

INF3580/4580 – Semantic Technologies – Spring 2017

Lecture 8: RDF and RDFS semantics

Martin Giese

6th March 2017



Department of
Informatics



University of
Oslo

Oblig 5

- Published today
- First delivery due 21 March
- Final delivery due 11 April
- Extra question for INF4580 students
- “Real” semantics of RDF and RDFS
- Foundations book: Section 3.2
- Still OK to ignore some complications, see oblig text
- We provide an excerpt of Sect. 3.2 with unimportant parts removed.
- Go to group sessions!

Today's Plan

- 1 Why we need semantics
- 2 Model-theoretic semantics from a birds-eye perspective
- 3 Repetition: Propositional Logic
- 4 Simplified RDF semantics

Outline

- 1 Why we need semantics
- 2 Model-theoretic semantics from a birds-eye perspective
- 3 Repetition: Propositional Logic
- 4 Simplified RDF semantics

Semantics—why do we need it?

A formal semantics for RDFS became necessary because

- 1 the previous informal specification
- 2 left plenty of room for interpretation of conclusions, whence
- 3 triple stores sometimes answered queries differently, thereby
- 4 obstructing interoperability and interchangeability.
- 5 The information content of data once more came to depend on applications

But RDF was supposed to be the **data liberation movement**

Another look at the Semantic Web cake

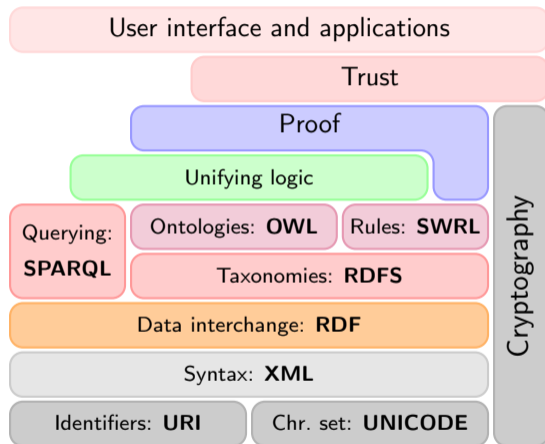


Figure: Semantic Web Stack

Absolute precision required

RDF is to serve as the foundation of the entire Semantic Web tower.

- It must therefore be sufficiently clear to sustain advanced reasoning, e.g.:
 - **type propagation/inheritance**,
 - “Tweety is a penguin and a penguin is a bird, so...”
 - **domain and range restrictions**,
 - “Martin has a birthdate, and only people have birthdates, so...”
 - **existential restrictions**.
 - “all persons have parents, and Martin is a person, so...”
- ...to which we shall return in later lectures

To ensure that infinitely many conclusions will be agreed upon,

- RDF must be furnished with a model-theory
- that specifies how the different node types should be interpreted
- and in particular what **entailment** should be taken to mean.

Example: What is the meaning of blank nodes?

Co-authors of Paul Erdős:

```
SELECT DISTINCT ?name WHERE {  
  _:pub dc:creator [foaf:name "Paul Erdős"] , [foaf:name ?name] .  
}
```

SPARQL must

- match the query to graph patterns
- which involves assigning values to variables and blank nodes

But,

- which values are to count?
- the problem becomes more acute under reasoning.
- Should a value for `foaf:familyname` match a query for `foaf:name`?
- Are blanks in SPARQL the same as blanks in RDF?

Outline

- 1 Why we need semantics
- 2 Model-theoretic semantics from a birds-eye perspective**
- 3 Repetition: Propositional Logic
- 4 Simplified RDF semantics

Formal semantics

- The study of how to model the meaning of a logical calculus.
- A logical calculus consists of:
 - A finite set of **symbols**,
 - a **grammar**, which specifies the formulae,
 - a set of **axioms** and **inference rules** from which we construct proofs.
- A logical calculus can be defined apart from any interpretation.
- A calculus that has not been furnished with a formal semantics,
 - is a 'blind' machine, a mere symbol manipulator,
 - the only criterion of correctness is **provability**.

