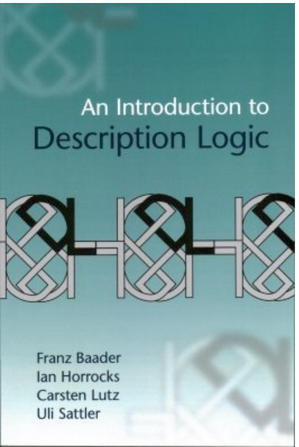
THE UNIVERSITY of EDINBURGH

INFR11215 Knowledge Graphs

Tractable Schema Reasoning

Jeff Z. Pan

http://knowledge-representation.org/j.z.pan/

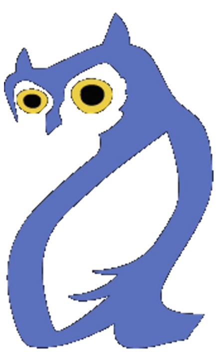


[Reading: Baader et al., Chapter 6]

Lecture Outline

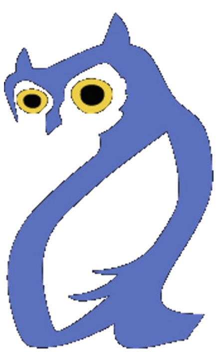
- Motivation
- Overview of EL and reasoning
- Detailed Discussions on EL and reasoning
- Practical

Motivations



- Web Ontology Language (OWL)
  - OWL v2 family
    - OWL 2 DL
    - OWL 2 EL, OWL 2 QL, OWL 2 RL
- ALC not a good starting point
  - its foundation FL_0 ($\sqcap$ and $\forall$) is not a good foundation
  - subsumption with GCI is EXPTime-complete
  - EL is PTime-complete
    - TBox reasoning
    - ABox reasoning
    - Query answering

Motivations

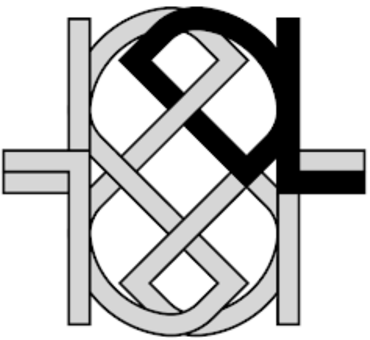


- SNOMED Clinical Terms is
  - Probably the single most comprehensive clinical terminology
  - Licensed for national use throughout the UK and the US
  - Content that covers most clinical concepts
  - A terminology model that supports retrieval of alternative representations of similar information
- SNOMED CT is an EL ontology
  - clear performance difference between EL algorithm and algorithms for ALC-extended logics

Lecture Outline

- Motivation
- Overview of EL and reasoning
- Detailed Discussions on EL and reasoning
- Practical

What is EL



- EL Class Description
  - existential restriction: $\exists r.C$
  - conjunction: $C \sqcap D$
  - the top class: $\top$
  - not including:
    - value restriction: $\forall r.C$
    - disjunction: $C \sqcup D$
    - the bottom class: $\bot$
- EL Axioms
  - GCI: $C \sqsubseteq D$

