Scalable Ontology-Based Information Systems

Ian Horrocks
<ian.horrocks@comlab.ox.ac.uk>
Information Systems Group
Oxford University Computing Laboratory

What is an Ontology?

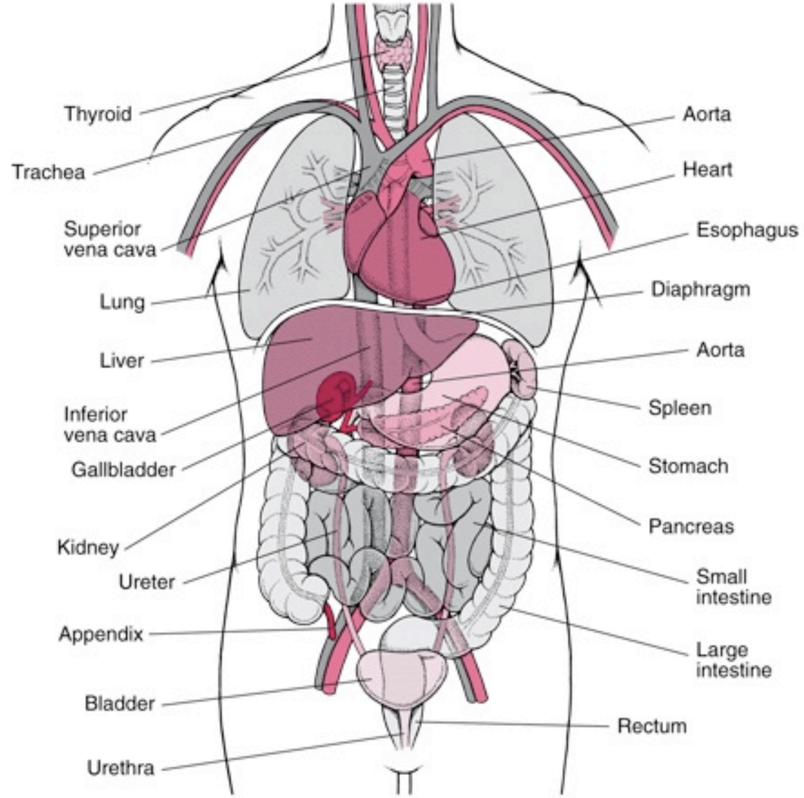
What is an Ontology?

A model of (some aspect of) the world

What is an Ontology?

A model of (some aspect of) the world

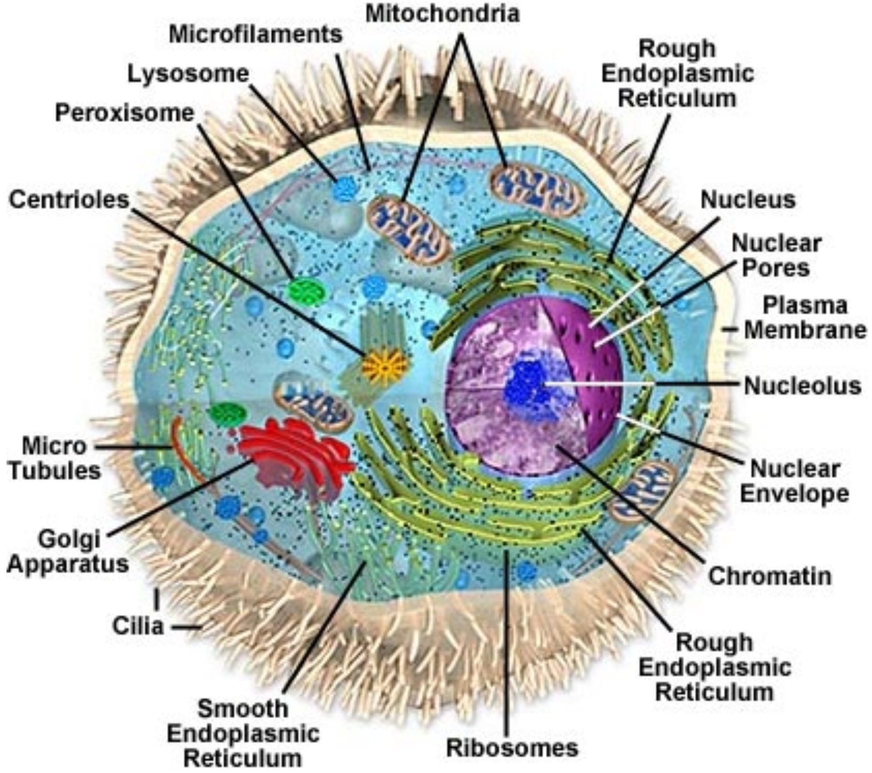
- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy



What is an Ontology?

A model of (some aspect of) the world

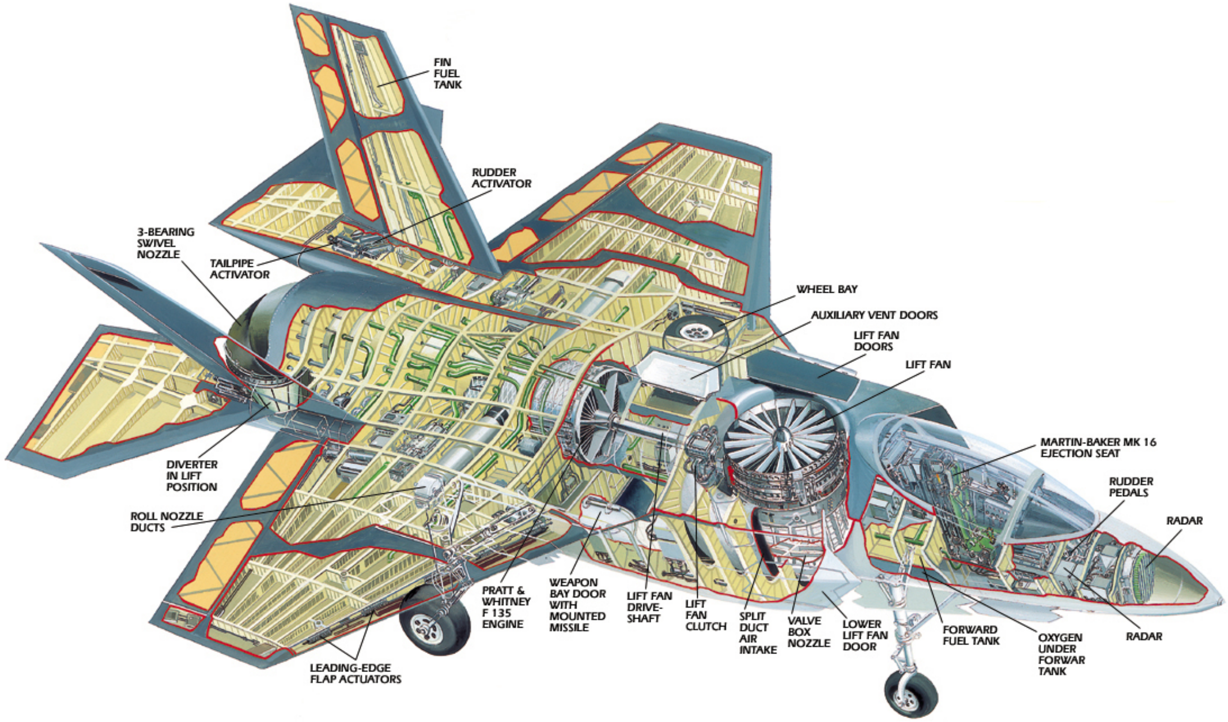
- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology



What is an Ontology?