Derivations

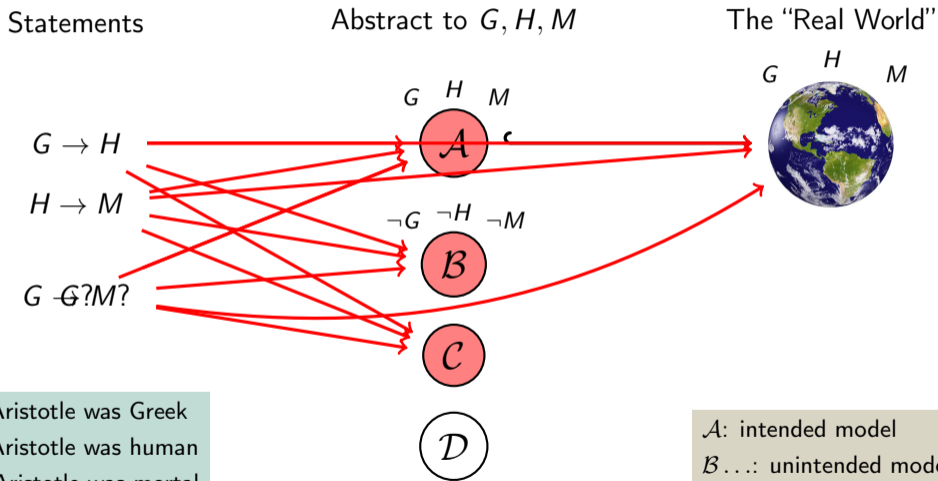
A proof typically looks something like this:

$$\frac{\frac{P \vdash Q, P \quad Q, P \vdash Q}{P \rightarrow Q, P \vdash Q} \quad \frac{R \vdash Q, P \quad Q, R \vdash Q}{P \rightarrow Q, R \vdash Q}}{P \rightarrow Q, P \vee R \vdash Q} \\
 \frac{\quad}{P \rightarrow Q \vdash (P \vee R) \rightarrow Q}$$

Where each line represents an application of an inference rule.

- How do we know that the inference rules are well-chosen?
- Which manipulations derive conclusions that hold **in the real world**?

Finding out stuff about the World



G : Aristotle was Greek
 H : Aristotle was human
 M : Aristotle was mortal

A : intended model
 $B \dots$: unintended models

Model-theoretic semantics

Basic idea: Asserting a sentence makes a claim about the world:

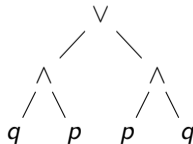
- A formula therefore limits the set of worlds that are possible.
- We can therefore encode meaning/logical content
 - by describing **models** of these worlds.
 - thus making *certain aspects* of meaning mathematically tractable
- The exact makeup of models varies from logic to logic, but they all
 - express a view on what kinds of things there are,
 - and the basic relations between these things
- By selecting a class of models one selects the basic features of the world
 - **as one chooses to see it.**
- Whatever these models all share can be said to be **entailed** by those features.

Outline

- 1 Why we need semantics
- 2 Model-theoretic semantics from a birds-eye perspective
- 3 Repetition: Propositional Logic**
- 4 Simplified RDF semantics

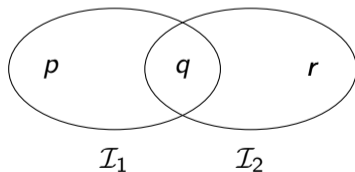
Propositional Logic: Formulas

- Formulas are defined “by induction” or “recursively”:
 - 1 Any letter p, q, r, \dots is a formula
 - 2 if A and B are formulas, then
 - $(A \wedge B)$ is also a formula (read: “ A and B ”)
 - $(A \vee B)$ is also a formula (read: “ A or B ”)
 - $\neg A$ is also a formula (read: “not A ”)
- Nothing else is. Only what rules [1] and [2] say is a formula.
- Examples of formulae: p $(p \wedge \neg r)$ $(q \wedge \neg q)$ $((p \vee \neg q) \wedge \neg p)$
- Formulas are just a kind of strings until now:
 - no meaning
 - but every formula can be “parsed” uniquely.

