Logical Foundations for the Semantic Web

Ian Horrocks and **Ulrike Sattler**
University of Manchester
Manchester, UK
{horrocks|sattler}@cs.man.ac.uk

Introduction

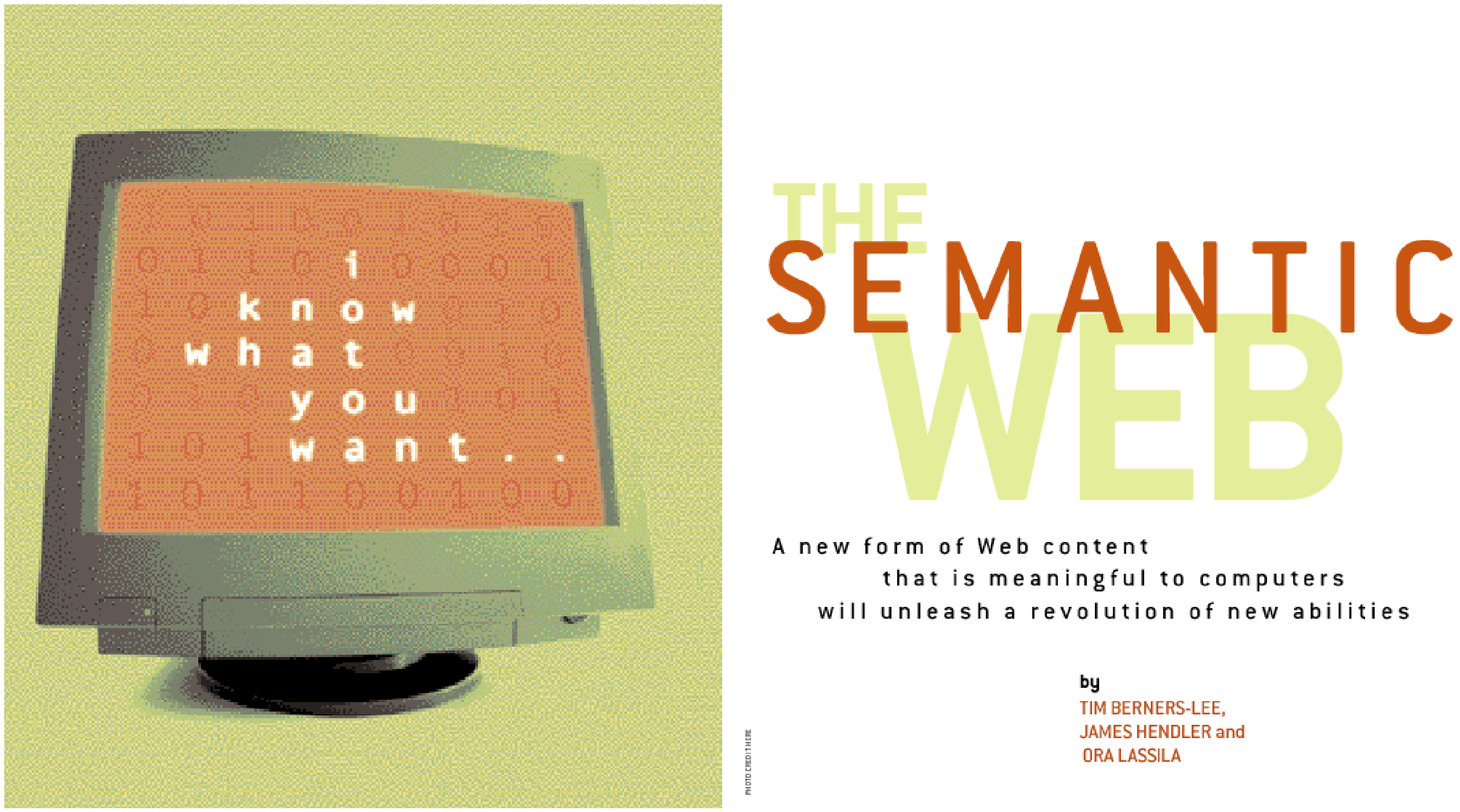
History of the Semantic Web

- **Web was "invented" by Tim Berners-Lee (amongst others), a physicist working at CERN**
- **TBL's original vision of the Web was much more ambitious than the reality of the existing (syntactic) Web:**

"... a goal of the Web was that, if the interaction between person and hypertext could be so intuitive that the **machine-readable** information space gave an accurate representation of the state of people's thoughts, interactions, and work patterns, then **machine analysis** could become a very powerful management tool, seeing patterns in our work and facilitating our working together through the typical problems which beset the management of large organizations."

- **TBL (and others) have since been working towards realising this vision, which has become known as the Semantic Web**
  - **E.g., article in May 2001 issue of Scientific American…**

Scientific American, May 2001:



THE SEMANTIC WEB

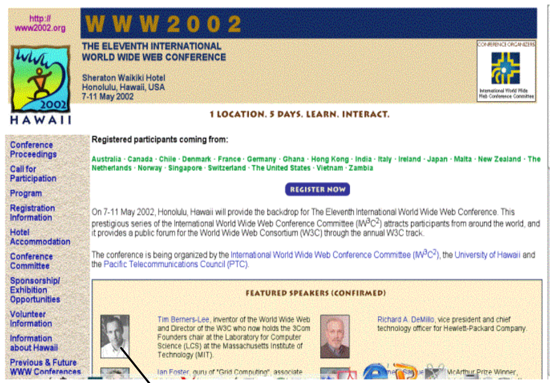
A new form of Web content that is meaningful to computers will unleash a revolution of new abilities

by
TIM BERNERS-LEE,
JAMES HENDLER and
ORA LASSILA

PHOTO CREDIT HERE

- Realising the complete “vision” is too hard for now (probably)
- But we can make a start by adding semantic annotation to web resources

Where we are Today: the Syntactic Web



http://
www2002.org

WWW2002

THE ELEVENTH INTERNATIONAL
WORLD WIDE WEB CONFERENCE

Sheraton Waikiki Hotel
Honolulu, Hawaii, USA
7-11 May 2002

CONFERENCE ORGANIZERS
International World Wide Web Conference Committee

HAWAII

1 LOCATION. 5 DAYS. LEARN. INTERACT.

Conference Proceedings
Call for Participation
Program
Registration Information
Hotel Accommodation
Conference Committee
Sponsorship/ Exhibition Opportunities
Volunteer Information
Information about Hawaii
Previous & Future WWW Conferences

Registered participants coming from:

Australia · Canada · Chile · Denmark · France · Germany · Ghana · Hong Kong · India · Italy · Ireland · Japan · Malta · New Zealand · The Netherlands · Norway · Singapore · Switzerland · The United States · Vietnam · Zambia

REGISTER NOW

On 7-11 May 2002, Honolulu, Hawaii will provide the backdrop for The Eleventh International World Wide Web Conference. This prestigious series of the International World Wide Web Conference Committee (IW^{3}C^2) attracts participants from around the world, and it provides a public forum for the World Wide Web Consortium (W3C) through the annual W3C track.

The conference is being organized by the International World Wide Web Conference Committee (IW^{3}C^2), the University of Hawaii and the Pacific Telecommunications Council (PTC).

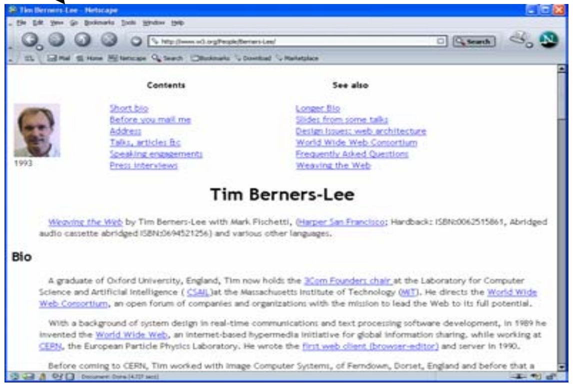
FEATURED SPEAKERS (CONFIRMED)

Tim Berners-Lee, inventor of the World Wide Web and Director of the W3C who now holds the 3Com Founders chair at the Laboratory for Computer Science (LCS) at the Massachusetts Institute of Technology (MIT).

Richard A. DeMillo, vice president and chief technology officer for Hewlett-Packard Company.

Ian Foster, guru of "Grid Computing", associate

McArthur Prize Winner,



Tim Berners-Lee - Netscape

File Edit View Go Bookmarks Tools Window Help

http://www.w3.org/People/Berners-Lee/

Search

Mail Home Netscape Search Bookmarks Download Marketplace

Contents	See also
Short bio	Longer Bio
Before you mail me	Slides from some talks
Address	Design Issues: web architecture
Talks, articles &c	World Wide Web Consortium
Speaking engagements	Frequently Asked Questions
Press interviews	Weaving the Web

1993

Tim Berners-Lee

Weaving the Web by Tim Berners-Lee with Mark Fischetti, (Harper San Francisco; Handback: ISBN:0062515861, Abridged audio cassette abridged ISBN:0694521256) and various other languages.

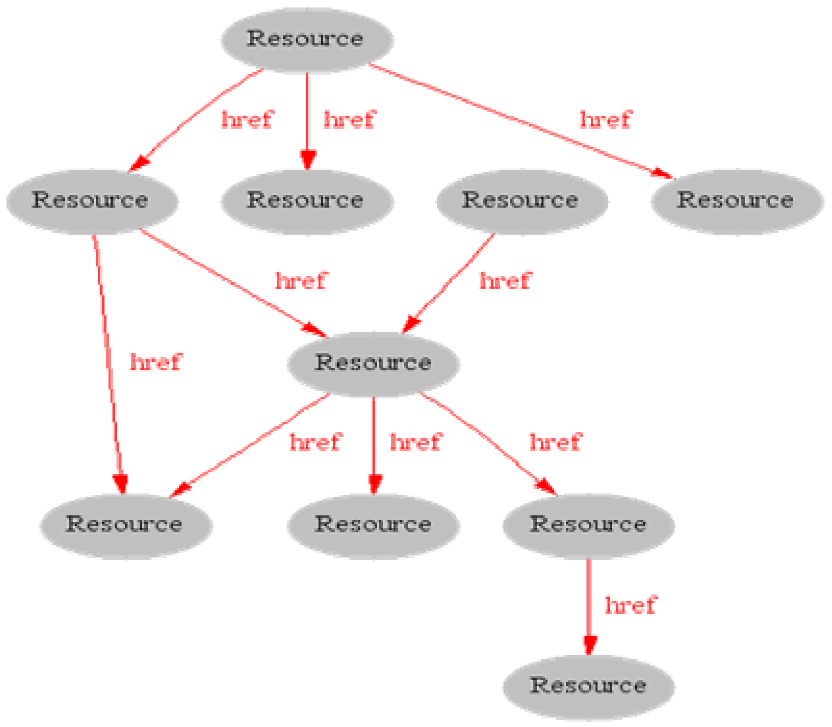
Bio

A graduate of Oxford University, England, Tim now holds the 3Com Founders chair at the Laboratory for Computer Science and Artificial Intelligence (CSAIL)at the Massachusetts Institute of Technology (MIT). He directs the World Wide Web Consortium, an open forum of companies and organizations with the mission to lead the Web to its full potential.

With a background of system design in real-time communications and text processing software development, in 1989 he invented the World Wide Web, an internet-based hypermedia initiative for global information sharing, while working at CERN, the European Particle Physics Laboratory. He wrote the first web client (browser-editor) and server in 1990.

Before coming to CERN, Tim worked with Image Computer Systems, of Ferndown, Dorset, England and before that a

Document: Done (4.727 secs)



$$\begin{tikzcd}
& \text{Resource} \arrow[dl, "href"'] \arrow[d, "href"] \arrow[drr, "href"] & & \\
\text{Resource} \arrow[dr, "href"] \arrow[dd, "href"'] & \text{Resource} & \text{Resource} \arrow[dl, "href"] & \text{Resource} \\
& \text{Resource} \arrow[dl, "href"] \arrow[d, "href"] \arrow[dr, "href"] & & \\
\text{Resource} & \text{Resource} & \text{Resource} \arrow[d, "href"] & \\
& & \text{Resource} &
\end{tikzcd}$$

[Hendler & Miller 02]

The Syntactic Web is…

- A hypermedia, a digital library
  - A library of documents called (web pages) interconnected by a hypermedia of links
- A database, an application platform
  - A common portal to applications accessible through web pages, and presenting their results as web pages
- A platform for multimedia
  - BBC Radio 4 anywhere in the world! Terminator 3 trailers!
- A naming scheme
  - Unique identity for those documents

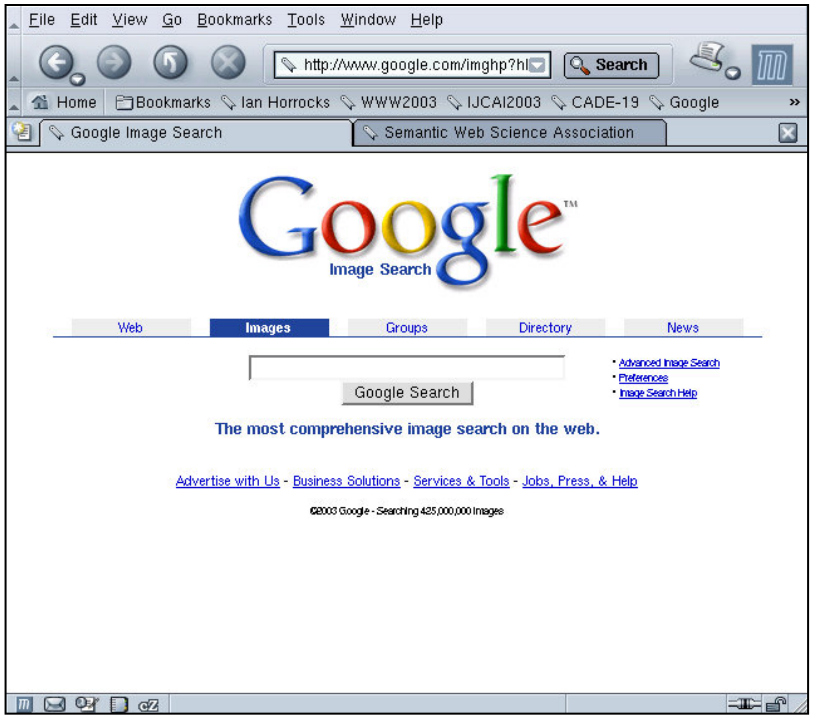
A place where computers do the presentation (easy) and people do the linking and interpreting (hard).

Why not get computers to do more of the hard work?

[Goble 03]

Hard Work using the Syntactic Web…

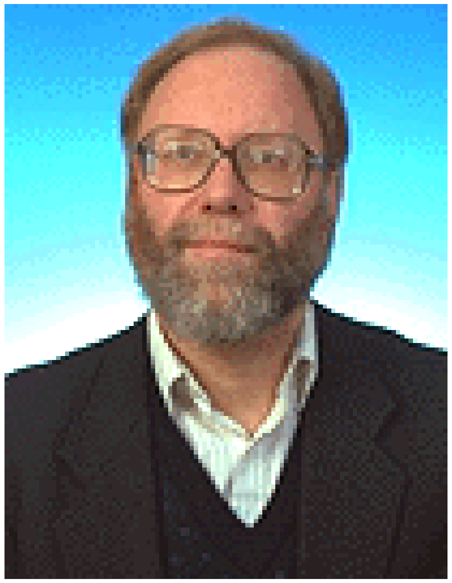
Find images of Peter Patel-Schneider, Frank van Harmelen and Alan Rector…



The most comprehensive image search on the web.

Advertise with Us - Business Solutions - Services & Tools - Jobs, Press, & Help

©2003 Google - Searching 425,000,000 images



Rev. Alan M. Gates, Associate Rector of the Church of the Holy Spirit, Lake Forest, Illinois

Hard Work using the Syntactic Web…

Independent.co.uk

To bee or not to bee

Search engines may be remarkable reso
Will a new 'semantic' web be clever enc
a flying insect from a work of music?

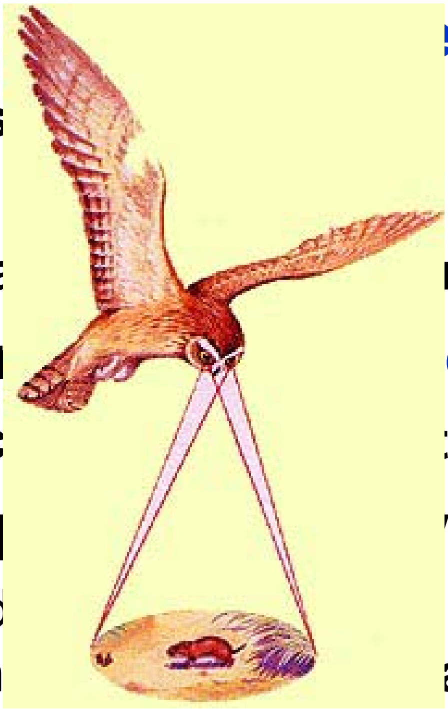
18 June 2003

Web searches have always been a bit hit and mi Even when your searches are clearly defined, you'll turn up irrelevant web ges that happen to have the same keywords. Looking for details of bumble bees' flight? Google's first result points to the composer Rimsky-Korsakov…

Semantic Web Hype:
“We’ll soon be letting machines do the thinking for us”

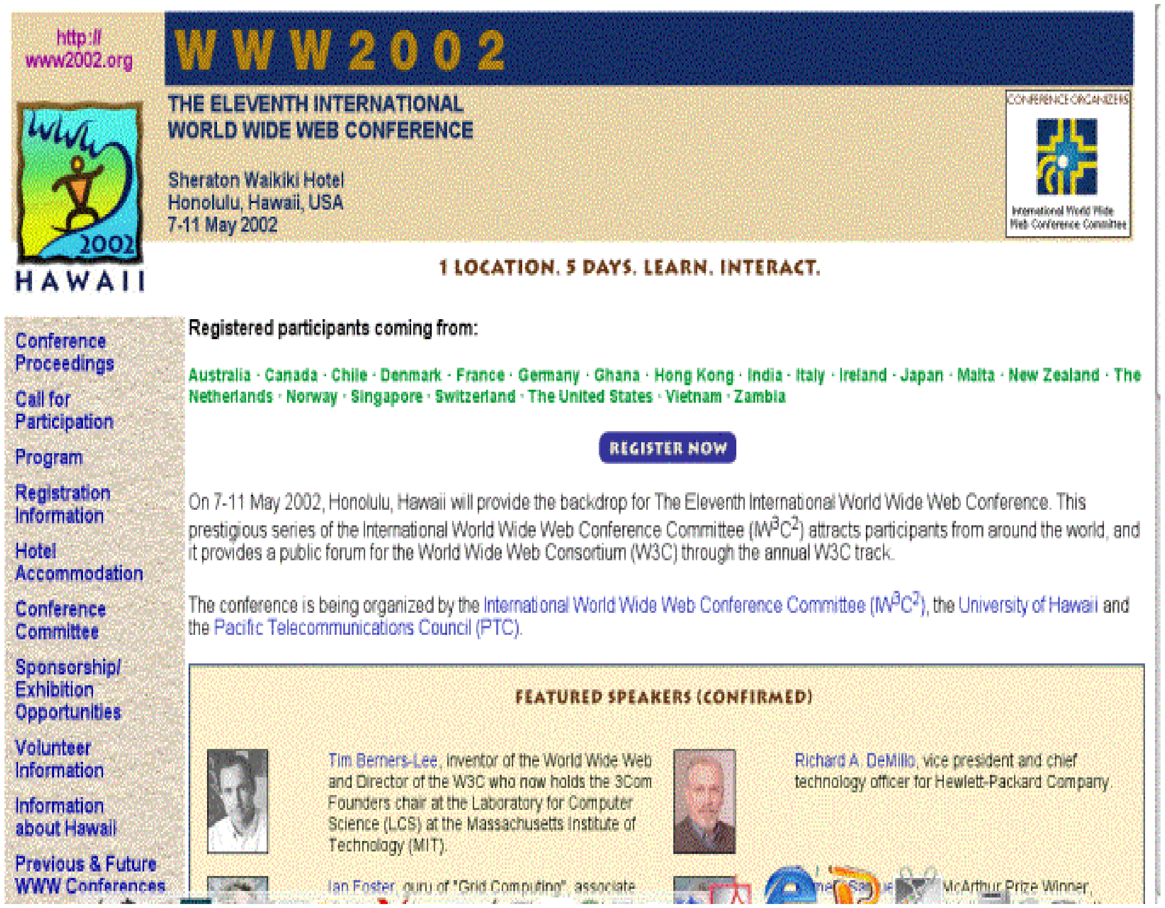
Impossible (?) using the Syntactic Web…

- Complex queries involving background knowledge
  - Find information about "animals that use sonar but are not either bats or dolphins", e.g., Barn Owl
- Locating inform epositories
  - Travel enquiries
  - Prices of goods
  - Results of huma ments
- Finding and usi es"
  - Visualise surfac ween two proteins
- Delegating comp eb "agents"
  - Book me a holid somewhere warm, not too far away, an ak French or English



What is the Problem?