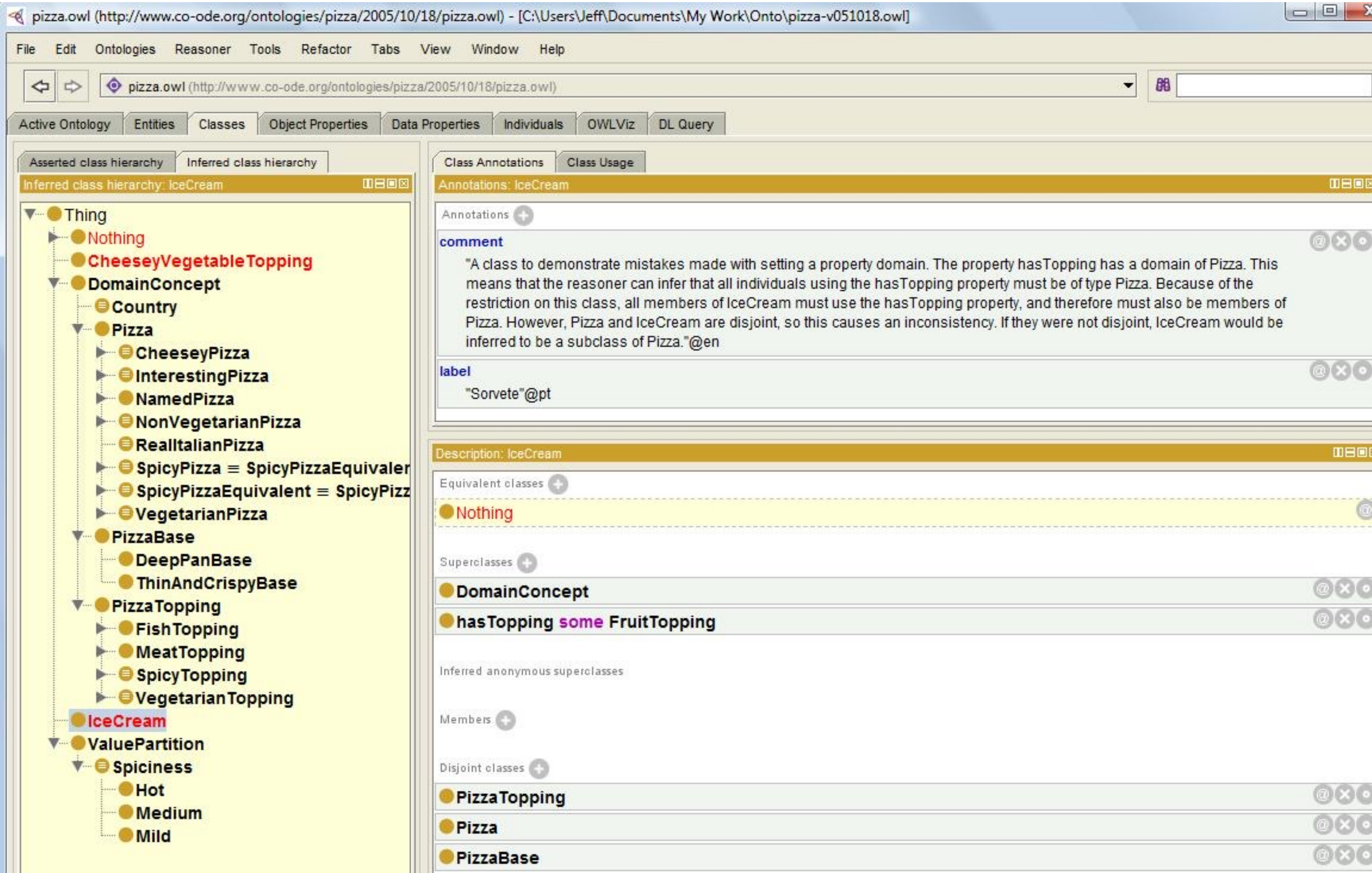
Class Satisfiability Checking in EL

- Every EL class is satisfiable
  - Why?
  - Class satisfiability checking is not an interesting problem
- Challenge
  - Subsumption checking in EL is non-trivial, as it cannot be reduced to class unsatisfiability
  - Why?
  - $O \models C \sqsubseteq D$ iff $C \sqcap \neg D$ is unsatisfiable

Subsumption Checking in EL

- Subsumption checking in EL (with general Tbox) is PTime-complete
  - For FL_0, it is EXPTime-complete
- Usually this is done in a batch mode: classification
  - A TBox reasoning service that computes subsumption relation among all named classes
- Given an EL TBox T, signature Sig (T) contains all class and property names used in T

Classification



Normalisation

- Idea
  - Simplify the axioms into some certain form so that reasoning algorithms can take advantage of it
  - example: NNF (negated normal form)
- Normal forms for EL
  - $A \sqsubseteq B$
  - $A1 \sqcap A2 \sqsubseteq B$
  - $A \sqsubseteq \exists r.B$
  - $\exists r.A \sqsubseteq B$
  - where A, A1, A2, B are either named class in Sig(T) or the top class $\top$

