# Semantic Import: An Approach for Partial Ontology Reuse (*)

Jeff Z. Pan[1], Luciano Serafini[2] and Yuting Zhao[2]

[1]University of Aberdeen, UK
jpan@csd.abdn.ac.uk
[2]ITC-IRST, Povo 38100 Trento, Italy
{yzhao,serafini}@itc.it

**Abstract.** As more and more applications use ontology to represent semantic information, how to support ontology reuse is becoming more and more important. In the OWL Web Ontology Language, the main primitive for ontology reuse is owl:imports. However, this "copy-and-paste" approach suffers a number of problems. In this paper, we propose a new import primitive, called *semantic import*, to facilitate *partial ontology reuse*. The main contributions of this paper include the semantics of semantic import and reasoning support for TBox reasoning in simple ontology spaces, where an ontology semantically imports vocabulary from another ontology. The semantic import approach provides a brand new way for partial ontology reuse.

## 1 Introduction

Ontologies [17] play an important role in the Semantic Web [2]. The advent of the connectivity and the Web makes an increasing number of ontologies widely available for reuse. As more and more applications use ontology to represent semantic information, how to support ontology reuse [18] is becoming more and more important. In this paper, we assume that ontologies are all written in a standard ontology language, such as the W3C recommendation OWL Web Ontology Language [12], and do not consider the translation of ontological axioms from one ontology language to another.

The basic primitive for in the OWL Web ontology language to support reuse is `owl:imports` [12], which allows one to "copy-and-paste" an ontology into another one. We call this operation *syntactic import*, to stress the fact the operation is defined on the set of axioms of an ontology, rather than the knowledge encoded in the ontology itself. The "copy-and-paste" approach provides only limited support for *partial ontology reuse*, which means importing only a (small) part of the knowledge encoded in an ontology, while ignoring the rest. We illustrate this issue with the following example.

(*) Thanks to Prof. Fausto Giunchiglia for the discussions on this topic, he has participated to write previous versions of this paper. This work is supported by the FP6 Network of Excellence EU project Knowledge Web (IST-2004-507842).

*Example 1.* Suppose that we want to build our own ontology for people transportation, let us say $\mathcal{W}_{\texttt{ppl}}$, by reusing other ontology present on the Web, e.g. SUMO [9]. In our ontology, one can use cars and horses as transportation devices; therefore, we would like to impose the following axioms:

$$\texttt{sumo:Horse} \sqsubseteq \texttt{sumo:TransportationDevice} \tag{1}$$

$$\texttt{sumo:Car} \sqsubseteq \texttt{sumo:TransportationDevice} \tag{2}$$

Furthermore, we would like to reuse some knowledge which is explicitly stated in or entailed by SUMO:

$$\texttt{sumo:Car} \sqsubseteq \exists \texttt{sumo:}\mathit{capability}.(\texttt{sumo:}\mathsf{Transportation}) \tag{3}$$

There are three approaches to reaching the *partial ontology reuse* through syntactic imports: (i) simply import the whole SUMO ontology into $\mathcal{W}_{\texttt{ppl}}$, and only use some of the axioms, ignoring the others; (ii) only "copy-and-paste" a subset of the SUMO axioms into $\mathcal{W}_{\texttt{ppl}}$; (iii) divide the SUMO ontology into different modules, based on some criteria, and import only the interesting module. All these approaches present some problems, as discussed below.

For (i), when importing the whole ontology and only using some of its axioms the following drawback can raise. Firstly, inconsistency can easily be reached by importing those conflicting axioms from the source ontology. Secondly, there exist partial disagreements in certain axioms; e.g., we might agree that Horses are quadrupeds and mammals, but we do not agree that they are not transportation devices. Thirdly, scalability could become a problem. This means that intention of reusing only a few objects and/or classes in a source ontology would end up importing the whole ontology.

For (ii), in order to import a subset of the axioms, users have to be carefully identify which set of axioms encodes the knowledge he wants to import; otherwise, it will cause information lost. Sometimes, the structure of the ontology is such that this is not possible. For example, given an ontology containing the axioms $A \sqsubseteq B$ and $B \sqsubseteq C$, a user wants to import only $A \sqsubseteq C$ without committing on $B$.

Approach (iii) actually is most promising. The main difficulty of (iii) is that the modularisation of an ontology is far from easily customisable and generalisable to any ontology. A recent approach in this line is the work by Cuenca-Grau et al. [5], which provides the theory and an algorithm for the modularisation of a wide class of ontologies, called safe ontologies.

The main contributions of this paper are as follows. Firstly, it proposes semantic imports, a new mechanism of partial ontology reuse. One distinguished feature of this new partial reuse mechanism is to allow users to semantically reuse vocabulary (classes, properties or individuals), while current approaches allow to reuse axioms. Moreover, semantic imports provide a flexible way to partially reuse ontologies, by allowing users (in their target ontologies) to agree or disagree with some subsumption relationships in a distant ontology. More specifically, in our mechanism all subsumption relations among the *semantically*

*imported named classes* in the distant ontology are transferred to the target ontology. Secondly, the paper address some logical properties and algorithm of a reasoning problem for a special case in our framework, i.e., semantic imports in simple ontology spaces, where an ontology semantically imports vocabulary from another ontology. Based on the notion of local/non-local classes first proposed by Cuenca-Grau et al. [5] (we call them positive/negative classes), several categories of subsumption relations among class descriptions over the semantically imported vocabulary can be transferred from a distant ontology to a target ontology in simple ontology spaces. Furthermore, it presents a distributed tableaux algorithm to compute class satisfiability in simple ontology spaces, in the presence of semantic import of $\mathcal{ALC}$ ontologies.

## 2 Ontologies with Imports

An OWL [12] ontology, as defined by the W3C standard, consists of a set of *axioms*, including class axioms, property axioms and individual axioms.[1] The reader is referred to [12] for details of syntax and semantic of OWL. An OWL ontology can import (other) OWL ontologies, with the help of imports annotations similar to the following.

```
Annotation(imports <http://www.car.org/car#>)
Annotation(imports <http://www.xyz.it/vehicle#>)
```

These imports annotations include all the axioms in the two source ontologies into the target ontology.

We extend the notion of ontology by allowing the semantic import of classes, properties, and individuals from source ontologies. To simplify the presentation, in this paper we only consider the abstract domain and leave the discussion of the datatype domain [10] as our future work.

**Definition 1 (Ontology).** *An* ontology *is a tuple* $\mathcal{W}_{\mathtt{i}} = \langle \mathtt{i}, \mathcal{M}_{\mathtt{i}}, \mathcal{S}_{\mathtt{i}}, \mathcal{O}_{\mathtt{i}} \rangle$*, where*

1. $\mathtt{i}$ *is the identifier of the ontology* $\mathcal{W}_{\mathtt{i}}$*,*
2. $\mathcal{M}_{\mathtt{i}}$ *(called the* syntactic import box*) is the set of ontology identifiers which are syntactically imported by* $\mathcal{W}_{\mathtt{i}}$ *(by the primitive* owl:imports*),*
3. $\mathcal{S}_{\mathtt{i}}$ *(called the* semantic import box*) is a set of classes, properties and/or individuals symbols which are semantically imported by* $\mathcal{W}_{\mathtt{i}}$ *(by the new primitive* owlx:semanticImports*).*
4. $\mathcal{O}_{\mathtt{i}}$ *(called the* axiom box*) contains a set of class, property and individual axioms.*

