An Introduction to Description Logics

2. Reasoning Tasks

G. Falquet

Reasoning Tasks

- Consistency
- Subsumption
- Open world
- Unique name
- Instance checking

Consider the axioms

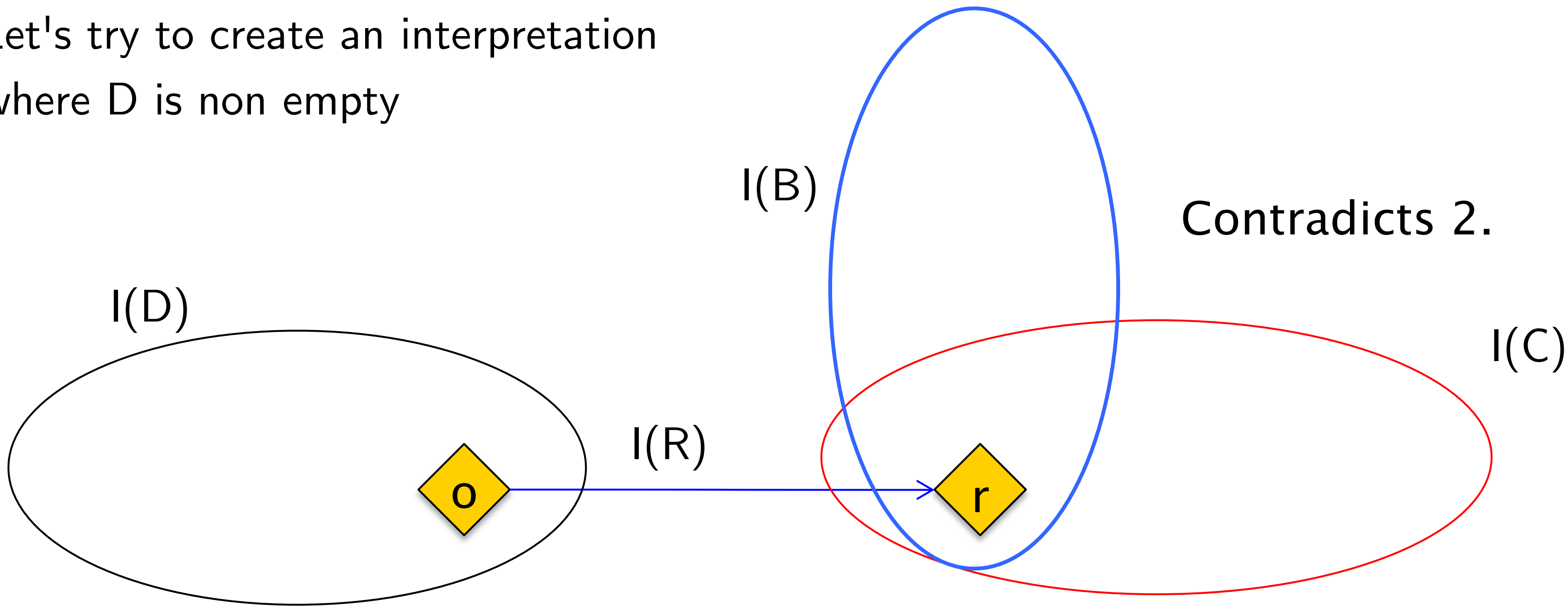
1. A $\sqsubseteq$ ($\forall$ R . B)
2. C disjoint B
3. D $\sqsubseteq$ (($\exists$ R . C) $\sqcap$ A)

Let's try to create an interpretation where D is non empty

Consider the axioms

1. A ⊑ (∀ R . B)
2. C disjoint B
3. D ⊑ ((∃ R . C) ⊓ A)

Let's try to create an interpretation where D is non empty



Consistency

- a knowledge base is consistent if there is an interpretation such that all the axioms are satisfied

- a concept C is consistent if we can populate the ontology so as to
  - satisfy all the axioms
  - have at least one object in C

i.e. there is an interpretation I such that

1. $I \models$ TBox
2. $I \not\models C \sqsubseteq \bot$

Example : TBox vs. Concept Consistency

TBox $\boldsymbol{T}$ =

$W \sqsubseteq \{w\}$

$W \sqsubseteq \exists r.\top$

$W1 \sqsubseteq W \sqcap (\forall r.X1)$

$W2 \sqsubseteq W \sqcap (\forall r.X2)$

$X1\ disjoint\ X2$

$\boldsymbol{T}$ is consistent but in every model I of **T**,
if $\mathrm{I}(W1)$ is non-empty then $\mathrm{I}(W2)$ is empty, and vice versa.

