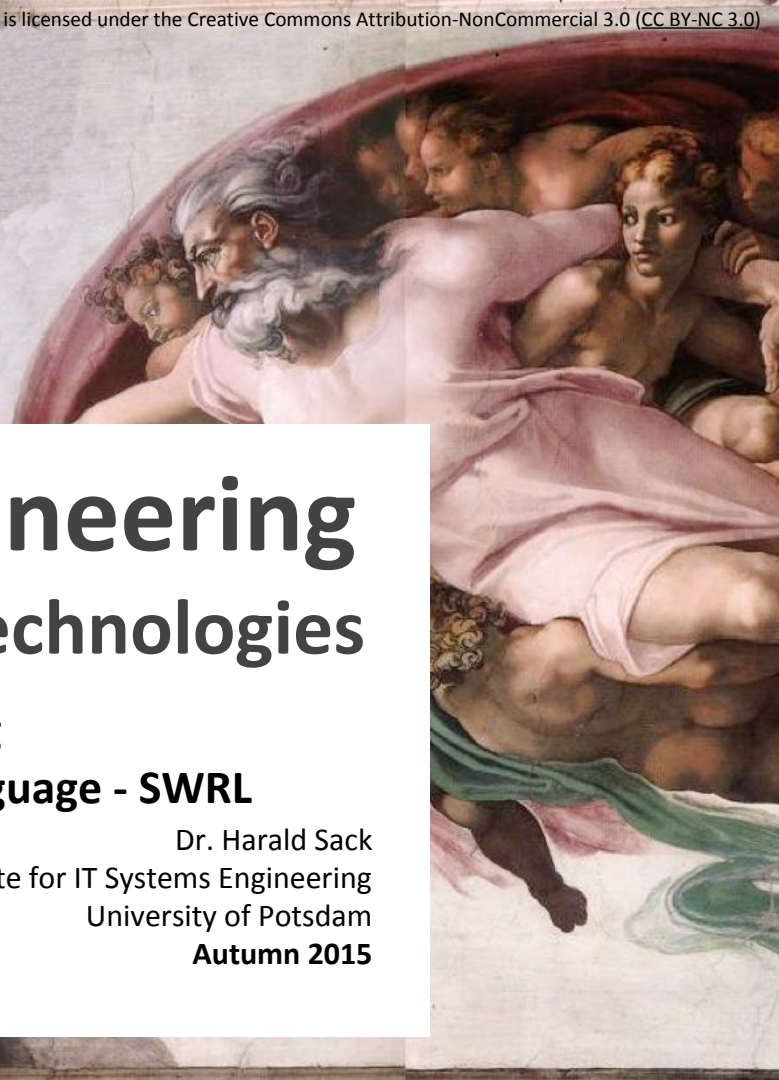
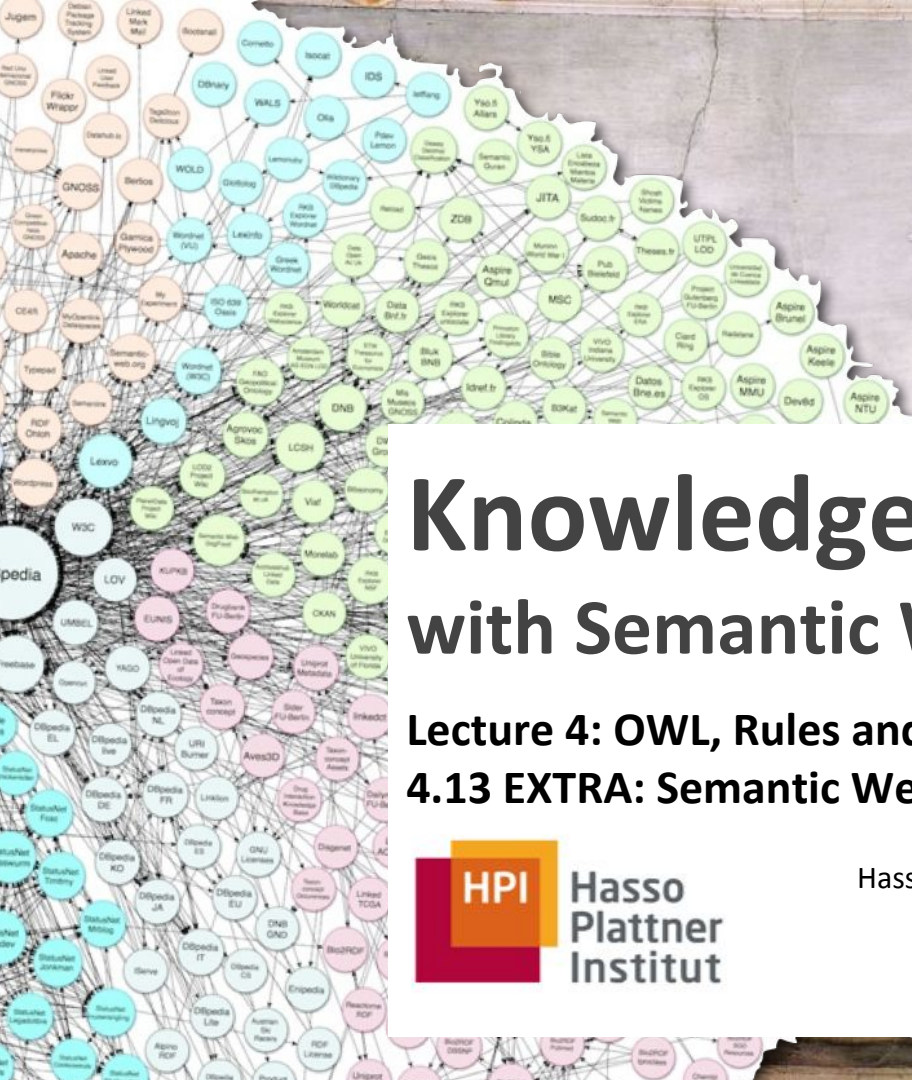




