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 $\mathrm{Ar}(P_i)$ or $\mathrm{Ar}(f_i)$.

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

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.

Case of the ACME PAYROLL System: Information

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

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

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 $(\texttt{employee})^{\mathcal{I}}$ 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

- $\mathsf{S}^{\mathsf{P}} = \{\texttt{employee}/1\}$
- $\mathsf{S}^{\mathsf{F}} = \{\texttt{employee-number}/1, \texttt{name}/1, \texttt{salary}/1\}$

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 $e \in (\texttt{employee})^{\mathcal{I}}$.
- 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

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

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.

Variables and Well-Formed Formulae

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 TERM(S), ATOM(S) and WFF(S), respectively.

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

Omit mention of S when clear from context.

Use ϕ and ψ (possibly subscripted) to refer to elements of WFF.

Variables and Well-Formed Formulae (cont'd)

The *logical parameters* in FOL:

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

Convenient to have additional logical parameters as syntactic shorthand:

- (*disjunction*) $\vee$: "$(\phi \vee \psi)$" $\rightsquigarrow$ "$\neg(\neg\phi \wedge \neg\psi)$"
- (*implication*) $\rightarrow$: "$(\phi \rightarrow \psi)$" $\rightsquigarrow$ "$(\neg\phi \vee \psi)$"
- (*equivalence*) $\equiv$: "$(\phi \equiv \psi)$" $\rightsquigarrow$ "$((\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi))$"
- (*universal quantification*) $\forall$: "$\forall x.\phi$" $\rightsquigarrow$ "$\neg\exists x.\neg\phi$".

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

"$(\phi_1 \wedge \phi_2 \wedge \phi_3)$" instead of "$(\phi_1 \wedge (\phi_2 \wedge \phi_3))$"

Free Variables

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:

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

$$\text{Fv}(\phi) = \begin{cases} \bigcup_{1 \leq i \leq n} \text{Fv}(t_i) & \text{if } \phi = \text{“}P(t_1, \ldots, t_n)\text{”,} \\ \text{Fv}(t_1) \cup \text{Fv}(t_2) & \text{if } \phi = \text{“}t_1 \approx t_2\text{”,} \\ \text{Fv}(\psi) & \text{if } \phi = \text{“}\neg \psi\text{”,} \\ \text{Fv}(\psi_1) \cup \text{Fv}(\psi_2) & \text{if } \phi = \text{“}(\psi_1 \wedge \psi_2)\text{”, and} \\ \text{Fv}(\psi) - \{x\} & \text{when } \phi = \text{“}\exists x.\psi\text{” otherwise.} \end{cases}$$

Sentences and Theories

A well-formed formula ϕ is *closed* if $\mathrm{Fv}(\phi) = \emptyset$. A closed well-formed formula is also called a *sentence*.

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

Semantics of FOL
Interpretations

Assume S denotes a signature in FOL.

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

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

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

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

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

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

Again omit mention of S when the signature is clear from context.

Valuations

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: $\mathsf{V} \to \triangle^{\mathcal{I}}$.

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

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

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

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

whenever $t = \text{“}f(t_1, \ldots, t_n)\text{”}$.

Models

Assume S is a signature in FOL and also that $\phi \in \text{WFF(S)}$.

An interpretation $\mathcal{I}$ of S and valuation $\mathcal{V}$ over $\mathcal{I}$ is a *model* of ϕ, written

$$\mathcal{I}, \mathcal{V} \models \phi,$$

iff one of the following conditions apply:

- $\phi =$ "$P(t_1, \ldots, t_n)$" and $\langle \mathcal{V}(t_1), \ldots, \mathcal{V}(t_n) \rangle \in (P)^{\mathcal{I}}$,
- $\phi =$ "$t_1 \approx t_2$" and $\mathcal{V}(t_1) = \mathcal{V}(t_2)$,
- $\phi =$ "$\neg\psi$" and $\mathcal{I}, \mathcal{V} \not\models \psi$,
- $\phi =$ "$(\psi_1 \wedge \psi_2)$", $\mathcal{I}, \mathcal{V} \models \psi_1$ and $\mathcal{I}, \mathcal{V} \models \psi_2$, or
- $\phi =$ "$\exists x.\psi$" and $\mathcal{I}, \mathcal{V}[x \mapsto e] \models \psi$ for some $e \in \triangle^{\mathcal{I}}$.