$x \in \mathrm{I}(W1)$ and $x' \in \mathrm{I}(W2) \Rightarrow x = I(w) = x'$

$x = x'$ cannot be in $\mathrm{I}(\forall r.X1)$ and in $\mathrm{I}(\forall r.X2)$

Reasoning tasks: subsumption

Given a TBox $\boldsymbol{T}$, C subsumes D if

for every model I of $\boldsymbol{T}$, $I(D) \subseteq I(C)$

or equivalently

$\boldsymbol{T} \cup \{D \sqcap \neg C\}$ is inconsistent

Reasoning task:

input: a Tbox T, two classes C, D

output: true iff C subsumes D for T

Reasoning tasks: Instance checking

1. check if $C(o)$ is a consequence of the axioms and asserted facts

   amounts to check if C subsumes the concept $\{o\}$

2. find all the individuals that belong to C

   similar to query answering in (deductive) databases

Example

Find facts about individuals belonging to classes.

1. Parent $\equiv$ $\exists$ hasChild . Person
2. hasChild(Bob, Alice)
3. Woman(Alice)
4. Woman $\sqsubseteq$ Person

consequence

Parent(Bob)

Open World Semantics

What is not explicitly asserted is unknown (maybe true maybe false). Leads to counter intuitive results:

1. GoParent $\equiv \forall$ hasChild . Girl
2. hasChild(Bob, Alice)
3. Girl(Alice)

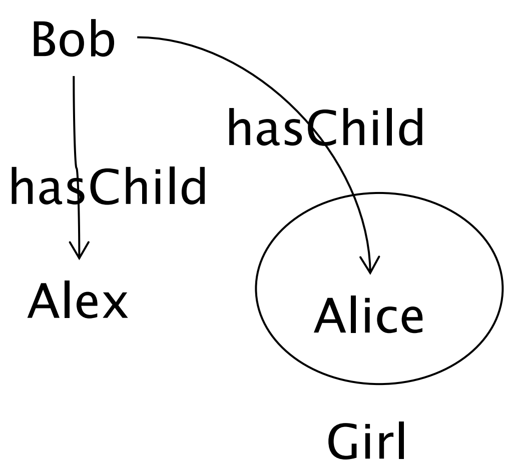
can we infer GoParent(Bob) ?

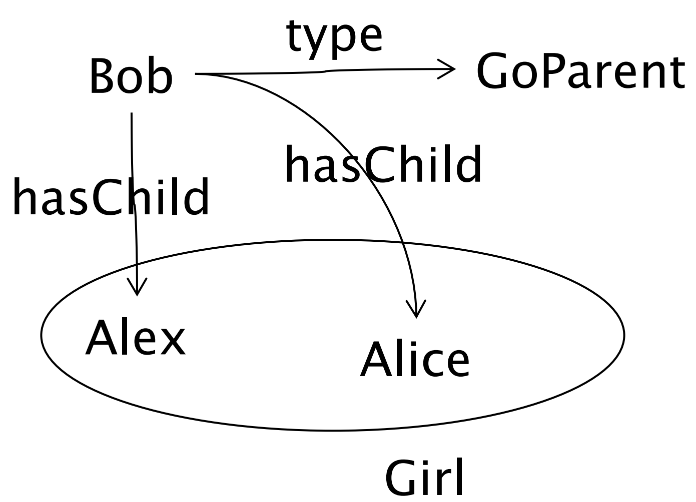
No, (Bob may have other children who are not girls)