Normalisation Rules

NF0	$\widehat{D} \sqsubseteq \widehat{E}$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ A \sqsubseteq \widehat{E}$
NF1$_r$	$C \sqcap \widehat{D} \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ C \sqcap A \sqsubseteq B$
NF1$_\ell$	$\widehat{D} \sqcap C \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ A \sqcap C \sqsubseteq B$
NF2	$\exists r.\widehat{D} \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ \exists r.A \sqsubseteq B$
NF3	$B \sqsubseteq \exists r.\widehat{D}$	$\longrightarrow$	$A \sqsubseteq \widehat{D},\ B \sqsubseteq \exists r.A$
NF4	$B \sqsubseteq D \sqcap E$	$\longrightarrow$	$B \sqsubseteq D,\ B \sqsubseteq E$

where C, D, E denote arbitrary $\mathcal{EL}$ concepts,
$\widehat{D}, \widehat{E}$ denote $\mathcal{EL}$ concepts that are neither concept names nor $\top$,
B is a concept name, and
A is a new concept name.

[credit: F Baader]

Example: Normalisation

NF0	$\widehat{D} \sqsubseteq \widehat{E}$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ A \sqsubseteq \widehat{E}$
$\mathsf{NF1}_r$	$C \sqcap \widehat{D} \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ C \sqcap A \sqsubseteq B$
$\mathsf{NF1}_\ell$	$\widehat{D} \sqcap C \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ A \sqcap C \sqsubseteq B$
NF2	$\exists r.\widehat{D} \sqsubseteq B$	$\longrightarrow$	$\widehat{D} \sqsubseteq A,\ \exists r.A \sqsubseteq B$
NF3	$B \sqsubseteq \exists r.\widehat{D}$	$\longrightarrow$	$A \sqsubseteq \widehat{D},\ B \sqsubseteq \exists r.A$
NF4	$B \sqsubseteq D \sqcap E$	$\longrightarrow$	$B \sqsubseteq D,\ B \sqsubseteq E$

- Input axiom
  - $\exists$r.A $\sqcap$ $\exists$r.$\exists$s.A $\sqsubseteq$ A $\sqcap$ B
- Normalisation
  1. $\exists$r.A $\sqcap$ $\exists$r.$\exists$s.A $\sqsubseteq$A0, A0 $\sqsubseteq$ A $\sqcap$ B (NF0)
  2. $\exists$r.A $\sqsubseteq$ A1, A1 $\sqcap$ $\exists$r.$\exists$s.A $\sqsubseteq$ A0 (NF1l)
  3. $\exists$r.$\exists$s.A $\sqsubseteq$ A2, A1 $\sqcap$ A2 $\sqsubseteq$ A0 (NF1r)
  4. $\exists$s.A $\sqsubseteq$ A3, $\exists$r.A3 $\sqsubseteq$ A2 (NF2)
  5. A0 $\sqsubseteq$ A, A0 $\sqsubseteq$ B (NF4)

[credit: F Baader]

Conservative Extension

- Given two EL TBoxes T1 and T2, T2 is a conservative extension of T1 if
  - Sig(T1) ⊆Sig (T2)
  - every model of T2 is a model of T1
  - for every model I1 of T1, there exists a model I2 of T2 such as I1 and I2 coincide on sig(T1) U T, i.e.,
    - $\Delta^{I1} = \Delta^{I2}$
    - $A^{I1} = A^{I2}$ for every named class in A ∈Sig(T1), and
    - $r^{I1} = r^{I2}$ for every named property in r ∈ Sig(T1)

Conservative Extension and EL

- Given two EL TBoxes T1 and T2, such that T2 is a conservative extension of T1, and C, D are EL class descriptions containing only class and property names from Sig(T1)
  - Then T1 $\models C \sqsubseteq D$ iff T2 $\models C \sqsubseteq D$

- Given two EL TBoxes T1 and T2, such that T2 is the normalised TBox obtained from T1
  - Then T2 is a conservative extension of T1
  - T2 is linear in the size of T1

