

Modelování a odvozování v RDFS

Doc. Ing. Vojtěch Svátek, Dr.

Letní semestr 2012

<http://nb.vse.cz/~svatek/rzzw.html>

Odvozování

- Odvozování (inference) není jen záležitostí **umělé inteligence** – expertních systémů apod.
- I v systémech spravujících „běžná data“ může odvozování na straně **zdrojových dat** být náhradou za sofistikovanou tvorbu **dotazů**
 - Typicky jednoduché, deterministické, monotónní odvozování bez neurčitosti
- V **sémantickém webu** přichází odvozování
 - v menší míře s RDFS
 - v rozsáhlé míře s OWL, případně pravidlovými jazyky nad ním (SWRL, RIF)

RDFS jako schémátový jazyk

- Každé schéma RDFS je samo zároveň množinou trojic RDF
- Primárně určen k **odvozování**, nikoliv k ověřování formální **validity** dokumentů
 - Pozn.: existují i implementace odvozovacích nástrojů, které fakticky provádějí kontrolu datové integrity podle schémat v RDFS a OWL, mají ale jen podpůrnou úlohu, viz <http://clarkparsia.com/pellet/icv>
- Odvozování lze popsat pomocí **inferenčních pravidel** typu **IF – THEN** nad zobecněnými grafy RDF(S)

Odvozovací konstrukty RDFS

- Všechny jsou současně predikáty RDF...
 - rdfs:subClassOf
 - rdfs:subPropertyOf
 - rdfs:domain
 - rdfs:range
- Proto je možné trojice s těmito predikáty přidat k ostatním, a odvozovat nad elementárními fakty RDFS a tvrzeními RDF jako jedním celkem

Propagace typu

IF ?A rdfs:subClassOf ?B.

AND ?x rdf:type ?A.

THEN ?x rdf:type ?B.

IF ex:Professor rdfs:subClassOf ex:Teacher.

AND ex:Petr rdf:type ex:Professor.

THEN ex:Petr rdf:type ex:Teacher.

- Odvozování příslušnosti zdroje ke třídě „směrem nahoru“
 - vs. dědění atributů a metod tříd v OOP „dolů“
 - neexistuje přepisování (overriding) na nižší úrovni
 - „vícenásobná dědičnost“ (více rodičovských tříd pro danou třídu) je definována jednoznačně
- Lze chápat i jako inkluzi **množin** zdrojů

Lze řešit na úrovni SPARQL?

- Jen pro předem daný počet hierarchických úrovní, přepsáním dotazu na složitější

```
...WHERE { ?x rdf:type ?B }
```



```
...WHERE
```

```
{
```

```
    { ?x rdf:type ?B }
```

```
UNION
```

```
    { ?A rdfs:subClassOf ?B.
```

```
      ?x rdf:type ?A .          }
```

```
}
```

Propagace relace

IF ?P rdfs:subPropertyOf ?Q.

AND ?x P ?y.

THEN ?x Q ?y.

IF ex:isHeadOf rdfs:subPropertyOf ex:isEmployedBy.

AND ex:Vilem ex:isHeadOf ex:KIZI.

THEN ex:Vilem ex:isEmployedBy ex:KIZI.

- Funguje stejně, jako propagace tříd
 - ale na úrovni **uspořádaných dvojic** zdrojů
 - vlastnost je binární relace, tj. množina uspořádaných dvojic – tj. **množinová inkluze** platí stejně jako u tříd
- Nemá analogii v OOP

Odvozování vs. modelování

- Z hlediska odvozování je korektní vztah
ex:**returned** rdfs:subPropertyOf ex:**borrowed**.
- Z hlediska modelování reality ale neplatí,
že by vrácení věci bylo „specializovaným
případem“ jejího půjčení!