*Example 2.* $\mathcal{W}_{\mathtt{ppl}} = \langle \mathtt{ppl}, \mathcal{M}_{\mathtt{ppl}}, \mathcal{S}_{\mathtt{ppl}}, \mathcal{O}_{\mathtt{ppl}} \rangle$ is an ontology, where $\mathcal{M}_{\mathtt{ppl}} = \emptyset$ (no syntatic imports), $\mathcal{S}_{\mathtt{ppl}} = \{\mathtt{sumo{:}Car}, \mathtt{sumo{:}Horse}, \mathtt{sumo{:}Transportation}, \mathtt{sumo{:}TransportationDevice}, \mathtt{sumo{:}}\mathit{capability}\}$ ($\mathcal{W}_{\mathtt{ppl}}$ semantically imports the classes $\mathtt{sumo{:}Car}$, $\mathtt{sumo{:}Horse}$, $\mathtt{sumo{:}Transportation}$ and $\mathtt{sumo{:}TransportationDevice}$, the

[1] Individual axioms are also called *facts*.

property sumo:*capability*). Furthermore, $\mathcal{O}_{\texttt{ppl}}$ contains the axioms:

$$\texttt{sumo:Horse} \sqsubseteq \exists \texttt{sumo:}\mathit{capability}.(\texttt{sumo:Transportation}) \quad (4)$$

$$\exists \texttt{sumo:}\mathit{capability}.(\texttt{sumo:Transportation}) \sqsubseteq \texttt{sumo:TransportationDevice} \quad (5)$$

$$\texttt{ppl:BelgianDraftHorse} \sqsubseteq \texttt{sumo:Horses} \quad (6)$$

$$\texttt{ppl:Ferrari} \sqsubseteq \texttt{sumo:Car} \quad (7)$$

They state that horses can be used for transportation (4), that those can be used for transportation are transportation devices (5), that Belgian Draft horses are horses (6), and that Ferrari is a type of car (7), which do not hold in SUMO. Intuitively the fact that $\mathcal{W}_{\texttt{ppl}}$ semantically imports sumo:Car means that $\mathcal{W}_{\texttt{ppl}}$ "demands" the semantics of sumo:Car from SUMO. In other words, an object of the domain of $\mathcal{W}_{\texttt{ppl}}$ cannot be a sumo:Car if it is not a sumo:Car in the domain of the SUMO ontology.

It is worth noting that, in the formal definition, a semantic import box is described as a set of classes, properties, and/or individuals. While in practice, this box can be specified either by explicitly listing its components, or, in a more intensional way, by, e.g., including all the sub-classes or all the individuals of some given classes. From the theoretical perspective this is clearly irrelevant, while it is very relevant from the usability point of view.

We end this section by clarifying the expected effects of semantic imports, which are all satisfied by the semantics of semantic imports to be provided in the next section (we assume that $\texttt{1:A}, \texttt{1:B}, \texttt{1:}R, \texttt{1:o}, \texttt{1:o2} \in \mathcal{S}_2$): (i) *Propagation of class hierarchy*: if $\mathcal{W}_1$ entails $\texttt{1:A} \sqsubseteq \texttt{1:B}$, then $\mathcal{W}_2$ entails $\texttt{1:A} \sqsubseteq \texttt{1:B}$. (ii) *Propagation of disjoint classes*: if $\mathcal{W}_1$ entails $\texttt{1:A} \sqsubseteq\neg\ \texttt{1:B}$, then $\mathcal{W}_2$ entails $\texttt{1:A} \sqsubseteq\neg\ \texttt{1:B}$. (iii) *Property Closure*: all the 1:$R$-successors and 1:$R$-predecessor of 1:o1 are also imported. (iv) *Propagation of (in)equality objects*: if $\mathcal{W}_1$ entails $\texttt{1:o1} \approx \texttt{1:o2}$ ($\texttt{1:o1} \not\approx \texttt{1:o2}$), then $\mathcal{W}_2$ entails $\texttt{1:o1} \approx \texttt{1:o2}$ ($\texttt{1:o1} \not\approx \texttt{1:o2}$).

## 3 Ontology Spaces

In order to give a formal semantics to semantic imports, we briefly introduce some basic notions.

**Definition 2 (Local language).** *Given an ontology $\mathcal{W}_{\texttt{i}}$, a* local class *w.r.t. $\mathcal{W}_{\texttt{i}}$ is a class name in $\mathcal{O}_{\texttt{i}}$ associated with the identifier* i. Local properties *and* local individuals *are defined analogously. The set of local classes, local properties and local individuals of $\mathcal{W}_{\texttt{i}}$ are denoted by $\mathbf{C}_{\texttt{i}}$, $\mathbf{R}_{\texttt{i}}$ and $\mathbf{I}_{\texttt{i}}$. The* local language *of $\mathcal{W}_{\texttt{i}}$, i.e., $\mathbf{L}_{\texttt{i}}$, is the union of them.*

For example, the local language of $\mathcal{W}_{\texttt{ppl}}$ is $\mathbf{L}_{\texttt{ppl}}$, which contains the two classes ppl:BelgianDraftHorse and ppl:Ferrari.

Now we introduce the concept of ontology spaces, in which we consider not only a single ontology but a set of ontologies. As described in the last section, OWL imports annotations do not guarantee the existence of imported ontologies.

Similarly, the occurrence of an URIrefs in an axiom does not impose requirement of the existence of an ontological resource associated with such a URIref. For instance, the existence of the axiom $\mathtt{car{:}Car} \sqsubseteq \mathtt{abc{:}ExpensiveGood}$ in the ontology $\mathcal{W}_{\mathtt{car}}$ does not guarantee the existence of the ontology $\mathcal{W}_{\mathtt{abc}}$; furthermore, even if $\mathcal{W}_{\mathtt{abc}}$ does exist, this axiom does not guarantee that $\mathtt{abc{:}ExpensiveGood}$ is in $\mathbf{L}_{\mathtt{abc}}$. When we consider an ontology space, we impose these requirements.

**Definition 3 (Ontology space).** *Let $I$ be a set of ontology identifiers and $\mathtt{j} \in I$ an ontology identifier. An* ontology space *on $I$ is a family of ontologies $\mathfrak{S} = \{\mathcal{W}_\mathtt{i}\}_{i \in I}$ such that (i) for each $\mathcal{W}_\mathtt{i} = \langle \mathtt{i}, \mathcal{M}_\mathtt{i}, \mathcal{S}_\mathtt{i}, \mathcal{O}_\mathtt{i} \rangle$, $\mathcal{M}_\mathtt{i} \subseteq I$, (ii) for each $\mathtt{j{:}x}$ occurring in $\mathcal{O}_\mathtt{i}$, $\mathtt{j{:}x} \in \mathcal{S}_\mathtt{i}$, and (iii) for each $\mathtt{j{:}x} \in \mathcal{S}_\mathtt{i}$, $\mathtt{j{:}x}$ appears in $\mathcal{O}_\mathtt{j}$, where $\mathtt{j} \in I$ and $\mathtt{j} \neq \mathtt{i}$.*

Note that, given an ontology space $\{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$, the above definition requires that: (i) all the imported ontologies of each ontology exist and are in $\{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$, and (ii) non-local classes, properties and individuals of each ontology in $\{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$ should be introduced by some ontologies in $\{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$. For example, $\mathfrak{S} = \langle \mathcal{W}_{\mathtt{sumo}}, \mathcal{W}_{\mathtt{ppl}} \rangle$ is an ontology space, where $\mathcal{W}_{\mathtt{ppl}}$ is defined in Example 2 and $\mathcal{W}_{\mathtt{sumo}} = \langle \mathtt{sumo}, \mathcal{M}_{\mathtt{sumo}}, \mathcal{S}_{\mathtt{sumo}}, \mathcal{O}_{\mathtt{sumo}} \rangle$ with $\mathcal{M}_{\mathtt{sumo}} = \emptyset$, $\mathcal{S}_{\mathtt{sumo}} = \emptyset$, (no ontology imports and no semantic import) and with $\mathcal{O}_{\mathtt{sumo}}$ equal to the set of axioms contained in SUMO.

