A Kernel Revision Operator for Terminologies

Guilin Qi[1], Peter Haase[1], Zhisheng Huang[2], and Jeff Z. Pan[3]

[1] Institute AIFB, University of Karlsruhe, Germany
{gqi,pha}@aifb.uni-karlsruhe.de
[2] Department of Mathematics and Computer Science, Vrije University Amsterdam
huang@cs.vu.nl
[3] Department of Computing Science, The University of Aberdeen
jpan@csd.abdn.ac.uk

Abstract. In this paper, we propose a new method for revising terminologies in description logic-based ontologies. Our revision method is a reformulation of the kernel revision operator in belief revision. We first define our revision operator for terminologies in terms of MIPS (minimal incoherence-preserving sub-terminologies), and we show that it satisfies some desirable logical properties. Second, two concrete algorithms are developed to implement the revision operator.

Key words: Description Logic, Incoherence, Revision

1 Introduction

Ontologies are typically not static entities, but they evolve over time. Firstly, the process of creating ontologies is in fact a process of change. Secondly, even when an ontology is stable enough, changes are often necessary when the scenario of its application changes. Furthermore, from time to time people might want to change an ontology to reuse it in a similar environment. Ontology evolution provides means for the timely adaptation of an ontology to such changes.

An important problem in ontology evolution is the consistent revision of the ontology, i.e. the accommodation of new knowledge in an ontology without introducing logical contradictions. Generally, we can distinguish two kinds of logical contradictions: inconsistency and incoherence. An ontology is inconsistent iff it has no model. An ontology is incoherent iff there exists some unsatisfiable concept (i.e, the semantics of an unsatisfiable concept corresponds to the empty set).

There exists a number of prior work on revision in DLs, such as those reported in [3, 5, 11, 2]. Most of them focus on postulates for revision operators. For example, an important principle is that one should delete information in the original ontology as little as possible to accommodate the new knowledge consistently. Theoretically, it is very important to know how to characterize a revision operator in terms of postulates. However, for practical applicability to real life ontologies, we require concrete revision operators that can be used. There are some concrete revision operators defined to deal with inconsistency

[5, 11]. However, to the best of our knowledge, there is no revision operator dealing specifically with incoherence (as opposed to inconsistency) in the context of revision.

In this paper, we propose a kernel revision operator in Description Logic-based ontologies based on MIPS (minimal incoherence-preserving sub-terminologies) and an incision function. The notion of MIPS is originally developed for non-standard reasoning service in debugging incoherent terminologies [14, 15]. It is similar to the notion of a *kernel set* in belief base change defined in [8]. The incision function is used to select axioms from each MIPS to remove from the original ontology. Our revision operator focuses on revising terminologies, i.e. ontologies with empty ABox. Two concrete algorithms are developed to define specific kernel revision operator. The first algorithm is based on Reiter's hitting set tree (HS-tree) algorithm [12] and MIPS. In this algorithm, we need to compute all the MIPS, which is computationally hard in general. Therefore, we propose another algorithm, which utilizes confidence values attached to axioms in the ontology to resolve unsatisfiability concepts without computing all the MIPS. Compared with the first algorithm, this algorithm is computationally easier but may delete more axioms from the original ontology after revision.

The rest of this paper is organized as follows: Section 2 provides a preliminary introduction to Description logics and various notions in ontolgy debugging. Section 3 presents a syntax-based method for revising terminologies by using the notions of MUPS and MIPS. Section 4 proposes some algorithms of the ontology revision. Section 5 overviews related work of ontology evolution and ontology revision, and Section 6 discusses future work before concluding the paper.

2 Preliminaries

2.1 Description Logics

This section introduces some basic notions of Description Logics (DLs) and some important notions used for debugging terminologies.

In our work, we focus DL-based terminological ontologies: An ontology $\mathcal{T}$ consists of concept axioms and role axioms (TBox). Concept axioms have the form $C \sqsubseteq D$ where C and D are (possibly complex) concept descriptions[1], and role axioms are expressions of the form $R \sqsubseteq S$, where R and S are role descriptions. We call both concept axioms and role axioms as terminology axioms.

