Merging taxonomies

Amélie Cordier Jean Lieber Julien Stevenot

May 11, 2012

Contents

1 Introduction **1**

2 Belief revision and belief merging **3**
2.1 Revision of a taxonomy by another one 4
2.2 Merging of knowledge base 4

3 Merging taxonomies **4**
3.1 Taxonomies 4
3.2 The notion of inconsistencies in $\mathcal{L}_\mathcal{T}$ 5
3.3 $CS_\mu(\psi)$ and $MCS_\mu(\psi)$ 7
3.4 Modelling the choice among several possibilities 7
3.5 An operator for merging taxonomies 8
3.6 Properties 9
3.7 Revising a taxonomy by another taxonomy 12

4 The UML that could be use to make an algorithm from this operator **14**

5 Conclusion and future work **15**

1 Introduction

This work is part of the Kolflow project.[1] Kolflow aims at investigating man-machine collaboration in continuous knowledge construction and this collaboration involves to make the conjunction of knowledge from different sources. If the conjunction of the different sources is inconsistent, a merging[2] operator is needed.

[1]Kolflow (`http://kolflow.univ-nantes.fr`, code: ANR-10-CONTINT-025) is supported by the French National Research agency (ANR) and is part of the CONTINT research program.

[2]Merging consistent knowledge bases consists in building a new consistent knowledge base containing as much pieces of knowledge of them as possible.

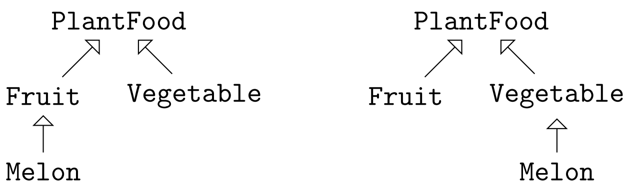


Figure 1: Two taxonomies waiting to be merged.

Kolflow uses semantic wikis as a way of representing knowledge. A good example of semantic wiki for studying collaboration is WikiTaaable.[3]

This wiki must be usable for members of different teams (to simplify, the formal part of WikiTaaable can be seen here as a kind of taxonomy,

i.e. knowledge is modelled through a class hierarchy by some concepts[4] and subsumption[5] relations between concepts).

There is a common stable version of WikiTaaable available on a web site so each team[6] can download it, work on it and make some updates to make its own version of the wiki, at the same time. This process will produce several versions of the same wiki which use similar vocabularies but which do not necessarily agree on everything. The case could happen that one version has been modified by a team and says "A melon is a fruit" whereas the common one, actually on the web, says "A melon is a vegetable" (and the two knowledge bases share the concepts `Vegetable` and `Fruit`) as modelled in the figure 1 (where $\sqsubseteq$ is represented by an arrow).

If this new version is merged with the common one in order to update it, it will be useful that the merging of these two raises a problem between "A melon is a fruit" and "A melon is a vegetable". Indeed, if someone knows the concept `Fruit` and says that melons are vegetables without saying that melons are fruits, he/she may mean that melons *are not* fruits.

The taxonomies form one of the simplest knowledge representation language and as such are interesting to study and to use because of the low

[3] `http://wikitaaable.loria.fr`

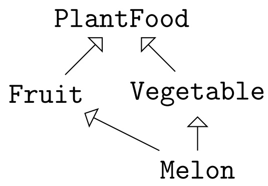
[4] A concept represents a class of objects. For example `Banana` is the concept representing the set of all the bananas.

[5] The subsumption is a relation between two concepts and allows to say that the former represents a subset of the set represented by the latter. It is denoted by $\sqsubseteq$. For example, the formula `Banana` $\sqsubseteq$ `Fruit` represents the knowledge bananas are fruits (the set of bananas is included in the set of fruits).

[6] Kolflow involve several teams in different places who use WikiTaaable as use case.

time and space complexity of their classical inferences. But with the classical semantics, the conjunction of two taxonomies, i.e. the union of their formulas, cannot be inconsistent

and, as such, cannot express all that a human could express like “Melons are not fruits”. For example, the conjunction of the two taxonomies seen in figure 1 is not inconsistent, it just means that melons are, at the same time, fruits and vegetables, as presented in figure 2.



is consistent

Figure 2: The consistent result of the union (conjunction) of the two taxonomies of figure 1.

So how to make arise some inconsistencies during this merging? A way of solving this issue is to increase the expressivity of the representation language but without significantly increasing its time and space complexity. To achieve this goal, this *report* proposal is to add an axiom construct for modelling that melons are not fruits, in the case where a concept `Fruit` exists with the axiom $\texttt{Melon} \sqsubseteq \texttt{Vegetable}$ but without the axiom $\texttt{Melon} \sqsubseteq \texttt{Fruit}$.

