Reasoning

German Rigau i Claramunt

german.rigau@ehu.es

IXA group

Departamento de Lenguajes y Sistemas Informáticos

UPV/EHU

Reasoning
Outline

- Reasoning mechanisms
- Graph-based reasoning
- MCR and consistency checking
  - TCO
  - SUMO
- Reasoning with SUMO
- Reasoning in KYOTO

Outline

$$\frac{A \rightarrow B \quad A}{B}$$

Outline

$$\frac{A \rightarrow B \quad A}{B}$$

$$\frac{A \rightarrow B \quad A}{?} \qquad \text{Deduction}$$

$$\frac{? \quad A}{B} \qquad \text{Induction}$$

$$\frac{A \rightarrow B \quad ?}{B} \qquad \text{Abduction}$$

eXtended WordNet

Introduction

- (Harabagiu 98, Moldovan 03)
- Commonse reasoning requires extensive knowledge
- ~ 100 millions of concepts and relations
- WordNet
  - represents almost all English words
  - 100.000 synsets
  - linked by semantic relations
- XWN
  - each synset has a gloss that, when disambiguated may increase the number of relations
  - WordNet glosses into semantic networks
  - NEW RELATIONS

eXtended WordNet

Text Inferences

German was hungry.
He opened the refrigerator.

- hungry (feeling a need or desire to eat)
- eat (take in solid food)
- refrigerator (an appliance in which foods can be stored at low temperature)

eXtended WordNet

Defining Features

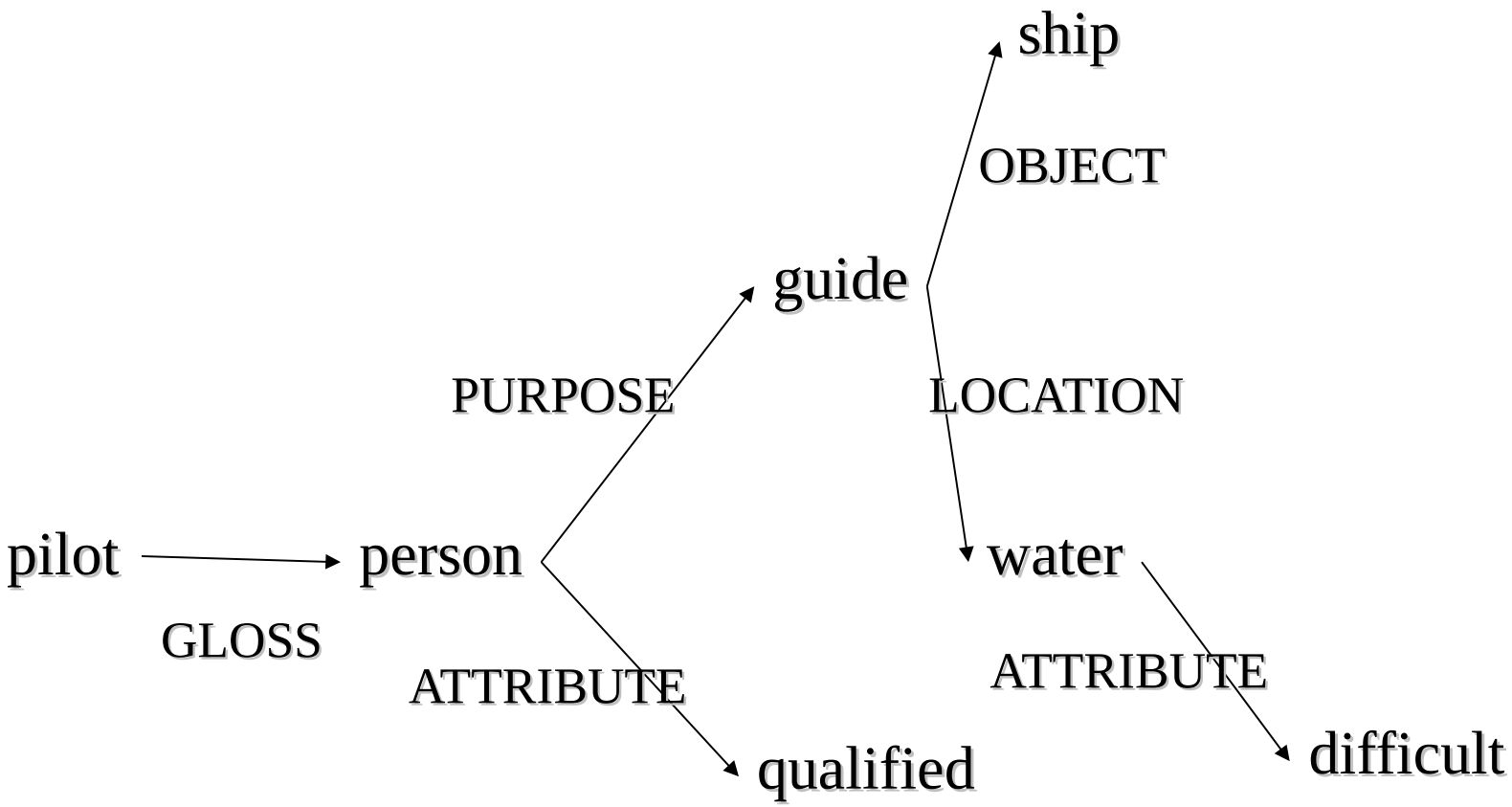
- Transform each concept's gloss into a graph where concepts are nodes and lexical relations are links

- <culture> (all the knowledge shared by society) =>
  <share> --AGENT--> <society>

- <doctor> (licensed medical practitioner) =>
  <medical practitioner> --ATRIBUTTE-->
  <licensed>

eXtended WordNet

Defining Features



eXtended WordNet

Inference rules

Rule 1

VC1	IS-A	VC2
VC2	IS-A	VC3
---	---	---
VC1	IS-A	VC3

Rule 2

VC1	IS-A	VC2
VC2	ENTAIL	VC3
---	---	---
VC1	ENTAIL	VC3

Rule 3

VC1	IS-A	VC2
VC2	R_IS-A	VC3
---	---	---
VC1	PLAUSIBLE	(not VC3)

Rule 2

VC1	IS-A	VC2
VC2	R_ENTAIL	VC3
---	---	---
VC1	EXPLAINS	VC3

- 16 + 1 rules

eXtended WordNet

Semantic Paths

1) Create and load the KB

2) Place markers on KB concepts

3) Propagate markers. The algorithm avoids cycles

4) Detect collisions. To each marker collision it corresponds a path

5) Extract Inferences

eXtended WordNet

Semantic Paths

Inference sequence

- German was hungry
- German felt a desire to eat
- German felt a desire to take in food

COLLISION: German=he felt a desire to take food, stored in an appliance, which he opened

- He opened an appliance where food is stored
- He opened the refrigerator

eXtended WordNet

What WN cannot do

Major WordNet limitations:

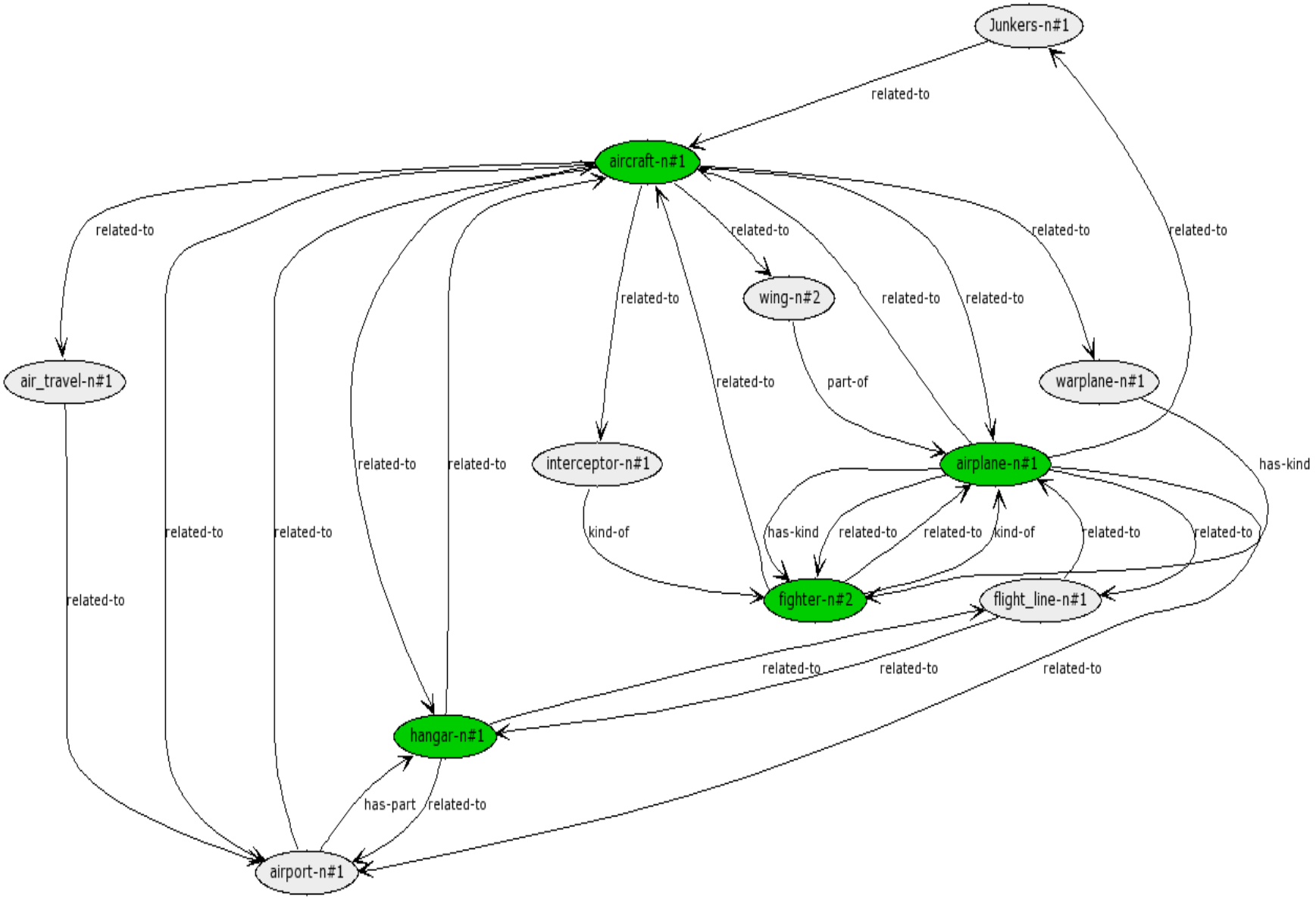
1) The lack of compound concepts

2) The small number of causation and entailment relations

3) The lack of preconditions for verbs

4) The absence of case relations

Graph-based Reasoning



Graph-based Reasoning

- SSI (Navigli & Velardi 2005)
- SSI-Dijkstra (Cuadros & Rigau 2008)
- UKB (Soroa & Agirre 2009)
- SSI-Dijkstra+ (Laparra & Rigau 2010)

MCR and consistency checking

00536235n blow &%Breathing+ anatomy
00005052v blow &%Breathing+ medicine

00003430v exhale &%Breathing+ biology
00003142v exhale &%Breathing+ medicine
00899001a exhaled &%Breathing+ factotum
00263355a exhaling &%Breathing+ factotum

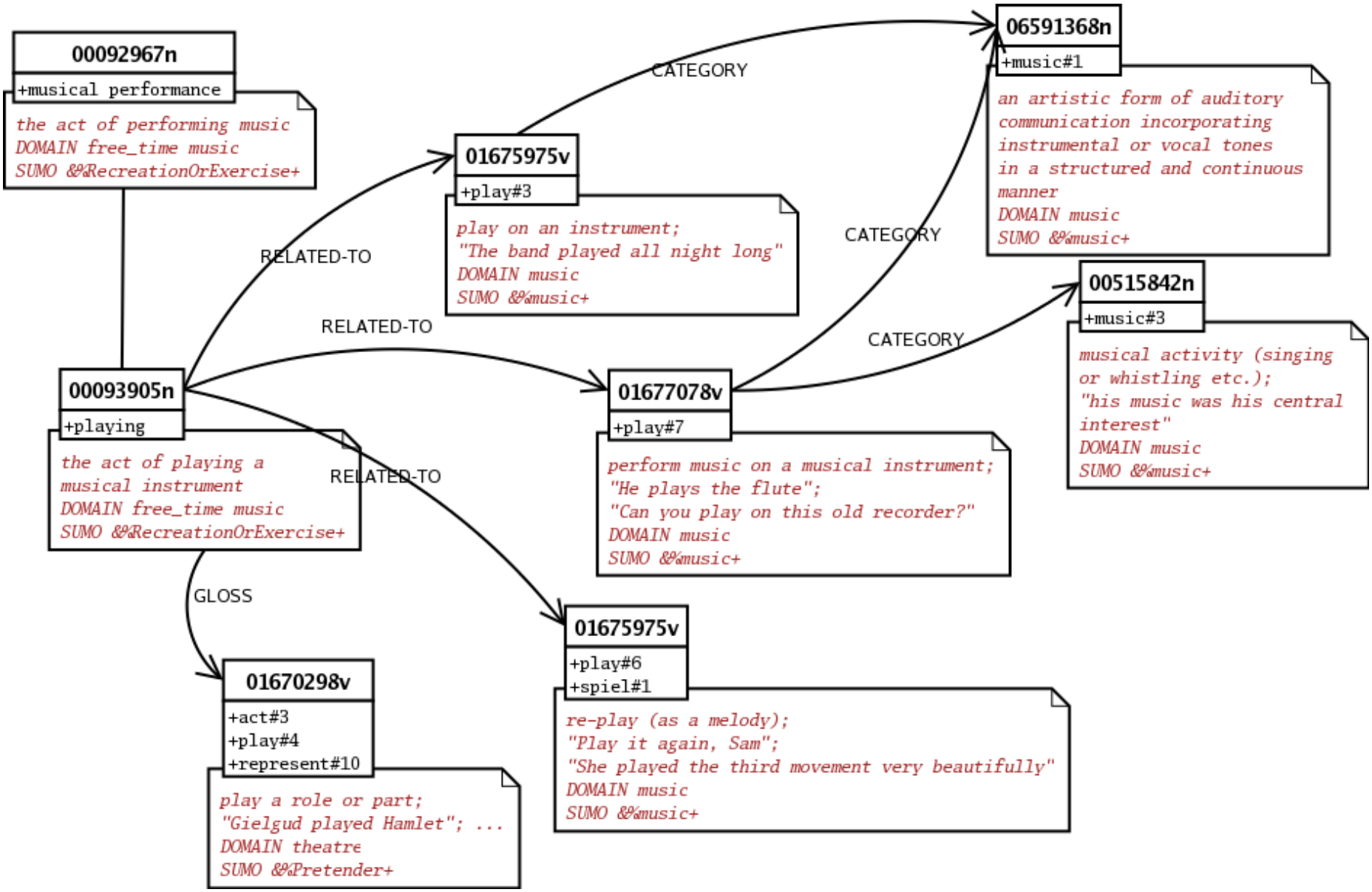
00536039n expiration &%Breathing+ anatomy
02849508a expiratory &%Breathing+ anatomy
00003142v expire &%Breathing+ medicine

02579534a inhalant &%Breathing+ anatomy
00536863n inhalation &%Breathing+ anatomy
00003763v inhale &%Breathing+ medicine
00898664a inhaled &%Breathing+ factotum
00263512a inhaling &%Breathing+ factotum