Open World Semantics

Some models of

1. GoParent ≡ ∀ hasChild . Girl
2. hasChild(Bob, Alice)
3. Girl(Alice)





closing the world

1. GoParent $\equiv$ $\forall$ hasChild . Girl
2. hasChild(Bob, Alice)
3. Girl(Alice)
4. ParentOf1 $\sqsubseteq$ hasChild $=_1$ Thing
5. ParentOf1(Bob)

now we can infer Bob a GoParent

No Unique Name Assumption (UNA)

1. BusyParent ≡ hasChild $\geq_2$ Person
2. hasChild (Cindy, Bob)
3. hasChild (Cindy, John)

consequence: BusyParent (Cindy) ?

no, because *Bob* and *John* may be the same person

yes if we add the axiom

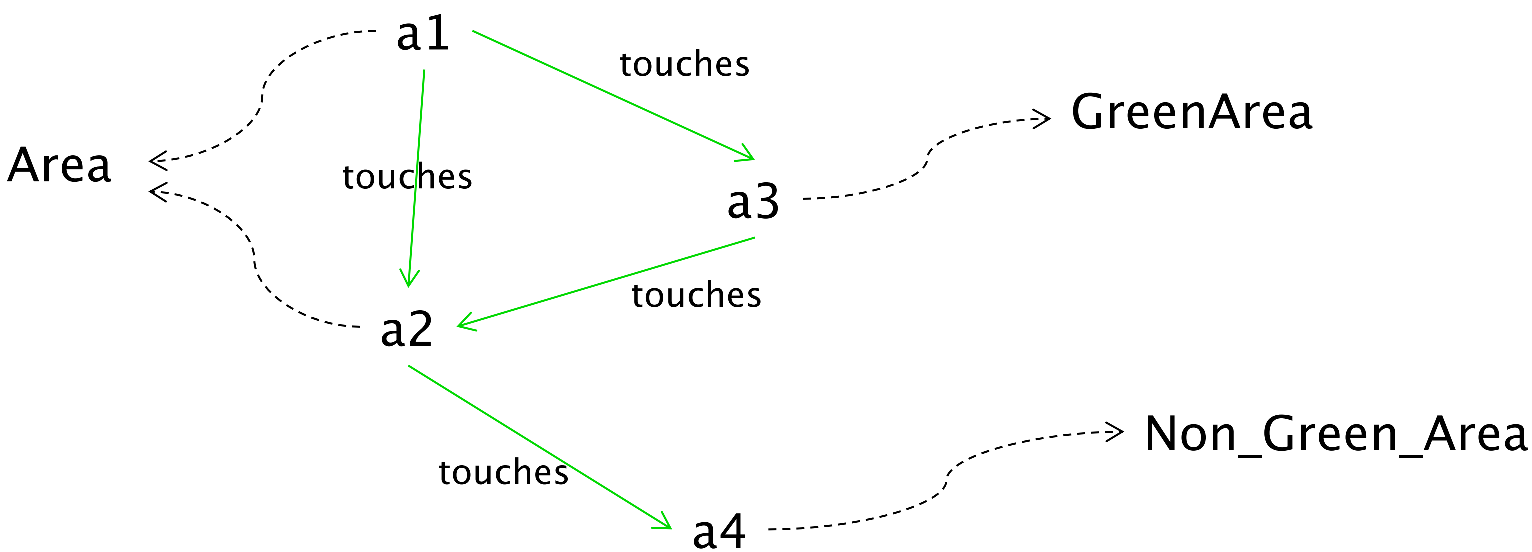
Bob ≠ John

Sophisticated “open world” reasoning

Terminological Axioms (TBox)

