Knowledge Representation for the Semantic Web

Winter Quarter 2010

Slides 5 – 01/26/2010

Pascal Hitzler
Kno.e.sis Center
Wright State University, Dayton, OH
http://www.knoesis.org/pascal/

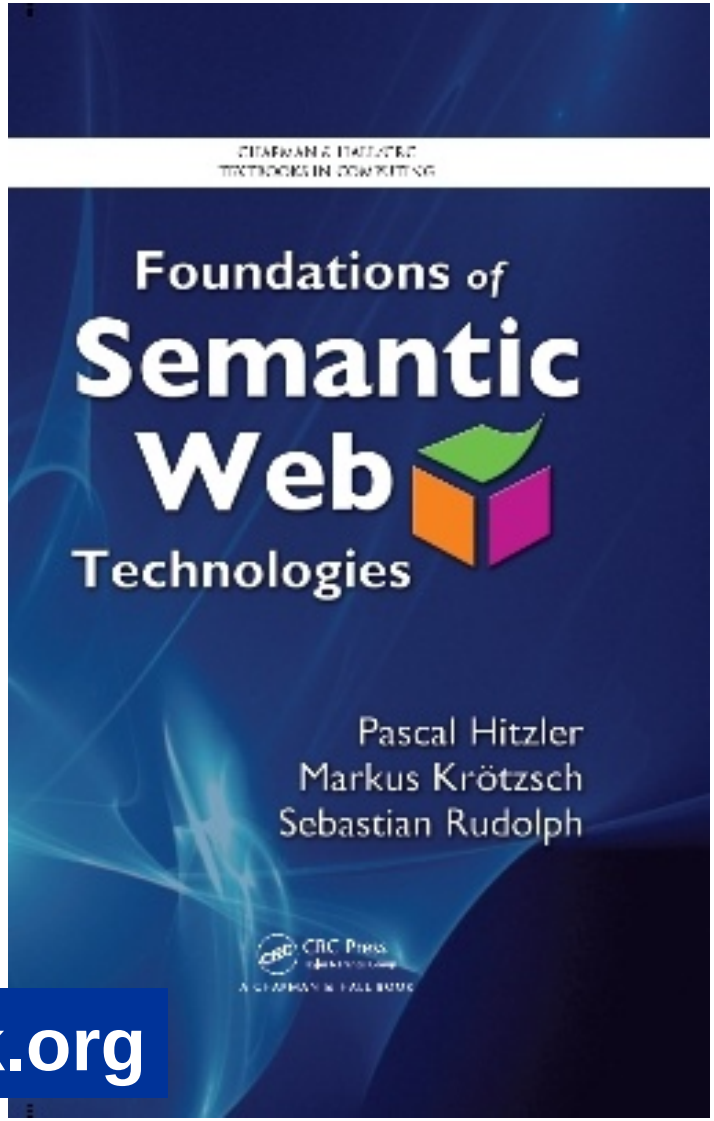
Slides are based on

Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph

Foundations of Semantic Web Technologies

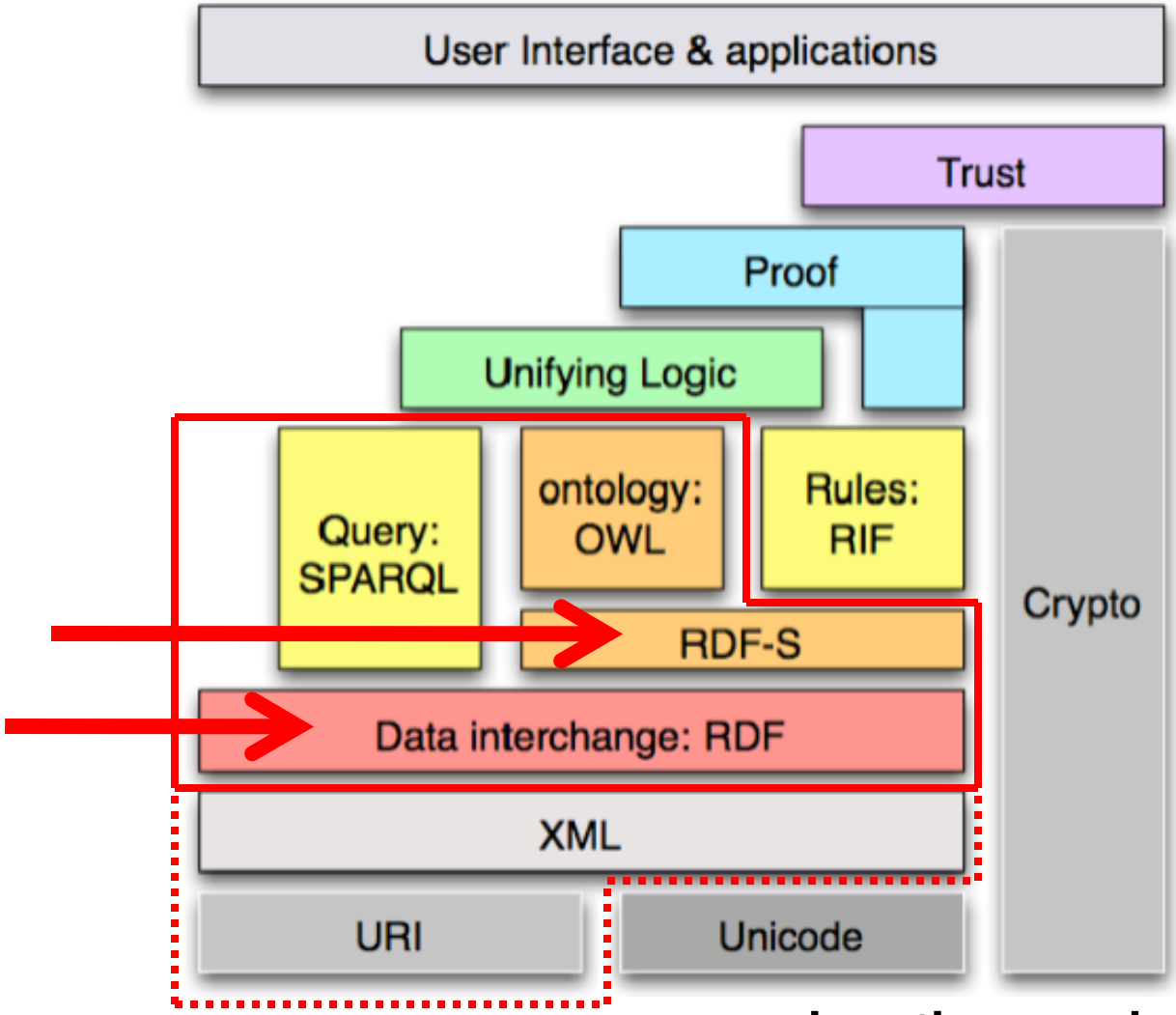
Chapman & Hall/CRC, 2010

Flyer with special offer is available.