With this addition, the conjunction of two taxonomies could raise some contradictions. An example of contradiction is: “A melon is a fruit but is not a vegetable” and “A melon is a vegetable but is not a fruit”. So, a part of the modelled knowledge has to be suppressed, in order to restore consistency. But how one could determine which part should be suppressed and which part should be preserved? In [3], a measure of the agreement and the disagreement between ontologies, that could be useful to make some preferences between pieces of knowledge, is defined. Following the ideas of this work, the idea is to preserve all the agreement and to select some pieces of knowledge of the disagreement.

This *report* is organized as follows.The notions and tools that are used in this *report* are defined in section 2. Section 3 is the core of this *report*: it presents an approach for merging taxonomies. The section 4 show an UML schema that could be use to design an algorithm based on the merging process. Finally, a conclusion and some future work are presented in section 5.

2 Belief revision and belief merging

This section is about the minimal change theory research field in which this *report* aims at contributing. Two important notions of this field are belief revision and belief merging.

2.1 Revision of a taxonomy by another one

If one have two consistent taxonomies, one that he trust and don't want to change, and one that he don't trust, he can revise the taxonomy that he doesn't trust by the other by making the union of them, and solve the conflict that could arise by deleting chosen part (that are in conflict) of the taxonomy that he doesn't trust until he got something consistent.

In [1], some general postulates of belief revision have been proposed. Then, in [5], some others postulates, have been proposed for the particular case of revision in propositional logics (and equivalent to the [1]'s one in this formalism). So the revision of a knowledge base ψ_1 by another one ψ_2 starts by making the conjunction of ψ_1 and ψ_2 and if this conjunction is consistent, it is the result of the revision. Else *minimal* modifications $\psi_1 \mapsto \psi_1'$ have to be done such that $\psi_1' \wedge \psi_2$ is consistent. ψ_2 must not be modified.

2.2 Merging of knowledge base

If one have two consistent taxonomies, with the same level of trust on both of them, he can merge these taxonomies by making the union of them, and solve the conflict that could arise by deleting chosen part (are in conflict) of the taxonomies that until he got something consistent.

Let Δ be a merging operator of consistent knowledge bases ψ_1, ψ_2, ..., ψ_n. If the conjunction of all the knowledge bases ψ_1, ψ_2, ..., ψ_n is consistent, the result of the merging is their conjunction $\bigwedge_i \psi_i = \psi_1 \wedge \psi_2 \wedge ... \wedge \psi_n$. Else, *minimal* modifications of all the bases $\psi_1 \mapsto \psi_1'$, $\psi_2 \mapsto \psi_2'$, ..., $\psi_n \mapsto \psi_n'$ such that $\psi_1' \wedge \psi_2' \wedge ... \wedge \psi_n'$ is consistent have to be done, and:

$$\Delta(\{\psi_1, \psi_2, ..., \psi_n\}) \equiv \psi_1' \wedge \psi_2' \wedge ... \wedge \psi_n'$$

Some postulates of merging, inspired from the postulates of revision, are presented in [6].

3 Merging taxonomies

3.1 Taxonomies

The term *taxonomy* has been created by biologists for talking about the classification of the species. But, etymologically, it means arrangement method and is used to refer to a class hierarchy. So, here the term is used for a class hierarchy which is represented formally by a language (called here $\mathcal{L}_\mathcal{T}$ for taxonomy's language).

$\mathcal{L}_\mathcal{T}$ is defined as follows (reusing the description logics notations [2]). Let $\mathcal{A}$ be a countable set: $A \in \mathcal{A}$ is called a concept (only atomic concepts are allowed in $\mathcal{L}_\mathcal{T}$). A formula of $\mathcal{L}_\mathcal{T}$ has the form $A \sqsubseteq B$ where A, $B \in \mathcal{A}$ and $A \neq B$,[7] meaning that the concept A is more specific than the concept B (formally: for each model ω of $A \sqsubseteq B$, $\omega(A) \subseteq \omega(B)$). A taxonomy is a knowledge base of $\mathcal{L}_\mathcal{T}$ (i.e., a finite set of $\mathcal{L}_\mathcal{T}$ formulas).

The vocabulary $\mathcal{V}(\psi)$ of a taxonomy ψ is defined as follows.
For $A, B \in \mathcal{A}$, $\mathcal{V}(A \sqsubseteq B) = \{A, B\}$. For a taxonomy ψ, $\mathcal{V}(\psi) = \bigcup\{\mathcal{V}(f) \mid f \in \psi\}$.

The language $\mathcal{L}_\mathcal{T}$ has been chosen because it is one of the simplest knowledge representation languages and, as such, its inferences are of low complexity, i.e. the sumbsomption test is linear for $\mathcal{L}_\mathcal{T}$ (it can be completed by searching a directed path in a graph). So an efficient (in term of time and space complexity) merging operator should be definable in this language. And, moreover, this language is sufficient to express most of the formal knowledge edited in WikiTaaable.

3.2 The notion of inconsistencies in $\mathcal{L}_\mathcal{T}$

Let us consider ψ_1 and ψ_2, the two taxonomies in figures 3 and 4. ψ_1 states that melons are fruits and ψ_2 states that melons are vegetables. Formally there is no contradiction there: ψ_1 (resp., ψ_2) does not entail that melons are not vegetables (resp., fruits).