- **Consider a typical web page:**



http://www2002.org

WWW2002

THE ELEVENTH INTERNATIONAL WORLD WIDE WEB CONFERENCE

Sheraton Waikiki Hotel
Honolulu, Hawaii, USA
7-11 May 2002

HAWAII

1 LOCATION. 5 DAYS. LEARN. INTERACT.

Conference Proceedings
Call for Participation
Program
Registration Information
Hotel Accommodation
Conference Committee
Sponsorship/ Exhibition Opportunities
Volunteer Information
Information about Hawaii
Previous & Future WWW Conferences

Registered participants coming from:

Australia · Canada · Chile · Denmark · France · Germany · Ghana · Hong Kong · India · Italy · Ireland · Japan · Malta · New Zealand · The Netherlands · Norway · Singapore · Switzerland · The United States · Vietnam · Zambia

REGISTER NOW

On 7-11 May 2002, Honolulu, Hawaii will provide the backdrop for The Eleventh International World Wide Web Conference. This prestigious series of the International World Wide Web Conference Committee (IW^{3}C^2) attracts participants from around the world, and it provides a public forum for the World Wide Web Consortium (W3C) through the annual W3C track.

The conference is being organized by the International World Wide Web Conference Committee (IW^{3}C^2), the University of Hawaii and the Pacific Telecommunications Council (PTC).

FEATURED SPEAKERS (CONFIRMED)

Tim Berners-Lee, inventor of the World Wide Web and Director of the W3C who now holds the 3Com Founders chair at the Laboratory for Computer Science (LCS) at the Massachusetts Institute of Technology (MIT).

Richard A. DeMillo, vice president and chief technology officer for Hewlett-Packard Company.

Ian Foster, guru of "Grid Computing", associate

McArthur Prize Winner,

- **Markup consists of:**
  - rendering information (e.g., font size and colour)
  - Hyper-links to related content
- **Semantic content is accessible to humans but not (easily) to computers…**

What information can we see…

WWW2002
The eleventh international world wide web conference
Sheraton waikiki hotel
Honolulu, hawaii, USA
7-11 may 2002
1 location 5 days learn interact
Registered participants coming from
australia, canada, chile denmark, france, germany, ghana, hong kong, india, ireland, italy, japan, malta, new zealand, the netherlands, norway, singapore, switzerland, the united kingdom, the united states, vietnam, zaire
Register now
On the 7th May Honolulu will provide the backdrop of the eleventh international world wide web conference. This prestigious event …
Speakers confirmed
Tim berners-lee
Tim is the well known inventor of the Web, …
Ian Foster
Ian is the pioneer of the Grid, the next generation internet …

What information can a machine see…

🕈🕈🕈📄📁📁📄

❄♒♏ ♏●♏❖♏■♦♒ ♓■♦♏❒■♋♦♓□■♋● ⬥□❒●♎ ⬥♓♎♏ ⬥♏♌ ♍□■♐♏❒♏■♍♏

💧♒♏❒♋♦□■ ⬥♋♓🙵♓🙵♓ ♒□♦♏●

☟□■□●◆●◆📪 ♒♋⬥♋♓♓📪 🕆💧✌

🖮📫📂📂 ❍♋⍓ 📄📁📁📄

📂 ●□♍♋♦♓□■ 🖳 ♎♋⍓⬧ ●♏♋❒■ ♓■♦♏❒♋♍♦

☼♏♑♓⬧♦♏❒♏♎ ◻♋❒♦♓♍♓◻♋■♦⬧ ♍□❍♓■♑ ♐❒□❍
♋◆⬧♦❒♋●♓♋📪 ♍♋■♋♎♋📪 ♍♒♓●♏ ♎♏■❍♋❒🙵📪 ♐❒♋■♍♏📪 ♑♏❒❍♋■⍓📪 ♑♒♋■♋ ♒□■♑ 🙵□■♑📪 ♓■♎♓♋📪 ♓❒♏●♋■♎📪 ♓♦♋●⍓📪 🙰♋◻♋■📪 ❍♋●♦♋📪 ■♏⬥ ⌘♏♋●♋■♎📪 ♦♒♏ ■♏♦♒♏❒●♋■♎⬧📪 ■□❒⬥♋⍓📪 ⬧♓■♑♋◻□❒♏📪 ⬧⬥♓♦⌘♏❒●♋■♎📪 ♦♒♏ ◆■♓♦♏♎ 🙵♓■♑♎□❍📪 ♦♒♏ ◆■♓♦♏♎ ⬧♦♋♦♏⬧📪 ❖♓♏♦■♋❍📪 ⌘♋♓❒♏

☼♏♑♓⬧♦♏❒ ■□⬥

⚐■ ♦♒♏ 🖮♦♒ 💣♋⍓ ☟□■□●◆●◆ ⬥♓●● ◻❒□❖♓♎♏ ♦♒♏ ♌♋♍🙵♎❒□◻ □♐ ♦♒♏ ♏●♏❖♏■♦♒ ♓■♦♏❒■♋♦♓□■♋● ⬥□❒●♎ ⬥♓♎♏ ⬥♏♌ ♍□■♐♏❒♏■♍♏📬 ❄♒♓⬧ ◻❒♏⬧♦♓♑♓□◆⬧ ♏❖♏■♦ ⑤

💧◻♏♋🙵♏❒⬧ ♍□■♐♓❒❍♏♎

❄♓❍ ♌♏❒■♏❒⬧📫●♏♏

❄♓❍ ♓⬧ ♦♒♏ ⬥♏●● 🙵■□⬥■ ♓■❖♏■♦□❒ □♐ ♦♒♏ 🕈♏♌📪 ⑤

✋♋■ ☞□⬧♦♏❒

✋♋■ ♓⬧ ♦♒♏ ◻♓□■♏♏❒ □♐ ♦♒♏ ☝❒♓♎📪 ♦♒♏ ■♏⌧♦ ♑♏■♏❒♋♦♓□■ ♓■♦♏❒■♏♦ ⑤

Solution: XML markup with “meaningful” tags?

<name>🕈🕈🕈📄📁📁📄
❄♒♏ ♏●♏❖♏■⧫♒ ♓■⧫♏❒■♋⧫♓□■♋● ⬥□❒●♎ ⬥♓♎♏ ⬥♏♌♍□■</name>

<location>💧♒♏❒♋⧫□■ ⬥♋♓🙵♓🙵♓ ♒□⧫♏●
☟□■□●◆●◆📪 ☟♋⬥♋♓♓📪 🕆💧✌</location>

<date>🖮📫📂📂 ❍♋⍓ 📄📁📁📄</date>

<slogan>📂 ●□♍♋⧫♓□■ 🗄 ♎♋⍓⬧ ●♏♋❒■ ♓■⧫♏❒♋♍⧫</slogan>

<participants>☼♏♑♓⬧⧫♏❒♏♎ ◻♋❒⧫♓♍♓◻♋■⧫⬧ ♍□❍♓■♑ ♐❒□❍
♋◆⬧⧫❒♋●♓♋📪 ♍♋■♋♎♋📪 ♍♒♓●♏ ♎♏■❍♋❒🙵📪 ♐❒♋■♍♏📪
♑♏❒❍♋■⍓📪 ♑♒♋■♋📪 ♒□■♑ 🙵□■♑📪 ♓■♎♓♋📪 ♓❒♏●♋■♎📪
♓⧫♋●⍓📪 🙰♋◻♋■📪 ❍♋●⧫♋📪 ■♏⬥ ⌘♏♋●♋■♎📪 ⧫♒♏
■♏⧫♒♏❒●♋■♎⬧📪 ■□❒⬥♋⍓📪 ⬧♓■♑♋◻□❒♏📪 ⬧⬥♓⧫⌘♏❒●♋■♎📪
⧫♒♏ ◆■♓⧫♏♎ 🙵♓■♑♎□❍📪 ⧫♒♏ ◆■♓⧫♏♎ ⬧⧫♋⧫♏⬧📪
❖♓♏⧫■♋❍📪 ⌘♋♓❒♏</participants>

<introduction>☼♏♑♓⬧⧫♏❒ ■□⬥
⚐■ ⧫♒♏ 🖮⧫♒ 💣♋⍓ ☟□■□●◆●◆ ⬥♓●● ◻❒□❖♓♎♏ ⧫♒♏
♌♋♍🙵♎❒□◻ □♐ ⧫♒♏ ♏●♏❖♏■⧫♒ ♓■⧫♏❒■♋⧫♓□■♋● ⬥□❒●♎
⬥♓♎♏ ⬥♏♌ ♍□■♐♏❒♏■♍♏📬 ❄♒♓⬧ ◻❒♏⬧⧫♓♑♓□◆⬧ ♏❖♏■⧫ ⑤
⬧◻♏♋🙵♏❒⬧ ♍□■♐♓❒❍♏♎</introduction>

<speaker>❄♓❍ ♌♏❒■♏❒⬧📫●♏♏</speaker>

<bio>❄♓❍ ♓⬧ ⧫♒♏ ⬥♏●● 🙵■□⬥■ ♓■❖♏■⧫□❒ □♐ ⧫♒♏
🕈♏♌📬</bio>…

But What About…

<conf>🕈🕈🕈📄📁📁📄 ❄♒♏ ♏●♏❖♏■⧫♒ ♓■⧫♏❒■♋⧫♓□■♋● ⬥□❒●♎ ⬥♓♎♏ ⬥♏♌♍□■</conf>

<place>💧♒♏❒♋⧫□■ ⬥♋♓🙵♓🙵♓ ♒□⧫♏● ☟□■□●◆●◆📪 ♒♋⬥♋♓♓📪 🕆💧✌</place>

<date>🖮📫📂📂 ❍♋⍓ 📄📁📁📄</date>

<slogan>📂 ●□♍♋⧫♓□■ 🗐 ♎♋⍓⬧ ●♏♋❒■ ♓■⧫♏❒♋♍⧫</slogan>

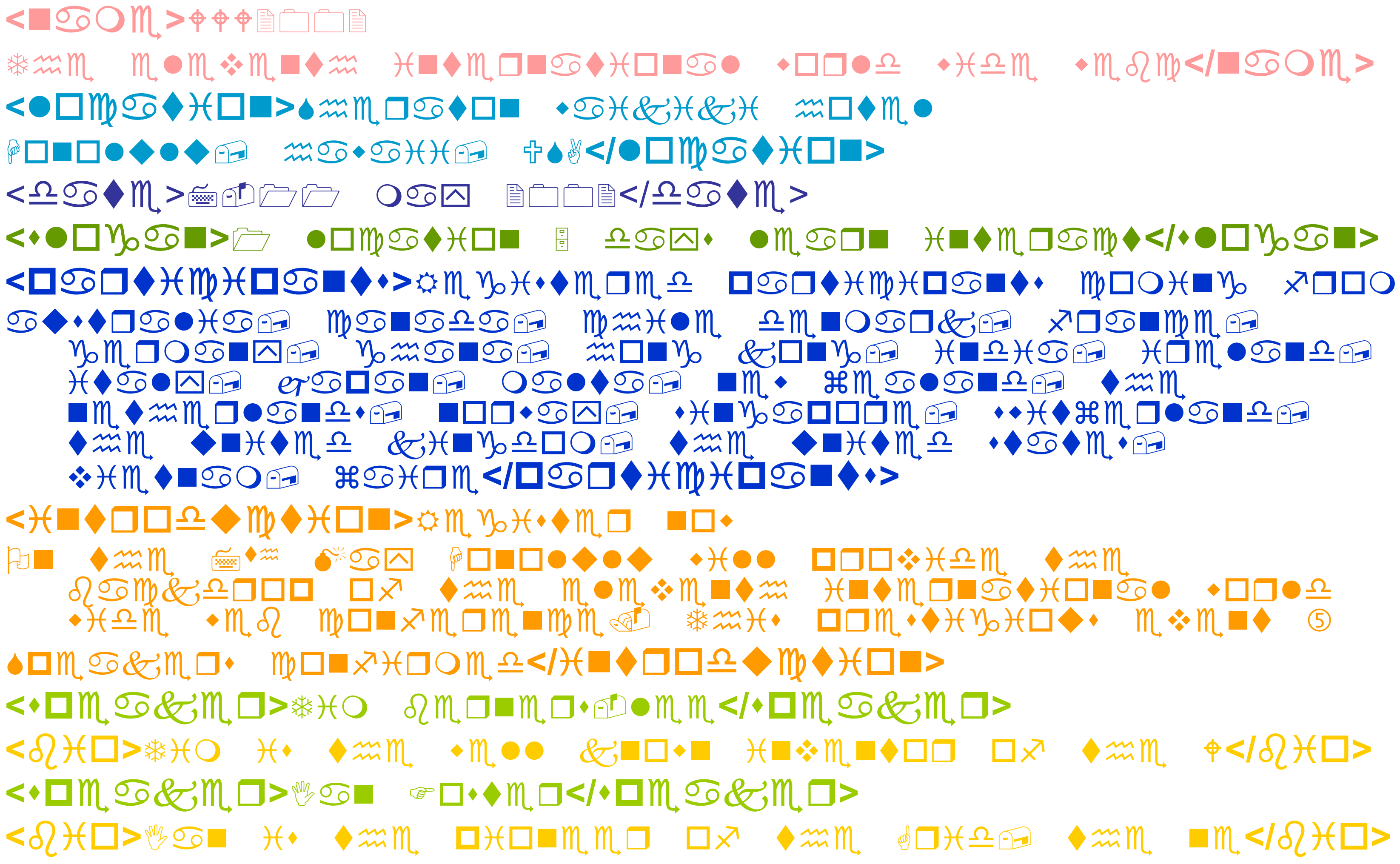
<participants>☼♏♑♓⬧⧫♏❒♏♎ □♋❒⧫♓♍♓□♋■⧫⬧ ♍□❍♓■♑ ♐❒□❍ ♋◆⬧⧫❒♋●♓♋📪 ♍♋■♋♎♋📪 ♍♒♓●♏ ♎♏■❍♋❒🙵📪 ♐❒♋■♍♏📪 ♑♏❒❍♋■⍓📪 ♑♒♋■♋ ♒□■♑ 🙵□■♑📪 ♓■♎♓♋📪 ♓❒♏●♋■♎📪 ♓⧫♋●⍓📪 🙰♋□♋■📪 ❍♋●⧫♋📪 ■♏⬧ ⌘♏♋●♋■♎📪 ⧫♒♏ ■♏⧫♒♏❒●♋■♎⬧📪 ■□❒⬥♋⍓📪 ⬧♓■♑♋□□❒♏📪 ⬧⬥♓⧫⌘♏❒●♋■♎📪 ⧫♒♏ ◆■♓⧫♏♎ 🙵♓■♑♎□❍📪 ⧫♒♏ ◆■♓⧫♏♎ ⬧⧫♋⧫♏⬧📪 ❖♓♏⧫■♋❍📪 ⌘♋♓❒♏</participants>

<introduction>☼♏♑♓⬧⧫♏❒ ■□⬥ ⚐■ ⧫♒♏ 🖮⧫♒ 💣♋⍓ ☟□■□●◆●◆ ⬥♓●● □❒□❖♓♎♏ ⧫♒♏ ♌♋♍🙵♎❒□□ □♐ ⧫♒♏ ♏●♏❖♏■⧫♒ ♓■⧫♏❒■♋⧫♓□■♋● ⬥□❒●♎ ⬥♓♎♏ ⬥♏♌ ♍□■♐♏❒♏■♍♏📬 ❄♒♓⬧ □❒♏⬧⧫♓♑♓□◆⬧ ♏❖♏■⧫ ⑤ ⬧□♏♋🙵♏❒⬧ ♍□■♐♓❒❍♏♎</introduction>

<speaker>❄♓❍ ♌♏❒■♏❒⬧📫●♏♏</speaker>

<bio>❄♓❍ ♓⬧ ⧫♒♏ ⬥♏●● 🙵■□⬥■ ♓■❖♏■⧫□❒ □♐ ⧫♒♏ ⬥♏♌📪…

Machine sees…



Need to Add “Semantics”

- External agreement on meaning of annotations
  - E.g., Dublin Core
    - Agree on the meaning of a set of annotation tags
  - Problems with this approach
    - Inflexible
    - Limited number of things can be expressed
- Use Ontologies to specify meaning of annotations
  - Ontologies provide a vocabulary of terms
  - New terms can be formed by combining existing ones
  - Meaning (semantics) of such terms is formally specified
  - Can also specify relationships between terms in multiple ontologies

Ontology: Origins and History

Ontology in Philosophy

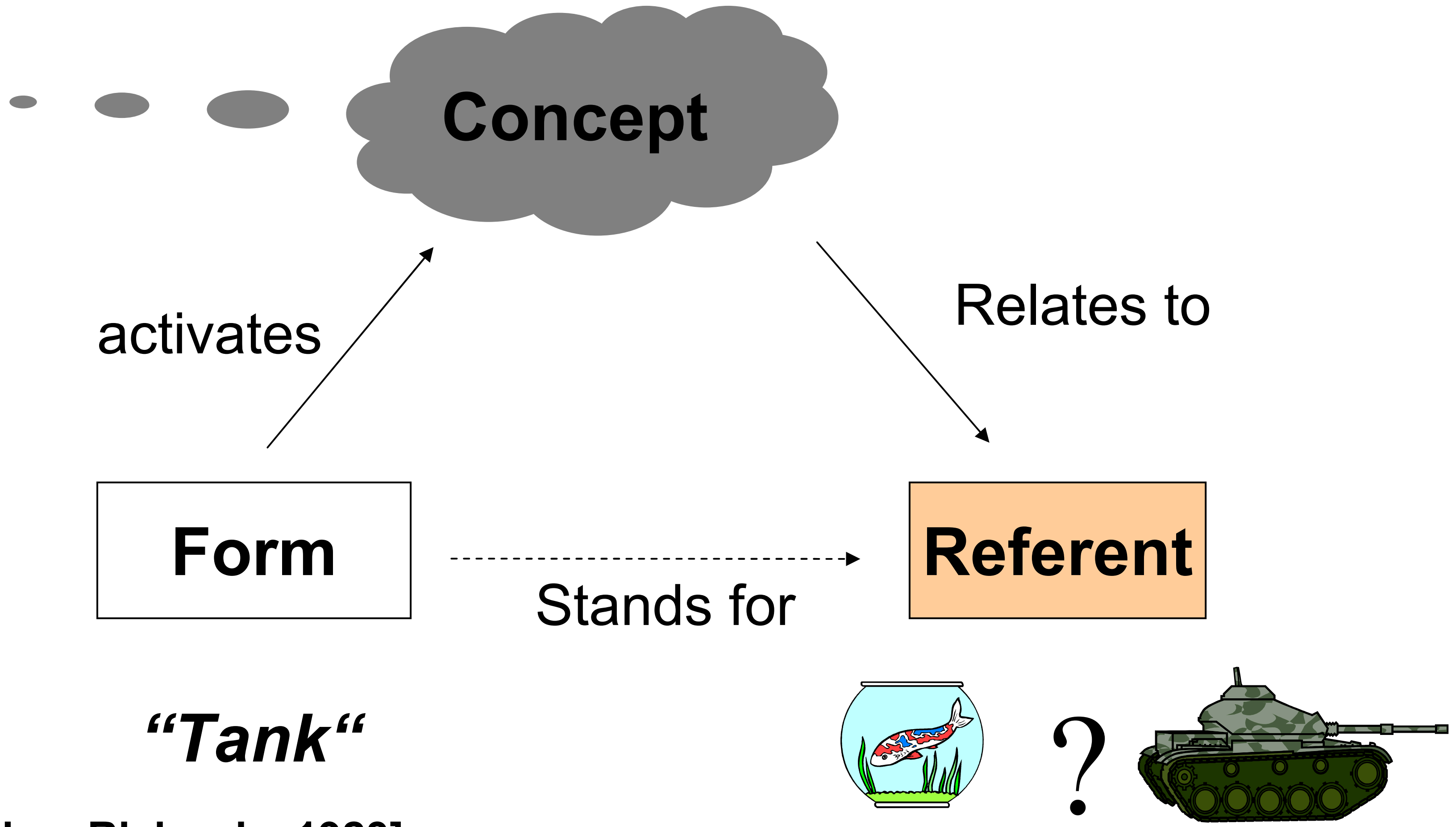
a philosophical discipline—a branch of philosophy that deals with the nature and the organisation of reality

- **Science of Being (Aristotle, Metaphysics, IV, 1)**
- **Tries to answer the questions:**

  What characterizes being?

