On the Use of Belief Revision to Merge Description Logic Terminologies

Martín O. Moguillansky  Marcelo A. Falappa

Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)
Departamento de Ciencias e Ingeniería de la Computación (DCIC)
Universidad Nacional del Sur (UNS)
Consejo de Investigaciones Científicas y Técnicas (CONICET)
Av. Alem 1253 - (B8000CPB) Bahía Blanca - Argentina
PHONE/FAX: (+54)(291)459-5136
E-MAIL: mom@cs.uns.edu.ar  mfalappa@cs.uns.edu.ar

ABSTRACT
Distributed ontologies expressed as description logics may define repeated information. To reason about concepts that these ontologies express, a possible option is to generate unique concept definitions in a different terminology or TBox. The creation of a new terminology from different ontologies need to be consistent, and expressed with non-monotonic logics to be further updated with new distributed ontologies. The model AGM of theory change seems to be an interesting framework to be studied in conjunction with description logics and generate a new non-monotonic description logics model.

1 INTRODUCTION

In order to reason about different ontologies, probably allocated in different places round the web\(^1\), we will consider translated OWL ontologies into description logics (DLs).

In DLs, the concept of Knowledge Base (KB) is composed of two main parts, TBoxes or Terminologies and ABoxes or Assertions. In this paper we’ll focus our investigation on how to reason about terminologies.

Here many possibilities come through. Just think about two distinct terminologies defining each two different main concepts, but containing a same subset of sub-concepts. Or just two distinct terminologies defining the same main concept, which naturally will define the same subset of sub-concepts.

It might be probably impossible to get two concepts defined by different persons with exactly the same logic intention. Here is where the theory change arises as relevant protagonist in order to join consistently two terminologies redefining or reinforcing sub-concepts.

The next section gives a brief description of the DL formalism, continued by section 3 with the analogous description of the AGM theory change model, section 4 contributes to the formalization of merging DL terminologies, and describes an example operation of two different terminologies, and finally section 5 concludes and explains the future work in the area.

2 THE DL BASIC FORMALISM

A Knowledge Representation (KR) system based on Description Logics (DL) provides a formalization to specify the knowledge base (KB) contents, a way to reason about it, and a process to infer implicit knowledge.

A KB is composed by two components. A TBox to manage the terminology of the application world and an ABox containing the assertions about named individuals in terms of the previous concepts.

A terminology is composed by atomic concepts which denote sets of individuals and atomic roles to manage relationships between individuals. Besides, complex concepts and roles are built from the atomics using given constructors.

Reasoning tasks are dedicated to determine whether a description is satisfiable (i.e., non-contradictory), or whether one description is more general than another one, that is, whether the first subsumes the second.

For an ABox, the problem is to verify the consistency of each set of assertions (i.e., test if there is a model for the set) and find out whether a

---

\(^1\)To simplify the complexity of this paper we will consider different ontologies as locals to the host in which the reasoner runs.
For every particular individual is an instance of a concept description in the TBox depending on the assertions in the ABox.

The environment will interact with the KR by querying the KB and finally by adding and retracting concepts, roles and assertions.

2.1 Description Languages

Description Languages are defined by the constructors they provide. In this paper we will consider a subset of the large DL constructors set investigated so far.

The basic Description Language introduced by [Schmidt-Schauß and Smolka, 1991] is the $\mathcal{AL}$ (Attribute Language). Let $A$ be an atomic-concept, $R$ an atomic-role, and $C, D$ complex concepts, the grammar for the $\mathcal{AL}$ language is defined as follows,

$$C, D \rightarrow A \mid \top \mid [\bot \neg A \mid C \cap D \mid \forall R.C \mid \exists R.\top]$$

To define the formal semantics of $\mathcal{AL}$-concepts we use interpretations $(I)$ that consist of a non-empty set $\Delta^I$ (the domain of the interpretation) and an interpretation function $I^j$, that assigns to every atomic concept $A$ a set $A^I \subseteq \Delta^I$ and to every atomic role $R$ a binary relation $R^I \subseteq \Delta^I \times \Delta^I$. The interpretation function $I = (\Delta^I, I^j)$ is extended to concept descriptions by the following inductive definitions,

$$\top^I = \Delta^I$$
$$\bot^I = \emptyset$$
$$(\neg A)^I = \Delta^I \setminus A^I$$
$$(C \cap D)^I = C^I \cap D^I$$
$$(\forall R.C)^I = \{a \in \Delta^I \mid \forall b. (a, b) \in R^I \rightarrow b \in C^I\}$$
$$(\exists R.\top)^I = \{a \in \Delta^I \mid \exists b. (a, b) \in R^I\}$$

Let two concepts $C, D$ be equivalent, writing $C \equiv D$, if $C^I = D^I$ for all interpretations $I$.