http://www.semantic-web-book.org

Today: RDF(S) semantics



+ conjunctive queries for OWL

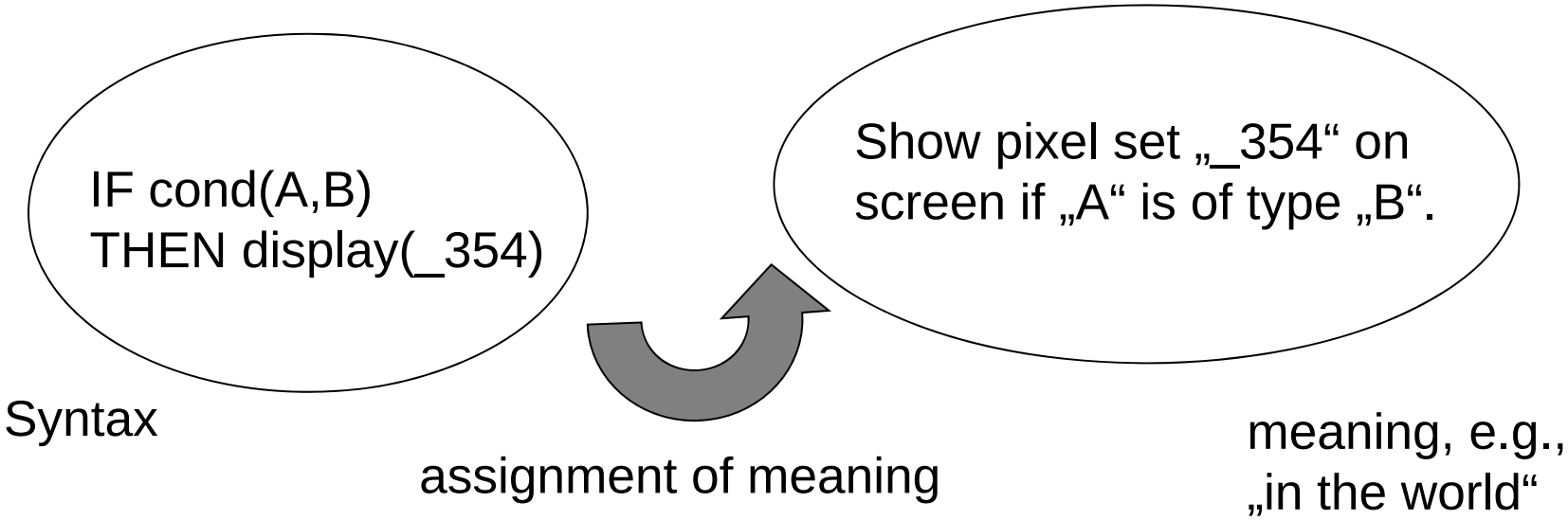
Today's Session: RDF(S) semantics

1. **What is Semantics?**
2. **What is Model-theoretic Semantics?**
3. **Model-theoretic Semantics for RDF(S)**
4. **What is Proof-theoretic Semantics?**
5. **Proof-theoretic Semantics for RDF(S)**
6. **Class Project**
7. **Class Presentations**

