Information Integration on the WEB with RDF, OWL and SPARQL

Review Material: *First Order Logic* (FOL)

Grant Weddell

October 7, 2013
Syntax of FOL
Signatures

Vocabularies are called signatures in FOL.

The non-logical parameters in FOL consist of infinite disjoint collections \( \{ P_1, P_2, \ldots \} \) and \( \{ f_1, f_2, \ldots \} \) of predicate symbols and function symbols, respectively.

The arity of each symbol is a non-negative integer \( n \), denoted \( \text{Ar}(P_i) \) or \( \text{Ar}(f_i) \).

- \( P/i \) denotes predicate symbol \( P \) where \( \text{Ar}(P) = i \).
- \( f/j \) denotes function symbol \( f \) where \( \text{Ar}(f) = j \).

Predicate symbols of arity 0 are also called propositions.

Function symbols of arity 0 are also called constants.

A signature \( S \) in FOL is a possibly infinite selection of non-logical parameters.

- \( S^P \) denotes all predicate symbols in \( S \).
- \( S^F \) denotes all function symbols in \( S \).
There is a rough dichotomy of information into what are commonly termed *data* and *metadata*.

**Example PAYROLL data important to ACME.**

1. Mary is an employee.
2. Mary’s employee number is 3412.
3. Mary’s salary is 72000.

**PAYROLL metadata specified by ACME.**

4. There is a kind of entity called an employee.
5. There are attributes called employee-number, name and salary.
6. Each employee entity has attributes employee-number, name and salary.
7. Employees are identified by their employee-number.
ACME Case: Signatures for PAYROLL

**Option 1**

- $S^P = \{\text{employee}/3\}$
- $S^F = \emptyset$

Fewest non-logical parameters: a single 3-ary predicate symbol.

- 1st arg: an employee number
- 2nd arg: an employee name
- 3rd arg: an employee salary

1st arg serves a special role: the set of employee number values is identified with the set of employees.

Each 3-tuple in $(\text{employee})^I$ suggests two things.

1. The employee number is a *visible object identifier* of some employee.
2. The remaining two components of the 3-tuple express two facts about the employee.
Signatures for PAYROLL

**Option 2**

- \( S^P = \{\text{employee/1}\} \)
- \( S^F = \{\text{employee-number/1, name/1, salary/1}\} \)

Trades the need to remember the role of argument positions with the need to learn and remember additional non-logical parameters.

Introduce
- unary predicates to capture the various kinds of entities, and
- unary functions to capture entity attributes.

Advantages:
- Separates entity classification from entity description: an entity \( e \) in a given interpretation \( \mathcal{I} \) is an employee exactly when \( e \in (\text{employee})^\mathcal{I} \).
- All information about entities, such as a name or salary, is captured by unary functions.
Option 1 versus Option 2

Latter allows the possibility that more than one employee can have the same combination of values for attributes employee-number, name and salary.

Replacing an $n$-ary predicate symbol with one unary predicate symbol and $n$ unary function symbols is called reification.

Disadvantages:

- Requires all entities to have a value for all attributes.
- Therefore requires simulating partial functions (e.g., “null inapplicable” values).
Signatures for PAYROLL

Option 3

- $S^P = \{\text{employee}/1, \text{employee-number}/2, \text{name}/2, \text{salary}/2\}$
- $S^F = \emptyset$

Overcomes disadvantages of Option 2: replaces each unary function symbol with a new binary predicate symbol.

Makes it possible for an entity (including employees) to have any number of employee numbers, names or salaries, including none.

Replacing function symbols with new predicate symbols is always possible when a function free signature is desired.
Denoted $V$, the *variables* in FOL are a countably infinite collection of symbols

$$\{x_1, x_2, \ldots\}$$

disjoint from the set of non-logical parameters.

Assume $S$ denotes an FOL signature.

The following grammars define the *terms*, *atoms* and *well formed formulae* induced by $S$, denoted $\text{TERM}(S)$, $\text{ATOM}(S)$ and $\text{WFF}(S)$, respectively.

- $\text{Term} ::= x \text{  (where } x \in V) \mid f(\text{Term}_1, \ldots, \text{Term}_n) \text{  (where } f/n \in S^F)$
- $\text{Atom} ::= \text{Term}_1 \approx \text{Term}_2 \mid P(\text{Term}_1, \ldots, \text{Term}_n) \text{  (where } P/n \in S^P)$
- $\phi, \psi ::= \text{Atom} \mid \neg \phi \mid (\phi \land \psi) \mid \exists x. \phi \text{  (where } x \in V)$

Omit mention of $S$ when clear from context.