  Eventually, what is being?

Ontology in Linguistics



[Ogden, Richards, 1923]

Ontology in Computer Science

- An ontology is an engineering artifact:
  - It is constituted by a specific vocabulary used to describe a certain reality, plus
  - a set of explicit assumptions regarding the intended meaning of the vocabulary.

- Thus, an ontology describes a formal specification of a certain domain:
  - Shared understanding of a domain of interest
  - Formal and machine manipulable model of a domain of interest

"An explicit specification of a conceptualisation" [Gruber93]

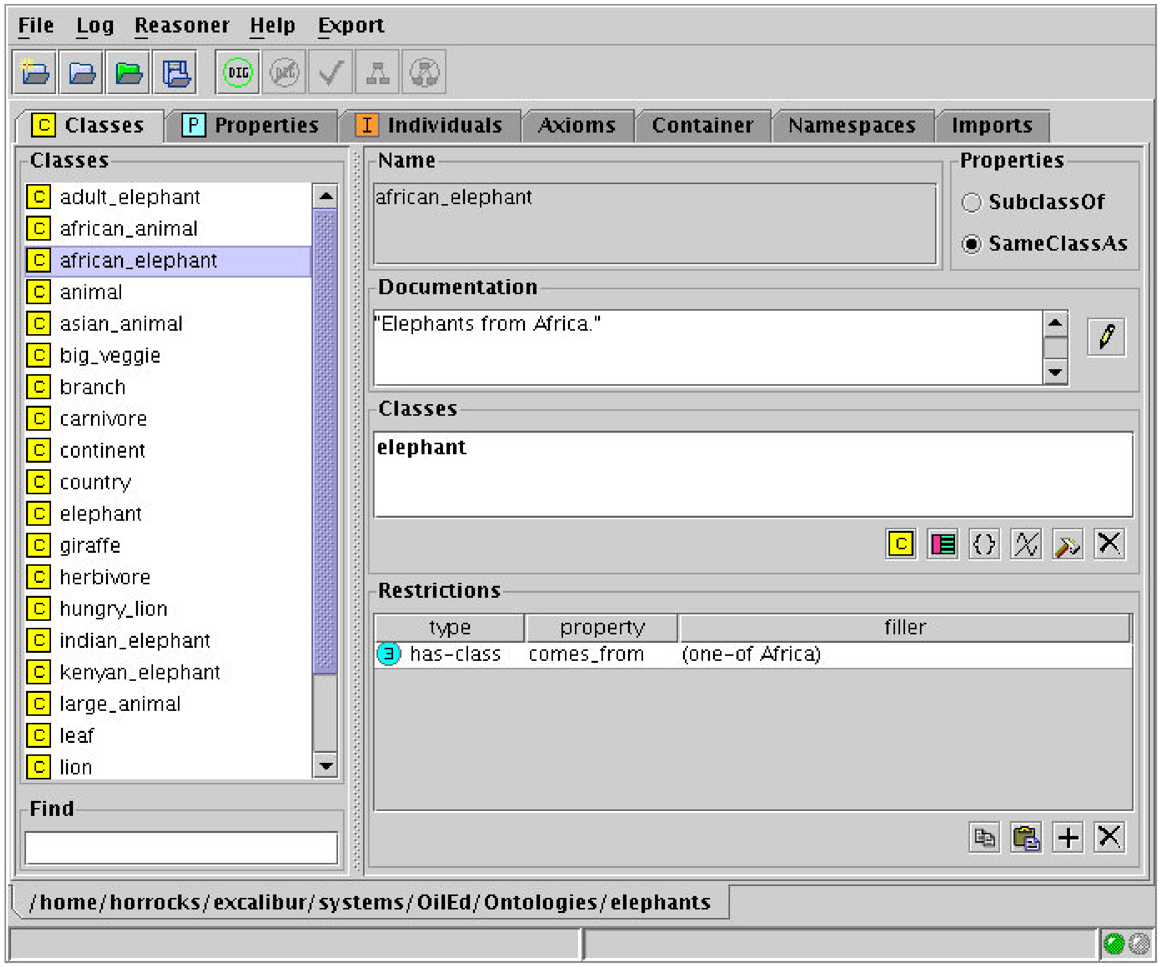
Structure of an Ontology

Ontologies typically have two distinct components:

- **Names for important concepts in the domain**
  - **Elephant is a concept whose members are a kind of animal**
  - **Herbivore is a concept whose members are exactly those animals who eat only plants or parts of plants**
  - **Adult_Elephant is a concept whose members are exactly those elephants whose age is greater than 20 years**

- **Background knowledge/constraints on the domain**
  - **Adult_Elephants weigh at least 2,000 kg**
  - **All Elephants are either African_Elephants or Indian_Elephants**
  - **No individual can be both a Herbivore and a Carnivore**

Example Ontology



Classes

- adult_elephant
- african_animal
- african_elephant
- animal
- asian_animal
- big_veggie
- branch
- carnivore
- continent
- country
- elephant
- giraffe
- herbivore
- hungry_lion
- indian_elephant
- kenyan_elephant
- large_animal
- leaf
- lion

Name: african_elephant

Properties: ( ) SubclassOf [✓ SameClassAs]

Documentation: "Elephants from Africa."

Classes: **elephant**

Restrictions

type	property	filler
∃ has-class	comes_from	(one-of Africa)

/home/horrocks/excalibur/systems/OilEd/Ontologies/elephants

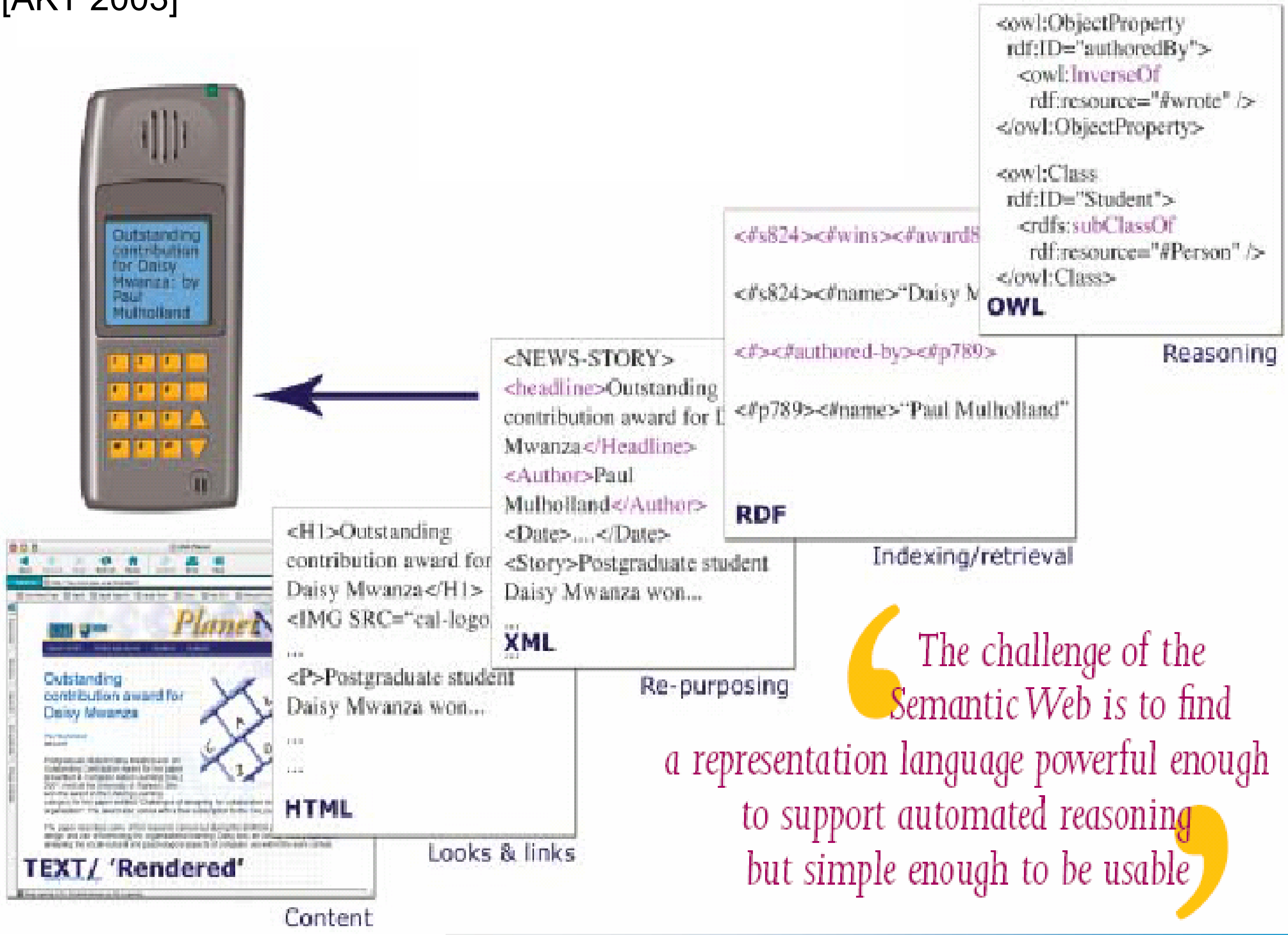
A Semantic Web — First Steps

Make web resources more accessible to automated processes

- **Extend existing rendering markup with semantic markup**
  - **Metadata annotations that describe content/funtion of web accessible resources**
- **Use Ontologies to provide vocabulary for annotations**
  - **“Formal specification” is accessible to machines**

- **A prerequisite is a standard web ontology language**
  - **Need to agree common syntax before we can share semantics**
  - **Syntactic web based on standards such as HTTP and HTML**

[AKT 2003]



Outstanding contribution for Daisy Mwanza: by Paul Mulholland
Outstanding contribution award for Daisy Mwanza
TEXT/ 'Rendered'
Content
HTML
Looks & links
XML
Re-purposing
RDF
Indexing/retrieval
OWL
Reasoning

```
<H1>Outstanding
contribution award for
Daisy Mwanza</H1>
<IMG SRC="cal-logo
...
<P>Postgraduate student
Daisy Mwanza won...
...
...
```

```
<NEWS-STORY>
<headline>Outstanding
contribution award for D
Mwanza</Headline>
<Author>Paul
Mulholland</Author>
<Date>....</Date>
<Story>Postgraduate student
Daisy Mwanza won...
...
```

```
<#s824><#wins><#award8

<#s824><#name>"Daisy M

<#><#authored-by><#p789>

<#p789><#name>"Paul Mulholland"
```

```
<owl:ObjectProperty
  rdf:ID="authoredBy">
    <owl:InverseOf
      rdf:resource="#wrote" />
</owl:ObjectProperty>

<owl:Class
  rdf:ID="Student">
    <rdfs:subClassOf
      rdf:resource="#Person" />
</owl:Class>
```

"The challenge of the Semantic Web is to find a representation language powerful enough to support automated reasoning but simple enough to be usable"

Ontology Design and Deployment

- Given key role of ontologies in the Semantic Web, it will be essential to provide tools and services to help users:
  - Design and maintain high quality ontologies, e.g.:
    - Meaningful — all named classes can have instances
    - Correct — captured intuitions of domain experts
    - Minimally redundant — no unintended synonyms
    - Richly axiomatised — (sufficiently) detailed descriptions
  - Store (large numbers) of instances of ontology classes, e.g.:
    - Annotations from web pages
  - Answer queries over ontology classes and instances, e.g.:
    - Find more general/specific classes
    - Retrieve annotations/pages matching a given description
  - Integrate and align multiple ontologies

Ontology Languages for the Semantic Web

Resources

- **Course material (including slides):**

  http://www.cs.man.ac.uk/~horrocks/ESSLLI2003/

- **Description Logic Handbook**

  http://books.cambridge.org/0521781760.htm

Ontology Languages

- Wide variety of languages for "Explicit Specification"
  - Graphical notations
    - Semantic networks
    - Topic Maps (see http://www.topicmaps.org/)
    - UML
    - RDF
  - Logic based
    - Description Logics (e.g., OIL, DAML+OIL, OWL)
    - Rules (e.g., RuleML, LP/Prolog)
    - First Order Logic (e.g., KIF)
    - Conceptual graphs
    - (Syntactically) higher order logics (e.g., LBase)
    - Non-classical logics (e.g., Flogic, Non-Mon, modalities)
  - Probabilistic/fuzzy
- Degree of formality varies widely
  - Increased formality makes languages more amenable to machine processing (e.g., automated reasoning)

Many languages use “object oriented” model based on:

- Objects/Instances/Individuals
  - Elements of the domain of discourse
  - Equivalent to constants in FOL
- Types/Classes/Concepts
  - Sets of objects sharing certain characteristics
  - Equivalent to unary predicates in FOL
- Relations/Properties/Roles
  - Sets of pairs (tuples) of objects
  - Equivalent to binary predicates in FOL

- Such languages are/can be:
  - Well understood
  - Formally specified
  - (Relatively) easy to use
  - Amenable to machine processing

Web "Schema" Languages

- **Existing Web languages extended to facilitate content description**
  - **XML → XML Schema (XMLS)**
  - **RDF → RDF Schema (RDFS)**
- **XMLS *not* an ontology language**
  - **Changes format of DTDs (document schemas) to be XML**
  - **Adds an extensible type hierarchy**
    - **Integers, Strings, etc.**
    - **Can define sub-types, e.g., positive integers**
- **RDFS *is* recognisable as an ontology language**
  - **Classes and properties**
  - **Sub/super-classes (and properties)**
  - **Range and domain (of properties)**

RDF and RDFS

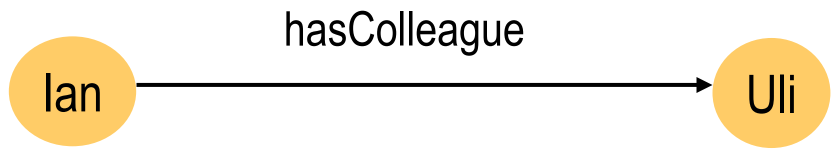
- RDF stands for Resource Description Framework
- It is a W3C candidate recommendation (http://www.w3.org/RDF)
- RDF is graphical formalism (+ XML syntax + semantics)
  - for representing metadata
  - for describing the semantics of information in a machine-accessible way
- RDFS extends RDF with “schema vocabulary”, e.g.:
  - Class, Property
  - type, subClassOf, subPropertyOf
  - range, domain

The RDF Data Model

- Statements are <subject, predicate, object> triples:

  `<Ian,hasColleague,Uli>`

- Can be represented as a graph:



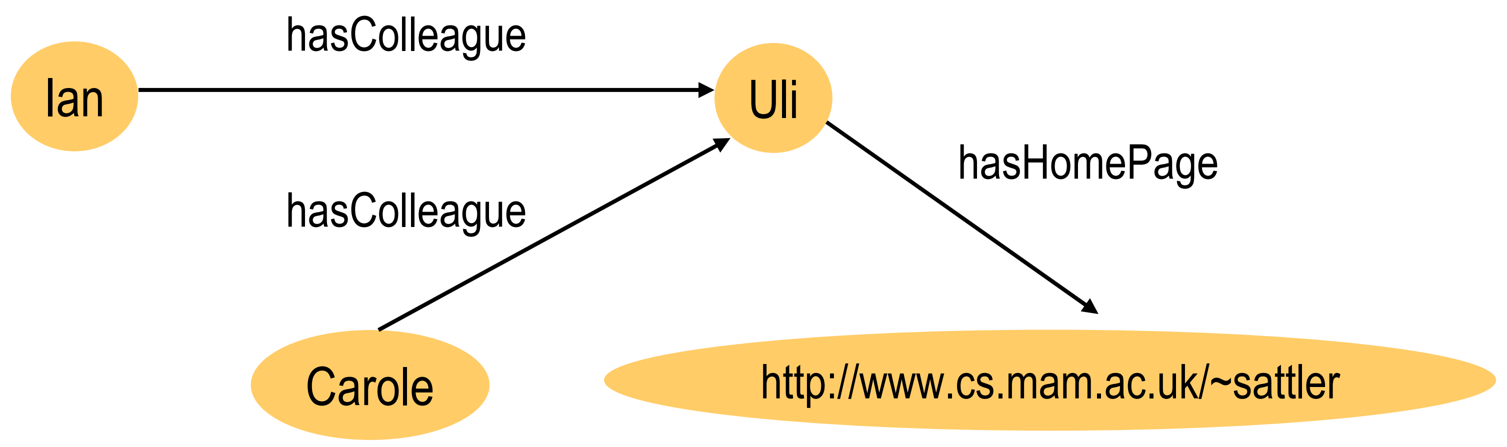
- Statements describe properties of resources
- A resource is any object that can be pointed to by a URI:
  - a document, a picture, a paragraph on the Web;
  - http://www.cs.man.ac.uk/index.html
  - a book in the library, a real person (?)
  - isbn://5031-4444-3333
  - …
- Properties themselves are also resources (URIs)

URIs

- URI = Uniform Resource Identifier
- "The generic set of all names/addresses that are short strings that refer to resources"
- URLs (Uniform Resource Locators) are a particular type of URI, used for resources that can be accessed on the WWW (e.g., web pages)
- In RDF, URIs typically look like “normal” URLs, often with fragment identifiers to point at specific parts of a document:
  - http://www.somedomain.com/some/path/to/file#fragmentID

Linking Statements

- **The subject of one statement can be the object of another**
- **Such collections of statements form a directed, labeled graph**



- **Note that the object of a triple can also be a “literal” (a string)**

RDF Syntax

- RDF has an XML syntax that has a specific meaning:
- Every `Description` element describes a resource
- Every attribute or nested element inside a `Description` is a `property` of that Resource
- We can refer to resources by using URIs

```
<Description about="some.uri/person/ian_horrocks">
   <hasColleague resource="some.uri/person/uli_sattler"/>
</Description>
<Description about="some.uri/person/uli_sattler">
   <hasHomePage>http://www.cs.mam.ac.uk/~sattler</hasHomePage>
</Description>
<Description about="some.uri/person/carole_goble">
   <hasColleague resource="some.uri/person/uli_sattler"/>
</Description>
```

RDF Schema (RDFS)

- **RDF gives a formalism for meta data annotation, and a way to write it down in XML, but it does not give any special meaning to vocabulary such as subClassOf or type**
  - **Interpretation is an arbitrary binary relation**

