MANCHESTER 1824

The University of Manchester

ONTOLOGY REUSE: BETTER SAFE THAN SORRY

Bernardo Guenca Grau, Ian Horrocks,
Yevgeny Kazakov and Ulrike Sattler

The University of Manchester

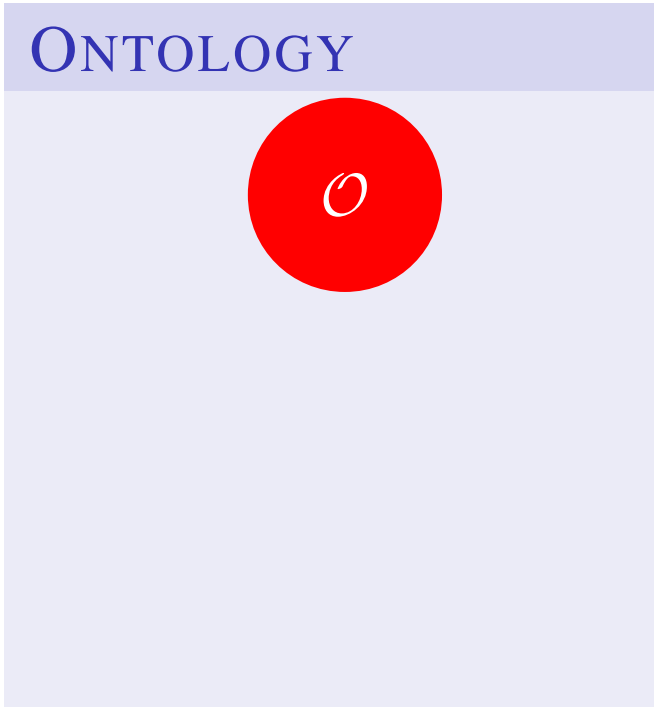
June 8, 2007

Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :

Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency

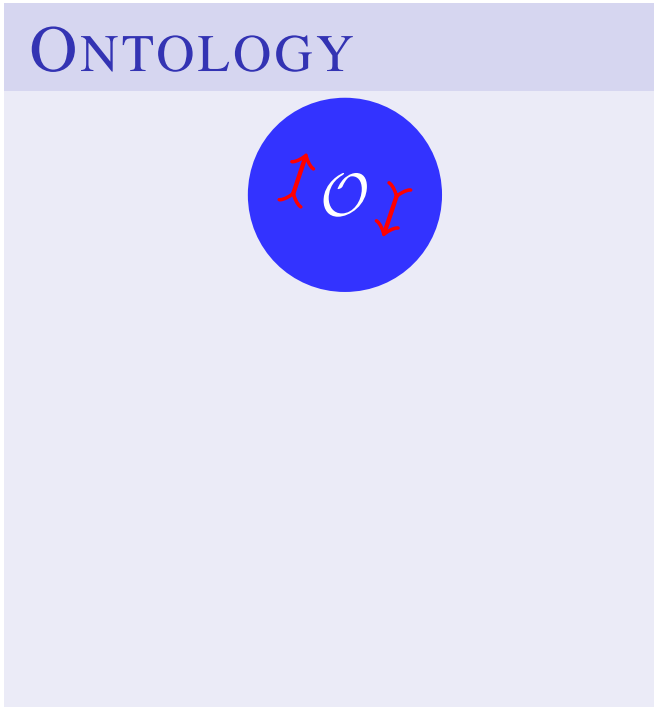


Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes

Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes
  - ✔ Detecting unintended subsumptions

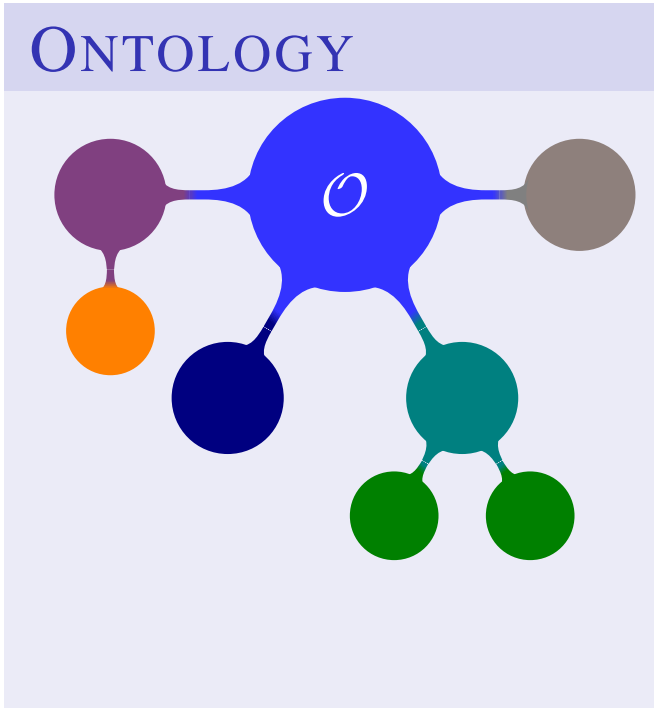


Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes
  - ✔ Detecting unintended subsumptions
- (No?) reasoning support for modular development of ontologies:

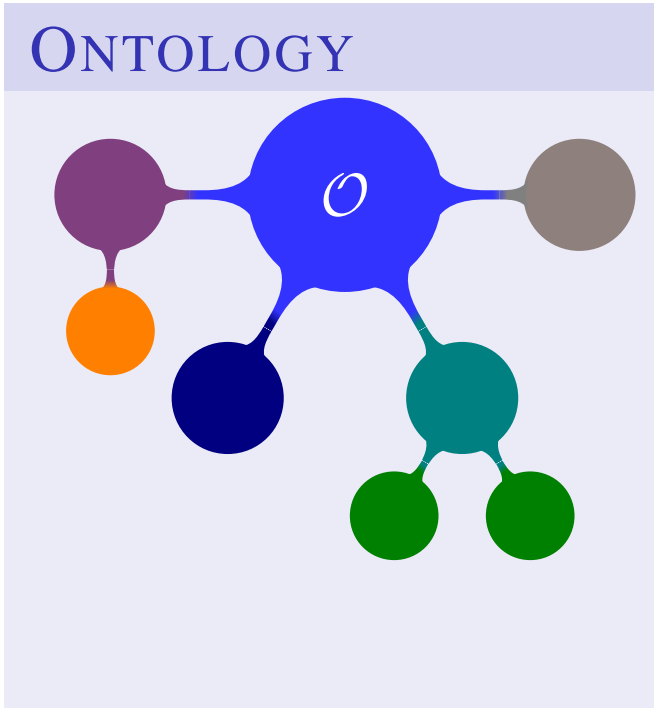
Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes
  - ✔ Detecting unintended subsumptions
- (No?) reasoning support for modular development of ontologies:
  - Build big ontologies from smaller once



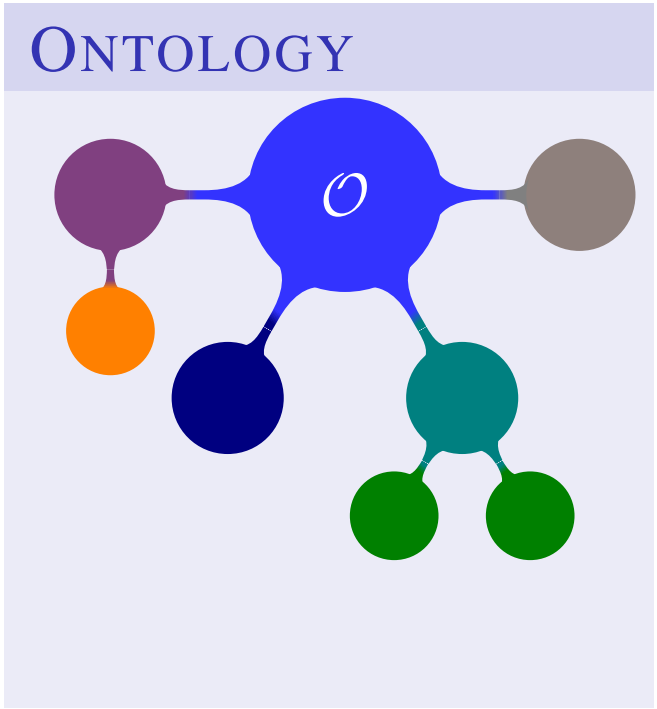
Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes
  - ✔ Detecting unintended subsumptions
- (No?) reasoning support for modular development of ontologies:
  - Build big ontologies from smaller once
  - Collaboratively



Reasoning Support for Ontology Development

- Currently DL-based tools provide reasoning support for development of ontologies as monolithic objects :
  - ✔ Checking global consistency
  - ✔ Detecting unsatisfiable classes
  - ✔ Detecting unintended subsumptions
- (No?) reasoning support for modular development of ontologies:
  - Build big ontologies from smaller once
  - Collaboratively
  - In a modular way



A Motivating Example

Ontology of Research Projects

$$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv \mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$$

$$\mathsf{Genetic\_Disorder\_Project} \equiv \mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$$

$$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$$

Ontology Reuse

A Motivating Example

Ontology of Research Projects

$$\text{Cystic\_Fibrosis\_EUProject} \equiv \text{EUProject} \sqcap \exists \text{has\_Focus}.\text{Cystic\_Fibrosis}$$

$$\text{Genetic\_Disorder\_Project} \equiv \text{Project} \sqcap \exists \text{has\_Focus}.\text{Genetic\_Disorder}$$

$$\text{EUProject} \sqsubseteq \text{Project}$$

Ontology Reuse

A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

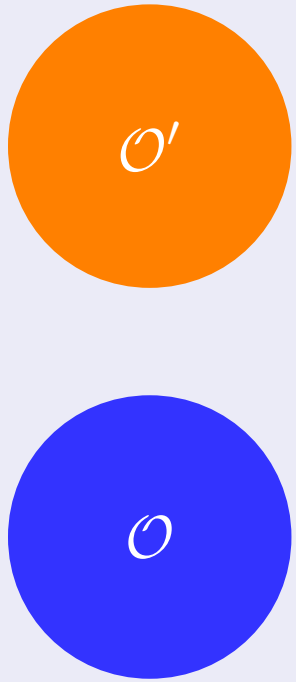
$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse



A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

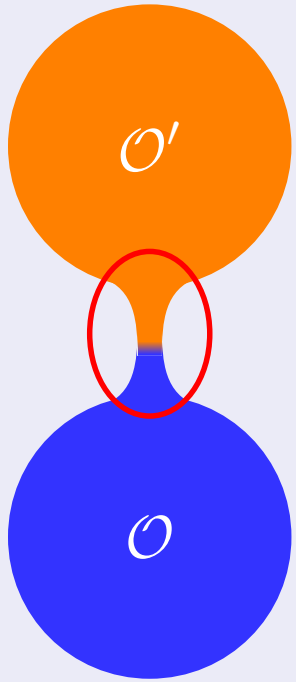
$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse



A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

$\models \quad \mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse

A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

$\models \quad \mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse

A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

$\models \quad \mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse

A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

$\models \quad \mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

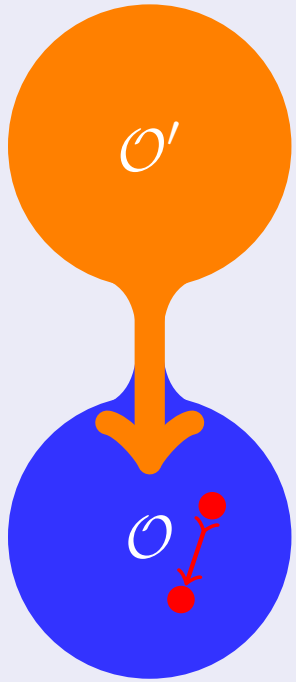
$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse



$\models \quad \mathsf{Cystic\_Fibrosis\_EUProject} \sqsubseteq \mathsf{Genetic\_Disorder\_Project}$

A Motivating Example

Ontology of Medical Terms

$\mathsf{Genetic\_Disorder} \equiv \ldots$

$\mathsf{Cystic\_Fibrosis} \equiv \ldots$

$\models \quad \mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$

Ontology of Research Projects

$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv$

$\mathsf{EUProject} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Cystic\_Fibrosis}$

$\mathsf{Genetic\_Disorder\_Project} \equiv$

$\mathsf{Project} \sqcap \exists \mathsf{has\_Focus}.\mathsf{Genetic\_Disorder}$

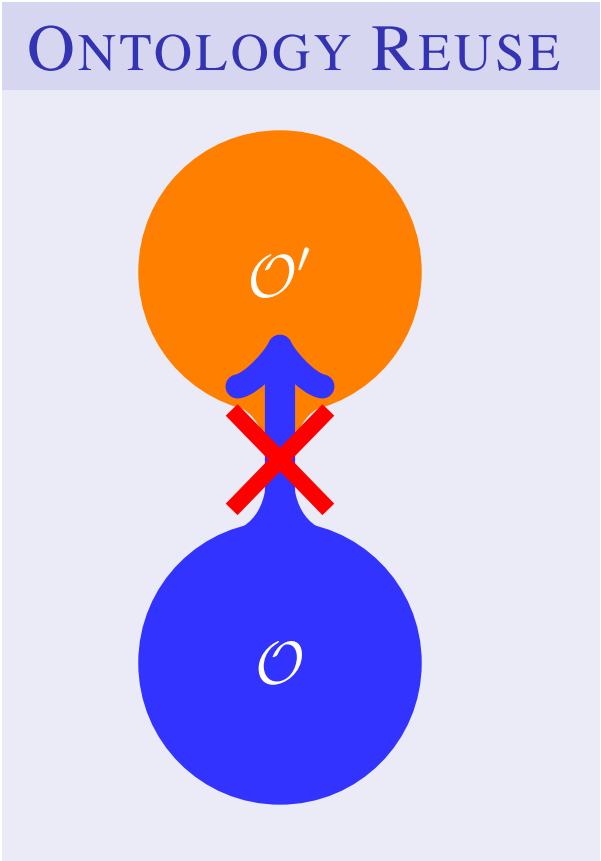
$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$

Ontology Reuse

$\models \quad \mathsf{Cystic\_Fibrosis\_EUProject} \sqsubseteq \mathsf{Genetic\_Disorder\_Project}$

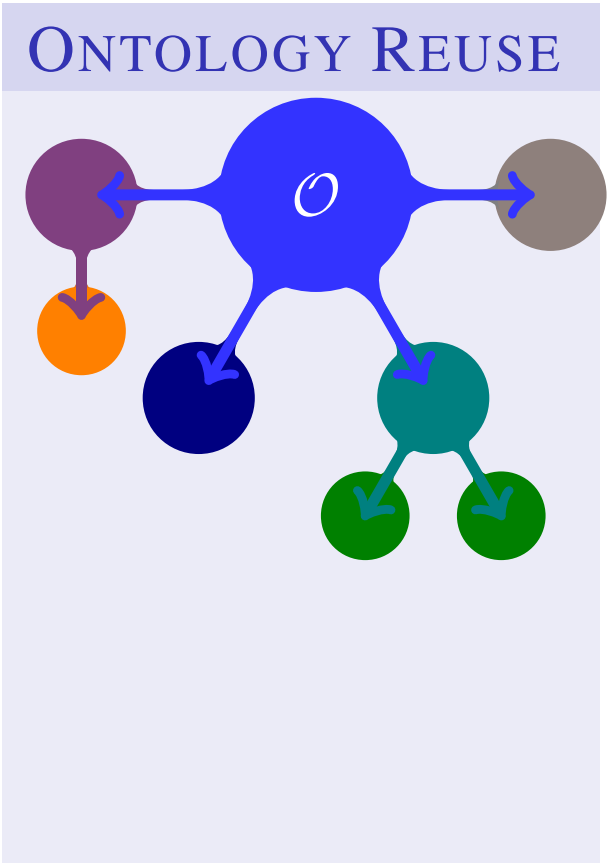
Why is It Important to Preserve the Meaning of the Imported Ontology?

