

# Modularity in Logical Theories and Ontologies



Thanks to Thomas Schneider and Ernesto Jiménez Ruiz for examples & graphics

## Plan for today

So far, we have discussed various properties of (various forms of) conservative extensions:

- decidability
- complexity
- robustness under vocabulary extensions, joins, replacement

Today, we look a little bit closer on how can we use these insights to help ontology engineers re-use ontologies

- in a controlled way
- without (unwanted) side-effects

Since approx. 2000, people have been designing **ontology languages**:

- useful, expressive KR formalisms (suitable to describe application domain) with
- useful, interesting, implementable inference problems

**OWL** has been and **OWL2** is being standardized within W3C:

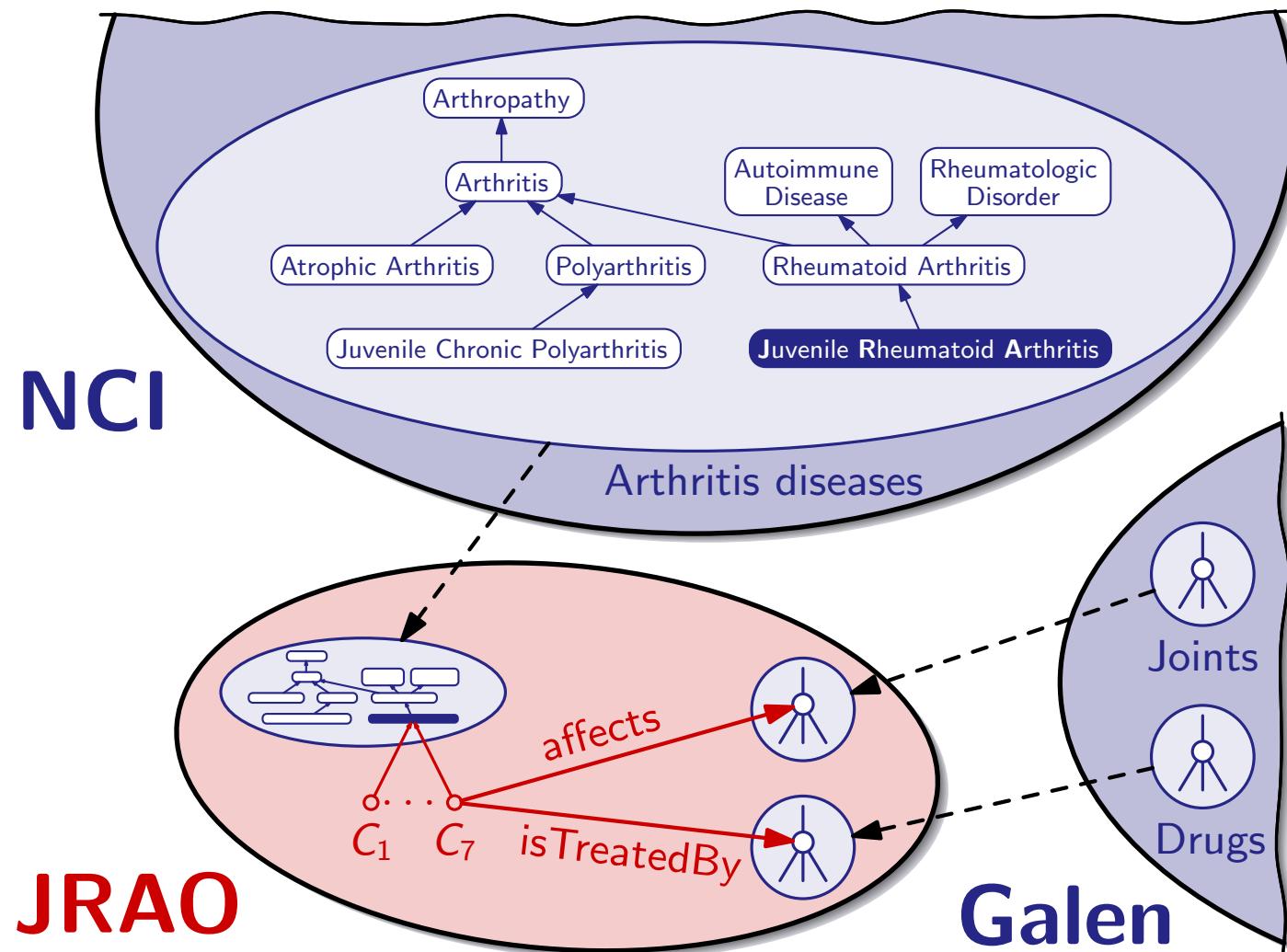
- extensions of  $\mathcal{ALC}$  with
  - inverse roles, transitive roles, role hierarchies, . . .
  - nominals (concept names denoting single elements)
  - number restrictions
  - datatypes (not discussed here)
- with standardized syntaxes & semantics and interesting fragments (called profiles)
- reasoners for consistency, subsumption hierarchy, query answering, . . .
- editors to design, maintain, analyze ontologies
- users from biology, clinical sciences, chemistry, engineering, etc.