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.

[1] A complex concept is a concept that is formed by some atomic concepts and constructors such as conjunction $\sqcap$ and disjunction $\sqcup$.

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., $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$). An interpretation $\mathcal{I}$ is called a *model* of a TBox $\mathcal{T}$, iff it satisfies each axiom in $\mathcal{T}$. We use $Mod(\mathcal{T})$ to denote all the models of TBox $\mathcal{T}$ A named concept C in a terminology $\mathcal{T}$, is unsatisfiable iff, for each model $\mathcal{I}$ of $\mathcal{T}$, $C^{\mathcal{I}} = \emptyset$. A terminology $\mathcal{T}$ is incoherent iff there exists an unsatisfiable named concept in $\mathcal{T}$. Two TBoxes $\mathcal{T}$ and $\mathcal{T}'$ are equivalent, denoted by $\mathcal{T} \equiv \mathcal{T}'$, iff $Mod(\mathcal{T}) = Mod(\mathcal{T}')$.

2.2 MUPS and MIPS

We introduce the notion of MIPS and MUPS which will be used to define our revision operator. They are originally defined in [14], and are used to pinpoint errors in an ontology.

Definition 1. *Let A be a named concept 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 $\mathcal{T}$ *w.r.t* A is the minimal sub-TBox of $\mathcal{T}$ in which A is unsatisfiable. We will abbreviate the set of MUPS of $\mathcal{T}$ *w.r.t* a concept name A by $mups(\mathcal{T}, A)$. Let us consider an example from [14]. Suppose $\mathcal{T}$ contains the following axioms:

$$ax_1 : A_1 \sqsubseteq \neg A \sqcap A_2 \sqcap A_3 \qquad ax_2 : A_2 \sqsubseteq A \sqcap A_4$$
$$ax_3 : A_3 \sqsubseteq A_4 \sqcap A_5 \qquad ax_4 : A_4 \sqsubseteq \forall s.B \sqcap C$$
$$ax_5 : A_5 \sqsubseteq \exists s.\neg B \qquad ax_6 : A_6 \sqsubseteq A_1 \sqcup \exists r.(A_3 \sqcap \neg C \sqcap A_4)$$
$$ax_7 : A_7 \sqsubseteq A_4 \sqcap \exists s.\neg B$$

where A, B and C are atomic concept names and A_i $(i = 1, ..., 7)$ are defined concept names, and r and s are atomic roles. In this example, the unsatisfiable concept names are A_1, A_3, A_6, A_7 and MUPS of $\mathcal{T}$ w.r.t A_i $(i = 1, 3, 6, 7)$ are:

$mups(\mathcal{T}, A_1) : \{\{ax_1, ax_2\}, \{ax_1, ax_3, ax_4, ax_5\}\}$
$mups(\mathcal{T}, A_3) : \{ax_3, ax_4, ax_5\}$
$mups(\mathcal{T}, A_6) : \{\{ax_1, ax_2, ax_4, ax_6\}, \{ax_1, ax_3, ax_4, ax_5, ax_6\}\}$
$mups(\mathcal{T}, A_7) : \{ax_4, ax_7\}$

MUPS are useful for relating sets of axioms to the unsatisfiability of specific concepts, but they can also be used to calculate a minimal incoherence preserving sub-TBox, which relates sets of axioms to the incoherence of a TBox in general and is defined as follows.