$$((q \wedge p) \vee (p \wedge q))$$


Interpretations

- Logic is about truth and falsity
- Truth of compound formulas depends on truth of letters.
- Idea: put all letters that are “true” into a set!
- Define: An *interpretation* \mathcal{I} is a set of letters.
- Letter p is true in interpretation \mathcal{I} if $p \in \mathcal{I}$.
- E.g., in $\mathcal{I}_1 = \{p, q\}$, p is true, but r is false.



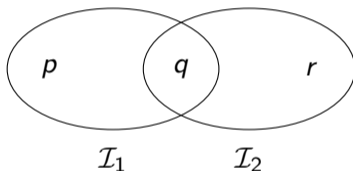
- But in $\mathcal{I}_2 = \{q, r\}$, p is false, but r is true.

Semantic Validity \models

- To say that p is true in \mathcal{I} , write

$$\mathcal{I} \models p$$

- For instance



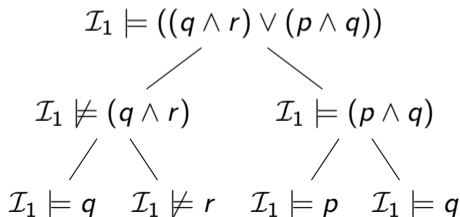
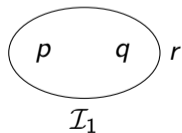
$$\mathcal{I}_1 \models p \quad \mathcal{I}_2 \not\models p$$

- In other words, for all letters p :

$$\mathcal{I} \models p \quad \text{if and only if} \quad p \in \mathcal{I}$$

Validity of Compound Formulas

- Is $((q \wedge r) \vee (p \wedge q))$ true in \mathcal{I} ?
- Idea: apply our rule recursively
- For any formulas A and B, \dots
- \dots and any interpretation \mathcal{I}, \dots
 - $\dots \mathcal{I} \models A \wedge B$ if and only if $\mathcal{I} \models A$ and $\mathcal{I} \models B$
 - $\dots \mathcal{I} \models A \vee B$ if and only if $\mathcal{I} \models A$ or $\mathcal{I} \models B$ (or both)
 - $\dots \mathcal{I} \models \neg A$ if and only if $\mathcal{I} \not\models A$.
- For instance



Truth Table

- Semantics of \neg , \wedge , \vee often given as *truth table*:

A	B	$\neg A$	$A \wedge B$	$A \vee B$
f	f	t	f	f
f	t	t	f	t
t	f	f	f	t
t	t	f	t	t

Tautologies

- A formula A that is true in *all* interpretations is called a *tautology*
- also *logically valid*
- also a *theorem* (of propositional logic)
- written:

$$\models A$$

- $(p \vee \neg p)$ is a tautology
- True whatever p means:
 - The sky is blue or the sky is not blue.
 - P.N. will win the 50km in 2016 or P.N. will not win the 50km in 2016.
 - The slithy toves gyre or the slithy toves do not gyre.
- Possible to derive true statements mechanically. . .
- . . . without understanding their meaning!
- . . . e.g. using truth tables for small cases.

Entailment

- Tautologies are true in all interpretations
- Some formulas are true only under certain assumptions
- A entails B , written $A \models B$ if

$$\mathcal{I} \models B$$

for all interpretations \mathcal{I} with $\mathcal{I} \models A$

- Also: “ B is a logical consequence of A ”
- Whenever A holds, also B holds
- For instance:

$$p \wedge q \models p$$

- Independent of meaning of p and q :
 - If it rains and the sky is blue, then it rains
 - If P.N. wins the race and the world ends, then P.N. wins the race
 - If 'tis brillig and the slythy toves do gyre, then 'tis brillig
- Also entailment can be checked mechanically, without knowing the meaning of words.