## Motivation: Modularity for Re-Use of Ontologies

Let's have a look at a real example:

- Build an ontology *JRAO* that describes **JRA** (Juvenile Rheumatoid Arthritis)
- Describe **JRA** subkinds by
  - Joints affected
  - Occurrence of concomitant symptoms, e.g., fever
  - Treatment with certain drugs
- Re-use information provided by biomedical ontologies
  - *NCI*: diseases, drugs, proteins etc.
  - *Galen*: human anatomy

## Motivation: Modularity for Re-Use of Ontologies



## Motivation: Modularity for Re-Use of Ontologies

### Why re-use ontology?

- Saves time & effort
- Provides access to well-established knowledge & terminology
- Doesn't require expertise in drugs, proteins, anatomy etc.

This implies that a tool supporting re-use should guarantee that

importing terms doesn't change their meaning (safe)  
import only (but all) relevant parts of external ontologies (economic)  
the order of imports doesn't matter (independence)

Does this sound like conservativity?

## Guarantees by Example

**Safety:** concerns the usage of (imported) terms in importing ontology:

$$\begin{aligned} JRAO \cup NCI \models JRA \sqsubseteq GeneticDisorder, \\ \text{for } JRA, GeneticDisorder \in \text{sig}(NCI) \\ \text{iff} \\ NCI \models JRA \sqsubseteq GeneticDisorder \end{aligned}$$

**Economy:** concerns what we would consider a **module**:

$$\begin{aligned} JRAO \cup NCI \models JRA \sqsubseteq GeneticDisorder, \\ \text{iff} \\ JRAO \cup NCI\text{-module} \models JRA \sqsubseteq GeneticDisorder \end{aligned}$$

**Independence:** if  $JRAO$  is safe for *Galen* and for  $NCI$ , then  
 $JRAO \cup NCI\text{-module}$  is still safe for *Galen* and  
 $JRAO \cup Galen\text{-module}$  is still safe for  $NCI$

### Safety for an ontology:

$\mathcal{O}_1$  imports  $\mathcal{O}_2$  in an  $\mathcal{L}$ -safe way

(or  $\mathcal{O}_1$  is safe for  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$ ) if

$\mathcal{O}_1 \cup \mathcal{O}_2$  is a deductive conservative extension of  $\mathcal{O}_2$  w.r.t  $\mathcal{L}$   
(and  $\Sigma = \text{sig}(\mathcal{O}_2)$ .)

### Problems:

- which  $\mathcal{L}$  to choose?
  - for ontology design: subsumptions between (possibly complex?) concepts
  - for ontology usage: my favourite query language
- we might not have control over  $\mathcal{O}_2$ , e.g.,  
 $\mathcal{O}_2 = NCI$  might change over time, and we want the latest version

### Solution: Safety for a signature!

## Safety for a Signature

**Definition:**  $\mathcal{O}_1$  is safe for  $\Sigma$  w.r.t.  $\mathcal{L}$  if,

for every  $\mathcal{L}$ -ontology  $\mathcal{O}_2$  with  $\text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_2) \subseteq \Sigma$ ,

$\mathcal{O}_1 \cup \mathcal{O}_2$  is a deductive  $\Sigma$ -conservative extension of  $\mathcal{O}_2$  w.r.t  $\mathcal{L}$ .

**Theorem:** 1. If  $\mathcal{O}$  is a model  $\Sigma$ -conservative extension of  $\emptyset$ ,  
then  $\mathcal{O}$  is safe for  $\Sigma$  w.r.t.  $\mathcal{L}$ .

2. Let  $\mathcal{L}$  be robust under replacements.

Then  $\mathcal{O}_1$  is safe for  $\Sigma$  w.r.t.  $\mathcal{L}$  iff  $\mathcal{O}_1 \equiv_{\Sigma}^{\mathcal{L}} \emptyset$ .

**Observation:**  $(\mathcal{ALC}, \mathcal{ALC})$  isn't robust under replacements. Frank's example  
rephrased: consider  $\mathcal{O}_1 = \{A \sqsubseteq \exists r.B\}$

Then  $\mathcal{O}_1 \equiv_{\Sigma}^{\mathcal{ALC}} \emptyset$ , but  $\mathcal{O}_1$  is not safe for  $\Sigma = \{A, B\}$  because  
 $\mathcal{O}_1 \cup \{A \doteq \top, B \doteq \perp\} \models \top \sqsubseteq \perp$   
and is thus not a d.  $\Sigma$ -c.d. of  $\{A \doteq \top, B \doteq \perp\}$ .

Importance of the right query language, e.g.,  $\mathcal{QL}_{\mathcal{ALC}}^B$

**Module** for an ontology:

$\mathcal{O}'_2 \subseteq \mathcal{O}_2$  is a **module** for  $\mathcal{O}_1$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  if  
 $\mathcal{O}_1 \cup \mathcal{O}'_2$  is a deductive  $\text{sig}(\mathcal{O}_1)$ -c.e. of  $\mathcal{O}_1 \cup \mathcal{O}'_2$  w.r.t.  $\mathcal{L}$ .