Odvození dle definičního oboru

IF ?P rdfs:domain ?A.
AND ?x P ?y.
THEN ?x rdf:type ?A.

```
IF ex:isEmployedBy rdfs:domain ex:Person.  
AND ex:Vilem ex:isEmployedBy ex:KIZI.  
THEN ex:Vilem rdf:type ex:Person.
```

- Díky OWA definiční obor (domain) nefunguje jako integritní omezení

```
IF ex:isEmployedBy rdfs:domain ex:Person.  
AND ex:LISp-Miner rdf:type ex:Software.  
AND ex:LISp-Miner ex:isEmployedBy ex:KIZI.  
THEN ???
```

Odvození dle definičního oboru

IF ?P rdfs:domain ?A.
AND ?x P ?y.
THEN ?x rdf:type ?A.

```
IF ex:isEmployedBy rdfs:domain ex:Person.  
AND ex:Vilem ex:isEmployedBy ex:KIZI.  
THEN ex:Vilem rdf:type ex:Person.
```

- Díky OWA definiční obor (domain) nefunguje jako integritní omezení

```
IF ex:isEmployedBy rdfs:domain ex:Person.  
AND ex:LISp-Miner rdf:type ex:Software.  
AND ex:LISp-Miner ex:isEmployedBy ex:KIZI.  
THEN ex:LISp-Miner rdf:type ex:Person
```

- ...ale jako odvozovací pravidlo!
 - Pozn.: odlišné třídy nejsou explicitně **disjunktní**

Odvození dle oboru hodnot

IF ?P rdfs:range ?A.
AND ?x P ?y.
THEN ?y rdf:type ?A.

```
IF ex:isEmployedBy rdfs:range ex:Organization.  
AND ex:Vilem ex:isEmployedBy ex:KIZI.  
THEN ex:KIZI rdf:type ex:Organization.
```

- Odvození dle oboru hodnot (range) funguje analogicky jako pro definiční obor

Subclass vs. domain/range

- Víceméně stejnou **realitu** a **odvozování** nad ní lze modelovat oběma způsoby
- `ex:isEmployedBy rdfs:range ex:Organization.`
IF `ex:Vojtech ex:isEmployedBy ex:KIZI.`
THEN `ex:KIZI rdf:type ex:Organization.`
- `ex:Employer rdfs:subClassOf ex:Organization.`
IF `ex:KIZI rdf:type ex:Employer.`
THEN `ex:KIZI rdf:type ex:Organization.`

RDFS – jednoduchý jazyk?

- RDFS „klame tělem“!
 - **Syntakticky** je velmi jednoduchý
 - Odvozování se **implementuje** také vcelku jednoduše
 - ...ale pro **uživatele** není snadné si na ně zvyknout!
 - Největší díl nejasností připadá na domain/range
 - Ale při kvalifikovaném využívání může přinést užitečný efekt
- Pro srovnání
 - RDF je jednoduché na pohled i podstatou
 - OWL je komplikovanější na pohled i podstatou

„Směr“ tvrzení domain/range

- Mějme tvrzení $P \text{ rdfs:domain } ?A$
 - Intuitivně bychom asi chápali jako: „Pro třídu A je definována vlastnost P“, tj. **od A směrem k P**
 - Správná interpretace ale je: „Pro levou stranu vlastnosti P je určena třída A“, tj. **od P směrem k A !!!**
- Analogicky pro range
- Na levé i pravé straně **jakékoli** vlastnosti se **vždy** může vyskytnout **jakýkoli** zdroj z univerza
 - tvrzení typu domain a range pouze umožňují tomuto zdroji odvozením přiřadit příslušnost k (další) třídě

Propagace domain/range

IF ?P rdfs:domain ?A.
AND ?A rdfs:subClassOf ?B.
THEN ?P rdfs:domain ?B.

```
IF ex:isEmployedBy rdfs:domain
ex:Person.
AND ex:Person rdfs:subClassOf
ex:BiologicalEntity.
THEN ex:isEmployedBy rdfs:domain
ex:BiologicalEntity.
```