Knowledge Engineering with Semantic Web Technologies

Lecture 4: OWL, Rules and Reasoning

4.13 EXTRA: Semantic Web Rule Language - SWRL



Dr. Harald Sack
Hasso Plattner Institute for IT Systems Engineering
University of Potsdam
Autumn 2015

SWRL Semantic Web Rule Language

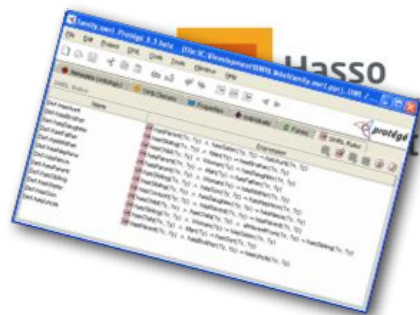
- based on combination of parts of **OWL** and **RuleML/DATALOG**
- **Idea:** DATALOG rules that apply on OWL ontologies
 - Symbols in rules can be OWL identifiers (or new DATALOG identifiers)
- W3C Submission (already in May 2004)
(developed by the Joint US/EU ad hoc Agent Markup Language Committee (JC) in collaboration with RuleML Initiative,
<http://www.w3.org/Submission/SWRL/>)
- **Syntax:**
 - XML Concrete Syntax (extends OWL XML Presentation Language)
 - RDF Concrete Syntax and abstract Syntax
- In SWRL rules are represented as
 - Implication of an **Antecedent** (Body) and a **Consequent** (Head)
 - SWRL is **undecidable**