Classification Procedure

- We assume that the input TBox axioms are all in normal form
  - The overall number of the normalised GCIs is polynomial in the size of the TBox
- Idea
  - start from the inputs GCIs and add implied GCIs using classification rules

Classification Rules

- To get the concrete we need to
  - replace meta-variables A, A1, A2, A3, B, B1 by concrete named classes and replace meta-variable r by a concrete named property
- Rule application
  - T' start as the TBox
  - If axioms appear on top of the line are in T', then add the axioms below into T' (unless they are already in)

$$\text{CR1}\ \frac{}{A \sqsubseteq A} \qquad \text{CR2}\ \frac{}{A \sqsubseteq \top}$$

$$\text{CR3}\ \frac{A_1 \sqsubseteq A_2 \quad A_2 \sqsubseteq A_3}{A_1 \sqsubseteq A_3} \qquad \text{CR4}\ \frac{A \sqsubseteq A_1 \quad A \sqsubseteq A_2 \quad A_1 \sqcap A_2 \sqsubseteq B}{A \sqsubseteq B}$$

$$\text{CR5}\ \frac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$$

[credit: F Baader]

Example: Classification Rules

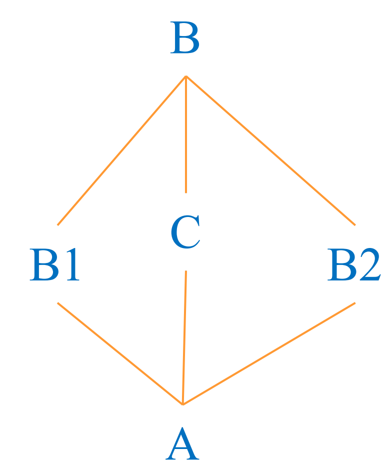
CR1 $\dfrac{}{A \sqsubseteq A}$ CR2 $\dfrac{}{A \sqsubseteq \top}$

CR3 $\dfrac{A_1 \sqsubseteq A_2 \quad A_2 \sqsubseteq A_3}{A_1 \sqsubseteq A_3}$ CR4 $\dfrac{A \sqsubseteq A_1 \quad A \sqsubseteq A_2 \quad A_1 \sqcap A_2 \sqsubseteq B}{A \sqsubseteq B}$

CR5 $\dfrac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$

$$\mathcal{T}_1 = \{A \sqsubseteq \exists r.A,\ \exists r.B \sqsubseteq B_1,\ \top \sqsubseteq B,\ A \sqsubseteq B_2,\ B_1 \sqcap B_2 \sqsubseteq C\}$$

1. A ⊑A, B ⊑B, B1 ⊑B1, B2 ⊑B2, C ⊑ C (CR1)
2. A ⊑ ⊤,B1 ⊑ ⊤, B2 ⊑ ⊤, C ⊑ ⊤, B ⊑ ⊤ (CR2)
3. A ⊑ ⊤, ⊤ ⊑ B => A ⊑ B (CR3)
4. B1 ⊑ ⊤, ⊤ ⊑ B => B1 ⊑ B (CR3)
5. B2 ⊑ ⊤, ⊤ ⊑ B => B2 ⊑ B (CR3)
6. C ⊑ ⊤, ⊤ ⊑ B => C ⊑ B (CR3)
7. A ⊑ ∃r.A, A ⊑ B, ∃r.B ⊑ B1 => A ⊑ B1 (CR5)
8. A ⊑ B1, A ⊑ B2, B1 ⊓ B2 ⊑ C => A ⊑ C (CR4)



Lecture Outline

- Motivation
- Overview of EL and reasoning
- Detailed Discussions on EL and reasoning
- Practical

Subsumption Checking

- Subsumption checking between two class descirptions $C \sqsubseteq D$ can be reduced to that between two named classes $A1 \sqsubseteq A2$
- More precisely
  - An EL TBox $T \models C \sqsubseteq D$ iff $T \cup \{A1 \sqsubseteq C, D \sqsubseteq A2\} \models A1 \sqsubseteq A2$

