Semantic Web - Summary
Semantic Web layer cake

- Trust
  - Signature
    - Encryption
  - Signature
- Proof
- Logic
- Rules / Query
- Ontology
- RDF Model & Syntax
- XML Query
- XML Schema
- XML
- Namespaces
- URI / IRI
- Unicode
Intro/Repetition

• Propositional logic/First order logic
  – (Un-)Decidability
  – Complexity of reasoning
  – Knowledge bases
  – Decision algorithms
  – Truth table, Resolution
  – Interesting subsets of FOL
XML

XML – Extensible Markup Language

• designed to describe semi-structured documents

• users may create their own tags (they can create their own specific languages)

• tags have no semantics indicating how to present documents through a Web browser
XML example

<? xml version="1.0" encoding="UTF-8" ?>
<book>
  <title>Semantic Web is Cool</title>
  <author>John Smith</author>
  <publisher>Springer</publisher>
  <year>1993</year>
</book>
XML

- Structure
- Attributes vs. elements
- Well-formedness
- Validity
- Parsing/Managing
  - DOM and SAX
- Mature technology by now
- Remaining problems
  - Over-complex specification
- Possible solution: Micro XML
XML – hints

• Understand
  – the syntax of XML
  – When is XML as a data format applicable?

• No need to learn DTD/XML-Schema syntax; only know their purpose/differences
RDF

- Triples (Subject-Predicate-Object)
- Resources, Properties, Values, Statements
- RDF/XML as one possible serialization
- Special Resources/Properties/Constructions
  - rdf:type
  - Blank nodes
  - Container
  - Reification
RDF Schema

- RDF provides a way to express simple statements about resources, using named properties and values, but

- We also need the ability to define the vocabularies (terms) they intend to use in those statements, specifically, to indicate that they are describing specific types or classes of resources

- RDF Schema: Towards a logical foundation
RDF Layer vs. RDFS Layer

- EducantAndEducators
  - Teacher
    - AssistantProfessor
    - AssociateProfessor
    - Professor
  - Student
    - PostgraduateStudent
    - PostdoctoralStudent
    - DoctoralStudent
  - Fausto Giunchiglia
    - isSupervisorOf
    - John

- subClassOf
- range
- domain
- type

Fausto Giunchiglia is a Professor who is the supervisor of John.
RDF Schema vs. XML Schema

- XML Schemas is all about **syntax**.
- RDF Schema is all about **semantics**.
- An XML Schema tool is intended to **validate** that an XML instance conforms to the syntax specified by the XML Schema.
- An RDF Schema tool is intended to **provide additional facts** to supplement the facts in RDF/XML instances.
- XML Schemas is **prescriptive** - an XML Schema prescribes what an element may contain, and the order the child elements may occur.
- RDF Schemas is **descriptive** - an RDF Schema simply describes classes and properties.
RDF(S) – hints

• It is sufficient to understand RDF’s graph view
• No need to learn RDF/XML serialization

• Be able to model a simple knowledge base in RDF
• You should know what can be modeled and what not
SPARQL 1.0

- Extraction of information from RDF graphs, Construction of new RDF graphs

PREFIX uni: <http://example.org/uni/>

SELECT ?name
FROM <http://example.org/personal>
WHERE { ?s uni:name ?name.
  ?s rdf:type uni:lecturer }
SPARQL 1.0 – language elements

- Prefix
- Select, From, Where
- Distinct
- Reduced
- Limit
- Offset
- Order By
SPARQL Summary

• Expressive graph query language

• Pro
  - Enables inferencing, Many implementations
  - Easy to use; similarity to SQL
  - RDF/RDF(S) -> SPARQL

• Con
  - No notion of privacy (not a real problem!)
  - Scalability not always clear
  - Locality as a problem: what is relevant for reasoning?
SPARQL – hints

• Be able to state simple SPARQL queries over RDF knowledge bases
• No need to learn complex syntax (beyond SPARQL 1.0)
• Remember some approaches which are used to store and query RDF knowledge bases
Description logic – hints

• Most important part of the lecture

=>

• Most important part for the exam
Description logics

- **Concept** names are equivalent to unary predicates
- **Role** names are equivalent to binary predicates
- **Individual** names are equivalent to constants
- **Operators** restricted so that:
  - Language is decidable and, if possible, of low complexity
  - No need for explicit use of variables
Description logics – Syntax/semantics

\[(C \sqcap D)^I = C^I \cap D^I\]
\[(C \sqcup D)^I = C^I \cup D^I\]
\[(\neg C)^I = \Delta^I \setminus C^I\]
\[\{x\}^I = \{x^I\}\]
\[(\exists R.C)^I = \{x | \exists y. \langle x, y \rangle \in R^I \land y \in C^I\}\]
\[(\forall R.C)^I = \{x | \forall y. \langle x, y \rangle \in R^I \Rightarrow y \in C^I\}\]
\[(\leq n R)^I = \{x | \#\{y | \langle x, y \rangle \in R^I\} \leq n\}\]
\[(\geq n R)^I = \{x | \#\{y | \langle x, y \rangle \in R^I\} \geq n\}\]
\[(R^-)^I = \{(x, y) | (y, x) \in R^I\}\]
DLs: Syntax/semantics 2