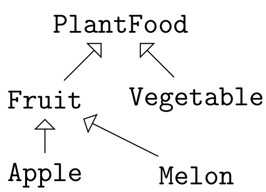


Figure 3: ψ_1.

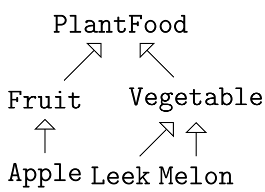


Figure 4: ψ_2.

More generally, if ψ_1 and ψ_2 are two taxonomies (two finite subsets of $\mathcal{L}_\mathcal{T}$), $\psi_1 \cup \psi_2$ is also a taxonomy and therefore, is consistent.[8]

[7] whithout loss of expressivity, the tautologies $A \sqsubseteq A$ are excluded of the formalism.

[8] Every taxonomy is satisfiable and thus consistent. Indeed, if $\psi = \{A_i \sqsubseteq B_i\}_i$ is a taxonomy, it is satisfied by the interpretation whose domain is $\{1\}$ and function ω associates, for any i, A_i to $\omega(A_i) = \{1\}$ and B_i to $\omega(B_i) = \{1\}$.

Now, when considering again ψ_1 and ψ_2 of figures 3 and 4, the fact that $\psi_1 \not\models \texttt{Melon} \sqsubseteq \texttt{Vegetable}$ and $\psi_2 \not\models \texttt{Melon} \sqsubseteq \texttt{Fruit}$ may have two intuitive interpretations:

- Either ψ_1 and ψ_2 are incomplete in the sense that the person in charge of the development of ψ_1 (resp., ψ_2) does not know whether melons are or are not vegetables (resp., fruits);
- Or the persons in charge of the development of ψ_1 and ψ_2 are in disagreement: the former thinks that melons are fruits and are not vegetables, the latter thinks that melons are vegetables and are not fruits.

Therefore the merging of ψ_1 and ψ_2 should lead to a taxonomy ψ satisfying one of the four possibilities:

$$
\begin{array}{l}
(a)\ \psi \models \texttt{Melon} \sqsubseteq \texttt{Fruit} \text{ and } \psi \models \texttt{Melon} \sqsubseteq \texttt{Vegetable} \\
(b)\ \psi \models \texttt{Melon} \sqsubseteq \texttt{Fruit} \text{ and } \psi \not\models \texttt{Melon} \sqsubseteq \texttt{Vegetable} \\
(c)\ \psi \not\models \texttt{Melon} \sqsubseteq \texttt{Fruit} \text{ and } \psi \models \texttt{Melon} \sqsubseteq \texttt{Vegetable} \\
(d)\ \psi \not\models \texttt{Melon} \sqsubseteq \texttt{Fruit} \text{ and } \psi \not\models \texttt{Melon} \sqsubseteq \texttt{Vegetable}
\end{array}
$$

Hence, if the conjunction of two taxonomies corresponds to their union, only situation (a) can occur. To prevent that situation, taxonomies are considered according to a closed world assumption (CWA):

$$\frac{\psi \not\models A \sqsubseteq B}{A \not\sqsubseteq B}\ \text{CWA}$$

This entails that the formulas $A \not\sqsubseteq B$ are considered. Let $\mathcal{L}_{\mathcal{T}}^{\neg}$ be the language of taxonomies with negations. A formula of $\mathcal{L}_{\mathcal{T}}^{\neg}$ is either a formula of $\mathcal{L}_{\mathcal{T}}$ or a formula $A \not\sqsubseteq B$ for $A, B \in \mathcal{A}$. The semantics of $\mathcal{L}_{\mathcal{T}}^{\neg}$ is as follows: ω satisfies $A \not\sqsubseteq B$ if $\omega(A) \not\subseteq \omega(B)$.

In order to integrate the closed-world assumption in the conjunction, for ψ an $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge base, let $\widehat{\psi}$ be the deductive closure (including CWA) of ψ defined by:

$$
\begin{aligned}
\widehat{\psi} &= \{A \sqsubseteq B \mid A, B \in \mathcal{V}(\psi) \text{ and } \psi \models A \sqsubseteq B\} \\
&\cup\ \{A \not\sqsubseteq B \mid A, B \in \mathcal{V}(\psi) \text{ and } \psi \not\models A \sqsubseteq B\}
\end{aligned}
$$

$\widehat{\psi}$ can be viewed as a clique whose vertices are elements of $\mathcal{V}(\psi)$ as illustrated on figure 5 where $A \not\sqsubseteq B$ is represented by a dashed bracket-headed arrow from A to B. For the sake of simplicity, in the next examples the deductive closure will not always be graphically represented.

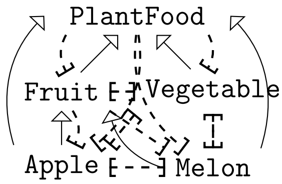


Figure 5: $\widehat{\psi_1}$, with the ψ_1 of figure 3.

