Semantic Web Languages Basics

Web Ontology Languages

- Wide variety of languages for “Explicit Specification”
  - Graphical notations
    - Semantic networks
    - UML
    - RDF/RDFS
  - Logic based
    - Description Logics (e.g., OIL, DAML+OIL, OWL, OWL-DL, OWL-Lite, OWL 2, OWL 2 EL, OWL 2 QL, OWL 2 RL)
    - Rules (e.g., RuleML, RIF, SWRL, LP/Prolog)
    - First Order Logic (e.g., KIF)
- RDF and OWL-DL are the major players (so far ...)
- OWL 2, OWL 2 EL, OWL 2 QL, OWL 2 RL (new OWL) is coming . . .
- RIF (Rule interchange Format) is coming . . .

RDF

- Statements are of the form

$$\langle \textcolor{red}{\textit{subject}}, \textcolor{red}{\textit{predicate}}, \textcolor{red}{\textit{object}} \rangle$$

  called triples: e.g.

$$\langle \textit{umberto}, \textit{plays}, \textit{soccer} \rangle$$

- can be represented graphically as:

$$\boxed{\text{umberto}} \xrightarrow{\textit{plays}} \boxed{\text{soccer}}$$

- Statements describe properties of resources
- A resource is any object that can be pointed to by a URI (Universal Resource Identifier):

RDF Schema (RDFS)

- RDF Schema allows you to define vocabulary terms and the relations between those terms
- RDF Schema terms (just a few examples):
  - Class
  - Property
  - type
  - subClassOf
  - range
  - domain
- These terms are the RDF Schema building blocks (constructors) used to create vocabularies:

```
<Person,type, Class>
<hasColleague, type, Property>
<Professor, subClassOf,Person>
<Carole, type,Professor>
<hasColleague, range,Person>
<hasColleague, domain,Person>
```

RDF Syntax

- Pairwise disjoint alphabets
  - **U** (RDF URI references)
  - **B** (Blank nodes)
  - **L** (Literals)
- For simplicity we will denote unions of these sets simply concatenating their names
- We call elements in **UBL** terms (denoted t)
- We call elements in **B** variables (denoted x)

- RDF triple (or RDF atom):

$$(s, p, o) \in \mathbf{UBL} \times \mathbf{U} \times \mathbf{UBL}$$

  - s is the subject
  - p is the predicate
  - o is the object
- Example:

$$(\mathit{airplane}, \mathit{has}, \mathit{enginefault})$$

ρdf (restricted RDF)

- ρdf (read rho-df, the ρ from restricted rdf)
- ρdf is defined as the following subset of the RDFS vocabulary:

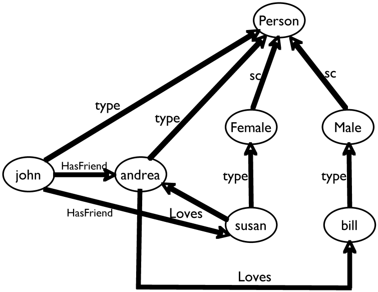
$$\rho\text{df} = \{\text{sp}, \text{sc}, \text{type}, \text{dom}, \text{range}\}$$

- (p, sp, q)
  - property p is a *sub property* of property q
- (c, sc, d)
  - class c is a *sub class* of class d
- (a, type, b)
  - a is of *type* b
- (p, dom, c)
  - *domain* of property p is c
- (p, range, c)
  - *range* of property p is c

- RDF graph (or simply a graph, or RDF Knowledge Base) is a set of RDF triples τ
- A subgraph is a subset of a graph
- The universe of a graph G, denoted by $universe(G)$ is the set of elements in **UBL** that occur in the triples of G
- The vocabulary of G, denoted by $voc(G)$ is the set $universe(G) \cap \mathbf{UL}$
- A graph is ground if it has no blank nodes (i.e. variables)

Example

$G = \{$ (*john*, type, *Person*), (*andrea*, type, *Person*), (*susan*, type, *Female*), (*bill*, type, *Male*), (*andrea*, *Loves*, *bill*), (*susan*, *Loves*, *andrea*), (*john*, *HasFriend*, *susan*), (*john*, *HasFriend*, *andrea*), (*Male*, sc, *Person*), (*Femal*, sc, *Person*) $\}$