1. Keeping the ontologies modular:
   - ➡ Every ontology developer is responsible for his own domain
   - ➡ The ontology which is merely reused, is not supposed to change even implicitly



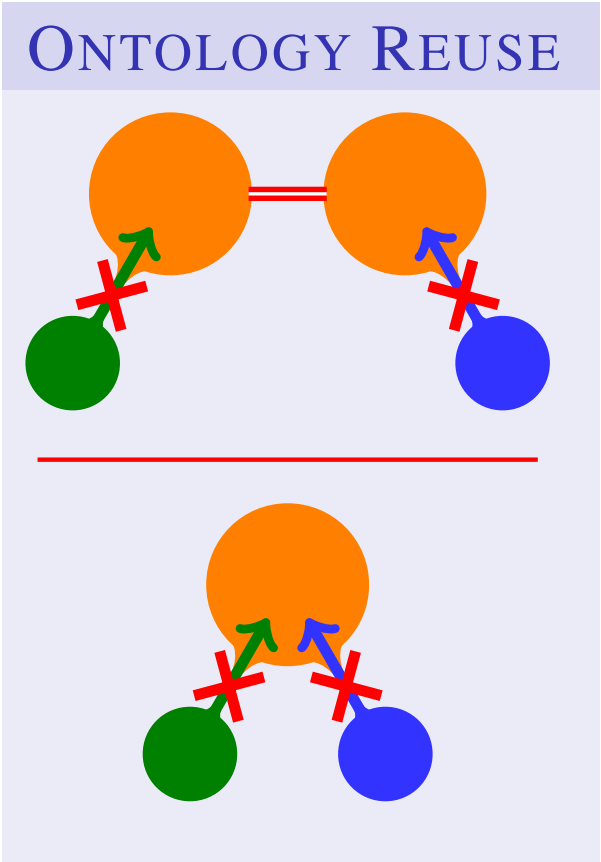
Why is It Important to Preserve the Meaning of the Imported Ontology?

1. Keeping the ontologies modular:
   - ➡ Every ontology developer is responsible for his own domain
   - ➡ The ontology which is merely reused, is not supposed to change even implicitly
2. Facilitates modular development of ontologies



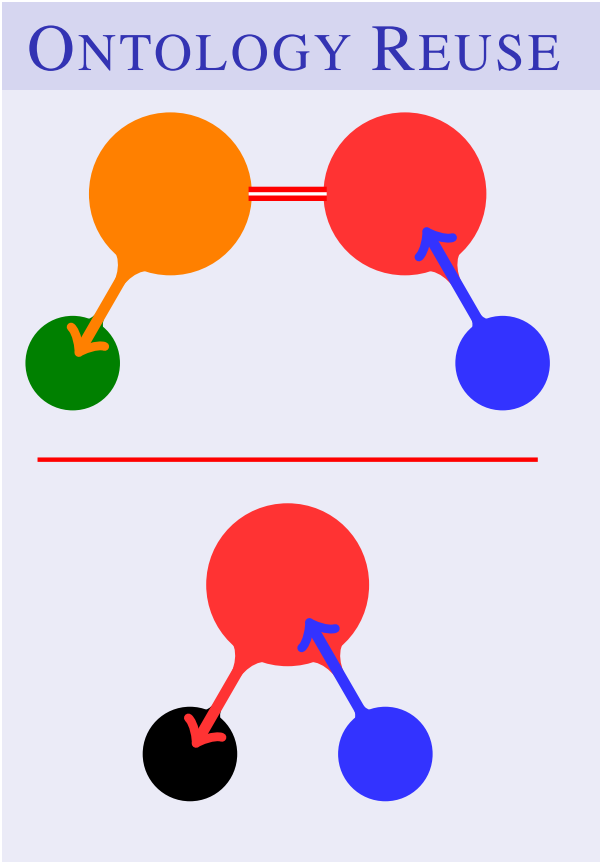
Why is It Important to Preserve the Meaning of the Imported Ontology?

1. Keeping the ontologies modular:
   - ➡ Every ontology developer is responsible for his own domain
   - ➡ The ontology which is merely reused, is not supposed to change even implicitly
2. Facilitates modular development of ontologies
   - ➡ Ontologies that use safely the same ontology can be safely combined



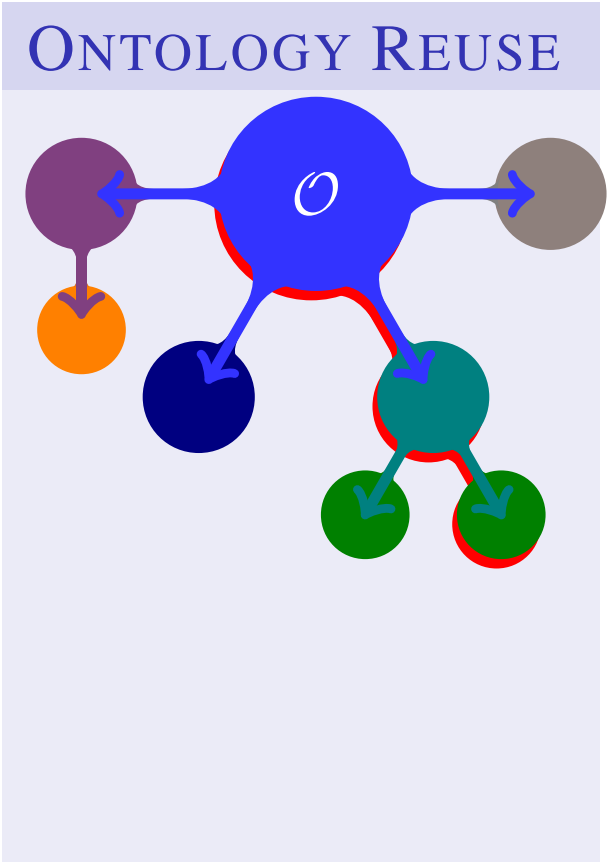
Why is It Important to Preserve the Meaning of the Imported Ontology?

1. Keeping the ontologies modular:
   - ➡ Every ontology developer is responsible for his own domain
   - ➡ The ontology which is merely reused, is not supposed to change even implicitly
2. Facilitates modular development of ontologies
   - ➡ Ontologies that use not safely the same ontology might not be safely combined



Why is It Important to Preserve the Meaning of the Imported Ontology?