- **RDF Schema allows you to define vocabulary terms and the relations between those terms**
  - **it gives "extra meaning" to particular RDF predicates and resources**
  - **this "extra meaning", or semantics, specifies how a term should be interpreted**

RDFS Examples

- RDF Schema terms (just a few examples):
  - Class
  - Property
  - type
  - subClassOf
  - range
  - domain
- These terms are the RDF Schema building blocks (constructors) used to create vocabularies:

```
<Person,type,Class>
<hasColleague,type,Property>
<Professor,subClassOf,Person>
<Carole,type,Professor>
<hasColleague,range,Person>
<hasColleague,domain,Person>
```

RDF/RDFS “Liberality”

- **No distinction between classes and instances (individuals)**

```
<Species,type,Class>
<Lion,type,Species>
<Leo,type,Lion>
```

- **Properties can themselves have properties**

```
<hasDaughter,subPropertyOf,hasChild>
<hasDaughter,type,familyProperty>
```

- **No distinction between language constructors and ontology vocabulary, so constructors can be applied to themselves/each other**

```
<type,range,Class>
<Property,type,Class>
<type,subPropertyOf,subClassOf>
```

RDF/RDFS Semantics

- **RDF has “Non-standard” semantics in order to deal with this**
- **Semantics given by RDF Model Theory (MT)**

Semantics and Model Theories

- **Ontology/KR languages aim to model (part of) world**
- **Terms in language correspond to entities in world**
- **Meaning given by, e.g.:**
  - Mapping to another formalism, such as FOL, with own well defined semantics
  - or a bespoke Model Theory (MT)
- **MT defines relationship between syntax and *interpretations***
  - Can be many interpretations (models) of one piece of syntax
  - Models supposed to be analogue of (part of) world
    - E.g., elements of model correspond to objects in world
  - Formal relationship between syntax and models
    - Structure of models reflect relationships specified in syntax
  - Inference (e.g., subsumption) defined in terms of MT
    - E.g., $\mathcal{T} \models$ A \sqsubseteq B iff in every model of $\mathcal{T}$, ext(A) \subseteq ext(B)

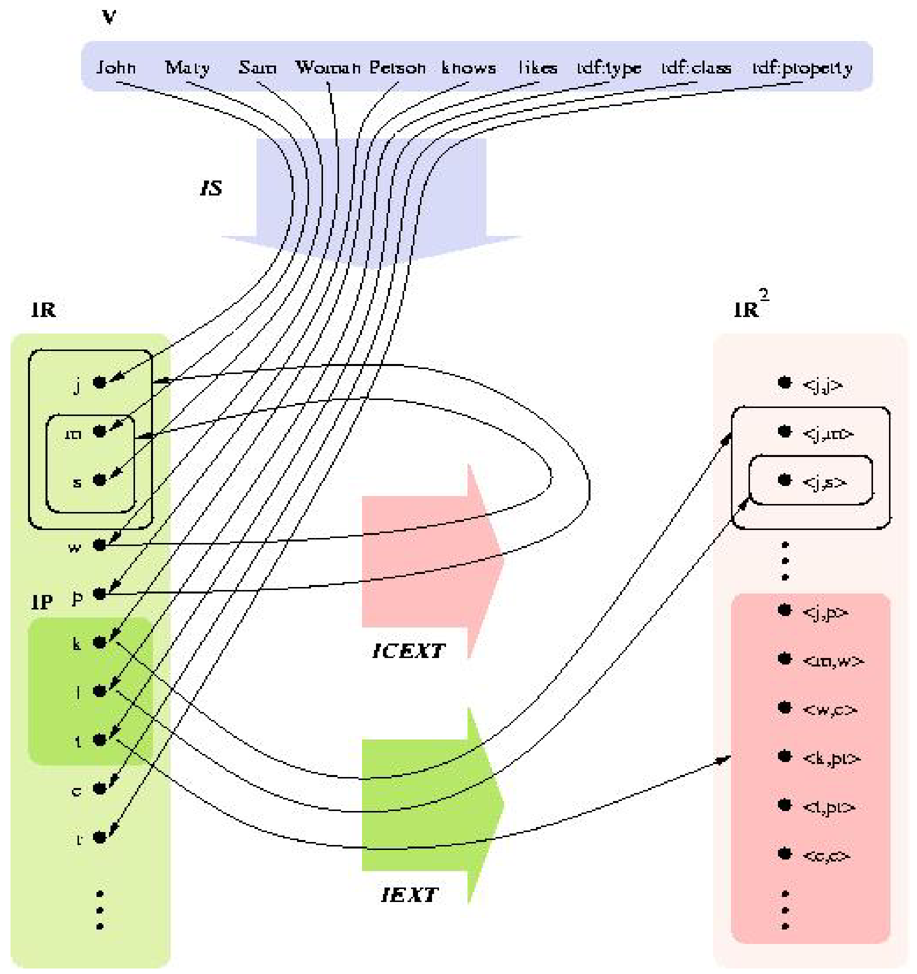
Semantics and Model Theories

- **Ontology/KR languages aim to model (part of) world**
- **Terms in language correspond to entities in world**
- **Meaning given by, e.g.:**
  - Mapping to another formalism, such as FOL, with own well defined semantics
  - or a bespoke Model Theory (MT)
- **MT defines relationship between syntax and *interpretations***
  - Can be many interpretations (models) of one piece of syntax
  - Models supposed to be analogue of (part of) world
    - E.g., elements of model correspond to objects in world
  - Formal relationship between syntax and models
    - Structure of models reflect relationships specified in syntax
  - Inference (e.g., subsumption) defined in terms of MT
    - E.g., $\mathcal{T} \models$ A \sqsubseteq B iff in every model of $\mathcal{T}$, ext(A) \subseteq ext(B)

RDF/RDFS Semantics

- RDF has “Non-standard” semantics in order to deal with this
- Semantics given by RDF Model Theory (MT)
- In RDF MT, an interpretation $\mathcal{I}$ of a vocabulary V consists of:
  - IR, a non-empty set of resources
  - IS, a mapping from V into IR
  - IP, a distinguished subset of IR (the properties)
    - A vocabulary element $v \in V$ is a property iff $IS(v) \in IP$
  - IEXT, a mapping from IP into the powerset of $IR \times IR$
    - I.e., a set of elements <x,y>, with x,y elements of IR
  - IL, a mapping from typed literals into IR
- Class interpretation ICEXT simply induced by IEXT(IS(`type`))
    - $ICEXT(C) = \{x \mid \langle x,C \rangle \in IEXT(IS(\texttt{type}))\}$

Example RDF/RDFS Interpretation



RDFS Interpretations

- **RDFS adds extra constraints on interpretations**
  - **E.g., interpretationss of `<C,subClassOf,D>` constrained to those where ICEXT(IS(C)) $\subseteq$ ICEXT(IS(D))**
- **Can deal with triples such as**
  - `<Species,type,Class>`
    `<Lion,type,Species>`
    `<Leo,type,Lion>`
  - `<SelfInst,type,SelfInst>`
- **And even with triples such as**
  - `<type,subPropertyOf,subClassOf>`
- **But not clear if meaning matches intuition (if there is one)**

Problems with RDFS

- RDFS too weak to describe resources in sufficient detail
  - No localised range and domain constraints
    - Can’t say that the range of hasChild is person when applied to persons and elephant when applied to elephants
  - No existence/cardinality constraints
    - Can’t say that all *instances* of person have a mother that is also a person, or that persons have exactly 2 parents
  - No transitive, inverse or symmetrical properties
    - Can’t say that isPartOf is a transitive property, that hasPart is the inverse of isPartOf or that touches is symmetrical
  - …
- Difficult to provide reasoning support
  - No “native” reasoners for non-standard semantics
  - May be possible to reason via FO axiomatisation

Web Ontology Language Requirements

Desirable features identified for Web Ontology Language:

- Extends existing Web standards
  - Such as XML, RDF, RDFS
- Easy to understand and use
  - Should be based on familiar KR idioms
- Formally specified
- Of “adequate” expressive power
- Possible to provide automated reasoning support

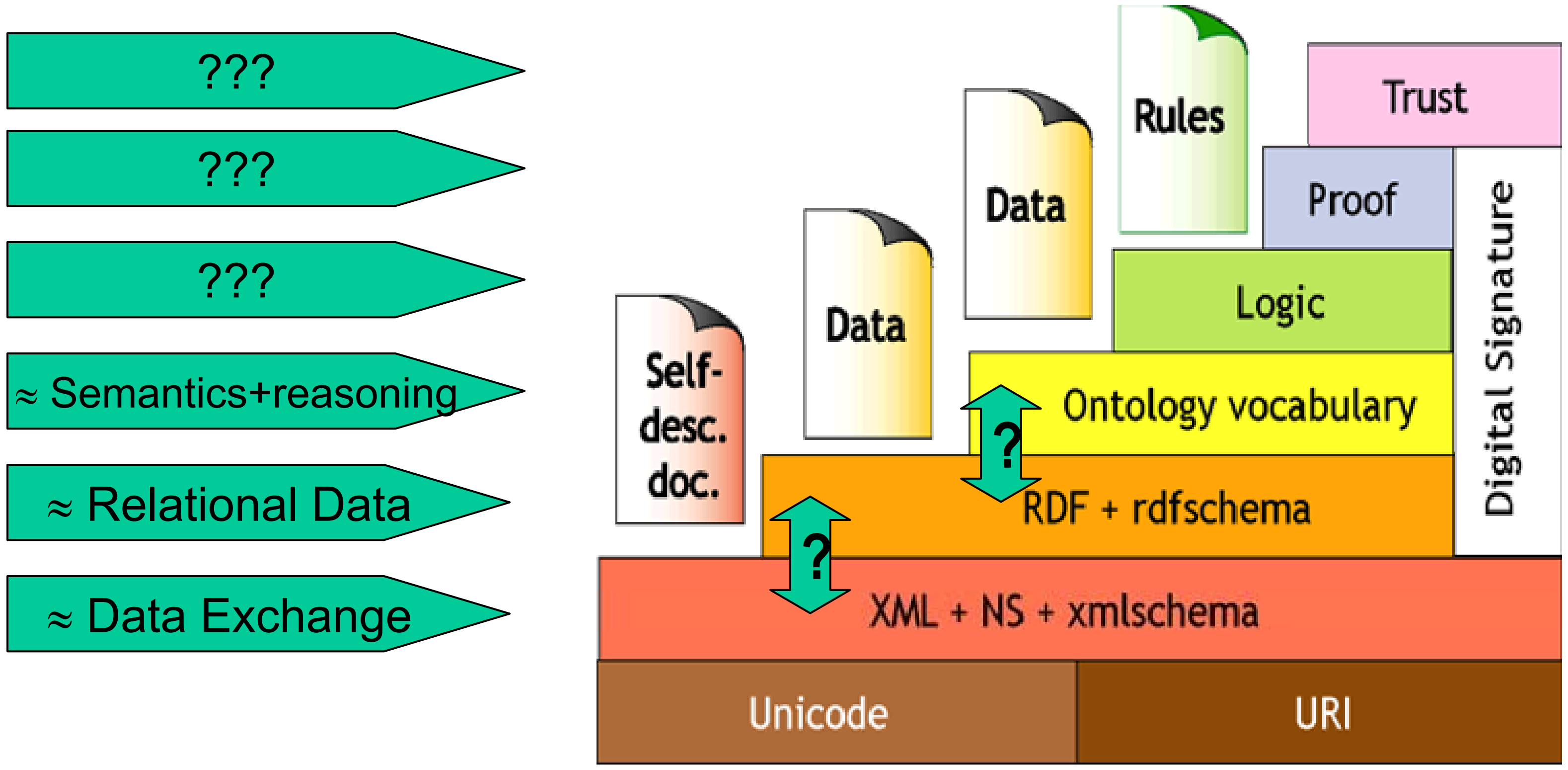
From RDF to OWL

- Two languages developed to satisfy above requirements
  - OIL: developed by group of (largely) European researchers (several from EU OntoKnowledge project)
  - DAML-ONT: developed by group of (largely) US researchers (in DARPA DAML programme)
- Efforts merged to produce DAML+OIL
  - Development was carried out by “Joint EU/US Committee on Agent Markup Languages”
  - Extends (“DL subset” of) RDF
- DAML+OIL submitted to W3C as basis for standardisation
  - Web-Ontology (WebOnt) Working Group formed
  - WebOnt group developed OWL language based on DAML+OIL
  - OWL language now a W3C Candidate Recommendation
  - Will soon become Proposed Recommendation

OWL Language

- Three species of OWL
  - OWL full is union of OWL syntax and RDF
  - OWL DL restricted to FOL fragment ($\approx$ DAML+OIL)
  - OWL Lite is “easier to implement” subset of OWL DL
- Semantic layering
  - OWL DL $\approx$ OWL full within DL fragment
  - DL semantics officially definitive
- OWL DL based on $\mathcal{SHIQ}$ Description Logic
  - In fact it is equivalent to $\mathcal{SHOIN}(\mathbf{D}_n)$ DL
- OWL DL Benefits from many years of DL research
  - Well defined semantics
  - Formal properties well understood (complexity, decidability)
  - Known reasoning algorithms
  - Implemented systems (highly optimised)

(In)famous "Layer Cake"



- **Relationship between layers is not clear**
- **OWL DL extends "DL subset" of RDF**

OWL Class Constructors

Constructor	DL Syntax	Example	Modal Syntax
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	Human ⊓ Male	$C_1 \wedge \ldots \wedge C_n$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	Doctor ⊔ Lawyer	$C_1 \vee \ldots \vee C_n$
complementOf	$\neg C$	¬Male	$\neg C$
oneOf	$\{x_1\} \sqcup \ldots \sqcup \{x_n\}$	{john} ⊔ {mary}	$x_1 \vee \ldots \vee x_n$
allValuesFrom	$\forall P.C$	∀hasChild.Doctor	$[P]C$
someValuesFrom	$\exists P.C$	∃hasChild.Lawyer	$\langle P \rangle C$
maxCardinality	$\leqslant n P$	⩽1hasChild	$[P]_{n+1}$
minCardinality	$\geqslant n P$	⩾2hasChild	$\langle P \rangle_n$

- **XMLS datatypes as well as classes in ∀P.C and ∃P.C**
  - **E.g., ∃hasAge.nonNegativeInteger**
- **Arbitrarily complex nesting of constructors**
  - **E.g., Person ⊓ ∀hasChild.Doctor ⊔ ∃hasChild.Doctor**