- A variable assignment: a function $\mu : \mathbf{UBL} \to \mathbf{UBL}$ preserving URIs and literals, i.e.,
  - $\mu(t) = t$, for all $t \in \mathbf{UL}$
- Given a graph G, we define

$$\mu(G) = \{(\mu(s), \mu(p), \mu(o)) \mid (s, p, o) \in G\}$$

- We speak of a variable assignment μ from G_1 to G_2, and write $\mu : G_1 \to G_2$, if μ is such that $\mu(G_1) \subseteq G_2$

Example

- Assume

$$
\begin{aligned}
G_1 &= \{(x_1, has\_part, wheel), (x_2, has\_part, engine)\} \\
G_2 &= \{(y, has\_part, wheel), (y, has\_part, engine)\} \\
G_3 &= \{(y, has\_part, wheel), (y, has\_part, clutch)\} \\
\mu &= \{x_1 \mapsto y, x_2 \mapsto y\}
\end{aligned}
$$

- Then
  - μ is a variable assignment from G_1 to G_2 ($\mu(G_1) \subseteq G_2$)
  - μ is NOT a variable assignment from G_1 to G_3 ($\mu(G_1) \not\subseteq G_3$)

RDF Semantics

- RDF interpretation $\mathcal{I}$ over a vocabulary V is a tuple

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle \ ,$$

  where
  - $\Delta_R, \Delta_P, \Delta_C, \Delta_L$ are the interpretations domains of $\mathcal{I}$
  - $P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}}$ are the interpretation functions of $\mathcal{I}$

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

1. Δ_R is a nonempty set of resources, called the domain or universe of $\mathcal{I}$;
2. Δ_P is a set of property names (not necessarily disjoint from Δ_R);
3. $\Delta_C \subseteq \Delta_R$ is a distinguished subset of Δ_R identifying if a resource denotes a class of resources;
4. $\Delta_L \subseteq \Delta_R$, the set of literal values, Δ_L contains all plain literals in $\mathbf{L} \cap V$;
5. $P[\![\cdot]\!]$ maps each property name $p \in \Delta_P$ into a subset $P[\![p]\!] \subseteq \Delta_R \times \Delta_R$, i.e. assigns an extension to each property name;
6. $C[\![\cdot]\!]$ maps each class $c \in \Delta_C$ into a subset $C[\![c]\!] \subseteq \Delta_R$, i.e. assigns a set of resources to every resource denoting a class;
7. $\cdot^{\mathcal{I}}$ maps each $t \in \mathbf{UL} \cap V$ into a value $t^{\mathcal{I}} \in \Delta_R \cup \Delta_P$, i.e. assigns a resource or a property name to each element of **UL** in V, and such that $\cdot^{\mathcal{I}}$ is the identity for plain literals and assigns an element in Δ_R to elements in **L**;
8. $\cdot^{\mathcal{I}}$ maps each variable $x \in \mathbf{B}$ into a value $x^{\mathcal{I}} \in \Delta_R$, i.e. assigns a resource to each variable in **B**.

Models

Intuitively,

- A ground triple (s, p, o) in an RDF graph G will be true under the interpretation $\mathcal{I}$ if
  - p is interpreted as a property name
  - s and o are interpreted as resources
  - the interpretation of the pair (s, o) belongs to the extension of the property assigned to p
- Blank nodes, i.e. variables, work as existential variables: a triple $((x, p, o)$ with $x \in \mathbf{B}$ would be true under $\mathcal{I}$ if
  - there exists a resource s such that (s, p, o) is true under $\mathcal{I}$

Models (cont.)

Let G be a graph over ρdf.