Extending $\mathcal{AL}$ by any of the constructors described in Table 1, yields a particular $\mathcal{AL}$-language $\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}][\mathcal{O}]^I$.

For example, given a role completedCourse a student is considered advanced if he has approved 15 courses of a total of 25,

$$\geq 15 \text{ completedCourse} \cap \leq 25 \text{ completedCourse}$$

Note 2: The use of $\mathcal{C}$ is for complement.

Note 3: For $N$ and $\Omega$, $n$ ranges over the nonnegative integers, and $\|X\|$ stands for the cardinality of the set $X$.

For $\Omega$ the number restriction is limited to affect a certain concept. For example, one can also say that an advanced student should approve at least 6 logic courses and 9 computational courses to be considered an advanced student of computer sciences,

$$\geq 6 \text{ completedCourse}\text{.LogicCourse} \cap$$
$$\geq 9 \text{ completedCourse}\text{.ComputationalCourse} \cap$$
$$\leq 25 \text{ completedCourse}$$

2.2 Terminologies

Terminological axioms indicate how concepts or roles are related to each other following the inclusion form, $C \subseteq D$ ( $R \subseteq S$), or the equality form, $C \equiv D$ ( $R \equiv S$), where $C$ and $D$ are concepts ($R$ and $S$ are roles).

An interpretation $J$ satisfies an inclusion $C \subseteq D$ if $C^J \subseteq D^J$, and it satisfies an equality $C \equiv D$ if $C^J = D^J$. Now given a set of axioms $\mathcal{T}$, an interpretation $J$ satisfies $\mathcal{T}$ iff $J$ satisfies each element of $\mathcal{T}$. If $J$ satisfies an axiom in $\mathcal{T}$, then we say that it is a model of this axiom in $\mathcal{T}$. Then two axioms or two set of axioms are equivalent if they have the same models.

Definitions are used to describe complex concepts and made abstraction of them using a single name. An atomic concept on the left side of an equality defines the complex description explained on its right side.

A set of definitions $\mathcal{T}$ is called a terminology or a TBox if a symbolic name is defined only once. A terminology $\mathcal{T}$ contains a cycle iff there exists an atomic concept in $\mathcal{T}$ that uses itself [BN02]; otherwise $\mathcal{T}$ is called acyclic. An acyclic terminology $\mathcal{T}$ can be expanded iteratively through each definition in it, replacing each occurrence of a name on the right hand side with the concepts that it stands for. Now, we say that a terminology $\mathcal{T}$ is definitional if it is acyclic, and we call to its semantics descriptive semantics.

Those semantics that are motivated by the use of intuitively cyclic definitions are called fixpoint semantics. We will not consider fixpoint semantics in this paper.

2.3 Role Constructors

Binary relations between concepts are modeled by roles. If every role name is considered a role description or atomic role, and if $R$ and $S$ are roles descriptions, then $R \cap S$ (intersection), $R \cup S$ (union), $\neg R$ (complement), $R \circ S$ (composition), $R^+$ (transitive closure), and $R^-$ (inverse) are also role descriptions. An interpretation $I$ is adapted to the inverse role description as follows,

$$\geq 6 \text{ completedCourse}\text{.LogicCourse} \cap$$
$$\geq 9 \text{ completedCourse}\text{.ComputationalCourse} \cap$$
$$\leq 25 \text{ completedCourse}$$
Table 1: Constructors to extend the expressivity of AL-languages.