A model of (some aspect of) the world

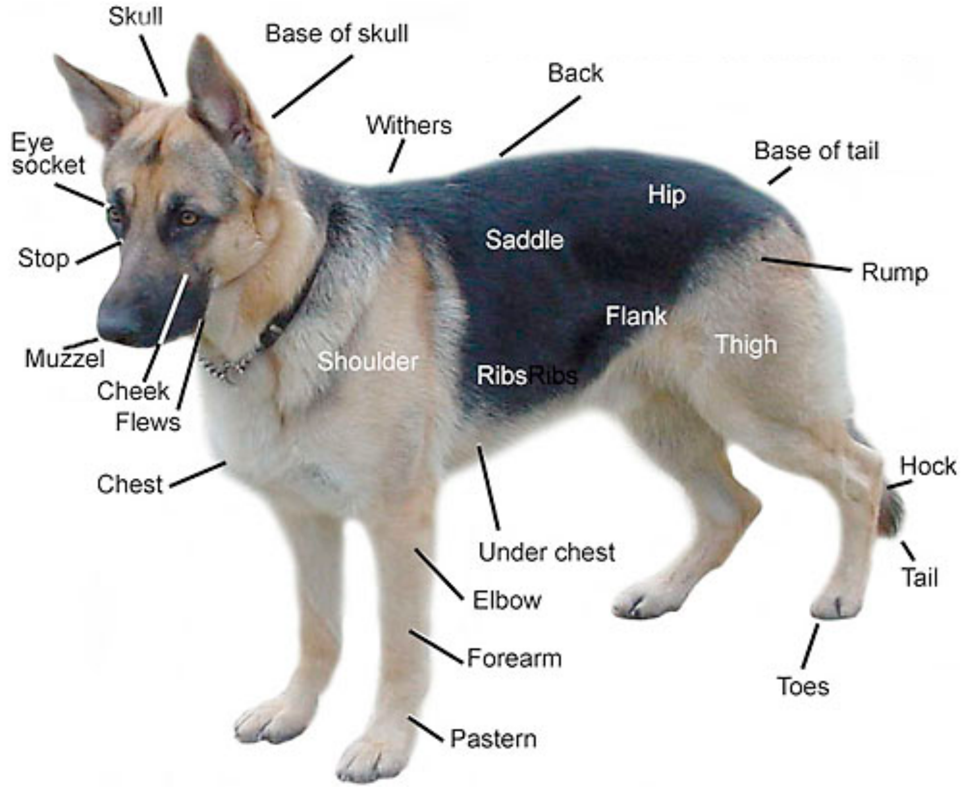
- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace



What is an Ontology?

A model of (some aspect of) the world

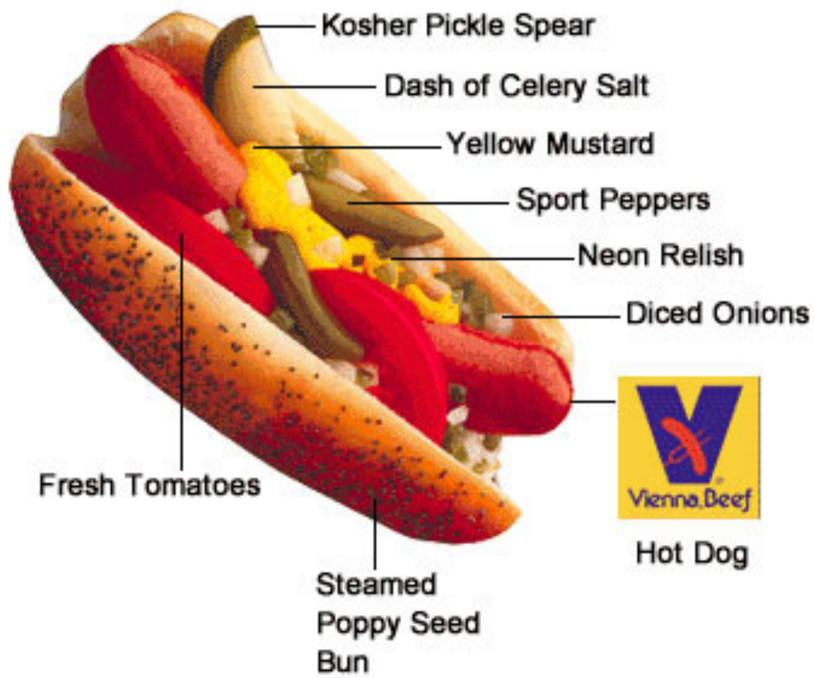
- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace
  - Dogs



What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain, e.g.:
  - Anatomy
  - Cellular biology
  - Aerospace
  - Dogs
  - Hotdogs
  - …

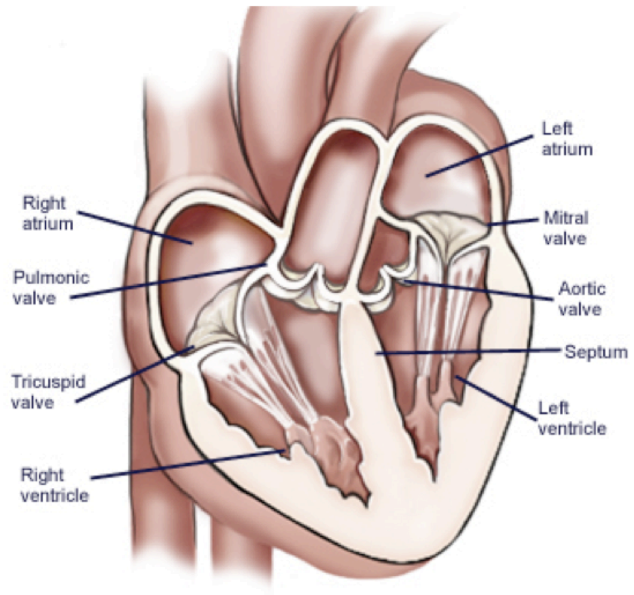


What is an Ontology?

A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain
- Specifies **meaning** (semantics) of terms

  Heart is a muscular organ that
  is part of the circulatory system



What is an Ontology?

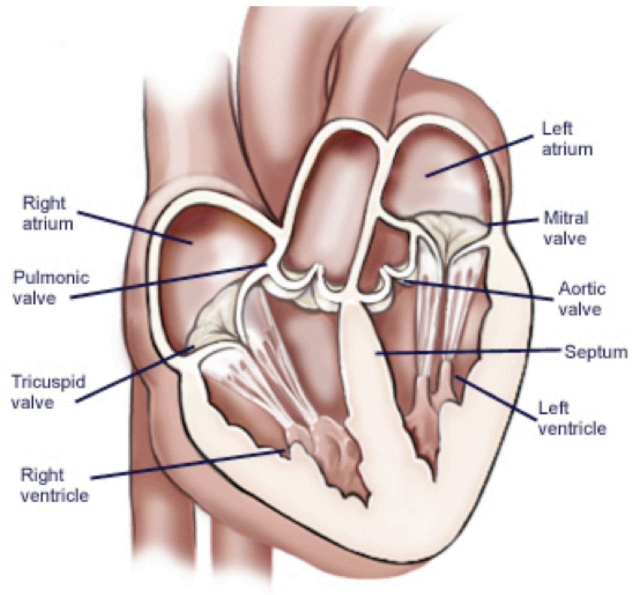
A model of (some aspect of) the world

- Introduces **vocabulary** relevant to domain
- Specifies **meaning** (semantics) of terms

  Heart is a muscular organ that
  is part of the circulatory system

- **Formalised** using suitable logic

$$\forall x.[\mathsf{Heart}(x) \rightarrow \mathsf{MuscularOrgan}(x) \wedge \exists y.[\mathsf{isPartOf}(x,y) \wedge \mathsf{CirculatorySystem}(y)]]$$



Web Ontology Language OWL (2)

- W3C **recommendation(s)**
- Motivated by **Semantic Web** activity

  Add meaning to web content by annotating it with terms defined in ontologies

- Supported by **tools and infrastructure**
  - APIs (e.g., OWL API, Thea, OWLink)
  - Development environments (e.g., Protégé, Swoop, TopBraid Composer, Neon)
  - Reasoners & Information Systems (e.g., Pellet, Racer, HermiT, Quonto, …)

- Based on **Description Logics** ($\mathcal{SHOIN}$ / $\mathcal{SROIQ}$)

Description Logics (DLs)

- Fragments of **first order logic** designed for KR
- Desirable computational properties
  - **Decidable** (essential)
  - Low complexity (desirable)
- Succinct and **variable free syntax**

$$\forall x.[\mathsf{Heart}(x) \rightarrow \mathsf{MuscularOrgan}(x) \wedge \exists y.[\mathsf{isPartOf}(x, y) \wedge \mathsf{CirculatorySystem}(y)]]$$

$$\mathsf{Heart} \sqsubseteq \mathsf{MuscularOrgan} \sqcap \exists \mathsf{isPartOf}.\mathsf{CirculatorySystem}$$

Description Logics (DLs)

DL **Knowledge Base** (KB) consists of two parts:

- Ontology (aka **TBox**) axioms define terminology (schema)