- An interpretation $\mathcal{I}$ is a model of G under ρdf, denoted $\mathcal{I} \models G$, iff
  - $\mathcal{I}$ is an interpretation over the vocabulary $\rho\text{df} \cup \mathit{universe}(G)$
  - $\mathcal{I}$ satisfies the following conditions:

Simple:

1. for each $(s, p, o) \in G$, $p^{\mathcal{I}} \in \Delta_P$ and $(s^{\mathcal{I}}, o^{\mathcal{I}}) \in P[\![p^{\mathcal{I}}]\!]$;

Subproperty:

1. $P[\![\text{sp}^{\mathcal{I}}]\!]$ is transitive over Δ_P;
2. if $(p, q) \in P[\![\text{sp}^{\mathcal{I}}]\!]$ then $p, q \in \Delta_P$ and $P[\![p]\!] \subseteq P[\![q]\!]$;

Models (cont.)

Subclass:

1. $P[\![sc^{\mathcal{I}}]\!]$ is transitive over Δ_C;
2. if $(c, d) \in P[\![sc^{\mathcal{I}}]\!]$ then $c, d \in \Delta_C$ and $C[\![c]\!] \subseteq C[\![d]\!]$;

Typing I:

1. $x \in C[\![c]\!]$ iff $(x, c) \in P[\![type^{\mathcal{I}}]\!]$;
2. if $(p, c) \in P[\![dom^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $x \in C[\![c]\!]$;
3. if $(p, c) \in P[\![range^{\mathcal{I}}]\!]$ and $(x, y) \in P[\![p]\!]$ then $y \in C[\![c]\!]$;

Typing II:

1. For each $e \in \rho df$, $e^{\mathcal{I}} \in \Delta_P$
2. if $(p, c) \in P[\![dom^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
3. if $(p, c) \in P[\![range^{\mathcal{I}}]\!]$ then $p \in \Delta_P$ and $c \in \Delta_C$
4. if $(x, c) \in P[\![type^{\mathcal{I}}]\!]$ then $c \in \Delta_C$

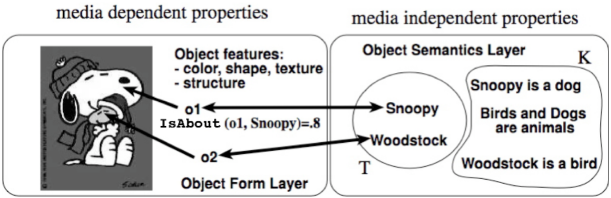
Entailment

- G entails H under ρdf, denoted $G \models H$, iff
  - every model under ρdf of G is also a model under ρdf of H
- Note: often $P[\![\mathrm{sp}^{\mathcal{I}}]\!]$ (resp. $C[\![\mathrm{sc}^{\mathcal{I}}]\!]$) is also *reflexive* over Δ_P (resp. Δ_C)
  - We omit this requirement and, thus, do NOT support inferences such as

$$\begin{array}{rcl} G & \models & (a, \mathrm{sp}, a) \\ G & \models & (a, \mathrm{sc}, a) \end{array}$$

    which anyway are of marginal interest

Example



$$G = \left\{ \begin{array}{ll} (o1, \mathit{IsAbout}, \mathit{snoopy}) & (o2, \mathit{IsAbout}, \mathit{woodstock}) \\ (\mathit{snoopy}, \mathrm{type}, \mathit{dog}) & (\mathit{woodstock}, \mathrm{type}, \mathit{bird}) \\ (\mathit{dog}, \mathrm{sc}, \mathit{animal}) & (\mathit{bird}, \mathrm{sc}, \mathit{animal}) \end{array} \right\}$$

Example (Model)

$$G = \left\{ \begin{array}{ll} (o1, IsAbout, snoopy) & (o2, IsAbout, woodstock) \\ (snoopy, \mathrm{type}, dog) & (woodstock, \mathrm{type}, bird) \\ (dog, \mathrm{sc}, animal) & (bird, \mathrm{sc}, animal) \end{array} \right\}$$