Now, the conjunction of two taxonomies ψ_1 and ψ_2 (of $\mathcal{L}_{\mathcal{T}}$ or of $\mathcal{L}_{\mathcal{T}}^{\neg}$) is defined by:

$$\psi_1 \wedge \psi_2 = \widehat{\psi_1} \cup \widehat{\psi_2}$$

With this definition, the conjunction of the taxonomies of the figures 3 and 4 is inconsistent since, e.g.,$\{\mathtt{Melon} \sqsubseteq \mathtt{Fruit}, \mathtt{Melon} \not\sqsubseteq \mathtt{Fruit}\} \subseteq \psi_1 \wedge \psi_2$.

With that, the merging of these two taxonomies raises two inconsistencies (or *clashes*) that have to be solved:

$$\begin{aligned} clash_1 &= \{\mathtt{Melon} \sqsubseteq \mathtt{Fruit}, \quad \mathtt{Melon} \not\sqsubseteq \mathtt{Fruit}\} \\ clash_2 &= \{\mathtt{Melon} \sqsubseteq \mathtt{Vegetable}, \quad \mathtt{Melon} \not\sqsubseteq \mathtt{Vegetable}\} \end{aligned}$$

3.3 $CS_\mu(\psi)$ and $MCS_\mu(\psi)$

Let μ and ψ be two $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases, such that μ is consistent. Let $CS_\mu(\psi)$ be the set of knowledge bases φ such that $\mu \subseteq \varphi \subseteq \psi \cup \mu$ and φ is consistent (CS stands for "consistent subsets"). $CS_\mu(\psi) \neq \emptyset$ since $\mu \in CS_\mu(\psi)$. Among the elements of $CS_\mu(\psi)$, the largest ones for inclusion constitute $MCS_\mu(\psi)$ (MCS stands for maximal consistent subset). If $\psi \cup \mu$ is consistent, then $MCS_\mu(\psi) = \{\psi \cup \mu\}$.

For example (using the notations of the previous sections), if $\psi = clash_1 \cup clash_2$, then $MCS_\emptyset(\psi)$ is composed of the four consistent knowledge bases (a), (b), (c), and (d).

3.4 Modelling the choice among several possibilities

As pointed out above, there may be several possibilities and so, it is necessary to make a choice among them. This possibility to make a choice is represented by a preorder $\leq$ on the knowledge bases of $\mathcal{L}_{\mathcal{T}}^{\neg}$ such that $\psi_1 < \psi_2$ means that ψ_1 is preferred to ψ_2 ($\psi_1 < \psi_2$ means that $\psi_1 \leq \psi_2$ and $\psi_2 \not\leq \psi_1$).

$\leq$ is assumed to be a total order up to the logical equivalence: it is reflexive and transitive, if $\psi_1 \leq \psi_2$ and $\psi_2 \leq \psi_1$ then ψ_1 and ψ_2 are equivalent, and for any ψ_1 and ψ_2, either $\psi_1 \leq \psi_2$ or $\psi_2 \leq \psi_1$. Therefore, if S is a finite set of $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases, the minimal of S for $\leq$ exists and is unique, modulo equivalence, and it is denoted by $Min_{\leq}(S)$.

Moreover, $\leq$ is assumed to prefer more specific knowledge bases, i.e., if $\psi_1 \subseteq \psi_2$ then $\psi_2 \leq \psi_1$. This property involves that $Min_{\leq}(CS_{\mu}(\psi)) = Min_{\leq}(MCS_{\mu}(\psi))$.

3.5 An operator for merging taxonomies

The merging operator presented in this section is inspired from the ideas of agreement and disagreement of two ontologies as introduced in [3]. Let ψ_1, ψ_2, ..., ψ_n be n consistent knowledge bases of $\mathcal{L}_{\mathcal{T}}^{\neg}$ (e.g., two taxonomies) and $E = \{\psi_1, \psi_2, ..., \psi_n\}$. The notions introduced below are illustrated with the taxonomies of figures 3 and 4.

The agreement α of ψ_1, ψ_2, ..., ψ_n is constituted by the pieces of knowledge common to them. formally:

$$\alpha = \bigcap_i \widehat{\psi_i} = \widehat{\psi_1} \cap \widehat{\psi_2} \cap ... \cap \widehat{\psi_n}$$

α is necessary consistent

(since $\alpha \subseteq \widehat{\psi_1}$ that is consistent).

Figure 7 shows a representation of α.

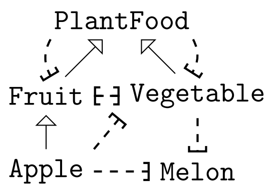


Figure 6: α: the agreement of the ψ_1 and ψ_2 of figures 3 and 4, represented without the edges that can be deduced by CWA.

Definition: The disagreement is intuitively defined as the pieces of knowledge that are not in agreement.[9] This disagreement is defined as $\delta = \bigcup_i \delta_i$

[9]This slightly differs from [3] where the agreement and the disagreement are not complementary.