Question

Given the letters

P – Ola answers none of the questions correctly

Q – Ola fails the exam

Which of the following are tautologies of propositional logic?

- 1 Q
- 2 $\neg Q$
- 3 $P \rightarrow Q$
- 4 $Q \rightarrow (P \rightarrow Q)$

Outline

- 1 Why we need semantics
- 2 Model-theoretic semantics from a birds-eye perspective
- 3 Repetition: Propositional Logic
- 4 Simplified RDF semantics**

Taking the structure of triples into account

Unlike propositions, triples have parts, namely:

- subject
- predicates, and
- objects

Less abstractly, these may be:

- URI references
- literal values, and
- blank nodes

Triples are true or false **on the basis of what each part refers to.**

On what there is: Resources, Properties, Literals

The RDF data model consists of three object types; **resources**, **properties** and **literals values**:

Resources: All things described by RDF are called **resources**. Resources are identified by URIs

Properties: A **property** is a specific aspect, characteristic, attribute or relation used to describe a resource. Properties are also resources, and therefore identified by URIs.

Literals: A literal value is a concrete data item, such as an integer or a string.

String literals name themselves, i.e.

- “Julius Ceasar” names **the string** “Julius Ceasar”
- “42” names **the string** “42”

The semantics of typed and language tagged literals is considerably more complex.

Restricting RDF/RDFS

- We will simplify things by only looking at certain kinds of RDF graphs.
- No triples “about” properties, classes, etc., except RDFS
- Assume Resources are divided into four disjoint kinds:
 - *Properties* like foaf:knows, dc:title
 - *Classes* like foaf:Person
 - *Built-ins*, a fixed set including rdf:type, rdfs:domain, etc.
 - *Individuals* (all the rest, “usual” resources)
- All triples have one of the forms:


```
individual property individual .
individual rdf:type class .

class rdfs:subClassOf class .
property rdfs:subPropertyOf property .
property rdfs:domain class .
property rdfs:range class .
```
- Forget blank nodes and literals for a while!

Short Forms

- Resources and Triples are no longer all alike
- No need to use the same general triple notation
- Use alternative notation

Triples	Abbreviation
indi prop indi .	$r(i_1, i_2)$
indi rdf:type class .	$C(i_1)$
class rdfs:subClassOf class .	$C \sqsubseteq D$
prop rdfs:subPropOf prop .	$r \sqsubseteq s$
prop rdfs:domain class .	$\text{dom}(r, C)$
prop rdfs:range class .	$\text{rg}(r, C)$

- This is called “Description Logic” (DL) Syntax
- Used much in particular for OWL

Example

- Triples:

```

ws:romeo ws:loves ws:juliet .
ws:juliet rdf:type ws:Lady .

ws:Lady rdfs:subClassOf foaf:Person .
ws:loves rdfs:subPropertyOf foaf:knows .
ws:loves rdfs:domain ws:Lover .
ws:loves rdfs:range ws:Beloved .

```

- DL syntax, without namespaces:

loves(romeo, juliet)

Lady(juliet)

Lady \sqsubseteq *Person*

loves \sqsubseteq *knows*

dom(*loves*, *Lover*)

rg(*loves*, *Beloved*)



Interpretations for RDF

- To interpret propositional formulas, we need to know how to interpret
 - Letters
- To interpret the six kinds of triples, we need to know how to interpret
 - *Individual URIs* as real or imagined objects
 - *Class URIs* as sets of such objects
 - *Property URIs* as relations between these objects
- A *DL-interpretation* \mathcal{I} consists of
 - A set $\Delta^{\mathcal{I}}$, called the *domain* (sorry!) of \mathcal{I}
 - For each individual URI i , an element $i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
 - For each *class* URI C , a subset $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
 - For each *property* URI r , a relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
- Given these, it will be possible to say whether a triple holds or not.