RDFS Syntax

E.g., Person ⊓ ∀hasChild.Doctor ⊔ ∃hasChild.Doctor:

```
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Person"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:toClass>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Doctor"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#hasChild"/>
            <owl:hasClass rdf:resource="#Doctor"/>
          </owl:Restriction>
        </owl:unionOf>
      </owl:toClass>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

OWL Axioms

Axiom	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human $\sqsubseteq$ Animal $\sqcap$ Biped
equivalentClass	$C_1 \equiv C_2$	Man $\equiv$ Human $\sqcap$ Male
disjointWith	$C_1 \sqsubseteq \neg C_2$	Male $\sqsubseteq \neg$Female
sameIndividualAs	$\{x_1\} \equiv \{x_2\}$	{President_Bush} $\equiv$ {G_W_Bush}
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	{john} $\sqsubseteq \neg${peter}
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter $\sqsubseteq$ hasChild
equivalentProperty	$P_1 \equiv P_2$	cost $\equiv$ price
inverseOf	$P_1 \equiv P_2^-$	hasChild $\equiv$ hasParent$^-$
transitiveProperty	$P^+ \sqsubseteq P$	ancestor$^+$ $\sqsubseteq$ ancestor
functionalProperty	$\top \sqsubseteq \leqslant 1 P$	$\top \sqsubseteq \leqslant$1hasMother
inverseFunctionalProperty	$\top \sqsubseteq \leqslant 1 P^-$	$\top \sqsubseteq \leqslant$1hasSSN$^-$

- **Axioms (mostly) reducible to inclusion ($\sqsubseteq$)**
  - C $\equiv$ D **iff both** C $\sqsubseteq$ D **and** D $\sqsubseteq$ C

XML Schema Datatypes in OWL

- **OWL supports XML Schema primitive datatypes**
  - **E.g., integer, real, string, …**
- **Strict separation between "object" classes and datatypes**
  - **Disjoint interpretation domain Δ_D for datatypes**
    - **For a datavalue** d, $d^{\mathcal{I}} \subseteq \Delta_D$
    - **And** $\Delta_D \cap \Delta^{\mathcal{I}} = \emptyset$
  - **Disjoint "object" and datatype properties**
    - **For a datatype propterty** P, $P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_D$
    - **For object property** S **and datatype property** P, $S^{\mathcal{I}} \cap P^{\mathcal{I}} = \emptyset$
- **Equivalent to the "**(D_n)**" in** $\mathcal{SHOIN}(D_n)$

Why Separate Classes and Datatypes?

- **Philosophical reasons:**
  - **Datatypes structured by built-in predicates**
  - **Not appropriate to form new datatypes using ontology language**
- **Practical reasons:**
  - **Ontology language remains simple and compact**
  - **Semantic integrity of ontology language not compromised**
  - **Implementability not compromised — can use hybrid reasoner**
    - **Only need sound and complete decision procedure for:**
      $d^{\mathcal{I}}_1 \cap \ldots \cap d^{\mathcal{I}}_n$, **where** d **is a (possibly negated) datatype**

OWL DL Semantics

- Mapping OWL to equivalent DL ($\mathcal{SHOIN}(D_n)$):
  - Facilitates provision of reasoning services (using DL systems)
  - Provides well defined semantics
- DL semantics defined by interpretations: $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where
  - $\Delta^{\mathcal{I}}$ is the domain (a non-empty set)
  - $\cdot^{\mathcal{I}}$ is an interpretation function that maps:
    - Concept (class) name $\mathrm{A} \rightarrow$ subset $\mathrm{A}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$
    - Role (property) name $\mathrm{R} \rightarrow$ binary relation $\mathrm{R}^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$
    - Individual name $\mathrm{i} \rightarrow \mathrm{i}^{\mathcal{I}}$ element of $\Delta^{\mathcal{I}}$

DL Semantics

- **Interpretation function $\cdot^{\mathcal{I}}$ extends to concept expressions in an obvious(ish) way, i.e.:**

$$
\begin{aligned}
&(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}\\
&(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}\\
&(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}\\
&\{x\}^{\mathcal{I}} = \{x^{\mathcal{I}}\}\\
&(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}\\
&(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.(x, y) \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}\\
&(\leqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \leqslant n\}\\
&(\geqslant n R)^{\mathcal{I}} = \{x \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \geqslant n\}
\end{aligned}
$$

DL Knowledge Bases (Ontologies)

- **An OWL ontology maps to a DL Knowledge Base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$**
  - $\mathcal{T}$ **(Tbox) is a set of axioms of the form:**
    - $C \sqsubseteq D$ **(concept inclusion)**
    - $C \equiv D$ **(concept equivalence)**
    - $R \sqsubseteq S$ **(role inclusion)**
    - $R \equiv S$ **(role equivalence)**
    - $R^+ \sqsubseteq R$ **(role transitivity)**
  - $\mathcal{A}$ **(Abox) is a set of axioms of the form**
    - $x \in D$ **(concept instantiation)**
    - $\langle x,y \rangle \in R$ **(role instantiation)**
- **Two sorts of Tbox axioms often distinguished**
  - **"Definitions"**
    - $C \sqsubseteq D$ **or** $C \equiv D$ **where** C **is a concept name**
  - **General Concept Inclusion axioms (GCIs)**
    - $C \sqsubseteq D$ **where** C **in an arbitrary concept**

Knowledge Base Semantics

- **An interpretation $\mathcal{I}$ satisfies (models) an axiom A ($\mathcal{I} \vDash \mathrm{A}$):**
  - $\mathcal{I} \vDash \mathrm{C} \sqsubseteq \mathrm{D}$ **iff** $\mathrm{C}^{\mathcal{I}} \subseteq \mathrm{D}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \mathrm{C} \equiv \mathrm{D}$ **iff** $\mathrm{C}^{\mathcal{I}} = \mathrm{D}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \mathrm{R} \sqsubseteq \mathrm{S}$ **iff** $\mathrm{R}^{\mathcal{I}} \subseteq \mathrm{S}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \mathrm{R} \equiv \mathrm{S}$ **iff** $\mathrm{R}^{\mathcal{I}} = \mathrm{S}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \mathrm{R}^{+} \sqsubseteq \mathrm{R}$ **iff** $(\mathrm{R}^{\mathcal{I}})^{+} \subseteq \mathrm{R}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \mathrm{x} \in \mathrm{D}$ **iff** $\mathrm{x}^{\mathcal{I}} \in \mathrm{D}^{\mathcal{I}}$
  - $\mathcal{I} \vDash \langle \mathrm{x},\mathrm{y} \rangle \in \mathrm{R}$ **iff** $(\mathrm{x}^{\mathcal{I}},\mathrm{y}^{\mathcal{I}}) \in \mathrm{R}^{\mathcal{I}}$
- **$\mathcal{I}$ satisfies a Tbox $\mathcal{T}$ ($\mathcal{I} \vDash \mathcal{T}$) iff $\mathcal{I}$ satisfies every axiom A in $\mathcal{T}$**
- **$\mathcal{I}$ satisfies an Abox $\mathcal{A}$ ($\mathcal{I} \vDash \mathcal{A}$) iff $\mathcal{I}$ satisfies every axiom A in $\mathcal{A}$**
- **$\mathcal{I}$ satisfies an KB $\mathcal{K}$ ($\mathcal{I} \vDash \mathcal{K}$) iff $\mathcal{I}$ satisfies both $\mathcal{T}$ and $\mathcal{A}$**

Inference Tasks

- **Knowledge is correct (captures intuitions)**
  - **C subsumes D w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$**
- **Knowledge is minimally redundant (no unintended synonyms)**
  - **C is equivallent to D w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $C^{\mathcal{I}} = D^{\mathcal{I}}$**
- **Knowledge is meaningful (classes can have instances)**
  - **C is satisfiable w.r.t. $\mathcal{K}$ iff there exists *some* model $\mathcal{I}$ of $\mathcal{K}$ s.t. $C^{\mathcal{I}} \neq \emptyset$**

- **Querying knowledge**
  - **x is an instance of C w.r.t. $\mathcal{K}$ iff for *every* model $\mathcal{I}$ of $\mathcal{K}$, $x^{\mathcal{I}} \in C^{\mathcal{I}}$**
  - **$\langle x,y \rangle$ is an instance of R w.r.t. $\mathcal{K}$ iff for, *every* model $\mathcal{I}$ of $\mathcal{K}$, $(x^{\mathcal{I}},y^{\mathcal{I}}) \in R^{\mathcal{I}}$**

- **Knowledge base consistency**
  - **A KB $\mathcal{K}$ is consistent iff there exists *some* model $\mathcal{I}$ of $\mathcal{K}$**

Logical Foundations for the Semantic Web

3. Reasoning Services and Algorithms

Help knowledge engineer and users to build and use ontologies

Ian Horrock and Ulrike Sattler
University of Manchester
Manchester, UK
{horrocks|sattler}@cs.man.ac.uk

Plan for today

1. "useful" reasoning services
2. relationship between DLs and other logics (briefly)
3. system demonstration
4. tableau algorithm for $\mathcal{ALC}$ and how to prove its correctness
5. how to extend this algorithm to DAML+OIL and OWL

Remember: Complexity of Ontology engineering

Remember ontology engineering **tasks**:

- design
- evolution
- inter-operation and Integration
- deployment

Further complications are due to

- sheer size of ontologies
- number of persons involved
- users not being knowledge experts
- natural laziness
- etc.

Reasoning Services: what we might want in the Design Phase

- be warned when making **meaningless** statements

  ➠ test **satisfiability** of defined concepts

  $$\textbf{SAT}(C, \mathcal{T}) \text{ iff there is a model } \mathcal{I} \text{ of } \mathcal{T} \text{ with } C^{\mathcal{I}} \neq \emptyset$$

  unsatisfiable, defined concepts are signs of faulty modelling

- see **consequences** of statements made

  ➠ test defined concepts for **subsumption**

  $$\textbf{SUBS}(C, D, \mathcal{T}) \text{ iff } C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \text{ for all model } \mathcal{I} \text{ of } \mathcal{T}$$

  unwanted or missing subsumptions are signs of imprecise/faulty modelling

- see **redundancies**

  ➠ test defined concepts for **equivalence**

  $$\textbf{SUBS}(C, D, \mathcal{T}) \text{ iff } C^{\mathcal{I}} = D^{\mathcal{I}} \text{ for all model } \mathcal{I} \text{ of } \mathcal{T}$$

  knowing about “redundant” classes helps avoid misunderstandings

Reasoning Services: what we might want when Modifying Ontologies

- the same system services as in the design phase, plus
- automatic generation of **concept definitions from examples**
  - ➠ given individuals $o_1, \ldots, o_n$ with assertions ("ABox") for them, create a (most specific) concept C such that each $o_i \in C^{\mathcal{I}}$ in each model $\mathcal{I}$ of $\mathcal{T}$

    "non-standard inferences"

- automatic generation of concept definitions for **too many siblings**
  - ➠ given concepts $C_1, \ldots, C_n$, create a (most specific) concept C such that **SUBS**$(C_i, C, \mathcal{T})$

    "non-standard inferences"

- etc.

Reasoning Services: what we might want when Integrating and Using Ontologies

For integration:

- the same system services as in the design phase, plus
- the possibility to abstract from concepts to **patterns** and compare patterns
  - ➠ e.g., compute those concepts D defined in $\mathcal{T}_2$ such that

$$\textbf{SUBS}(\texttt{Human} \sqcap (\forall \texttt{child}.(X \sqcap \forall \texttt{child}.Y)), D, \mathcal{T}_1 \cup \mathcal{T}_2)$$

“non-standard inferences”

When using ontologies:

- the same system services as in the design phase and the integration phase, plus
- automatic classification of indidivuals
  - ➠ given individual o with assertions, return all defined concepts D such that

$$o \in D^{\mathcal{I}} \text{ for all models } \mathcal{I} \text{ of } \mathcal{T}$$

Reasoning Services: what we can do

(many) reasoning problems are **inter-reducible**:

$$\textbf{EQUIV}(C, D, \mathcal{T}) \text{ iff } \textbf{sub}(C, D, \mathcal{T}) \text{ and } \textbf{sub}(D, C, \mathcal{T})$$

$$\textbf{SUBS}(C, D, \mathcal{T}) \text{ iff } \textbf{not SAT}(C \sqcap \neg D, \mathcal{T})$$

$$\textbf{SAT}(C, \mathcal{T}) \text{ iff } \textbf{not SUBS}(C, A \sqcap \neg A, \mathcal{T})$$

$$\textbf{SAT}(C, \mathcal{T}) \text{ iff } \textbf{cons}(\{o\colon C\}, \mathcal{T})$$

➠ In the following, we concentrate on **SAT**$(C, \mathcal{T})$

Do Reasoning Services need to be Decidable?

We know **SAT** is reducible to co-**SUBS** and vice versa

Hence **SAT** is undecidable iff **SUBS** is

SAT is semi-decidable iff **co-SUBS** is

➠ if **SAT** is **undecidable but semi-decidable**, then

there exists a **complete SAT** algorithm:
SAT$(C, \mathcal{T}) \Leftrightarrow$ "satisfiable", but might not terminate if not **SAT**$(C, \mathcal{T})$

there is a complete co-**SUBS** algorithm:
SUBS$(C, \mathcal{T}) \Leftrightarrow$ "subsumption", but might not terminate if **SUBS**$(C, D, \mathcal{T})$)

1. Do **expressive** ontology languages exist with **decidable** reasoning problems?

2. Is there a practical difference between ExpTime-hard and non-terminating?

Do Reasoning Services need to be Decidable?

We know **SAT** is reducible to co-**SUBS** and vice versa

Hence **SAT** is undecidable iff **SUBS** is

SAT is semi-decidable iff **co-SUBS** is

➠ if **SAT** is **undecidable but semi-decidable**, then

there exists a **complete SAT** algorithm:
SAT$(C, \mathcal{T})$ ⇔ "satisfiable", but might not terminate if not **SAT**$(C, \mathcal{T})$

there is a complete co-**SUBS** algorithm:
SUBS$(C, \mathcal{T})$ ⇔ "subsumption", but might not terminate if **SUBS**$(C, D, \mathcal{T})$)

1. Do **expressive** ontology languages exist with **decidable** reasoning problems?
   Yes: DAML+OIL and OWL
2. Is there a practical difference between ExpTime-hard and non-terminating?
   let's see

Relationship with other Logics

(slide with translation)

- $\mathcal{SHI}$ is a fragment of **first order logic**
- $\mathcal{SHIQ}$ is a fragment of **first order logic with counting quantifiers**
  equality
- $\mathcal{SHI}$ without transitivity is a fragment of first order with **two variables**
