

Ontologie, OWL a deskripční logika

Doc. Ing. Vojtěch Svátek, Dr.

Letní semestr 2012

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

Témata

- Ontologie v informatice
- Jazyk OWL
- Deskripční logika

Pojem ontologie

- Ve filosofii:
 - „nauka o bytí“
 - „univerzální soustava znalostí o světě“
(tak, jak je, nezávisle na usuzování o něm...)
 - Kategorie „jsoucen“ (viz např. Aristoteles)
- V informatice:
 - „soustava znalostí o světě“
 - bez nároku na ucelenost
(mnoho dílčích ontologií pro různé domény)
 - často účelově vzniklý artefakt zohledňující způsob použití v informačním systému / aplikaci

Definice ontologie

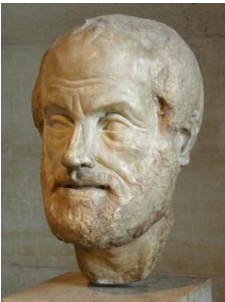
- (T. Gruber, 1993) „explicitní specifikace konceptualizace“
- (W. Borst, 1997) „*formální* specifikace *sdílené* konceptualizace“
- (B. Swartout, 1997) „hierachicky strukturovaná množina *termínů* popisujících určitou věcnou oblast...“
- a mnoho dalších

Typická struktura ontologie

- **Termíny**, resp. **pojmy**, které se používají k popisu nějaké reality – jim odpovídají reálné objekty
 - Rozlišují se obecné pojmy – **třídy**, a jejich **instance**
- **Hierarchické** uspořádání pojmů z hlediska obecnosti (nadtrída - podtrída)
- I **nehierarchické** vazby – „pojmenované“ vztahy, ve kterých se mohou objekty reálného světa vystupovat

Dva pohledy na ontologii

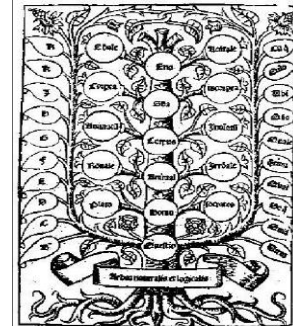
Aristoteles: Definitio
per **genus proximum**
et **differentia specifica**



Porfyriův strom:
myslíci vs. materiální
živé vs. neživé
racionální vs. neracionální
...

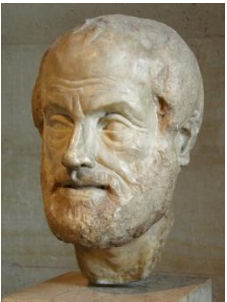
...

Doktorand je **student**
který úspěšně ukončil magisterskou úroveň
studia a věnuje se určitému výzkumnému tématu
pod vedením kvalifikovaného školitele



Dva pohledy na ontologii

Aristoteles: Definitio
per **genus proximum**
et **differentia specifica**



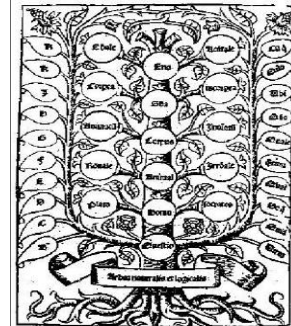
Modulární
systém
propojených
definic

Doktorand je **student**
který úspěšně ukončil magisterskou úroveň
studia a věnuje se určitému výzkumnému tématu
pod vedením kvalifikovaného školitele

Porfyriův strom:
myslí vs. materiální
živé vs. neživé
racionální vs. neracionální

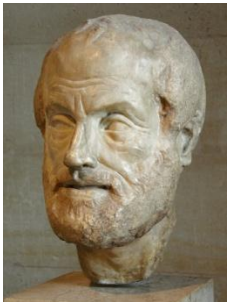
...

Systematická
taxonomie



Dva pohledy na ontologii

Aristoteles: Definitio
per **genus proximum**
et **differentia specifica**



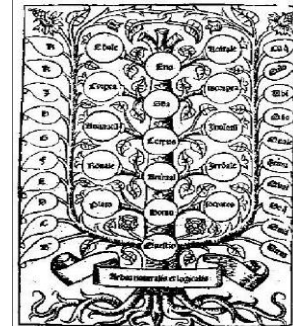
Modulární
systém
propojených
definic

Jedno z využití
deskripční logiky...

Porfyriův strom:
myslíci vs. materiální
živé vs. neživé
racionální vs. neracionální
...