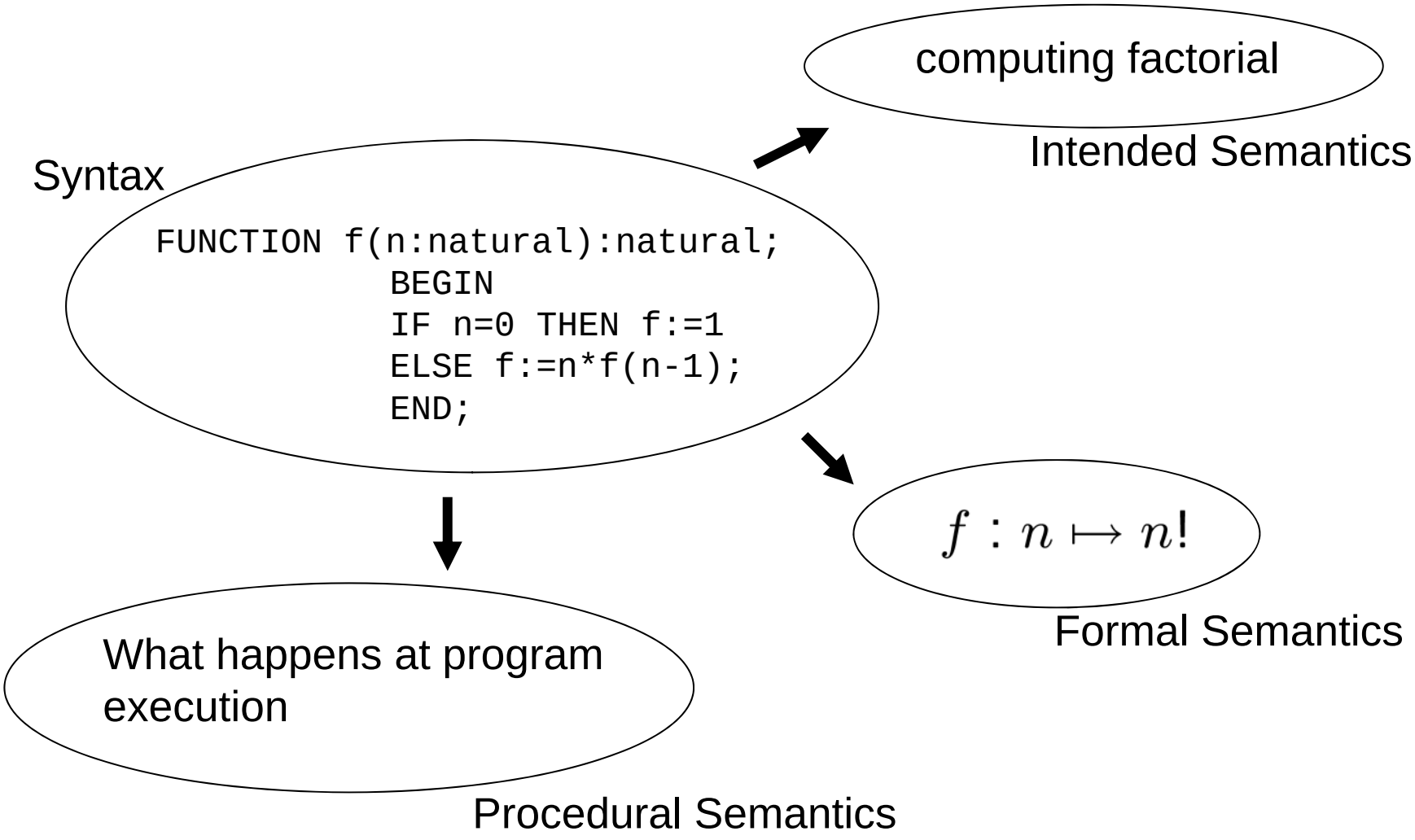
Syntax and Semantics

Syntax: character strings without meaning

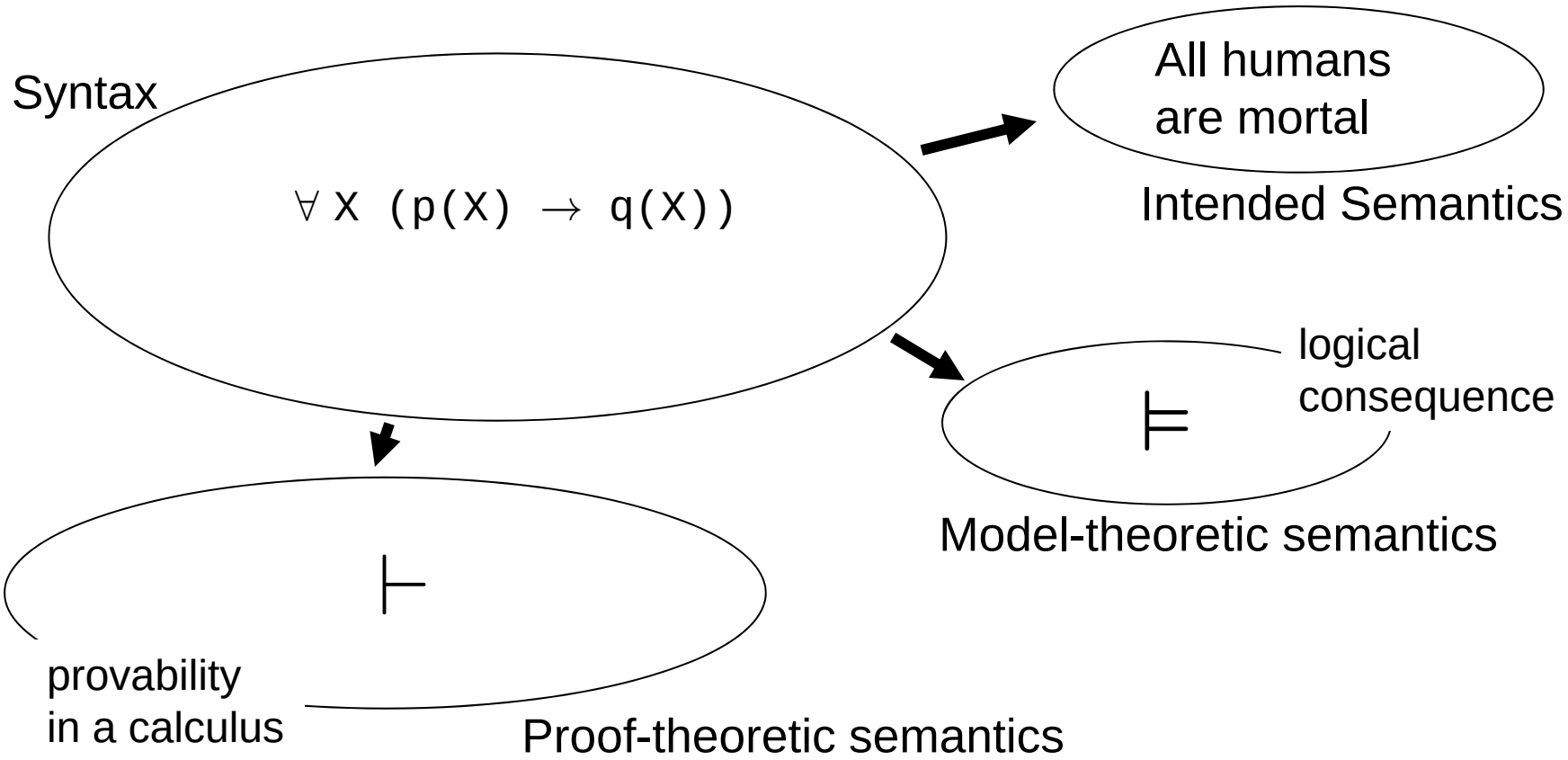
Semantics: meaning of the character strings



Semantics of Programming Languages



Semantics of Logic



Today's Session: RDF(S) semantics

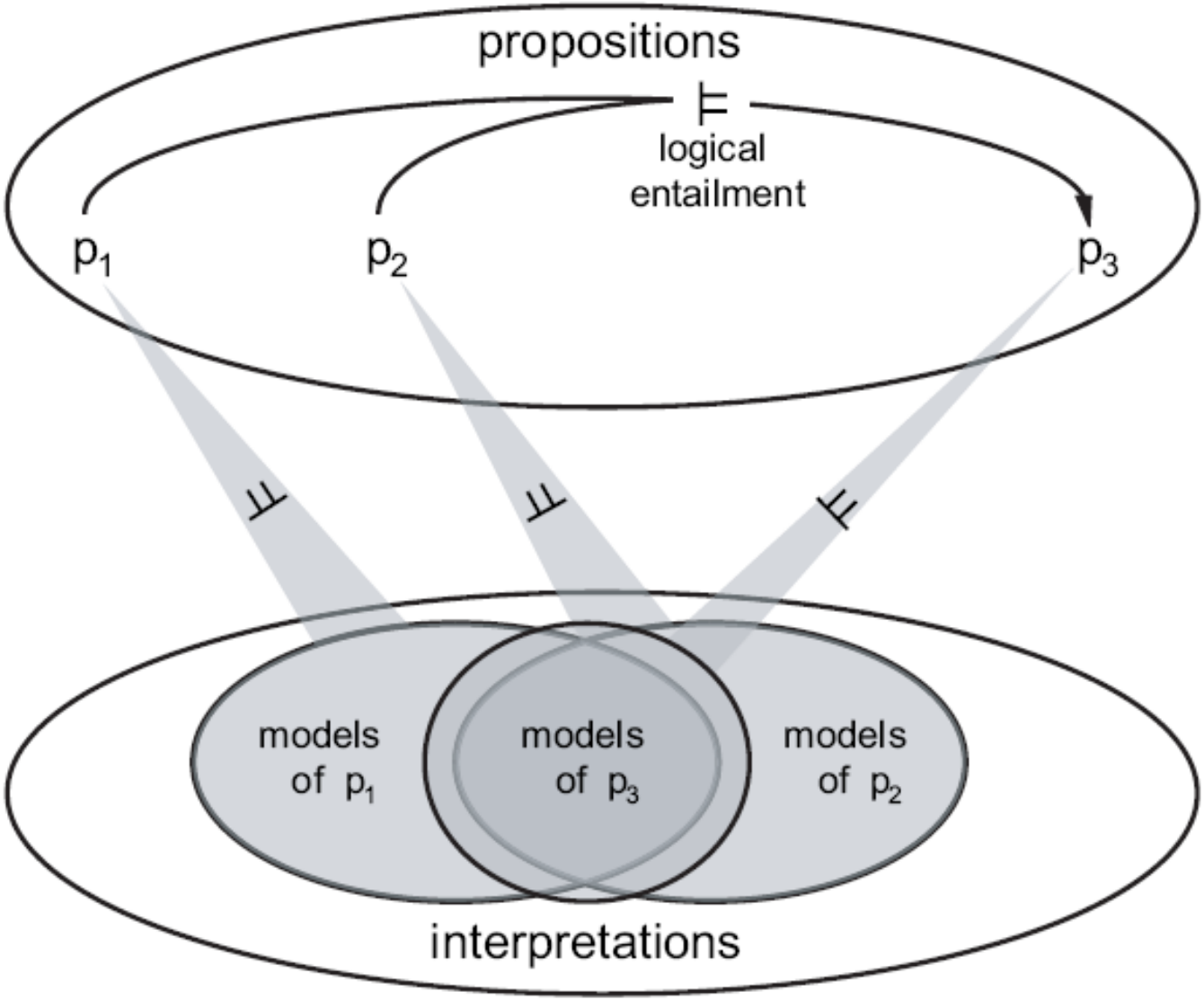
1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