- $\mathcal{ALC}$ is a notational variant of the **multi modal logic** K

  inverse roles are closely related to converse/past modalities

  transitive roles are closely related to transitive frames/axiom 4

  number restrictions are closely related to deterministic programs in PDL

system demonstration

Deciding Satisfiability of $\mathcal{SHIQ}$

Remember: $\mathcal{SHIQ}$ is OWL-DL without datatypes and individuals

Next: **tableau-based decision procedure for SAT (C,$\mathcal{T}$)**
we start with $\mathcal{ALC}$ ($\sqcap, \sqcup, \neg, \exists, \forall$) instead of $\mathcal{SHIQ}$ and **SAT**$(C, \emptyset)$

Technical: all concepts are assumed to be in **Negation Normal Form**
transform C into equivalent **NNF**(C) by pushing negation inwards, using

$$\begin{array}{lcl@{\qquad}lcl} \neg(C \sqcap D) & \equiv & \neg C \sqcup \neg D & \neg(C \sqcup D) & \equiv & \neg C \sqcap \neg D \\ \neg(\exists R.C) & \equiv & (\forall R.\neg C) & \neg(\forall R.C) & \equiv & (\exists R.\neg C) \end{array}$$

The algorithm decides **SAT**$(C, \emptyset)$ by trying to construct a **model** $\mathcal{I}$ for C

A Tableau Algorithm for $\mathcal{ALC}$

The algorithm works on a **completion tree** with

- nodes x corresponding to elements $x \in \Delta^{\mathcal{I}}$
- node labels $C \in \mathfrak{L}(x)$ meaning $x \in C^{\mathcal{I}}$
- edge labels (x, R, y) representing role successorships $(x, y) \in R^{\mathcal{I}}$

starts with root x with $\mathfrak{L}(x) = \{C\}$

applies **rules** that infer constraints on $\mathcal{I}$

answers "C *is satisfiable*" if rules

- **can be applied** (non-deterministic rules!)
- **exhaustively** (until no more rules apply)
- without generating a **clash** (node label with $\{A, \neg A\} \subseteq \mathfrak{L}(x)$)

Rules: see slide **Example:** $A \sqcap \exists R.A \sqcap \forall R.(\neg A \sqcup B)$ see blackboard

A Tableau Algorithm for $\mathcal{ALC}$

Theorem The tableau algorithm decides satisfiability of $\mathcal{ALC}$ concepts

Lemma let C be an $\mathcal{ALC}$ concept in NNF.

(a) the t-algorithm terminates when started with C
(b) **SAT**(C) $\Leftrightarrow$ rules can be applied exhaustively without generating a clash

Proof: (a) the t-algorithm builds a completion tree

- in a **monotonic** way
- whose **depth is bounded** by $|C|$: if y is an R-successor of x, then

$$\max\{|D| \mid D \in \mathfrak{L}(y)\} < \max\{|D| \mid D \in \mathfrak{L}(x)\}$$

- whose **breadth is bounded** by $|C|$: at most one successor per $\exists R.D \in$ **sub**(C)

A Tableau Algorithm for $\mathcal{ALC}$

Lemma let C be an $\mathcal{ALC}$ concept in NNF.

(a) the t-algorithm terminates when started with C
(b) **SAT**$(C) \Leftrightarrow$ rules can be applied exhaustively without generating a clash

Proof: (b) $\Leftarrow$ the **clash-free, complete** tree built for C corresponds to a model $\mathcal{I}$ of C:

- set $\Delta^{\mathcal{I}}$ to the nodes
- set $x \in A^{\mathcal{I}}$ iff $A \in \mathfrak{L}(x)$
- set $(x, y) \in R^{\mathcal{I}}$ iff (x, R, y) in completion tree
- prove that, if $D \in \mathfrak{L}(x)$, then $x \in D^{\mathcal{I}}$, by induction on structure of D
  Details: see blackboard

(this finishes the proof since $C \in \mathfrak{L}(x_0)$)

A Tableau Algorithm for $\mathcal{ALC}$

Lemma let C be an $\mathcal{ALC}$ concept in NNF.

(a) the t-algorithm terminates when started with C
(b) **SAT**(C) $\Leftrightarrow$ rules can be applied exhaustively without generating a clash

Proof: (b) $\Rightarrow$ use a model $\mathcal{I}$ of C with $a \in C^{\mathcal{I}}$ to steer rule application via mapping

$$\pi : \text{nodes of completion tree into } \Delta^{\mathcal{I}}$$

built together with completion tree that satisfies

1. if $C \in \mathfrak{L}(x)$, then $\pi(x) \in C^{\mathcal{I}}$
2. if (x, R, y), then $(\pi(x), \pi(y)) \in R^{\mathcal{I}}$

Existence of π implies clash-freeness of tree (1), termination is already proven
Construction of π: see blackboard with previous example

A Tableau Algorithm for $\mathcal{ALC}$ with TBoxes

Remember:

- A **GCI** is of the form $C \dot{\sqsubseteq} D$ for C, D (complex) concepts
- A **(general) TBox** is a finite set of GCIs
- $\mathcal{I}$ **satisfies** $C \dot{\sqsubseteq} D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ is **a model of TBox** $\mathcal{T}$ iff $\mathcal{I}$ satisfies each GCI in $\mathcal{T}$
- recall translation of GCIs into FOL

Extend $\mathcal{ALC}$ tableau algorithm to decide **SAT**$(C, \mathcal{T})$ for TBox

$$\mathcal{T} = \{C_i \dot{\sqsubseteq} D_i \mid 1 \leq i \leq n\} :$$

Add a new rule

$$\begin{array}{ll} \rightarrow_{\text{GCI}}\text{: If} & (\neg C_i \sqcup D_i) \notin \mathfrak{L}(x) \text{ for some } 1 \leq i \leq n \\ \text{Then} & \mathfrak{L}(x) \rightarrow \mathfrak{L}(x) \cup \{(\neg C_i \sqcup D_i)\} \end{array}$$

A tableau algorithm for $\mathcal{ALC}$ with general TBoxes

Example: Consider TBox $\{C \doteq \exists R.C\}$. Is C satisfiable w.r.t. this TBox?

A tableau algorithm for $\mathcal{ALC}$ with general TBoxes

Example: Consider TBox $\{C \doteq \exists R.C\}$. Is C satisfiable w.r.t. this TBox?

tableau algorithm no longer terminates!

Reason: the size of concepts no longer decreases along paths in a completion tree

Observation: most nodes in example completion tree are similar,

algorithm is **repeating** the same nodes

Solution: **Regain termination with cycle-detection**

if $\mathfrak{L}(x)$ and $\mathfrak{L}(y)$ are “very similar”, only extend $\mathfrak{L}(x)$

A tableau algorithm for $\mathcal{ALC}$ with general TBoxes: Cycle-detection

Blocking:

- x is **directly blocked** if it has an ancestor y with $\mathfrak{L}(x) \subseteq \mathfrak{L}(y)$
- in this case (and if y is the "closest" such node to x), x **is blocked by** y
- A node is **blocked** if it is directly blocked or one of its ancestors is blocked

$\oplus$ restrict the application of all rules to nodes which are not blocked

$\rightsquigarrow$ **Tableau algorithm for $\mathcal{ALC}$ w.r.t. TBoxes**

Example: check previous example

Theorem The extended t-algorithm decides satisfiability of $\mathcal{ALC}$ concepts w.r.t. TBoxes

A tableau algorithm for $\mathcal{ALC}$ with general TBoxes: Cycle-detection

Lemma let C be an $\mathcal{ALC}$ concept and $\mathcal{T}$ a TBox in in NNF.

(a) the t-algorithm terminates when started with C and $\mathcal{T}$
(b) **SAT**$(C, T) \Leftrightarrow$ rules can be applied exhaustively without generating a clash

Proof: (a) the t-algorithm builds a completion tree

- in a monotonic way
- whose depth is bounded by $2^{|C|}$:
  on any longer path, blocking would occur and
  paths with blocked nodes do not become longer
- whose breadth is bounded by $|C|$: at most one successor per $\exists R.D \in$ **sub**(C)

A tableau algorithm for $\mathcal{ALC}$ with general TBoxes: Cycle-detection

Lemma let C be an $\mathcal{ALC}$ concept and $\mathcal{T}$ a TBox in in NNF.

(a) the t-algorithm terminates when started with C and $\mathcal{T}$
(b) **SAT**$(C, \mathcal{T}) \Leftrightarrow$ rules can be applied exhaustively without generating a clash

Proof: (b) $\Rightarrow$ similar to previous

$\Leftarrow$ the **clash-free, complete** tree built for C corresponds to a model $\mathcal{I}$ of C **and** $\mathcal{T}$:

- set $\Delta^{\mathcal{I}}$ to the **unblocked** nodes
- set $x \in A^{\mathcal{I}}$ iff $A \in \mathcal{L}(x)$
- set $(x, y) \in R^{\mathcal{I}}$ iff (x, R, y) **or** (x, R, y') **and** y **blocks** y
- prove that, if $D \in \mathcal{L}(x)$, then $x \in D^{\mathcal{I}}$, by induction on structure of D
  Details: see blackboard

(this finishes the proof since $C \in \mathcal{L}(x_0)$ **and** $\neg C_i \sqcup D_i \in \mathcal{L}(x)$**, for all** i, x)

A tableau algorithm for $\mathcal{SHIQ}$: Transitive Roles

Remember: $\mathcal{SHIQ}$ allows to state **transitivity** of roles **trans**(R)

Problem: if $\forall R.C \in \mathfrak{L}(x)$ for R transitiv and

(x, R, y) and (y, R, z) in completion tree, C **must go to** $\mathfrak{L}(z)$

Solution1: add edge (x, R, z) ➠ destroys handy tree structure

Solution2: new $\forall$ rule

$\rightarrow_{\forall}^{+}$: If $\forall R.C \in \mathfrak{L}(x)$ **and** (x, R, y) **with** R **transitive**

and $\forall R.C \notin \mathfrak{L}(y)$

Then $\mathfrak{L}(y) \rightarrow \mathfrak{L}(y) \cup \{\forall R.C\}$

Proof of "the Lemma" is similar to previous case, but for model construction:

- if **trans**(R): $R^{\mathcal{I}} = \{(x, y) \mid (x, R, y)$ or (x, R, y') and y' blocks $y\}^{+}$

A tableau algorithm for $\mathcal{SHIQ}$: Role Hierarchies

Remember: $\mathcal{SHIQ}$ allows to state **role inclusions** $R \,\dot{\sqsubseteq}\, S$

Problem: if (x, R, y) and $R \,\dot{\sqsubseteq}^{+}\, S$, then $(x, y) \in S^{\mathcal{I}}$

Solution: define y **being an** S**-successor of** x if (x, R, y) for some $R \,\dot{\sqsubseteq}^{*}\, S$

in rules, replace "(x, R, y)" with "y **is** R**-successor of** x"

Problem2: if $\forall S.C \in \mathcal{L}(x)$ and R transitive and $R \,\dot{\sqsubseteq}\, S$ and

(x, R, y) and (y, R, z) in completion tree, then C **must go to** $\mathcal{L}(z)$

Solution: modify new $\forall$ rule

$\rightarrow_{\forall}^{+}$: If $\forall S.C \in \mathcal{L}(x)$, x has R-successor y for

R transitive and $R \,\dot{\sqsubseteq}^{*}\, S$ and $\forall R.C \notin \mathcal{L}(y)$

Then $\mathcal{L}(y) \rightarrow \mathcal{L}(y) \cup \{\forall R.C\}$

A tableau algorithm for $\mathcal{SHIQ}$: Inverse Roles

Remember: $\mathcal{SHIQ}$ allows to use role names and **inverse roles** R^-, e.g. $\forall R^-.C$

Problem1: concepts need get **pushed up** the completion tree

Example: $\exists R.(A \sqcap \forall R^-.(B \sqcap \exists S^-.(B \sqcap \forall S.\neg A)))$

Solution: treat role names and inverse roles **symmetrically**

define R-**neighbours** and replace “successor” with “neighbour” in rules

Problem2: algorithm not correct

Example: **SAT**$(A \sqcap \forall R^-.(A \sqcap \neg A),\ \{A \dot{\sqsubseteq} \exists R.C\})$

Solution: modify **direct blocking** condition: x **blocks** y **if** $\mathcal{L}(x) = \mathcal{L}(y)$

A tableau algorithm for $\mathcal{SHIQ}$: Number Restrictions

Remember: $\mathcal{SHIQ}$ allows to use **number restrictions** $(\geqslant nR.C)$, $(\leqslant nR.C)$

Obvious: new rules that **generate** R-successors y_i of x for $(\geqslant nr.C) \in \mathfrak{L}(x)$

new rules that **identify** surplus R-successors of x with $(\leqslant nr.C) \in \mathfrak{L}(x)$

Example: $(\geqslant 2R.A) \sqcap (\geqslant 2R.(A \sqcap B)) \sqcap (\leqslant 3S.A)$

Less obvious: new **choose rule** required

Example: $(\geqslant 3R.A) \sqcap (\leqslant 1R.A) \sqcap (\leqslant 1R.\neg A)$

Tricky: new blocking condition required

Proofs of Lemma become more demanding, i.e., **model construction**
uses enhanced "unravelling" to construct **possibly infinite models...**

Models of $\mathcal{SHIQ}$

For $\mathcal{SHIQ}$ **without number restriction**, we built **finite models**

ok since $\mathcal{SHI}$ has **finite model property**, i.e.,
SAT$(C, \mathcal{T}) \Rightarrow C$, $\mathcal{T}$ have a **finite model**

For full $\mathcal{SHIQ}$, we built **infinite tree models**

ok since $\mathcal{SHIQ}$ has **tree model property**, i.e.,
SAT$(C, \mathcal{T}) \Rightarrow C$, $\mathcal{T}$ have a **tree model**

ok since $\mathcal{SHIQ}$ **lacks finite model property**, i.e.,
there are C and $\mathcal{T}$ with **SAT**$(C, \mathcal{T})$,
but each of their models is **infinite**

Example: for $F \sqsubseteq R$ and R transitive,

$$\neg A \sqcap \exists F.A \sqcap \forall R.(A \sqcap \exists F.A \sqcap (\leq 1\ F^{-}\ \top))$$

is satisfiable, but each model has an infinite F-chain (blackboard)

Logical Foundations
for the
Semantic Web

4. Reasoning Services and Algorithms

Help knowledge engineer and users to build and use ontologies

Ian Horrock and Ulrike Sattler
University of Manchester
Manchester, UK
{horrocks|sattler}@cs.man.ac.uk

Plan for today

1. a few interesting complexity results for DLs
2. why full DAML+OIL and OWL-DL are more complex
3. some interesting undecidability results
4. implementing and optimising tableau algorithm

Remember Yesterday

Yesterday, we have seen a **tableau-based algorithm** that **decides**

satisfiability of $\mathcal{SHIQ}$ concepts w.r.t. $\mathcal{SHIQ}$ TBoxes

Still missing from $\mathcal{SHIQ}$ to OWL-DL:

- **data types** (integers, strings, with comparisons)
  e.g., $\texttt{Human} \sqcap \exists \texttt{age}.{>}18$ extension of algorithm not too difficult
- **nominals** (or **nominals**) ➠ $\mathcal{SHIQO}$
  e.g., $\texttt{Human} \sqcap \exists \texttt{met}.\texttt{Pope}$ extension of algorithm **very** difficult

Properties of $\mathcal{SHIQO}$

- decidable — not yet proven (but there are good reasons)
- **no** tree model property: *makes reasoning more difficult!*
- **more complex** than $\mathcal{SHIQ}$

Complexity of DLs: Summary

Deciding satisfiability (or subsumption) of

concepts in	Definition	without a TBox is	w.r.t. a TBox is
$\mathcal{ALC}$	$\sqcap$, $\sqcup$, $\neg$, $\exists R.C$, $\forall R.C$,	PSpace-c	ExpTime-c
$\mathcal{S}$	$\mathcal{ALC}$ + transitive roles	PSPace-c	ExpTime-c
$\mathcal{SI}$	$\mathcal{SI}$ + inverse roles	PSPace-c	ExpTime-c
$\mathcal{SH}$	$\mathcal{S}$ + role hierarchies	ExpTime-c	ExpTime-c