- Podobně jako příslušnost instance se v hierarchii tříd „dědí nahoru“ i definiční obor
 - a opět analogicky i obor hodnot

Vícenásobná domain/range

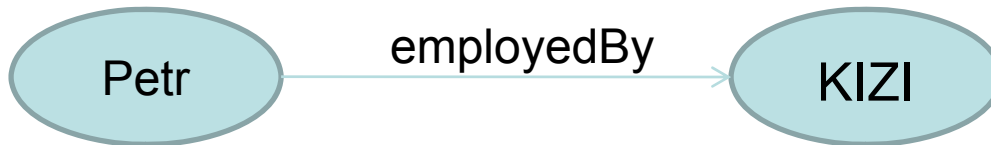
ex:teaches rdfs:domain ex:Person.

ex:teaches rdfs:domain ex:Organization.

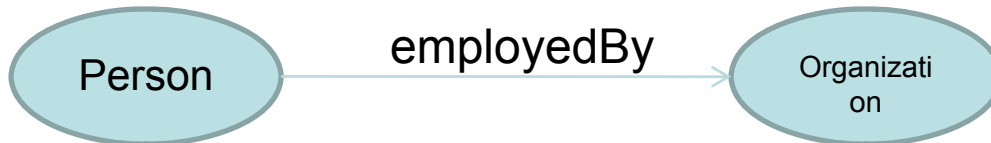
- Intuitivně: učit mohou lidé i celé organizace... je tato intuice správná?
- Ne! Jde o dva nezávislé axiomy! Pokud se nějaký zdroj vyskytne na levé straně tvrzení s ex:teaches, bude přiřazen jako instance k **oběma** třídám, tj. do jejich **průniku**
- V RDFS je třeba pečlivě zvažovat, zda dva způsoby použití vlastnosti nevyžadují dvě odlišné vlastnosti; nebo je nutno původní vlastnost nechat bez domain/range
- V OWL lze řešit tzv. lokálními restrikcemi

Vlastnost jako propojení tříd?

- **Úroveň instancí:** dva zdroje propojené vlastností



- **Úroveň tříd:** dvě třídy propojené vlastností, která je má v domain resp. range?



- Grafická notace ala RDF se často takto používá i pro úroveň tříd, to ale svádí k omylu
 - Nejde o jednu trojici RDF, ale o dvě; a u obou je employedBy subjektem, nikoliv predikátem!
 - Protože hodnotou domain/range nejsou jen explicitně zadané třídy, ale i jejich nadtřídy (viz předchozí slide), nelze ani v obecném smyslu chápat jako vztah mezi dvěma třídami!

Modelování ekvivalence

- Přímou lze jen v OWL
- V RDFS lze nahradit dvěma jednosměrnými vztahy

`?A rdfs:subClassOf ?B.`

`?B rdfs:subClassOf ?A.`

- To ovšem vede k nižší srozumitelnosti

Výsledek odvození v RDFS

- Nové trojice se přidají do databáze (pokud v ní ještě nejsou)
- Nemůže dojít k nekonzistenci
 - na rozdíl od odvození v OWL

Odvozování při integraci dat

- Náhrada termínu při zachování vazby na stará data

`pcvse:Autor rdfs:subClassOf isis:Person.`

- Propojení více terminologií
- Napojení proprietární vlastnosti na standardizovanou
 - Umožní např. zobrazovat hodnoty vlastnosti z proprietárního namespace v generické aplikaci

`isis:name rdfs:subPropertyOf rdfs:label.`

- atd.

RDFS nad rámec odvozování a modelování reality

- `rdfs:label`
 - Lidsky srozumitelný název entity
- `rdfs:seeAlso`
 - URI odkaz na libovolný související zdroj
- `rdfs:isDefinedBy`
 - Jde o podvlastnost k `seeAlso`
 - Odkaz na „definující“ zdroj
- `rdfs:comment`
 - Textový komentář