Model-theoretic Semantics

- You need:
  - a language/syntax
  - a notion of *model* for sentences in the language
- Models
  - are made such that each sentence is either true or false in each model
  - If a sentence α is true in a model M, then we write M$\models\alpha$
- Logical consequence:
  - β is a logical consequence of α (written $\alpha\models\beta$), if for all M with M$\models\alpha$, we also have M$\models\beta$ are
  - If K is a set of sentences, we write K$\models\beta$ if M$\models\beta$ for each $M\models$K
  - If J is another set of sentences, we write K$\models$J if K$\models\beta$ for each $\beta \in$ J

  (note that the notation $\models$ is overloaded)

Logical Consequence



Model theory (contrived) example

- **Language:**
  variables ...,w,x,y,z,...
  symbol η
  allowed sentences: **a η b** **(for a, b any variables)**

- **We want to know:**

  What are the logical consequences of the set

  {x η y, y η z}

- **To answer this, we must say what the models in our semantics are.**

Model theory (contrived) example

- Say, a model I of a set K of sentences consists of
  - a set C of cars and
  - a function I(·) which maps each variable to a car in C

  such that, for each sentence a η b in K we have that
  I(a) has more horsepower than I(b).

- We now claim that {x η y, y η z} $\models$ x η z.
- Proof: Consider any model M of {x η y, y η z}.
  Since M$\models$ {x η y, y η z}, we know that
  M(x) has more horsepower than M(y) and
  M(y) has more horsepower than M(z).
  Hence, M(x) has more horsepower than M(z), i.e. M$\models$ x η z.

  This argument holds for all models of {x η y, y η z}, therefore
  {x η y, y η z}$\models$ x η z.

Model theory (contrived) example

- Say, a model I of a set K of sentences consists of
  - a set C of cars and
  - a function I(·) which maps each variable to a car in C

  such that, for each sentence a η b in K we have that
  I(a) has more horsepower than I(b).

- **An *interpretation* I for a our language consists of**
  - **a set C of cars and**
  - **a function I(·) which maps each variable to a car in C.**

  (and that's it, i.e. no information whether a sentence is true or false with respect to I).

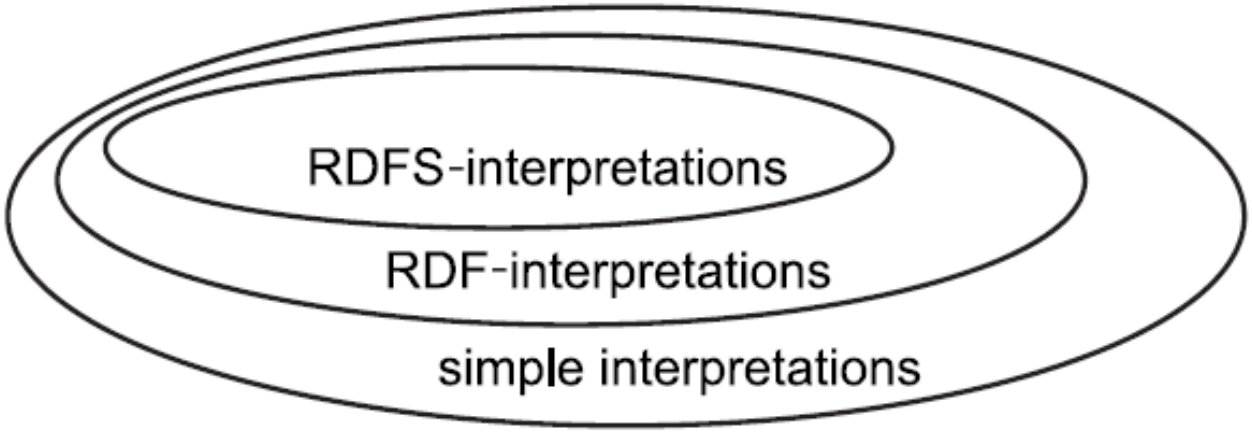
Today's Session: RDF(S) semantics

1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

Now let's do this for RDF(S)

- **Language: Whatever is valid RDF(S).**
- **Sentences are triples. (Graphs are sets of triples.)**

- **Interpretations are given via sets and functions from language vocabularies to these sets.**
- **Models are defined such that they capture the intended meaning of the RDF(S) vocabulary.**
- **And there are three different notions:**



Simple Interpretations

So we define: a *simple interpretation* $\mathcal{I}$ of a given vocabulary V consists of

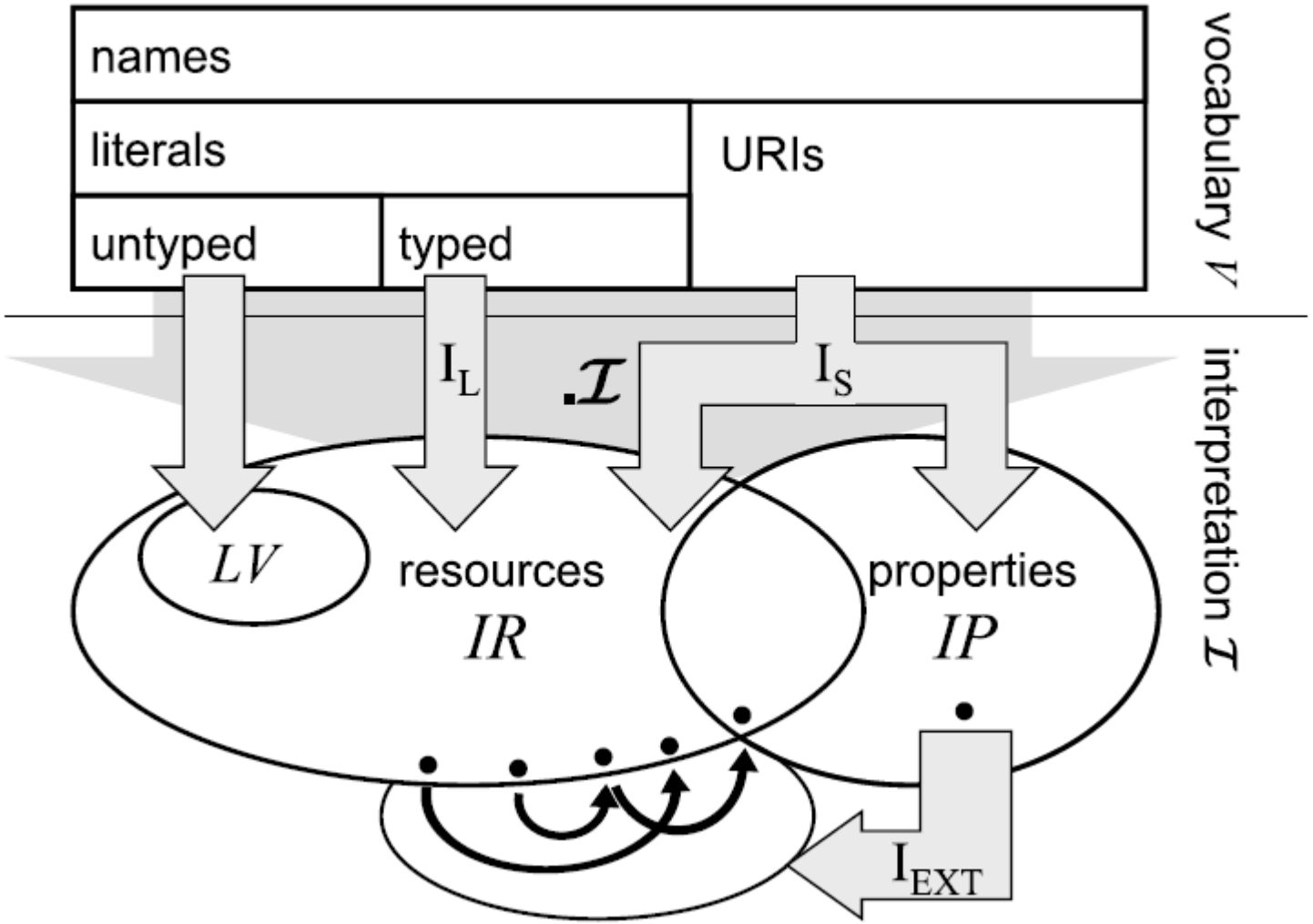
- IR, a non-empty set of *resources*, alternatively called domain or universe of discourse of $\mathcal{I}$,
- IP, the set of *properties* of $\mathcal{I}$ (which may overlap with IR),
- $\mathrm{I_{EXT}}$, a function assigning to each property a set of pairs from IR, i.e. $\mathrm{I_{EXT}} : IP \rightarrow 2^{IR \times IR}$, where $\mathrm{I_{EXT}}(p)$ is called the *extension* of the property p,
- $\mathrm{I_S}$, a function, mapping URIs from V into the union of the sets IR and IP, i.e. $\mathrm{I_S} : V \rightarrow IR \cup IP$,
- $\mathrm{I_L}$, a function from the typed literals from V into the set IR of resources and
- LV, a particular subset of IR, called the set of *literal values*, containing (at least) all untyped literals from V.