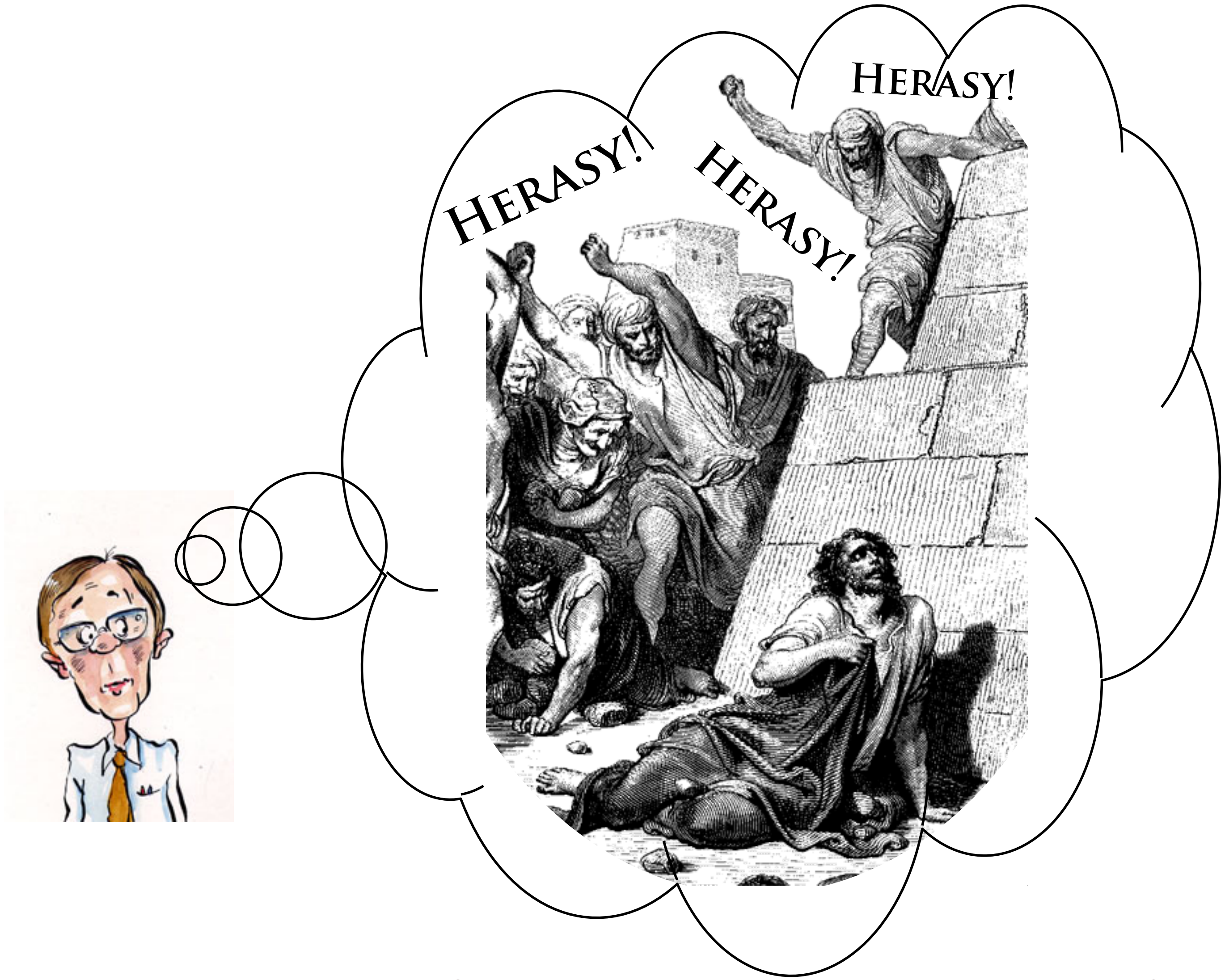
$$\begin{aligned}\mathsf{Heart} &\sqsubseteq \mathsf{MuscularOrgan} \sqcap \\ &\quad \exists \mathsf{isPartOf}.\mathsf{CirculatorySystem} \\ \mathsf{HeartDisease} &\equiv \mathsf{Disease} \sqcap \\ &\quad \exists \mathsf{affects}.\mathsf{Heart} \\ \mathsf{VascularDisease} &\equiv \mathsf{Disease} \sqcap \\ &\quad \exists \mathsf{affects}.(\exists \mathsf{isPartOf}.\mathsf{CirculatorySystem})\end{aligned}$$

- Ground facts (aka **ABox**) use the terminology (data)

$$\begin{aligned}\mathsf{John} : \mathsf{Patient} &\sqcap \\ &\exists \mathsf{suffersFrom}.\mathsf{HeartDisease}\end{aligned}$$

Why Care About Semantics?

Why Care About Semantics?

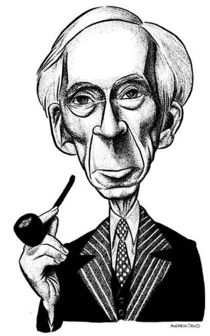


Why Care About Semantics?

Why should I care about semantics?

Why Care About Semantics?

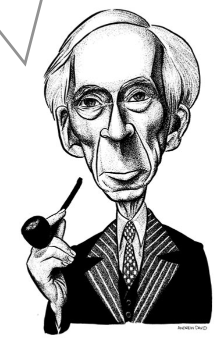
Why should I care about semantics?



Why Care About Semantics?

Why should I care about semantics?

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.

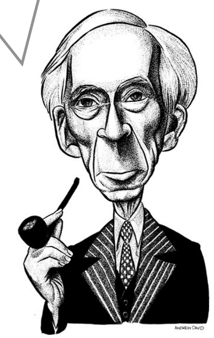


Why Care About Semantics?

Why should I care about semantics?

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.

That’s OK, but I don’t get paid for philosophy.



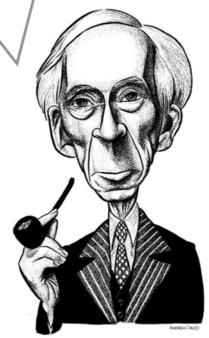
Why Care About Semantics?

Why should I care about semantics?

Well, from a philosophical POV, we need to specify the relationship between statements in the logic and the existential phenomena they describe.

That's OK, but I don't get paid for philosophy.

From a practical POV, in order to specify and test (ontology-based) information systems we need to precisely define their intended behaviour



What are Ontologies Good For?

- Coherent **user-centric view** of domain
  - Help identify and resolve disagreements
- Ontology-based **Information Systems**
  - View of data that is independent of logical/ physical schema
  - Answers reflect schema & data, e.g.:

    "Patients suffering from Vascular Disease"

Now... *that* should clear up a few things around here

What are Ontologies Good For?

$$\begin{aligned}
\mathsf{Heart} &\sqsubseteq \mathsf{MuscularOrgan} \sqcap \\
&\quad \exists \mathsf{isPartOf}.\mathsf{CirculatorySystem} \\
\mathsf{HeartDisease} &\equiv \mathsf{Disease} \sqcap \\
&\quad \exists \mathsf{affects}.\mathsf{Heart} \\
\mathsf{VascularDisease} &\equiv \mathsf{Disease} \sqcap \\
&\quad \exists \mathsf{affects}.(\exists \mathsf{isPartOf}.\mathsf{CirculatorySystem})
\end{aligned}$$

$$\begin{aligned}
\mathsf{John} : \mathsf{Patient} &\sqcap \\
&\exists \mathsf{suffersFrom}.\mathsf{HeartDisease}
\end{aligned}$$

What are Ontologies Good For?