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

$$\begin{array}{rcl} \Delta_R & = & \{o1, o2, snoopy, woodstock, dog, bird, animal\} \\ \Delta_P & = & \{IsAbout, \mathrm{type}, \mathrm{sc}\} \\ \Delta_C & = & \{dog, bird, animal\} \\ P[\![IsAbout]\!] & = & \{\langle o1, snoopy\rangle, \langle o2, woodstock\rangle\} \\ P[\![\mathrm{type}]\!] & = & \{\langle snoopy, dog\rangle, \langle woodstock, bird\rangle, \langle snoopy, animal\rangle, \langle woodstock, animal\rangle\} \\ P[\![\mathrm{sc}]\!] & = & \{\langle dog, animal\rangle, \langle bird, animal\rangle\} \\ C[\![dog]\!] & = & \{snoopy\} \\ C[\![bird]\!] & = & \{woodstock\} \\ C[\![animal]\!] & = & \{snoopy, woodstock\} \\ t^{\mathcal{I}} & = & t \text{ for all } t \in \mathbf{UL} \end{array}$$

$$\mathcal{I} \models G \qquad \mathcal{I} \text{ is a model of } G$$

Example (Entailment)

$$G = \left\{ \begin{array}{ll} (o1, \mathit{IsAbout}, \mathit{snoopy}) & (o2, \mathit{IsAbout}, \mathit{woodstock}) \\ (\mathit{snoopy}, \text{type}, \mathit{dog}) & (\mathit{woodstock}, \text{type}, \mathit{bird}) \\ (\mathit{dog}, \text{sc}, \mathit{animal}) & (\mathit{bird}, \text{sc}, \mathit{animal}) \end{array} \right\}$$

$$\mathcal{I} = \langle \Delta_R, \Delta_P, \Delta_C, \Delta_L, P[\![\cdot]\!], C[\![\cdot]\!], \cdot^{\mathcal{I}} \rangle$$

$$\begin{array}{rcl} \Delta_R & = & \{o1, o2, \mathit{snoopy}, \mathit{woodstock}, \mathit{dog}, \mathit{bird}, \mathit{animal}\} \\ \Delta_P & = & \{\mathit{IsAbout}, \text{type}, \text{sc}\} \\ \Delta_C & = & \{\mathit{dog}, \mathit{bird}, \mathit{animal}\} \\ P[\![\mathit{IsAbout}]\!] & = & \{\langle o1, \mathit{snoopy}\rangle, \langle o2, \mathit{woodstock}\rangle\} \\ P[\![\text{type}]\!] & = & \{\langle \mathit{snoopy}, \mathit{dog}\rangle, \langle \mathit{woodstock}, \mathit{bird}\rangle, \langle \mathit{snoopy}, \mathit{animal}\rangle, \langle \mathit{woodstock}, \mathit{animal}\rangle\} \\ P[\![\text{sc}]\!] & = & \{\langle \mathit{dog}, \mathit{animal}\rangle, \langle \mathit{bird}, \mathit{animal}\rangle\} \\ C[\![\mathit{dog}]\!] & = & \{\mathit{snoopy}\} \\ C[\![\mathit{bird}]\!] & = & \{\mathit{woodstock}\} \\ C[\![\mathit{animal}]\!] & = & \{\mathit{snoopy}, \mathit{woodstock}\} \\ t^{\mathcal{I}} & = & t \text{ for all } t \in \textbf{UL} \end{array}$$

$$G \models (\mathit{snoopy}, \text{type}, \mathit{animal})$$

In all models $\mathcal{I}$ of G, $\langle \mathit{snoopy}, \mathit{animal}\rangle \in P[\![\text{type}]\!]$

Deduction System for RDF

- The system is arranged in groups of rules that captures the semantic conditions of models
- In every rule, A, B, C, X, and Y are meta-variables representing elements in **UBL**
- An instantiation of a rule is a uniform replacement of the metavariables occurring in the triples of the rule by elements of **UBL**, such that all the triples obtained after the replacement are well formed RDF triples

Deduction System for RDF (cont.)