1. Keeping the ontologies modular:
   - ➡ Every ontology developer is responsible for his own domain
   - ➡ The ontology which is merely reused, is not supposed to change even implicitly
2. Facilitates modular development of ontologies
   - ➡ Ontologies that use safely the same ontology can be safely combined
   - ➡ The developer of every ontologies can work independantly and only with ontologies that are imported.



FORMALISING "SAFE REUSE OF ONTOLOGIES"

INFORMALLY DEFINITION

An ontology $\mathcal{O}$ safely reuses ontology $\mathcal{O}'$ if $\mathcal{O}$ does not change the "meaning" of the reused symbols from $\mathcal{O}'$ during the import.

ONTOLOGY REUSE

Formalising "Safe Reuse of Ontologies"

Definition (1)

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. ontology language $\mathcal{L}$ if for every axiom α over $\mathcal{O}'$ expressed in $\mathcal{L}$, we have:

$$\mathcal{O}' \cup \mathcal{O} \models \alpha \quad \text{iff} \quad \mathcal{O}' \models \alpha$$

Informally Definition

An ontology $\mathcal{O}$ safely reuses ontology $\mathcal{O}'$ if $\mathcal{O}$ does not change the "meaning" of the reused symbols from $\mathcal{O}'$ during the import.

Ontology Reuse

Formalising "Safe Reuse of Ontologies"

Definition (1)

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. ontology language $\mathcal{L}$ if for every axiom α over $\mathcal{O}'$ expressed in $\mathcal{L}$, we have:

$$\mathcal{O}' \cup \mathcal{O} \models \alpha \quad \text{iff} \quad \mathcal{O}' \models \alpha$$

Example (1)

$$\mathcal{O}' = \left\{ \begin{array}{l} A \equiv \cdots \\ B \equiv \cdots \end{array} \right. \qquad \not\models B \sqsubseteq A$$

$$\mathcal{O} = \left\{ \begin{array}{l} C_1 \equiv A \sqcap C_2 \\ B \sqsubseteq C_1 \end{array} \right. \qquad \models B \sqsubseteq A$$

$\mathcal{O}' \cup \mathcal{O}$ is not a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L} = \mathcal{ALC}$.

Ontology Reuse

Formalising "Safe Reuse of Ontologies"

Definition (1)

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. ontology language $\mathcal{L}$ if for every axiom α over $\mathcal{O}'$ expressed in $\mathcal{L}$, we have:

$$\mathcal{O}' \cup \mathcal{O} \models \alpha \quad \text{iff} \quad \mathcal{O}' \models \alpha$$

Example (2)

$$\mathcal{O}' = \{ \; A \equiv \cdots \qquad \not\models \top \sqsubseteq A,\ A \sqsubseteq \bot$$

$$\mathcal{O} = \begin{cases} a : (A \sqcap B) \\ b : (A \sqcap \neg B) \end{cases} \qquad \not\models \top \sqsubseteq A,\ A \sqsubseteq \bot$$

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L} = \mathcal{ALC}$

Ontology Reuse

Formalising "Safe Reuse of Ontologies"

Definition (1)

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. ontology language $\mathcal{L}$ if for every axiom α over $\mathcal{O}'$ expressed in $\mathcal{L}$, we have:

$$\mathcal{O}' \cup \mathcal{O} \models \alpha \quad \text{iff} \quad \mathcal{O}' \models \alpha$$

Example (2)

$$\mathcal{O}' = \{ \; \mathsf{A} \equiv \cdots \qquad \not\models \top \sqsubseteq \mathsf{A},\ \mathsf{A} \sqsubseteq \bot$$

$$\mathcal{O} = \begin{cases} a : (\mathsf{A} \sqcap \mathsf{B}) & \not\models \top \sqsubseteq \mathsf{A},\ \mathsf{A} \sqsubseteq \bot \\ b : (\mathsf{A} \sqcap \neg \mathsf{B}) & \models |\mathsf{A}| \geq 2 \end{cases}$$

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L} = \mathcal{ALC}$

The "meaning" of A has been changed, but $\mathcal{L} = \mathcal{ALC}$ cannot "sense" it using axioms.

Ontology Reuse

Formalising "Safe Reuse of Ontologies"

Definition (2)

$\mathcal{O}' \cup \mathcal{O}$ is a model conservative extension of $\mathcal{O}'$ w.r.t. ontology language $\mathcal{L}$ if every model of $\mathcal{O}'$ can be expanded to a model of $\mathcal{O}' \cup \mathcal{O}$:

$$\forall \mathcal{I} \models \mathcal{O}' \; \exists \mathcal{J} \models \mathcal{O} : \; \mathcal{I}|_{\mathcal{O}'} = \mathcal{J}|_{\mathcal{O}'}$$

Example (2)

$$\mathcal{O}' = \{ \; \mathsf{A} \equiv \cdots \qquad \not\models \top \sqsubseteq \mathsf{A}, \; \mathsf{A} \sqsubseteq \bot$$

$$\mathcal{O} = \begin{cases} a : (\mathsf{A} \sqcap \mathsf{B}) & \not\models \top \sqsubseteq \mathsf{A}, \; \mathsf{A} \sqsubseteq \bot \\ b : (\mathsf{A} \sqcap \neg \mathsf{B}) & \models |\mathsf{A}| \geq 2 \end{cases}$$

$\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L} = \mathcal{ALC}$, but not model conservative

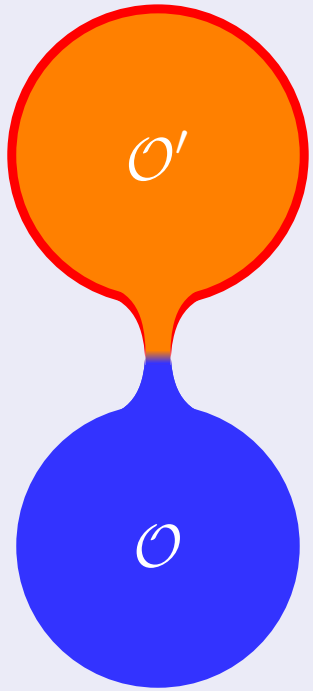
The "meaning" of A has been changed, but $\mathcal{L} = \mathcal{ALC}$ cannot "sense" it using axioms.