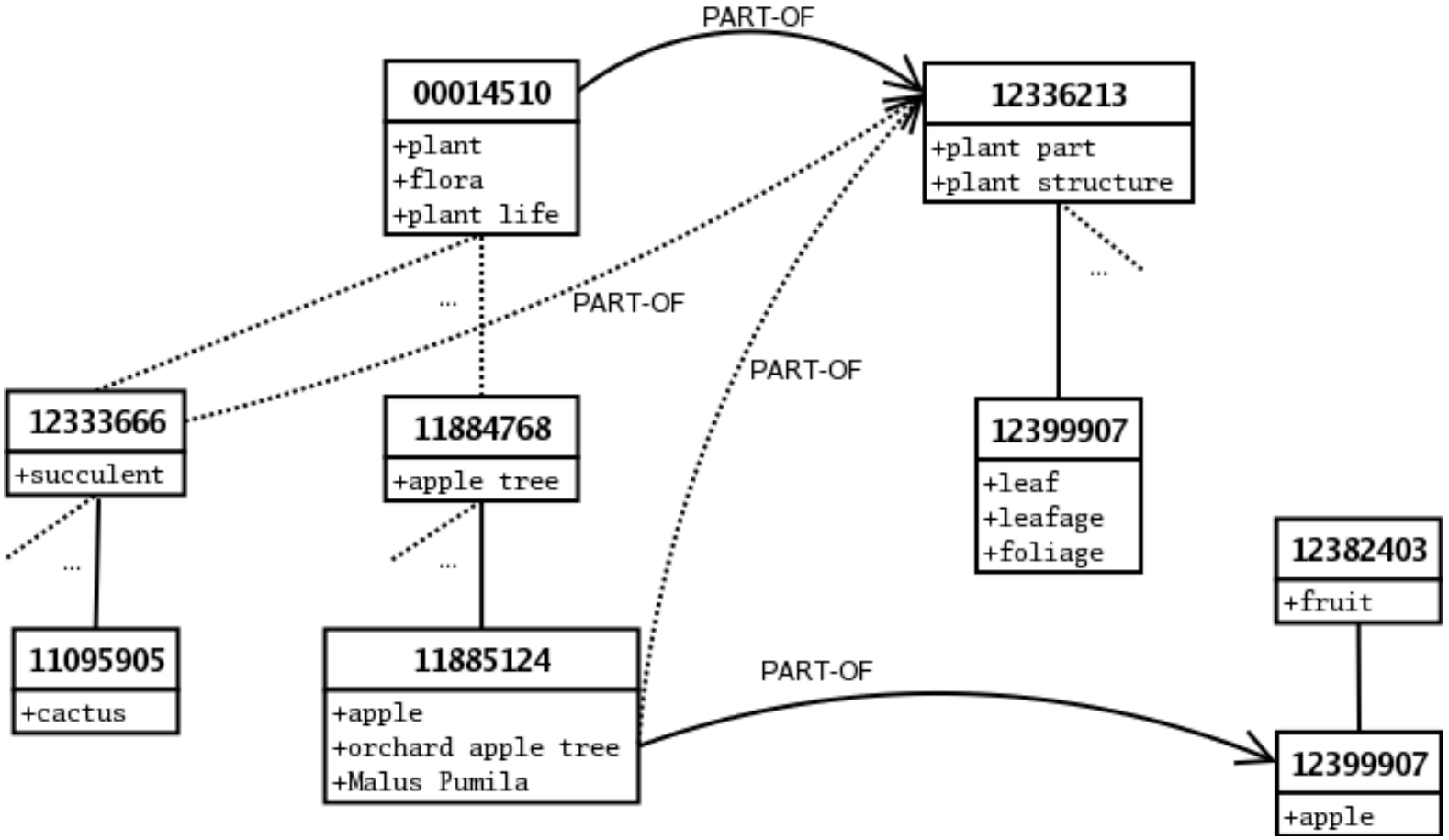
00537041n pant &%Breathing+ anatomy
00004002v pant &%Breathing+ medicine
00535106n panting &%Breathing+ anatomy
00264603a panting &%Breathing+ factotum
00411482r pantingly &%Breathing+ factotum

...

MCR and consistency checking



MCR and consistency checking



- Does an orchard apple tree have leaves?
- Does an orchad apple tree have fruits?
- Does a cactus have leaves?

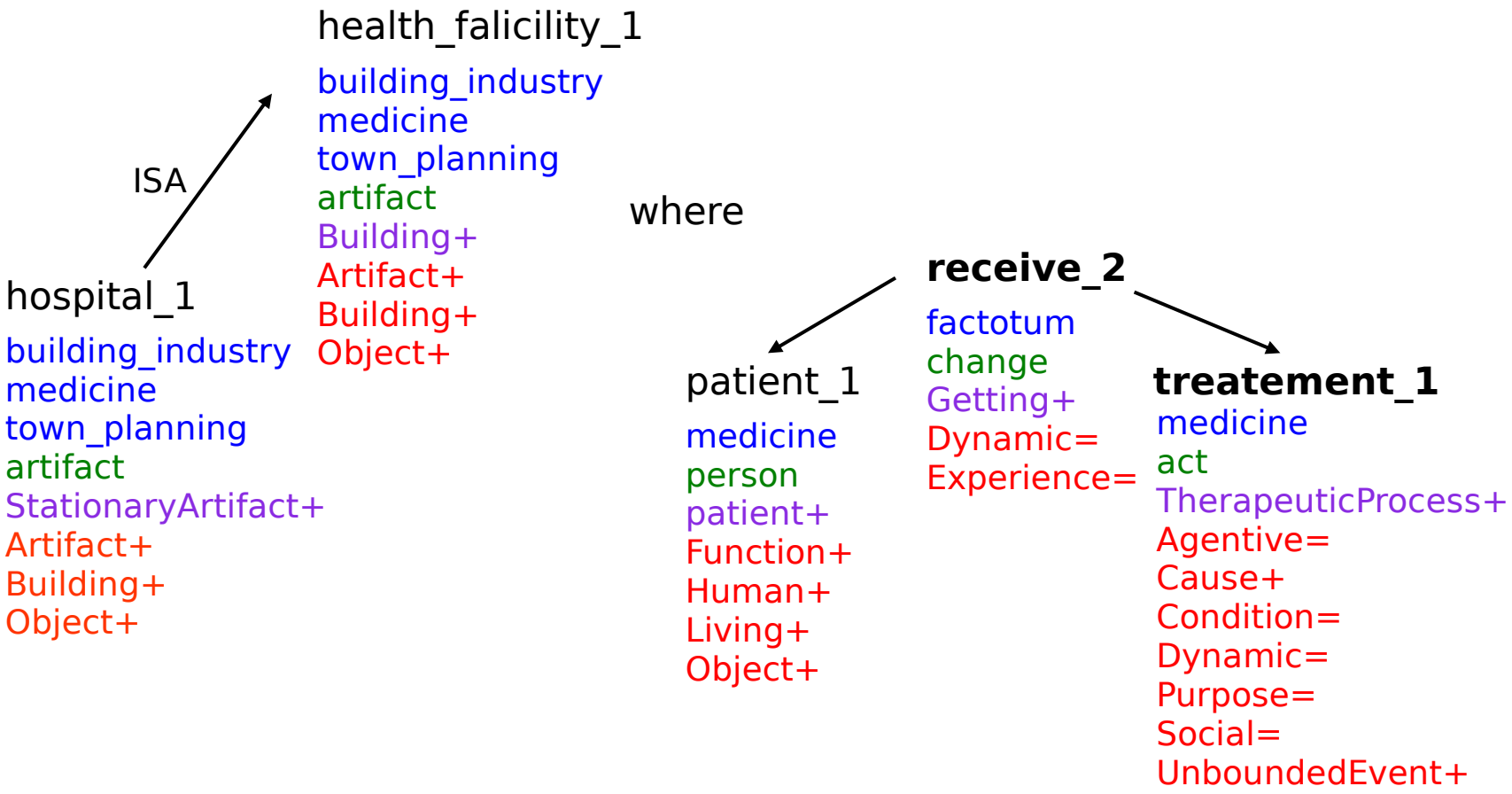
MCR and consistency checking

Example SUMO: TherapeuticProcess

- (documentation TherapeuticProcess EnglishLanguage "A process that is carried out for the purpose of curing, improving or reducing the pain associated with a disease or syndrome.")
- (subclass TherapeuticProcess Repairing)
- (=>
  (instance ?M Medicine)
  (exists (?P)
  (and
  (instance ?P TherapeuticProcess)
  (instrument ?P ?M))))
- If an object is an instance of medicine
  then there exists a process so that process is an instance of therapeutic process and object is an instrument for process