Use $\phi$ and $\psi$ (possibly subscripted) to refer to elements of $\text{WFF}$. 

The *logical parameters* in FOL:

- *(equality)* \( \approx \)
- *(negation)* \( \neg \)
- *(conjunction)* \( \land \)
- *(existential quantification)* \( \exists \)

Convenient to have additional logical parameters as syntactic shorthand:

- *(disjunction)* \( \lor \): "\((\phi \lor \psi)\)" \(\leadsto\) "\(\neg(\neg\phi \land \neg\psi)\)"
- *(implication)* \( \rightarrow \): "\((\phi \rightarrow \psi)\)" \(\leadsto\) "\(\neg\phi \lor \psi\)"
- *(equivalence)* \( \equiv \): "\((\phi \equiv \psi)\)" \(\leadsto\) "\(((\phi \rightarrow \psi) \land (\psi \rightarrow \phi))\)"
- *(universal quantification)* \( \forall \): "\(\forall x.\phi\)" \(\leadsto\) "\(\neg\exists x.\neg\phi\)."

Also common practice to omit parenthesis in well-formed formulae when intentions are clear, e.g.:

"\((\phi_1 \land \phi_2 \land \phi_3)\)" instead of "\((\phi_1 \land (\phi_2 \land \phi_3))\)"
Assume $S$ denotes an FOL signature, and recall TERM and WFF denote the terms and well-formed formulae induced by $S$.

Given $t \in \text{TERM}$ or $\phi \in \text{WFF}$: $\text{Fv}(t)$ and $\text{Fv}(\phi)$ denote the free variables of a term and of a well formed formula, respectively.

Inductively defined as follows:

\[
\text{Fv}(t) = \begin{cases} 
\{x\} & \text{if } t = "x", \text{ and} \\
\bigcup_{1 \leq i \leq n} \text{Fv}(t_i) & \text{when } t = "f(t_1, \ldots, t_n)" \text{ otherwise.} 
\end{cases}
\]

\[
\text{Fv}(\phi) = \begin{cases} 
\bigcup_{1 \leq i \leq n} \text{Fv}(t_i) & \text{if } \phi = "P(t_1, \ldots, t_n)" , \\
\text{Fv}(t_1) \cup \text{Fv}(t_2) & \text{if } \phi = "t_1 \approx t_2" , \\
\text{Fv}(\psi) & \text{if } \phi = "\neg \psi" , \\
\text{Fv}(\psi_1) \cup \text{Fv}(\psi_2) & \text{if } \phi = "(\psi_1 \land \psi_2)" , \text{ and} \\
\text{Fv}(\psi) - \{x\} & \text{when } \phi = "\exists x.\psi" \text{ otherwise.}
\end{cases}
\]
A well-formed formula $\phi$ is *closed* if $\text{Fv}(\phi) = \emptyset$. A closed well-formed formula is also called a *sentence*.

A *theory* over signature $S$ in FOL, written $\Sigma(S)$, is a (possibly infinite) subset of $\text{WFF}(S)$.

Usually omit mention of $S$ when the signature is clear from context, or just describe a theory assuming the signature consists of all non-logical parameters mentioned in the theory.
Assume $S$ denotes a signature in FOL.

An *interpretation* $\mathcal{I}(S)$ of $S$ is a pair $\langle \Delta^{\mathcal{I}(S)}, (\cdot)^{\mathcal{I}(S)} \rangle$.

1. $\Delta^{\mathcal{I}(S)}$ is a non-empty *domain* of entities.
2. $(\cdot)^{\mathcal{I}(S)}$ is an *interpretation function*.

For each $P/n \in S^P$, $(P/n)^{\mathcal{I}(S)}$ is a subset of $(\Delta^{\mathcal{I}(S)})^n$.

For each $f/n \in S^F$, $(f/n)^{\mathcal{I}(S)}$ is a total function: $(\Delta^{\mathcal{I}(S)})^n \rightarrow \Delta^{\mathcal{I}(S)}$.

Write $\langle e_1, \ldots, e_n \rangle$ to denote an *$n$-tuple*, an element of $(\Delta^{\mathcal{I}(S)})^n$.

Identity $e$ and $\langle e \rangle$. Thus: $(P)^{\mathcal{I}(S)} \subseteq \Delta^{\mathcal{I}(S)}$, for any unary predicate symbol $P$.

Again omit mention of $S$ when the signature is clear from context.
Assume $\mathcal{I}$ is an interpretation of signature $S$.