Ontology Reuse

Safety for Evolving Ontologies

- Ontologies are developed $\Rightarrow$ evolve

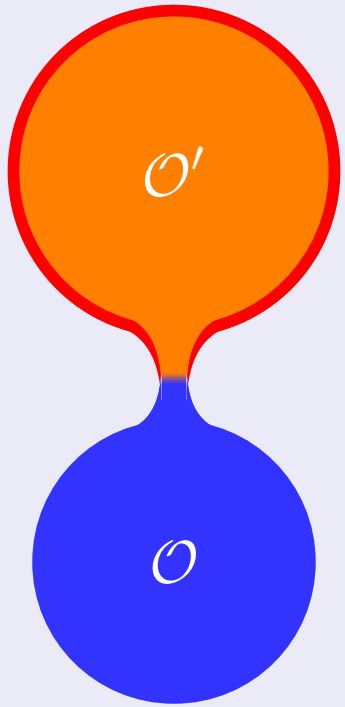
Ontology Reuse



Safety for Evolving Ontologies

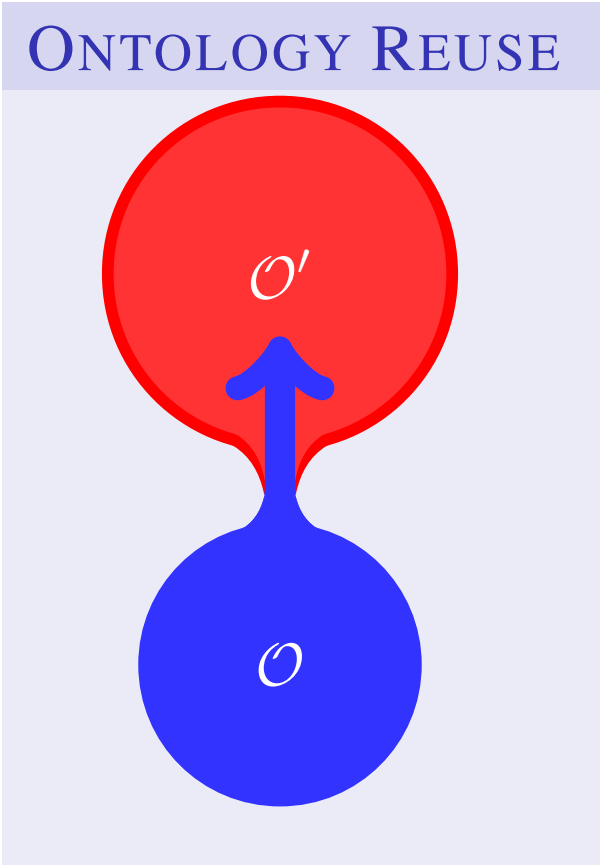
- Ontologies are developed $\Rightarrow$ evolve

Ontology Reuse



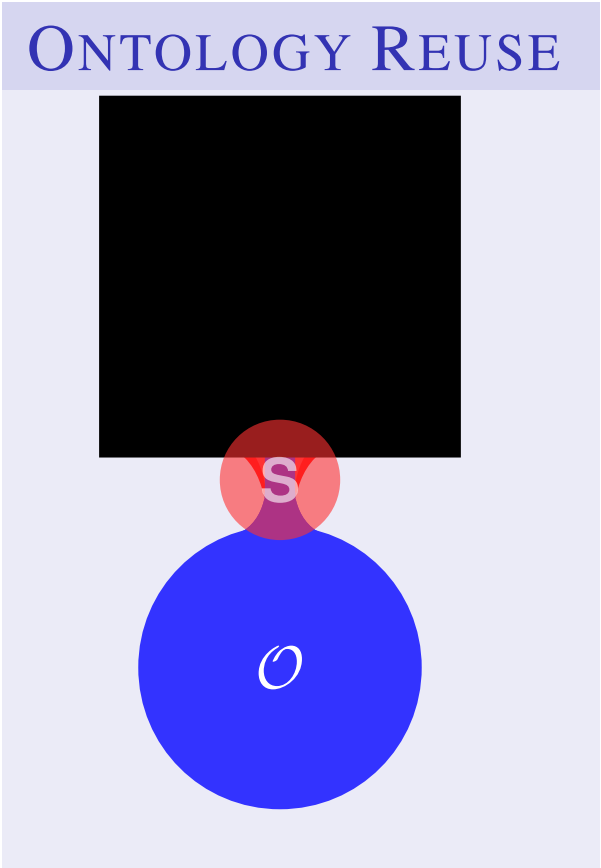
Safety for Evolving Ontologies

- Ontologies are developed $\Rightarrow$ evolve
- Even if $\mathcal{O}$ is importing safely one version of $\mathcal{O}'$, this might no longer hold for another version



Safety for Evolving Ontologies

- Ontologies are developed $\Rightarrow$ evolve
- Even if $\mathcal{O}$ is importing safely one version of $\mathcal{O}'$, this might no longer hold for another version
- Instead of focusing on the reused ontology one could focus just on the reused symbols and treat the ontology as a "black box".