$\mathcal{SHIQ}$	$\mathcal{SHI}$ + number restrictions	ExpTime-c	ExpTime-c
$\mathcal{SHIQO}$	$\mathcal{SHI}$ + nominals	NExpTime-c	NExpTime-c
$\mathcal{SHIQ}^+$	$\mathcal{SHIQ}$ + “naive number restrictions”	undecidable	undecidable
$\mathcal{SH}^+$	$\mathcal{SH}$ + “naive role hierarchies”	undecidable	undecidable

$\mathcal{ALC}$ is in PSpace

The **NExpTime** tableau algorithm for **SAT**$(\mathcal{ALC}, \emptyset)$
can be modified easily to run in **PSpace**:

For an $\mathcal{ALC}$-concept C_0,

1. the c-tree can be built **depth-first**
2. **branches are independent** $\leadsto$ keep only one branch in memory at any time
3. **length** of branch $\leq |C_0|$
4. for each node x, $\mathfrak{L}(x) \subseteq$ **sub**(C_0) and $\#$ **sub**(C_0) is linear in $|C_0|$

$\leadsto$ **non-deterministic PSpace decision procedure for CSAT**$(\mathcal{ALC})$
and Savitch: PSpace $=$ NPSpace

Adding TBoxes to $\mathcal{ALC}$ yields ExpTime-hardness

Why is reasoning w.r.t. TBoxes more complex, i.e., **ExpTime**-hard?

Intuitively: we can enforce paths of **exponential length**, i.e.,

there are C, $\mathcal{T}$ such that, in each model $\mathcal{I}$ of C and $\mathcal{T}$, there is a path $x_1, \ldots, x_n$ with $(x_i, x_{i+1}) \in R^{\mathcal{I}}$ and $n \geq 2^{(|C|+|\mathcal{T}|)^2}$

C and $\mathcal{T}$ represent **binary incrementation using k bits**
i-th bit is coded in concept name X_i (X_k is lowest bit, $C \Rightarrow D$ short for $\neg C \sqcup D$)

$$A = \neg X_1 \sqcap \neg X_2 \sqcap \ldots \sqcap \neg X_k$$

$$\begin{aligned}
\mathcal{T} = \{ \quad & A \mathrel{\dot\sqsubseteq} \exists R.A \\
& A \mathrel{\dot\sqsubseteq} (X_k \Rightarrow \forall R.\neg X_k) \sqcap (\neg X_k \Rightarrow \forall R.X_k) \\
\texttt{for } i < k: \; & \bigsqcap_{j<i} X_j \mathrel{\dot\sqsubseteq} (X_i \Rightarrow \forall R.\neg X_i) \sqcap (\neg X_i \Rightarrow \forall R.X_i) \\
& \bigsqcup_{j<i} \neg X_j \mathrel{\dot\sqsubseteq} (X_i \Rightarrow \forall R.X_i) \sqcap (\neg X_i \Rightarrow \forall R.\neg X_i) \}
\end{aligned}$$

Adding TBoxes to $\mathcal{ALC}$ yields ExpTime-hardness

Why is reasoning w.r.t. TBoxes more complex, i.e., **ExpTime**-hard?

Lemma: Satisfiability of $\mathcal{ALC}$ w.r.t. TBoxes can be reduced to the **Halting Problem** of **polynomial-space-bounded alternating Turing machines**

We know: the **HP-f-PSB-A-TM** is **ExpTime**-hard

Proof of Lemma: beyond the scope of this tutorial, but not difficult

Complexity of $\mathcal{SHIQ}$

$\mathcal{SHIQ}$ **is ExpTime-hard** because $\mathcal{ALC}$ with TBoxes is and $\mathcal{SHIQ}$ can **internalise TBoxes:** polynomially reduce **SAT**$(C, \mathcal{T})$ to **SAT**$(C_{\mathcal{T}}, \emptyset)$

$$C_{\mathcal{T}} := C \sqcap \bigsqcap_{C_i \dot{\sqsubseteq} D_i \in \mathcal{T}} (C_i \Rightarrow D_i) \sqcap \forall U. \bigsqcap_{C_i \dot{\sqsubseteq} D_i \in \mathcal{T}} (C_i \Rightarrow D_i)$$

for U new role with **trans**(U), and

$R \dot{\sqsubseteq} U, R^{-} \dot{\sqsubseteq} U$ for all roles R in $\mathcal{T}$ or C

Lemma: C is satisfiable w.r.t. $\mathcal{T}$ iff $C_{\mathcal{T}}$ is satisfiable

Why is $\mathcal{SHIQ}$ in ExpTime?

Tableau algorithms runs in worst-case **non-deterministic double exponential space** using **double exponential time....**

$\mathcal{SHIQ}$ is in ExpTime

Translation of $\mathcal{SHIQ}$ into **Büchi Automata on infinite trees**

$$C, \mathcal{T} \quad \rightsquigarrow \quad A_{C,\mathcal{T}}$$

such that

1. **SAT**$(C, \mathcal{T})$ iff $L(A_{C,\mathcal{T}}) \neq \emptyset$
2. $|A_{C,\mathcal{T}}|$ is exponential in $|C| + |\mathcal{T}|$
   (states of ${}_{C,\mathcal{T}}$ are sets of subconcepts of C and $\mathcal{T}$)

This yields ExpTime decision procedure for **SAT**$(C, \mathcal{T})$ since

emptyness of $L(A)$ can be decided in time polynomial in $|A|$

Problem $A_{C,\mathcal{T}}$ needs (?) to be constructed before being tested: best-case ExpTime

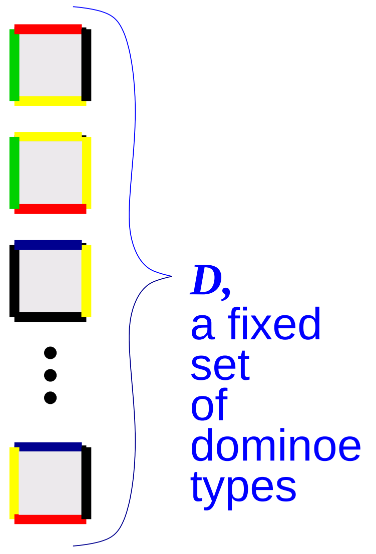
$\mathcal{SHIQO}$ is NExpTime-hard

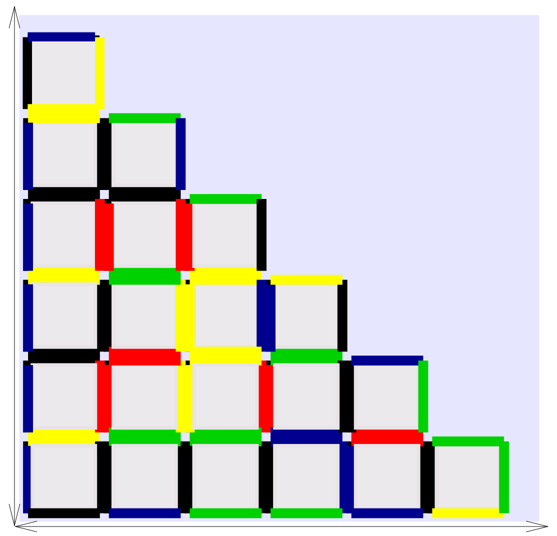
FaCT: for $\mathcal{SHIQ}$ and $\mathcal{SHOQ}$, **SAT**$(C, \mathcal{T})$ are **ExpTime-complete**

$\mathcal{SHOQ}$ is $\mathcal{SHIQ}$ without inverse roles, with nominals

Lemma: their combination is **NExpTime-hard**
even for $\mathcal{ALCQIO}$, **SAT**$(C, \mathcal{T})$ is **NExpTime-hard**

Proof: by reduction of a NExpTime version of the **domino problem:**





can we tile a 2^n x 2^n square using *D?*

NExpTime DLs: $\mathcal{ALCQIO}$ is NExpTime-hard

Definition: A **domino system** $\mathcal{D} = (D, H, V)$

- set of domino **types** $D = \{D_1, \ldots, D_d\}$, and
- horizontal and vertical **matching conditions** $H \subseteq D \times D$ and $V \subseteq D \times D$

A **tiling** of the $\mathbb{N} \times \mathbb{N}$ grid using $\mathcal{D}$:

$$\begin{aligned} t : \mathbb{N} \times \mathbb{N} \rightarrow\ & D \text{ such that} \\ & \langle t(m,n), t(m+1,n) \rangle \in H \text{ and} \\ & \langle t(m,n), t(m,n+1) \rangle \in V \end{aligned}$$

Domino problem			
	standard:	has $\mathcal{D}$ a tiling?	undecidable
	exponential:	has $\mathcal{D}$ a tiling for a $2^n \times 2^n$ square?	NExpTime-c.

NExpTime DLs: $\mathcal{ALCQIO}$ is NExpTime-hard

Reducing the NExpTime domino problem to **CSAT**$(\mathcal{ALCQIO})$ $\rightsquigarrow$ **four tasks:**

① **each object carries exactly one domino type D_i**
$\rightsquigarrow$ use concept name D_i for each domino type and

$$\top \dot\sqsubseteq \bigsqcup_{1 \leq i \leq d} (D_i \sqcap \bigsqcap_{j \neq i} \neg D_j)$$

② **each element x has exactly one H-successor**
exactly one V-successor
whose domino types satisfy the horizontal/vertical matching conditions:

$$\top \dot\sqsubseteq \bigsqcap_{1 \leq i \leq n} \Big(D_i \Rightarrow ((\leqslant 1 V.\top) \sqcap (\exists V. \bigsqcup_{(D_i, D_j) \in V} D_j)) \sqcap ((\leqslant 1 H.\top) \sqcap (\exists H. \bigsqcup_{(D_i, D_j) \in H} D_j)) \Big)$$

NExpTime DLs: $\mathcal{ALCQIO}$ is NExpTime-hard

③ the model must be large enough, i.e., have $2^n \times 2^n$ elements
$\rightsquigarrow$ encode the position (x, y) of each point using binary coding in the concept names $X_1, \dots, X_n, Y_1, \dots, Y_n$:

$$\top \;\dot{\sqsubseteq}\; \exists H.\top \sqcap \exists V.\top$$

$$\top \;\dot{\sqsubseteq}\; (X_k \Rightarrow \forall R.\neg X_k) \sqcap (\neg X_k \Rightarrow \forall R.X_k) \sqcap (\text{same for } Y_i)$$

$$\texttt{for } i < k : \; \bigsqcap_{j<i} X_j \;\dot{\sqsubseteq}\; (X_i \Rightarrow \forall R.\neg X_i) \sqcap (\neg X_i \Rightarrow \forall R.X_i) \sqcap (\text{same for } Y_i)$$

$$\bigsqcup_{j<i} \neg X_j \;\dot{\sqsubseteq}\; (X_i \Rightarrow \forall R.X_i) \sqcap (\neg X_i \Rightarrow \forall R.\neg X_i) \sqcap (\text{same for } Y_i)$$

E.g., if $x \in (\neg X_1 \sqcap X_2 \sqcap X_3 \sqcap Y_1 \sqcap \neg Y_2 \sqcap Y_3)^{\mathcal{I}}$, then
x represents $(011, 101)$, and thus the point $(3, 5)$

NExpTime DLs: $\mathcal{ALCQIO}$ is NExpTime-hard

④ ensure that the $V \circ H$-successor of each node coincides with its $H \circ V$-successor

$\rightsquigarrow$ enforce that each object is the H-successor of at most one element (and the same for V):

$$\top \doteq (\leqslant 1 V^{-}.\top) \sqcap (\leqslant 1 H^{-}.\top)$$

$\rightsquigarrow$ enforce that there is ≤ 1 object in the upper right corner:

$$X_1 \sqcap \ldots \sqcap X_n \sqcap Y_1 \sqcap \ldots \sqcap Y_n \mathrel{\dot\sqsubseteq} N$$

for nominal N

Harvest:

$$\neg X_1 \sqcap \ldots \sqcap \neg X_n \sqcap \neg Y_1 \sqcap \ldots \sqcap \neg Y_n$$

is satisfiable w.r.t. to $\mathcal{T}_D$ defined above iff D has a $2^n \times 2^n$-tiling

An Undeciable Extension for $\mathcal{SHIQ}$

In $\mathcal{SHIQ}$, each role R in a number restriction $(\bowtie\; n\, R; C)$ must be **simple**, i.e., **not** $({}^{+}S)$ **for any sub-role** S **of** R

Without this restriction, $\mathcal{SHIQ}$ (better: $\mathcal{SHQ}$) becomes **undecidable**

Proof by a reduction of the standard, unbounded domino problem

An Undeciable Extension for $\mathcal{SHIQ}$

Remember 4 tasks in the previous domino reduction:

① **each object carries exactly one domino type D_i**

$\rightsquigarrow$ use concept name D_i for each domino type and

$$\top \dot{\sqsubseteq} \bigsqcup_{1 \leq i \leq d} (D_i \sqcap \bigsqcap_{j \neq i} \neg D_j)$$

② **each element x has exactly one H-successor**
exactly one V-successor

whose domino types satisfy the horizontal/vertical matching conditions:

$$\top \dot{\sqsubseteq} \bigsqcap_{1 \leq i \leq n} \Big(D_i \Rightarrow ((\leqslant 1 V.\top) \sqcap (\exists V. \bigsqcup_{(D_i,D_j) \in V} D_j)) \sqcap ((\leqslant 1 H.\top) \sqcap (\exists H. \bigsqcup_{(D_i,D_j) \in H} D_j))\Big)$$

An Undeciable Extension for $\mathcal{SHIQ}$

Remember 4 tasks in the previous domino reduction:

③ **model must be large enough**

$$\top \stackrel{.}{\sqsubseteq} \exists V.\top \sqcap \exists H.\top$$

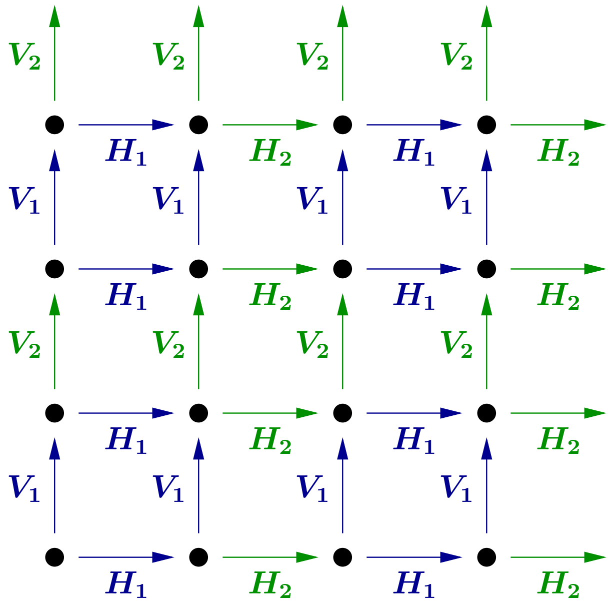
④ **vertical-horizontal and horizontal-vertical succcessor coincide**

- use additional roles $V_1, V_2 \stackrel{.}{\sqsubseteq} V$, $V_1, V_2 \stackrel{.}{\sqsubseteq} V$ with additional GCIs, e.g.,

$$\top \stackrel{.}{\sqsubseteq} (\exists V_1.\top \sqcap \forall V_1.\forall V_1.\bot) \sqcup \ldots$$

- **transitive roles** $D_{i,j}$ with $H_i, V_j \stackrel{.}{\sqsubseteq} D_{i,j}$
- number restrictions

$$\top \stackrel{.}{\sqsubseteq} \bigsqcap_{i,j} (\leq \ 3\ D_{i,j}.\top)$$



Implementing the $\mathcal{SHIQ}$ Tableau Algorithm

Naive implementation of $\mathcal{SHIQ}$ tableau algorithm is **doomed to failure:**

Construct a tree of **exponential depth** in a **non-deterministic way**
$\rightsquigarrow$ requires backtracking in a deterministic implementation

Optimisations are crucial
concern every aspect of the
help in "many" cases (which?)

In the following: a selection of some vital optimisations

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

FaCT provides service **"classify all concepts defined $\mathcal{T}$"**, i.e.,

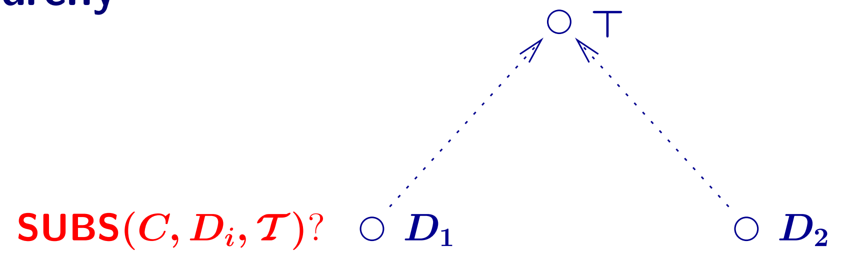
for all concept names C, D defined in $\mathcal{T}$, FaCT decides whether $C \sqsubseteq_{\mathcal{T}} D$ **and** $D \sqsubseteq_{\mathcal{T}} C$

$\rightsquigarrow$ **SAT**$(C \sqcap \neg D, \mathcal{T})$ and **SAT**$(D \sqcap \neg C, \mathcal{T})$

$\rightsquigarrow$ n^2 satisfiability tests!

Goal: reduce number of satisfiability tests when classifying TBox

Idea: trickle new concept into hierarchy computed so far



Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

FaCT provides service **"classify all concepts defined $\mathcal{T}$"**, i.e.,

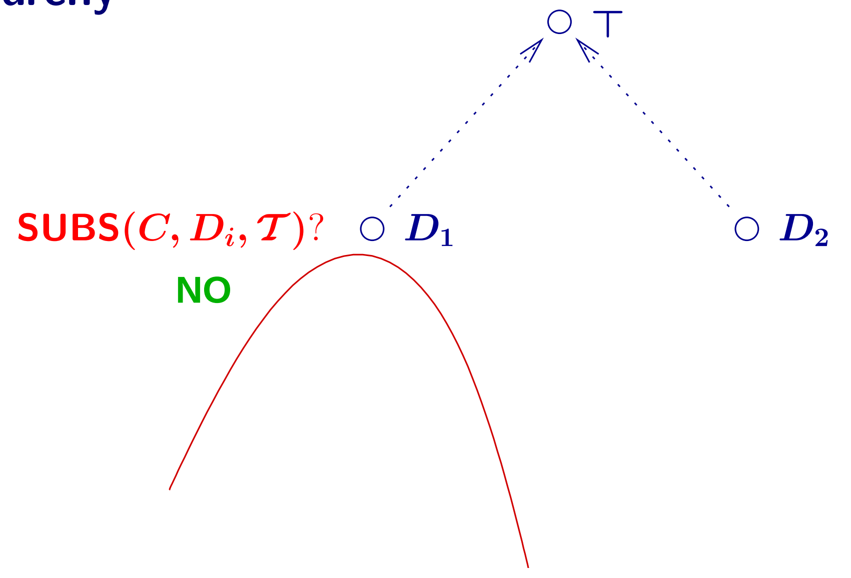
for all concept names C, D defined in $\mathcal{T}$, FaCT decides whether $C \sqsubseteq_{\mathcal{T}} D$ and $D \sqsubseteq_{\mathcal{T}} C$

$\rightsquigarrow$ **SAT**$(C \sqcap \neg D, \mathcal{T})$ and **SAT**$(D \sqcap \neg C, \mathcal{T})$

$\rightsquigarrow$ n^2 satisfiability tests!

Goal: reduce number of satisfiability tests when classifying TBox

Idea: trickle new concept into hierarchy computed so far



Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

FaCT provides service **"classify all concepts defined $\mathcal{T}$"**, i.e.,

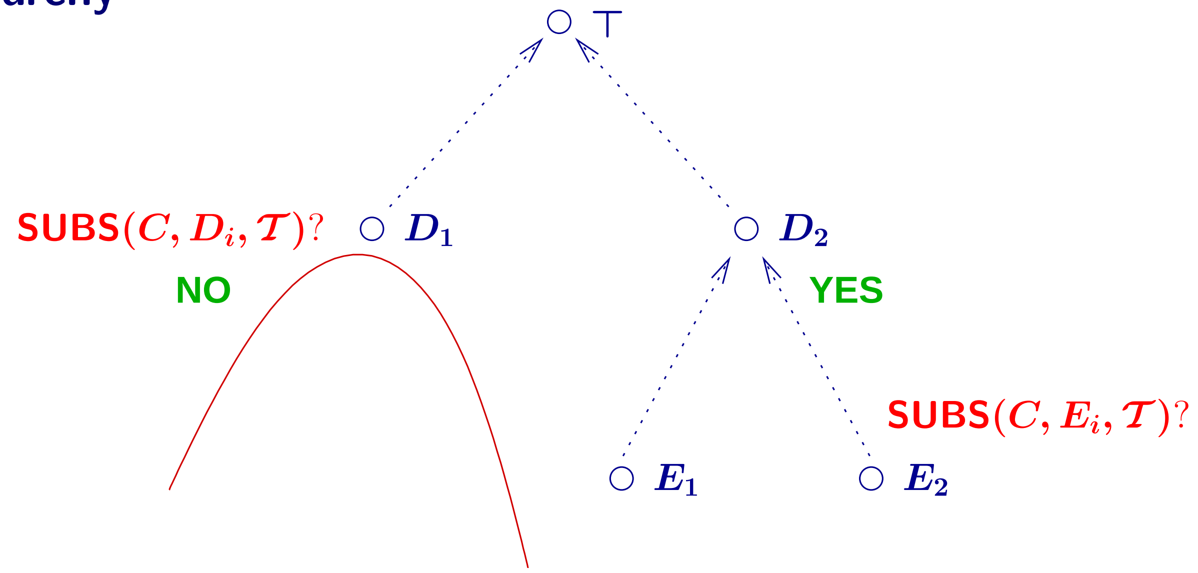
for all concept names C, D defined in $\mathcal{T}$, FaCT decides whether $C \sqsubseteq_{\mathcal{T}} D$ **and** $D \sqsubseteq_{\mathcal{T}} C$

$\leadsto$ **SAT**$(C \sqcap \neg D, \mathcal{T})$ and **SAT**$(D \sqcap \neg C, \mathcal{T})$

$\leadsto$ n^2 satisfiability tests!

Goal: reduce number of satisfiability tests when classifying TBox

Idea: "trickle" new concept into hierarchy computed so far



Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Remember: $\rightarrow_{\mathrm{GCI}}$: If $(\neg C_i \sqcup D_i) \notin \mathcal{L}(x)$ for some $1 \leq i \leq n$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{(\neg C_i \sqcup D_i)\}$

Problem: 1 disjunction per GCI $\rightsquigarrow$ high degree of **non-determinism**
huge **search space**

Observation: many GCIs are of the form $A \sqcap \ldots \dot{\sqsubseteq} C$ for concept name A
e.g., $\mathtt{Human} \sqcap \ldots \dot{\sqsubseteq} C$ versus $\mathtt{Device} \sqcap \ldots \dot{\sqsubseteq} C$

Idea: restrict applicability of $\rightarrow_{\mathrm{GCI}}$ by translating

$A \sqcap X \dot{\sqsubseteq} C$ into equivalent $A \dot{\sqsubseteq} \neg X \sqcup C$

e.g., $\mathtt{Human} \sqcap \exists \mathtt{owns.Pet} \dot{\sqsubseteq} C$ becomes $\mathtt{Human} \dot{\sqsubseteq} \neg \exists \mathtt{owns.Pet} \sqcup C$

this yields **localisation** of GCIs to As

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

For $\mathcal{SHIQ}$, the blocking condition is:

y is blocked by y' if

for x the predecessor of y, x' the predecessor of y'

1. $\mathfrak{L}(x) = \mathfrak{L}(x')$
2. $\mathfrak{L}(y) = \mathfrak{L}(y')$
3. (x, R, y) iff (x', R, y')

$\rightsquigarrow$ blocking occurs **late**
$\rightsquigarrow$ search space if **huge**

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

For $\mathcal{SHIQ}$, the blocking condition is:

y is blocked by y' if

for x the predecessor of y, x' the predecessor of y'

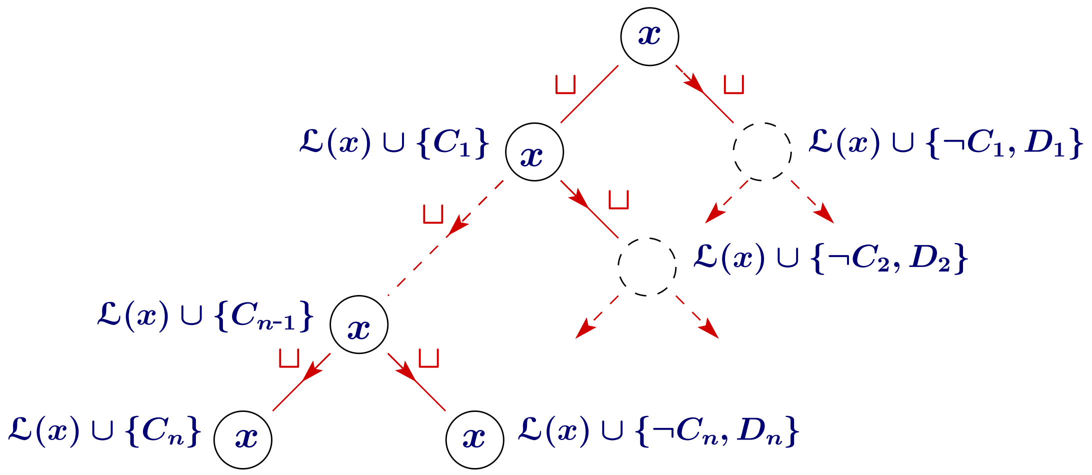
1. $\mathfrak{L}(x) = \mathfrak{L}(x')$	1. $\mathfrak{L}(x) \cap RC = \mathfrak{L}(x') \cap RC$
2. $\mathfrak{L}(y) = \mathfrak{L}(y')$	2. $\mathfrak{L}(y) \cap RC = \mathfrak{L}(y') \cap RC$