Satisfiability and Logical Consequence

Assume Σ is a theory (over signature S).

We say the following.

1. The pair $\mathcal{I}, \mathcal{V}$ is a *model* of Σ if $\mathcal{I}, \mathcal{V} \models \psi$ for all $\psi \in \Sigma$.
2. Σ is *satisfiable* if it has a model and *unsatisfiable* otherwise.
3. ϕ is a *logical consequence* of Σ, written

$$\Sigma \models \phi,$$

   iff $\mathcal{I}, \mathcal{V} \models \phi$ for any model $\mathcal{I}, \mathcal{V}$ of Σ.

The fundamental problem of reasoning in a given FOL theory Σ(S) is the problem of *logical implication*: establishing which $\phi \in$ WFF(S) are logical consequences of Σ(S).

ACME Case: Logical Constraints for PAYROLL

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

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

Ensures that each employee is associated with a single 3-tuple in $(\texttt{employee})^{\mathcal{I}}$ 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 $\mathcal{I}$.

Logical Constraints for PAYROLL (identification cont'd)

For S given by OPTION 2:

$$\begin{array}{l}\forall x, y.(\\ \quad (\texttt{employee}(x) \wedge \texttt{employee}(y) \\ \qquad \wedge \texttt{employee-number}(x) \approx \texttt{employee-number}(y)) \\ \quad \rightarrow x \approx y).\end{array}$$

For S given by OPTION 3:

$$\begin{array}{l}\forall x, y.(\\ \quad \exists z.(\texttt{employee}(x) \wedge \texttt{employee}(y) \\ \qquad \wedge \texttt{employee-number}(x, z) \wedge \texttt{employee-number}(y, z)) \\ \quad \rightarrow x \approx y).\end{array}$$

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 Σ to ensure the attributes are *partial functions*.

$$\forall x, y.(\exists z.(\texttt{employee-number}(z, x) \wedge \texttt{employee-number}(z, y)) \rightarrow (x \approx y))$$

$$\forall x, y.(\exists z.(\texttt{name}(z, x) \wedge \texttt{name}(z, y)) \rightarrow (x \approx y))$$

$$\forall x, y.(\exists z.(\texttt{salary}(z, x) \wedge \texttt{salary}(z, y)) \rightarrow (x \approx y))$$

Equality Generating Dependencies

Assume ϕ is a well formed formula of the form

$$(\psi_1 \wedge \cdots \wedge \psi_m)$$

in which each ψ_i is a function free atom.

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

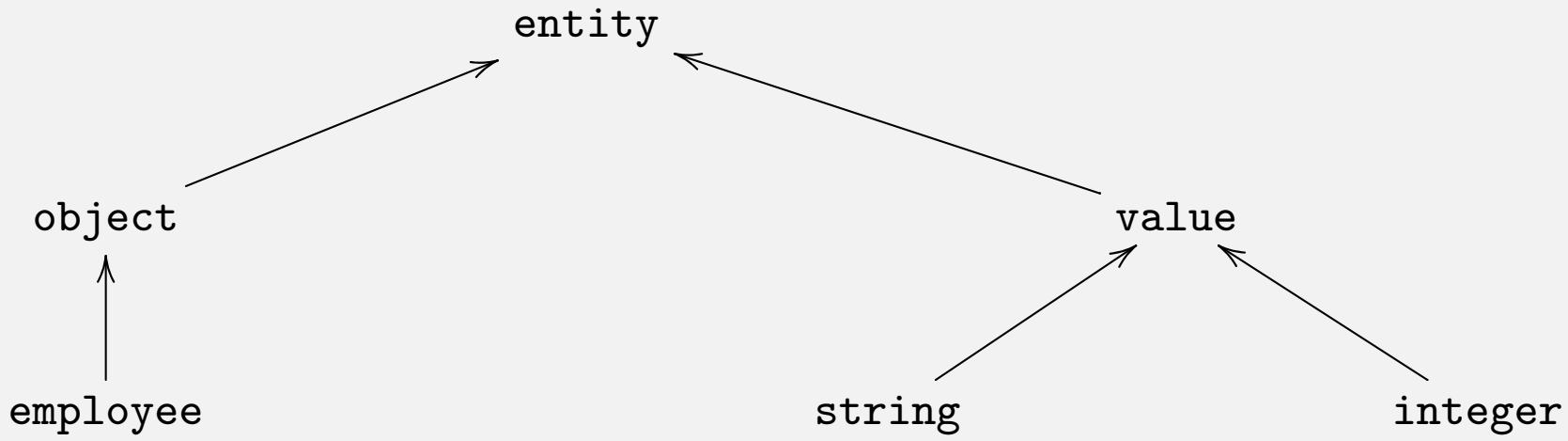
$$\forall x_1, x_2.(\exists x_3, \cdots, x_n.\phi \rightarrow (x_1 \approx x_2))$$