- Syntax/Semantics for concept descriptions lead to syntax/semantics for TBox/ABox-axioms
DL Naming

- Basic description logic is $ALC$ (equiv modal $K_{(m)}$)
  - Concepts constructed using $u$, $t$, $\top$, $9$ and $8$
- $S$ often used for $ALC$ with transitive roles
- Additional letters indicate other extension, e.g.:
  - $H$ for role inclusion axioms (role hierarchy)
  - $O$ for nominals (singleton classes, written $\{x\}$)
  - $I$ for inverse roles
  - $N$ for number restrictions (of form $6 \cdot n \ R$, $> n \ R$)
  - $Q$ for qualified number restrictions (of form $6 \cdot n \ R.C$, $> n \ R.C$)
  - ...
Basic Inference Tasks

- Ontology $O$: Tbox + ABox
- Knowledge is **correct** (captures intuitions)
  - Does $C$ subsume $D$ w.r.t. ontology $O$?
- Knowledge is **minimally redundant** (no unintended synonyms)
  - Is $C$ equivalent to $D$ w.r.t. $O$?
- Knowledge is **meaningful** (classes can have instances)
  - Is $C$ is satisfiable w.r.t. $O$?
- **Querying** knowledge
  - Is $x$ an **instance** of $C$ w.r.t. $O$?
  - Is $(x,y)$ an **instance** of $R$ w.r.t. $O$?

- How to solve these problems (with tableaux algorithms)?
DLs: Reasoning

• Tableaux algorithms
• Basic data structure: Tree over ABoxes
• TBox is internalized
• Basically four or more tableaux rules, one corresponding to each constructor
• Apply rules until
  – Contradiction in each path
  – No more rule applicable in one path (plus: no contradiction)
OWL 1

- OWL DL based on SHIQ Description Logic
  - In fact it is equivalent to SHOIN(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 fact there are three “species” of OWL (!)
  - OWL full is union of OWL syntax and RDF
  - OWL DL restricted to First Order fragment
  - OWL Lite is “simpler” subset of OWL DL
OWL1: big problem intractability

• No polynomial reasoning
• Solution:
  – Remove constructors
  – Consider their interaction carefully
• Examples: FL₀, EL++
OWL2: Features and Rationale

• Syntactic sugar
• New constructs for properties
  – Self
  – Qualified cardinality restrictions
• Extended datatypes
• Punning
• Some smaller stuff
OWL 2 summary

• The language-fragment landscape becomes too complicated
• Choice of fragment is a design choice again
  – No easy change at a later stage
• Probably one fragment will turn out to be most valuable
  – After an initial (already faded) hype about OWL QL, Rule Languages, especially Datalog +/-, are on the rise!
Ontology engineering: manual process

1. Determine scope
2. Consider reuse
3. Enumerate terms
4. Define taxonomy
5. Define properties
6. Define constraints
7. Add instances
8. Check for anomalies

Not a linear process!
Ontology engineering: hard ontologies

- There are “hard” and “not so hard” ontologies
- What distinguishes them?
Ontology engineering – hints

• Know how to engineer an ontology (manual process)
• Be able to engineer a simple ontology for a given domain
• How to align/merge different ontologies?
Rule systems

FOL: (All except (6)), (2)+(3)+(4): DLs
(4): Description Logic Programs (DLP), (3): Classical Negation
(4)+(5): Horn Logic Programs, (4)+(5)+(6): LP
(6): Non-monotonic features (like NAF, etc.)
Proposed rule systems

• Description Logic Programs
  – “Replacement of OWL layer”

• Semantic Web Rule Language
  – “on top of OWL layer”
Rule systems – hints

• Be able to state background knowledge with rules
• Transform a simple DL ontology into a DLP
  – What can be transformed, what not?
Open questions?

- If you have a doubt during your exam preparation, send me an email any time
Q-Teams @ bologna.lab

- bologna.lab
  - Tries to investigates new kinds of education and learning techniques at HU Berlin
  - Q-Teams, Q-Modul, Q-Tutorium, Q-Kolleg,…

- A Q-Team is a small research group composed of students
  - 5+ students
  - one semester

- The Q-Team should address open research problems
- Students get credit points
- Are you interested in a Q-Team about Semantic Web?
Q-Team: possible directions

• Developing a scalable semantic web inference system
  – A lot of optimization/heuristics work
• Implement an ontology in a domain of your interest
  – The research challenge should be clear here!
• More theoretical work
  – It is hard to predict the outcome