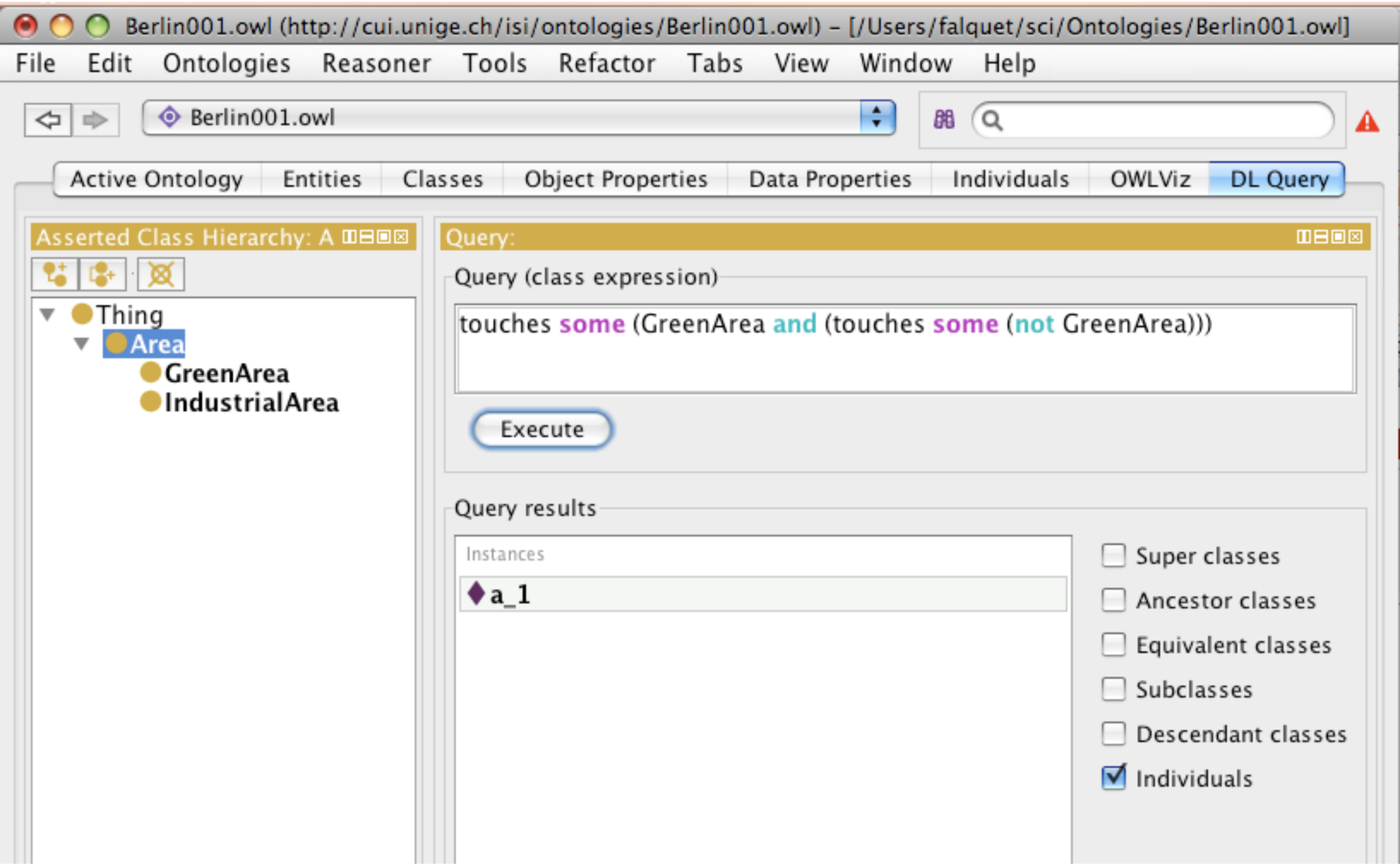
1. $\text{Green\_Area} \sqsubseteq \text{Area}$
2. $\text{Non\_Green\_Area} \equiv \text{Area} \sqcap (\neg\ \text{Green\_Area})$

ABox



Q: Does a1 touch some Green Area that touches some non Green Area?