- Coherent **user-centric view** of domain
  - Help identify and resolve disagreements
- Ontology-based **Information Systems**
  - View of data that is independent of logical/ physical schema
  - Answers reflect schema & data, e.g.:

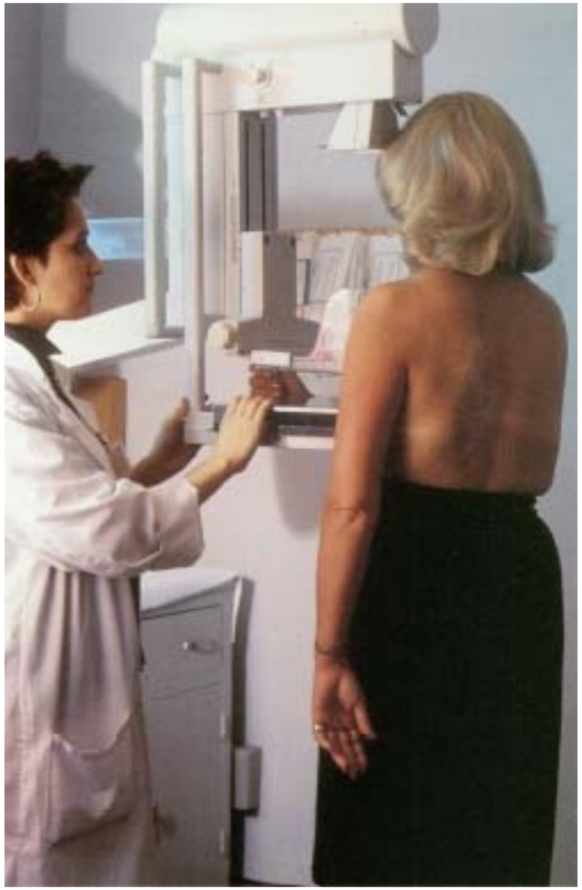
    "Patients suffering from Vascular Disease"
  - Query expansion/navigation/refinement
  - Incomplete and semi-structured data
  - Integration of heterogeneous sources

Now... *that* should clear up a few things around here

Information-Based Decisions

Increasingly critical in many areas:

- In Healthcare industry
  - Too much screening harms patients and wastes money
  - Too little screening costs lives



Information-Based Decisions

Increasingly critical in many areas:

- In Oil and Gas industry
  - Better quality information could add €1B/year net value to Statoil production
  - Poorer quality information and analysis costs €6M/weekend!

Information-Based Decisions

Increasingly critical in many areas:

- In IT industry
  - SAP deals with 80,000 queries/month at a cost of approx. €16M
  - SAP estimate 50% of support staff time spent searching for relevant information

Information-Based Decisions

Increasingly critical in many areas:

- In Transport Security
  - Failures can cost hundreds of lives

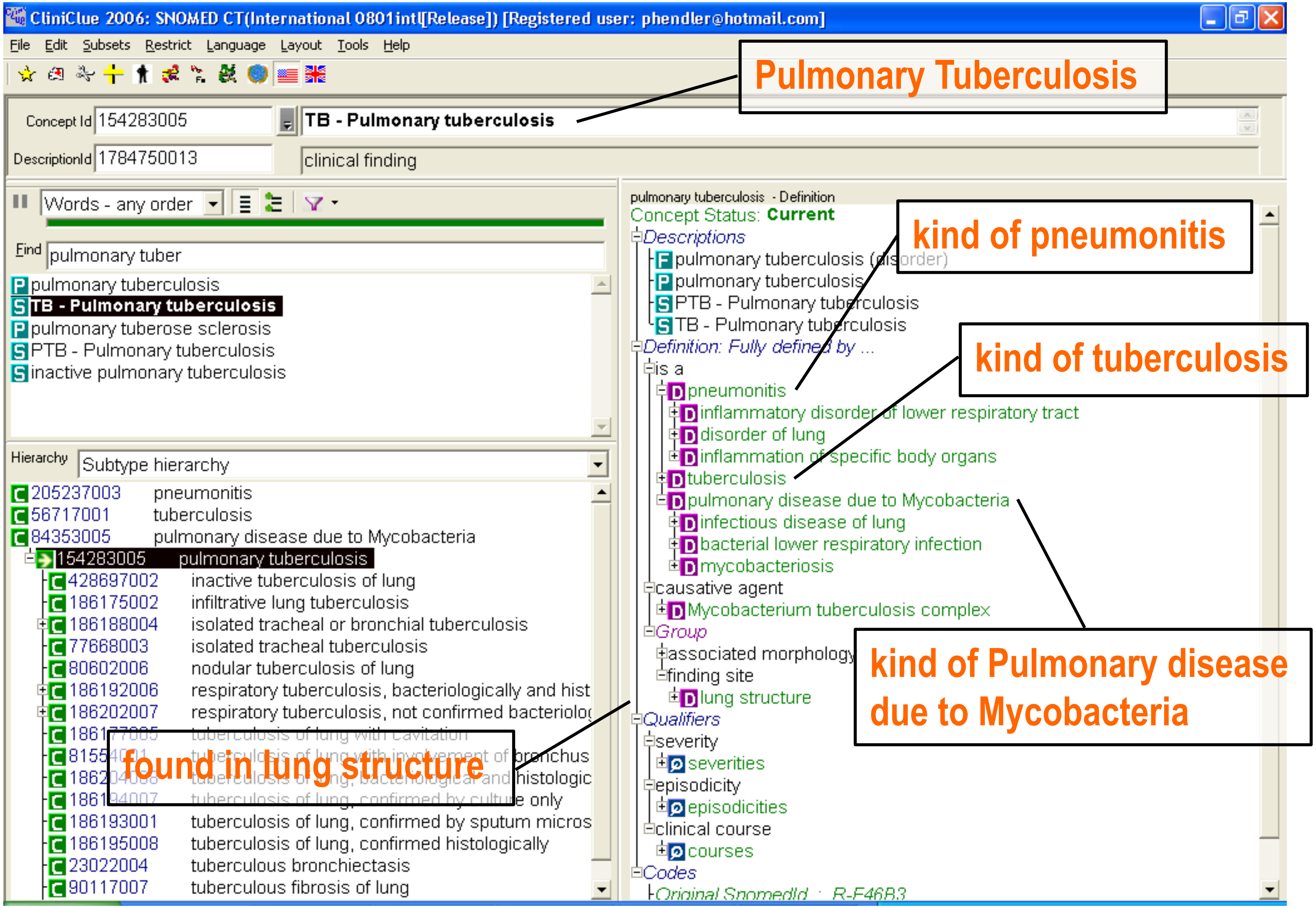
“We had sufficient information, but failed to integrate and understand it”

Healthcare

- UK NHS **£10 billion** "Connecting for Health" IT programme
- Key component is **Care Records Service** (CRS)
  - "Live, interactive patient record service accessible 24/7"
  - Patient **data distributed** across local centres in 5 regional clusters, and a national DB
  - **SNOMED-CT** ontology provides common **vocabulary** for data
    - Clinical data uses terms drawn from ontology

SNOMED-CT

- It's **BIG** – over **400,000 concepts**
- Language used is **EL profile of OWL 2**
- **Multiple hierarchies** and **rich definitions**


CliniClue 2006: SNOMED CT(International 0801intl[Release]) [Registered user: phendler@hotmail.com]
File Edit Subsets Restrict Language Layout Tools Help
Concept Id 154283005
TB - Pulmonary tuberculosis
DescriptionId 1784750013
clinical finding
Words - any order
Find pulmonary tuber
pulmonary tuberculosis
TB - Pulmonary tuberculosis
pulmonary tuberose sclerosis
PTB - Pulmonary tuberculosis
inactive pulmonary tuberculosis
Hierarchy Subtype hierarchy
205237003 pneumonitis
56717001 tuberculosis
84353005 pulmonary disease due to Mycobacteria
154283005 pulmonary tuberculosis
428697002 inactive tuberculosis of lung
186175002 infiltrative lung tuberculosis
186188004 isolated tracheal or bronchial tuberculosis
77668003 isolated tracheal tuberculosis
80602006 nodular tuberculosis of lung
186192006 respiratory tuberculosis, bacteriologically and hist
186202007 respiratory tuberculosis, not confirmed bacteriolog
186193001 tuberculosis of lung, confirmed by sputum micros
186195008 tuberculosis of lung, confirmed histologically
23022004 tuberculous bronchiectasis
90117007 tuberculous fibrosis of lung
pulmonary tuberculosis - Definition
Concept Status: Current
Descriptions
pulmonary tuberculosis (disorder)
pulmonary tuberculosis
PTB - Pulmonary tuberculosis
TB - Pulmonary tuberculosis
Definition: Fully defined by ...
is a
pneumonitis
inflammatory disorder of lower respiratory tract
disorder of lung
inflammation of specific body organs
tuberculosis
pulmonary disease due to Mycobacteria
infectious disease of lung
bacterial lower respiratory infection
mycobacteriosis
causative agent
Mycobacterium tuberculosis complex
Group
associated morphology
finding site
lung structure
Qualifiers
severity
severities
episodicity
episodicities
clinical course
courses
Codes
Original SnomedId : R-F46B3
Pulmonary Tuberculosis
kind of pneumonitis
kind of tuberculosis
kind of Pulmonary disease due to Mycobacteria
found in lung structure

What About Scalability?

