem for the unsatisfiability ratio, we can consider only the root unsatisfiable concept names.

Definition 10. *Let $\mathcal{T}$ be a TBox. Suppose $Con(\mathcal{T})$ is the set of all concept names and $RU(\mathcal{T})$ be the set of all root unsatisfiable concept names in $\mathcal{T}$ respectively, the refined unsatisfiability ratio for $\mathcal{T}$, denoted d_{RU}, is defined as follows:*

$$d_{RU}(\mathcal{T}) = \frac{|RU(\mathcal{T})|}{|Con(\mathcal{T})|}.$$

Let us consider Example 1 again. The set of concept names are $\{A, B, C, D, E, F\}$ and root unsatisfiable concept names are A and E, so $d_{RU}(\mathcal{T}) = \frac{2}{5}$.

Both the unsatisfiability ratio and the refined unsatisfiability ratio do not consider the number of terminology axioms that are in conflict. So we define another incoherence measure for TBoxes.

Definition 11. *Let $\mathcal{T}$ be a TBox. Suppose $MI(\mathcal{T})$ is the set of all MIPSs of $\mathcal{T}$, then the incoherence ratio for $\mathcal{T}$, denoted d_{IR}, is defined as follows:*

$$d_{IR}(\mathcal{T}) = \frac{|\cup_{\mathcal{T}_i \in MI(\mathcal{T})} \mathcal{T}_i|}{|\mathcal{T}|}.$$

The incoherence ratio measures the percentage of axioms in a TBox that are in conflict. It differentiates the root unsatisfiable concept names and derived unsatisfiable concept names. This is because any axiom whose left hand is a derived unsatisfiable concept name is not in an MIPS.

Example 2. Let $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq \neg B, C \sqsubseteq A\}$ and $\mathcal{T}' = \{A \sqsubseteq B, A \sqsubseteq \neg B, C \sqsubseteq \bot\}$. Then $US(\mathcal{T}) = US(\mathcal{T}') = \{A, C\}$ and $d_{UR}(\mathcal{T}') = d_{UR}(\mathcal{T}') = \frac{2}{3}$: T and T' have the same unsatisfiability ratio. However, $MI(\mathcal{T}) = \{\{A \sqsubseteq B, A \sqsubseteq \neg B\}\}$ and $MI(\mathcal{T}') = \{T'\}$. So $d_{IR}(\mathcal{T}) = \frac{2}{3}$ and $d_{IR}(\mathcal{T}') = 1$.

[2] http://protege.cim3.net/file/pub/ontologies/tambis/tambis-full.owl.

The problem for the incoherence ratio is that it says nothing about to which extent the MIPSs in $MI(\mathcal{T})$ overlap.

Example 3. Let $\mathcal{T} = \{A\sqsubseteq B, A\sqsubseteq\neg B, C\sqsubseteq D, C\sqsubseteq\neg D\}$ and $\mathcal{T}' = \{A\sqsubseteq B, A\sqsubseteq\neg B, B\sqsubseteq C, A\sqsubseteq\neg C\}$ be two coherent TBoxes, where A, B, C, D are concept names. By Definition 6, $\mathcal{T}$ has two MIPSs $\{A\sqsubseteq B, A\sqsubseteq\neg B\}$ and $\{C\sqsubseteq D, C\sqsubseteq\neg D\}$, and $\mathcal{T}'$ has two MIPSs $\{A\sqsubseteq B, A\sqsubseteq\neg B\}$ and $\{A\sqsubseteq B, B\sqsubseteq C, A\sqsubseteq\neg C\}$. According to Definition 11, we have $d_{IR}(\mathcal{T}) = d_{IR}(\mathcal{T}') = 1$. However, MIPSs in $MI(\mathcal{T})$ have no overlap whilst the MIPSs in $MI(\mathcal{T}')$ have a common axiom $A\sqsubseteq B$. Therefore, we may conclude that $\mathcal{T}$ is less coherent than $\mathcal{T}'$.