Definition 2. *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 a minimal sub-TBox of $\mathcal{T}$ which is incoherent. The set of MIPS for a TBox $\mathcal{T}$ is abbreviated with $mips(\mathcal{T})$. For $\mathcal{T}$ in the above example, we get 3 MIPS:

$mips(\mathcal{T}) = \{\{ax_1, ax_2\}, \{ax_3, ax_4, ax_5\}, \{ax_4, ax_7\}\}$

3 Kernel Revision Operator for Terminologies

In this section, we define our revision operator based on the notion of MIPS. Originally, the notion of a MIPS is defined on a single TBox, whereas a revision operator deals with conflict between two TBoxes. We therefore generalize MIPS by considering two TBoxes: the TBox $\mathcal{T}$ to be revised, and the newly received TBox $\mathcal{T}_0$. We further assume that both $\mathcal{T}$ and $\mathcal{T}_0$ are coherent in the following.

Definition 3. *Let $\mathcal{T}$ and $\mathcal{T}_0$ be two TBoxes. A minimal incoherence-preserving sub-TBox (MIPS) $\mathcal{T}'$ of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ is a sub-TBox of $\mathcal{T}$ which satisfies (1) $\mathcal{T}' \cup \mathcal{T}_0$ is incoherent; (2) $\forall \mathcal{T}'' \subset \mathcal{T}'$, $\mathcal{T}'' \cup \mathcal{T}_0$ is coherent. We denote the set of all MIPSs of $\mathcal{T}$ w.r.t $\mathcal{T}_0$ by $MIPS_{\mathcal{T}_0}(\mathcal{T})$.*

A MIPS of TBox $\mathcal{T}$ *w.r.t.* TBox $\mathcal{T}_0$ is the minimal sub-TBox of $\mathcal{T}$ that is incoherent with $\mathcal{T}_0$. It is similar to the *kernel* defined by Hansson in [8]. Similar to Definition 3, we can define a MUPS of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ and an unsatisfiable concept of $\mathcal{T} \cup \mathcal{T}_0$. When $\mathcal{T}_0$ is an empty set, then Definition 3 is reduced to Definition 2. In classical logic, given a knowledge base A which is a set of classical formulas and a formula ϕ, a ϕ-kernel of A is the minimal subbase of A that implies ϕ. To define a contraction function for removing knowledge from a knowledge base, called kernel contraction, Hansson defines an incision function which selects formulas to be discarded in each ϕ-kernel of A. We adapt the incision function to define our revision operator.

Definition 4. *Let $\mathcal{T}$ be a TBox. An* incision function *for $\mathcal{T}$, denoted as σ, is a function such that for each TBox $\mathcal{T}_0$*

(i) $\sigma(MIPS_{\mathcal{T}_0}(\mathcal{T})) \subseteq \bigcup_{\mathcal{T}_i \in MIPS_{\mathcal{T}_0}(\mathcal{T})} \mathcal{T}_i$;
(ii) if $\mathcal{T}' \in MIPS_{\mathcal{T}_0}(\mathcal{T})$, *then* $\mathcal{T}' \cap \sigma(MIPS_{\mathcal{T}_0}(\mathcal{T})) \neq \emptyset$.

An incision function for a TBox $\mathcal{T}$ is a function such that for each TBox $\mathcal{T}_0$, it selects formulas from every MIPS of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ if this MIPS is not empty. Condition (i) says the axioms selected by an incision function must belong to some MIPSs of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$. Condition (ii) says each MIPS of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ must have at least one axiom selected. The incision function plays a similar role as concept pinpointing in [14]. However, the latter is only applied to a single ontology.

An important incision function is the one which is called *minimal incision function* [1]. The idea of this incision function is to select a minimal subset of elements from the set of kernel sets. We adapt this incision function as follows.

Definition 5. *Let $\mathcal{T}$ be a TBox. An incision function σ for $\mathcal{T}$ is* minimal *if there is no other incision function σ' such that there is a TBox $\mathcal{T}_0$, $\sigma'(MIPS_{\mathcal{T}_0}(\mathcal{T})) \subset \sigma(MIPS_{\mathcal{T}_0}(\mathcal{T}))$.*

A minimal incision function selects a minimal subset of $\mathcal{T}$ w.r.t. the set inclusion. However, among all the minimal incision functions, some of them select more axioms than others. To make the number of selected axioms minimal, we define a cardinality-minimal incision function.