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Written</th>
<th>Interpreted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union (⊔)</td>
<td>((C \sqcup D))</td>
<td>((C \sqcup D)^{\prime} = C^\prime \sqcup D^\prime)</td>
</tr>
<tr>
<td>Negation (¬)</td>
<td>¬C</td>
<td>(¬C^\prime = \Delta^\prime \setminus C^\prime)</td>
</tr>
<tr>
<td>Existential Quantification (∃)</td>
<td>∃R.C</td>
<td>((\exists R.C)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>Number</td>
<td>≥ nR</td>
<td>(≥ nR)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>Restrictions</td>
<td>≤ nR</td>
<td>(≤ nR)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>(N)</td>
<td>= nR</td>
<td>(= nR)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>Qualified</td>
<td>≥ nR.C</td>
<td>(≥ nR.C)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>Number</td>
<td>≤ nR.C</td>
<td>(≤ nR.C)^{\prime} = {a \in \Delta</td>
</tr>
<tr>
<td>Restrictions (Q)</td>
<td>= nR.C</td>
<td>(= nR.C)^{\prime} = {a \in \Delta</td>
</tr>
</tbody>
</table>

Inverse (\(\exists\) or \(^{-1}\)):
\[(R^{-1})^{\prime} = \{(b, a) \in \Delta^2 \times \Delta^2 | (a, b) \in R^\prime\}\]

For instance a hasParent role is obtained applying the inverse role constructor to the given hasChild role.

3 THE AGM CHANGE MODEL

A belief base is a knowledge state represented through a set of sentences not necessarily closed under logical consequence. We also know that a belief set is a set of sentences of a determined language, closed under logical consequence. In general, a belief set is infinite and that’s why it is impossible to deal with them in a computer. Instead, it is possible to characterize the properties that must satisfy each of the change operations on finite representations of a knowledge state.

Expansion operations (“+”) add a new belief to the epistemic state, without guaranteeing its consistence after the operation.

Contraction operations (“−”) eliminate a belief \(\alpha\) from the epistemic state and those beliefs that make possible its deduction or inference. The sentences to eliminate might represent the minimal change on the epistemic state.

Revision operations (“∗”) consist of the insertion of sentences to the epistemic state, guaranteeing consistence (if it was consistent before the operation). This means that a revision adds a new belief and perhaps it eliminates others to avoid inconsistencies.

3.1 Kernel Contractions

The Kernel Contraction operator is applicable to belief bases and belief sets. It consist of a contraction operator capable of the selection and elimination of those beliefs in \(K\) that contribute to infer \(\alpha\).

Definition 3.1.1 [Han94]: Let \(K\) be a set of sentences and \(\alpha\) a sentence. The set \(K^+\alpha\), called set of kernels is the set of sets \(K^\prime\) such that (1) \(K^\prime \subseteq K\), (2) \(K^\prime \vdash \alpha\), and (3) if \(K'' \subset K^\prime\), then \(K'' \not\vdash \alpha\). The set \(K^+\alpha\) is also called set of \(\alpha\)-kernels and each one of its elements are called \(\alpha\)-kernel.

For the success of a contraction operation, we need to eliminate, at least, an element of each \(\alpha\)-kernel. The elements to be eliminated are selected by an Incision Function.

Definition 3.1.2 [Han94]: A function “\(\sigma\)” is an incision function for a set \(K\), if for all sentence \(\alpha\) it verifies, (1) \(\sigma(K^+\alpha) \subseteq \bigcup\{K^+\alpha\}\) and (2) If \(K'' \subseteq K\) and \(K'' \not\equiv \emptyset\) then \(K'' \cap \sigma(K^+\alpha) \not\equiv \emptyset\).

Once that the incision function was applied, we must eliminate from \(K\) those sentences that the incision function selects, i.e. that the new belief base would consist of all those sentences that were not selected by \(\sigma\).

Definition 3.1.3 [Han94]: Let \(K\) be a set of sentences, \(\alpha\) a sentence, and \(K^+\alpha\) the set of \(\alpha\)-kernels of \(K\). Let “\(\sigma\)” be an incision function for \(K\). The operator “\(\Rightarrow_{\sigma}\)”, called kernel contraction determined by “\(\sigma\)”, is defined as, \(K \Rightarrow_{\sigma} \alpha = K \setminus \sigma(K^+\alpha)\).

Finally, an operator “\(\Rightarrow\)” is a kernel contraction operator for \(K\) if and only if there exist an incision function “\(\sigma\)” such that \(K \Rightarrow \alpha = K \Rightarrow_{\sigma} \alpha\) for all sentence \(\alpha\).