We have defined two measures and argue that they are not fine grained enough. Next, we define an incoherence measure for TBoxes which is based on the scoring functions.

Definition 12. *Let $\mathcal{T}$ be a TBox. The scoring function for $\mathcal{T}$ is a function $S_{\mathcal{T}} : \wp(\mathcal{T}) \mapsto N$ such that for all $\mathcal{T}'\in\wp(\mathcal{T})$*

$$S_{\mathcal{T}}(\mathcal{T}') = |\{\mathcal{T}_i\in MI(\mathcal{T}) : \mathcal{T}_i\cap\mathcal{T}'\neq\emptyset\}|.$$

The scoring function $S_{\mathcal{T}}$ for $\mathcal{T}$ returns for each subset $\mathcal{T}'$ of $\mathcal{T}$ the number of MIPSs of $\mathcal{T}$ that have an overlap with $\mathcal{T}'$.

We have the following proposition for the scoring function.

Proposition 4. *Let $S_{\mathcal{T}}$ be the scoring function for $\mathcal{T}$. For $\mathcal{T}_i, \mathcal{T}_j\in\wp(\mathcal{T})$, we have $S_{\mathcal{T}}(\mathcal{T}_i\cap\mathcal{T}_j)\leq min(S_{\mathcal{T}}(\mathcal{T}_i), S_{\mathcal{T}}(\mathcal{T}_j))$ and $max(S_{\mathcal{T}}(\mathcal{T}_i), S_{\mathcal{T}}(\mathcal{T}_j))\leq S_{\mathcal{T}}(\mathcal{T}_i\cup\mathcal{T}_j)$.*

The scoring function can be used to define an ordering between two terminology axioms.

Definition 13. *Let $\mathcal{T}$ be a TBox and $S_{\mathcal{T}}$ be its scoring function. A score-based ordering on terminology axioms in $\mathcal{T}$, denoted $\prec_{S_{\mathcal{T}}}$, is defined as follows: for any $\phi, \psi\in\mathcal{T}$,*

$$\phi\preceq_{S_{\mathcal{T}}}\psi \quad iff \quad S_{\mathcal{T}}(\{\phi\})\leq S_{\mathcal{T}}(\{\psi\}).$$

As usual, $\phi\prec_{S_{\mathcal{T}}}\psi$ denotes $\phi\preceq_{S_{\mathcal{T}}}\psi$ and $\psi\not\preceq_{S_{\mathcal{T}}}\phi$. $\phi \preceq_{S_{\mathcal{T}}} \psi$ means that ϕ is less incoherent than ψ with respect to the scoring function. That is, ϕ is contained in less MIPSs of $\mathcal{T}$ than ψ. It is clear that $\preceq_{S_{\mathcal{T}}}$ is a total pre-order, i.e. a pre-order which is complete.

Example 4. (Example 1 Continued) There are three MIPSs of $\mathcal{T}$: $\mathcal{T}_1 = \{A\sqsubseteq B, A\sqsubseteq C, B\sqsubseteq D, C\sqsubseteq\neg D\}$, $\mathcal{T}_2 = \{E\sqsubseteq F, E\sqsubseteq\neg F\}$ and $\mathcal{T}_3 = \{E\sqsubseteq F, F\sqsubseteq D, E\sqsubseteq\neg D\}$. So $S_{\mathcal{T}}(\{E\sqsubseteq F\}) = 2$ and $S_{\mathcal{T}}(\{\phi\}) = 1$ for all other $\phi \in \mathcal{T}$. Therefore, $\phi\prec_{S_{\mathcal{T}}}E\sqsubseteq F$ for all $\phi \in \mathcal{T}$ and $\phi\neq E\sqsubseteq F$.