**Problems:** 1. which  $\mathcal{L}$  to chose?

- for ontology design: subsumptions between (possibly complex?) concepts
- for ontology usage: my favourite query language

2. the module shouldn't depend on the importing ontology, but only on the signature we want to use

**Solution:** Module for a signature!

## Module for a Signature

**Definition:**  $\mathcal{O}'_2 \subseteq \mathcal{O}_2$  is a **module** for  $\Sigma$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  if,  
for every  $\mathcal{L}$ -ontology  $\mathcal{O}_1$  with  $\text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_2) \subseteq \Sigma$ ,  
 $\mathcal{O}'_2$  is a module for  $\mathcal{O}_1$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$   
i.e.,  $\mathcal{O}_1 \cup \mathcal{O}_2$  is a deductive  $\text{sig}(\mathcal{O}_1)$ -c.e. of  $\mathcal{O}_1 \cup \mathcal{O}'_2$  w.r.t.  $\mathcal{L}$ .

**Observation:**

1. Let  $\mathcal{L}$  be robust under replacements. Then  
 $\mathcal{O}'_2 \subseteq \mathcal{O}_2$  is a module for  $\Sigma$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$   
iff  
 $\mathcal{O}_2 \setminus \mathcal{O}'_2 \equiv_{\Sigma}^{\mathcal{L}} \emptyset$
2. if  $\mathcal{O}'_2 \subseteq \mathcal{O}_2$  and  $\mathcal{O}_2$  is a model  $\Sigma$ -conservative extension of  $\mathcal{O}'_2$ ,  
then  $\mathcal{O}'_2$  is a module for  $\Sigma$  in  $\mathcal{O}_2$  (w.r.t. FO)

## Modules and Safety are closely related

The following is immediate by the previous observations:

Let  $\mathcal{O}_1, \mathcal{O}'_2 \subseteq \mathcal{O}_2$  be ontologies in  $\mathcal{L}$  and  $\Sigma$  a signature. Then

1.  $\mathcal{O}_1$  is safe for  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  iff  $\emptyset$  is a module for  $\mathcal{O}_2$  in  $\mathcal{O}_1$  w.r.t.  $\mathcal{L}$   
(iff  $\mathcal{O}_1$  constrains interpretation of terms in  $\mathcal{O}_2$  as much as  $\emptyset$ )
2. If  $\mathcal{O}_2 \setminus \mathcal{O}'_2$  is safe for  $\mathcal{O}_1 \cup \mathcal{O}'_2$  w.r.t.  $\mathcal{L}$ , then  $\mathcal{O}'_2$  is a module for  $\mathcal{O}_1$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$   
(because  $\mathcal{O}_2 \setminus \mathcal{O}'_2$  doesn't constrain interpretation of terms from  $\mathcal{O}_1 \cup \mathcal{O}'_2$ )
3.  $\mathcal{O}_1$  is safe for  $\Sigma$  w.r.t.  $\mathcal{L}$  iff  $\emptyset$  is a  $\Sigma$ -module in  $\mathcal{O}_1$  w.r.t.  $\mathcal{L}$   
(iff  $\mathcal{O}_1$  constrains interpretation of terms in  $\Sigma$  as much as  $\emptyset$ )
4. If  $\mathcal{O}_2 \setminus \mathcal{O}'_2$  is safe for  $\Sigma \cup \text{sig}(\mathcal{O}'_2)$  w.r.t.  $\mathcal{L}$ ,  
then  $\mathcal{O}'_2$  is a  $\Sigma$ -module in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$   
(because  $\mathcal{O}_2 \setminus \mathcal{O}'_2$  doesn't constrain interpretation of terms from  $\Sigma \cup \text{sig}(\mathcal{O}'_2)$ )

## Independence Guarantee in Detail

A basic requirement if we want to 'import' ontologies in parallel/independently:

**Independence:** safety is preserved under imports:

if  $\mathcal{O}_1$  is safe for  $\Sigma_i/\mathcal{O}_i$ , then  
 $\mathcal{O}_1 \cup \mathcal{O}_j$  is still safe for  $\Sigma_i/\mathcal{O}_i$ .

**Observation:** **Independence is difficult to guarantee:**

- if  $\Sigma_i$  share terms, then **Independence isn't guaranteed**:  
e.g.,  $\mathcal{O}_1 = \{A \sqsubseteq \top\}$  is safe for  $\Sigma = \{A, B\}$ , but  
 $\mathcal{O}_1 \cup \{A \sqsubseteq B\}$  is **not** safe for  $\Sigma$
- even if  $\Sigma_i$  don't share terms, **Independence is difficult**:  
E.g.,  $\mathcal{O}_1 = \{A_2 \sqsubseteq A_3\}$  is safe for  $\Sigma_i = \{A_i\}$ , but  
 $\mathcal{O}_1 \cup \{A_3 \sqsubseteq \perp\}$  is **not** safe for  $\Sigma_2$  and  
 $\mathcal{O}_1 \cup \{A_2 \doteq \top\}$  is **not** safe for  $\Sigma_3$

## Problems one needs to solve to support Ontology Engineering

Given 'our' ontology  $\mathcal{O}_1$  and ontologies  $\mathcal{O}_i$  we want to re-use terms  $\Sigma_i$  from,