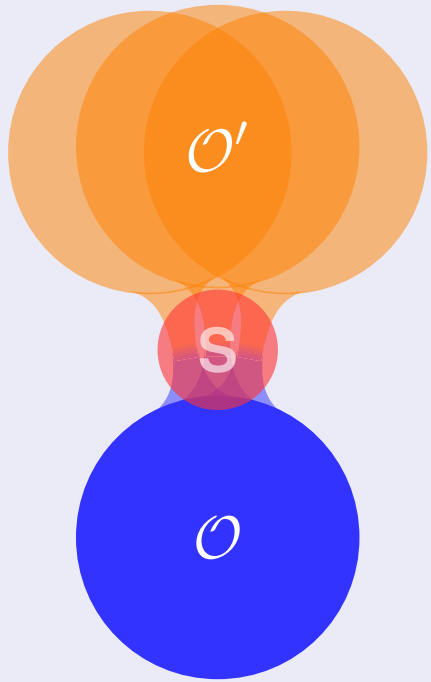


Safety of an Ontology for a Signature

Definition (Safety for a Signature)

$\mathcal{O}$ is safe for a signature **S** w.r.t. an ontology language $\mathcal{L}$ if for every $\mathcal{O}'$ formulated over $\mathcal{L}$ with $\mathsf{Sg}(\mathcal{O}') \cap \mathsf{Sg}(\mathcal{O}) \subseteq \mathbf{S}$, we have that $\mathcal{O} \cup \mathcal{O}'$ is a conservative extension of $\mathcal{O}'$.

Ontology Reuse



Safety of an Ontology for a Signature

Definition (Safety for a Signature)

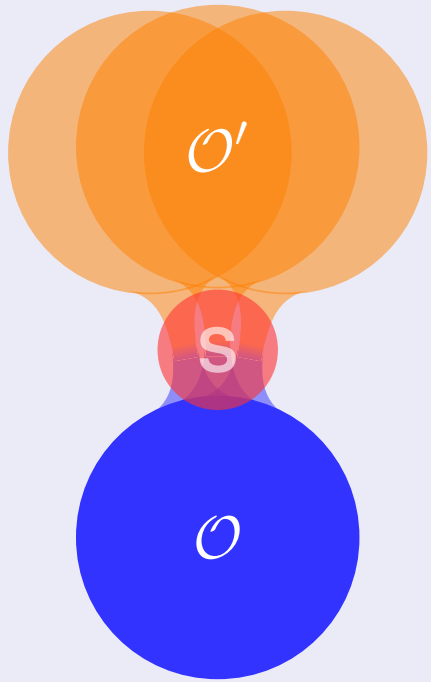
$\mathcal{O}$ is safe for a signature $\mathbf{S}$ w.r.t. an ontology language $\mathcal{L}$ if for every $\mathcal{O}'$ formulated over $\mathcal{L}$ with $\mathsf{Sg}(\mathcal{O}') \cap \mathsf{Sg}(\mathcal{O}) \subseteq \mathbf{S}$, we have that $\mathcal{O} \cup \mathcal{O}'$ is a conservative extension of $\mathcal{O}'$.

Theorem (Sufficient Condition)

An ontology $\mathcal{O}$ is safe for a signature $\mathbf{S}$ if for every interpretation $\mathcal{I}$ there exists a model $\mathcal{J}$ of $\mathcal{O}$ which coincides with $\mathcal{I}$ on $\mathbf{S}$:

$$\forall\, \mathcal{I}\ \exists\, \mathcal{J} \models \mathcal{O} :\ \mathcal{I}|_{\mathbf{S}} = \mathcal{J}|_{\mathbf{S}}$$

Ontology Reuse



Deciding Safety: How Hard is It?

- Checking if $\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L}$:
  - is 2-EXPTIME-complete for $\mathcal{L} = \mathcal{ALCQI}$
    [Ghilardi, Lutz & Wolter, 2006]
  - is uncecidable for $\mathcal{L} = \mathcal{ALCQIO}$
    [Lutz, Walther & Wolter, 2007]

Deciding Safety: How Hard is It?

- Checking if $\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L}$:
  - is 2-EXPTIME-complete for $\mathcal{L} = \mathcal{ALCQI}$
    [Ghilardi, Lutz & Wolter, 2006]
  - is uncecidable for $\mathcal{L} = \mathcal{ALCQIO}$
    [Lutz, Walther & Wolter, 2007]
- Checking if $\mathcal{O}' \cup \mathcal{O}$ is a model-conservative extension of $\mathcal{O}'$ is highly undecidable for $\mathcal{ALC}$-ontologies.
  [Ghilardi, Lutz & Wolter, 2006]

Deciding Safety: How Hard is It?

- Checking if $\mathcal{O}' \cup \mathcal{O}$ is a conservative extension of $\mathcal{O}'$ w.r.t. $\mathcal{L}$:
  - is 2-EXPTIME-complete for $\mathcal{L} = \mathcal{ALCQI}$
    [Ghilardi, Lutz & Wolter, 2006]
  - is uncecidable for $\mathcal{L} = \mathcal{ALCQIO}$
    [Lutz, Walther & Wolter, 2007]
- Checking if $\mathcal{O}' \cup \mathcal{O}$ is a model-conservative extension of $\mathcal{O}'$ is highly undecidable for $\mathcal{ALC}$-ontologies.
  [Ghilardi, Lutz & Wolter, 2006]

Theorem (Undecidability for Safety)

Given and $\mathcal{ALC}$-ontology $\mathcal{O}$ and a signature $\mathbf{S}$, it is undecidable whether $\mathcal{O}$ is safe for $\mathbf{S}$ w.r.t. $\mathcal{L} = \mathcal{ALCO}$.

Proof.