Let $\mathcal{T}$ and $\mathcal{T}'$ be two TBox. Let $M_{\mathcal{T}} = \cup_{\mathcal{T}_i\in MI(\mathcal{T})}\mathcal{T}_i$ and $M_{\mathcal{T}'} = \cup_{\mathcal{T}_j\in MI(\mathcal{T})'}\mathcal{T}_j$. Suppose $|M_{\mathcal{T}}|<|M_{\mathcal{T}'}|$, then we add some dummy axioms to $M_{\mathcal{T}}$ such that $|M_{\mathcal{T}}| = |M_{\mathcal{T}'}|$. An ordering on TBoxes can be defined by the scoring functions as follows.

Definition 14. *Assume that $S_{\mathcal{T}}$ and $S_{\mathcal{T}'}$ are the scoring functions for two TBoxes $\mathcal{T}$ and $\mathcal{T}'$ respectively. $S_{\mathcal{T}} \preceq S_{\mathcal{T}'}$ iff there is a bijection $f : \wp(M_{\mathcal{T}}) \rightarrow \wp(M_{\mathcal{T}'})$ such that the following condition is satisfied:*

$$\forall \mathcal{T}' \in \wp(M_1),\ S_{\mathcal{T}}(\mathcal{T}') \leq S_{\mathcal{T}'}(f(\mathcal{T}')).$$

As usual, $S_{\mathcal{T}} \prec_S S_{\mathcal{T}'}$ denotes $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$ and $S_{\mathcal{T}'} \not\preceq_S S_{\mathcal{T}}$, and $S_{\mathcal{T}} \equiv_S S_{\mathcal{T}'}$ denotes $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$ and $S_{\mathcal{T}'} \preceq_S S_{\mathcal{T}}$. The score ordering, denoted $\preceq_S$, is defined as: for any two TBoxes $\mathcal{T}$ and $\mathcal{T}'$,

$$\mathcal{T} \preceq_S \mathcal{T}' \quad iff \quad S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}.$$

Intuitively, $S_{\mathcal{T}} \preceq S_{\mathcal{T}'}$ means that the MIPSs of $\mathcal{T}$ are less overlapping than those of $\mathcal{T}'$. So $\mathcal{T}$ is less incoherent than $\mathcal{T}'$ with respect to the score ordering iff the MIPSs of $\mathcal{T}$ is less overlapping than those of $\mathcal{T}'$.

Example 5. Given two TBoxes $\mathcal{T} = \{A \sqsubseteq B \sqcap C, B \sqcap C \sqsubseteq \bot\}$ and $\mathcal{T}' = \{A \sqsubseteq B, A \sqsubseteq C, B \sqcap C \sqsubseteq \bot\}$, it is clear that $\mathcal{T} \equiv \mathcal{T}'$. $\mathcal{T}$ has only one MIPS which is $\mathcal{T}$ and $\mathcal{T}'$ has only one MIPS which is $\mathcal{T}'$. So $M_{\mathcal{T}} = \mathcal{T}$ and $M_{\mathcal{T}'} = \mathcal{T}'$. Since $|M_{\mathcal{T}}| < |M_{\mathcal{T}'}|$, we add a dummy axiom to $M_{\mathcal{T}}$ such that $M_{\mathcal{T}} = \{A \sqsubseteq B \sqcap C, B \sqcap C \sqsubseteq \bot, D \sqsubseteq \top\}$, where D is a new concept name. Let $S_{\mathcal{T}}$ and $S_{\mathcal{T}'}$ be the scoring functions for $\mathcal{T}$ and $\mathcal{T}'$ respectively, we then have
$S_{\mathcal{T}}(\{A \sqsubseteq B \sqcap C\}) = 1$, $S_{\mathcal{T}}(\{B \sqcap C \sqsubseteq \bot\}) = 1$, $S_{\mathcal{T}}(\{D \sqsubseteq \top\}) = 0$, and
$S_{\mathcal{T}'}(\{A \sqsubseteq B\}) = 1$, $S_{\mathcal{T}'}(\{A \sqsubseteq C\}) = 1$, $S_{\mathcal{T}'}(\{B \sqcap C \sqsubseteq \bot\}) = 1$.
So $S_{\mathcal{T}} \prec S_{\mathcal{T}'}$ and $\mathcal{T} \prec_S \mathcal{T}'$.