1. make sure that  $\mathcal{O}_1$  is safe for  $\Sigma_i$
2. determine modules for  $\Sigma_i$  from  $\mathcal{O}_i$  – but which?
  - (a) did engineer 'forget something' when specifying  $\Sigma_i$ ?
  - (b) should modules be as small as possible?
  - (c) even minimal modules are not unique (see next slide) – which one to use?
3. add modules  $\mathcal{O}'_i$  to  $\mathcal{O}_1$ 
  - (a) static/call-by-value: determine and add  $\mathcal{O}'_i$
  - (b) dynamic/call-by-name: always use 'freshest'  $\mathcal{O}'_i$  — how?  
(we need to provide mechanisms/syntax for this)

## Example

Suppose *Galen* contains:

(1)

$$\text{Knee} \doteq \text{Joint} \sqcap \exists \text{hasPart}.\text{Patella} \sqcap \exists \text{hasFunct}.\text{Hinge}$$

(2)

$$\text{Patella} \sqsubseteq \text{Bone} \sqcap \text{Sesamoid}$$

(3)

$$\text{Ginglymus} \doteq \text{Joint} \sqcap \exists \text{hasFunct}.\text{Hinge}$$

(4)

$$\text{Joint} \sqcap \exists \text{hasPart}.(\text{Bone} \sqcap \text{Sesamoid}) \sqsubseteq \text{Ginglymus}$$

(5)

$$\text{Ginglymus} \doteq \text{HingeJoint}$$

(6)

$$\text{Meniscus} \doteq \text{FibroCartilage} \sqcap \exists \text{locatedIn}.\text{Knee}$$

Both  $\{(1), (2), (4), (5)\}$  and  $\{(1), (3), (5)\}$  are  $\sqsubseteq$ -minimal modules for  $\{\text{Knee}, \text{HingeJoint}\}$ .

Note: a module for  $\Sigma$  does not necessarily contain all axioms that use terms from  $\Sigma$ .

## Bad News for expressive DLs/ontology languages such as OWL?

Big, sad theorem: Let  $\mathcal{O}_1, \mathcal{O}'_2 \subseteq \mathcal{O}_2$  be ontologies in  $\mathcal{L}$  and  $\Sigma$  a signature.

- (1) Determining whether  $\mathcal{O}_1$  is safe for  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  or  
whether  $\mathcal{O}'_2$  is a module for  $\mathcal{O}_1$  in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  is

ExpTime-complete for  $\mathcal{L} = \mathcal{EL}$  (only  $\Box$  and  $\exists r.C$ , more tomorrow),  
2-ExpTime-complete for  $\mathcal{ALC} \leq \mathcal{L} \leq \mathcal{ALCQI}$ , and  
undecidable for  $\mathcal{L} = \mathcal{ALCQIO}$  and therefore for OWL

- (2) Determining whether a  $\mathcal{O}_1$  is safe for a signature  $\Sigma$  or  
whether  $\mathcal{O}'_2$  is a  $\Sigma$ -module in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$  is

undecidable w.r.t.  $\mathcal{L} = \mathcal{ALCO}$  (even if  $\mathcal{O}_1$  is in  $\mathcal{ALC}$ ).

## Bad News for expressive DLs/ontology languages such as OWL?

Proof: (1) easy consequence of results on deductive conservative extensions (dces):

- (upper bounds) transfer by definition of safety/modules
- (lower bounds) any algorithm for deciding safety/modules can be used to decide dces:

$\mathcal{O} \supseteq \mathcal{O}'$  is a dce of  $\mathcal{O}'$  w.r.t.  $\mathcal{L}$

iff

$\emptyset$  is a module for  $\mathcal{O}'$  in  $\mathcal{O} \setminus \mathcal{O}'$  w.r.t.  $\mathcal{L}$ .

(2) Variation of construction by [Lutz+07], uses reduction of tiling problem:  
well known that

- periodically tilable domino systems  $\mathcal{D}_{pt}$  and
- untilable domino systems  $\mathcal{D}_u = \mathcal{D} \setminus \mathcal{D}_t$

are **recursively inseparable**, i.e., no decidable  $\mathcal{D}'$  with  $\mathcal{D}_{pt} \subseteq \mathcal{D}' \subseteq \mathcal{D}_t$ .

## Continuation of proof of (2)

**Plan:** (i) for domino system  $D \in \mathcal{D}$ , construct  $\mathcal{O}_D$  with 1  $\mathcal{ALC}$  axiom for, signature  $\Sigma(D)$ , and set

$$\mathcal{D}' := \{D \mid \mathcal{O}_D \text{ is not safe for } \Sigma(D) \text{ w.r.t. } \mathcal{ALCO}\},$$

(ii) then show that,

- if  $D \in \mathcal{D}'$ , then  $D \in \mathcal{D}_t$ , i.e.,  $\mathcal{D}'$  is not too large
- if  $D \notin \mathcal{D}'$ , then  $D \notin \mathcal{D}_{pt}$ , i.e.,  $\mathcal{D}'$  is large enough

i.e., we have indeed  $\mathcal{D}_{pt} \subseteq \mathcal{D}' \subseteq \mathcal{D}_t$ .