Definition 6. *Let $\mathcal{T}$ be a TBox. An incision function σ for $\mathcal{T}$ is* cardinality-minimal *if there is no other incision function σ' such that there is a TBox $\mathcal{T}_0$, $|\sigma'(MIPS_{\mathcal{T}_0}(\mathcal{T}))| < |\sigma(MIPS_{\mathcal{T}_0}(\mathcal{T}))|$.*

It is clear that a cardinality-minimal incision function is always a minimal incision function.

Proposition 1. *Let $\mathcal{T}$ be a TBox. Suppose σ is a cardinality-minimal incision function for $\mathcal{T}$, then it is a minimal incision function.*

From each incision function, we can define an operator for revising a TBox $\mathcal{T}$ by a newly received TBox $\mathcal{T}_0$. The idea is that we first calculate the MIPS of TBox $\mathcal{T}$ *w.r.t* TBox $\mathcal{T}_0$, then delete axioms in $\mathcal{T}$ selected by the incision function. After that, we take the union of the modified TBox and $\mathcal{T}_0$ as the result of revision.

Definition 7. *Let $\mathcal{T}$ be a TBox, and σ be an incision function for $\mathcal{T}$. The* kernel revision *operator $\circ_\sigma$ for $\mathcal{T}$ is defined as follows: for each TBox $\mathcal{T}_0$,*

$$\mathcal{T} \circ_\sigma \mathcal{T}_0 = (\mathcal{T} \setminus \sigma(MIPS_{\mathcal{T}_0}(\mathcal{T}))) \cup \mathcal{T}_0.$$

The resulting TBox of the kernel revision operator only contains one TBox. According to the definition of an incision function, the resulting TBox of the kernel revision operator is always a unique coherent TBox.

Proposition 2. *Let $\mathcal{T}$ be a TBox, and σ be an incision function for $\mathcal{T}$. The operator $\circ_\sigma$ satisfies the following properties: for any TBoxes $\mathcal{T}_0$, $\mathcal{T}_0'$*
(R1) $\mathcal{T}_0 \subseteq \mathcal{T} \circ_\sigma \mathcal{T}_0$.
(R2) If $\mathcal{T} \cup \mathcal{T}_0$ is coherent, then $\mathcal{T} \circ_\sigma \mathcal{T}_0 = \mathcal{T} \cup \mathcal{T}_0$.
(R3) If $\mathcal{T}_0$ is coherent, then $\mathcal{T} \circ_\sigma \mathcal{T}_0$ is coherent.
(R4) If $\mathcal{T}_0 \equiv \mathcal{T}_0'$, then $\mathcal{T} \circ_\sigma \mathcal{T}_0 \equiv \mathcal{T} \circ_\sigma \mathcal{T}_0'$.
(R5) If $\phi \in \mathcal{T}$ and $\phi \notin \mathcal{T} \circ_\sigma \mathcal{T}_0$, then there is a subset S of $\mathcal{T}$ and a subset S_0 of $\mathcal{T}_0$ such that $S \cup S_0$ is coherent, but $S \cup S_0 \cup \{\phi\}$ is not.

Properties (R1)-(R4) are adapted from postulates (O+1), (O+2*), (O+3*) and (O+4) in [2]. (R1) says every axiom in the new TBox should be accepted after revision. (R2) says if two TBoxes have no contradiction, then we do not need to change anything. (R3) means that if the new TBox is coherent, then the result of revision should also be coherent. (R4) is a weakened form of syntax-independence. That is, the revision operator is independent of the syntactic form of axioms in the new TBox. (R5) is a new property which is adapted from the core-retainment postulate in [8]. It states that if an axiom is deleted after revision, then it must be responsible for the conflict.