Reasoning with the MCR

hospital_1 a health facility where patients receive treatment



Reasoning with the MCR

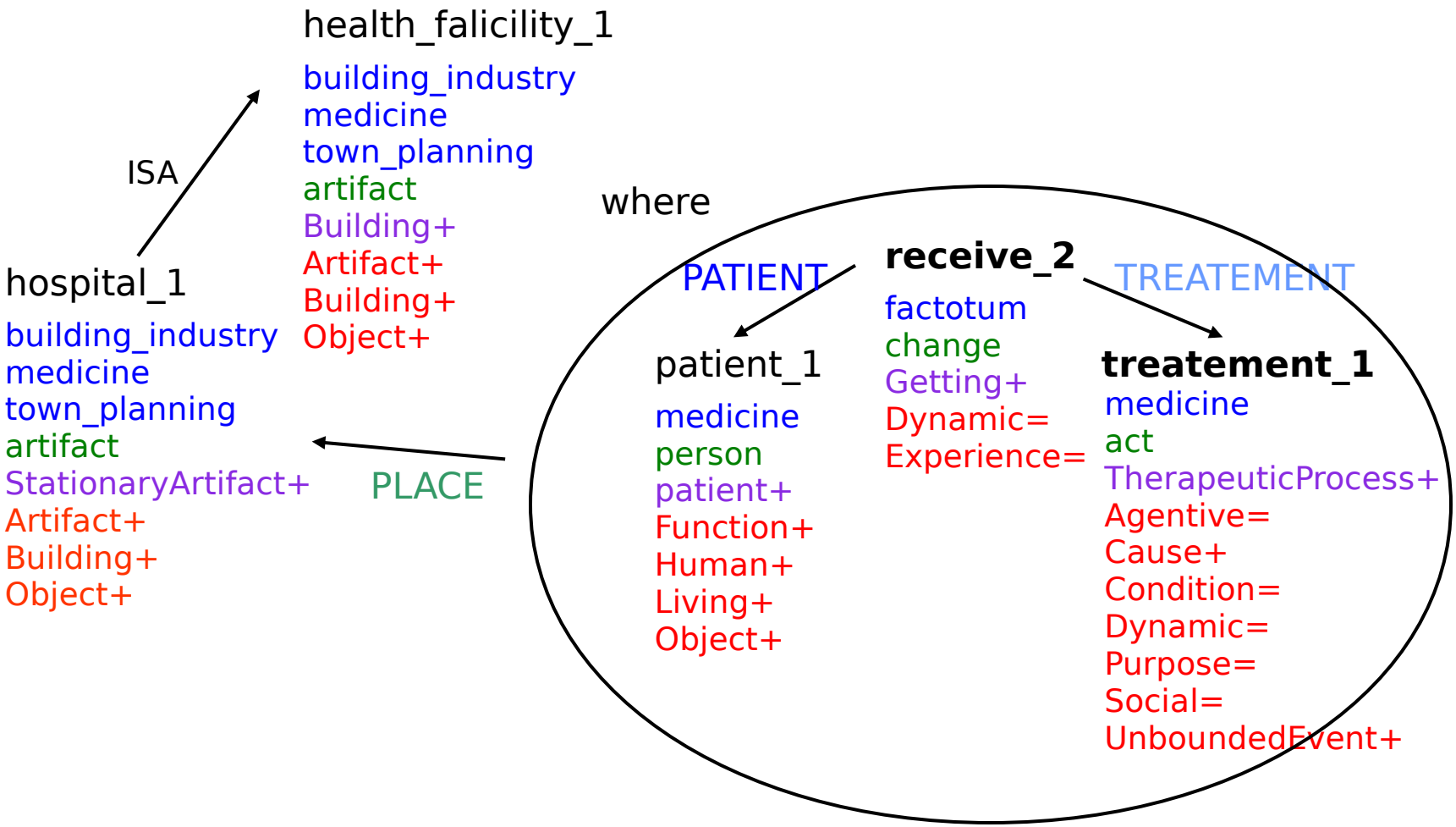
FRAMENET: cure.n

Frame Elements	Core Type
Affliction	Core
Body_part	Core
Degree	Peripheral
Duration	Extra-Thematic
Healer	Core
Manner	Peripheral
Medication	Core
Motivation	Extra-Thematic
Patient	Core
Place	Peripheral
Purpose	Extra-Thematic
Time	Peripheral
Treatment	Core

Reasoning with the MCR

hospital_1 a health facility where patients receive treatment

PLACE PATIENT TREATEMENT



Reasoning

- Reasoning with formal ontologies
  - Using a simple Prolog program
  - Top Concept Ontology
    - Consistent mapping to WN1.6 (Álvez et al. 08)
    - http://lpg.uoc.edu/files/wei-topontology.2.2.rar
  - Using FOL theorem provers like Vampire, E-prover, ...
  - SUMO does not work on FOL theorem provers directly BUT
  - We can work with a subset of SUMO including
    - the main hierarchy
      - structural properties, subclass, instance, etc.
    - Mapping to WN1.6

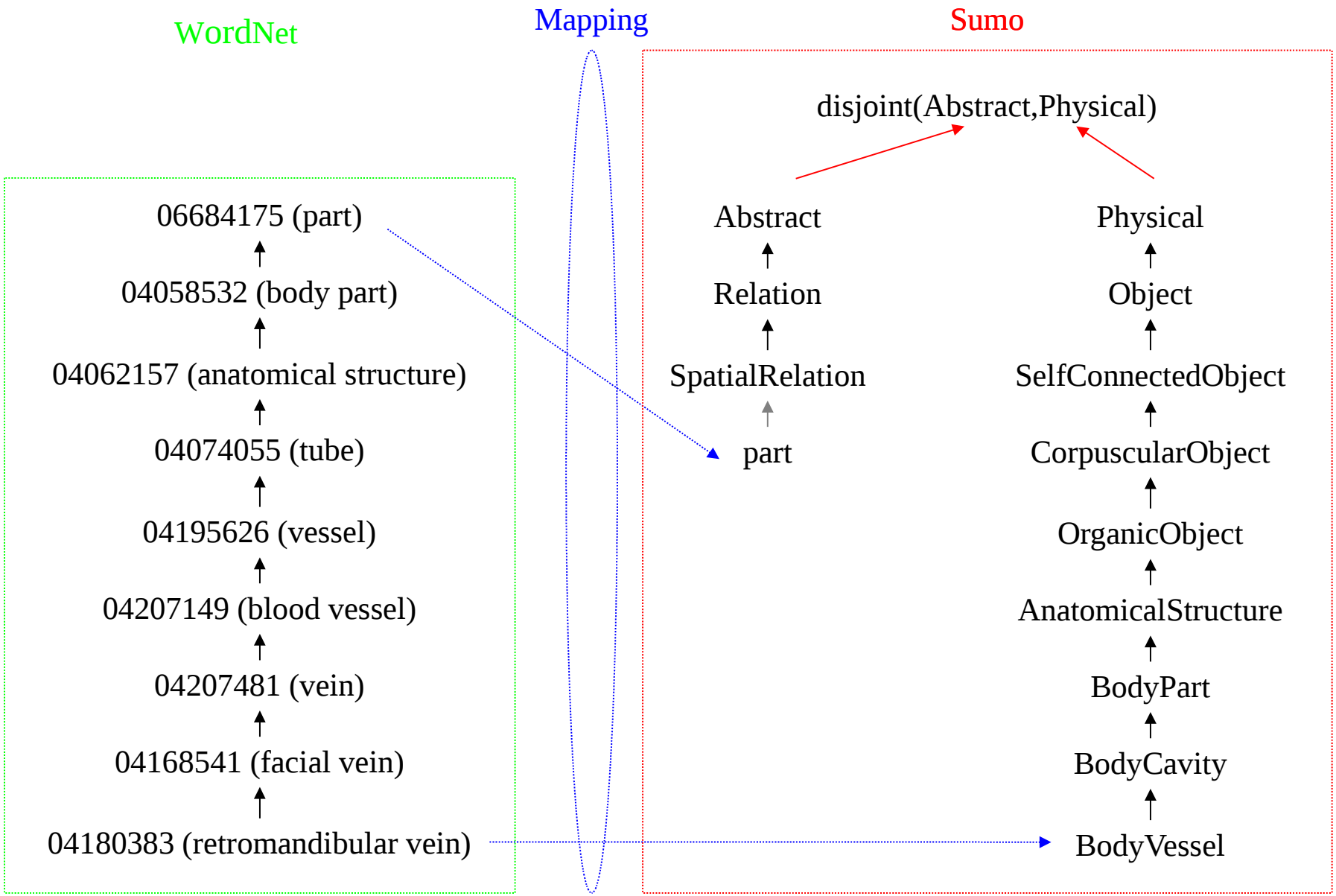
MCR and consistency checking

- body_covering_1
  - skin_4
  - plumage_1 feather_1
    - down_1
    - sickle_feather_1
  - protective_covering_2
  - skin_1
    - pellicle_1
    - dewlap_1
    - prepuce_2
    - scalp_1
    - animal_skin_1
      - parchment_2
      - leather_1
        - piece_of_leather_1
          - heel_4
          - toe_2
        - cordovan_1
      - fur_1
        - bearskin_1
        - lapin_1
  - hair_1
    - coat_3
    - hairball_2
    - mane_1
    - beard_3
    - postiche_1
    - hairdo_1
      - afro_1
    - pubic_hair_1
    - eyebrow_1
    - eyelash_1