Example: Subsumption Checking

CR1 $\dfrac{}{A \sqsubseteq A}$ CR2 $\dfrac{}{A \sqsubseteq \top}$

CR3 $\dfrac{A_1 \sqsubseteq A_2 \quad A_2 \sqsubseteq A_3}{A_1 \sqsubseteq A_3}$ CR4 $\dfrac{A \sqsubseteq A_1 \quad A \sqsubseteq A_2 \quad A_1 \sqcap A_2 \sqsubseteq B}{A \sqsubseteq B}$

CR5 $\dfrac{A \sqsubseteq \exists r.A_1 \quad A_1 \sqsubseteq B_1 \quad \exists r.B_1 \sqsubseteq B}{A \sqsubseteq B}$

$$\mathcal{T}_1 = \{A \sqsubseteq \exists r.A,\ \exists r.B \sqsubseteq B_1,\ \top \sqsubseteq B,\ A \sqsubseteq B_2,\ B_1 \sqcap B_2 \sqsubseteq C\}$$

Question: Check if A ⊓ C ⊑ ∃r.B holds

1. Extend the KB with {A’ ⊑ A ⊓ C, ∃r.B ⊑ B’}, which is normalised as {A’ ⊑ A, A’ ⊑ C, ∃r.B ⊑ B’} (NF4)
2. A ⊑A, B ⊑B, A’ ⊑A’, B’ ⊑B’, B1 ⊑B1, B2 ⊑B2, C ⊑ C (CR1)
3. A ⊑ ⊤, A’ ⊑ ⊤ B1 ⊑ ⊤, B2 ⊑ ⊤, C ⊑ ⊤, B ⊑ ⊤, B’ ⊑ ⊤ (CR2)
4. A ⊑ ⊤, ⊤ ⊑ B => A ⊑ B (CR3)
5. B1 ⊑ ⊤, ⊤ ⊑ B => B1 ⊑ B (CR3)
6. B2 ⊑ ⊤, ⊤ ⊑ B => B2 ⊑ B (CR3)
7. C ⊑ ⊤, ⊤ ⊑ B => C ⊑ B (CR3)
8. A ⊑ ∃r.A, A ⊑ B, ∃r.B ⊑ B1 => A ⊑ B1 (CR5)
9. A ⊑ ∃r.A, A ⊑ B, ∃r.B ⊑ B’ => A ⊑ B’ (CR5)
10. A’ ⊑A, A ⊑B’ =>A’ ⊑B’ (CR3)
11. Since A’ ⊑B’ holds, we have A ⊓ C ⊑ ∃r.B

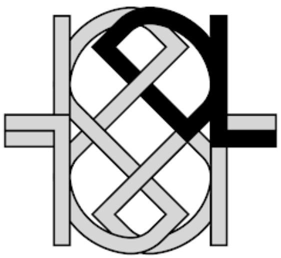
EL Family

- EL+ extends EL with
  - property chain inclusion: r1 o …o rk ⊑ r
  - concrete domain (n-ary dataype predicate): D(f1,…fn)
- EL++ extends EL+ with
  - the bottom class: ⊥
  - norminal: {a}

Classification: OWL 2 EL vs OWL 2 DL

- OWL 2 DL
  - subsumption checking is N2EXPTime-Complete
  - GCI-rule is expensive
  - many new optimisations but still challenging when there are large number of classes (SNOMED CT has over 300K)
- OWL 2 EL
  - Batch mode
  - Good base for approximation (such as those used by the TrOWL reasoner)