An example “intended” interpretation

- $\Delta^{\mathcal{I}_1} = \left\{ \text{romeo}, \text{juliet}, \text{person} \right\}$
- $\text{romeo}^{\mathcal{I}_1} = \text{romeo_img}$ $\text{juliet}^{\mathcal{I}_1} = \text{juliet_img}$
- $\text{Lady}^{\mathcal{I}_1} = \left\{ \text{juliet_img} \right\}$ $\text{Person}^{\mathcal{I}_1} = \Delta^{\mathcal{I}_1}$
- $\text{Lover}^{\mathcal{I}_1} = \text{Beloved}^{\mathcal{I}_1} = \left\{ \text{romeo_img}, \text{juliet_img} \right\}$
- $\text{loves}^{\mathcal{I}_1} = \left\{ \left\langle \text{romeo_img}, \text{juliet_img} \right\rangle, \left\langle \text{juliet_img}, \text{romeo_img} \right\rangle \right\}$
- $\text{knows}^{\mathcal{I}_1} = \Delta^{\mathcal{I}_1} \times \Delta^{\mathcal{I}_1}$

An example “non-intended” interpretation

- $\Delta^{\mathcal{I}_2} = \mathbb{N} = \{1, 2, 3, 4, \dots\}$
- $romeo^{\mathcal{I}_2} = 17$
 $juliet^{\mathcal{I}_2} = 32$
- $Lady^{\mathcal{I}_2} = \{2^n \mid n \in \mathbb{N}\} = \{2, 4, 8, 16, 32, \dots\}$
 $Person^{\mathcal{I}_2} = \{2n \mid n \in \mathbb{N}\} = \{2, 4, 6, 8, 10, \dots\}$
 $Lover^{\mathcal{I}_2} = Beloved^{\mathcal{I}_2} = \mathbb{N}$
- $loves^{\mathcal{I}_2} = < = \{\langle x, y \rangle \mid x < y\}$
 $knows^{\mathcal{I}_2} = \leq = \{\langle x, y \rangle \mid x \leq y\}$
- Just because names (URIs) look familiar, they don't need to denote what we think!
- In fact, there is *no way* of ensuring they denote only what we think!

Validity in Interpretations (RDF)

- Given an interpretation \mathcal{I} , define \models as follows:

- $\mathcal{I} \models r(i_1, i_2)$ iff $\langle i_1^{\mathcal{I}}, i_2^{\mathcal{I}} \rangle \in r^{\mathcal{I}}$

- $\mathcal{I} \models C(i)$ iff $i^{\mathcal{I}} \in C^{\mathcal{I}}$

- Examples:

- $\mathcal{I}_1 \models \text{loves}(\text{juliet}, \text{romeo})$ because

$$\langle \text{img1}, \text{img2} \rangle \in \text{loves}^{\mathcal{I}_1} = \left\{ \langle \text{img2}, \text{img1} \rangle, \langle \text{img1}, \text{img2} \rangle \right\}$$

- $\mathcal{I}_1 \models \text{Person}(\text{romeo})$ because

$$\text{romeo}^{\mathcal{I}_1} = \text{img3} \in \text{Person}^{\mathcal{I}_1} = \Delta^{\mathcal{I}_1}$$

- $\mathcal{I}_2 \not\models \text{loves}(\text{juliet}, \text{romeo})$ because

$$\text{loves}^{\mathcal{I}_2} = < \text{ and } \text{juliet}^{\mathcal{I}_2} = 32 \not\prec \text{romeo}^{\mathcal{I}_2} = 17$$

- $\mathcal{I}_2 \not\models \text{Person}(\text{romeo})$ because

- $\text{romeo}^{\mathcal{I}_2} = 17 \notin \text{Person}^{\mathcal{I}_2} = \{2, 4, 6, 8, 10, \dots\}$

Validity in Interpretations, cont. (RDFS)