The screenshot shows a Windows XP desktop environment. A Notepad++ window is open, displaying a list of IP addresses, likely from a network scan. The list includes various IP ranges and specific addresses, such as 192.168.1.1, 192.168.1.2, 192.168.1.3, 192.168.1.4, 192.168.1.5, 192.168.1.6, 192.168.1.7, 192.168.1.8, 192.168.1.9, 192.168.1.10, 192.168.1.11, 192.168.1.12, 192.168.1.13, 192.168.1.14, 192.168.1.15, 192.168.1.16, 192.168.1.17, 192.168.1.18, 192.168.1.19, 192.168.1.20, 192.168.1.21, 192.168.1.22, 192.168.1.23, 192.168.1.24, 192.168.1.25, 192.168.1.26, 192.168.1.27, 192.168.1.28, 192.168.1.29, 192.168.1.30, 192.168.1.31, 192.168.1.32, 192.168.1.33, 192.168.1.34, 192.168.1.35, 192.168.1.36, 192.168.1.37, 192.168.1.38, 192.168.1.39, 192.168.1.40, 192.168.1.41, 192.168.1.42, 192.168.1.43, 192.168.1.44, 192.168.1.45, 192.168.1.46, 192.168.1.47, 192.168.1.48, 192.168.1.49, 192.168.1.50, 192.168.1.51, 192.168.1.52, 192.168.1.53, 192.168.1.54, 192.168.1.55, 192.168.1.56, 192.168.1.57, 192.168.1.58, 192.168.1.59, 192.168.1.60, 192.168.1.61, 192.168.1.62, 192.168.1.63, 192.168.1.64, 192.168.1.65, 192.168.1.66, 192.168.1.67, 192.168.1.68, 192.168.1.69, 192.168.1.70, 192.168.1.71, 192.168.1.72, 192.168.1.73, 192.168.1.74, 192.168.1.75, 192.168.1.76, 192.168.1.77, 192.168.1.78, 192.168.1.79, 192.168.1.80, 192.168.1.81, 192.168.1.82, 192.168.1.83, 192.168.1.84, 192.168.1.85, 192.168.1.86, 192.168.1.87, 192.168.1.88, 192.168.1.89, 192.168.1.90, 192.168.1.91, 192.168.1.92, 192.168.1.93, 192.168.1.94, 192.168.1.95, 192.168.1.96, 192.168.1.97, 192.168.1.98, 192.168.1.99, 192.168.1.100, 192.168.1.101, 192.168.1.102, 192.168.1.103, 192.168.1.104, 192.168.1.105, 192.168.1.106, 192.168.1.107, 192.168.1.108, 192.168.1.109, 192.168.1.110, 192.168.1.111, 192.168.1.112, 192.168.1.113, 192.168.1.114, 192.168.1.115, 192.168.1.116, 192.168.1.117, 192.168.1.118, 192.168.1.119, 192.168.1.120, 192.168.1.121, 192.168.1.122, 192.168.1.123, 192.168.1.124, 192.168.1.125, 192.168.1.126, 192.168.1.127, 192.168.1.128, 192.168.1.129, 192.168.1.130, 192.168.1.131, 192.168.1.132, 192.168.1.133, 192.168.1.134, 192.168.1.135, 192.168.1.136, 192.168.1.137, 192.168.1.138, 192.168.1.139, 192.168.1.140, 192.168.1.141, 192.168.1.142, 192.168.1.143, 192.168.1.144, 192.168.1.145, 192.168.1.146, 192.168.1.147, 192.168.1.148, 192.168.1.149, 192.168.1.150, 192.168.1.151, 192.168.1.152, 192.168.1.153, 192.168.1.154, 192.168.1.155, 192.168.1.156, 192.168.1.157, 192.168.1.158, 192.168.1.159, 192.168.1.160, 192.168.1.161, 192.168.1.162, 192.168.1.163, 192.168.1.164, 192.168.1.165, 192.168.1.166, 192.168.1.167, 192.168.1.168, 192.168.1.169, 192.168.1.170, 192.168.1.171, 192.168.1.172, 192.168.1.173, 192.168.1.174, 192.168.1.175, 192.168.1.176, 192.168.1.177, 192.168.1.178, 192.168.1.179, 192.168.1.180, 192.168.1.181, 192.168.1.182, 192.168.1.183, 192.168.1.184, 192.168.1.185, 192.168.1.186, 192.168.1.187, 192.168.1.188, 192.168.1.189, 192.168.1.190, 192.168.1.191, 192.168.1.192, 192.168.1.193, 192.168.1.194, 192.168.1.195, 192.168.1.196, 192.168.1.197, 192.168.1.198, 192.168.1.199, 192.168.1.200, 192.168.1.201, 192.168.1.202, 192.168.1.203, 192.168.1.204, 192.168.1.205, 192.168.1.206, 192.168.1.207, 192.168.1.208, 192.168.1.209, 192.168.1.210, 192.168.1.211, 192.168.1.212, 192.168.1.213, 192.168.1.214, 192.168.1.215, 192.168.1.216, 192.168.1.217, 192.168.1.218, 192.168.1.219, 192.168.1.220, 192.168.1.221, 192.168.1.222, 192.168.1.223, 192.168.1.224, 192.168.1.225, 192.168.1.226, 192.168.1.227, 192.168.1.228, 192.168.1.229, 192.168.1.230, 192.168.1.231, 192.168.1.232, 192.168.1.233, 192.168.1.234, 192.168.1.235, 192.168.1.236, 192.168.1.237, 192.168.1.238, 192.168.1.239, 192.168.1.240, 192.168.1.241, 192.168.1.242, 192.168.1.243, 192.168.1.244, 192.168.1.245, 192.168.1.246, 192.168.1.247, 192.168.1.248, 192.168.1.249, 192.168.1.250, 192.168.1.251, 192.168.1.252, 192.168.1.253, 192.168.1.254, 192.168.1.255. A terminal window is also open, showing a netmap command being executed. A large 'PASS0' watermark is visible in the background.