4 Algorithms

The kernel revision operator is defined by an incision function. However, we have not given any incision function. Inspired by the work reported in [16], in

Algorithm 1: Algorithm for Repair based on scoring function

Data: Two TBoxes $\mathcal{T}$ and $\mathcal{T}_0$, where $\mathcal{T}$ is the TBox to be revised
Result: A repaired coherent TBox $\mathcal{T} \circ_\sigma \mathcal{T}_0$
begin
  $\mathcal{C} = \emptyset$
  calculate $MIPS_{\mathcal{T}_0}(\mathcal{T})$
  for $ax \in \bigcup_{\mathcal{T}_i \in MIPS_{\mathcal{T}_0}(\mathcal{T})} \mathcal{T}_i$ **do**
    $w_{ax} := S_{MIPS_{\mathcal{T}_0}(\mathcal{T})}(\{ax\})$
  for $\mathcal{T}_i \in MIPS_{\mathcal{T}_0}(\mathcal{T})$ **do**
    $A_i := \{ax \in \mathcal{T}_i : \nexists ax' \in \mathcal{T}_i, w_{ax'} > w_{ax}\}$
    $\mathcal{C} := \mathcal{C} \cup \{A_i\}$
  $\sigma(MIPS_{\mathcal{T}_0}(\mathcal{T})) := HSTree(\mathcal{C})$
  $\mathcal{T} \circ_\sigma \mathcal{T}_0 := \mathcal{T} \setminus \sigma(MIPS_{\mathcal{T}_0}(\mathcal{T})) \cup \mathcal{T}_0$
  return $\mathcal{T} \circ_\sigma \mathcal{T}_0$
end

the following, we propose some algorithms for computing an incision function based on Reiter's hitting set tree (HST) algorithm [12] which is reformulated in [15]. We select one arbitrary minimal hitting set of $MIPS_{\mathcal{T}_0}(\mathcal{T})$ given by HST algorithm in [15]. We denote the revised HST algorithm as $HSTree$.

Next, we give two algorithms to repair a TBox. The first one is based on the *scoring function on axioms*[2] which is defined as follows.

Definition 8. *Let $\mathcal{T}$ be a TBox and $\mathcal{M}$ be a set of sub-TBoxes of $\mathcal{T}$. Let $\mathcal{P}(\mathcal{T})$ be the power set of $\mathcal{T}$. The scoring function for $\mathcal{T}$ w.r.t. $\mathcal{M}$, is a function $S_{\mathcal{M}} : \mathcal{P}(\mathcal{T}) \mapsto N$ such that for all $\mathcal{T}' \in \mathcal{P}(\mathcal{T})$*

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

The scoring function $S_{\mathcal{M}}$ for $\mathcal{T}$ returns for each subset $\mathcal{T}'$ of $\mathcal{T}$ the number of elements of $\mathcal{M}$ that have an overlap with $\mathcal{T}'$. If we apply the scoring function to each singleton $\{ax_i\}$, where ax_i is an axiom in $\mathcal{T}$, then we can attach each axiom in $\mathcal{T}$ a degree.

In Algorithm 1, we first calculate all the MIPSs of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ (MIPSs for short). The approach for calculating all the MIPSs is based on a black-box algorithm for finding all justifications in [9]. We then compute the score of each axiom which is in the union of the MIPSs (see the first "for" loop). For each MIPS, we select a subset of it which contains those axioms whose scores are maximal among all the axioms in the MIPS and apply the modified HST algorithm to them (see the second "for" loop and the line after it). The result of the modified HST algorithm is the set of axioms to be deleted, i.e., $\sigma(MIPS_{\mathcal{T}_0}(\mathcal{T}))$. After removing these axioms, we restore coherence of the TBox $\mathcal{T}$ w.r.t. $\mathcal{T}_0$. In

[2] Scoring function has been used in [10] to measuring inconsistency in a single ontology and is defined by MIPS. Our definition is slightly different from theirs in that ours is not defined by MIPS.

our algorithm, we use subsets of MIPSs consisting of axioms with highest scores as an input to the HST algorithm, instead of using all the MIPSs, therefore, the number of axioms removed may not be minimal. This is due to efficiency consideration because there may have a large number of hitting sets if we use all the MIPSs and the algorithm will be very slow.