Let's describe the usual tiling conditions, i.e.,

$$\begin{aligned} C_D := & (T_1 \sqcup \dots \sqcup T_k) \sqcap \\ & \neg(T_1 \sqcap T_2) \sqcap \neg(T_1 \sqcap T_3) \sqcap \dots \\ & \neg T_1 \sqcup (\exists h. \bigvee_{(T_1, T) \in H} T \sqcap \exists v. \bigvee_{(T_1, T) \in V} T) \sqcap \\ & \neg T_2 \sqcup (\exists h. \bigvee_{(T_2, T) \in H} T \sqcap \exists v. \bigvee_{(T_2, T) \in V} T) \sqcap \dots \end{aligned}$$

## Continuation of proof of (2)

$\mathcal{O}_D$  uses  $C_D$  to describe an “illegal” part of a model of  $\top \sqsubseteq C_D$ , i.e.,

$$\mathcal{O}_D := \{\top \sqsubseteq \exists s. (\neg C_D \sqcup ((\exists h. \exists v. B) \sqcap (\exists v. \exists h. \neg B)))\}$$

Set  $\Sigma(D) = \{h, v, T_1, T_2 \dots T_k\}$ , i.e.,  $s, B \notin \Sigma(D)$ .

(a) if  $\mathcal{O}_D$  is **not** safe for  $\Sigma(D)$  (iff  $D \in \mathcal{D}'$ ), show that  $D \in \mathcal{D}_t$  (by contraposition):

if  $D \in D_u$ , then every  $\Sigma(D)$ -interpretation  $\mathcal{I}$  can be extended to a model of  $\mathcal{O}_D$  by connecting every element, via  $s^{\mathcal{I}}$ , with either

- an instance of  $\neg C_D$  or, if  $\mathcal{I}$  is a model of  $\top \sqsubseteq C_D$  with
- an element for which  $h^{\mathcal{I}}v^{\mathcal{I}}$  and  $v^{\mathcal{I}}h^{\mathcal{I}}$  are “not confluent”

Hence  $\mathcal{O}_D$  is safe for  $\Sigma(D)$ .

## Continuation of proof of (2)

$\mathcal{O}_D$  uses  $C_D$  to describe a “non-grid” part of a model of  $\top \sqsubseteq C_D$ , i.e.,

$$\mathcal{O}_D := \{\top \sqsubseteq \exists s. (\neg C_D \sqcup ((\exists h. \exists v. B) \sqcap (\exists v. \exists h. \neg B)))\}$$

Set  $\Sigma(D) = \{h, v, T_1, T_2 \dots T_k\}$ , i.e.,  $s, B \notin \Sigma(D)$ .

(b) if  $\mathcal{O}_D$  is safe for  $\Sigma(D)$  (iff  $D \notin \mathcal{D}'$ ), show that  $D \notin \mathcal{D}_{pt}$  (by contraposition):  
 from a periodic tiling, say  $n \times m$ , construct an  $\mathcal{ALCO}$  ontology  $\mathcal{O}_p$  with  
 $\text{sig}(\mathcal{O}_p) \cap \text{sig}(\mathcal{O}_D) \subseteq \Sigma(D)$  such that  $\mathcal{O}_D \cup \mathcal{O}_p \models \top \sqsubseteq \perp$ :  
 $n \times m$  nominals  $a_{i,j}$  are used to describe finite model with periodic, cyclic tiling:

$$\begin{aligned}\mathcal{O}_p := \{ & a_{0,0} \sqsubseteq \exists h. a_{1,0} \sqcap \forall h. a_{1,0} \sqcap \exists v. a_{0,1} \sqcap \forall v. a_{0,1}, \\ & a_{0,1} \sqsubseteq \exists h. a_{1,1} \sqcap \forall h. a_{1,1} \sqcap \exists v. a_{0,2} \sqcap \forall v. a_{0,2}, \\ & \dots \qquad \qquad \dots \\ & a_{n,m} \sqsubseteq \exists h. a_{0,m} \sqcap \forall h. a_{0,m} \sqcap \exists v. a_{n,0} \sqcap \forall v. a_{n,0}, \\ & \top \sqsubseteq a_{0,0} \sqcup a_{1,0} \sqcup \dots \sqcup a_{n,m}\}\end{aligned}$$

## Consequences for safety/modules of expressive DLs

Deciding safety/modules is highly complex/undecidable for expressive DLs.

**Question:** What to do?

- give up? No: modules/safety clearly too important
- reduce expressivity of logic? Yes – more tomorrow!
- approximate for expressive logics? Yes – but from the 'right' direction!

Next, we will see 2 approximations, i.e., sufficient conditions for safety:

1. based on semantic locality
2. based on syntactic locality

To capture that we have numerous ways of approximating safety, introduce **safety class**:

**Definition:** a **safety class**  $\mathcal{S} : \Sigma \mapsto 2^{\cup \mathcal{O}}$  for  $\mathcal{L}$  maps a signature to a set of ontologies such that, for each  $\mathcal{O} \in \mathcal{S}(\Sigma)$ ,  $\mathcal{O}$  is safe for  $\Sigma$  w.r.t.  $\mathcal{L}$ .