Based on an ontology space, we introduce the concept of import closure and foreign languages. Intuitively speaking, an import closure of an ontology $\mathcal{W}_\mathtt{i}$ is the set of axioms considering its syntactic imports, while the foreign language of $\mathcal{W}_\mathtt{i}$ is the set of names of classes, properties and individuals in an ontology space that are not in its local language.

**Definition 4 (Import closure).** *Given an ontology space $\mathfrak{S} = \{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$, the* import closure *of an ontology $\mathcal{W}_\mathtt{i}$ w.r.t. $\mathfrak{S}$, written as $\mathcal{O}_\mathtt{i}^{\mathfrak{S}}$, is a set of class, property, and individual axioms recursively defined as follows:*

1. $\mathcal{O}_\mathtt{i} \subseteq \mathcal{O}_\mathtt{i}^{\mathfrak{S}}$;
2. *if* $\mathtt{j} \in \mathcal{M}_\mathtt{i}$ *then* $\mathcal{O}_\mathtt{j}^{\mathfrak{S}} \subseteq \mathcal{O}_\mathtt{i}^{\mathfrak{S}}$;
3. *nothing else is in* $\mathcal{O}_\mathtt{i}^{\mathfrak{S}}$.

**Definition 5 (Foreign language).** *Given an ontology space $\mathfrak{S} = \{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$, the* foreign language *of $\mathcal{W}_\mathtt{i}$ w.r.t. $\mathfrak{S}$, written as $\mathbf{F}_\mathtt{i}^{\mathfrak{S}}$, is the set of classes, properties and individuals of the form* $\mathtt{j{:}x}$ *($\mathtt{j} \neq \mathtt{i}$) which occur in $\mathcal{S}_\mathtt{i}$ or in $\mathcal{O}_\mathtt{i}^{\mathfrak{S}}$.*

Intuitively, the semantic for an ontology space is a family of interpretations one for each ontology, such that the axioms of the import closure of each ontology are satisfied, and there is an agreement between the interpretations on the semantics of semantically imported symbols, which will be formally defined below. It should be noted that the interpretation domains of the family of interpretations could be disjoint or overlapping with each other.

**Definition 6 (Interpretation of an ontology space).** *Let $\mathfrak{S} = \{\mathcal{W}_\mathtt{i}\}_{\mathtt{i} \in I}$ be an ontology space on $I$. An interpretation of $\mathfrak{S}$ is a family $\mathfrak{I} = \{\mathcal{I}_\mathtt{i}\}_{\mathtt{i} \in I}$, where each $\mathcal{I}_\mathtt{i}$, called the* local interpretation *of $\mathcal{W}_\mathtt{i}$, is an interpretation (including hole interpretations) of the local and foreign language of $\mathcal{W}_\mathtt{i}$, such that*

1. $\mathcal{I}_\mathtt{i} \models \mathcal{O}_\mathtt{i}^{\mathfrak{S}}$;
2. *if* $\mathtt{j{:}C}$ *is a class name in* $\mathcal{S}_\mathtt{i}$, *then* $(\mathtt{j{:}C})^{\mathcal{I}_\mathtt{i}} = (\mathtt{j{:}C})^{\mathcal{I}_\mathtt{j}} \cap \Delta^{\mathcal{I}_\mathtt{i}}$;
3. *if* $\mathtt{j{:}a}$ *is an individual name in* $\mathcal{S}_\mathtt{i}$, *then* $(\mathtt{j{:}a})^{\mathcal{I}_\mathtt{i}} = (\mathtt{j{:}a})^{\mathcal{I}_\mathtt{j}}$;
4. *if* $\mathtt{j{:}}R$ *is a property name in* $\mathcal{S}_\mathtt{i}$, *then*
   **forward closure** *for all* $d \in \Delta^{\mathcal{I}_\mathtt{i}} \cap \Delta^{\mathcal{I}_\mathtt{j}}$, *for all* $d' \in \Delta^{\mathcal{I}_\mathtt{j}}$, $\langle d, d'\rangle \in (\mathtt{j{:}}R)^{\mathcal{I}_\mathtt{i}}$ *iff* $\langle d, d'\rangle \in (\mathtt{j{:}}R)^{\mathcal{I}_\mathtt{j}}$;
   **backward closure** *for all* $d \in \Delta^{\mathcal{I}_\mathtt{i}} \cap \Delta^{\mathcal{I}_\mathtt{j}}$, *for all* $d' \in \Delta^{\mathcal{I}_\mathtt{j}}$, $\langle d', d\rangle \in (\mathtt{j{:}}R)^{\mathcal{I}_\mathtt{i}}$ *iff* $\langle d', d\rangle \in (\mathtt{j{:}}R)^{\mathcal{I}_\mathtt{j}}$.

*Example 3.* The interpretation $\mathfrak{I} = \{\mathcal{I}_{\mathtt{sumo}}, \mathcal{I}_{\mathtt{ppl}}\}$ described in Table 1 is an interpretation of $\mathfrak{S} = \langle \mathcal{W}_{\mathtt{sumo}}, \mathcal{W}_{\mathtt{ppl}}\rangle$.

| symbol | $\mathcal{I}_{\mathtt{sumo}}$ | $\mathcal{I}_{\mathtt{ppl}}$ |
|---|---|---|
| $\top$ | $\{c, f, h, k, d, t, \dots\}$ | $\{c, f, h, v, d, t\}$ |
| sumo:Car | $\{c, f\}$ | $\{c, f\}$ |
| sumo:Horse | $\{h, k\}$ | $\{h\}$ |
| sumo:TransportationDevice | $\{c, f, d\}$ | $\{c, f, h, v\}$ |
| sumo:Transportation | $\{t\}$ | $\{t\}$ |
| sumo:*capability* | $\left\{\begin{matrix}\langle c, t\rangle, \langle h, t\rangle \\ \langle f, t\rangle, \langle d, t\rangle\end{matrix}\right\}$ | $\left\{\begin{matrix}\langle c, t\rangle, \langle h, t\rangle \\ \langle f, t\rangle, \langle d, t\rangle\end{matrix}\right\}$ |
| ppl:BelgianDraftHorse | undef | $\{h\}$ |
| ppl:Ferrari | undef | $\{f\}$ |
| sumo:OrganicObject | $\{h, k\}$ | undef |
| sumo:Artifact | $\{c, f, d\}$ | undef |
| ... | ... | undef |

**Table 1.** An interpretation for the ontology space $\langle \mathcal{W}_{\mathtt{sumo}}, \mathcal{W}_{\mathtt{ppl}}\rangle$

It is worth noting the following points: (i) In a local interpretation of $\mathcal{W}_{\mathtt{ppl}}$, only symbols in local and foreign languages are considered. That is why local interpretations of $\mathcal{W}_{\mathtt{ppl}}$, such as $\mathcal{I}_{\mathtt{ppl}}$ above, do not have to provide an interpretation mapping for symbols (such as sumo:OrganicObject and sumo:Artifact) that are not in the local or foreign languages of $\mathcal{W}_{\mathtt{ppl}}$. (ii) Due to the backward closure, $\langle d, t\rangle$ is in $(\mathtt{sumo{:}}\textit{capability})^{\mathcal{I}_{\mathtt{ppl}}}$.

**Definition 7 (Logical consequence).** *Let* $\mathfrak{S} = \{\mathcal{W}_\mathtt{i}\}_{\mathtt{i}\in I}$ *be an ontology space on* $I$. *An OWL DL axiom* $\alpha$ *is a logical consequence of* $\mathfrak{S}$ *in* $\mathcal{W}_\mathtt{j}$, *written* $\mathfrak{S} \models_\mathtt{j} \alpha$, *iff, for every interpretation* $\mathfrak{I} = \{\mathcal{I}_\mathtt{j}\}_{\mathtt{j}\in I}$ *of* $\mathfrak{S}$, $\mathcal{I}_\mathtt{j} \models \alpha$.