Simple Interpretations

Now define an interpretation function $\cdot^{\mathcal{I}}$ (written as exponent).

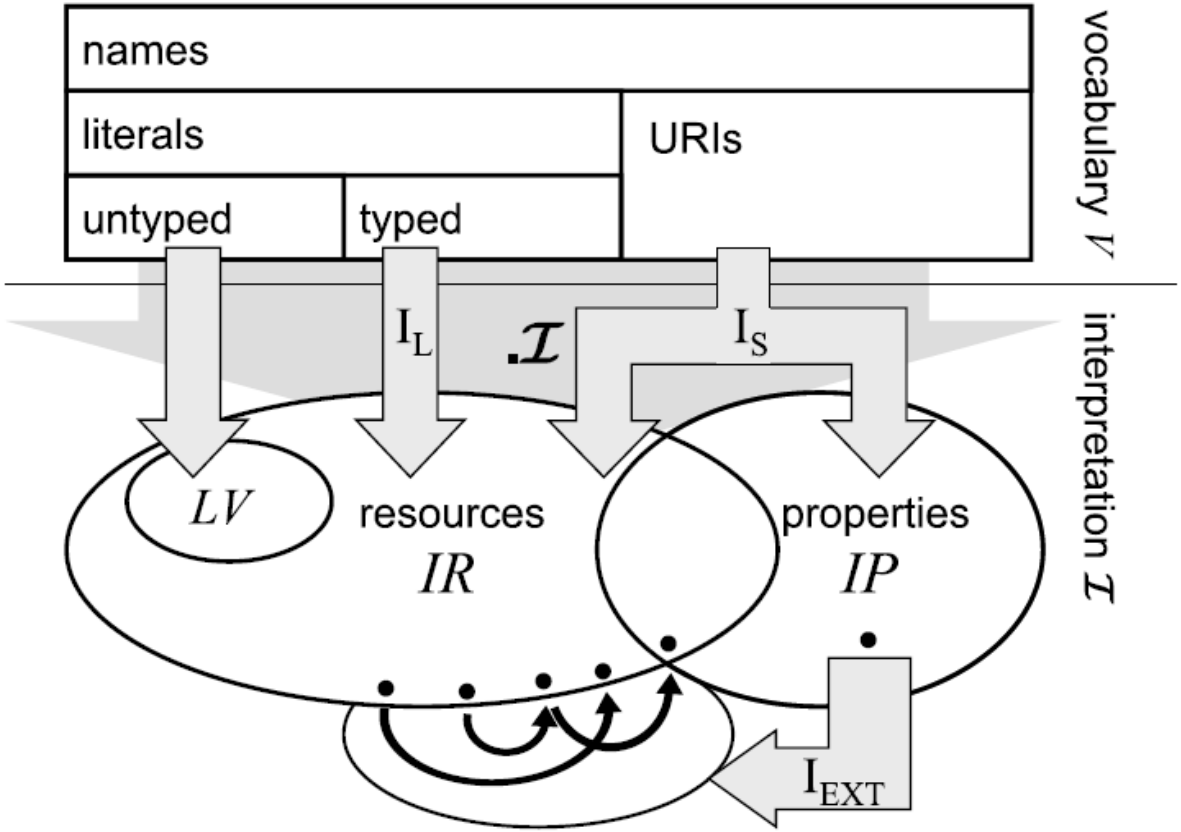
- every untyped literal $\texttt{"}a\texttt{"}$ is mapped to a, formally: $(\texttt{"}a\texttt{"})^{\mathcal{I}} = a$,
- every untyped literal carrying language information $\texttt{"}a\texttt{"@}t$ is mapped to the pair $\langle a, t \rangle$, i.e. $(\texttt{"}a\texttt{"@}t)^{\mathcal{I}} = \langle a, t \rangle$,
- every typed literal l is mapped to $\mathrm{I_L}(l)$, formally: $l^{\mathcal{I}} = \mathrm{I_L}(l)$, and
- every URI u is mapped to $\mathrm{I_S}(u)$, i.e. $u^{\mathcal{I}} = \mathrm{I_S}(u)$.