Conjunctive Queries

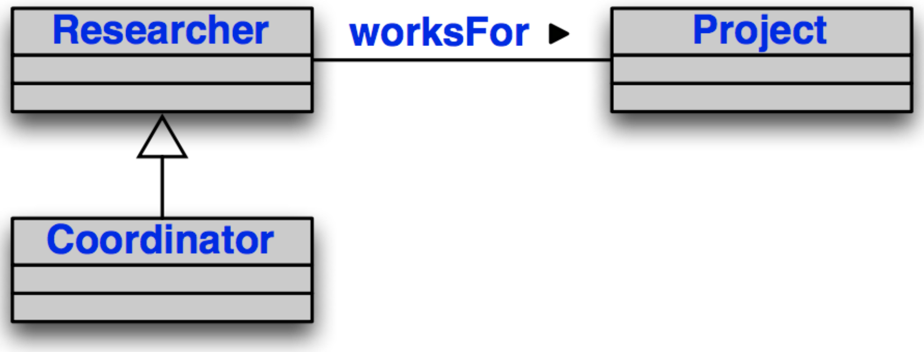


- A conjunctive query q($\vec{x}$) has the form
  - $\exists y1,\ldots,ym.(\boldsymbol{\alpha}1 \wedge \ldots \wedge \boldsymbol{\alpha}n)$, where m>=0, n>=1
  - each atom **α**i is a concept atom A(x) or a property atom r(x,y)
  - y1,…,ym are called quantified variables
  - quantified variables that appear only in one atom are called unbounded variables
- CQs without constants are called pure CQs
- CQs can be reduced to pure CQs in polynomial time
- An FO query is called a Boolean query if its arity is 0.

Example: Conjunctive Queries

- Assuming that we have three tables Professor, supervises and Student
- Return all pairs of supervisors and students
  - $q1(x1,x2) = Professor(x1) \wedge supervise(x1,x2) \wedge Student(x2)$
  - also written as $q1(x1,x2) \leftarrow Professor(x1) \wedge supervise(x1,x2) \wedge Student(x2)$
- Return all students whom are supervised by some professors
  - $q2(x) = \exists y.Professor(y) \wedge supervises(y,x) \wedge Student(x)$

Ontology Based QA: Example 1



We assume that each concept/relationship of the ontology is mapped directly to a database table.

But the database tables may be **incompletely specified**, or even missing for some concepts/relationships.

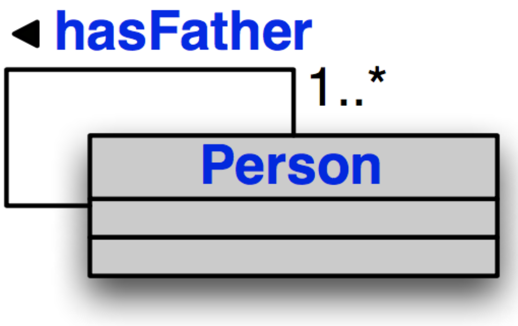
DB: Coordinator $\supseteq$ { serge, marie }
Project $\supseteq$ { webdam, diadem }
worksFor $\supseteq$ { (serge,webdam), (georg,diadem) }

Query: $q(x) \leftarrow \text{Researcher}(x)$

Answer: { serge, marie, georg }

[credit: G Xiao]

Ontology Based QA : Example 2



Each person has a father, who is a person.

DB: Person $\supseteq$ { john, nick, toni }
hasFather $\supseteq$ { (john,nick), (nick,toni) }

Queries: $q_1(x, y) \leftarrow \text{hasFather}(x, y)$

$q_2(x) \leftarrow \exists y. \text{hasFather}(x, y)$

$q_3(x) \leftarrow \exists y_1, y_2, y_3. \text{hasFather}(x, y_1) \wedge \text{hasFather}(y_1, y_2) \wedge \text{hasFather}(y_2, y_3)$

$q_4(x, y_3) \leftarrow \exists y_1, y_2. \text{hasFather}(x, y_1) \wedge \text{hasFather}(y_1, y_2) \wedge \text{hasFather}(y_2, y_3)$