## 4 Reasoning in Simple Ontology Spaces

Note that the framework proposed in the previous sections is very general; e.g., there can be an arbitrary (finite) number of ontologies in an ontology space and any ontology involved can semantically vocabulary from any other ontologies in the ontology space. As a first step, in the rest of this paper we only consider a

special kind of ontology spaces, called *simple ontology spaces*. Informally speaking, a simple ontology space consists of a pair of ontologies $\mathcal{W}_\mathtt{i}$ and $\mathcal{W}_\mathtt{j}$, where there is no syntactic imports, and $\mathcal{W}_\mathtt{j}$ semantically imports some vocabulary from $\mathcal{W}_\mathtt{i}$, and $\mathcal{W}_\mathtt{i}$ does not (semantically) import anything.

**Definition 8.** *A* simple ontology space *is of the form* $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i} \rangle$*, where* $\mathcal{M}_\mathtt{j} = \mathcal{M}_\mathtt{i} = \emptyset$ *(no syntactic imports) and* $\mathcal{S}_\mathtt{j} = \emptyset$*.*

What is the effect on the logical consequence of semantic import? And how such effect can be computed? In what follows, we will answer these questions by characterizing the effect of class constructors on transferring knowledge (in the form of class subsumptions) from a distant ontology to a target ontology. Let us illustrate our characterization by revisiting our example about $\langle \mathcal{W}_\mathtt{sumo}, \mathcal{W}_\mathtt{ppl} \rangle$.

*Example 4.* In our example, the subsumption (3) is propagated from $\mathcal{W}_\mathtt{sumo}$ to $\mathcal{W}_\mathtt{ppl}$. Intuitively this is because $\mathcal{W}_\mathtt{ppl}$ semantically agrees with $\mathcal{W}_\mathtt{sumo}$ on the meanings of all the terms of this subsumption. Instead, the subsumption $\mathtt{sumo{:}Horses} \sqsubseteq \neg \mathtt{sumo{:}TransportationDevice}$ which holds in $\mathcal{W}_\mathtt{sumo}$, does not hold in $\mathcal{W}_\mathtt{ppl}$ because $\mathcal{W}_\mathtt{ppl}$ disagrees with $\mathcal{W}_\mathtt{sumo}$ on the meaning of sumo:TransportationDevice (see the different interpretations in table 1).

Generalizing the previous example, one might falsely infer that, if the ontology $\mathcal{W}_\mathtt{i}$ semantically imports a set of symbols $\mathcal{S}_\mathtt{i}$ from the ontology $\mathcal{W}_\mathtt{j}$, then all the class subsumptions in the language of $\mathcal{S}_j$ that hold in $\mathcal{W}_\mathtt{j}$ are propagated to $\mathcal{W}_\mathtt{i}$. This is true with an exception, due to the effect of coverage axioms. Let us explain this point with an example.

*Example 5 (Closure Axioms are not Propagated).* [2] Suppose that the ontology $\mathcal{W}_\mathtt{xyz}$ contains the axiom

$$\neg \mathtt{xyz{:}Fast} \sqsubseteq \mathtt{xyz{:}Slow}. \tag{8}$$

Such an axioms partitions the domain of $\mathcal{W}_\mathtt{xyz}$ into two sets, the fast objects and the slow objects. Even if $\mathcal{W}_\mathtt{ppl}$ semantically imports all vocabulary from $\mathcal{W}_\mathtt{xyz}$, axiom (8) is not propagated form $\mathcal{W}_\mathtt{xyz}$ to $\mathcal{W}_\mathtt{ppl}$. The reason is that the domain of ontology $\mathcal{W}_\mathtt{ppl}$ might contain objects which are not in the domain of $\mathcal{W}_\mathtt{xyz}$, and these objects could be neither fast nor slow.

### 4.1 Characterizing Logical Consequence in Simple Ontology Spaces

In Sections 2 and 3, we mainly considered subsumption relations among semantically imported named classes. Now we introduce related class descriptions.

**Definition 9 (Semantically imported classes descriptions).** *The* semantically imported class descriptions*, or simply semantically imported classes, of an ontology* $\mathcal{W}_\mathtt{i}$ *are* $\mathcal{SHOIN}$*-class descriptions that can be constructed from classes, properties and individuals contained in* $\mathcal{S}_\mathtt{i}$*. We say that* $\mathcal{W}_\mathtt{i}$ *semantically imports a description* $X$ *from* $\mathcal{W}_\mathtt{j}$ *if* $X$ *is constructed by the local language of* $\mathcal{W}_\mathtt{j}$*.*

[2] B. Cuenca-Grau presented this example at the DL-2006 workshop.

**Definition 10.** *Given $\mathcal{S}$ an semantic import box, we say that a class description $C$ (property description $R$) is $\mathcal{S}$-related if all the vocabulary in $C$ ($R$, respectively) are in $\mathcal{S}$.*

For example, given the semantic box $\mathcal{S}_{\texttt{ppl}} = \{\texttt{sumo:Car}, \texttt{sumo:Horse}, \texttt{sumo:Transportation}, \texttt{sumo:TransportationDevice}$ (see Example 2), $\texttt{sumo:Car}$, $\texttt{sumo:}$*capability*, and $\exists(\texttt{sumo:}$*capability*$).(\texttt{sumo:Transportation})$ are $\mathcal{S}_{\texttt{ppl}}$-related.

On the base of the intuition given in the example presented at the beginning of Section 4, in the following we will provide a precise class characterization on which subsumptions propagate as a consequence of semantic imports in simple ontology spaces. Formally speaking, we accomplish this by using the notion of local/non-local classes first proposed by Cuenca-Grau et al. [5]. In this paper, we call them positive/negative classes.

**Definition 11 (Polarity).** *The polarity of a semantically imported $\mathcal{SHOIN}$-classes is defined as follows (let $A$ and $B$ be classes, $\texttt{a}$ an individual and $R$ an object property): any atomic class is positive, $\bot$ is positive, $\top$ is negative, $\{\texttt{a}\}$ is positive, $A \sqcap B$ is positive if either $A$ or $B$ are positive, $A \sqcup B$ is positive if both $A$ and $B$ are positive, $\neg A$ is positive iff $A$ is negative, $\exists R.A$ is positive, $\forall R.A$ is negative, $(\geq 0)R$ is negative, $(\geq n)R$ with $n \geq 1$ is positive, $(\leq n)R$ is negative, every class which is not positive is negative.*

*Property 1.* For every positive concept semantically imported from $\mathcal{W}_{\texttt{j}}$, $C^{\mathcal{I}_{\texttt{i}}} = C^{\mathcal{I}_{\texttt{j}}} \cap \Delta^{\mathcal{I}_{\texttt{i}}}$. For every negative concept semantically imported from $\mathcal{W}_{\texttt{j}}$, $C^{\mathcal{I}_{\texttt{i}}} = (C^{\mathcal{I}_{\texttt{j}}} \cap \Delta^{\mathcal{I}_{\texttt{i}}}) \cup \Delta^{\mathcal{I}_{\texttt{i}} - \mathcal{I}_{\texttt{j}}}$ (where $\Delta^{\mathcal{I}_{\texttt{i}} - \mathcal{I}_{\texttt{j}}}$ is a shortcut of $\Delta^{\mathcal{I}_{\texttt{i}}} \setminus \Delta^{\mathcal{I}_{\texttt{j}}}$).