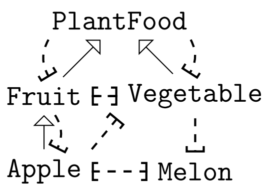


Figure 7: α: the agreement of the ψ_1 and ψ_2 of figures 3 and 4, represented without some of the edges that can be deduced by CWA. (** + clair ?**)

where δ_i represents the pieces of knowledge of ψ_i that are not in agreement with the ψ_j's ($j \neq i$):

$$\delta_i = \widehat{\psi_i} \setminus \alpha$$

Since ψ_i is consistent, δ_i is also consistent. Figures 8 and 9 illustrate δ_1 and δ_2.

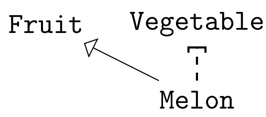


Figure 8: δ_1.

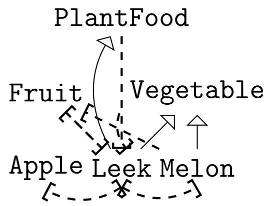


Figure 9: δ_2.

So, here, δ is the union of δ_1 and δ_2.

Then, a subset β of δ has to be chosen. $\alpha \cup \beta$ has to be consistent and has to keep as much knowledge as possible, i.e. $\beta \in MCS_\alpha(\delta)$. If the choice is made according to $\leq$ (cf section 3.4) then:

$$\beta = Min_{\leq}(MCS_\alpha(\delta))$$

Finally, the result of the merging is a knowledge base of $\mathcal{L}_\mathcal{T}$ such that:

$$\widehat{\Delta(E)} = \widehat{\beta}$$

Figures 10 to 13 present the four possibilities for $\Delta(\psi_1, \psi_2)$, depending on the choice $\leq$.

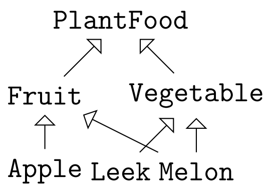


Figure 10: Result of the merging after choosing (a).

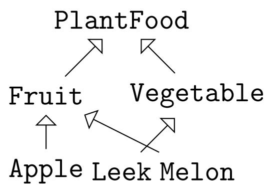


Figure 11: Result of the merging after choosing (b).

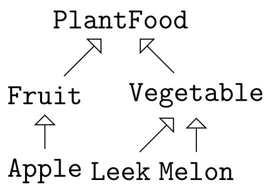


Figure 12: Result of the merging after choosing (c).

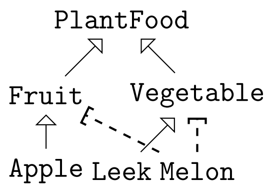


Figure 13: Result of the merging after choosing (d).

3.6 Properties

First, Δ can be confronted to the postulates of [6]. These postulates are used for characterizing a merging operator in propositional logic, but can be reused in the $\mathcal{L}_{\mathcal{T}}$ formalism. These postulates deal with the merging of multisets of knowledge bases, but, since for the operator Δ, the number of occurrences has no importance, we will consider only sets of knowledge bases.

These postulates are (for $E,\ E_1,\ E_2$: sets of knowledge bases; $\psi_1,\ \psi_2$: knowledge bases):

(A1) $\Delta(E)$ is consistent.

(A2) If $\bigwedge E$ is consistent then $\Delta(E)$ is equivalent to $\bigwedge E$.

(A3) If there is a bijection F from E_1 to E_2 such that $F(\psi)$ is equivalent with ψ, then $\Delta(E_1)$ is equivalent to $\Delta(E_2)$ (this postulates states that the syntax is irrelevant for Δ).

(A4) If $\psi_1 \wedge \psi_2$ is not consistent, then $\Delta(\{\psi_1, \psi_2\}) \not\models \psi_1$.

(A5) $\Delta(E_1) \wedge \Delta(E_2) \models \Delta(E_1 \cup E_2)$.

(A6) If $\Delta(E_1) \wedge \Delta(E_2)$ is consistent, then $\Delta(E_1 \cup E_2) \models \Delta(E_1) \wedge \Delta(E_2)$.

Δ satisfies (A1).

Indeed, $\Delta(\{\psi_1, ..., \psi_n\}) \in MCS_\alpha(\delta)$ and thus is consistent.

Δ satisfies (A2).

To prove it, let us assume that $\bigwedge E$ is consistent. $\bigwedge E = \bigwedge_i \widehat{\psi_i} = \alpha \cup \delta$. Thus $\alpha \cup \delta$ is consistent and so $MCS_\alpha(\delta) = \{\alpha \cup \delta\}$. Hence $\Delta(E) = \alpha \cup \delta = \bigwedge E$. Therefore, if $\bigwedge E$ is consistent then $\Delta(E) = \bigwedge E$ which proves (A2).

Δ satisfies (A3), which states the irrelevance of syntax.

Indeed, for any knowledge bases ψ_1 and ψ_2 of $\mathcal{L}_{\mathcal{T}}^{\neg}$, ψ_1 is equivalent to ψ_2 iff $\widehat{\psi_1} = \widehat{\psi_2}$. Since Δ is defined thanks to the $\widehat{\psi_i}$'s, $\Delta(E)$ does not change when substituting a ψ_i by an equivalent knowledge base.