Example 1. Suppose that we have two TBoxes :

$$\mathcal{T} = \{Example \sqsubseteq Knowledge, Document \sqsubseteq \neg Knowledge, Form \sqsubseteq Knowledge, \\ Firm \sqsubseteq Organisation\}$$

$$\mathcal{T}_0 = \{Document \sqsubseteq Example, Knowhow\_document \sqsubseteq Document, \\ Form \sqsubseteq Document, KnowledgeManagement\_service \sqsubseteq Service\}.$$

The TBoxes are taken from a real life ontology $km1500\_i500$[3] which is obtained by ontology learning. The MIPSs of $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ are

$\mathcal{T}' = \{Example \sqsubseteq Knowledge, Document \sqsubseteq \neg Knowledge\}$

and

$\mathcal{T}'' = \{Document \sqsubseteq \neg Knowledge, Form \sqsubseteq Knowledge\}$.

The score of the disjointness axiom $Document \sqsubseteq \neg Knowledge$ is 2 because it belongs to both MIPSs. The scores of other axioms are 1. Therefore, we delete $Document \sqsubseteq \neg Knowledge$ and the result of revision is

$$\mathcal{T} \circ_s \mathcal{T}_0 = \{Example \sqsubseteq Knowledge,\ Form \sqsubseteq Knowledge,\ Firm \sqsubseteq Organisation, \\ Document \sqsubseteq Example, Knowhow\_document \sqsubseteq Document, \\ Form \sqsubseteq Document, KnowledgeManagement\_service \sqsubseteq Service\}.$$

In some cases, there are confidence values attached to axioms in an ontology. These confidence values can be generated during ontology learning process (see [7]) or are given by human experts (They are not generated by an automated algorithm.). When axioms in the TBox are attached to confidence values, we do not need to calculate all the MIPSs before we repair the TBox. We have the following algorithm which utilizes confidence values of axioms to decide which axioms should be deleted. We use w_{ax} to denote the confidence value of the axiom ax.

In Algorithm 2, when resolving incoherence of a TBox, we do not compute all the MUPSs and MIPSs. Instead, we resolve incoherence by iteratively dealing with unsatisfiable concepts. That is, we remove axioms in MUPSs of an unsatisfiable concept and make it satisfiable and then go to deal with another unsatisfiable concept, and so on. The algorithm which computes all the MUPSs of C in $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ is similar to the algorithm to compute MUPS in [9]. For each

[3] The ontology is available from http://wasp.cs.vu.nl/knowledgeweb/d2163/learning.html.

Algorithm 2: Algorithm for Repair based on confidence values

Data: Two TBoxes $\mathcal{T}$ and $\mathcal{T}_0$, where $\mathcal{T}$ is the TBox to be revised, axioms in $\mathcal{T}$ are attached with confidence values
Result: A repaired coherent TBox $\mathcal{T} \circ_w \mathcal{T}_0$
begin
  $\mathcal{C} := \emptyset$
  for $C \in GETALLCONCEPTS(\mathcal{T} \cup \mathcal{T}_0)$ **do**
    while $\mathcal{T} \cup \mathcal{T}_0 \models C \sqsubseteq \bot$ **do**
      $\mathcal{M}_{C,\mathcal{T},\mathcal{T}_0} := GETMUPS_{\mathcal{T}_0}(C, \mathcal{T})$
      for $\mathcal{T}_i \in \mathcal{M}_{C,\mathcal{T},\mathcal{T}_0}$ **do**
        $\underline{\mathcal{T}_i} := \{ax \in \mathcal{T}_i : \nexists ax' \in \mathcal{T}_i, w_{ax'} < w_{ax}\}$
      $\mathcal{C} := \{\underline{\mathcal{T}_i} : \mathcal{T}_i \in \mathcal{M}_{C,\mathcal{T},\mathcal{T}_0}\}$
      $\mathcal{T}_C := HSTree(\mathcal{C})$
      $\mathcal{T} \circ_w \mathcal{T}_0 := (\mathcal{T} \setminus \mathcal{T}_C) \cup \mathcal{T}_0$
      $\mathcal{C} := \emptyset$
  return $\mathcal{T} \circ_w \mathcal{T}_0$
end