1. Simple:

$$(a) \quad \frac{G}{G'} \text{ for a map } \mu : G' \to G \qquad (b) \quad \frac{G}{G'} \text{ for } G' \subseteq G$$

2. Subproperty:

$$(a) \quad \frac{(A,\text{sp},B),(B,\text{sp},C)}{(A,\text{sp},C)} \qquad (b) \quad \frac{(A,\text{sp},B),(X,A,Y)}{(X,B,Y)}$$

3. Subclass:

$$(a) \quad \frac{(A,\text{sc},B),(B,\text{sc},C)}{(A,\text{sc},C)} \qquad (b) \quad \frac{(A,\text{sc},B),(X,\text{type},A)}{(X,\text{type},B)}$$

4. Typing:

$$(a) \quad \frac{(A,\text{dom},B),(X,A,Y)}{(X,\text{type},B)} \qquad (b) \quad \frac{(A,\text{range},B),(X,A,Y)}{(Y,\text{type},B)}$$

5. Implicit Typing:

$$(a) \quad \frac{(A,\text{dom},B),(C,\text{sp},A),(X,C,Y)}{(X,\text{type},B)} \qquad (b) \quad \frac{(A,\text{range},B),(C,\text{sp},A),(X,C,Y)}{(Y,\text{type},B)}$$

Deduction System for RDF (cont.)

- Notion of proof:
  - Let G and H be graphs
  - Then $G \vdash H$ iff there is a sequence of graphs $P_1, \ldots, P_k$ with $P_1 = G$ and $P_k = H$, and for each j ($2 \leq j \leq k$) one of the following holds:
    1. there exists a map $\mu : P_j \to P_{j-1}$ (rule (1a));
    2. $P_j \subseteq P_{j-1}$ (rule (1b));
    3. there is an instantiation $\frac{R}{R'}$ of one of the rules (2)–(5), such that $R \subseteq P_{j-1}$ and $P_j = P_{j-1} \cup R'$.
- The sequence of rules used at each step (plus its instantiation or map), is called a proof of H from G.

Proposition (Soundness and completeness)

The RDF proof system $\vdash$ is sound and complete for $\models$, that is, $G \vdash H$ iff $G \models H$.

Example (Proof)

$$G = \left\{ \begin{array}{ll} (o1, IsAbout, snoopy) & (o2, IsAbout, woodstock) \\ (snoopy, \text{type}, dog) & (woodstock, \text{type}, bird) \\ (dog, \text{sc}, animal) & (bird, \text{sc}, animal) \end{array} \right\}$$

Let us proof that

$$G \models (snoopy, \text{type}, animal)$$

$$\begin{array}{llcll} G & \vdash & (snoopy, \text{type}, dog) & (1) & \text{Rule Simple (b)} \\ G & \vdash & (dog, \text{sc}, animal) & (2) & \text{Rule Simple (b)} \\ G & \vdash & (snoopy, \text{type}, animal) & (3) & \text{Rule SubClass (b) applied to (1) + (2)} \end{array}$$

RDF Query Answering

- We assume that a RDF graph G is *ground* and *closed*, i.e., G is closed under the application of the rules (2)-(5)
- Conjunctive query: is a Datalog-like rule of the form

$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\tau_1, \ldots, \tau_n$$

where

  - $n \geq 1$, $\tau_1, \ldots, t_n$ are triples
  - $\mathbf{x}$ is a vector of variables occurring in $\tau_1, \ldots, \tau_n$, called the *distinguished variables*
  - $\mathbf{y}$ are so-called *non-distinguished variables* and are distinct from the variables in $\mathbf{x}$
  - each variable occurring in τ_i is either a distinguished variable or a non-distinguished variable
- If clear from the context, we may omit the exitential quantification $\exists \mathbf{y}$
- For instance, the query

$$q(x, y) \leftarrow (x, \mathit{creates}, y), (x, \mathrm{type}, \mathit{Flemish}), (x, \mathit{paints}, y), (y, \mathit{exhibited}, \mathit{Uffizi})$$

has intended meaning to retrieve all the artifacts x created by Flemish artists y, being exhibited at Uffizi Gallery

RDF Query Answering (cont.)