(A4) is not satisfied by Δ as the following counterexample shows.

Let $\psi_1 = \{A \sqsubseteq B\}$ and $\psi_2 = \{A \not\sqsubseteq B\}$. Then $\widehat{\psi_1} = \{A \sqsubseteq B, B \not\sqsubseteq A\}$ and $\widehat{\psi_2} = \{A \not\sqsubseteq B, B \not\sqsubseteq A\}$. $\psi_1 \wedge \psi_2 = \{A \sqsubseteq B, A \not\sqsubseteq B, B \not\sqsubseteq A\}$, $\alpha = \{B \not\sqsubseteq A\}$, $\delta_1 = \{A \sqsubseteq B\}$, $\delta_2 = \{A \not\sqsubseteq B\}$, $\delta = \{A \sqsubseteq B, A \not\sqsubseteq B\}$, $MCS_\alpha(\delta) = \{\{A \sqsubseteq B, B \not\sqsubseteq A\}, \{A \not\sqsubseteq B, B \not\sqsubseteq A\}\}$.

Thus according to the choice performed by $\leq$, $\Delta(\{\psi_1, \psi_2\}) \models \psi_1$ or $\Delta(\{\psi_1, \psi_2\}) \models \psi_2$. (A4) is called in [6] the fairness property: it states that Δ should not make a preference between the knowledge bases to be merged. Our interpretation of the non fairness of our operator is that the $\mathcal{L}_{\mathcal{T}}^{\neg}$ language does not permit to express disjunctions and so, the operator has to make a choice (that is why $\leq$ has to be a total order).

Indeed, let us consider $\mathcal{L}_{\mathcal{T}}^{\neg\vee}$ the extension of $\mathcal{L}_{\mathcal{T}}^{\neg}$ with disjunction: if ψ_1 and ψ_2 are $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases, then $\psi_1 \vee \psi_2$ is an $\mathcal{L}_{\mathcal{T}}^{\neg\vee}$ knowledge base and ω satisfies it if ω satisfies ψ_1 or ω satisfies ψ_2. Now,let ∇ be the merging operator defined by $\nabla(E) = \bigvee MCS_\alpha(\delta)$(E: a set of $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases, $\nabla(E)$: an $\mathcal{L}_{\mathcal{T}}^{\neg\vee}$ knowledge base). ∇ satisfies (A1), (A2), and (A3) (similar proofs than the proofs for Δ) and it satisfies also (A4):

Proof. Let ψ_1, ψ_2 be two consistent $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases such that $\psi_1 \wedge \psi_2$ is consistent. Thus, $\alpha = \widehat{\psi_1} \cap \widehat{\psi_2}$, $\beta_1 = \widehat{\psi_1} \setminus \alpha$, $\beta_2 = \widehat{\psi_2} \setminus \alpha$. $\alpha \cup \beta_1 = \widehat{\psi_1}$ and $\alpha \cup \beta_2 = \widehat{\psi_2}$ are consistent, so there exist ϕ_1 and ϕ_2 such that $\phi_i \in MCS_\alpha(\psi_1 \wedge \psi_2)$, $\widehat{\psi_i} \subseteq \phi_i (i \in \{1, 2\})$, and $\phi_1 \cup \phi_2$ is inconsistent (since $\phi_1 \cup \phi_2 \equiv \widehat{\psi_1} \cup \widehat{\psi_2} = \psi_1 \wedge \psi_2$ that is inconsistent). Therefore $\phi_1 \wedge \phi_2 \models \nabla(\{\psi_1, \psi_2\})$, $\phi_1 \wedge \phi_2 \not\models \phi_1$ (since $\phi_1 \not\models \phi_2$), $\phi_1 \wedge \phi_2 \not\models \phi_2$ (since $\phi_2 \not\models \phi_1$). Hence, $\nabla(\{\psi_1, \psi_2\}) \not\models \phi_i$ for $i \in \{1, 2\}$. $\square$

This is why the non fairness of Δ is interpreted as a consequence of the necessity to make choices, in the $\mathcal{L}_{\mathcal{T}}^{\neg}$ formalism.

At this point, we have neither proven that Δ satisfies (A5) and/or (A6), nor found any counterexample.

A detailed complexity analysis has still to be carried out.