Systematická
taxonomie

Doktorand je **student**
který úspěšně ukončil magisterskou úroveň
studia a věnuje se určitému výzkumnému tématu
pod vedením kvalifikovaného školitele



Témata

- Ontologie v informatice
- Jazyk OWL
- Deskripční logika

Jazyk OWL

- První verze doporučení W3C z r. 2004
- Aktuálně platná verze OWL 2 z r. 2009
 - Pro úvodní seznámení vhodný OWL 2 Primer <http://www.w3.org/TR/owl2-primer/>
- Několik syntaxí
 - „funkční“ syntaxe – normativní
 - zápis v podobě trojic RDF (v XML, Turtle npod.) – málo přehledné, ale umožňuje zpracovávat nástroji pro RDF
 - Manchesterská syntaxe – nepokrývá celý jazyk, ale je lidsky nejčitelnější
 - syntaxe OWL přímo v XML

OWL jako nadstavba RDFS

- OWL využívá všechny konstrukce z RDFS
 - subClassOf, subPropertyOf, domain, range, aj.
- Nejvýznamnějším rozšířením je možnost používat nejen pojmenované třídy, ale i **anonymní třídy** definované logickým výrazem
 - Konjunkce/disjunkce tříd, třídy definované restrikcí nad vlastností, výčtové třídy...
 - Lze do sebe vnořovat – skládat z jednodušších pojmů složitější
- Vlastnosti se explicitně rozlišují jako **objektové** (hodnotou je instance nějaké třídy) a **datové** (hodnotou je literál)
 - Vlastnostem lze přiřadit **charakteristiky**, jako je funkčnost, symetrie, vzájemná inverznost (pro dvě vlastnosti) apod.
- Možnost formulovat negativní tvrzení

Témata

- Ontologie v informatice
- Jazyk OWL
- Deskripční logika

Deskripční logika

- Rozvoj v 90. letech, označovaná také jako terminologická logika
- Vychází z predikátové logiky 1. řádu, ale omezenější
 - pouze binární relace (nazývané „role“...): $R(x,y)$, ne $R(x,y,z)$
 - bez funkčních symbolů: nelze $R(f(x,y))$
- Ne jeden kalkul, ale „rodina“ kalkulů, které se liší svou vyjadřovací silou
 - odlišeny akronymy, např. ALC, SHIQ *apod.*, podle konstrukcí, které umožňují

Role DL pro sémantický web

- Formální základ pro odvozování nad ontologiemi
- Její využití dodává ontologiím přidanou hodnotu oproti „ad hoc“ konceptuálním modelům
- Všechny konstrukce používané v jazyce OWL DL („střední proud“ OWL, podporovaný např. Protégé) mají své zakotvení v příslušné deskripční logice

Teorie v DL

- V praxi pojem „teorie“ často splývá s pojmem „ontologie“
- Teorie je množina formulí označovaných jako *axiomy* (nejde ovšem o axiomy jako „základní předpoklady“ určitého kalkulu)
- Axiomy se konstruují z *výrazů* (deskripcí)
 - *subsumpční* axiom (operátor \sqsubseteq) vyjadřuje, že je jeden výraz z logického hlediska zahrnut v druhém
 - *ekvivalenční* axiom (operátor \sqsubseteq) vyjadřuje logickou ekvivalenci dvou výrazů

Výrazy - deskripce

- Rozlišují se
 - konceptové výrazy („concept expressions“) – interpretací je množina objektů
 - relační výrazy („role expressions“) – interpretací je množina uspořádaných dvojic
- Axiomy přípustné v ontologických jazycích mají obvykle
 - na levé straně atomický výraz (jméno konceptu/relace)
 - na druhé straně atomický výraz nebo složený výraz

Příklady axiomů o konceptech

Doctor \mathbf{v} Person

- subsumpce dvou pojmenovaných tříd

HappyParent \mathbf{v}

Person \mathbf{u} δ hasChild.(Doctor \mathbf{t} η hasChild.Doctor)

- ekvivalence mezi pojmenovanou třídou a anonymní třídou (složeným konceptovým výrazem)
- anonymní třída je definována jako konjunkce pojmenované třídy Person a další anonymní třídy δ hasChild.(Doctor \mathbf{t} η hasChild.Doctor)
- tato vnořená anonymní třída označuje množinu entit, jejichž všechny děti jsou buďto sami doktoři nebo alespoň mají dítě, které je doktor

Příklady axiomů o relacích

hasDaughter \mathbf{v} hasChild

- subsumpce dvou pojmenovaných relací

isChildOf $\hat{=}$ hasChild $^{-}$

- ekvivalence mezi pojmenovanou relací a anonymní relací (označující inverzní relaci k hasChild)