- Given an interpretation \mathcal{I} , define \models as follows:
- $\mathcal{I} \models C \sqsubseteq D$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I} \models r \sqsubseteq s$ iff $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
- $\mathcal{I} \models \text{dom}(r, C)$ iff for all $\langle x, y \rangle \in r^{\mathcal{I}}$, we have $x \in C^{\mathcal{I}}$
- $\mathcal{I} \models \text{rg}(r, C)$ iff for all $\langle x, y \rangle \in r^{\mathcal{I}}$, we have $y \in C^{\mathcal{I}}$
- Examples:

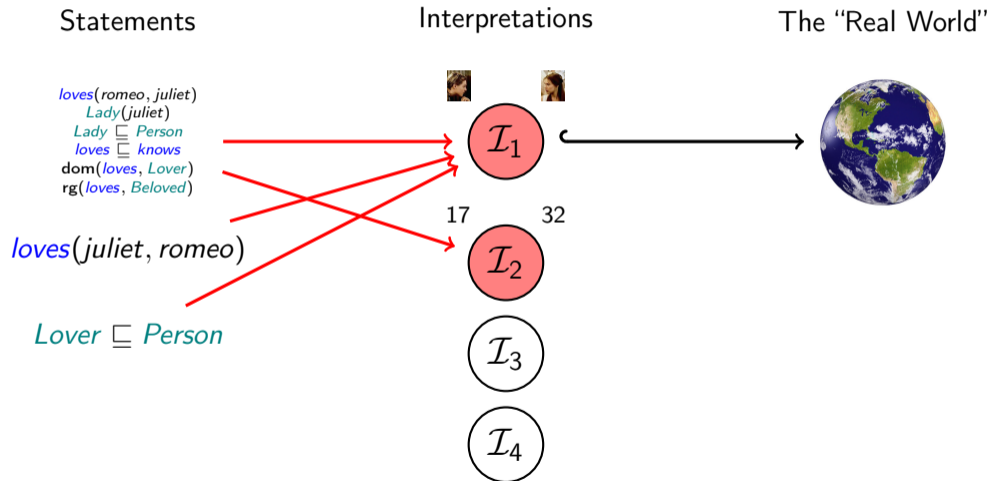
- $\mathcal{I}_1 \models \text{Lover} \sqsubseteq \text{Person}$ because

$$\text{Lover}^{\mathcal{I}_1} = \left\{ \text{[Levinson]}, \text{[Lara Croft]} \right\} \subseteq \text{Person}^{\mathcal{I}_1} = \left\{ \text{[Levinson]}, \text{[Lara Croft]}, \text{[Miles Teller]} \right\}$$

- $\mathcal{I}_2 \not\models \text{Lover} \sqsubseteq \text{Person}$ because

$$\text{Lover}^{\mathcal{I}_2} = \mathbb{N} \text{ and } \text{Person}^{\mathcal{I}_2} = \{2, 4, 6, 8, 10, \dots\}$$

Finding out stuff about Romeo and Juliet



Example: Range/Domain semantics

$$\mathcal{I}_2 \models \text{dom}(\textit{knows}, \textit{Beloved})$$

because...

$$\textit{knows}^{\mathcal{I}_2} = \leq = \{\langle x, y \rangle \mid x \leq y\}$$

$$\textit{Beloved}^{\mathcal{I}_2} = \mathbb{N}$$

and for any x and y with

$$\langle x, y \rangle \in \textit{knows}^{\mathcal{I}_2}, \quad \text{i.e.} \quad x \leq y,$$

we also have

$$x \in \mathbb{N} \quad \text{i.e.} \quad x \in \textit{Beloved}^{\mathcal{I}_2}$$

Interpretation of Sets of Triples

- Given an interpretation \mathcal{I}
- And a set of triples \mathcal{A} (any of the six kinds)
- \mathcal{A} is valid in \mathcal{I} , written

$$\mathcal{I} \models \mathcal{A}$$

- iff $\mathcal{I} \models A$ for all $A \in \mathcal{A}$.
- Then \mathcal{I} is also called a model of \mathcal{A} .
- Examples:

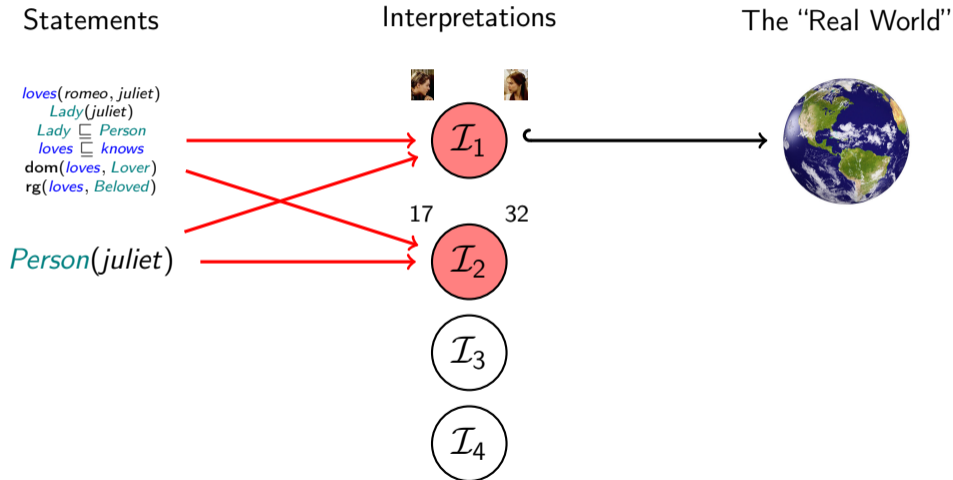
$$\mathcal{A} = \{ \text{loves}(\text{romeo}, \text{juliet}), \text{Lady}(\text{juliet}), \text{Lady} \sqsubseteq \text{Person}, \\ \text{loves} \sqsubseteq \text{knows}, \text{dom}(\text{loves}, \text{Lover}), \text{rg}(\text{loves}, \text{Beloved}) \}$$

- Then $\mathcal{I}_1 \models \mathcal{A}$ and $\mathcal{I}_2 \models \mathcal{A}$

Entailment

- Given a set of triples \mathcal{A} (any of the six kinds)
- And a further triple T (also any kind)
- T is entailed by \mathcal{A} , written $\mathcal{A} \models T$
- iff
 - For any interpretation \mathcal{I} with $\mathcal{I} \models \mathcal{A}$
 - $\mathcal{I} \models T$.
- $\mathcal{A} \models \mathcal{B}$ iff $\mathcal{I} \models \mathcal{B}$ for all \mathcal{I} with $\mathcal{I} \models \mathcal{A}$
- Example:
- $\mathcal{A} = \{\dots, \text{Lady}(\text{juliet}), \text{Lady} \sqsubseteq \text{Person}, \dots\}$ as before
- $\mathcal{A} \models \text{Person}(\text{juliet})$ because...
- in *any* interpretation $\mathcal{I} \dots$
- if $\text{juliet}^{\mathcal{I}} \in \text{Lady}^{\mathcal{I}}$ and $\text{Lady}^{\mathcal{I}} \subseteq \text{Person}^{\mathcal{I}} \dots$
- then by set theory $\text{juliet}^{\mathcal{I}} \in \text{Person}^{\mathcal{I}}$

Finding out stuff about Romeo and Juliet



Countermodels

- If $\mathcal{A} \not\models T, \dots$
- then there is an \mathcal{I} with
 - $\mathcal{I} \models \mathcal{A}$
 - $\mathcal{I} \not\models T$
- Vice-versa: if $\mathcal{I} \models \mathcal{A}$ and $\mathcal{I} \not\models T$, then $\mathcal{A} \not\models T$
- Such an \mathcal{I} is called a *counter-model* (for the assumption that \mathcal{A} entails T)
- To show that $\mathcal{A} \models T$ does *not* hold:
 - Describe an interpretation \mathcal{I} (using your fantasy)
 - Prove that $\mathcal{I} \models \mathcal{A}$ (using the semantics)
 - Prove that $\mathcal{I} \not\models T$ (using the semantics)