Simple Interpretations



Simple models

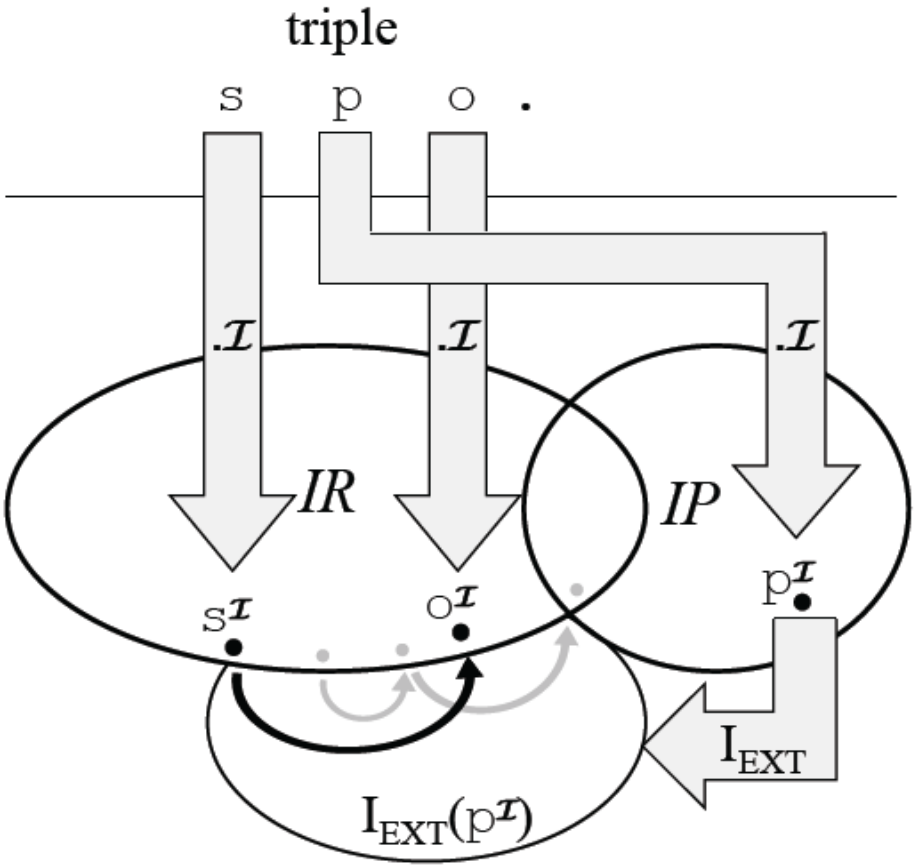
- The truth value $\mathtt{s\ p\ o.}^{\mathcal{I}}$ of a (grounded*) triple $\mathtt{s\ p\ o.}$ is true exactly if ($\mathtt{s}, \mathtt{p}, \mathtt{o}$ are contained in V) and $\langle \mathtt{s}^{\mathcal{I}}, \mathtt{o}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\mathtt{p}^{\mathcal{I}})$.



* A grounded triple does not contain a blank node.

Simple models

- The truth value $\mathtt{s\ p\ o.}^{\mathcal{I}}$ of a (grounded*) triple $\mathtt{s\ p\ o.}$ is true exactly if ($\mathtt{s}, \mathtt{p}, \mathtt{o}$ are contained in V) and $\langle \mathtt{s}^{\mathcal{I}}, \mathtt{o}^{\mathcal{I}} \rangle \in \mathrm{I_{EXT}}(\mathtt{p}^{\mathcal{I}})$.



* A grounded triple does not contain a blank node.

What about blank nodes?

- **Say, A is a function from blank nodes to URIs.**
  [these URIs need not be contained in the graph we're looking at]
- **If, in a graph G, we replace each blank node x by A(x), then we obtain a graph G' which we call a *grounding* of G.**
- **We know how to do the semantics for the grounded graphs.**
- **So define:**
  $I \models G$ if and only if $I \models G'$ for at least one grounding G' of G.

Simple entailment

- **A graph G *simply entails* a graph G' if every simple interpretation that is a model of G is also a model of G'.**

- **(Recall that a simple interpretation is a model of a graph G if it is a model of each triple in G.)**

It’s really simple

- **Basically, G ⊨ G’ if and only if G’ can be obtained from G by replacing some nodes in G by blank nodes.**

- **It’s really *simple* entailment.**

RDF-Interpretations Part 1

An *RDF-interpretation* of a vocabulary V is a simple interpretation of the vocabulary $V \cup V_{\mathrm{RDF}}$ that additionally satisfies the following conditions:

- $x \in IP$ exactly if $\langle x, \texttt{rdf:Property}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$.
- if "s"^^rdf:XMLLiteral is contained in V and s is a well-typed XML-Literal, then
  - I_{L}("s"^^rdf:XMLLiteral) is the XML value of s;
  - I_{L}("s"^^rdf:XMLLiteral) $\in LV$;
  - $\langle \mathrm{I}_{\mathrm{L}}(\text{"}s\text{"}\texttt{\^{}\^{}rdf:XMLLiteral}), \texttt{rdf:XMLLiteral}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$
- if "s"^^rdf:XMLLiteral is contained in V and s is an ill-typed XML literal, then
  - I_{L}("s"^^rdf:XMLLiteral) $\notin LV$ and
  - $\langle \mathrm{I}_{\mathrm{L}}(\text{"}s\text{"}\texttt{\^{}\^{}rdf:XMLLiteral}), \texttt{rdf:XMLLiteral}^{\mathcal{I}} \rangle \notin \mathrm{I}_{\mathrm{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$.

RDF-Interpretations Part 2

- **In addition, each RDF-interpretation has to evaluate all the following triples to true:**

rdf : type	rdf : type	rdf : Property.
rdf : subject	rdf : type	rdf : Property.
rdf : predicate	rdf : type	rdf : Property.
rdf : object	rdf : type	rdf : Property.
rdf : first	rdf : type	rdf : Property.
rdf : rest	rdf : type	rdf : Property.
rdf : value	rdf : type	rdf : Property.
rdf : $\_i$	rdf : type	rdf : Property.
rdf : nil	rdf : type	rdf : List.

RDFS-Interpretations Part 1

- **Define (for a given RDF-interpretation $\mathcal{I}$):**
  - $\mathrm{I}_{\mathrm{CEXT}} : IR \rightarrow 2^{IR}$: We define $\mathrm{I}_{\mathrm{CEXT}}(y)$ to contain exactly those elements x for which $\langle x, y \rangle$ is contained in $\mathrm{I}_{\mathrm{EXT}}(\texttt{rdf:type}^{\mathcal{I}})$. The set $\mathrm{I}_{\mathrm{CEXT}}(y)$ is then also called the *(class) extension* of y.
  - $IC = \mathrm{I}_{\mathrm{CEXT}}(\texttt{rdfs:Class}^{\mathcal{I}})$.
- $IR = \mathrm{I}_{\mathrm{CEXT}}(\texttt{rdfs:Resource}^{\mathcal{I}})$
- $LV = \mathrm{I}_{\mathrm{CEXT}}(\texttt{rdfs:Literal}^{\mathcal{I}})$
- If $\langle x, y \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:domain}^{\mathcal{I}})$ and $\langle u, v \rangle \in \mathrm{I}_{\mathrm{EXT}}(x)$, then $u \in \mathrm{I}_{\mathrm{CEXT}}(y)$.
- If $\langle x, y \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:range}^{\mathcal{I}})$ and $\langle u, v \rangle \in \mathrm{I}_{\mathrm{EXT}}(x)$, then $v \in \mathrm{I}_{\mathrm{CEXT}}(y)$.
- $\mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$ is reflexive and transitive on IP.