A: Yes
- a2 is either green or non green (axioms 1 and 2)
- if it is green a1 satisfies the condition (using a3, a2)
- if it is non green a1 satisfies the condition (using a2, a4)


Berlin001.owl (http://cui.unige.ch/isi/ontologies/Berlin001.owl) – [/Users/falquet/sci/Ontologies/Berlin001.owl]
File Edit Ontologies Reasoner Tools Refactor Tabs View Window Help
Berlin001.owl
Active Ontology
Entities
Classes
Object Properties
Data Properties
Individuals
OWLViz
DL Query
Asserted Class Hierarchy: A
Thing
Area
GreenArea
IndustrialArea
Query:
Query (class expression)
touches some (GreenArea and (touches some (not GreenArea)))
Execute
Query results
Instances
a_1
Super classes
Ancestor classes
Equivalent classes
Subclasses
Descendant classes
Individuals

Reasoning Services for DL Ontologies

- In most description logics consistency and subsumption can be computed (with sophisticated tableau algorithms), with different time and space complexities
- Consequences
  - the consistency of an ontology can be checked
  - it is possible to compute the class subsumption hierarchy
  - it is possible to find the closest concept corresponding to a query
- There are description logics for which consistency and subsumption can be computed in polynomical time or better
  - OWL-RL, OWL-QL

Everything about DL

- at http://dl.kr.org/
- and http://www.cs.man.ac.uk/~ezolin/dl/

Complexity of reasoning in Description Logics

Note: the information here is (always) incomplete and updated often

Base description logic: *Attributive Language with Complements*

$\mathcal{ALC}$ ::= ⊥ | ⊤ | A | ¬C | $C \sqcap D$ | $C \sqcup D$ | ∃$R.C$ | ∀$R.C$

Concept constructors:

- [ ] $\mathcal{F}$ – functionality[2]: $(\leq 1\ R)$
- [x] $\mathcal{N}$ – (unqualified) number restrictions: $(\geq n\ R)$, $(\leq n\ R)$
- [x] $\mathcal{Q}$ – qualified number restrictions: $(\geq n\ R.C)$, $(\leq n\ R.C)$
- [x] $\mathcal{O}$ – nominals: $\{a\}$ or $\{a_1, \ldots, a_n\}$ ("one-of")
- [ ] μ – least fixpoint operator: $\mu X.C$

Forbid complex roles[5] in number restrictions[6]

Role constructors: trans reg

- [x] $\mathcal{I}$ – role inverse: R^-
- [ ] ∩ – role intersection[3]: $R \cap S$
- [ ] ∪ – role union: $R \cup S$
- [ ] ¬ – role complement: ¬R full
- [ ] o – role chain (composition): $R \circ S$
- [ ] * – reflexive-transitive closure[4]: R^*
- [ ] id – concept identity: $id(C)$

TBox (concept axioms) is *internalizable* in extensions of $\mathcal{ALCIO}$, see [82, Lemma 4.12], [61, p.3]

- (•) empty TBox
- ( ) acyclic TBox ($A \equiv C$, A is a concept name; no cycles)
- ( ) general TBox ($C \sqsubseteq D$, for arbitrary concepts C and D)

RBox (role axioms): OWL-Lite OWL-DL OWL 1.1

- [x] $\mathcal{S}$ – role transitivity: Tr(R)
- [x] $\mathcal{H}$ – role hierarchy: $R \sqsubseteq S$
- [ ] $\mathcal{R}$ – complex role inclusions: $R \circ S \sqsubseteq R$, $R \circ S \sqsubseteq S$
- [ ] s – some additional features (click to see them)

Reset — You have selected a Description Logic: $\mathcal{SHOIQ}$

Complexity[7] of reasoning problems[8]

Concept satisfiability	**NExpTime-complete**	• Hardness of even $\mathcal{ALCFIO}$ is proved in [82, Corollary 4.13]. • A different proof of the NExpTime-hardness for $\mathcal{ALCFIO}$ is given in [61] (even with 1 nominal, and inverse roles not used in number restrictions). • Upper bound for $\mathcal{SHOIQ}$ is proved in [12, Corollary 6.31] with numbers coded in unary (for binary coding, the upper bound remains an open problem for all logics in between $\mathcal{ALCNIO}$ and $\mathcal{SHOIQ}$). • A tableaux algorithm for $\mathcal{SHOIQ}$ is presented in [51]. • **Important:** in number restrictions, only *simple* roles (i.e. which are neither transitive nor have a transitive subroles) are allowed; otherwise we gain undecidability even in $\mathcal{SHN}$; see [54]. • **Remark:** recently [55] it was observed that, in many cases, one can use transitive roles in number restrictions –