According to Example 5, the scoring function defined by Definition 12 is syntax sensitive in the sense that there may exist two TBoxes $\mathcal{T}$ and $\mathcal{T}'$ where $\mathcal{T} \equiv \mathcal{T}'$ and $S_{\mathcal{T}}$ is the scoring function for $\mathcal{T}$ and $S_{\mathcal{T}'}$ is the scoring function for $\mathcal{T}'$, but $S_{\mathcal{T}} \not\equiv S_{\mathcal{T}'}$. To give a more precise measure of incoherence, we can simply split the axioms in a TBox $\mathcal{T}$ into "smaller" axioms to obtain an equivalent TBox $\mathcal{T}_s$ using the algorithm in [8]. For instance, in Example 5, the axiom $A \sqsubseteq B \sqcap C$ can be split into $A \sqsubseteq B$ and $A \sqsubseteq C$. Then it is clear that $S_{\mathcal{T}} \equiv_S S_{\mathcal{T}'}$.

The score ordering $\preceq_S$ is related to the incoherence ratio.

Proposition 5. *Let $\mathcal{T}$ and $\mathcal{T}'$ be two TBoxes, and $|\mathcal{T}| = |\mathcal{T}'|$. Suppose $S_{\mathcal{T}}$ and $S_{\mathcal{T}'}$ are scoring function for $\mathcal{T}$ and $\mathcal{T}'$ respectively, then $S_{\mathcal{T}} \preceq_S S_{\mathcal{T}'}$ implies $d_{IR}(\mathcal{T}) \leq d_{IR}(\mathcal{T}')$. The converse does not hold.*

Proposition 5 shows that if two TBoxes have the same cardinality, then a TBox $\mathcal{T}$ is less coherent than another one $\mathcal{T}'$ implies that $\mathcal{T}$ contains fewer conflicting terminology axioms.

5 Applications of Measures of Incoherence

The ordering relations on unsatisfiable concepts can provide important information for repairing incoherence in ontologies. When resolving incoherence in an ontology, we may need to repair unsatisfiable concepts one by one [8]. In this case, the ordering on unsatisfiable concepts can be used to decide which unsatisfiable concepts should be dealt with first. Another strategy to resolve incoherence is to remove some unsatisfiable concepts in an ontology [8]. Based on our ordering relation $\leq_I$, those unsatisfiable concepts that are more incoherent $w.r.t \leq_I$ should be removed before those that are less incoherent.

Based on the score ordering on the TBoxes, we can give an ordering on ontologies. That is, an ontology O is more important or reliable than another one O' if it is less coherent than O'. The ordering on ontologies provides us important information for evaluating different ontologies.

We have applied the scoring function to define an ordering between two axioms in an ontology. Alternatively, we can compute the Shapley values of each axiom and then obtain an ordering on axioms. The ordering on axioms are important context information for dealing with incoherence and inconsistency in an ontology. It has been widely accepted that priorities play an important role in dealing with inconsistency in propositional logic. Recently, several priority-based approaches for reasoning with inconsistent ontologies have been proposed [13–15]. A challenging problem for these approaches is to obtain an ordering on axioms. This problem can be solved by considering our score ordering.

6 Evaluation

We have implemented the approaches for measuring incoherence described in previous sections in JAVA using KAON2. All tests were performed on a laptop computer with a 1.5 GHz Intel processor, 512M of RAM ((with 512M heap space allocated to JVM)). The operating system was Windows XP Pro SP2. The data sets were miniTambis (we do not use Tambis because KAON2 does not support it) and a revised version of Chemical (we denoted it by Chemical*) provided by University of Maryland [3], and *proton_100_all*[4]. Some information about the ontologies is given in Table 1. The second column of the table shows the number of terminology axioms, MIPSs and the cardinality of union of all MIPSs and the third column of the table shows the number of concepts/properties/individuals and unsatisfiable concepts in the ontology. To find all the MUPSs of an unsatisfiable concept, we implemented an approach which is based on the algorithm for finding minimal unsatisfiable subsets in [2]. We then obtained all the MIPSs of an ontology using the approach given in [17].