RDFS-Interpretation Part 2

- If $\langle x, y \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$,
  then $x, y \in IP$ and $\mathrm{I}_{\mathrm{EXT}}(x) \subseteq \mathrm{I}_{\mathrm{EXT}}(y)$.
- If $x \in IC$,
  then $\langle x, \texttt{rdfs:Resource}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$.
- If $\langle x, y \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$,
  then $x, y \in IC$ and $\mathrm{I}_{\mathrm{CEXT}}(x) \subseteq \mathrm{I}_{\mathrm{CEXT}}(y)$.
- $\mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$ is reflexive and transitive on IC.
- If $x \in \mathrm{I}_{\mathrm{CEXT}}(\texttt{rdfs:ContainerMembershipProperty}^{\mathcal{I}})$,
  then $\langle x, \texttt{rdfs:member}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subPropertyOf}^{\mathcal{I}})$.
- If $x \in \mathrm{I}_{\mathrm{CEXT}}(\texttt{rdfs:Datatype}^{\mathcal{I}})$,
  then $\langle x, \texttt{rdfs:Literal}^{\mathcal{I}} \rangle \in \mathrm{I}_{\mathrm{EXT}}(\texttt{rdfs:subClassOf}^{\mathcal{I}})$

RDFS-Interpretations Part 3

- **Furthermore, all of the following must be satisfied.**

```
rdf:type              rdfs:domain      rdfs:Resource .
rdfs:domain           rdfs:domain      rdf:Property .
rdfs:range            rdfs:domain      rdf:Property .
rdfs:subPropertyOf    rdfs:domain      rdf:Property .
rdfs:subClassOf       rdfs:domain      rdfs:Class .
rdf:subject           rdfs:domain      rdf:Statement .
rdf:predicate         rdfs:domain      rdf:Statement .
rdf:object            rdfs:domain      rdf:Statement .
rdfs:member           rdfs:domain      rdfs:Resource .
rdf:first             rdfs:domain      rdf:List .
rdf:rest              rdfs:domain      rdf:List .
rdfs:seeAlso          rdfs:domain      rdfs:Resource .
rdfs:isDefinedBy      rdfs:domain      rdfs:Resource .
```

RDFS-Interpretations Part 4

- **Furthermore, all of the following must be satisfied.**

```
rdfs:comment        rdfs:domain         rdfs:Resource .
rdfs:label          rdfs:domain         rdfs:Resource .
rdf:value           rdfs:domain         rdfs:Resource .

rdf:type            rdfs:range          rdfs:Class .
rdfs:domain         rdfs:range          rdfs:Class .
rdfs:range          rdfs:range          rdfs:Class .
rdfs:subPropertyOf  rdfs:range          rdf:Property .
rdfs:subClassOf     rdfs:range          rdfs:Class .
rdf:subject         rdfs:range          rdfs:Resource .
rdf:predicate       rdfs:range          rdfs:Resource .
rdf:object          rdfs:range          rdfs:Resource .
rdfs:member         rdfs:range          rdfs:Resource .
rdf:first           rdfs:range          rdfs:Resource .
rdf:rest            rdfs:range          rdf:List .
rdfs:seeAlso        rdfs:range          rdfs:Resource .
rdfs:isDefinedBy    rdfs:range          rdfs:Resource .
rdfs:comment        rdfs:range          rdfs:Literal .
rdfs:label          rdfs:range          rdfs:Literal .
rdf:value           rdfs:range          rdfs:Resource .
```

RDFS-Interpretations Part 5

- **Furthermore, all of the following must be satisfied.**

```
rdfs:ContainerMembershipProperty
                        rdfs:subClassOf     rdf:Property .
rdf:Alt                 rdfs:subClassOf     rdfs:Container .
rdf:Bag                 rdfs:subClassOf     rdfs:Container .
rdf:Seq                 rdfs:subClassOf     rdfs:Container .

rdfs:isDefinedBy        rdfs:subPropertyOf  rdfs:seeAlso .

rdf:XMLLiteral          rdf:type            rdfs:Datatype .
rdf:XMLLiteral          rdfs:subClassOf     rdfs:Literal .
rdfs:Datatype           rdfs:subClassOf     rdfs:Class .

rdf:_i                  rdf:type
                          rdfs:ContainerMembershipProperty .
rdf:_i                  rdfs:domain         rdfs:Resource .
rdf:_i                  rdfs:range          rdfs:Resource .
```

Today's Session: RDF(S) semantics

1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

Back to our contrived example

- Say, a model I of a set K of sentences consists of
  - a set C of cars and
  - a function I(·) which maps each variable to a car in C

  such that, for each sentence a η b in K we have that
  I(a) has more horsepower than I(b).

- **Can we find an algorithm to compute all logical consequences of a set of sentences?**
- **Algorithm Input: set K of sentences**
  1. **The algorithm non-deterministically selects two sentences from K. If the first sentence is a η b, and the second sentence is b η c, then add a η c to K.**

     IF a η b $\in$ K and b η c $\in$ K THEN K$\leftarrow$ {a η c}
  2. **Repeat step 1 until no selection results in a change of K.**
  3. **Output: K**

Back to the example

- **The algorithm produces only logical consequences: it is sound with respect to the model-theoretic semantics.**
- **The algorithm produces all logical consequences: it is complete with respect to the model-theoretic semantics.**
- **The algorithm always terminates.**

- **The algorithm is non-deterministic.**

- **What is the computational complexity of this algorithm?**

And actually, the algorithm just given is *not* sound and complete. Do you see, why?

What do we gain?

- **Recall:**
  - **β is a logical consequence of α (written $\alpha \models \beta$), if for all M with M$\models\alpha$, we also have M$\models\beta$ are**

- **Implementing model-theoretic semantics directly is not feasible: We would have to deal with *all* models of a knowledge base. Since there are a lot of cars in this world, we would have to check a lot of possibilities.**

- **Proof theory reduces model-theoretic semantics to symbol manipulation! It removes the models from the process.**

Deduction rules

IF $a \eta b \in K$ and $b \eta c \in K$ THEN $K \leftarrow \{a \eta c\}$

is a so-called *deduction rule*. Such rules are usually written schematically as

$$\frac{a \eta b \qquad b \eta c}{a \eta c}$$

Today's Session: RDF(S) semantics

1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

First, some notation

- a and b can refer to arbitrary URIs (i.e. anything admissible for the predicate position in a triple),
- $\_:n$ will be used for the ID of a blank node,
- u and v refer to arbitrary URIs or blank node IDs (i.e. any possible subject of a triple),
- x and y can be used for arbitrary URIs, blank node IDs or literals (i.e. anything admissible for the object position in a triple), and
- l may be any literal.