3.2 Consistent Merge of Belief Bases

The union of two different belief bases may be inconsistent. Restoring this property to the resultant union may be thought in terms of a deductively maximally consistent (d.m.c.) subset of the union as.

Definition 3.2.1, Partial Meet Merge [Fuh96]: A prima facie candidate for the merge of two sets is any d.m.c. of their union. For each
set \( K \), a set \( X \) is a d.m.c. subset of \( K \), if (1) \( X \subseteq K \), (2) \( X \not\subseteq \bot \), and (3) \( \forall Y : X \subset Y \subseteq K \) implies \( Y \models \bot \). It is easy to see that a set \( X \) is a d.m.c. subset of \( K \) just in case \( X \in K \bot \{\bot\}\).

Finally, Fuhrmann defined in [Fuh96] a partial meet merge operation as a union of two bases, not necessarily closed under logic consequence, and a later consistence restoring applying a bottom contraction. Inspired on it we propose a merge operation over two bases, defined by means of the Kernel Contraction operator, and determined by an Incision Function, as follows,

**Definition 3.2.2** : Let " - " be a kernel contraction for a union of two belief bases \( K_1 \cup K_2 \), determined by an incision function "\( \sigma \)". Then the Merge for Belief Bases operator \( \oplus \) is defined as, \( K_1 \oplus K_2 = (K_1 \cup K_2) - \sigma \bot \).

**4 CONSISTENT UNION OF TERMINOLOGIES**

Let \( T_1 \) and \( T_2 \) be two terminologies to be unified by a consistent union operation, and let \( * \) be the new proposed consistent union of terminologies operator, such that \( T_1 * T_2 \).

Let \( T_1 \) and \( T_2 \) be composed of \( n \) and \( m \) distinct definitions named \( D_{i}^{T_1} \) and \( D_{j}^{T_2} \) respectively. An operator \( * \) needs to evaluate whether \( D_{i}^{T_1,1 \leq i \leq n} \) specifies the same concept that \( D_{j}^{T_2,1 \leq j \leq m} \) does.

For this we define a mapping \( h_{\text{names}} \) that identifies two concept names on distinct terminologies, defining both a same logical concept as, \( D_{i}^{T_1} = h_{\text{names}}(D_{j}^{T_2}) \).

We propose a DL operator \( \sqcup \oplus \) to be the consistent union of concept definitions such that,

\[
D_{k}^{T} = D_{i}^{T_1} \sqcup \oplus D_{j}^{T_2}
\]

A method to verify the consistent union of two such a concept definitions may intuitively be thought as a belief merge of both concepts generating a new unique and consistent concept \( D_{k}^{T} \), where \( \max(n, m) \leq l < (n + m) \).

So let \( K(D_{i}^{T_1}) \) be the belief base (not necessarily closed under logical consequence) that contains the concept definition \( D_{i}^{T_1} \) in a terminology \( T_1 \), and \( K(D_{j}^{T_2}) \) be the belief base that contains the concept definition \( D_{j}^{T_2} \) in a terminology \( T_2 \). The DL operator \( \sqcup \oplus \) between concept definitions will be translated in the merge operator \( \oplus \) between belief bases as defined in the previous section, such that,

\[
D_{i}^{T_1} \sqcup \oplus D_{j}^{T_2} = K(D_{i}^{T_1}) \oplus K(D_{j}^{T_2})
\]

Let \( D_{i}^{T_1} \) and \( D_{j}^{T_2} \) be two expanded concept definitions, and \( D_{i}^{T_1} \sqcup \oplus D_{j}^{T_2} \) be the **Consistent Union of Concept Definitions** operation determined by a Merge of Belief Bases operator \( \oplus \), such that

\[
D = D_{i}^{T_1} \sqcup \oplus D_{j}^{T_2} = D_{i,1}^{T_1} \cap D_{j,2}^{T_2}
\]

where exists \( K_{\sigma}, \) a d.m.c. subset of the incision function \( \sigma((K(D_{i}^{T_1}) \cup K(D_{j}^{T_2})) \perp \bot) \) from now on identified by \( \sigma(C) \), and the base \( K(D_{i}^{T_1}) \), such that exists \( K_{\sigma,\oplus,m,c} \subseteq K(D_{i}^{T_1}) \cap \sigma(C) \) and it gives definition to the new sub-concept

\[
D_{i,1}^{T_1} \equiv D_{i}^{T_1} \setminus K_{\sigma},
\]

\[
D_{j,1}^{T_1} \equiv D_{j}^{T_1} \cap K_{\sigma},
\]

inserted in the resultant terminology \( T \).