**Remarks:**

1. each  $\mathcal{S}$  can be seen as a **safety test**, but there can be many
2. we assume that  $\emptyset \in \mathcal{S}(\Sigma)$
3. interesting properties of  $\mathcal{S}$ :
  - (a) **anti-monotonic**:  $\Sigma_1 \subseteq \Sigma_2$  implies  $\mathcal{S}(\Sigma_2) \subseteq \mathcal{S}(\Sigma_1)$
  - (b) **compact**:  $\mathcal{O} \in \mathcal{S}(\Sigma)$  implies  $\mathcal{O} \in \mathcal{S}(\Sigma \cap \text{sig}(\mathcal{O}))$
  - (c) **subset-closed**:  $\mathcal{O}_1 \subseteq \mathcal{O}_2$  and  $\mathcal{O}_2 \in \mathcal{S}(\Sigma)$  implies  $\mathcal{O}_1 \in \mathcal{S}(\Sigma)$
  - (d) **union-closed**:  $\mathcal{O}_1, \mathcal{O}_2 \in \mathcal{S}(\Sigma)$  implies  $\mathcal{O}_1 \cup \mathcal{O}_2 \in \mathcal{S}(\Sigma)$

**Remember:**  $\mathcal{O}$  is  $\Sigma$ -safe w.r.t.  $\mathcal{L}$  if  $\mathcal{O}$  is a model  $\Sigma$ -c.e. of  $\emptyset$ ,  
i.e., for each  $\mathcal{I}$ , there is  $\mathcal{J} \models \mathcal{O}$  with  $\mathcal{I}|_\Sigma = \mathcal{J}|_\Sigma$

**Definition:** we call a set of interpretations  $\mathbf{I}$   $\Sigma$ -local if  
for each  $\mathcal{I}$ , there exists  $\mathcal{J} \in \mathbf{I}$  with  $\mathcal{I}|_\Sigma = \mathcal{J}|_\Sigma$ .

Let  $\mathbf{I}(\cdot)$  be a mapping that associates a signature  $\Sigma$  with  
a set of interpretations  $\mathbf{I}(\Sigma)$ .

We say that a safety class  $\mathcal{S}(\cdot)$  is based on  $\mathbf{I}(\cdot)$  if, for all  $\Sigma$ ,  
 $\mathcal{S}(\Sigma) = \{\mathcal{O} \mid \text{each } \mathcal{I} \in \mathbf{I}(\Sigma) \text{ is a model of } \mathcal{O}\}$

Consider  $\mathbf{I}^\emptyset(\cdot)$  with  $\mathbf{I}^\emptyset(\Sigma) = \{\mathcal{I} \mid A^\mathcal{I} = \emptyset = r^\mathcal{I} \text{ for all } r, A \notin \Sigma\}$ .

- each  $\mathbf{I}^\emptyset(\Sigma)$  is  $\Sigma$ -local, hence we call  $\mathbf{I}^\emptyset(\cdot)$  local
- consider  $\mathcal{S}(\cdot)$  based on  $\mathbf{I}^\emptyset(\Sigma)$  as follows:  
$$\mathcal{S}(\Sigma) = \{\mathcal{O} \mid \mathcal{O} \text{ contains only axioms } \alpha \text{ with } \mathcal{I} \models \alpha, \forall \mathcal{I} \in \mathbf{I}^\emptyset(\Sigma)\}$$
- indeed,  $\mathcal{S}(\cdot)$  is a safety class!

## Safety classes based on local sets of interpretations

**Theorem:** if  $I(\cdot)$  is local and  $\mathcal{S}(\cdot)$  based on it, then  $\mathcal{S}(\cdot)$  is a safety class, and  $\mathcal{S}(\cdot)$  subset-closed and union-closed.

**Remark:** other properties transfer from classes of interpretations to safety classes

**Question:** are there other local classes of interpretations?

Yes: e.g., consider

$$I^\Delta(\Sigma) = \{\mathcal{I} \mid A^{\mathcal{I}} = \Delta^{\mathcal{I}} \text{ for all } A \notin \Sigma \text{ and } r^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \text{ if } r \notin \Sigma\}$$

or

$$I_\emptyset^\Delta(\Sigma) = \{\mathcal{I} \mid A^{\mathcal{I}} = \Delta^{\mathcal{I}} \text{ for all } A \notin \Sigma \text{ and } r^{\mathcal{I}} = \emptyset \text{ if } r \notin \Sigma\}$$

or ...

But which ones are useful in practice?

And whose safety classes can we decide in practice?

## Testing locality

**Remember:** let  $S^\emptyset(\cdot)$  be the safety class based on  $I^\emptyset(\cdot)$ .

Then  $\mathcal{O} \in S^\emptyset(\Sigma)$  if,

for each  $\alpha \in \mathcal{O}$  and each  $\mathcal{I}$  where all  $r, A \notin \Sigma$  are interpreted as  $\emptyset$ ,  
we have  $\mathcal{I} \models \alpha$ .