where $n \geq 2$ and where $\mathsf{Fv}(\phi) = \{x_1, \ldots, x_n\}$.

ACME Case: Logical Constraints for PAYROLL

On taxonomic knowledge.

A simple PAYROLL taxonomy.



"$\langle \mathrm{id} \rangle_1 \rightarrow \langle \mathrm{id} \rangle_2$" asserts: "set of all $\langle \mathrm{id} \rangle_1$" $\subseteq$ "set of all $\langle \mathrm{id} \rangle_2$".

More relevant to OPTION 2 or OPTION 3 for signature S of PAYROLL system.

Logical Constraints for PAYROLL

Add unary predicates `entity`, `object`, `value`, `string` and `integer` to S^{P}.

Add following logical constraints to Σ.

1. Everything is an entity.

$$\forall x.\texttt{entity}(x)$$

2. More specific entities are objects and values.

$$\forall x.(\texttt{object}(x) \rightarrow \texttt{entity}(x))$$

$$\forall x.(\texttt{value}(x) \rightarrow \texttt{entity}(x))$$

3. More specific objects are employees.

$$\forall x.(\texttt{employee}(x) \rightarrow \texttt{object}(x))$$

Logical Constraints for PAYROLL

4. More specific values are strings and integers.

$$\forall x.(\texttt{string}(x) \rightarrow \texttt{value}(x))$$

$$\forall x.(\texttt{integer}(x) \rightarrow \texttt{value}(x))$$

5. Nothing is just an entity or a value.

$$\forall x.(\texttt{entity}(x) \rightarrow (\texttt{object}(x) \vee \texttt{value}(x)))$$

$$\forall x.(\texttt{value}(x) \rightarrow (\texttt{string}(x) \vee \texttt{integer}(x)))$$

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

$$\forall x.(\texttt{object}(x) \rightarrow \neg\texttt{value}(x))$$

$$\forall x.(\texttt{string}(x) \rightarrow \neg\texttt{integer}(x))$$

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:

$$\forall x, y, z.(\texttt{employee}(x, y, z) \rightarrow (\texttt{integer}(x) \wedge \texttt{string}(y) \wedge \texttt{integer}(z)))$$

For S given by OPTION 2:

$$\forall x.(\texttt{employee}(x) \rightarrow (\texttt{integer}(\texttt{employee-number}(x)) \wedge \texttt{string}(\texttt{name}(x)) \wedge \texttt{integer}(\texttt{salary}(x))))$$

Logical Constraints for PAYROLL

For S given by OPTION 3:

$$\forall x.(\texttt{employee}(x) \rightarrow \exists y, z, w.(\texttt{employee-number}(x, y) \wedge \texttt{integer}(y) \\ \wedge\ \texttt{name}(x, z) \wedge \texttt{string}(z)) \\ \wedge\ \texttt{salary}(x, w) \wedge \texttt{integer}(w)))$$

OPTION 3 also makes it possible to say that *only employees have employee numbers*.

$$\forall x.(\exists y.\texttt{employee-number}(x, y) \rightarrow \texttt{employee}(x))$$

Tuple Generating Dependencies

Assume ϕ and ψ denote well formed formulae with the respective forms $(\phi_1 \wedge \cdots \wedge \phi_m)$ and $(\psi_1 \wedge \cdots \wedge \psi_n)$ in which each ϕ_i and ψ_i is a function free atom.

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

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

where $\mathrm{Fv}(\phi)$ and $\mathrm{Fv}(\psi)$ 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., $\mathrm{Fv}(\psi) = \{x_1, \ldots, x_i\}$).