A valuation over $\mathcal{I}$ is written $\mathcal{V}(\mathcal{I})$ (or as $\mathcal{V}$ when $\mathcal{I}$ is clear from context) and is a total function: $\mathcal{V} \to \Delta^\mathcal{I}$.

For a given $x \in V$ and $e \in \Delta^\mathcal{I}$, the valuation $\mathcal{V}[x \mapsto e]$ is defined as follows:

$$\mathcal{V}[x_1 \mapsto e](x_2) = \begin{cases} e & \text{if } "x_1" = "x_2" , \text{ and} \\ \mathcal{V}(x_2) & \text{otherwise.} \end{cases}$$

A valuation $\mathcal{V}$ is extended to apply to any $t \in \text{TERM}$ in the way that satisfies

$$\mathcal{V}(t) = (f)^\mathcal{I}(\mathcal{V}(t_1), \ldots, \mathcal{V}(t_n))$$

whenever $t = "f(t_1, \ldots, t_n)"$. 
Assume $S$ is a signature in FOL and also that $\phi \in \text{WFF}(S)$.

An interpretation $I$ of $S$ and valuation $V$ over $I$ is a *model* of $\phi$, written

$$I, V \models \phi,$$

iff one of the following conditions apply:

- $\phi = "P(t_1, \ldots, t_n)"$ and $\langle V(t_1), \ldots, V(t_n) \rangle \in (P)^I$,
- $\phi = "t_1 \approx t_2"$ and $V(t_1) = V(t_2)$,
- $\phi = "\neg \psi"$ and $I, V \not\models \psi$,
- $\phi = "(\psi_1 \land \psi_2)"$, $I, V \models \psi_1$ and $I, V \models \psi_2$, or
- $\phi = "\exists x. \psi"$ and $I, V[x \mapsto e] \models \psi$ for some $e \in \Delta^I$. 
Assume $\Sigma$ is a theory (over signature $S$).

We say the following.

1. The pair $I, V$ is a model of $\Sigma$ if $I, V \models \psi$ for all $\psi \in \Sigma$.
2. $\Sigma$ is satisfiable if it has a model and unsatisfiable otherwise.
3. $\phi$ is a logical consequence of $\Sigma$, written
   \[ \Sigma \models \phi, \]
   iff $I, V \models \phi$ for any model $I, V$ of $\Sigma$.

The fundamental problem of reasoning in a given FOL theory $\Sigma(S)$ is the problem of logical implication: establishing which $\phi \in \text{WFF}(S)$ are logical consequences of $\Sigma(S)$. 
On identification.

Assume S is given by Option 1.

The condition that employees can be identified by their employee number can be expressed as the FOL sentence

$$\forall x_1, x_2, y_1, y_2. (\exists z. \text{employee}(z, x_1, x_2) \land \text{employee}(z, y_1, y_2)) \rightarrow ((x_1 \approx y_1) \land (x_2 \approx y_2)).$$

Ensures that each employee is associated with a single 3-tuple in $(\text{employee})^\mathcal{I}$ in any interpretation $\mathcal{I}$ for ACME’s PAYROLL system.$^1$

Called a functional dependency in relational schema.

$^1$Remember introductory comments: the collection of all data corresponds to an interpretation $\mathcal{I}$. 
For S given by **Option 2**:

\[ \forall x, y. ( \neg \text{employee}(x) \lor \neg \text{employee}(y) \lor \text{employee}-\text{number}(x) \approx \text{employee}-\text{number}(y) \rightarrow x \approx y). \]

For S given by **Option 3**:

\[ \forall x, y. ( \exists z. (\text{employee}(x) \land \text{employee}(y) \land \text{employee}-\text{number}(x, z) \land \text{employee}-\text{number}(y, z) \rightarrow x \approx y). \]

More accurately, latter states that: *no pair of distinct employees may have any employee number at all in common* (becomes possible in **Option 3** for employees to have any number of employee numbers).
Logical Constraints for PAYROLL

On property functionality.

Assume Option 3 chosen by ACME’s APS department.

Then necessary to disallow the number of possible values for an employee-number, name, or salary attribute for a given employee to exceed one.