However, a naive algorithm for Δ gives a polynomial complexity for the computation α and δ and an exponential complexity for the computation of $MCS_\alpha(\delta)$ (exponential in the size of δ). Therefore, with this algorithm, the computation of Δ is tractable when the taxonomies are similar. Indeed $\delta = \bigcup_i \widehat{\psi_i} - \bigcap_i \widehat{\psi_i}$ contains the formulas that are not shared by the taxonomies, so $|\delta|$ can be used to characterize the dissimilarities of the ψ_i's. Hence making frequents merging of taxonomies that have forked from a same taxonomy is useful. [10]

3.7 Revising a taxonomy by another taxonomy

Reusing the previously seen notations, a revision operator $\dot{+}$ on taxonomies can be defined:

$$\psi_1 \dot{+} \psi_2 = Min_{\leq}(MCS_{\widehat{\psi_2}}(\widehat{\psi_1} \cup \widehat{\psi_2}))$$

where ψ_1 and ψ_2 are two taxonomies (or, more generally, two $\mathcal{L}_{\mathcal{T}}^{\neg}$ knowledge bases). This means that the revision of ψ_1 by ψ_2 is chosen among the knowledge bases that entail ψ_2 and makes a minimal generalisation on ψ_1 (which is consistent with the intuitive definition of revision given in section 2).

The properties of this revision operator remains to be studied, in particular, according to the postulates of [5]. Its complexity is similar to the complexity of Δ and thus remains tractable.

[10] This can be likened to the usefulness of frequent commits in a version management system like subversion, as noticed by Fabien Gandon. Thanks for this relevant remark, Fabien !

4 The UML that could be use to make an algorithm from this operator

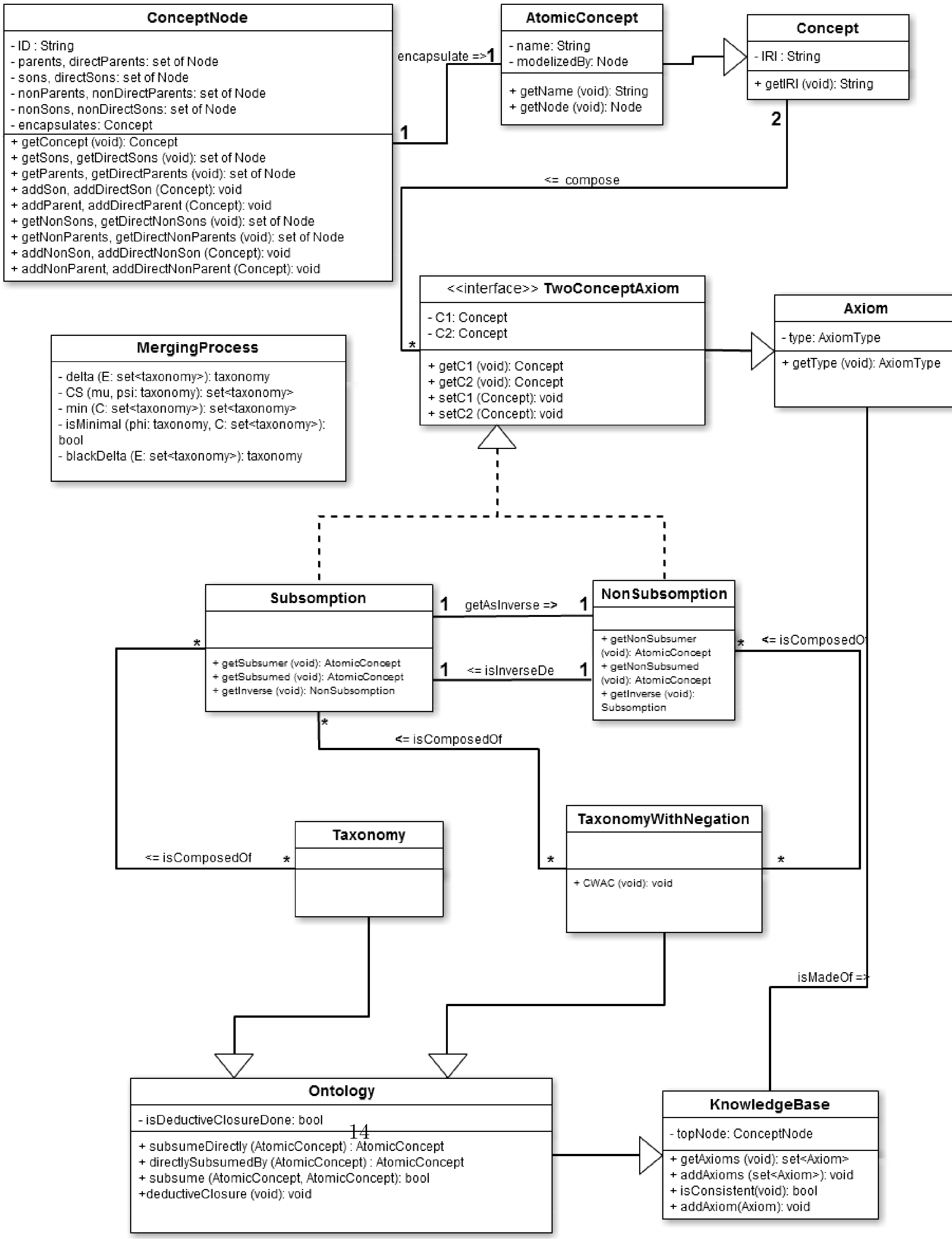


Figure 14: The UML schema

5 Conclusion and future work

This *report* has presented an operator for merging similar taxonomies that satisfies a subset of the postulates defined in [6] but some proof are still to be done.