Antecedent $\rightarrow$ Consequent

- Antecedent and Consequent are **Conjunctions** of assertions (atoms) of the form
 - $C(x)$ or $P(x, y)$
 - `sameAs(x, y)`, `differentFrom(x, y)`
- where x, y are variables, OWL individuals or elements of an OWL concrete domain
- $C(x)$ is an OWL class description
- $P(x, y)$ is an OWL property description

SWRL Semantic Web Rule Language



- SWRL Rule

$a \leftarrow b_1, \dots, b_n$ where a : head, $b_1, \dots, b_n$: body

- SWRL Knowledge Base

$k = (\Sigma, P)$ where Σ is an OWL knowledge base
 P is a finite rule set

- Atoms are defined as

$\text{Atom} \leftarrow C(i) \mid D(v) \mid R(i, j) \mid U(i, v) \mid \text{builtIn}(p, v_1, \dots, v_n) \mid i=j \mid i \neq j$

- | | |
|---|---|
| ○ $C \dots$ Class, $D \dots$ Datatype | ○ $v, v_1, \dots, v_n \dots$ Datatype Variable / Value Identifier |
| ○ $R \dots$ Object Property | ○ $p \dots$ name of a BuiltIn function |
| ○ $U \dots$ Datatype Property | |
| ○ $i, j \dots$ Variable / Individual identifier | |

- OWL DL (Description Logics) and **DATALOG** are applying the same interpretations
 - OWL *individuals* are **DATALOG constants**
 - OWL *classes* are unary **DATALOG predicates**
 - OWL *properties* are binary **DATALOG predicates**
- Interpretation can be model for OWL ontology as well as for a set of **DATALOG rules**



Entailment for OWL/DATALOG combination is possible

- Let $I = (\Delta^I, \Delta^D, \cdot^I, \cdot^D)$ an interpretation with
 - Δ^I = Object Interpretation domain
 - Δ^D = Datatype Interpretation domain
 - $\cdot^I$ = Object Interpretation function
 - $\cdot^D$ = Datatype Interpretation function
 - with $\Delta^I \cap \Delta^D = \perp$
 - V_{IX} are object variables with $V_{IX} \rightarrow 2^{\Delta^I}$
 - V_{DX} are datatype variables with $V_{DX} \rightarrow 2^{\Delta^D}$