- Only **useful in practice** if we can deal with large ontologies and/or large data sets
- Unfortunately, many ontology languages are highly intractable
  - OWL 2 satisfiability is **2NEXPTIME-complete** w.r.t. schema
  - and **NP-Hard** w.r.t. data (upper bound open)
- Problem addressed in practice by
  - Algorithms that work well in **typical cases**
  - Highly **optimised implementations**
  - Use of tractable fragments (aka **profiles**)

Tableau Reasoning Algorithms

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

– Reasoning tasks reducible to (un)**satisfiability**

  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
    KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

– Reasoning tasks reducible to (un)**satisfiability**

  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
    KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable

– Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
    KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
    KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease

x : HeartDisease

x : Disease

x : ∃affects.Heart

(x, y) : affects

y : Heart

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

– Reasoning tasks reducible to (un)**satisfiability**

- E.g., KB $\models$ HeartDisease $\sqsubseteq$ VascularDisease iff
  KB $\cup$ {x:(HeartDisease $\sqcap$ $\neg$VascularDisease)} is *not* satisfiable

– Algorithm tries to construct (an abstraction of) a model:

$$
\begin{aligned}
x &: \mathsf{HeartDisease} \sqcap \neg\mathsf{VascularDisease}\\
x &: \mathsf{HeartDisease}\\
x &: \mathsf{Disease}\\
x &: \exists\mathsf{affects}.\mathsf{Heart}\\
(x,y) &: \mathsf{affects}\\
y &: \mathsf{Heart}\\
y &: \mathsf{MuscularOrgan}\\
y &: \exists\mathsf{isPartOf}.\mathsf{CirculatorySystem}
\end{aligned}
$$

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease x : ¬VascularDisease

x : HeartDisease

x : Disease

x : ∃affects.Heart

(x, y) : affects

y : Heart

y : MuscularOrgan

y : ∃isPartOf.CirculatorySystem

(y, z) : isPartOf

z : CirculatorySystem

Tableau Reasoning Algorithms

- Standard technique based on (hyper-) **tableau**
  - Reasoning tasks reducible to (un)**satisfiability**
    - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
  - Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : ¬VascularDisease
x : ¬Disease ⊔ ¬∃affects.(∃isPartOf.CirculatorySystem)

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

– Reasoning tasks reducible to (un)**satisfiability**

  • E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable

– Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : ¬VascularDisease
x : ¬Disease ⊔ ¬∃affects.(∃isPartOf.CirculatorySystem)
x : ¬Disease

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model:

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : ¬VascularDisease
x : ¬Disease ⊔ ¬∃affects.(∃isPartOf.CirculatorySystem)
x : ¬Disease

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

– Reasoning tasks reducible to (un)**satisfiability**

- E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
  KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable

– Algorithm tries to construct (an abstraction of) a model

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : ¬VascularDisease
x : ¬Disease ⊔
¬∃affects.(∃isPartOf.CirculatorySystem)

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB $\models$ HeartDisease $\sqsubseteq$ VascularDisease iff KB $\cup$ {x:(HeartDisease $\sqcap$ $\neg$VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model

x : HeartDisease $\sqcap$ $\neg$VascularDisease
x : HeartDisease
x : Disease
x : $\exists$affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : $\exists$isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : $\neg$VascularDisease
x : $\neg$Disease $\sqcup$ $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)
x : $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff
    KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model

$$
\begin{array}{rl}
x: & \mathsf{HeartDisease} \sqcap \neg \mathsf{VascularDisease} \\
x: & \mathsf{HeartDisease} \\
x: & \mathsf{Disease} \\
x: & \exists \mathsf{affects}.\mathsf{Heart} \\
(x,y): & \mathsf{affects} \\
y: & \mathsf{Heart} \\
y: & \mathsf{MuscularOrgan} \\
y: & \exists \mathsf{isPartOf}.\mathsf{CirculatorySystem} \\
(y,z): & \mathsf{isPartOf} \\
z: & \mathsf{CirculatorySystem}
\end{array}
$$

$$
\begin{array}{rl}
x: & \neg \mathsf{VascularDisease} \\
x: & \neg \mathsf{Disease} \sqcup \\
 & \neg \exists \mathsf{affects}.(\exists \mathsf{isPartOf}.\mathsf{CirculatorySystem}) \\
x: & \neg \exists \mathsf{affects}.(\exists \mathsf{isPartOf}.\mathsf{CirculatorySystem}) \\
x: & \forall \mathsf{affects}.(\forall \mathsf{isPartOf}.\neg \mathsf{CirculatorySystem})
\end{array}
$$

Tableau Reasoning Algorithms

- Standard technique based on (hyper-) **tableau**
  - Reasoning tasks reducible to (un)**satisfiability**
    - E.g., KB $\models$ HeartDisease $\sqsubseteq$ VascularDisease iff KB $\cup$ {x:(HeartDisease $\sqcap$ $\neg$VascularDisease)} is *not* satisfiable
  - Algorithm tries to construct (an abstraction of) a model

x : HeartDisease $\sqcap$ $\neg$VascularDisease
x : HeartDisease
x : Disease
x : $\exists$affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : $\exists$isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : $\neg$VascularDisease
x : $\neg$Disease $\sqcup$ $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)
x : $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)
x : $\forall$affects.($\forall$isPartOf.$\neg$CirculatorySystem)
y : $\forall$isPartOf.$\neg$CirculatorySystem

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB $\models$ HeartDisease $\sqsubseteq$ VascularDisease iff KB $\cup$ {x:(HeartDisease $\sqcap$ $\neg$VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model

x : HeartDisease $\sqcap$ $\neg$VascularDisease
x : HeartDisease
x : Disease
x : $\exists$affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : $\exists$isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : $\neg$VascularDisease
x : $\neg$Disease $\sqcup$ $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)
x : $\neg\exists$affects.($\exists$isPartOf.CirculatorySystem)
x : $\forall$affects.($\forall$isPartOf.$\neg$CirculatorySystem)
y : $\forall$isPartOf.$\neg$CirculatorySystem
z : $\neg$CirculatorySystem

Tableau Reasoning Algorithms

Standard technique based on (hyper-) **tableau**

- Reasoning tasks reducible to (un)**satisfiability**
  - E.g., KB ⊨ HeartDisease ⊑ VascularDisease iff KB ∪ {x:(HeartDisease ⊓ ¬VascularDisease)} is *not* satisfiable
- Algorithm tries to construct (an abstraction of) a model

x : HeartDisease ⊓ ¬VascularDisease
x : HeartDisease
x : Disease
x : ∃affects.Heart
(x, y) : affects
y : Heart
y : MuscularOrgan
y : ∃isPartOf.CirculatorySystem
(y, z) : isPartOf
z : CirculatorySystem

x : ¬VascularDisease
x : ¬Disease ⊔ ¬∃affects.(∃isPartOf.CirculatorySystem)
x : ¬∃affects.(∃isPartOf.CirculatorySystem)
x : ∀affects.(∀isPartOf.¬CirculatorySystem)
y : ∀isPartOf.¬CirculatorySystem
z : ¬CirculatorySystem

Highly Optimised Implementations

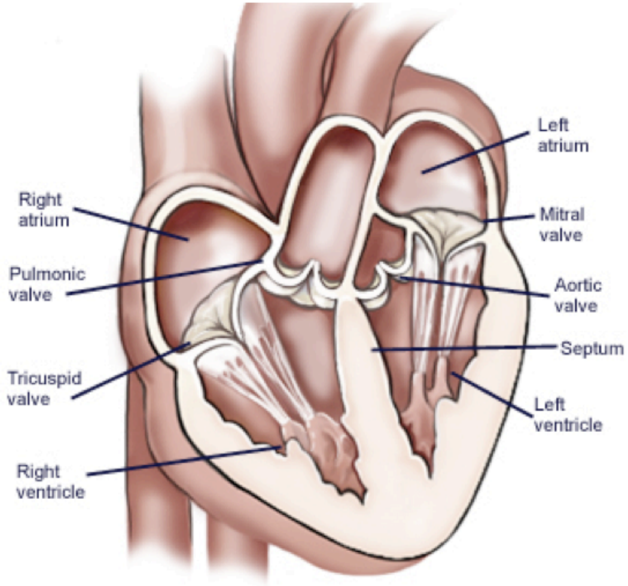
- Lazy unfolding
- Simplification and rewriting,

  e.g., $A \sqcap B \sqsubseteq C \quad \longrightarrow \quad A \sqsubseteq C \sqcup \neg B$
- HyperTableau (reduces non-determinism)
- Fast semi-decision procedures
- Search optimisations
- Reuse of previous computations
- Heuristics