Why is this algorithm useful for Kolflow ?

- With this algorithm, the user don't have to download and update the whole taxonomy in order to add just a few knowledge. He can just submit his own knowledge.
- Compare to the previous system where the user submit a full consistent ontology, containing all the knowledge he want to add, which is rejected or not by the test campaign, here, even if all his knowledge are not accepted, a part of it can be preserved and integrate to the system.
- We could keep traces of the choices of the users and use it to improve the choice method.

In conclusion, the operator presented herein seems to address Kolflow's issues for taxonomy merging. There is still work to do in order to study its properties and to design an efficient algorithm.

This operator is used to design an efficient algorithm for this merging when the taxonomies are similar, which is the case when they are originated from the same taxonomy and have not diverged for a too long time. This algorithm, in order to be efficient, should not compute $\widehat{\psi}$ (this operation is too complex and is too time and space consuming: $|\widehat{\psi}| = |\mathcal{V}(\psi)|^2 - |\mathcal{V}(\psi)|$).

The design of such an algorithm involves that the relation $\leq$ has to be specified. Indeed, the operator presented in this *report* is based on the maximal consistent subsets of formulas issued from the conjunction of the knowledge bases to be merged. The relation $\leq$ has to define a preference order between these subsets of formulas. One possibility for specifying $\leq$ is to ask a human user but it implies the loss of some properties like the determinism that could be seen as desirable and it raises another question: how to present these choices to the user? The merging of two big taxonomies could lead to a great number of choices. So it could be a good thing to only pop up the most relevant ones. For example, in the previously seen example, there were four possibilities. One can think that when there is two inconsistent minimal subsets like $\{A \sqsubseteq B, A \not\sqsubseteq B\}$ and $\{A \sqsubseteq C, A \not\sqsubseteq C\}$, the two relevant choices should be ($A \sqsubseteq B$ *and* $A \not\sqsubseteq C$) or ($A \sqsubseteq C$ *and* $A \not\sqsubseteq B$). The way this choice can be formalized is another relevant issue for future work.

Another way to specify the relation $\leq$ is to take into account a history of the previous choices, i.e., for example, if the choices between two knowledge bases ϕ_1 and ϕ_2 has already been done in the past, $\leq$ will choose the same that have been chosen. This idea remains to be studied in details.

Once this algorithm is efficiently implemented, it could be used, in the Kolflow project, for example, to update the previously seen common version of WikiTaaable. But Kolflow does not limit itself to $\mathcal{L}_\mathcal{T}$ and there is a large spectrum of languages ranging from $\mathcal{L}_\mathcal{T}$ to, e.g., OWL DL. One advantage of $\mathcal{L}_\mathcal{T}$ is that its inferences are much less complex than OWL DL's (e.g., the sumbsumption test is linear for $\mathcal{L}_\mathcal{T}$ whereas it is NExpTime-hard in OWL DL). The question we intend to address in a future work is what are the extensions of $\mathcal{L}_\mathcal{T}$ for which we will design a merging operator. Since $\mathcal{L}_\mathcal{T}$ can be considered as the fragment of RDFS with only one possible properties, SubClassOf (corresponding to $\sqsubseteq$), some larger fragments should be considered (using other properties). Indeed in the particular case of WikiTaaable, some properties are more used or important and some are easier to compute than other ones so one can think of a kind of anytime approach where the algorithm will consecutively consider the RDFS properties starting by subClassOf.

A kind of equivalent to the MCS is the $MUPS$ that are used in the system Pellet:[11] this system contains a tool for debugging inconsistent ontologies which allows to find the $MUPS$ [4] of an inconsistent ontology. A MUPS (Minimal Unsatisfiability Preserving Sub-TBoxes) is a minimal subset of axioms which causes the inconsistency. If we find all the $MUPS$ of a knowledge base issued from the conjunction of two other ones, the set of all the possible consistent knowledge bases made from the conjunction of all the $MUPS$ after deleting one formula on each of them, is equivalent to the MCS. As Pellet works on knowledge bases on OWL DL it could be a lead to pass from $\mathcal{L}_\mathcal{T}^\neg$ to OWL DL. It also could allow to compare our algorithm to the results of Pellet's debugging tool.

References

[1] C. E. Alchourrón, P. Gärdenfors, and D. Makinson. On the Logic of Theory Change: partial meet functions for contraction and revision. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications.* Cambridge University Press, Cambridge, 2003.

[11] http://clarkparsia.com/pellet/

[3] M. d'Aquin. Formally measuring agreement and disagreement in ontologies. In *Proceedings of the fifth international conference on Knowledge capture*, K-CAP '09, pages 145–152, New York, NY, USA, 2009. ACM.

[4] B. C. Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Debugging OWL ontologies. pages 633–640, 2005.

[5] H. Katsuno and A. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3):263–294, 1991.

[6] S. Konieczny and R. Pino-Pérez. On the logic of merging, 1998.