The Property 1 is the same as the property of local/non-local classes presented in the proof of Theorem 1 of [5], in which the authors have proved: (1) for each local (positive) class description $C$, $C^{\mathcal{I}_{\texttt{i}}} = C^{\mathcal{I}_{\texttt{j}}}$, and (2) for each non-local (negative) class description $C$, $C^{\mathcal{I}_{\texttt{i}}} = C^{\mathcal{I}_{\texttt{j}}} \cup \Delta^{\mathcal{I}_{\texttt{i}} - \mathcal{I}_{\texttt{j}}}$. Note that the property from [5] is based on a different kind of cross-domain semantics, namely domain expansion[3], rather than that of semantic imports presented in Definition 6. It should be pointed out that, in order to maintain properties like Property 1, the kind of cross-domain semantics should be carefully selected. For example, if we replace the semantics of semantically imported properties (defined in Definition 6) with $R^{\mathcal{I}_{\texttt{i}}} = R^{\mathcal{I}_{\texttt{j}}} \cap (\Delta^{\mathcal{I}_{\texttt{i}}} \times \Delta^{\mathcal{I}_{\texttt{i}}})$ (as in C-OWL [3]), the Property 1 no longer holds.

Based on the notion of positive/negative (i.e. local/non-local) classes and the above property, the following Definition 12, Theorem 1 and 2 and can be seen as counterparts (under the semantics of semantic imports) of the Theorem 2 of [5], which identifies safe axioms, i.e. the kind of axioms that are satisfies by any domain expansions of their interpretations.

[3] According to [5], a domain expansion $\mathcal{I}_{\texttt{i}}$ of an interpretation $\mathcal{I}_{\texttt{j}}$ is an interpretation such that: (1) $\Delta^{\mathcal{I}_{\texttt{i}}} = \Delta^{\mathcal{I}_{\texttt{j}}} \cup \Phi$, with $\Phi$ a non-empty set disjoint with $\Delta^{\mathcal{I}_{\texttt{j}}}$, (2) $A^{\mathcal{I}_{\texttt{i}}} = A^{\mathcal{I}_{\texttt{j}}}$ for each class name, and (3) $R^{\mathcal{I}_{\texttt{i}}} = R^{\mathcal{I}_{\texttt{j}}}$ for each property name.

**Definition 12.** *Let $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i} \rangle$ be a simple ontology space. The* subsumptions transfer *from $\mathcal{W}_\mathtt{j}$ to $\mathcal{W}_\mathtt{i}$, written as $\mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j})$, is the set of subsumption statements of the form $X \sqsubseteq Y$, where $X$ and $Y$ are $\mathcal{S}_\mathtt{i}$-related class descriptions, $\mathcal{W}_\mathtt{j} \models X \sqsubseteq Y$ and, either $X$ is positive or $Y$ is negative.*

Intuitively, the subsumption transfer from $\mathcal{W}_\mathtt{i}$ to $\mathcal{W}_\mathtt{j}$ is the set of all the possible subsumptions in $\mathcal{W}_\mathtt{i}$ that are semantically propagated in $\mathcal{W}_\mathtt{j}$. The following theorem shows that the subsumption transfer defined above can be used for a sound and complete characterization of logical consequences in simple ontology spaces.

**Theorem 1 (Soundness).** *Let $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i} \rangle$ be a simple ontology space. If $X$ and $Y$ are $\mathcal{S}_\mathtt{i}$-related class descriptions, then*

$$\mathcal{W}_\mathtt{i} \cup \mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j}) \models X \sqsubseteq Y \quad \Longrightarrow \quad \mathfrak{S} \models_\mathtt{i} X \sqsubseteq Y.$$

**Theorem 2 (Completeness).** *Let $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i} \rangle$ be a simple ontology space. If $X$ and $Y$ are $\mathcal{S}_\mathtt{i}$-related class descriptions, then*

$$\mathcal{W}_\mathtt{i} \cup \mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j}) \models X \sqsubseteq Y \quad \Longleftarrow \quad \mathfrak{S} \models_\mathtt{i} X \sqsubseteq Y.$$

*Proof.* Suppose that $\mathcal{O}_\mathtt{i} \cup \mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j}) \not\models X \sqsubseteq Y$ and let us build a model for $\mathfrak{S}$ such that $\mathfrak{S} \not\models_\mathtt{i} X \sqsubseteq Y$. If $\mathcal{O}_\mathtt{i} \cup \mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j}) \not\models X \sqsubseteq Y$ then let $\mathcal{I}_\mathtt{i}$ be a model of $\mathcal{O}_\mathtt{i} \cup \mathcal{S}_{\mathtt{ji}}(\mathcal{W}_\mathtt{j})$ such that $X^{\mathcal{I}_\mathtt{i}} \not\subseteq Y^{\mathcal{I}_\mathtt{i}}$.

Suppose that $\mathcal{O}_\mathtt{j}$ is unsatisfiable, then for all concept $C \in \mathcal{S}_\mathtt{i}$, $\mathcal{O}_\mathtt{j} \models C \sqsubseteq \bot$; for all nominal $\{x\} \in \mathcal{S}_\mathtt{i} i$, $\mathcal{O}_\mathtt{j} \models \{x\} \sqsubseteq \bot$; for all role $R \in \mathcal{S}_\mathtt{i}$, $\mathcal{O}_\mathtt{j} \models \top \sqsubseteq \forall R.\bot$. In all the cases we have that the corresponding subsumptions belongs to $\mathcal{S}_{\mathtt{ji}}(\mathcal{O}_\mathtt{j})$. The fact that $\mathcal{I}_\mathtt{i} \models \mathcal{S}_{\mathtt{ji}}(\mathcal{O}_\mathtt{j})$, implies that $\mathcal{S}_\mathtt{i}$ does not contains nominals (otherwise $\mathcal{I}_\mathtt{i} \models \{x\} \sqsubseteq \bot$, which is not possible), and for all $C$ and $R$ in $\mathcal{S}_\mathtt{i}$, they are interpreted by $\mathcal{I}_\mathtt{i}$ in the empty set. This implies that the interpretation $\langle \mathcal{I}_\mathtt{j}, \mathcal{I}_\mathtt{i} \rangle$ with $\mathcal{I}_\mathtt{j}$ being a hole interpretation, satisfies $\mathfrak{S}$.

Now let us consider the case were $\mathcal{O}_\mathtt{j}$ is consistent. Let $\Delta^* \subseteq \Delta^{\mathcal{I}_\mathtt{i}}$ be the set

$$\left\{ x \in \Delta^{\mathcal{I}_\mathtt{i}} \;\middle|\; \begin{array}{l} x \in B^{\mathcal{I}_\mathtt{i}} \text{ for some } B \in \mathcal{S}_\mathtt{i} \text{ or} \\ x = \{a\}^{\mathcal{I}_\mathtt{i}} \text{ for some } a \in \mathcal{S}_\mathtt{i} \text{ or} \\ \langle x, y \rangle \in R^{\mathcal{I}_\mathtt{i}} \text{ for some } y \text{ and } R \in \mathcal{S}_\mathtt{i} \text{ or} \\ \langle y, x \rangle \in R^{\mathcal{I}_\mathtt{i}} \text{ for some } y \text{ and } R \in \mathcal{S}_\mathtt{i} \end{array} \right\}$$

If $\Delta^* = \emptyset$ then, the interpretation $\langle \mathcal{I}_\mathtt{j}, \mathcal{I}_\mathtt{i} \rangle$ where $\mathcal{I}_\mathtt{i}$ is the hole interpretation is a model of $\mathfrak{S}$ which does not satisfy $X \sqsubseteq Y$. Suppose, therefore that $\Delta^*$ contains at least an element $x_0$.