MCR and consistency checking

{body_covering_1 [*Living= Part= Covering=*]}
--- {skin_4 pelt_2 [*Living+ Part+ Covering+ Object=*]}
--- {plumage_1 feather_1 [*Living:Animal= Part+ Covering+ Substance:Solid=*]}
--- {down_1 [*Living:Animal+ Part+ Covering+ Substance:Solid+*]}
-x- {sickle_feather_1 [*Living:Animal= Part= Covering= Object=*]}
--- {protective_covering_2 [*Living+ Part+ Covering+ Object=*]}
--- {skin_1 tegument_1 [*Living+ Part+ Covering+ Substance:Solid =*]}
--- {pellicle_1 [*Living+ Part+ Covering+ Substance:Solid =*]}
-x- {dewlap_1 [*Object= Living:Animal= Part=*]}
-x- {prepuce_2 [*Object= Living:Animal= Part=*]}
-x- {scalp_1 [*Object= Living:Animal= Part=*]}
--- {animal_skin_1 [*Living+ Part+ Covering+ Substance:Solid =*]}
-x- {parchment_2 [*Substance:Solid= Artifact=*]}
-x- {leather_1 [*Substance:Solid= Artifact=*]}
-x- {piece_of_leather_1 [*Object= Artifact=*]}
--- heel_4 [*Object+ Artifact+ Garment= Part=*]}
--- toe_2 [*Object+ Artifact+ Garment= Part=*]}
--- {cordovan_1 [*Substance:Solid+ Artifact+*]}
-x- {fur_1[*Object= Artifact=*]}
--- {bearskin_1 [*Object+ Artifact+*]}
--- {lapin_1 [*Object+ Artifact+*]}
--- {hair_1 [*Living+ Part+ Covering+ Substance:Solid=*]}
--- {coat_3 [*Living+ Part+ Covering+ Substance:Solid=*]}
-x- {hairball_2 [*Object= Living=*]
-x- {mane_1 [*Object= Living:Animal= Part=*]
-x- {beard_3 [*Object= Living:Animal= Part= Covering=*]
-x- {postiche_1 [*Object+ Artifact+ Covering+ Garment+*][1]
------> {disguise_2}
-x- {hairdo_1 [*Property= Manner=*]}
--- afro_1 [*Property+ Manner+*]}
--- {pubic_hair_1 [*Living+ Part+ Covering+ Substance:Solid+*]}
-x- {eyebrow_1 [*Object= Living:Human= Part=*]}
-x- {eyelash_1 [*Object= Living= Part=*]}

Reasoning with SUMO



Reasoning with SUMO

- Firsts experiments with SUMO
- Meeting with Adam Pease
- Debugging SUMO
- First results ...
  - AdimenSUMO

Reasoning with SUMO

- The beginning: Sigma Knowlege Engineering Environment
- Ask option did not reason properly
- How did it work?
  - Translate SUMO from KIF to TPTP
  - An *ad hoc* version of Vampire
- Drawbacks:
  - Many features of SUMO were not correctly translated
  - No possibility of using another ATP's

Meeting with Adam Pease

- SUMO & FO automatic provers (Vampire, E Prover)
- Special features of SUMO (non-FO features)
- Types, functions, temporal formulas
- Consistency
- Decidability

First attempts

- A first translator: syntactic translation

```
(=>
    (instance ?RESIDENCE TemporaryResidence)
    (not
        (exists (?PERSON)
            (home ?PERSON ?HOTEL))))
```

- More sophisticated translation: types, row variables, schemas, ...
  - Do plants have brain? FO-automatic provers did not answer NO!!!

Main problems of SUMO

- SUMO is auto-defined
- SUMO does not distinguish classes and meta-classes
- In combination with type information, this fact blocks inferencing
- However, if we remove type information, many “inconsistencies” arise
- Missing information (lists, etc.)

Reasoning with SUMO

- Decidability problems with FOL Theorem Provers BUT ...
- A first-order theory is decidable if
  - the number of constants (0-arity functions) is finite
  - the number of non-recursive functions (of arity n>0) is finite
  - we consider the Closed World Assumption (maybe many-sorted)

- Automatic Theorem Provers (E Prover):
  - eprover -xAuto -tAuto --tstp-in sumoWN.tstp

Another experiment

- Obtain a complete theory
- Ask to the ATP's both the goal and the negated goal (Prole 08)
- The system answers to every query
- Drawbacks:
  - It is very difficult to define a complete theory
  - Current ATP's are not suitable for this kind of reasoning
  - Scalability

Our current proposal

- Transform SUMO into a decidable theory: Closure
- Use a very simple translation
- Build an *ad hoc* theorem prover for large FOL ontologies
- Result: the *ad hoc* theorem prover is able to answer every query

Reasoning with SUMO

```
(not
   (and (instance ?BRAIN Brain)
        (instance ?PLANT Plant)
        (part ?BRAIN ?PLANT)))

(=> (and (subclass ?X ?Y)
         (instance ?Z ?X))
    (instance ?Z ?Y))

(partition4 Organism Animal Plant Microorganism)

(subclass Brain AnimalAnatomicalStructure)

(subclass Plant Organism)

(=> (and (instance ?STRUCTURE AnimalAnatomicalStructure)
         (instance ?ANIMAL Organism)
         (part ?STRUCTURE ?ANIMAL))
    (instance ?ANIMAL Animal))
```

Reasoning with SUMO

- Does a plant have a brain?

- Using E-prover (or Vampire, etc.)

1 : conj : ![X1]:![X2]:((instance(X2,object)&instance(X1,object))=>~(((instance(X2,brain)&
instance(X1,plant))&part(X2,X1)))) : initial(‘‘brain.eprover.tstp’’, goal)

…

444 : neg : [] : 443 : ’proof’