**Algorithm:** Input:  $\Sigma, \mathcal{O}$  ALC TBox

Set safe = true

For each  $C_1 \sqsubseteq C_2 \in \mathcal{O}$  with  $C_i$  in NNF,

    replace all  $A \notin \Sigma$  with  $\perp$

    replace all  $\exists r.C$  with  $r \notin \Sigma$  with  $\perp$

    replace all  $\forall r.C$  with  $r \notin \Sigma$  with  $\top$

Call result  $C'_1 \sqsubseteq C'_2$

Set safe = false if  $C'_1 \sqcap \neg C'_2$  is satisfiable % can find countermodel

Return(safe)

The algorithm answers “true” iff  $\mathcal{O} \in S^\emptyset(\Sigma)$   
and can be extended to more expressive DLs easily.

Our algorithm decides  $\mathcal{S}^\emptyset(\Sigma)$ , and thus can be used to test for  $\Sigma$ -safety.

- But:
1. there are many kinds of locality, and  
together they only approximate  $\Sigma$ -safety.
  2. we still need to perform **reasoning**:  
for each axiom  $\alpha$ , test satisfiability of  $C'_1 \sqcap \neg C'_2$   
there are highly optimised reasoners available to do so, but...  
? isn't there a **cheaper** approximation?  
testing satisfiability in  $\mathcal{ALC}$  is ExpTime-complete!

Answer: we can use **syntactic approximation** of locality!

## Safety helps to guarantee Independence Guarantee

Using safety, we can guarantee independence as follows:

**Lemma:** Let  $\mathcal{O}_1, \mathcal{O}_2$  be ontologies and  $\Sigma_2, \Sigma_3$  disjoint signatures and let  $X$  be  $\emptyset$  or  $\Delta$ . If

1.  $\mathcal{O}_1 \in \mathcal{S}^X(\Sigma_3)$ ,
2.  $\text{sig}(\mathcal{O}_2) \cap \Sigma_3 = \emptyset$ , and
3.  $\mathcal{O}_2 \in \mathcal{S}^X(\emptyset)$ ,

then  $\mathcal{O}_1 \cup \mathcal{O}_2 \in \mathcal{S}^X(\Sigma_3)$ .

**Note:** for the above “scenario” to work, we would, additionally, require that  $\mathcal{O}_1$  is safe for  $\mathcal{O}_2$ , e.g., because  $\mathcal{O}_1 \in \Sigma^Y(\Sigma_2)$  and  $\text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_2) \subseteq \Sigma_2$ .

## Syntactic approximation of locality

We call an axiom  $\alpha$  **syntactically  $\Sigma$ -local** if it is of the form  $C \sqsubseteq C^\Delta$  or  $C^\emptyset \sqsubseteq C$ , for  $C^\emptyset$  and  $C^\Delta$  given by the following grammars:

start with  $A^\Sigma, r^\Sigma$  terms **not** in  $\Sigma$ , and  $r, C$  any term

$$\begin{aligned} C^\emptyset &::= A^\Sigma \mid \neg C^\Delta \mid C \sqcap C^\emptyset \mid C^\emptyset \sqcap C \mid \exists r^\Sigma.C \mid \exists r.C^\emptyset \\ C^\Delta &::= \top \mid \neg C^\emptyset \mid C^\Delta \sqcap C^\Delta \end{aligned}$$

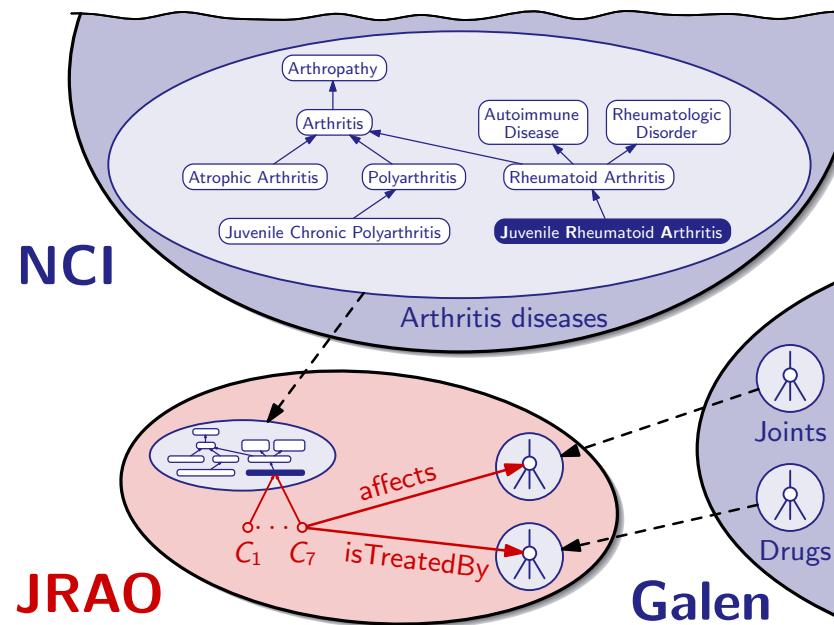
An ontology is **syntactically  $\Sigma$ -local** if it contains only syntactically  $\Sigma$ -local axioms.