Not computationally optimal,
but effective with many realistic ontologies

Scalability Issues

- Problems with very **large and/or cyclical ontologies**



$$\begin{aligned}
\mathsf{LeftSide} &\sqsubseteq \exists \mathsf{hasComponent}.\mathsf{AorticValve} \\
\mathsf{LeftSide} &\sqsubseteq \exists \mathsf{hasComponent}.\mathsf{MitralValve} \\
\mathsf{AorticValve} &\sqsubseteq \exists \mathsf{hasConnection}.\mathsf{LeftVentircle} \\
\mathsf{MitralValve} &\sqsubseteq \exists \mathsf{hasConnection}.\mathsf{LeftVentircle} \\
\mathsf{LeftVentricle} &\sqsubseteq \exists \mathsf{isDivisionOf}.\mathsf{LeftSide}
\end{aligned}$$

- Ontologies may define 10s/100s of thousands of terms
- Can lead to construction of *very* large models

Scalability Issues

- Problems with **large data sets** (ABoxes)
  - Main reasoning problem is (conjunctive) query answering, e.g., retrieve all patients suffering from vascular disease:
    $Q(x) \leftarrow \mathsf{Patient}(x) \wedge \mathsf{suffersFrom}(x,y) \wedge \mathsf{VascularDisease}(y)$
  - Decidability still open for OWL, although minor restrictions (on cycles in non-distinguished variables) restore decidability
  - Query answering reduced to standard decision problem, e.g., by checking for each individual x if KB $\models \mathcal{Q}(x)$
  - Model construction starts with *all* ground facts (data)
- Typical applications may use data sets with **10s/100s of millions** of individuals (or more)

OWL 2 Profiles

- OWL recommendation now updated to **OWL 2**
- OWL 2 defines several **profiles** – fragments with desirable computational properties
  - **OWL 2 EL** targeted at very large ontologies
  - **OWL 2 QL** targeted at very large data sets

OWL 2 EL

- A (near maximal) fragment of OWL 2 such that
  - Satisfiability checking is in PTime (**PTime-Complete**)
  - Data complexity of query answering also PTime-Complete
- Based on $\mathcal{EL}$ family of description logics
- Can exploit **saturation** based reasoning techniques
  - Computes classification in "one pass"
  - Computationally optimal
  - Can be extended to Horn fragment of OWL DL

Saturation-based Technique (basics)

- Normalise ontology axioms to standard form:

$$A \sqsubseteq B \quad A \sqcap B \sqsubseteq C \quad A \sqsubseteq \exists R.B \quad \exists R.B \sqsubseteq C$$

- Saturate using inference rules:

$$\frac{A \sqsubseteq B \quad B \sqsubseteq C}{A \sqsubseteq C} \qquad \frac{A \sqsubseteq B \quad A \sqsubseteq C \quad B \sqcap C \sqsubseteq D}{A \sqsubseteq D}$$

$$\frac{A \sqsubseteq \exists R.B \quad B \sqsubseteq C \quad \exists R.C \sqsubseteq D}{A \sqsubseteq D}$$

- Extension to Horn fragment requires (many) more rules

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$

$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$

$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

Saturation-based Technique (basics)

Example:

$$\text{OrganTransplant} \equiv \text{Transplant} \sqcap \exists \text{site}.\text{Organ}$$
$$\text{HeartTransplant} \equiv \text{Transplant} \sqcap \exists \text{site}.\text{Heart}$$
$$\text{Heart} \sqsubseteq \text{Organ}$$

$$\text{OrganTransplant} \sqsubseteq \text{Transplant}$$
$$\text{OrganTransplant} \sqsubseteq \exists \text{site}.\text{Organ}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$

$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$

$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$

$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$

$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$

$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$

$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$

$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$

$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$

$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$

$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$

$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$

$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$

$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$

$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$

$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\frac{A \sqsubseteq \exists R.B \quad B \sqsubseteq C \quad \exists R.C \sqsubseteq D}{A \sqsubseteq D}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\frac{A \sqsubseteq \exists R.B \quad B \sqsubseteq C \quad \exists R.C \sqsubseteq D}{A \sqsubseteq D}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{SO}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\frac{A \sqsubseteq B \quad A \sqsubseteq C \quad B \sqcap C \sqsubseteq D}{A \sqsubseteq D}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{SO}$$

Saturation-based Technique (basics)

Example:

$$\mathsf{OrganTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Organ}$$
$$\mathsf{HeartTransplant} \equiv \mathsf{Transplant} \sqcap \exists \mathsf{site}.\mathsf{Heart}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\frac{A \sqsubseteq B \quad A \sqsubseteq C \quad B \sqcap C \sqsubseteq D}{A \sqsubseteq D}$$

$$\mathsf{OrganTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{OrganTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Organ}$$
$$\exists \mathsf{site}.\mathsf{Organ} \sqsubseteq \mathsf{SO}$$
$$\mathsf{Transplant} \sqcap \mathsf{SO} \sqsubseteq \mathsf{OrganTransplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{Transplant}$$
$$\mathsf{HeartTransplant} \sqsubseteq \exists \mathsf{site}.\mathsf{Heart}$$
$$\exists \mathsf{site}.\mathsf{Heart} \sqsubseteq \mathsf{SH}$$
$$\mathsf{Transplant} \sqcap \mathsf{SH} \sqsubseteq \mathsf{HeartTransplant}$$
$$\mathsf{Heart} \sqsubseteq \mathsf{Organ}$$

$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{SO}$$
$$\mathsf{HeartTransplant} \sqsubseteq \mathsf{OrganTransplant}$$

Saturation-based Technique

Performance with large bio-medical ontologies:

	GO	NCI	Galen v.0	Galen v.7	SNOMED
Concepts:	20465	27652	2748	23136	389472
FaCT++	15.24	6.05	465.35	—	650.37
HermiT	199.52	169.47	45.72	—	—
Pellet	72.02	26.47	—	—	—
CEL	1.84	5.76	—	—	1185.70
CB	1.17	3.57	0.32	9.58	49.44
Speed-Up:	1.57X	1.61X	143X	∞	13.15X

OWL 2 QL

- A (near maximal) fragment of OWL 2 such that
  - Data complexity of conjunctive query answering in **AC^0**
- Based on **DL-Lite** family of description logics
- Can exploit **query rewriting** based reasoning technique
  - Computationally optimal
  - Data storage and query evaluation can be delegated to standard RDBMS
  - Can be extended to more expressive languages (beyond AC^0) by delegating query answering to a Datalog engine

Query Rewriting Technique (basics)

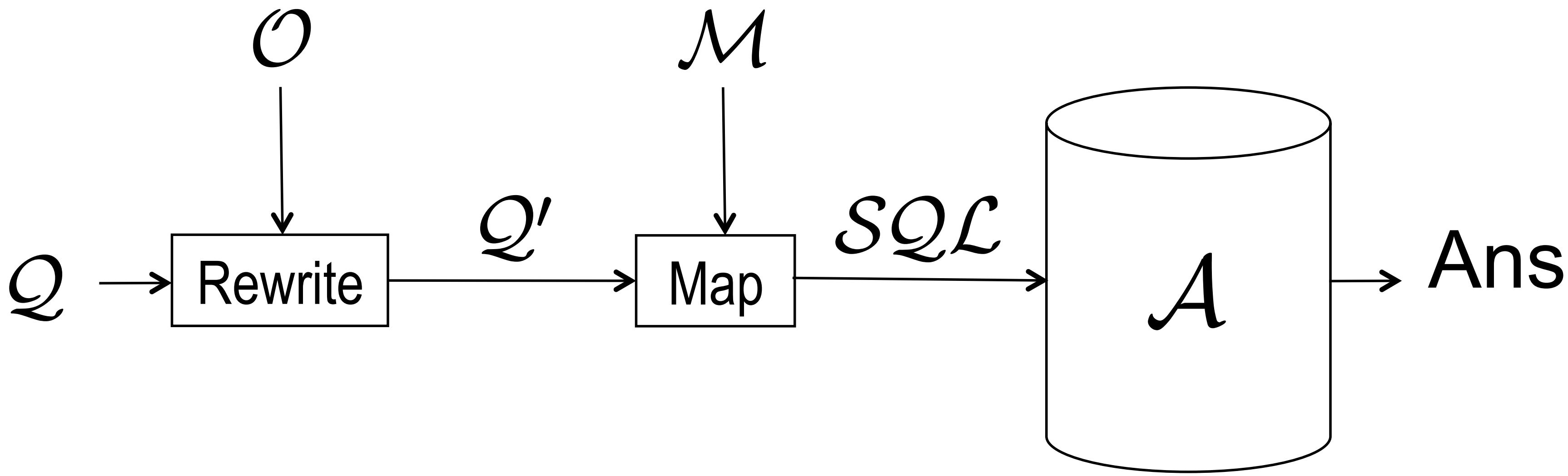
- Given ontology $\mathcal{O}$ and query $\mathcal{Q}$, use $\mathcal{O}$ to rewrite $\mathcal{Q}$ as $\mathcal{Q}'$ s.t., for any set of ground facts $\mathcal{A}$:
  - ans($\mathcal{Q}$, $\mathcal{O}$, $\mathcal{A}$) = ans($\mathcal{Q}'$, $\emptyset$, $\mathcal{A}$)