Let $\mathcal{W}_\mathtt{i}^*$ be obtained by extending the semantic import $\mathcal{S}_\mathtt{i}$ of $\mathcal{W}_\mathtt{i}$ with the set of constants in $\Delta^*$ (we can suppose with no loss of generality that $\Delta^*$ is disjoint from the language of $\mathcal{W}_\mathtt{i}$), and with a new role $R_0$. a new nominal $\{x_0\}$. Let $\mathcal{I}_\mathtt{i}^*$ be the extension of $\mathcal{I}_\mathtt{i}$ such that $x^{\mathcal{I}_\mathtt{i}^*} = x$ for all $x \in \Delta^*$ and $x_0^{\mathcal{I}_\mathtt{i}^*}$ be any value in $\Delta^*$. Clearly $\mathcal{I}_\mathtt{i}^*$ is an interpretation of $\mathcal{W}_\mathtt{i}^*$ that satisfies $\mathcal{O}_\mathtt{i}^* = \mathcal{O}_\mathtt{i}$.

Let $\mathcal{W}_\mathtt{j}^*$ be obtained by extending $\mathcal{O}_\mathtt{j}$ of $\mathcal{W}_\mathtt{j}$, with the following set of axioms. We denote the resulting set with $\mathcal{O}_\mathtt{j}^*$.

1. $\{x_0\} \sqsubseteq \exists R_0 \{x\}$ for all $x \in \Delta^*$
2. $\{x\} \sqsubseteq \neg \{y\}$ for every $x \neq y \in \Delta^*$.
3. $\{x\} \sqsubseteq B$, if $B \in \mathcal{S}_\mathtt{i}$ and $x \in B^{\mathcal{I}_\mathtt{i}}$;
4. $\{x\} \sqsubseteq \exists R.\{y\}$, if $R \in \mathcal{S}_\mathtt{i}$ and $\langle x, y\rangle \in R^{\mathcal{I}_\mathtt{i}}$.

Consider the ontology space $\mathfrak{S}^* = \langle \mathcal{W}^*_\mathtt{j}, \mathcal{W}^*_\mathtt{i}\rangle$, If $\mathcal{O}^*_\mathtt{j}$ is consistent, then there is an interpretation $\mathcal{I}^*_\mathtt{j}$ for $\mathcal{O}^*_\mathtt{j}$ in which $x^{\mathcal{I}^*_\mathtt{j}} = x$. Let $\mathcal{I}_\mathtt{j}$ be the restriction of $\mathcal{I}^*_\mathtt{j}$ on the language of $\mathcal{W}_\mathtt{i}$, then by construction $\langle \mathcal{I}_\mathtt{j}, \mathcal{I}_\mathtt{i}\rangle$ is an interpretation for $\mathfrak{S}$.

If $\mathcal{O}^*_\mathtt{j}$ is inconsistent, then, since $\mathcal{O}_\mathtt{j}$ is consistent, there is a finite subset $\mathcal{O}^0_\mathtt{j}$ of the axioms 1–4 added to $\mathcal{O}_\mathtt{j}$ that make it inconsistent. Let $C^*$ be the concept

$$\bigsqcup_{X \sqsubseteq Y \in \mathcal{O}^0_\mathtt{j}} X \sqcap \neg Y.$$

We have that $\mathcal{O}_\mathtt{j} \cup C^* \sqsubseteq \bot$ is inconsistent. This implies that

$$\mathcal{O}_\mathtt{j} \models \{x_0\} \sqsubseteq \exists R_0 C^*.$$

Notice that both $\{x_0\}$ and $\exists R_0 C^*$ are positive classes, which implies that $\{x_0\} \sqsubseteq \exists R_0 C^*$ belongs to $\mathcal{S}_{\mathtt{ij}}(\mathcal{W}_\mathtt{i})$ which generates a contradiction.

### 4.2 Distributed Tableaux for Simple Ontology Spaces

In this section, we consider ontologies represented as $\mathcal{ALC}$ [13] TBoxes (which consists of only class axioms) and describe a distributed tableaux algorithm to compute concept consistency in simple ontology spaces.

Tableaux algorithms (first by [14]) are very useful to solve class satisfiability problem. They test the satisfiability of a class $E$[4] by trying to construct an interpretation for $E$, which is represented by a *completion tree* $\mathcal{T}$: nodes in $\mathcal{T}$ represent individuals in the model; each node $y$ is labeled with $\mathcal{L}(y)$ and each edge $\langle y, z\rangle$ is labeled with $\mathcal{L}(\langle y, z\rangle)$. A tableaux algorithm starts from an labelled initial tree (usually simply a root node), and is expanded by repeatedly applying the completion rules. The algorithm terminates either when $\mathcal{T}$ is *complete* (no further completion rules can be applied) or when an obvious contradiction, or *clash*, has been revealed.

Formally, a completion tree is a tuple $\mathcal{T} = \langle x, N, E, \mathcal{L}\rangle$, where $x$ is the root of $\mathcal{T}$, $N$ and $E$ are the sets of nodes and edges, respectively, of $\mathcal{T}$, and $\mathcal{L}$ a function that maps each node $y$ (each edge $\langle y, z\rangle$) in $\mathcal{T}$ to its label $\mathcal{L}(y)$ ($\mathcal{L}(\langle y, z\rangle)$, respectively). Let $\mathcal{T}$ be a completion tree, $\mathcal{S}$ a semantic import box. We say that a completion tree $\mathcal{T}$ is $\mathcal{S}$-related if there exist some $\mathcal{S}$-related class or property descriptions in the labels of all edges and all non-leaf nodes of $\mathcal{T}$. In what follows, we define the main operation on completion trees we need in our algorithm.

**Definition 13 (Projection of Completion Tree).** *Let $\mathcal{S}$ be a semantic import box and $\mathcal{T}$ an $\mathcal{S}$-related clash-free completion tree with root $x_0$. The projection of $\mathcal{T}$ w.r.t. $\mathcal{S}$, denoted as $\pi_\mathcal{S}(\mathcal{T})$, is a completion tree which*

[4] Here we assume $E$ is in negation normal form; i.e., negation is only applied to class names.

1. *has the same root* $x_0$,
2. *contains the exact set of nodes and edges as* $\mathcal{T}$, *and*
3. *for each label* $\mathcal{L}(x')$ $(\mathcal{L}(\langle x', y'\rangle))$ *of a node* $x$ *(an edge* $\langle x', y'\rangle$, *respectively) is the subset of the label* $\mathcal{L}(x)$ $(\mathcal{L}(\langle x, y\rangle)$, *respectively) of the corresponding node* $x$ *(edge* $\langle x, y\rangle$, *respectively). If all the* $\mathcal{S}$*-related class descriptions in* $\mathcal{L}(x)$ *is negative,* $\mathcal{L}(x') = \emptyset$; *otherwise,* $\mathcal{L}(x')$ *contains all the* $\mathcal{S}$*-related class descriptions in* $\mathcal{L}(x)$ . $\mathcal{L}(\langle x', y'\rangle)$ *contains all the* $\mathcal{S}$*-related property descriptions in* $\mathcal{L}(\langle x, y\rangle)$.

According to Theorem 1 and 2, we do not project any $\mathcal{S}$-related class descriptions in $\mathcal{L}(x)$ to $\mathcal{L}(x')$ if *all* the $\mathcal{S}$-related class descriptions in $\mathcal{L}(x)$ are negative.

For a simple ontology space $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i}\rangle$, the procedure S-Tab(i, $E$) verifies the satisfiability of an $\mathcal{ALC}$ class description $E$ in ontology $\mathcal{W}_\mathtt{i}$. The procedure Tab($\mathcal{O}_\mathtt{x}, \mathcal{T}$) is a (local) tableaux algorithm to expand $\mathcal{T}$ w.r.t. a local ontology $\mathcal{O}_\mathtt{x}$. Tab($\mathcal{O}_\mathtt{x}, \mathcal{T}$) has two distinguished features that we need: (i) it takes not only a single node but an arbitrary initial completion tree (see line 8 in Algorithm 1), (ii) the algorithm can cache reasoning states (see line 11 in Algorithm 1).