- We will also write a query as
$$q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$$
where $\varphi(\mathbf{x}, \mathbf{y})$ is $\tau_1, \ldots, \tau_n$
- Furthermore, $q(\mathbf{x})$ is called the head of the query, while $\exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$ is is called the body of the query
- Finally, a disjunctive query (or, *union of conjunctive queries*) $\mathbf{q}$ is, as usual, a finite set of conjunctive queries in which all the rules have the same head
- For instance, the disjunctive query

$$\begin{array}{lcl} q(x, y) & \leftarrow & (x, \mathit{creates}, y), (x, \text{type}, \mathit{Flemish}), (x, \mathit{paints}, y), (y, \mathit{exhibited}, \mathit{Uffizi}) \\ q(x, y) & \leftarrow & (x, \mathit{creates}, y), (x, \text{type}, \mathit{Flemish}), (x, \mathit{paints}, y), (y, \mathit{exhibited}, \mathit{Louvre}) \end{array}$$

has intended meaning to retrieve all the artifacts x created by Flemish artists y, being exhibited either at Uffizi Gallery or at the Louvre Museum

RDF Query Answering (cont.)

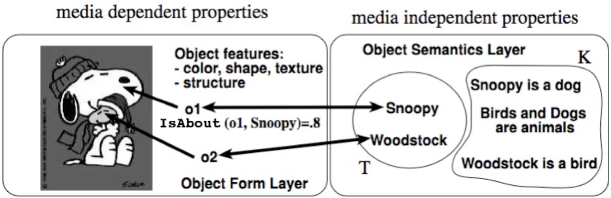
- Consider a graph G, a query $q(\mathbf{x}) \leftarrow \exists \mathbf{y}.\varphi(\mathbf{x}, \mathbf{y})$, and a vector $\mathbf{t}$ of terms in **UL**
- We say that $q(\mathbf{t})$ is entailed by G, denoted $G \models q(\mathbf{t})$, iff
  - in any model $\mathcal{I}$ of G, there is a vector $\mathbf{t}'$ of terms in **UL** such that $\mathcal{I}$ is a model of $\varphi(\mathbf{t}, \mathbf{t}')$
- If $G \models q(\mathbf{t})$ then $\mathbf{t}$ is called an answer to q
- For a disjunctive query $\mathbf{q} = \{q_1, \ldots, q_m\}$, we say that $\mathbf{q}(\mathbf{t})$ is entailed by G, denoted $G \models \mathbf{q}(\mathbf{t})$, iff $G \models q_i(\mathbf{t})$ for some $q_i \in \mathbf{q}$
- The answer set of $\mathbf{q}$ w.r.t. G is defined as

$$ans(G, \mathbf{q}) = \{\mathbf{t} \mid G \models \mathbf{q}(\mathbf{t})\}$$

RDF Query Answering (cont.)

- A simple query answering procedure is the following:
  - Compute the closure of a graph off-line
  - Store the RDF triples into a Relational database
  - Translate the query into a SQL statement
  - Execute the SQL statement over the relational database
- In practice, some care should be in place due to the large size of data: $\geq 10^9$ triples
- To date, several systems exists

Example



$$G = \left\{ \begin{array}{ll} (o1, IsAbout, snoopy) & (o2, IsAbout, woodstock) \\ (snoopy, \text{type}, dog) & (woodstock, \text{type}, bird) \\ (dog, \text{sc}, animal) & (bird, \text{sc}, animal) \end{array} \right\}$$

Consider the query

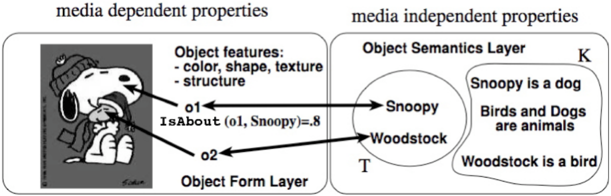
$$q(x) \quad \leftarrow \quad (x, IsAbout, y), (y, \text{type}, Animal)$$

Then

$$ans(G, q) \quad = \quad \{o1, o2\}$$

Representing degrees in RDF

- How can we represent degrees of uncertainty and vagueness in RDF/RDFS?
- Unfortunately, no standard exists yet
- So far, an option is to uses special purpose properties and reification



$$\langle statement1, hasSubject, o1 \rangle$$
$$\langle statement1, hasProperty, IsAbout \rangle$$
$$\langle statement1, hasObject, snoopy \rangle$$
$$\langle statement1, hasDegree, 0.8 \rangle$$

- But, then such statements have to be appropriately be managed by the system according to the underlying uncertainty or vagueness theory