3. (x, R, y) iff (x', R, y')	3. (x, R, y) iff (x', R, y')
	for "**relevant concepts** RC"
$\rightsquigarrow$ blocking occurs **late**	$\rightsquigarrow$ blocking occurs **earlier**
$\rightsquigarrow$ search space if **huge**	$\rightsquigarrow$ search space if **smaller**

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Remember If a **clash** $(A, \neg A \in \mathcal{L}(x))$ is encountered, algorithm **backtracks**

i.e., returns to last non-deterministic choice and tries other possibility

Example $\exists R.(A \sqcap B) \sqcap (C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_1 \sqcup D_1) \sqcap \forall R.\neg A \in \mathcal{L}(x)$



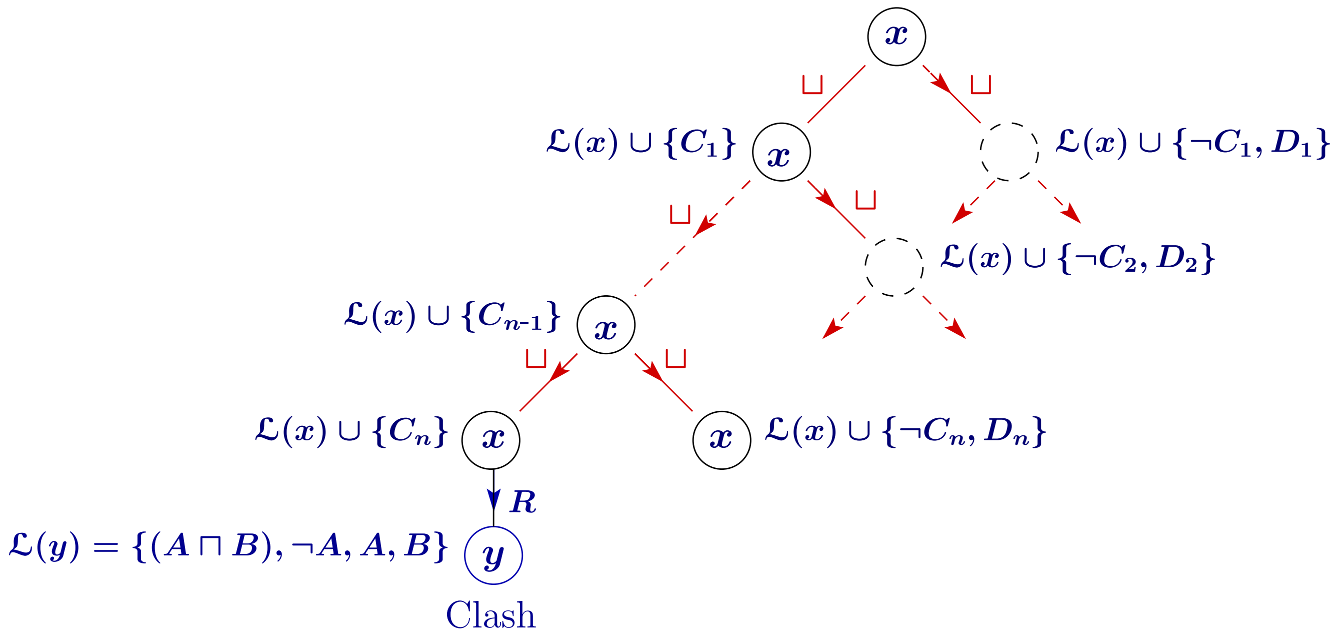
x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_1\}$
$\mathcal{L}(x) \cup \{\neg C_1, D_1\}$
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{\neg C_2, D_2\}$
$\mathcal{L}(x) \cup \{C_{n\text{-}1}\}$
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_n\}$
$\mathcal{L}(x) \cup \{\neg C_n, D_n\}$

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Remember If a **clash** ($A, \neg A \in \mathcal{L}(x)$) is encountered, algorithm **backtracks**

i.e., returns to last non-deterministic choice and tries other possibility

Example $\exists R.(A \sqcap B) \sqcap (C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_1 \sqcup D_1) \sqcap \forall R.\neg A \in \mathcal{L}(x)$



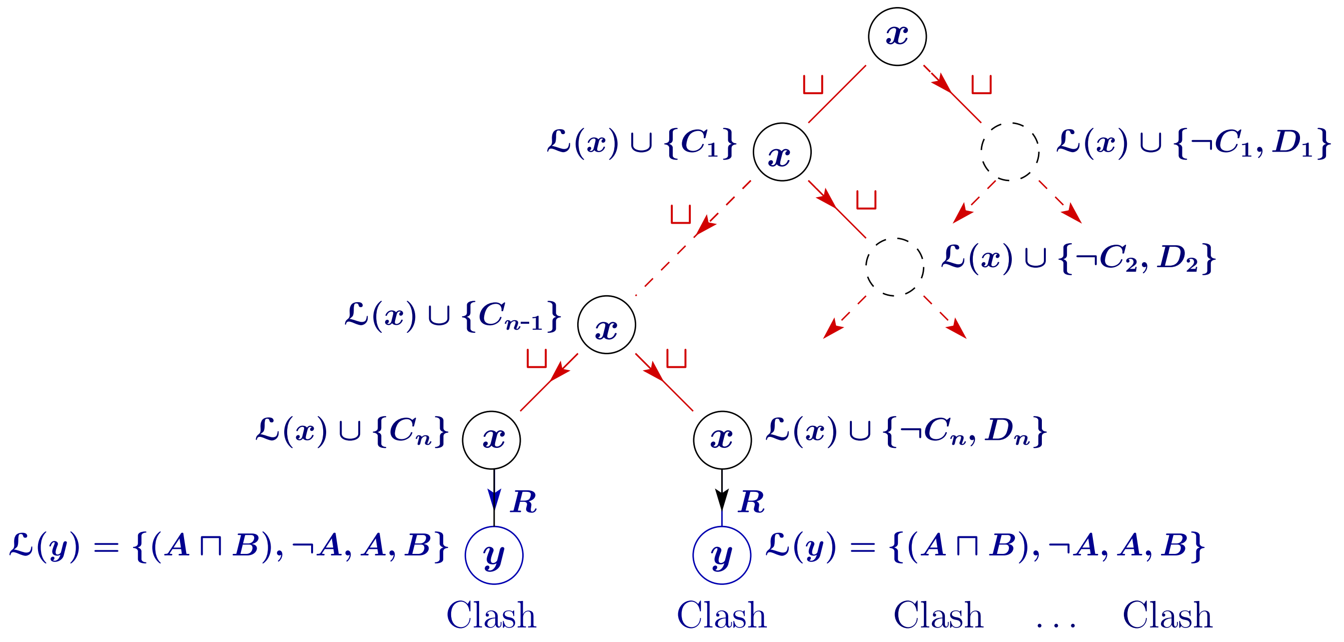
x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_1\}$
$\mathcal{L}(x) \cup \{\neg C_1, D_1\}$
$\mathcal{L}(x) \cup \{\neg C_2, D_2\}$
$\mathcal{L}(x) \cup \{C_{n-1}\}$
$\mathcal{L}(x) \cup \{C_n\}$
$\mathcal{L}(x) \cup \{\neg C_n, D_n\}$
R
$\mathcal{L}(y) = \{(A \sqcap B), \neg A, A, B\}$
y
Clash

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Remember If a **clash** $(A, \neg A \in \mathcal{L}(x))$ is encountered, algorithm **backtracks**

i.e., returns to last non-deterministic choice and
tries other possibility

Example $\exists R.(A \sqcap B) \sqcap (C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_1 \sqcup D_1) \sqcap \forall R.\neg A \in \mathcal{L}(x)$



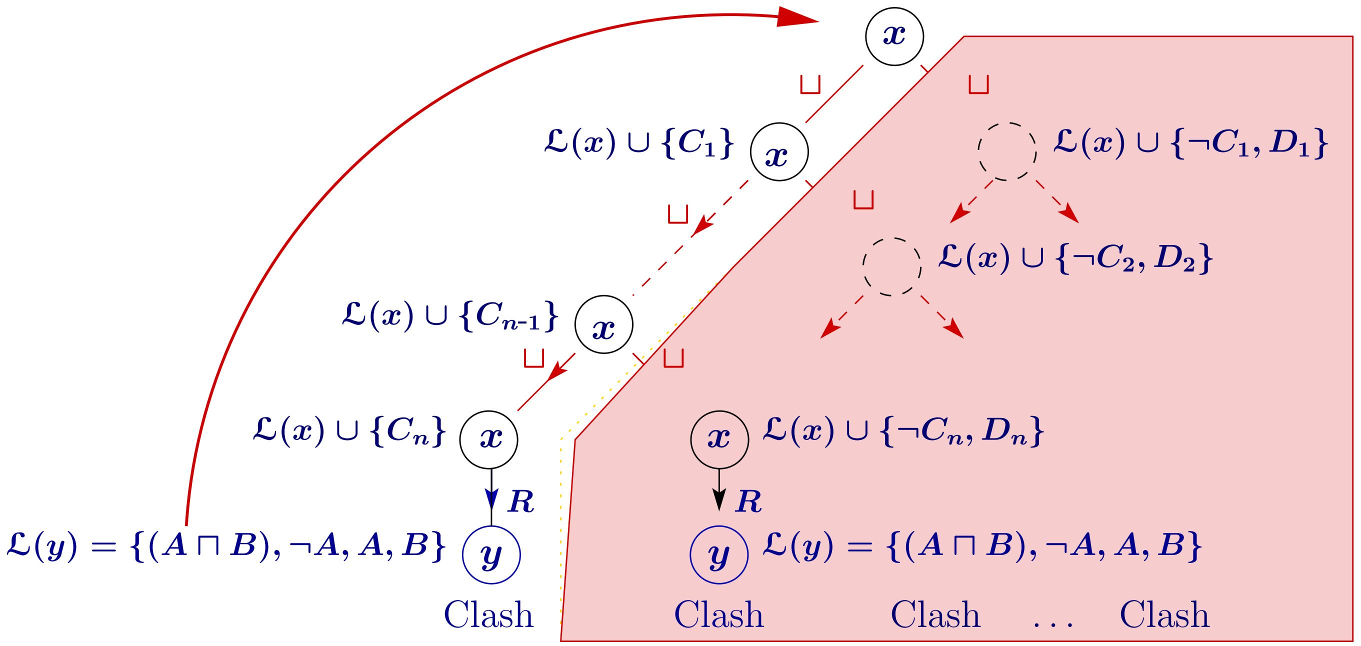
x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_1\}$
x
$\mathcal{L}(x) \cup \{\neg C_1, D_1\}$
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{\neg C_2, D_2\}$
$\mathcal{L}(x) \cup \{C_{n\text{-}1}\}$
x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_n\}$
x
x
$\mathcal{L}(x) \cup \{\neg C_n, D_n\}$
R
R
$\mathcal{L}(y) = \{(A \sqcap B), \neg A, A, B\}$
y
y
$\mathcal{L}(y) = \{(A \sqcap B), \neg A, A, B\}$
Clash
Clash
Clash
...
Clash

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Remember If a **clash** ($A, \neg A \in \mathcal{L}(x)$) is encountered, algorithm **backtracks**

i.e., returns to last non-deterministic choice and tries other possibility

Example $\exists R.(A \sqcap B) \sqcap (C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_1 \sqcup D_1) \sqcap \forall R.\neg A \in \mathcal{L}(x)$



x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_1\}$
x
$\mathcal{L}(x) \cup \{\neg C_1, D_1\}$
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{\neg C_2, D_2\}$
$\mathcal{L}(x) \cup \{C_{n\text{-}1}\}$
x
$\sqcup$
$\sqcup$
$\mathcal{L}(x) \cup \{C_n\}$
x
x
$\mathcal{L}(x) \cup \{\neg C_n, D_n\}$
R
R
$\mathcal{L}(y) = \{(A \sqcap B), \neg A, A, B\}$
y
y
$\mathcal{L}(y) = \{(A \sqcap B), \neg A, A, B\}$
Clash
Clash
Clash
...
Clash

Optimising the $\mathcal{SHIQ}$ Tableau Algorithm

Finally: $\mathcal{SHIQ}$ extends **propositional logic**

$\leadsto$ heuristics developed for **SAT** are relevant

Summing up: optimisations at each aspect of tableau algorithm

can dramatically enhance performance

$\leadsto$ do they interact?

$\leadsto$ how?

$\leadsto$ which combination works best for which "cases"?

$\leadsto$ is the optimised algorithm still correct?

5. Future Challenges, Outlook, and Leftovers

Ian Horrock and Ulrike Sattler
University of Manchester
Manchester, UK
{horrocks|sattler}@cs.man.ac.uk

Plan for today

1. ABoxes and instances
2. “non-standard” reasoning services
3. Nominals
4. Propagation
5. Concrete Domains
6. Keys
7. uuups - I get carried away

ABoxes and Instances

Remember: when using ontologies, we would like to automatically classify individuals described in an ABox

an **ABox** $\mathcal{A}$ is a finite set of **assertions** of the form

$$C(a) \text{ or } R(a,b)$$

How to decide whether **Inst**$(a, \mathcal{A}, \mathcal{T})$? I.e., whether $a \in C^{\mathcal{I}}$ in all models $\mathcal{I}$ of $\mathcal{T}$?

$\leadsto$ extend tableau algorithm to start with ABox $C(a) \in \mathcal{A} \Rightarrow C \in \mathfrak{L}(a)$

$R(a,b) \in \mathcal{A} \Rightarrow$ (a,R,y)

work on **forest** (rather than on a single tree)

i.e., trees whose root nodes intertwine

theoretically not too complicated

many problems in implementation

Non-Standard Reasoning Services

For Ontology Engineering, useful reasoning services can be based on **SAT** and **SUBS**

Are all useful reasoning services based on **SAT** *and* **SUBS***?*

Remember: to support modifying ontologies, we wanted

- automatic generation of **concept definitions from examples**
  - ➠ given ABox $\mathcal{A}$ and individuals a_i create
    a (most specific) concept C such that each $a_i \in C^{\mathcal{I}}$ in each model $\mathcal{I}$ of $\mathcal{T}$

$$\textbf{msc}(a_1, \ldots, a_n), \mathcal{A}, \mathcal{T})$$

- automatic generation of concept definitions for **too many siblings**
  - ➠ given concepts $C_1, \ldots, C_n$, create
    a (most specific) concept C such that **SUBS**$(C_i, C, \mathcal{T})$

$$\textbf{lcs}(C_1, \ldots, C_n), \mathcal{A}, \mathcal{T})$$

Non-Standard Reasoning Services: Most Specific Concept

Unlike **SAT**, **SUBS**, etc., msc is a **computation problem** (not decision problem)

Idea: $\textsf{msc}(a_1, \ldots, a_n, \mathcal{A}, \mathcal{T}) = \textsf{lcs}(\textsf{msc}(a_1, \mathcal{A}, \mathcal{T}), \ldots, \textsf{msc}(a_n, \mathcal{A}, \mathcal{T}))$

Known Results:

- lcs in DLs with $\sqcup$ is useless
- $\textsf{msc}(a_1, \mathcal{A}, \mathcal{T})$ does not need to exist

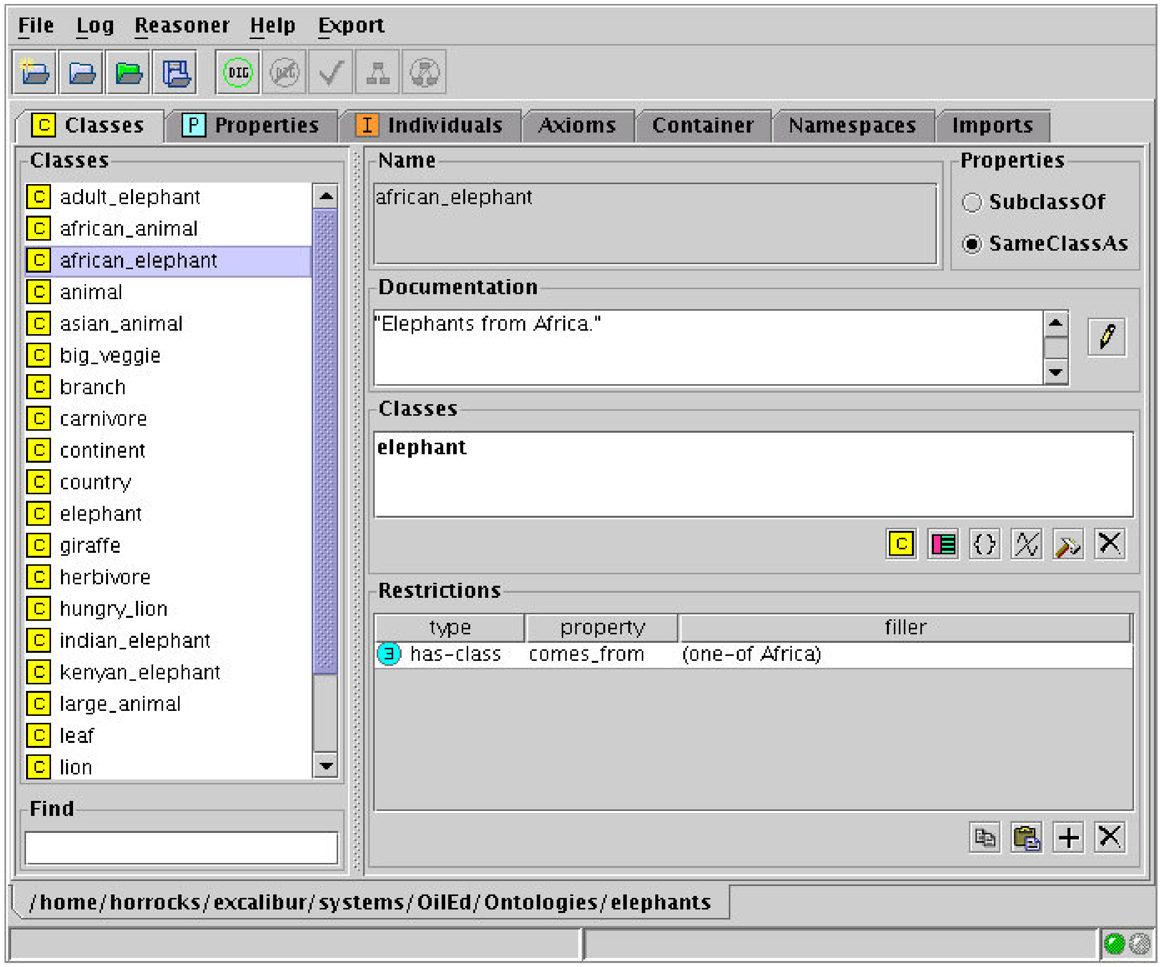
Acknowledgements

Thanks to various people from whom I "borrowed" material:

- Jeen Broekstra
- Carole Goble
- Frank van Harmelen
- Austin Tate
- Raphael Volz

And thanks to all the people from whom they borrowed it ☺

Intelligent Tools Demo



Classes

- adult_elephant
- african_animal
- african_elephant
- animal
- asian_animal
- big_veggie
- branch
- carnivore
- continent
- country
- elephant
- giraffe
- herbivore
- hungry_lion
- indian_elephant
- kenyan_elephant
- large_animal
- leaf
- lion

Find

Name: african_elephant

Properties: ( ) SubclassOf [✓ SameClassAs]

Documentation: "Elephants from Africa."

Classes: **elephant**

Restrictions

type	property	filler
∃ has-class	comes_from	(one-of Africa)

/home/horrocks/excalibur/systems/OilEd/Ontologies/elephants

Resources

- **Course material (including slides, tools and ontologies):**
  - **http://www.cs.man.ac.uk/~horrocks/ESSLLI2003/**
- **Description Logic Handbook**
  - **http://books.cambridge.org/0521781760.htm**

Additional Material

Tableau rules for $\mathcal{ALC}$

$\rightarrow_{\sqcap}$: **If** $C \sqcap D \in \mathcal{L}(x)$ **but** $\{C, D\} \cap \mathcal{L}(x) = \emptyset$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C, D\}$

$\rightarrow_{\sqcup}$: **If** $C \sqcup D \in \mathcal{L}(x)$ **but** $\{C, D\} \not\subseteq \mathcal{L}(x)$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{E\}$ **for** $E \in \{C, D\}$

$\rightarrow_{\exists}$: **If** $\exists R.C \in \mathcal{L}(x)$ **but** x **has no** R**-successor** y **with** $C \in \mathcal{L}(y)$
Then **create new** R**-successor** y **of** x **with** $\mathcal{L}(y) = \{C\}$

$\rightarrow_{\forall}$: **If** $\forall R.C \in \mathcal{L}(x)$ **and** x **has an** R**-successor** y **with** $C \notin \mathcal{L}(y)$
Then $\mathcal{L}(y) \rightarrow \mathcal{L}(y) \cup \{C\}$

Tableau rules for $\mathcal{ALC}$ with GCIs

$$\{C_i \dot{\sqsubseteq} D_i \mid 1 \leq i \leq n\}$$

applicable only to nodes x that are not blocked:

y **is blocked by an ancestor x if $\mathcal{L}(y) \subseteq \mathcal{L}(x)$**

$\rightarrow_{\sqcap}$: **If** $C \sqcap D \in \mathcal{L}(x)$ **but** $\{C, D\} \cap \mathcal{L}(x) = \emptyset$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C, D\}$

$\rightarrow_{\sqcup}$: **If** $C \sqcup D \in \mathcal{L}(x)$ **but** $\{C, D\} \not\subseteq \mathcal{L}(x)$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{E\}$ **for** $E \in \{C, D\}$

$\rightarrow_{\exists}$: **If** $\exists R.C \in \mathcal{L}(x)$ **but** x **has no** R**-successor** y
with $C \in \mathcal{L}(y)$
Then **create new** R**-successor** y **of** x **with**
$\mathcal{L}(y) = \{C\}$

$\rightarrow_{\forall}$: **If** $\forall R.C \in \mathcal{L}(x)$ **and** x **has an** R**-successor** y
with $C \notin \mathcal{L}(y)$
Then $\mathcal{L}(y) \rightarrow \mathcal{L}(y) \cup \{C\}$

$\rightarrow_{\text{GCI}}$: **If** $(\neg C_i \sqcup D_i) \notin \mathcal{L}(x)$
for some $1 \leq i \leq n$
Then $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{\neg C_i \sqcup D_i\}$

Tableau rules for $\mathcal{ALCI}$ with GCIs

$$\{C_i \dot{\sqsubseteq} D_i \mid 1 \leq i \leq n\}$$

applicable only to nodes x that are not blocked:

y is blocked by an ancestor x if $\mathfrak{L}(x) = \mathfrak{L}(y)$

$\rightarrow_\sqcap$: **If** $C \sqcap D \in \mathfrak{L}(x)$ **but** $\{C, D\} \cap \mathfrak{L}(x) = \emptyset$
Then $\mathfrak{L}(x) \rightarrow \mathfrak{L}(x) \cup \{C, D\}$

$\rightarrow_\sqcup$: **If** $C \sqcup D \in \mathfrak{L}(x)$ **but** $\{C, D\} \not\subseteq \mathfrak{L}(x)$
Then $\mathfrak{L}(x) \rightarrow \mathfrak{L}(x) \cup \{E\}$ **for** $E \in \{C, D\}$

$\rightarrow_\exists$: **If** $\exists R.C \in \mathfrak{L}(x)$ **but** x **has no** R**-neighbour** y **with** $C \in \mathfrak{L}(y)$
Then **create new** R**-successor** y **of** x **with** $\mathfrak{L}(y) = \{C\}$

$\rightarrow_\forall$: **If** $\forall R.C \in \mathfrak{L}(x)$ **and** x **has an** R**-neighbour** y **with** $C \notin \mathfrak{L}(y)$
Then $\mathfrak{L}(y) \rightarrow \mathfrak{L}(y) \cup \{C\}$

$\rightarrow_{\mathrm{GCI}}$: **If** $(\neg C_i \sqcup D_i) \notin \mathfrak{L}(x)$ **for some** $1 \leq i \leq n$
Then $\mathfrak{L}(x) \rightarrow \mathfrak{L}(x) \cup \{C_\mathcal{T}\}$

Additional tableau rules for $\mathcal{ALCQI}$ with GCIs applicable only to nodes x that are not blocked:

y is blocked by an ancestor y' if there are x, x' with

- **y is succ. of x, y' is succ. of x',**
- **$\mathfrak{L}(x) = \mathfrak{L}(y)$, $\mathfrak{L}(x') = \mathfrak{L}(y')$, and**
- **$\mathfrak{L}(\langle x, y\rangle) = \mathfrak{L}(\langle x', y'\rangle)$.**

$\rightarrow_{\geq}$: If $(\geqslant nR.C) \in \mathfrak{L}(x)$, x **is not blocked, and** x **has less than** n R**-neighbours** y_i **with** $C \in \mathfrak{L}(y_i)$
Then **create** n **new** R**-successor** $y_1, \ldots, y_n$ **of** x **with** $\mathfrak{L}(y_i) := \{C\}$ **and** $y_i \dot{\neq} y_j$ **for all** $i \neq j$

$\rightarrow_{\leq}$: If $(\leqslant nR.C) \in \mathfrak{L}(x)$, x **is not indirectly blocked,** x **has** $n+1$ R**-neighbours** $y_0, \ldots, y_n$ **with** $C \in \mathfrak{L}(y_i)$**, and there are** i, j **with** *not* $y_i \dot{\neq} y_j$ **and** y_j **is** *not* **an ancestor of** y_i
Then $\mathfrak{L}(y_i) \rightarrow \mathfrak{L}(y_i) \cup \mathfrak{L}(y_j)$,
make y_j**'s successors to successors of** y_i,
add $y_i \dot{\neq} z$ **for each** z **with** $y_j \dot{\neq} z$,
remove y_j **from the tree**

$\rightarrow_{choice}$: If $(\leqslant nR.C) \in \mathfrak{L}(x)$, x **is not indirectly blocked,** x **has an** R**-neighbour** y **with** $\{C, \dot{\neg}C\} \cap \mathfrak{L}(y) = \emptyset$
Then $\mathfrak{L}(y) \rightarrow \mathfrak{L}(y) \cup \{D\}$ **for some** $D \in \{C, \dot{\neg}C\}$

Translation of $\mathcal{ALCQIO}$-concepts into C2

(The mapping t_y is obtained by switching the roles of x and y in t_x)

$$\begin{aligned}
t_x(A) &= \mathbf{A}(x), \\
t_x(\neg C) &= \neg t_x(C) \\
t_x(C \sqcap D) &= t_x(C) \wedge t_x(D), \\
t_x(C \sqcup D) &= t_x(C) \vee t_x(D), \\
t_x(\exists R.C) &= \exists y.\mathbf{R}(x,y) \wedge t_y(C) \\
t_x(\forall R.C) &= \forall y.\neg\mathbf{R}(x,y) \vee t_y(C) \\
t_x(\geqslant nR.C) &= \exists^{\geq n} y.\mathbf{R}(x,y) \wedge t_y(C), \\
t_x(\geqslant nR^-.C) &= \exists^{\geq n} y.\mathbf{R}(y,x) \wedge t_y(C), \\
t_x(\leqslant nR.C) &= \exists^{\leq n} y.\mathbf{R}(x,y) \wedge t_y(C), \\
t_x(\leqslant nR^-.C) &= \exists^{\leq n} y.\mathbf{R}(y,x) \wedge t_y(C)
\end{aligned}$$

$$t(\mathcal{T}) = \bigwedge_{C \dot{\sqsubseteq} D \in \mathcal{T}} \forall x.t_x(C) \Rightarrow t_x(D)$$

$$\begin{aligned}
t(R \dot{\sqsubseteq} S) &= \forall x,y.\mathbf{R}(x,y) \Rightarrow \mathbf{S}(x,y) \\
t(\mathsf{trans}(R)) &= \forall x,y,z.(\mathbf{R}(x,y) \wedge \mathbf{R}(y,z)) \Rightarrow \mathbf{R}(x,z) \\
t_x(o) &= (x = a_o), \text{ for nominal } o \text{ and constant } a_o
\end{aligned}$$

Lemma:

1. sat$(C, \mathcal{T})$ **iff** $t_x(C) \wedge t(\mathcal{T})$ **is satisfiable**

2. sat$(C, D, \mathcal{T})$ **iff** $t(\mathcal{T}) \Rightarrow (\forall x.t_x(C) \Rightarrow t_x(D)$