E-prover

```
 1 : conj : ![X1]:![X2]:((instance(X2,object)&instance(X1,object))=>~(((instance(X2,brain)&
            instance(X1,plant))&part(X2,X1)))) : initial(''brain.eprover.tstp'', goal)
 5 : : ![X3]:![X4]:![X5]:((instance(X3,X4)&subclass(X4,X5))=>instance(X3,X5)) :
                                                  initial(''brain.eprover.tstp'', predefinitionsB4)
 6 : : ![X6]:![X7]:(disjoint(X6,X7)<=>![X8]:~((instance(X8,X6)&instance(X8,X7)))) :
                                                  initial(''brain.eprover.tstp'', predefinitionsB5)
 7 : : ![X9]:![X10]:![X11]:![X12]:(partition4(X9,X10,X11,X12)<=>(exhDecomp4(X9,X10,X11,X12)&
            disDecomp4(X9,X10,X11,X12))) : initial(''brain.eprover.tstp'', predefinitionsB6)
 9 : : ![X9]:![X10]:![X11]:![X12]:(disDecomp4(X9,X10,X11,X12)<=>((disjoint(X10,X11)&
            disjoint(X10,X12))&disjoint(X11,X12))) : initial(''brain.eprover.tstp'', predefinitionsB8)
10 : : ![X13]:![X14]:((instance(X13,object)&instance(X14,object))=>(((instance(X13,animalAS)&
            instance(X14,organism))&part(X13,X14))=>instance(X14,animal))) :
                                                  initial(''brain.eprover.tstp'', merge158B1)
11 : : subclass(brain,animalAS) : initial(''brain.eprover.tstp'', milo184B1)
17 : : subclass(plant,organism) : initial(''brain.eprover.tstp'', merge158B7)
20 : : partition4(organism,animal,plant,microorganism) : initial(''brain.eprover.tstp'', merge158B10)
21 : neg : ~(![X1]:![X2]:((instance(X2,object)&instance(X1,object))=>~(((instance(X2,brain)&
            instance(X1,plant))&part(X2,X1))))) : assume_negation(1)
...
25 : neg : [++part(esk2_0,esk1_0)] : split_conjunct(24)                          *** from (21)
26 : neg : [++instance(esk1_0,plant)] : split_conjunct(24)                       *** from (21)
27 : neg : [++instance(esk2_0,brain)] : split_conjunct(24)                       *** from (21)
28 : neg : [++instance(esk1_0,object)] : split_conjunct(24)                      *** from (21)
29 : neg : [++instance(esk2_0,object)] : split_conjunct(24)                      *** from (21)
...
40 : : [++instance(X1,X2),--subclass(X3,X2),--instance(X1,X3)] : split_conjunct(39)   *** from (5)
...
48 : : [--disjoint(X1,X2),--instance(X3,X2),--instance(X3,X1)] : split_conjunct(45)   *** from (6)
...
53 : : [++disDecomp4(X1,X2,X3,X4),--partition4(X1,X2,X3,X4)] : split_conjunct(51)    *** from (7)
...
71 : : [++disjoint(X2,X3),--disDecomp4(X1,X2,X3,X4)] : split_conjunct(67)        *** from (9)
...
74 : : [++instance(X1,animal),--part(X2,X1),--instance(X1,organism),--instance(X2,animalAS),
            --instance(X1,object),--instance(X2,object)] : split_conjunct(24)    *** from (10)
75 : : [++subclass(brain,animalAS)] : split_conjunct(11)
81 : : [++subclass(plant,organism)] : split_conjunct(17)
84 : : [++partition4(organism,animal,plant,microorganism)] : split_conjunct(20)
...
157 : : [++instance(X1,animalAS),--instance(X1,brain)] : spm(156,125)            *** from (40,75)
158 : : [++instance(X1,organism),--instance(X1,plant)] : spm(156,126)            *** from (40,81)
...
183 : neg : [++instance(X1,animal),--part(esk2_0,X1),--instance(esk2_0,animalAS),
            --instance(X1,object),--instance(X1,organism)] : spm(181,122)        *** from (74,29)
...
215 : : [++disDecomp4(organism,animal,plant,microorganism)] : spm(214,135)       *** from (53,84)
...
239 : neg : [++instance(esk2_0,animalAS)] : spm(238,123)                         *** from (157,27)
...
249 : neg : [++instance(esk1_0,organism)] : spm(248,121)                         *** from (158,26)
...
340 : : [++disjoint(animal,plant)] : spm(180,337)                                *** from (71,215)
...
373 : : [--instance(X1,plant),--instance(X1,animal)] : spm(176,372)              *** from (48,340)
...
387 : neg : [--instance(esk1_0,animal)] : spm(386,121)                           *** from (373,26)
...
436 : neg : [++instance(X1,animal),--part(esk2_0,X1),--$true,--instance(X1,object),
            --instance(X1,organism)] : rw(187,247)                               *** from (183,239)
...
439 : neg : [++instance(esk1_0,animal),--instance(esk1_0,object),--instance(esk1_0,organism)] :
                                                  spm(438,124)                   *** from (436,25)
440 : neg : [++instance(esk1_0,animal),--$true,--instance(esk1_0,organism)] : rw(439,120)  *** from (439,28)
441 : neg : [++instance(esk1_0,animal),--$true,--$true] : rw(440,257)             *** from (440,249)
442 : neg : [++ instance(esk1_0,animal)] : cn(441)
443 : neg : [] : sr(442,396)                                                     *** from (442,387)
444 : neg : [] : 443 : 'proof'
```

Reasoning with SUMO

- SUMO does not know if animals have lungs
- SUMO does not know what kind of animals have lungs
- SUMO knows that if an organism has lungs, then it is a animal
- SUMO knows that if a fish or an animal breathes, then it must have lungs.
- SUMO does not know if a fish or animal breathes.
- SUMO does not know if an animal has lungs or not.
- SUMO knows that a fish can not breathe without lungs.
- SUMO thinks that a table can have lungs.
- SUMO believes that a male (man) cannot be pregnant or lay eggs.
- SUMO believes that urine is solid
- SUMO knows that plants can not have a headache.

Reasoning in KYOTO

- Three layered knowledge Representation
  - Vocabulary: lookup table (database)
  - Concepts: wordnets
  - Types: OWL-DL ontology
- The three layers are interconected
- Each layer has different inferencing mechanisms

Reasoning in KYOTO

- Vocabulary
  - Millions of terms and concepts
    - Species 2000
    - Geonames, DBpedia, wikipedia, etc.
  - Aligned automatically to WN3.0
    - SSI-Dijkstra (Cuadros and Rigau 2008)
  - Manual selecion
    - Base Concepts (Izquierdo et al. 2007)

Reasoning in KYOTO: example

- “cape teal”
- “anas capensis”
- “Yellow-billed Pintail”
- …

File Edit View History Bookmarks Tools Help

http://en.wikipedia.org/wiki/Anas

Most Visited Release Notes Error Fedora Project Red Hat Free Content

Anas - Wikipedia, the free ... | linux which java installed - ... | Pellet 2.0.1 Release Downl... | Gmail - Inbox (4524) - griga...

Try Beta Log in / create account

article discussion edit this page history

Anas

From Wikipedia, the free encyclopedia

For other uses, see Anas (disambiguation).

Anas is a genus of dabbling ducks. It includes mallards, wigeons, teals, pintails and shovelers in a number of subgenera. Some authorities prefer to elevate the subgenera to genus rank[1]. Indeed, as the moa-nalos are very close to this clade and may have evolved later than some of these lineages, it is rather the absence of a thorough review than lack of necessity that this genus is rather over-lumped.

Contents [hide]

Systematics [edit]

The phylogeny of this genus is one of the most confounded ones of all living birds. Research is hampered by the fact the radiation of the two major groups of *Anas* - the teals and mallard groups -; took place in a very short time and fairly recently, roughly in the mid-late Pleistocene. Furthermore, hybridization may have long played a major role in *Anas* evolution, with within-subgenus hybrids regularly and between-subgenus hybrids not infrequently being fully fertile[1] see also Mariana Mallard. The relationships between species are much obscured by this fact, and mtDNA sequence data is of dubious value in resolving their relationships[2]; on the other hand, nuclear DNA sequences evolve too slowly to resolve the phylogeny of the subgenus *Anas* for example.

Some major clades can be discerned. For example, that the traditional subgenus *Anas*, the mallard group, forms a

Anas

Female Mallard (*Anas platyrhynchos*) with brood of young, a typical member of this genus.

Scientific classification

Kingdom:	Animalia
Phylum:	Chordata
Class:	Aves
Subclass:	Neornithes
Infraclass:	Neognathae
Superorder:	Galloanserae
Order:	Anseriformes

navigation
- Main page
- Contents
- Featured content
- Current events
- Random article

search
Go Search

interaction
- About Wikipedia
- Community portal
- Recent changes
- Contact Wikipedia
- Donate to Wikipedia
- Help