6.1 Score ordering on unsatisfiable concepts

We implemented the approach for score ordering on unsatisfiable concepts. The performance of our approach is analyzed in Table 2. In Table 2, the second column and third column show the average runtime (in seconds) of calculating

[3] http://www.mindswap.org/2005/debugging/ontologies/
[4] http://wasp.cs.vu.nl/knowledgeweb/d2163/learning.html

Table 1. Sample OWL Data

Ontology	#axiom/#MIPS/#MIPUnion	C/P/I/U
miniTambis	173/3/12	178/35/0/30
Chemical*	123/4/8	48/19/0/35
*proton*_100_*all*	1788/3/10	266/111/30/3

MUPS of each unsatisfiable concept (prepare time) and average runtime of generating the score ordering on unsatisfiable concepts (compare time) respectively. The fourth column is the sum of the prepare time and compare time.

For all test ontologies, the time spent on comparing concepts is less than that spent on calculating MUPS. For ontology *proton*_100_*all*, which has only three unsatisfiable concepts and the maximal cardinality of their MUPSs is 5, it takes 0.071 seconds to compare concepts. Much time has been spent on calculating MUPS because there exist 1788 axioms in this ontology. In contrast, for ontologies miniTambis and Chemical* which contains about 30 unsatisfiable concepts, there is increasing time spent on comparison. Especially, for Chemical* in which the cardinalities of some MUPSs are more than 10, the compare time is almost the same as the prepare time. Therefore, we can conclude that the efficiency of computation of score ordering on unsatisfiable concepts depends on the cardinalities of MUPSs. Even for a big ontology like *proton*_100_*all*, if the cardinalities of the MUPSs of its unsatisfiable concepts are small, the score ordering can still be computed quickly if we ignore the time spent on calculating MUPS.

Table 2. Score ordering on unsatisfiable concepts (time is in seconds)

Ontology	Prepare time	Compare time	Total time
miniTambis	233.230	78.540	311.770
Chemical*	873.12	652.75	1525.87
*proton*_100_*all*	319.83	0.071	319.901

6.2 Score ordering on terminologies

We also implemented the approaches for score orderings on terminology axioms and on ontologies. The performance of the approach for score ordering on terminology axioms is analyzed in Table 3. In Table 3, the second column and third column show the average runtime (in milliseconds) of calculating MIPS of the ontology (prepare time) and average runtime of generating the score ordering on axioms (compare time) respectively. The prepare time in Table 3 does not count the time spent on calculating all MUPSs in an ontology, which can be checked in Table 2. According to the table, it takes less than one second to calculate all the MIPSs from MUPSs for all test ontologies.

Table 3. Score ordering on terminologies (time is in milliseconds)

Ontology	Prepare time	Compare time	Total time
miniTambis	171	10	181
Chemical*	511	30	541
proton_100_all	90	0.1	90.1

According to our experiment, the score of every terminology axiom in miniTambis is 1. This means that the MIPSs in miniTambis are pair-wise disjoint. For Chemical* and *proton_100_all*, however, axioms may get different scores. For example, in Chemical*, the axiom $PublishedWork \sqsubseteq \neg Person$ gets score 2 and the axiom $NerveAgentRelatedPublishedWork \sqsubseteq PublishedWork$ gets score 4 and so the former is less incoherent than the latter *w.r.t.* the score ordering. According to Table 3, the time spent on comparison is much less than that spent on calculating MIPSs.