- Interpretation of SWRL atoms:

SWRL atom	Interpretation
$C(i)$	$i^I \in C^I$
$R(i, j)$	$(i^I, j^I) \in R^I$
$U(i, v)$	$(i^I, v^D) \in U^I$
$D(v)$	$v^D \in D^D$
$\text{builtIn}(p, v_1, \dots, v_n)$	$v_1^D, \dots, v_n^D \in p^D$
$i = j$	$i^I = j^I$
$i \neq j$	$i^I \neq j^I$

- **SWRL Antecedent** is satisfied, iff
 - antecedent is empty (trivial) **or**
 - all atoms of the antecedent are satisfied
- **SWRL Consequent** is satisfied, iff
 - it is not empty **and**
 - the atom of the consequent is satisfied
- A **Rule** is satisfied for an Interpretation **I**, iff
 - the Interpretation **I**, which satisfies the antecedent also satisfies the consequent.

SWRL Example

$\text{hasUncle}(?x, ?z) \leftarrow \text{hasParent}(?x, ?y) \wedge \text{hasBrother}(?y, ?z)$

```
<ruleml:imp>
  <ruleml:_rlab ruleml:href="#onkel"/>
  <owlx:Annotation>
    <owlx:Documentation>The Uncle...</owlx:Documentation>
  </owlx:Annotation>
  <ruleml:_body>
    <swrlx:individualPropertyAtom swrlx:property="&family;hasParent">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>y</ruleml:var>
    </swrlx:individualPropertyAtom>
    <swrlx:individualPropertyAtom swrlx:property="&family;hasBrother">
      <ruleml:var>y</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:_body>
  <ruleml:_head>
    <swrlx:individualPropertyAtom swrlx:property="&family;hasUncle">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:individualPropertyAtom>
  </ruleml:_head>
</ruleml:imp>
```

SWRL is undecidable

- Logical Inference for OWL+SWRL is **undecidable**
- There is **no known algorithm that is able to entail all possible inferences for all SWRL knowledge bases**,
- even with unlimited resources and time
- But, from a **practical perspective**, there are
 - Algorithms that are able to entail **all possible inferences for some SWRL knowledge bases**
 - Algorithms that are able to entail **some inferences for all SWRL knowledge bases**

Tool Support for SWRL

- Bossam, R2ML, Hoolet, Pellet, KAON2, RacerPro,
- Jess, SWRLTab, SWRLQueryTab

The screenshot displays the SWRLTab web interface for the ontology 'book-example0' (URL: <http://www.semanticweb.org/harald/ontologies/2014/5/book-example0>). The interface includes a search bar, tabs for 'Active Ontology', 'Entities', 'Classes', 'Object Properties', 'Data Properties', 'Individuals by class', and 'DL Query'. The 'Classes' tab is active, showing a class hierarchy on the left and a detailed view of the 'Thing' class on the right. The class hierarchy includes 'Thing', 'Genre', 'Person', 'ReadingMaterial', and 'ThingsInStore'. The detailed view for 'Thing' shows 'Annotations', 'Description', and 'Rules' sections. The 'Rules' section contains two SWRL rules:
1. `Poetry(?x), Writer(?y), author(?x, ?y) -> Poet(?y)`
2. `Book(?x), Writer(?y), author(?x, ?y), literaryGenre(?x, ScienceFiction) -> ScienceFictionAuthor(?y)`
The interface also shows 'Reasoner active' and a 'Show Inferences' checkbox.



14 EXTRA - Rules expressible in OWL

OpenHPI - Course Knowledge Engineering with Semantic Web Technologies

Lecture 4: OWL, Rules, and Reasoning