Answers: to q_1: { (john,nick), (nick,toni) }

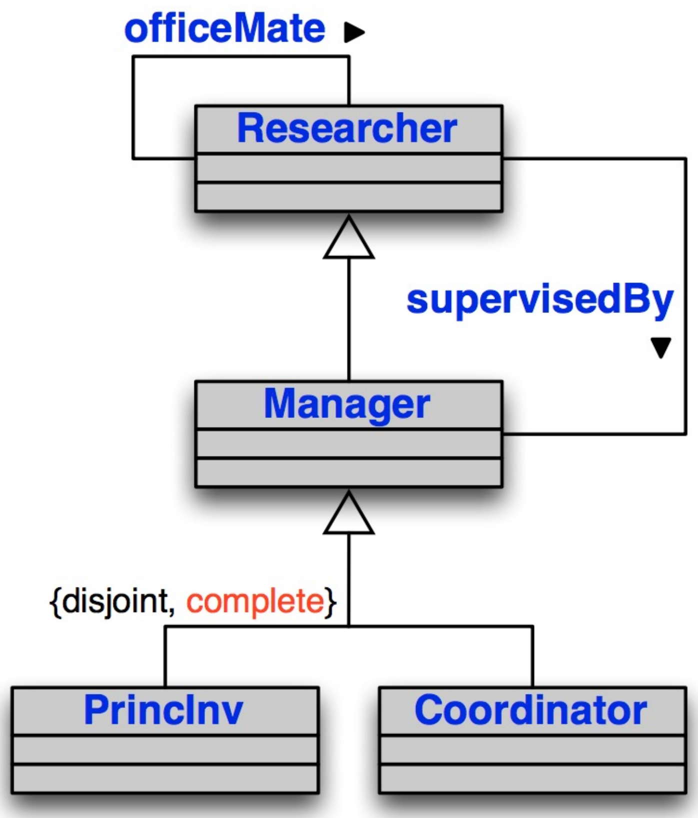
to q_2: { john, nick, toni }

to q_3: { john, nick, toni }

to q_4: { }

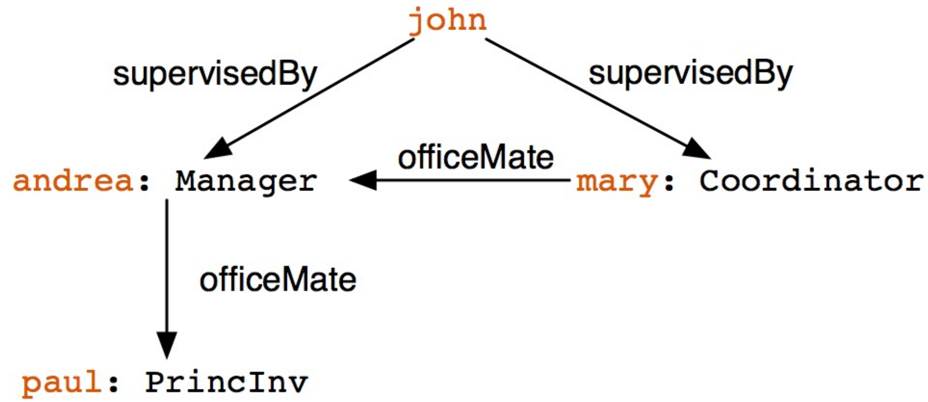
[credit: G Xiao]

Ontology Based QA : Example 3



$\text{Manager} \equiv \text{PrincInv} \sqcup \text{Coordinator}$

$\text{Researcher} \supseteq \{ \texttt{andrea}, \texttt{paul}, \texttt{mary}, \texttt{john} \}$
$\text{Manager} \supseteq \{ \texttt{andrea}, \texttt{paul}, \texttt{mary} \}$
$\text{PrincInv} \supseteq \{ \texttt{paul} \}$
$\text{Coordinator} \supseteq \{ \texttt{mary} \}$
$\text{supervisedBy} \supseteq \{ (\texttt{john},\texttt{andrea}), (\texttt{john},\texttt{mary}) \}$
$\text{officeMate} \supseteq \{ (\texttt{mary},\texttt{andrea}), (\texttt{andrea},\texttt{paul}) \}$



$q(x) \leftarrow \exists y, z.$
$\text{supervisedBy}(x, y),\ \text{Coordinator}(y),$
$\text{officeMate}(y, z),\ \text{PrincInv}(z)$

Answer: { john }

To obtain this answer, we need to **reason by cases**.

[credit: A Schaerf]

Lecture Outline

- Motivation：efficient and scalable reasoning
- Introduction: the EL description logic
- Focus: subsumption checking in EL
- Tutorial
  - Normailisation
  - Classification
  - Subsumption
-