Must add constraints to the logical constraints $\Sigma$ to ensure the attributes are partial functions.

\[
\forall x, y. (\exists z. (\text{employee-number}(z, x) \land \text{employee-number}(z, y)) \rightarrow (x \approx y))
\]

\[
\forall x, y. (\exists z. (\text{name}(z, x) \land \text{name}(z, y)) \rightarrow (x \approx y))
\]

\[
\forall x, y. (\exists z. (\text{salary}(z, x) \land \text{salary}(z, y)) \rightarrow (x \approx y))
\]
Assume $\phi$ is a well formed formula of the form

$$(\psi_1 \land \cdots \land \psi_m)$$

in which each $\psi_i$ is a function free atom.

An equality generating dependency (EGD) is a sentence in FOL with the form

$$\forall x_1, x_2. (\exists x_3, \cdots, x_n. \phi \rightarrow (x_1 \approx x_2))$$

where $n \geq 2$ and where $\text{Fv}(\phi) = \{x_1, \ldots, x_n\}$. 
On taxonomic knowledge.

A simple PAYROLL taxonomy.

```
<table>
<thead>
<tr>
<th>employee</th>
<th>entity</th>
<th>object</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>value</td>
<td>integer</td>
</tr>
</tbody>
</table>
```

“⟨id⟩₁ → ⟨id⟩₂” asserts: “set of all ⟨id⟩₁” ⊆ “set of all ⟨id⟩₂”.

More relevant to Option 2 or Option 3 for signature S of PAYROLL system.
Add unary predicates entity, object, value, string and integer to $S^p$.

Add following logical constraints to $\Sigma$.

1. Everything is an entity.
$$\forall x. \text{entity}(x)$$

2. More specific entities are objects and values.
$$\forall x. (\text{object}(x) \rightarrow \text{entity}(x))$$
$$\forall x. (\text{value}(x) \rightarrow \text{entity}(x))$$

3. More specific objects are employees.
$$\forall x. (\text{employee}(x) \rightarrow \text{object}(x))$$
Logical Constraints for PAYROLL

4. More specific values are strings and integers.

\[ \forall x. (\text{string}(x) \rightarrow \text{value}(x)) \]

\[ \forall x. (\text{integer}(x) \rightarrow \text{value}(x)) \]

5. Nothing is just an entity or a value.

\[ \forall x. (\text{entity}(x) \rightarrow (\text{object}(x) \lor \text{value}(x))) \]

\[ \forall x. (\text{value}(x) \rightarrow (\text{string}(x) \lor \text{integer}(x))) \]

6. Objects are distinct from values, and strings from integers.

\[ \forall x. (\text{object}(x) \rightarrow \neg \text{value}(x)) \]

\[ \forall x. (\text{string}(x) \rightarrow \neg \text{integer}(x)) \]
Logical Constraints for PAYROLL

On typing.

The additional unary predicates in the PAYROLL signature can be used to ensure that attribute values are of appropriate types.

For S given by OPTION 1:

\[ \forall x, y, z. (\text{employee}(x, y, z) \rightarrow (\text{integer}(x) \land \text{string}(y) \land \text{integer}(z))) \]

For S given by OPTION 2:

\[ \forall x. (\text{employee}(x) \rightarrow (\text{integer}(\text{employee-number}(x)) \land \text{string}(\text{name}(x)) \land \text{integer}(\text{salary}(x)))) \]
For \( S \) given by \textbf{Option 3}:

\[
\forall x. (\text{employee}(x) \rightarrow \exists y, z, w. (\text{employee-number}(x, y) \land \text{integer}(y) \\
\land \text{name}(x, z) \land \text{string}(z)) \\
\land \text{salary}(x, w) \land \text{integer}(w)))
\]

\textbf{Option 3} also makes it possible to say that \textit{only employees have employee numbers}.

\[
\forall x. (\exists y. \text{employee-number}(x, y) \rightarrow \text{employee}(x))
\]
Assume $\phi$ and $\psi$ denote well formed formulae with the respective forms $(\phi_1 \land \cdots \land \phi_m)$ and $(\psi_1 \land \cdots \land \psi_n)$ in which each $\phi_i$ and $\psi_i$ is a function free atom.

A tuple generating dependency (TGD) is a sentence in FOL with the form

$$\forall x_1, \ldots, x_i. (\exists y_1, \ldots, y_j. \phi \rightarrow \exists z_1, \ldots, z_k. \psi)$$

where $\text{Fv}(\phi)$ and $\text{Fv}(\psi)$ are the respective sets \{x_1, \ldots, x_i\} $\cup$ \{y_1, \ldots, y_j\} and \{x_1, \ldots, x_i\} $\cup$ \{z_1, \ldots, z_k\}.

A full TGD also satisfies $k = 0$ (i.e., $\text{Fv}(\psi) = \{x_1, \ldots, x_i\}$).