Query Rewriting Technique (basics)

- Given ontology $\mathcal{O}$ and query $\mathcal{Q}$, use $\mathcal{O}$ to rewrite $\mathcal{Q}$ as $\mathcal{Q}'$ s.t., for any set of ground facts $\mathcal{A}$:
  - ans($\mathcal{Q}$, $\mathcal{O}$, $\mathcal{A}$) = ans($\mathcal{Q}'$, $\emptyset$, $\mathcal{A}$)
- Use (GAV) mapping $\mathcal{M}$ to map $\mathcal{Q}'$ to SQL query

Query Rewriting Technique (basics)

- Given ontology $\mathcal{O}$ and query $\mathcal{Q}$, use $\mathcal{O}$ to rewrite $\mathcal{Q}$ as $\mathcal{Q}'$ s.t., for any set of ground facts $\mathcal{A}$:
  - ans($\mathcal{Q}$, $\mathcal{O}$, $\mathcal{A}$) = ans($\mathcal{Q}'$, $\emptyset$, $\mathcal{A}$)
- Use (GAV) mapping $\mathcal{M}$ to map $\mathcal{Q}'$ to SQL query



Query Rewriting Technique (basics)

- Given ontology $\mathcal{O}$ and query $\mathcal{Q}$, use $\mathcal{O}$ to rewrite $\mathcal{Q}$ as $\mathcal{Q}'$ s.t., for any set of ground facts $\mathcal{A}$:
  - ans($\mathcal{Q}$, $\mathcal{O}$, $\mathcal{A}$) = ans($\mathcal{Q}'$, $\emptyset$, $\mathcal{A}$)
- Use (GAV) mapping $\mathcal{M}$ to map $\mathcal{Q}'$ to SQL query
- Resolution based query rewriting
  - **Clausify** ontology axioms
  - **Saturate** (clausified) ontology and query using resolution
  - **Prune** redundant query clauses

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$

$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$

Query Rewriting Technique (basics)

- Example:

$$\text{Doctor} \sqsubseteq \exists \text{treats}.\text{Patient}$$
$$\text{Consultant} \sqsubseteq \text{Doctor}$$

$$\text{treats}(x, f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Patient}(f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Doctor}(x) \leftarrow \text{Consultant}(x)$$

$$Q(x) \leftarrow \text{treats}(x, y) \wedge \text{Patient}(y)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$$

Query Rewriting Technique (basics)

- Example:

$$\text{Doctor} \sqsubseteq \exists \text{treats}.\text{Patient}$$
$$\text{Consultant} \sqsubseteq \text{Doctor}$$

$$\text{treats}(x, f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Patient}(f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Doctor}(x) \leftarrow \text{Consultant}(x)$$

$$Q(x) \leftarrow \text{treats}(x, y) \wedge \text{Patient}(y)$$
$$Q(x) \leftarrow \text{Doctor}(x) \wedge \text{Patient}(f(x))$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$

$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$

$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$

$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$

$$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$$

$$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \wedge \mathsf{Doctor}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$$
$$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \wedge \mathsf{Doctor}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \land \mathsf{Patient}(y)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x) \land \mathsf{Patient}(f(x))$$
$$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \land \mathsf{Doctor}(x)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$$
$$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \wedge \mathsf{Doctor}(x)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$$
$$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \wedge \mathsf{Doctor}(x)$$
$$Q(x) \leftarrow \mathsf{Doctor}(x)$$
$$Q(x) \leftarrow \mathsf{Consultant}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\text{Doctor} \sqsubseteq \exists \text{treats}.\text{Patient}$$
$$\text{Consultant} \sqsubseteq \text{Doctor}$$

$$\text{treats}(x, f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Patient}(f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Doctor}(x) \leftarrow \text{Consultant}(x)$$

$$Q(x) \leftarrow \text{treats}(x, y) \wedge \text{Patient}(y)$$
$$Q(x) \leftarrow \text{Doctor}(x) \wedge \text{Patient}(f(x))$$
$$Q(x) \leftarrow \text{treats}(x, f(x)) \wedge \text{Doctor}(x)$$
$$Q(x) \leftarrow \text{Doctor}(x)$$
$$Q(x) \leftarrow \text{Consultant}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\mathsf{Doctor} \sqsubseteq \exists \mathsf{treats}.\mathsf{Patient}$$
$$\mathsf{Consultant} \sqsubseteq \mathsf{Doctor}$$

$$\mathsf{treats}(x, f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Patient}(f(x)) \leftarrow \mathsf{Doctor}(x)$$
$$\mathsf{Doctor}(x) \leftarrow \mathsf{Consultant}(x)$$

$$Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y)$$
~~$Q(x) \leftarrow \mathsf{Doctor}(x) \wedge \mathsf{Patient}(f(x))$~~
~~$Q(x) \leftarrow \mathsf{treats}(x, f(x)) \wedge \mathsf{Doctor}(x)$~~
$$Q(x) \leftarrow \mathsf{Doctor}(x)$$
$$Q(x) \leftarrow \mathsf{Consultant}(x)$$

Query Rewriting Technique (basics)

- Example:

$$\text{Doctor} \sqsubseteq \exists\text{treats}.\text{Patient}$$
$$\text{Consultant} \sqsubseteq \text{Doctor}$$

$$\text{treats}(x, f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Patient}(f(x)) \leftarrow \text{Doctor}(x)$$
$$\text{Doctor}(x) \leftarrow \text{Consultant}(x)$$

$$Q(x) \leftarrow \text{treats}(x, y) \wedge \text{Patient}(y)$$
~~$Q(x) \leftarrow \text{Doctor}(x) \wedge \text{Patient}(f(x))$~~
~~$Q(x) \leftarrow \text{treats}(x, f(x)) \wedge \text{Doctor}(x)$~~
$$Q(x) \leftarrow \text{Doctor}(x)$$
$$Q(x) \leftarrow \text{Consultant}(x)$$

- For DL-Lite, result is a union of conjunctive queries

$$Q(x) \leftarrow (\text{treats}(x, y) \wedge \text{Patient}(y)) \vee \text{Doctor}(x) \vee \text{Consultant}(x)$$

Query Rewriting Technique (basics)

- Data can be stored/left in **RDBMS**
- Relationship between ontology and DB defined by **mappings**, e.g.:

$$\begin{aligned} \text{Doctor} &\mapsto \text{SELECT Name FROM Doctor} \\ \text{Patient} &\mapsto \text{SELECT Name FROM Patient} \\ \text{treats} &\mapsto \text{SELECT DName, PName FROM Treats} \end{aligned}$$

Query Rewriting Technique (basics)

- Data can be stored/left in **RDBMS**
- Relationship between ontology and DB defined by **mappings**, e.g.:

$$\begin{aligned}\mathsf{Doctor} &\mapsto \mathsf{SELECT\ Name\ FROM\ Doctor} \\ \mathsf{Patient} &\mapsto \mathsf{SELECT\ Name\ FROM\ Patient} \\ \mathsf{treats} &\mapsto \mathsf{SELECT\ DName,\ PName\ FROM\ Treats}\end{aligned}$$