Reduction to the domino tiling problems. □

LOCALITY: SAFER THAN THE SAFEST SAFETY

The main idea:

- To prove that $\mathcal{O}$ is safe for **S** it is sufficient to extend any interpretation $\mathcal{I}$ of symbols from **S** to a model of $\mathcal{O}$
- Let us try to extend $\mathcal{I}$ by interpreting every new symbol as the empty set

Locality: Safer than the Safest Safety

The main idea:

- To prove that $\mathcal{O}$ is safe for **S** it is sufficient to extend any interpretation $\mathcal{I}$ of symbols from **S** to a model of $\mathcal{O}$
- Let us try to extend $\mathcal{I}$ by interpreting every new symbol as the empty set

Example

$$\mathcal{O} = \begin{cases} A \equiv B \sqcap \exists r.C \\ A \sqcap B \sqsubseteq \bot \\ \exists r.\top \sqsubseteq C \end{cases} \xrightarrow[A \leftarrow \emptyset]{r \leftarrow \emptyset} \begin{array}{ll} \bot \equiv B \sqcap \bot & \checkmark \\ \bot \sqcap B \sqsubseteq \bot & \checkmark \\ \bot \sqsubseteq C & \checkmark \end{array}$$

Locality: Safer than the Safest Safety

The main idea:

- To prove that $\mathcal{O}$ is safe for **S** it is sufficient to extend any interpretation $\mathcal{I}$ of symbols from **S** to a model of $\mathcal{O}$
- Let us try to extend $\mathcal{I}$ by interpreting every new symbol as the empty set

Example

$$\mathcal{O} = \begin{cases} A \equiv B \sqcap \exists r.C \\ A \sqcap B \sqsubseteq \bot \\ \exists r.\top \sqsubseteq C \end{cases} \xrightarrow[A \leftarrow \emptyset]{r \leftarrow \emptyset} \begin{array}{ll} \bot \equiv B \sqcap \bot & \checkmark \\ \bot \sqcap B \sqsubseteq \bot & \checkmark \\ \bot \sqsubseteq C & \checkmark \end{array}$$

Definition (Locality for $\mathcal{L} = \mathcal{SHOIQ}$)

An ontology $\mathcal{O}$ is local w.r.t. **S** if $\mathcal{J} \models \mathcal{O}$ for every $\mathcal{J}$ which interpret all concept and role names not in **S** as the empty set.

Properties of Locality

+ If every $\mathcal{O}$ is local w.r.t. **S** then $\mathcal{O}$ is safe for **S**:

Properties of Locality

- If every $\mathcal{O}$ is local w.r.t. **S** then $\mathcal{O}$ is safe for **S**:
- Checking locality can be done using any standard DL-reasoner.

Properties of Locality

- If every $\mathcal{O}$ is local w.r.t. **S** then $\mathcal{O}$ is safe for **S**:
- Checking locality can be done using any standard DL-reasoner.
- There is a sufficient syntactical condition for locality which can be verified in polynomial time.

Imperial Evaluation

- We have implemented our algorithm and tried it on a library of 300 OWL ontologies.
- It turned out that in almost all cases when OWL ontologies import each other our syntactic locality conditions hold
  1. There are 96 ontologies that import others
  2. All except for 11 enjoy our syntactical conditions
  3. Among non-local, 7 are written in OWL-Full
  4. In the remaining 4 the problem is caused by mapping axioms $A \equiv B$ and can be fixed by replacing A with B in $\mathcal{O}$.

CONCLUSIONS

- We formalized the requirements for safe ontology reuse using the notions of conservative extensions
- We proved that safety is undecidable for extensions of $\mathcal{ALCO}$
- We formulated sufficient conditions for safety using the semantic and syntactic localities
- Preliminary empirical evaluation is encouraging

1. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U.Sattler. A logical framework for modularity of ontologies. In Proc. of IJCAI 2007
2. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Just the right amout: Extracting modules from ontologies. In Proc. of WWW 2007

Syntactic Locality

Syntactic Locality

$$C^{\emptyset} ::= A^{\emptyset} \mid C^{\emptyset} \sqcap C \mid C^{\emptyset} \sqcup C^{\emptyset} \mid \neg C^{\Delta} \mid \exists r^{\emptyset}.C \mid \exists r.C^{\emptyset}$$

$$C^{\Delta} ::= C^{\Delta} \sqcup C \mid C^{\Delta} \sqcap C^{\Delta} \mid \neg C^{\emptyset} \mid \forall r^{\emptyset}.C \mid \forall r.C^{\Delta}$$

$$Ax\_synt\_local ::= C^{\emptyset} \sqsubseteq C \mid C \sqsubseteq C^{\Delta}$$

Other Locality Conditions

Other locality conditions can be defined by choosing different ways to interpret the symbols that are not in **S**:

Examples and Comparison of Different Localities

$r \leftarrow$	$\emptyset$	$\Delta \times \Delta$	id	$\emptyset$	$\Delta \times \Delta$	id
$A \leftarrow$	$\emptyset$	$\emptyset$	$\emptyset$	Δ	Δ	Δ
$A \equiv B \sqcap \exists r.C$	✓	✓	✓	✗	✗	✗
$A \sqcap C \sqsubseteq \bot$	✓	✓	✓	✗	✗	✗
$\exists r.\top \sqsubseteq A$	✓	✗	✗	✓	✓	✓
$\mathit{Functional}(r)$	✓	✗	✓	✓	✗	✓
$a : A$	✗	✗	✗	✓	✓	✓
$r(a, b)$	✗	✓	✗	✗	✓	✗
$\forall r.C \sqsubseteq \exists r.D$	✗	✗	✗	✗	✗	✗