Similarly, exists \( K_{\sigma,\oplus,m,c} \subseteq K(D_{j}^{T_2}) \cap \sigma(C) \) and it gives definition to the new sub-concept

\[
D_{j,2}^{T_1} \equiv D_{j}^{T_1} \setminus K_{\sigma},
\]

\[
D_{j,2}^{T_1} \equiv D_{j}^{T_1} \cap K_{\sigma},
\]

in the same resultant terminology \( T \).

It is important to note that for evaluate and solve this operation, is mandatory to unify first each component of the \( \sqcup \oplus \) operator, i.e. \( D_{i}^{T_1} \) and \( D_{j}^{T_2} \) with the correspondent mapping (if it exists) \( h_{names}(D_{i}^{T_1}) \) and \( h_{names}(D_{j}^{T_2}) \), respectively.

**4.2 Example**

The following tables show how two different terminologies might be consistently merged in a new one, following the previous definitions.

| Animal \( \equiv \) Bird \cap \text{Bipedal} \cap \text{Oviparous} \cap \text{hasFeathers} \wedge = 2\text{hasWings} |
| Mammal \( \equiv \) Animal \cap \text{VqiveBirth.LiveBirth} |
| Oviparous \( \equiv \) Animal \cap \text{VqiveBirth.LiveBirth} |
| Bipedal \( \equiv \) 2\text{hasFoot} |
| LiveBirth \( \equiv \) \neg\text{Egg} \wedge \text{hasHeartBeat} \wedge \text{hasVoluntaryMovement} |

Table 2: A terminology \( T_1 \) (TBox) with concepts about animals.
Table 3: A terminology $\mathcal{T}_2$ (TBox) with concepts about animals.

$$∀\text{giveBirth}.\text{Egg} ∨ ∀\neg\text{giveBirth}.\text{Egg}$$

Note that following the proposed method for Consistent Union of Concept Definitions operation, we not only eliminate the incoherence when merging both terminologies, but also keep all information as part of the resultant terminology, by identifying and splitting the problematic concept in two interrelated, revisiting the hierarchy technic of the object oriented paradigm.

Bird $≡$ Animal $∩$ Bipedal $∩$ Oviparous $∩$

hasFeathers $≡$ 2 hasWings

Platypus $≡$ Aquatic $∩$ Monotreme

Monotreme $≡$ Mammal $∩$ Oviparous

Mammal $≡$ Mammal $1$ $∩$ $∀\text{giveBirth}.\text{LiveBirth}$

Oviparous $≡$ Animal $∩$ $∀\text{giveBirth}.\text{Egg}$$

LiveBirth $≡$ $∀\neg\text{Egg}$. hasHeartBeat $∩$

hasVoluntaryMovement $≡$ 2 hasFoot

Table 4: The resultant terminology $\mathcal{T} = \mathcal{T}_1 \star \mathcal{T}_2$ with concepts about animals.

5 CONCLUSIONS AND FUTURE WORK

A union operation of terminologies probably yields contradictions on concept definitions and further inconsistence in the resultant terminology. The use of a belief revision framework to define terminologies in order to meet a consistent merge operation is proposed and generates a new Non-monotonic Description Logics model as a powerful theory to be applied on future Semantic Web researches.

A deeper investigation on Epistemic Entrenchment methods is needed for semi-automate the well functioning of a reasonable incision function $(\sigma)$ to cut the $\alpha$-kernels obtained by the use of merge operations. This means that the $(\sigma)$ function will select those sub-concepts with less epistemic entrenchment to be cut off the resulting definition.

To achieve this, we’ll investigate to incorporate confidence levels on terminologies and definitions, in a way that the origin of a definition will be conditioned to the terminology from which it was “learned”, and a terminology confidence level depending of a probabilistic method deduced from its general confidence level.

References