Example:

$\bar{B} \sqsubseteq A$	form $C \sqsubseteq C^\emptyset$ , thus not $\{\bar{B}, \dots\}$ -local,
$A \sqsubseteq \bar{B} \sqcap \exists r.\bar{C}$	form $C^\emptyset \sqsubseteq C$ , thus $\{\bar{B}, \bar{C}\}$ -local,
$X \sqcap A \sqsubseteq Y$	is $\Sigma$ -local if, e.g., $A \notin \Sigma$ ,
$\bar{B} \sqcap \exists r.\bar{C} \sqsubseteq A$	is $\{\bar{B}, \bar{C}\}$ -local,
$\bar{A} \sqsubseteq \bar{A} \sqcup \bar{B}$	is not $\{\bar{A}, \bar{B}\}$ -local, yet a tautology!

Theorem: syntactic  $\Sigma$ -locality implies  $\Sigma$ -locality implies  $\Sigma$ -safety

## Our Example



In *JRAO*: we can 'syntactically safely' write

$$\begin{aligned} JRA &\doteq \overline{\text{Arthritis}} \sqcap \exists \text{affects}.(\overline{\text{Joint}} \sqcap \exists \text{locatedIn}. \text{Juvenile}) \\ KJRA &\doteq JRA \sqcap \exists \text{affects}. \overline{\text{Knee}} \end{aligned}$$

i.e., we can reference and refine existing terms from *NCI* and *Galen*

If we want to generalise terms, we use different syntactic locality: based on  $I^\Delta(\cdot)$ .

## Locality for modules

**Remember:** if  $\mathcal{O}_2 \setminus \mathcal{O}'_2$  is safe for  $\Sigma \cup \text{sig}(\mathcal{O}'_2)$  w.r.t.  $\mathcal{L}$ ,  
then  $\mathcal{O}'_2$  is a  $\Sigma$ -module in  $\mathcal{O}_2$  w.r.t.  $\mathcal{L}$ .

⇒ a poly-time algorithm to determine whether  $\mathcal{O}'_2$  is a  $\Sigma$ -module in  $\mathcal{O}_2$   
or to compute a  $\Sigma$ -module in  $\mathcal{O}_2$ :

**Algorithm:** Input:  $\Sigma$ ,  $\mathcal{O}$  TBox

$M := \emptyset$ ,  $\Sigma_n := \Sigma_o := \Sigma$

Repeat  $\Sigma_o := \Sigma_n$

    For each  $\alpha \in \mathcal{O}_2 \setminus M$

        if  $\alpha$  is not  $\Sigma_n$ -safe, then add  $\alpha$  to  $M$  and  $\text{sig}(\alpha)$  to  $\Sigma_n$

Until  $\Sigma_o = \Sigma_n$

Return( $M$ ,  $\Sigma_n$ )

**Observation:**  $M$  is a  $\Sigma_n$ -module in  $\mathcal{O}$  and therefore a  $\Sigma$ -module

(since  $\Sigma \subseteq \Sigma_n$  and  $\mathcal{S}(\cdot)$  based on  $I^\emptyset(\cdot)$  is anti-monotonic)

- clearly, we can use other safety checks (e.g., for safety based on  $I^\Delta(\cdot)$ ) in our algorithm
- we can combine syntactic with semantic locality:
  1. compute syntactically  $\Sigma$ -local module  $\mathcal{O}'_2$   
which is hopefully much smaller than  $\mathcal{O}_2$
  2. try to make  $\mathcal{O}'_2$  smaller using semantic locality:  
guess axiom  $\alpha \in \mathcal{O}'_2$  and  
check whether it can be removed, i.e., whether  
 $\mathcal{O}_2 \setminus (\mathcal{O}'_2 \setminus \{\alpha\})$  is semantically  $\Sigma \cup \text{sig}(\mathcal{O}'_2 \setminus \{\alpha\})$ -local

This line of research is 'new' for DLs & ontology languages, and many questions are (half)open:

- how can we support designer of  $\mathcal{O}_1$  to pick  $\mathcal{O}_2$ ,  $\Sigma$ ,  $\mathcal{L}$  so that
  - they want to import  $\Sigma$ -module in  $\mathcal{O}_2$ ?
  - they want to make sure that  $\mathcal{O}_1$  remains  $\Sigma$ -safe?
- how can we show  $\mathcal{O}'_2$ , a  $\Sigma$ -module in  $\mathcal{O}_2$  to designer of  $\mathcal{O}_1$  to ensure that they really want to import it?
- how can we ensure safety of  $\mathcal{O}_1$  for various signatures if 'imported' ontologies are unknown?
- how can we use (semantic and syntactic) locality to compute 'good' modules?
- how can we visualise the modular structure of an ontology – see crop circle demo
- how can we explain that  $X$  is not safe for  $Y$ ?
- how can we use modules to speed up reasoning?

- DL: <http://dl.kr.org/>
- Stuff around OWL from Manchester: <http://owl.cs.manchester.ac.uk/>
- DL reasoners: <http://www.cs.man.ac.uk/~sattler/reasoners.html>
- OWL2 working group: <http://www.w3.org/2007/OWL/>
- OWLED: <http://www.webont.org/owled/>

Tomorrow: Modularity for Lightweight Description Logics

Thanks for your interest  
and please feel free to ask questions, discuss things, etc.