unsatisfiable concept, we take the subset of every MUPS which contain axioms with minimal confidence values and apply the HS-Tree algorithm to select those axioms to be deleted. The revision operator implemented by this algorithm may not be a kernel revision operator because we do not calculate all the MIPSs. However, this algorithm still achieve minimal change when resolve unsatisfiability of a concept and it is more efficient than Algorithm 1.

Example 2. Let us consider Example 1 again. Suppose axioms in the TBox $\mathcal{T}$ are attached with confidence values as follows:
$w_{Example \sqsubseteq Knowledge} = 0.4$,
$w_{Document \sqsubseteq \neg Knowledge} = 0.8$,
$w_{Form \sqsubseteq Knowledge} = 0.6$,
$w_{Firm \sqsubseteq Organisation} = 0.9$.
The axioms in $\mathcal{T}_0$ are assigned weight 1, i.e., they are firmly believed. There are two unsatisfiable concepts in $\mathcal{T} \cup \mathcal{T}_0$: $Document$ and $Form$. Suppose our algorithm chooses $Form$ first. The MUPSs of $Form$ in $\mathcal{T}$ w.r.t. $\mathcal{T}_0$ are
$\mathcal{T}' = \{Document \sqsubseteq \neg Knowledge, Form \sqsubseteq Knowledge\}$ and
$\mathcal{T}'' = \{Example \sqsubseteq Knowledge, Document \sqsubseteq \neg Knowledge\}$.
So $\mathcal{M}_{Form,\mathcal{T},\mathcal{T}_0} = \{\mathcal{T}', \mathcal{T}''\}$. Since $w_{Form \sqsubseteq Knowledge} < w_{Document \sqsubseteq \neg Knowledge}$ and $w_{Example \sqsubseteq Knowledge} < w_{Document \sqsubseteq \neg Knowledge}$, we have
$\mathcal{C} = \{\{Form \sqsubseteq Knowledge\}, \{Example \sqsubseteq Knowledge\}\}$.
So $\mathcal{T}_{\mathcal{C}} = \{Form \sqsubseteq Knowledge, Example \sqsubseteq Knowledge\}$.
Let $\mathcal{T} = \mathcal{T} \setminus \{Form \sqsubseteq Knowledge, Example \sqsubseteq Knowledge\}$.
It is easy to check that $\mathcal{T} \cup \mathcal{T}_0$ is coherent now. So the algorithm terminates and the result of revision is

$\mathcal{T} \circ_w \mathcal{T}_0 = \{Document \sqsubseteq \neg Knowledge, Knowhow\_document \sqsubseteq Document,$

$Document \sqsubseteq Example, KnowledgeManagement\_service \sqsubseteq Service,$
$Firm \sqsubseteq Organisation, Form \sqsubseteq Document\}$.

5 Related Work

The problem of revision in DLs has been extensively studied in literature. From a theoretical point of view, the AGM framework is the most influencing work on belief change [4], which captures appropriate conditions for the intuitive rationality of belief change operators. In [3], Flouris, Plexousakis and Antoniou generalize the AGM framework in order to apply the rationalities behind the AGM framework to a wider class of logics, i.e. a larger class of logics which are AGM-compliant. In [2] a framework for the distinction between incoherence and inconsistency of an ontology is proposed. A set of rational postulates for a revision operator in DLs is proposed based on the distinction between the coherent negation and consistent negation. However, in [2], no concrete revision operator is proposed. In [11], reformulated AGM postulates for revision are adapted to DLs. The authors also propose two revision operators that satisfy the adapted postulates, but no algorithm to implement any of the operator is introduced.