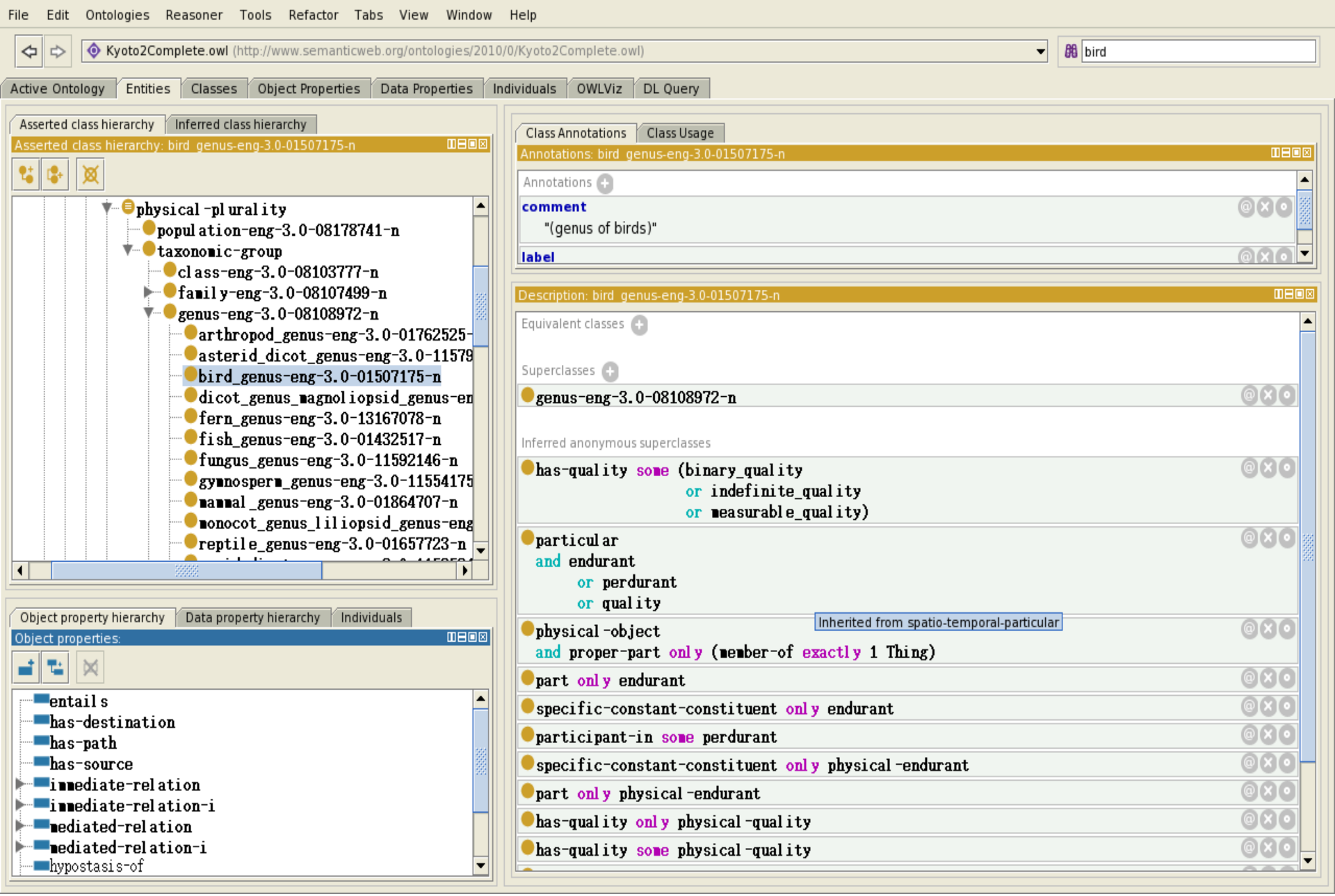
toolbox
- What links here
- Related changes
- Upload file
- Special pages

Done

Reasoning

Reasoning in KYOTO: example

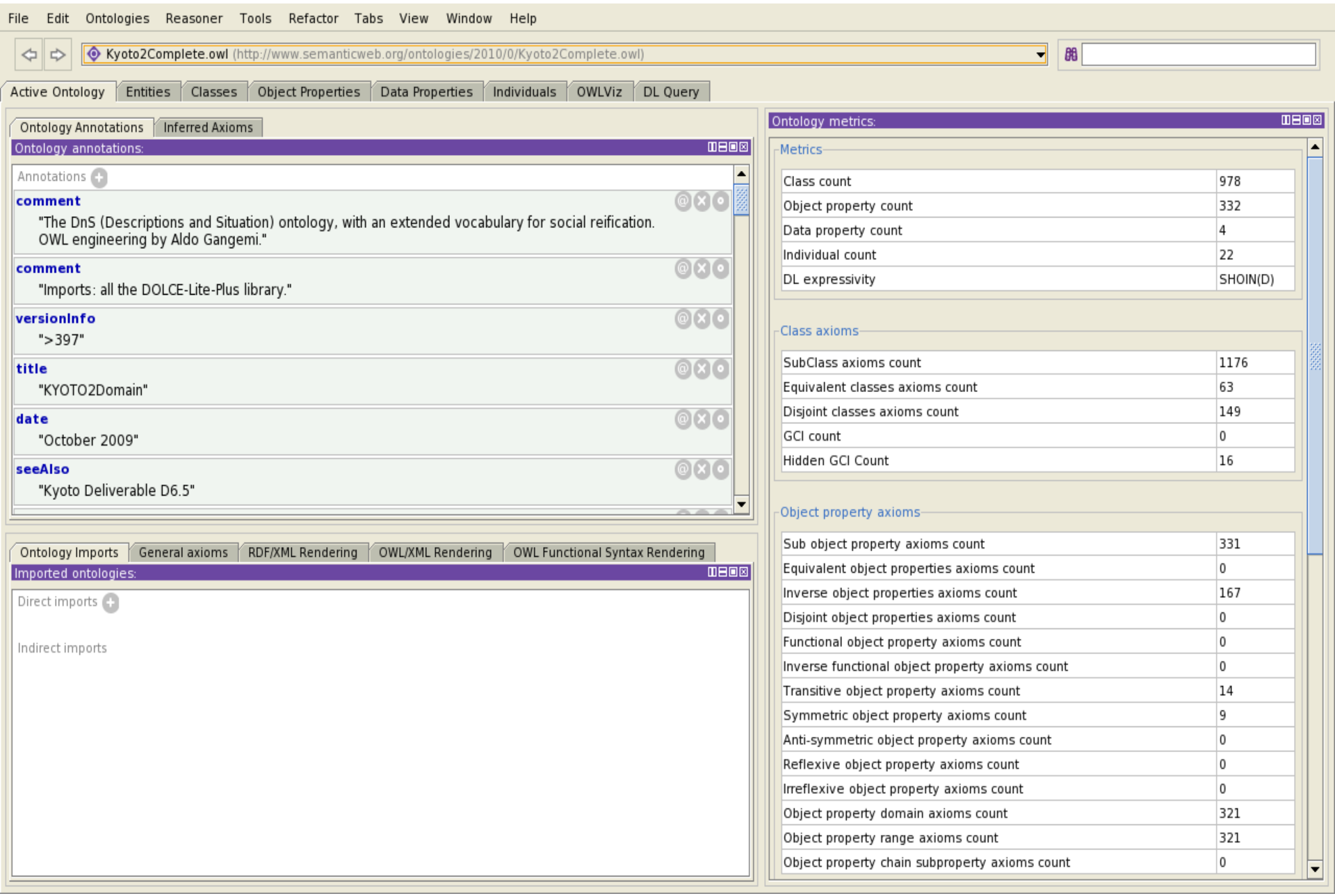
- 268 Anas in Species 2000 concepts
- Animalia/Chordata/Aves/Anseriformes/Anatidae/Anas/ITS-175103 : Yellow-billed Pintail
- eng-3.0-01847565-n <Anas, genus Anas>
- 297 WN3.0 Base Concepts
  - 01507175-n 05 399 bird_genus
- Connected to KYOTO ontology
  - bird_genus-eng-3.0-01507175-n type


File Edit Ontologies Reasoner Tools Refactor Tabs View Window Help
Kyoto2Complete.owl (http://www.semanticweb.org/ontologies/2010/0/Kyoto2Complete.owl)
bird
Active Ontology Entities Classes Object Properties Data Properties Individuals OWLViz DL Query
Asserted class hierarchy Inferred class hierarchy
Asserted class hierarchy: bird_genus-eng-3.0-01507175-n
physical-plurality
population-eng-3.0-08178741-n
taxonomic-group
class-eng-3.0-08103777-n
family-eng-3.0-08107499-n
genus-eng-3.0-08108972-n
arthropod_genus-eng-3.0-01762525-
asterid_dicot_genus-eng-3.0-11579
bird_genus-eng-3.0-01507175-n
dicot_genus_magnoliopsid_genus-en
fern_genus-eng-3.0-13167078-n
fish_genus-eng-3.0-01432517-n
fungus_genus-eng-3.0-11592146-n
gymnosperm_genus-eng-3.0-11554175
mammal_genus-eng-3.0-01864707-n
monocot_genus_liliopsid_genus-eng
reptile_genus-eng-3.0-01657723-n
Object property hierarchy Data property hierarchy Individuals
Object properties:
entails
has-destination
has-path
has-source
immediate-relation
immediate-relation-i
mediated-relation
mediated-relation-i
hypostasis-of
Class Annotations Class Usage
Annotations: bird_genus-eng-3.0-01507175-n
Annotations
comment
"(genus of birds)"
label
Description: bird_genus-eng-3.0-01507175-n
Equivalent classes
Superclasses
genus-eng-3.0-08108972-n
Inferred anonymous superclasses
has-quality some (binary_quality
or indefinite_quality
or measurable_quality)
particular
and endurant
or perdurant
or quality
Inherited from spatio-temporal-particular
physical-object
and proper-part only (member-of exactly 1 Thing)
part only endurant
specific-constant-constituent only endurant
participant-in some perdurant
specific-constant-constituent only physical-endurant
part only physical-endurant
has-quality only physical-quality
has-quality some physical-quality