Similar to the score ordering on terminology axioms, if we ignore the time spent on calculating MIPS, it also takes less than 1 second to obtain the score orderings on ontologies. This is reasonable because both the number of MIPS in each ontology and the size of the union of MIPS in each ontology are small. In contrast, the MUPS of an unsatisfiable concept is more likely to contain a large number of axioms and there are usually many unsatisfiable concepts in a coherent ontology. So it takes longer to obtain a score ordering on unsatisfiable concepts. The test ontologies are not comparable *w.r.t.* the score ordering $\preceq_S$. That is, there does not exist two ontologies such that one is more incoherent than the other *w.r.t.* the score ordering.

7 Conclusions and Future Work

In this paper, we have defined two classes of measures for incoherent ontologies: measures of incoherence for unsatisfiable concepts and measures of incoherence for terminologies. The first class of measures gives us information on comparing unsatisfiable concept names. The second class of measures gives us information on comparing terminology axioms and comparing ontologies. These measures of incoherence can provide important information for dealing with incoherence and evaluating ontologies. Finally, we have implemented the approaches for measuring incoherence and reported on some preliminary but interesting experimental results.

For future work, we plan to apply the approaches of measuring incoherence to deal with incoherence and inconsistency in an ontology. The score ordering defined on unsatisfiable concepts is not applicable for ontologies which contain a large number of conflicting terminology axioms. We will explore some approximation techniques to solve this problem.

8 Acknowledgments

This work is partially supported by the EU under the IST project NeOn (IST-2006-027595, http://www.neon-project.org/).

References

1. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, implementation and application*. Cambridge University Press, 2003.
2. Maria Garcia de la Banda, Peter J. Stuckey, and Jeremy Wazny. Finding all minimal unsatisfiable subsets. In *Proceedings of the 5th ACM SIGPLAN international conference on Principles and practice of declaritive programming*, pages 32–43, 2003.
3. Xi Deng, Volker Haarslev, and Nematollaah Shiri. Measuring inconsistency in ontologies. In *Proc. of ESWC'07*, 2007, to appear.
4. Giorgos Flouris, Zhisheng Huang, Jeff Z. Pan, Dimitris Plexousakis, and Holger Wache. Inconsistencies, negations and changes in ontologies. In *Proc. of AAAI'06*, pages 1295–1300, 2006.
5. Anthony Hunter. Measuring inconsistency in knowledge via quasi-classical models. In *Proc. of AAAI'02*, pages 68–73, 2002.
6. Anthony Hunter. Logical comparison of inconsistent perspectives using scoring functions. *Knowledge and Information Systems*, 6(5):528–543, 2004.
7. Anthony Hunter and Sébastien Konieczny. Shapley inconsistency values. In *Proc. of KR'06*, pages 249–259, 2006.
8. Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and Bernardo Cuenca Grau. Repairing unsatisfiable concepts in owl ontologies. In *Proc. of ESWC'06*, pages 170–184, 2006.
9. Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler. Debugging unsatisfiable classes in owl ontologies. *Journal of Web Semantics*, 3(4):268–293, 2005.
10. Kevin M. Knight. Measuring inconsistency. *Journal of Philosophical Logic*, 31:77–98, 2001.
11. Sébastien Konieczny, Jérôme Lang, and Pierre Marquis. Quantifying information and contradiction in propositional logic through epistemic tests. In *Proc. of IJCAI'03*, pages 106–111, 2003.
12. Yue Ma, Guilin Qi, Pascal Hitzler, and Zuoquan Lin. Measuring inconsistency for description logics based on paraconsistent semantics. In *Proc. of DL'07*, 2007, to appear.
13. 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.
14. Guilin Qi, Weiru Liu, and David A. Bell. A revision-based algorithm for handling inconsistency in description logics. In *Proc. of NMR'06*, pages 124–132, 2006.
15. Guilin Qi, Jeff Z. Pan, and Qiu Ji. A possibilistic extension of description logics. In *Proc. of DL'07*, 2007, to appear.
16. Stefan Schlobach. Diagnosing terminologies. In *Proc. of AAAI'05*, pages 670–675. AAAI Press, 2005.
17. Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proc. of IJCAI'03*, pages 355–362. 2003.