Similar to our revision operator, the revision operator defined in [13] also utilizes an *incision function* to select axioms in the original ontology to delete. Our work differs from theirs in serval aspects. First, our revision operator deals with incoherence instead of inconsistency. Second, the incision function used in [13] is different from ours. Third, we provide algorithms for computation of specific revision operators and discuss evaluation results on their implementation. This work is also related to the work presented in [6], in which an algorithm is given to determine consistent sub-ontologies by adding an axiom to an ontology. The algorithm is based on on a selection function by assuming that all axioms in the ontology are connected.

6 Conclusion and Future Work

In this paper, we have proposed a kernel revision operator for terminologies based on MIPS and an incision function, which has desirable properties such as those listed in propositions 1 and 2. By implementing an incision function, we can obtain a concrete revision operator. We provided two algorithms to instantiate our revision operator. One future work is to implement the algorithms and report evaluation results. Another future work is to give other interesting incision function to define new kernel revision operators.

Acknowledgments

The first and second authors are partially supported by the EU in the IST project NeOn (EU IST-2006-027595, `http://www.neon-project.org/`). The third author is partially supported by the EU in the IST project OpenKnowledge (EU IST-2006-027253, `http://www.openk.org/`).

References

1. Marcelo A. Falappa, Eduardo L. Fermé, and Gabriele Kern-Isberner. On the logic of theory change: Relations between incision and selection functions. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI'06)*, pages 402–406, Riva del Garda, Italy, Aug 2006.
2. 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.
3. Giorgos Flouris, Dimitris Plexousakis, and Grigoris Antoniou. On applying the AGM theory to DLs and OWL. In *Proceedings of the 4th International Conference on Semantic Web (ISWC'05)*, pages 216–231. Galway, Ireland, Nov 2005.
4. Peter Gardenfors. *Knowledge in Flux-Modeling the Dynamic of Epistemic States*. The MIT Press, Cambridge, Mass, 1988.
5. Peter Haase and Ljiljana Stojanovic. Consistent evolution of OWL ontologies. In *Proc. of ESWC'05*, pages 182–197, 2005.
6. Peter Haase, Frank van Harmelen, Zhisheng Huang, Heiner Stuckenschmidt, and York Sure:. A framework for handling inconsistency in changing ontologies. In *Proceedings of 4th International Semantic Web Conference (ISWC'05)*, pages 353–367. Springer, 2005.
7. Peter Haase and Johanna Völker. Ontology learning and reasoning - dealing with uncertainty and inconsistency. In Paulo C. G. da Costa, Kathryn B. Laskey, Kenneth J. Laskey, and Michael Pool, editors, *Proceedings of the Workshop on Uncertainty Reasoning for the Semantic Web (URSW'05)*, pages 45–55, Galway, Ireland, Nov 2005.
8. Sven Ove Hansson. Kernel contraction. *Journal of Symbolic Logic*, 59(3):845–859, 1994.
9. Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of OWL DL entailments. In *Proc. of ISWC/ASWC'07*, pages 267–280, 2007.
10. Guilin Qi and Anthony Hunter. Measuring incoherence in description logic-based ontologies. In *Proc. of ISWC/ASWC'07*, pages 381–394, 2007.
11. Guilin Qi, Weiru Liu, and David A. Bell. Knowledge base revision in description logics. In *Proceedings of the 10th European Conference on Logics in Artificial Intelligence (JELIA'06)*, pages 386–398, Liverpool, UK, Sep 2006. Springer Verlag.
12. Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.
13. Márcio Moretto Ribeiro and Renata Wassermann. Base revision in description logics - preliminary results. In *Proc. of IWOD'07*, 2007.
14. Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, pages 355–362, Acapulco, Mexico, Aug 2003.
15. Stefan Schlobach, Zhisheng Huang, Ronald Cornet, and Frank van Harmelen. Debugging incoherent terminologies. *Journal of Automated Reasoning*, 39(3):317–349, 2007.
16. Renata Wassermann. An algorithm for belief revision. In *Proc. of KR'00*, pages 345–352, 2000.