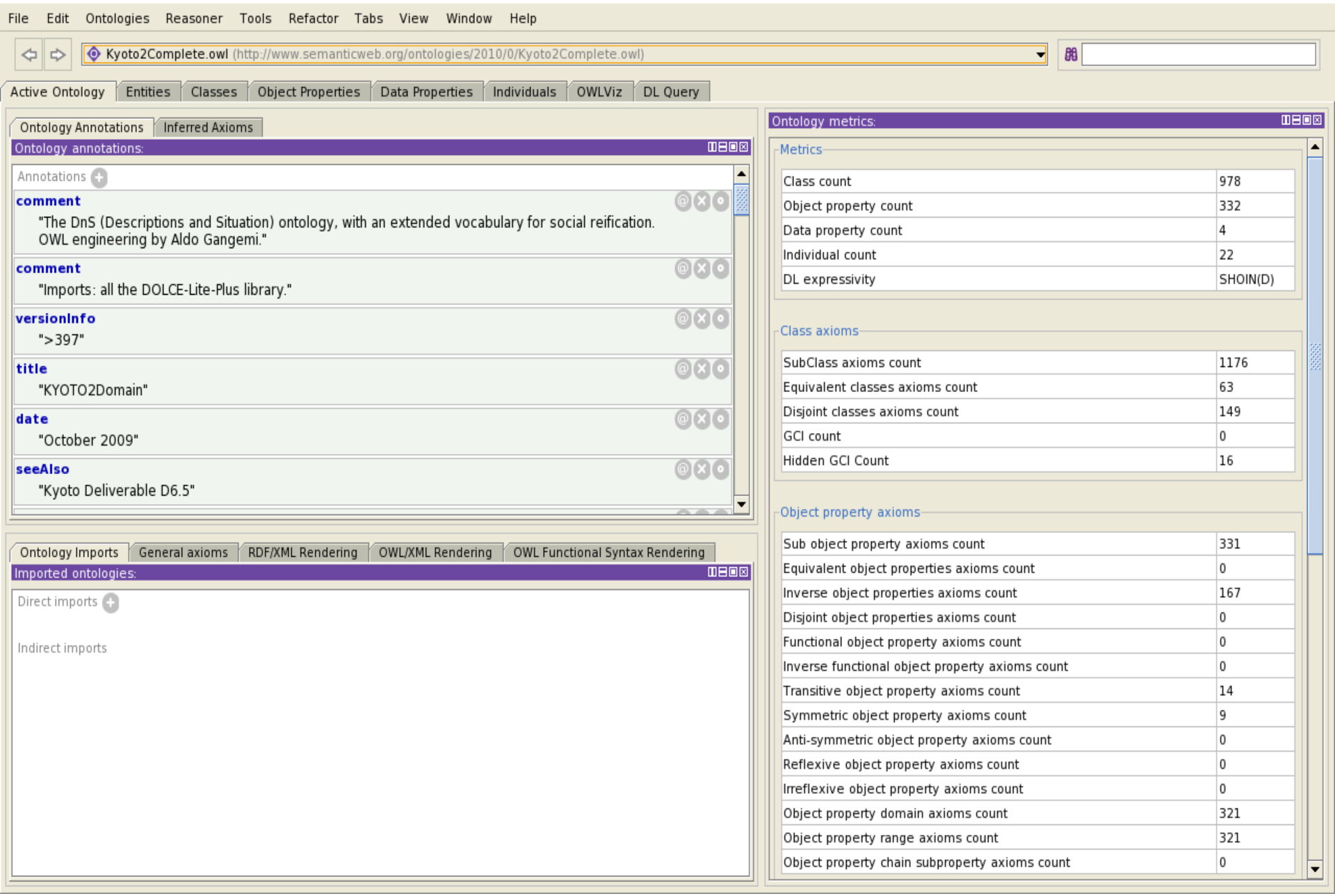
Reasoning

Reasoning in KYOTO: types of Reasoning

- Three layered knowledge Representation
  - Vocabulary:
    - minimal reasoning
  - Concepts: wordnets:
    - Graph-based, similarity, etc.
  - Types: OWL-DL ontology
    - Protégé
    - Formal reasoning: Fact++, Pellet, etc.


File Edit Ontologies Reasoner Tools Refactor Tabs View Window Help
Kyoto2Complete.owl (http://www.semanticweb.org/ontologies/2010/0/Kyoto2Complete.owl)
Active Ontology Entities Classes Object Properties Data Properties Individuals OWLViz DL Query
Ontology Annotations Inferred Axioms
Ontology annotations:
Annotations
comment
"The DnS (Descriptions and Situation) ontology, with an extended vocabulary for social reification. OWL engineering by Aldo Gangemi."
comment
"Imports: all the DOLCE-Lite-Plus library."
versionInfo
">397"
title
"KYOTO2Domain"
date
"October 2009"
seeAlso
"Kyoto Deliverable D6.5"
Ontology Imports General axioms RDF/XML Rendering OWL/XML Rendering OWL Functional Syntax Rendering
Imported ontologies:
Direct imports
Indirect imports
Ontology metrics:
Metrics
Class count 978
Object property count 332
Data property count 4
Individual count 22
DL expressivity SHOIN(D)
Class axioms
SubClass axioms count 1176
Equivalent classes axioms count 63
Disjoint classes axioms count 149
GCI count 0
Hidden GCI Count 16
Object property axioms
Sub object property axioms count 331
Equivalent object properties axioms count 0
Inverse object properties axioms count 167
Disjoint object properties axioms count 0
Functional object property axioms count 0
Inverse functional object property axioms count 0
Transitive object property axioms count 14
Symmetric object property axioms count 9
Anti-symmetric object property axioms count 0
Reflexive object property axioms count 0
Irreflexive object property axioms count 0
Object property domain axioms count 321
Object property range axioms count 321
Object property chain subproperty axioms count 0

Reasoning


File Edit Ontologies Reasoner Tools Refactor Tabs View Window Help
Kyoto2Complete.owl (http://www.semanticweb.org/ontologies/2010/0/Kyoto2Complete.owl)
Active Ontology Entities Classes Object Properties Data Properties Individuals OWLViz DL Query
Ontology Annotations Inferred Axioms
Ontology annotations:
Annotations
comment
"The DnS (Descriptions and Situation) ontology, with an extended vocabulary for social reification. OWL engineering by Aldo Gangemi."
comment
"Imports: all the DOLCE-Lite-Plus library."
versionInfo
">397"
title
"KYOTO2Domain"
date
"October 2009"
seeAlso
"Kyoto Deliverable D6.5"
Ontology Imports General axioms RDF/XML Rendering OWL/XML Rendering OWL Functional Syntax Rendering
Imported ontologies:
Direct imports
Indirect imports
Ontology metrics:
Metrics
Class count 978
Object property count 332
Data property count 4
Individual count 22
DL expressivity SHOIN(D)
Class axioms
SubClass axioms count 1176
Equivalent classes axioms count 63
Disjoint classes axioms count 149
GCI count 0
Hidden GCI Count 16
Object property axioms
Sub object property axioms count 331
Equivalent object properties axioms count 0
Inverse object properties axioms count 167
Disjoint object properties axioms count 0
Functional object property axioms count 0
Inverse functional object property axioms count 0
Transitive object property axioms count 14
Symmetric object property axioms count 9
Anti-symmetric object property axioms count 0
Reflexive object property axioms count 0
Irreflexive object property axioms count 0
Object property domain axioms count 321
Object property range axioms count 321
Object property chain subproperty axioms count 0

Reasoning

Reasoning in KYOTO: Using Pellet or Fact++

- **Classify** the ontology and display the hierarchy
- Check the **consistency** of an ontology
- **Explains** one or more inferences in a given ontology including ontology inconsistency
- **Extract** a set of inferences from an ontology
- SPARQL-DL **Query** Engine
- Find the **unsatisfiable** classes in the ontology
- ...

Reasoning

German Rigau i Claramunt

german.rigau@ehu.es

IXA group

Departamento de Lenguajes y Sistemas Informáticos

UPV/EHU