- UCQ translated into **SQL query**:

$$Q(x) \leftarrow (\mathsf{treats}(x,y) \wedge \mathsf{Patient}(y)) \vee \mathsf{Doctor}(x) \vee \mathsf{Consultant}(x)$$

$$\rightsquigarrow$$

SELECT Name FROM Doctor UNION
SELECT DName FROM Treats, Patient WHERE PName=Name

Problems & Research Challenges

- Combining best features of DLs & DBs
  - In particular, integrating OWA and CWA
- Hard to find a coherent semantic framework
  - Problems mainly due to existential quantifiers: should existentially implied objects be considered different?
    - Does a person owning a phone and an ipod own 2 things?
    - Does a person owning a phone and an iphone own 2 things?
    - Does a person owning a phone and a phone own 2 things?
- Interesting ideas emerging in DL & DB communities, e.g.:
  - *Calì et al. Datalog±: a unified approach to ontologies and integrity constraints. ICDT 2009.*
  - *Motik et al. Bridging the gap between OWL and relational databases. WWW 2007.*

Problems & Research Challenges

- Open questions w.r.t. query rewriting

Problems & Research Challenges

- Open questions w.r.t. query rewriting
  - Currently only for very weak ontology languages

Problems & Research Challenges

- Open questions w.r.t. query rewriting
  - Currently only for very weak ontology languages
  - Even for these languages, queries can get very large (order $(|\mathcal{O}| \cdot |\mathcal{Q}|)^{|\mathcal{Q}|}$), and existing RDBMSs may behave poorly
    - Not clear if this will be a problem in practice, see, e.g., *Savo et al. MASTRO at Work: Experiences on Ontology-based Data Access. DL 2010.*

Problems & Research Challenges

- Open questions w.r.t. query rewriting
  - Currently only for very weak ontology languages
  - Even for these languages, queries can get very large (order $(|\mathcal{O}| \cdot |\mathcal{Q}|)^{|\mathcal{Q}|}$), and existing RDBMSs may behave poorly
    - Not clear if this will be a problem in practice, see, e.g., *Savo et al. MASTRO at Work: Experiences on Ontology-based Data Access. DL 2010.*
  - Larger fragments require (at least) Datalog engines and/or extension to technique (e.g., partial materialisation)
    - Promising new work in this area, see, e.g., *Lutz et al. Conjunctive Query Answering in the Description Logic EL Using a Relational Database System. IJCAI 2009.*

Problems & Research Challenges

- Infrastructure

Problems & Research Challenges

- Infrastructure
  - Standardised query language
    - SPARQL standard for RDF
    - Currently being extended for OWL, see *http://www.w3.org/2009/sparql/wiki/Main_Page*

Problems & Research Challenges

- Infrastructure
  - Standardised query language
    - SPARQL standard for RDF
    - Currently being extended for OWL, see *http://www.w3.org/2009/sparql/wiki/Main_Page*
  - Privacy and information hiding
    - May want to keep parts of data/schema private
    - Difficulties compounded when information can be inferred, see, e.g., *Cuenca Grau et al. Privacy-preserving query answering in logic-based information systems. ECAI 2008.*

Problems & Research Challenges

- Infrastructure
  - Standardised query language
    - SPARQL standard for RDF
    - Currently being extended for OWL, see *http://www.w3.org/2009/sparql/wiki/Main_Page*
  - Privacy and information hiding
    - May want to keep parts of data/schema private
    - Difficulties compounded when information can be inferred, see, e.g., *Cuenca Grau et al. Privacy-preserving query answering in logic-based information systems. ECAI 2008.*
  - ...

The 9th International Semantic Web Conference
Shanghai International Convention Center, Shanghai, China
Nov 7th -11th, 2010
http://iswc2010.semanticweb.org
ISWC 2010
Semantics for a Better Web!
General Chair
Ian Horrocks
Program Chairs
Peter F. Patel-Schneider
Yue Pan
Local Chair
Yong Yu
Workshop & Tutorial Chairs
Philippe Cudré-Mauroux
Bijan Parsia
Poster & Demo Chairs
Huajun Chen
Axel Polleres
Industry & Semantic Web in Use Chairs
Pascal Hitzler
Peter Mika
Doctoral Consortium Chair
Jeff Pan
Semantic Web Challenge Chairs
Chris Bizer
Diana Maynard
Publicity Chair
Sebastian Rudolph
Metadata Chair
Jie Bao
Proceedings Chair
Birte Glimm
Sponsor Chairs
Kendall Clark
Anand Ranganathan
Local Organization
Dingyi Han
Gui-Rong Xue
Haofen Wang
Lei Zhang
Stunning Venue
Great Time
Fascinating City
Excellent Food
SWSA
Semantic Web Science Association
上海交通大學
SHANGHAI JIAO TONG UNIVERSITY
IBM
Research

Thanks To

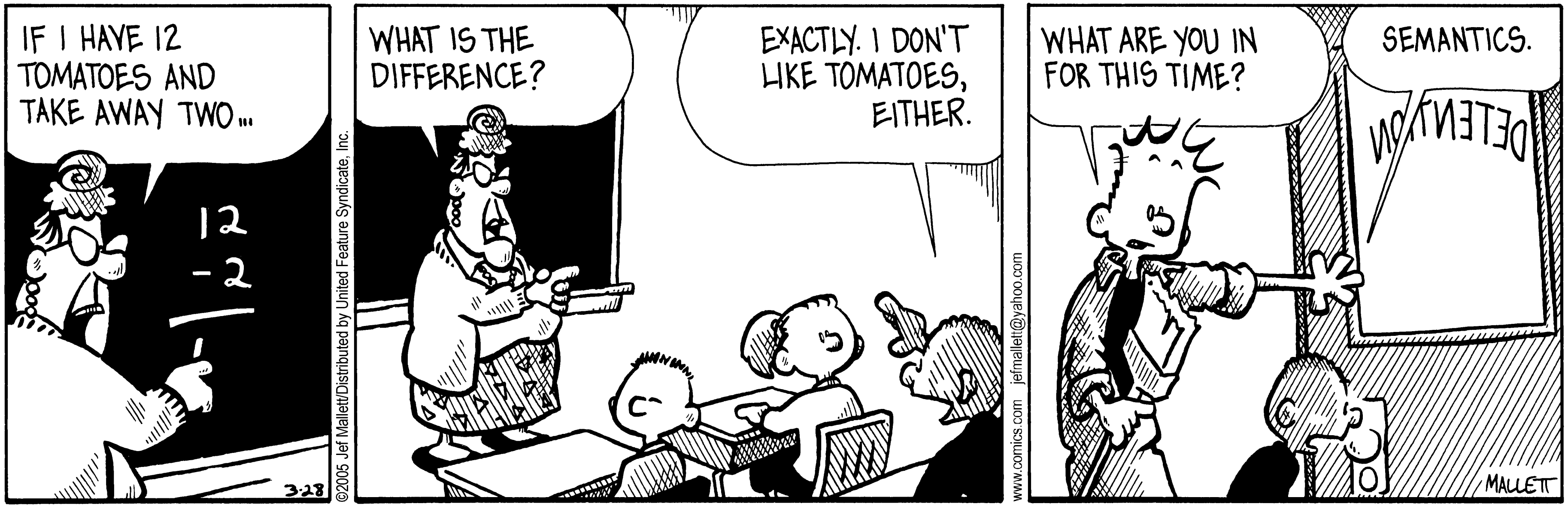
- Boris Motik
- Yevgeny Kazakov
- Héctor Pérez-Urbina
- Rob Shearer
- Bernardo Cuenca Grau
- Birte Glimm






Thank you for listening

Thank you for listening



FRAZZ: © Jeff Mallett/Dist. by United Feature Syndicate, Inc.

Any questions?

Select Bibliography

[1] Baader, Horrocks, and Sattler. Description Logics. In *Handbook of Knowledge Representation*. Elsevier, 2007.

[2] Motik, Shearer, and Horrocks. Hypertableau reasoning for description logics. J. of Artificial Intelligence Research, 2009.

[3] Baader, Brandt, and Lutz. Pushing the EL envelope. IJCAI 2005, pages 364–369, 2005.

[4] Kazakov. Consequence-driven reasoning for Horn-SHIQ ontologies. IJCAI 2009, pages 2040–2045, 2009.

[5] Calvanese, De Giacomo, Lembo, Lenzerini, and Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Automated Reasoning, 39(3):385–429, 2007.

[6] Perez-Urbina, Motik, and Horrocks. Tractable query answering and rewriting under description logic constraints. J. of Applied Logic, 2009.

[7] Andrea Calì, Georg Gottlob, Thomas Lukasiewicz. Datalog±: a unified approach to ontologies and integrity constraints. ICDT 2009: 14–30.