**Algorithm 1**: S-Tab(i, $E$)

1: let $\mathcal{T}$ :=Tab($\mathcal{O}_\mathtt{i}, \langle x_0, \{x_0\}, \emptyset, \{\mathcal{L}(x_0) = \{E\}\}\rangle$) //local expansion w.r.t. $\mathcal{O}_\mathtt{i}$
2: **repeat**
3: **if** $\mathcal{T}$ has a clash **then**
4: **return** *unsatisfiable*
5: **end if**
6: let $\mathcal{T}_1, \ldots, \mathcal{T}_n$ be the maximal $\mathcal{S}_\mathtt{i}$-related sub-trees of $\mathcal{T}$ with roots $x_1, ..., x_n$, repectively
7: $\mathcal{T}_1' := \pi_{\mathcal{S}_\mathtt{i}}(\mathcal{T}_1), \ldots, \mathcal{T}_n' := \pi_{\mathcal{S}_\mathtt{i}}(\mathcal{T}_n)$ //sub-trees projection
8: $\mathcal{T}_1' :=$ Tab($\mathcal{O}_\mathtt{j}, \mathcal{T}_1'), \ldots, \mathcal{T}_n' :=$ Tab($\mathcal{O}_\mathtt{j}, \mathcal{T}_n'$) //local expansion w.r.t. $\mathcal{O}_\mathtt{j}$
9: **if** any of $\mathcal{T}_1', \ldots, \mathcal{T}_n'$ has a clash **then**
10: **if** $\mathcal{T}$ is backtrackable **then**
11: $\mathcal{T}$ :=Tab($\mathcal{O}_\mathtt{i}, \mathcal{T}, backtrack$) //backtrack and expand
12: **else**
13: **return** *unsatisfiable*
14: **end if**
15: **else**
16: **return** *satisfiable*
17: **end if**
18: **until** false

The algorithm needs some explanation and clarification. $\mathcal{T}$ is initialised with a root $x_0$ with $\mathcal{L}(x_0) = \{E\}$, and is expanded by local completion rules w.r.t. $\mathcal{O}_\mathtt{i}$ (line 1). As $\mathcal{T}$ might not be $\mathcal{S}_\mathtt{i}$-related, maximal $\mathcal{S}_\mathtt{i}$-related sub-trees then should be projected and expanded by local completion rules w.r.t. $\mathcal{O}_\mathtt{j}$ (lines 6-8). If any of the projected sub-tree has a clash, $\mathcal{T}$ needs to be backtracked, expanded and start the checking all over again. In what follows, let us use a simple example to illustrate its main idea of the algorithm.

*Example 6.* Given the following ontology space $\mathfrak{S} = \langle \mathcal{W}_\mathtt{j}, \mathcal{W}_\mathtt{i}\rangle$, where $\mathcal{W}_\mathtt{j}$ and $\mathcal{W}_\mathtt{i}$ are defined as follows: $\mathcal{W}_\mathtt{j} = \langle \mathtt{j}, \mathcal{M}_\mathtt{j} = \emptyset, \mathcal{S}_\mathtt{j} = \emptyset, \mathcal{O}_\mathtt{j} = \{\mathtt{j}{:}\mathsf{A} \sqsubseteq \forall \mathtt{j}{:}R.(\mathtt{j}{:}\mathsf{C}),$

$\mathtt{j}{:}\mathsf{C} \sqsubseteq \mathtt{j}{:}\mathsf{B}\}\rangle$ and $\mathcal{W}_\mathtt{i} = \langle \mathtt{i}, \mathcal{M}_\mathtt{i} = \emptyset, \mathcal{S}_\mathtt{i} = \{\mathtt{j}{:}\mathsf{A}, \mathtt{j}{:}\mathsf{B}, \mathtt{j}{:}R\}, \mathcal{O}_\mathtt{i} = \emptyset\rangle$. We want to check the satisfiability of the following class $\mathtt{j}{:}\mathsf{A} \sqcap \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B})$ in $\mathcal{W}_\mathtt{i}$ w.r.t. $\mathfrak{S}$.

We use $\mathrm{Tab}(\mathcal{O}_\mathtt{i}, \langle x_0, \{x_0\}, \emptyset, \{\mathcal{L}(x_0) = \{\mathtt{j}{:}\mathsf{A} \sqcap \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B})\}\}\rangle)$ to build a clash-free completed completion tree $\mathcal{T}$. It is obvious that $\mathcal{T}$ is $\mathcal{S}_\mathtt{i}$-related, so there is only one maximal $\mathcal{S}_\mathtt{i}$-related sub-tree of $\mathcal{T}$, which is itself. Therefore, $\mathcal{T}'$ consists of two connected nodes $x'_0$ and $x'_1$, where $x'_0$ is labelled with $\mathcal{L}(x'_0) = \{\mathtt{j}{:}\mathsf{A} \sqcap \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B}), \mathtt{j}{:}\mathsf{A}, \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B})\}$, $x'_1$ is labelled with $\mathcal{L}(x'_1) = \{\neg \mathtt{j}{:}\mathsf{B}\}$ and the edge $\langle x'_0, x'_1\rangle$ is labelled with $\mathcal{L}(\langle x'_0, x'_1\rangle) = \mathtt{j}{:}R$. Then we expand $\mathcal{T}'$ with mega-constraints:[5] we first expand $\mathcal{L}(x'_0)$ as $\{\mathtt{j}{:}\mathsf{A} \sqcap \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B}), \mathtt{j}{:}\mathsf{A}, \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B}), \neg \mathtt{j}{:}\mathsf{A} \sqcup \forall \mathtt{1}{:}R.(\mathtt{j}{:}\mathsf{C}), \neg \mathtt{j}{:}\mathsf{C} \sqcup \mathtt{j}{:}\mathsf{B}\}$; for the disjunction $\neg \mathtt{j}{:}\mathsf{A} \sqcup \forall \mathtt{j}{:}R.(\mathtt{j}{:}\mathsf{C})$ in $\mathcal{L}(x'_0)$, we have to choose $\forall \mathtt{j}{:}R.(\mathtt{j}{:}\mathsf{C})$ since $\mathtt{j}{:}\mathsf{A} \in \mathcal{L}(x'_0)$; then we expand $\mathcal{L}(x'_1)$ and add $C$ in it, so $\mathcal{L}(x'_1) = \{\neg \mathtt{j}{:}\mathsf{B}, \neg \mathtt{j}{:}\mathsf{A} \sqcup \forall \mathtt{j}{:}R.(\mathtt{j}{:}\mathsf{C}), \neg \mathtt{j}{:}\mathsf{C} \sqcup \mathtt{j}{:}\mathsf{B}, C\}$; obviously there is a clash in $\mathcal{L}(x'_1)$ because $\neg \mathtt{j}{:}\mathsf{C} \sqcup \mathtt{j}{:}\mathsf{B}, \mathtt{j}{:}\mathsf{C}$ and $\neg \mathtt{j}{:}\mathsf{B}$ are in $\mathcal{L}(x'_1)$. Hence, $\mathcal{T}'$ and $\mathcal{T}$ are not complete and $\mathtt{j}{:}\mathsf{A} \sqcap \exists \mathtt{j}{:}R.(\neg \mathtt{j}{:}\mathsf{B})$ is unsatisfiable in $\mathcal{W}_\mathtt{i}$ w.r.t. $\mathfrak{S}$.

To prove the algorithm correct, we need to show that: (1) The algorithm always terminates. (2) The algorithm returns unsatisfiable iff $\mathfrak{S} \not\models_\mathtt{i} E$. (3) The algorithm returns satisfiable iff $\mathfrak{S} \models_\mathtt{i} E$.