Simple Entailment Rules

$$\frac{u \quad a \quad x \quad .}{u \quad a \quad \_:n \quad .}\ \text{se1}$$

$$\frac{u \quad a \quad x \quad .}{\_:n \quad a \quad x \quad .}\ \text{se2}$$

_:n must not be contained in the graph the rule is applied to

Additional RDF-entailment Rules

$$\frac{}{u \;\; a \;\; x}\ \text{rdfax}$$

for all RDF axiomatic triples $u \;\; a \;\; x$.

$$\frac{u \;\; a \;\; l \;\; .}{u \;\; a \;\; \_:n \;\; .}\ \text{lg}$$

where _:n does not yet occur in the graph

$$\frac{u \;\; a \;\; y \;\; .}{a \;\; \texttt{rdf:type} \;\; \texttt{rdf:Property} \;\; .}\ \text{rdf1}$$

$$\frac{u \;\; a \;\; l \;\; .}{\_:n \;\; \texttt{rdf:type} \;\; \texttt{rdf:XMLLiteral}}\ \text{rdf2}$$

where _:n does not yet occur in the graph, unless it has been introduced by a preceding application of the lg rule

Additional RDFS-entailment Rules - I

$$\frac{}{u\ \ a\ \ x}\ \text{rdfax}$$

for all RDFS axiomatic triples $u\ \ a\ \ x\ .$

$$\frac{u\ \ a\ \ l\ .}{\_{:}n\ \ \texttt{rdf:type}\ \ \texttt{rdfs:Literal}\ .}\ \text{rdfs1}$$

with _:n as usual

$$\frac{a\ \ \texttt{rdfs:domain}\ \ x\ .\qquad u\ \ a\ \ y\ .}{u\ \ \texttt{rdf:type}\ \ x\ .}\ \text{rdfs2}$$

$$\frac{a\ \ \texttt{rdfs:range}\ \ x\ .\qquad u\ \ a\ \ v\ .}{v\ \ \texttt{rdf:type}\ \ x\ .}\ \text{rdfs3}$$

$$\frac{u\ \ a\ \ x\ .}{u\ \ \texttt{rdf:type}\ \ \texttt{rdfs:Resource}\ .}\ \text{rdfs4a}$$

$$\frac{u\ \ a\ \ v\ .}{v\ \ \texttt{rdf:type}\ \ \texttt{rdfs:Resource}\ .}\ \text{rdfs4b}$$

Additional RDFS-entailment Rules - II

$$\frac{u \texttt{ rdfs:subPropertyOf } v \texttt{ .} \qquad v \texttt{ rdfs:subPropertyOf } x \texttt{ .}}{u \texttt{ rdfs:subPropertyOf } x \texttt{ .}} \text{ rdfs5}$$

$$\frac{u \texttt{ rdf:type rdf:Property .}}{u \texttt{ rdfs:subPropertyOf } u \texttt{ .}} \text{ rdfs6}$$

$$\frac{a \texttt{ rdfs:subPropertyOf } b \texttt{ .} \qquad u \texttt{ } a \texttt{ } y \texttt{ .}}{u \texttt{ } b \texttt{ } y \texttt{ .}} \text{ rdfs7}$$

$$\frac{u \texttt{ rdf:type rdfs:Class .}}{u \texttt{ rdfs:subClassOf rdfs:Resource .}} \text{ rdfs8}$$

$$\frac{u \texttt{ rdfs:subClassOf } x \texttt{ .} \qquad v \texttt{ rdf:type } u \texttt{ .}}{v \texttt{ rdf:type } x \texttt{ .}} \text{ rdfs9}$$

$$\frac{u \texttt{ rdf:type rdfs:Class .}}{u \texttt{ rdfs:subClassOf } u \texttt{ .}} \text{ rdfs10}$$

Additional RDFS-entailment Rules - III

$$\frac{u \text{ rdfs:subClassOf } v \ . \qquad v \text{ rdfs:subClassOf } x \ .}{u \text{ rdfs:subClassOf } x \ .} \text{ rdfs11}$$

$$\frac{u \text{ rdf:type rdfs:ContainerMembershipProperty } .}{u \text{ rdfs:subPropertyOf rdfs:member } .} \text{ rdfs12}$$

$$\frac{u \text{ rdf:type rdfs:Datatype } .}{u \text{ rdfs:subClassOf rdfs:Literal } .} \text{ rdfs13}$$

$$\frac{u \ a \ \_:n \ .}{u \ a \ l \ .} \text{ gl}$$

where _:n identifies a blank node introduced by an earlier "weakening" of the literal l via the rule lg

Completeness?

- **The deduction rules for simple and RDF entailment are sound and complete.**

- **The deduction rules for RDFS entailment are sound.**

  The spec says, they are also complete, but they are not:

```
ex:isHappilyMarriedTo    rdfs:subPropertyOf       _:bnode .
_:bnode                  rdfs:domain              ex:Person .
ex:markus                ex:isHappilyMarriedTo    ex:anja .
```

  has as logical consequence

```
ex:markus                rdf:type                 ex:Person .
```

  but this is not derivable using the deduction rules.

Complexity

Simple, RDF, and RDFS entailment are NP-complete problems.

If we disallow blank nodes, all three entailment problems are polynomial.

Does RDFS semantics do what it should?

Does

```
ex:speaksWith   rdfs:domain      ex:Homo .
ex:Homo         rdfs:subClassOf  ex:Primates .
```

entail

```
ex:speaksWith   rdfs:domain      ex:Primates .
```

?

Today's Session: RDF(S) semantics

1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

Class project: next step

- keep bugfixing
- find, in your RDF Schema ontology, each of the following:
  - a triple which is RDFS-entailed, but not RDF-entailed
  - a triple which is RDF-entailed, but not simply entailed
  - a triple which is simply entailed
- For each of them, write down a justification why it is entailed.

- send to me by next Sunday
  - the current version of your Turtle RDF Schema document
  - the three entailed triples with explanations.

Today's Session: RDF(S) semantics

1. What is Semantics?
2. What is Model-theoretic Semantics?
3. Model-theoretic Semantics for RDF(S)
4. What is Proof-theoretic Semantics?
5. Proof-theoretic Semantics for RDF(S)
6. Class Project
7. Class Presentations

Class presentations – scheduled

- **RDFa – embedding RDF in HTML (W3C standard)**
  Pavan, Thursday 28th of January

- **Scalable Distributed Reasoning using MapReduce (Urbani, Kotoulas, Oren, van Harmelen, ISWC2009)**
  Wenbo, Thursday 28th of January

- **Semantic MediaWiki, Vinh, to be scheduled**
- **Linked Open Data, Ashutosh, to be scheduled**
- **FOAF, Hemant, to be scheduled**

Class presentations – open topics

Applications:

- **The SNOMED ontology (major biomedical ontology)**
- **Yahoo! Search Monkey (enhancing web search)**

Standards:

- **SKOS – data model for sharing and linking knowledge organization systems via the Web (W3C standard)**

Tools:

- **Protege – Ontology editing tool**
- **Jena – Java framework for Semantic Web by HP**
- **RDF triple stores (Virtuoso, Redland, Sesame, AllegroGraph)**

Class presentations – open topics

Research papers:

- **Parallel Materialization of the Finite RDFS Closure for Hundreds of Millions of Triples (Weaver, Hendler, ISWC2009)**
- **Umberto Straccia. A Minimal Deductive System for General Fuzzy RDF. In Proceedings of the 3rd International Conference on Web Reasoning and Rule Systems (RR-09), 2009**
- **Boris Motik. On the Properties of Metamodeling in OWL. In Yolanda Gil, Enrico Motta, Richard V. Benjamins, and Mark Musen, editors, Proc. of the 4th Int. Semantic Web Conference (ISWC 2005), volume 3729 of LNCS, pages 548–562, Galway, Ireland, November 6–10 2005. Springer.**

Class Planning

Thursday 28st of January: 2 class presentations

Tuesday 2nd of February: Exercise session

Thursday 4th of February

Estimated breakdown of sessions:

Intro + XML: 2

RDF: 3.3

OWL and Logic: 4.7

SPARQL and Querying: 2

Class Presentations: 3

Exercise sessions: 3