Countermodel Example

- \mathcal{A} as before:

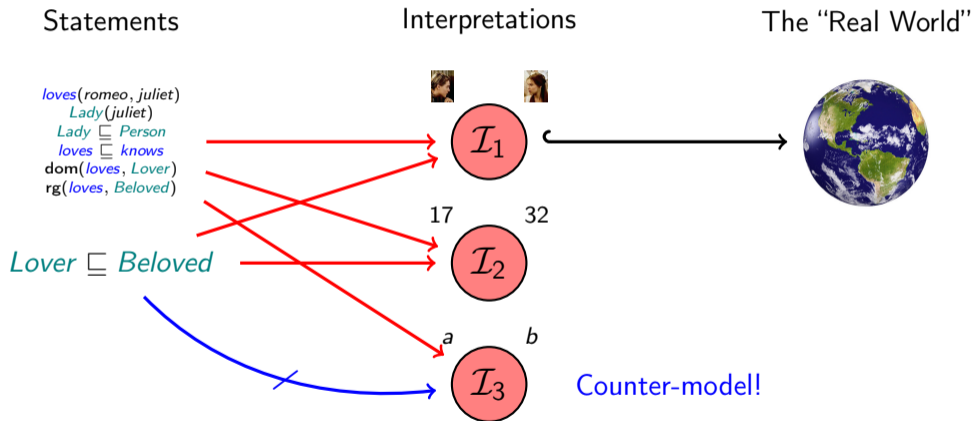
$$\mathcal{A} = \{ \text{loves}(\text{romeo}, \text{juliet}), \text{Lady}(\text{juliet}), \text{Lady} \sqsubseteq \text{Person}, \\ \text{loves} \sqsubseteq \text{knows}, \text{dom}(\text{loves}, \text{Lover}), \text{rg}(\text{loves}, \text{Beloved}) \}$$

- Does $\mathcal{A} \models \text{Lover} \sqsubseteq \text{Beloved}$?
- Holds in \mathcal{I}_1 and \mathcal{I}_2 .
- Try to find an interpretation with $\Delta^{\mathcal{I}} = \{a, b\}$, $a \neq b$.
- Interpret $\text{romeo}^{\mathcal{I}} = a$ and $\text{juliet}^{\mathcal{I}} = b$
- Then $\langle a, b \rangle \in \text{loves}^{\mathcal{I}}$, $a \in \text{Lover}^{\mathcal{I}}$, $b \in \text{Beloved}^{\mathcal{I}}$.
- With $\text{Lover}^{\mathcal{I}} = \{a\}$ and $\text{Beloved}^{\mathcal{I}} = \{b\}$, $\mathcal{I} \not\models \text{Lover} \sqsubseteq \text{Beloved}$!
- Choose

$$\text{loves}^{\mathcal{I}} = \text{knows}^{\mathcal{I}} = \{ \langle a, b \rangle \} \quad \text{Lady}^{\mathcal{I}} = \text{Person}^{\mathcal{I}} = \{ b \}$$

to complete the counter-model while satisfying $\mathcal{I} \models \mathcal{A}$

Countermodels about Romeo and Juliet



Take aways

- 1 Model-theoretic semantics yields an unambiguous notion of entailment,
- 2 which is necessary in order to liberate data from applications.
- 3 Shown today: A simplified semantics for parts of RDF
 - 1 Only RDF/RDFS vocabulary to talk “about” predicates and classes
 - 2 Literals and blank nodes next time

Supplementary reading on RDF and RDFS semantics:

- <http://www.w3.org/TR/rdf-mt/>
- Section 3.2 in Foundations of SW Technologies