Due to limited space, we only provide a brief discussion. For (1), observe that Algorithm 1 does not change $\mathcal{T}$ at all and simply invokes backtrack when there exists an external clash. Algorithm 1 terminates because $\mathrm{Tab}(\mathcal{O}, E)$ terminates. For (2), observe that $\mathfrak{S} \not\models_\mathtt{i} E$ either because $\mathcal{O}_\mathtt{i} \not\models E$ (this would be detected by lines 1-3) or there exists some $X \sqsubseteq Y \in \mathcal{S}_\mathtt{ji}$ s.t. $\neg X \sqcup Y$ is disjoint with a $\mathcal{S}_\mathtt{i}$-related sub-class of $E$ (this would be detected by lines 7-9). For (3), it can be shown that we can construct an interpretation of $E$ whenever we have a complete $\mathcal{T}$, and vice versa.

## 5 Related Work

Ontology reuse has been an important topic [19, 18, 8]. Uschold et al [18] conclude that reusing an ontology requires significant effort on considering the context and intended usage of both the source and target ontologies as well as the specific task of the target application. This is consistent with our observation that users need to be able to customise their partial reuse of ontologies.

The semantic for ontology import is an evolution of the semantics C-OWL language [3]. The main difference concerns the interpretation of foreign relations. In C-OWL $(\mathtt{i}{:}R)^{\mathcal{I}_\mathtt{j}} = (\mathtt{i}{:}R)^{\mathcal{I}_\mathtt{i}} \cap \Delta^{\mathcal{I}_\mathtt{j}} \times \Delta^{\mathcal{I}_\mathtt{j}}$ while in semantic import we imposed the forward and backtrack closure. This update was necessary to allow the propagation of properties on relations.

Our approach is also related to existing work on reasoning with distributed Description Logics [15] and distributed First Order Logics [6]. In particular it is still an open question if semantic imports can be obtained by suitably mapping a concept (role or individual) on its native ontology with the occurrence of the

[5] In $\mathrm{Tab}(\mathcal{O}_\mathtt{j}, \mathcal{T}')$, meta-constraints $M(\mathcal{T}') = \{A_1, \ldots, A_n\}$ is merged into the label of each node in $\mathcal{T}'$, where $A_i = \neg C_i \sqcup D_i$ for each $C_i \sqsubseteq D_i$ in $\mathcal{T}'$ $(1 \leq i \leq n)$.

concept in the importer ontology. The conjecture is that semantically importing $\mathtt{i{:}C}$ into the ontology $\mathcal{W}_\mathtt{j}$ corresponds to adding the mappings $\mathtt{i{:}C} \xrightarrow{\equiv} \mathtt{j{:}C}$, $\neg\mathtt{i{:}C} \xrightarrow{\equiv} \neg\mathtt{j{:}C}$. [6].

Ontology modularisation [16, 4, 11, 5, 1] is an active topic in the field of ontology reuse. StuckenSchmidt and Klein [16] propose a structure-based approach for partitioning large class hierarchies. Cuenca-Grau et al. [4] present a partitioning algorithm, based on e-connections [7]. The partitioning of an ontology is automatic, although the modularisation is not always possible. Later, Cuenca-Grau et al. [5] propose the notion of safe ontologies, which are ontologies that can be modularised. The idea of safe ontologies is based on the notion of local/non-local classes (which we call positive/negative classes); see Section 4.1 for more details. Besides using different kind of cross-domain semantics, the main differences between our approach and Cuenca-Grau et al.'s approach include: (1) the reusable units in our approach are some vocabulary (such as classes or properties) rather than axioms; (2) our approach allows users to agree or disagree with some subsumption relations in distant ontologies, while modules of ontologies are required to preserve subsumptions among named classes; (3) unlike the modularisation operator, the semantic imports operator can be used between any two ontologies in an ontology space.

## 6 Discussion

How to support ontology reuse is becoming more and more important. While the "copy-and-paste" approach suffers a number of problems, this paper proposes a new import primitive, called *semantic import*, to facilitate *partial ontology reuse*. In this paper, we have identified four semantic requirements for semantic import: propagation of class hierarchy, disjoint classes, property closure and (in)equality objects. Secondly, we have investigated the logical properties of semantic imports in simple ontology spaces. Based on the notion of positive/negative classes first proposed by Cuenca-Grau et al. [5], several categories of subsumption relations (knowledge) class descriptions over the agreed vocabulary have been shown to be transferrable form distant ontologies to the target ontology. Finally, we have presented a distributed algorithm for TBox reasoning of semantic import of $\mathcal{ALC}$ ontologies in simple ontology spaces.

An important "side effect" of introducing semantic import into ontology languages such as OWL DL is to open a door to support the intended usage of URI references, i.e. identifying resources. In OWL DL, the same individual URIref can be interpreted as different objects in different ontologies, where URI references are not really playing a role on identifying resources. In the presence of semantic import, however, the semantic of ontology space guarantees that an semantically imported individual is interpreted as the same object in multiple ontologies. This suggests that semantic import could play a very fundamental role in Web ontologies under the unique identification assumption.

[6] Thanks to Andrei Tamilin for this intuition

As for future work, we would like to further investigate logical consequence of more complex ontology spaces and distributed tableaux algorithms for more expressive ontology languages, such as OWL DL.

## References


1. J. Bao, D. Caragea, and V. Honavar. A Distributed Tableau Algorithm for Package-based Description Logics. In *Proc. of the Second International Workshop on Context Representation and Reasoning*, 2006.
2. T. Berners-lee. Semantic Web Road Map. W3C Design Issues. `URLhttp://www.w3.org/DesignIssues/Semantic.html`, Oct. 1998.
3. P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. C-owl – contextualizing ontologies. In *Proc. of the 2nd International Semantic Web Conference*, pages 164–179, 2003.
4. B. Cuenca-Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularizing OWL Ontologies. In *Proc. of the KCAP-2005 Workshop on Ontology Management.*
5. B. Cuenca-Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularity and Web Ontologies. In *KR2006*, 2006.
6. C. Ghidini and L. Serafini. Distributed First Order Logics. In *Frontiers of Combining Systems 2, Studies in Logic and Computation*, 2000.
7. O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. E-connections of description logics. In *Proc. of DL-2003*, pages 178–187, 2003.
8. A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz. An infrastructure for searching, reusing and evolving distributed ontologies. In *Prof. of WWW2003*, pages 439–448.
9. SUMO ontology. `http://ontology.teknowledge.com/`, 2005.
10. J. Z. Pan and I. Horrocks. OWL-Eu: Adding Customised Datatypes into OWL. In *Proc. of Second European Semantic Web Conference (ESWC 2005)*, 2005. Its revised and extended version is accepted in Journal of Web Semantics, 4(1).
11. E. B. Paslaru, M. Mochol, and R. Tolksdorf. Case Studies on Ontology Reuse. In *Proc. of the 5th Int. Conf. on Knowledge Management*, 2005.
12. P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. Technical report, W3C, Feb. 2004. W3C Recommendation, URL `http://www.w3.org/TR/2004/REC-owl-semantics-20040210/`.
13. K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Int. Joint Conf. on Artificial Intelligence*, pages 466–471, 1991.
14. M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.
15. L. Serafini, A. Borgida, and A. Tamilin. Aspects of Distributed and Modular Ontology Reasoning. In *Int. Joint Conf. on Artificial Intelligence*, pages 570–575, 2005.
16. H. Stuckenschmidt and M. Klein. Structure-based partitioning of large concept hierarchies. In *Proc. of Int. Semantic Web Conf.*, 2004.
17. M. Uschold and M. Gruninger. Ontologies: Principles, Methods and Applications. *The Knowledge Engineering Review*, 1996.
18. M. Uschold, M. Healy, K. Williamson, P. Clark, and S. Woods. Ontology reuse and application. In *Proc of the Int Conf on FOIS'98.*
19. M. Uschold and M. King. Towards a methodology for building ontologies. In *Proc. of the IJCAI95, Workshop on Basic Ontological Issues in Knowledge Sharing*, 1995.