Odvozovací úlohy

- Elementární odvozovací úlohy nad teorií
 - *test splnitelnosti*: zda je určitá třída splnitelná (existence jejích instancí není axiomy teorie logicky vyloučena), resp. zda se teorie skládá pouze ze splnitelných tříd
 - *odvození taxonomie*: do teorie jsou přidány vztahy subsumpce mezi pojmenovanými třídami, které logicky plynou z axiomů
 - Tyto dvě úlohy jsou na sebe převoditelné

DL a fakta o individuích

- Kromě obecné teorie, tzv. T-boxu (terminologické části) může báze znalostí obsahovat i množinu konkrétních faktů, tzv. A-box (část tvrzení - assertions)
- A-box může obsahovat tvrzení dvou typů
 - Instanciace třídy, např. John:HappyParent
 - Instanciace relace, např. John hasChild Mary
- Odvozování nad A-boxem spočívá např. ve zjištění, zda instance patří do určité třídy

Přesněji, dle specifikace OWL

- **Ontology Consistency:** Check whether a given ontology has at least one model.
- **Concept Satisfiability:** Given an ontology O and a class A , verify whether there is a model of O in which the interpretation of A is a non-empty set.
- **Concept Subsumption:** Given an ontology O and two classes A, B , verify whether the interpretation of A is a subset of the interpretation of B in every model of O
- **Instance Checking:** Given an ontology, an individual a and a class A , verify whether a is an instance of A in every model of the ontology.
- **Conjunctive Query Answering:** Given an ontology O and a conjunctive query q , return the answers of the query with respect to O .

Odvozovací postupy: tablové algoritmy

- Podstatou je konstrukce konkrétního modelu dané teorie v podobě *orientovaného grafu*, tzv. grafu zúplnění
- Každý uzel grafu
 - zastupuje individuum (explicitní nebo implicitní)
 - má k sobě přiřazenou množinu konceptových výrazů, kterých je toto individuum instancí
- Graf nedeterministicky expanduje pomocí aktivace *expanzních pravidel*
- Výpočet končí při vyčerpání možností aktivace pravidel nebo ve chvíli, kdy má některý uzel přiřazenu spornou množinu výrazů

Hlavní expanzní pravidla

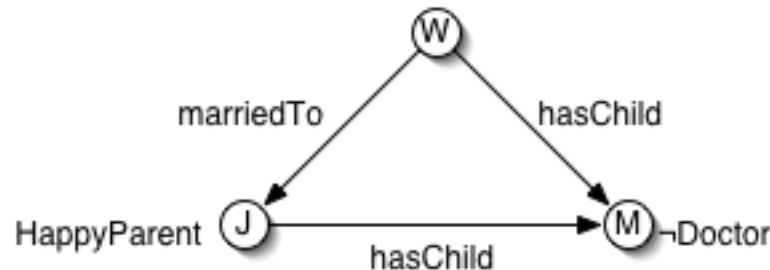
- **u -pravidlo:** Je-li v ohodnocení vrcholu i koncept $(C_1 \cup C_2)$, ale nejsou v něm oba konjunkty C_1, C_2 , potom je k vrcholu i přidáme
- **t -pravidlo:** Je-li v ohodnocení vrcholu i koncept $(C_1 \cap C_2)$, ale není v něm alespoň jeden z disjunktů C_1, C_2 , potom tento graf rozdělíme na dva grafy G_1, G_2 , lišící se pouze v ohodnocení vrcholu i
 - V G_1 bude v ohodnocení i navíc C_1 a v G_2 bude v ohodnocení i navíc C_2
- **9 -pravidlo:** Je-li v ohodnocení vrcholu i koncept $(9R.C)$ a současně vrchol i není spojen hranou ohodnocenou rolí R s nějakým vrcholem ohodnoceným konceptem C , potom vytvoříme nový vrchol, ohodnotíme jej konceptem C a spojíme s ním vrchol i hranou ohodnocenou rolí R
- **8 -pravidlo:** Je-li v ohodnocení vrcholu i koncept $(8R.C)$ a současně je vrchol i spojen hranou ohodnocenou rolí R s vrcholem j , který není ohodnocen konceptem C , potom j konceptem C ohodnotíme
- Pravidla pro kardinalitní omezení jsou zobecněním 9 –pravidla...

Příklad odvození tabla

- Teorie: T-box plus A-box

{HappyParent \sqsubset Person \sqcup \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor),
John:HappyParent, John hasChild Mary, Mary:: Doctor
Wendy hasChild Mary, Wendy marriedTo John }

- Nejprve se zkonstruuje graf obsahující pouze fakta
- Pak se uzly ohodnocené levou stranou (subsumpčních nebo ekvivalenčních) axiomů ohodnotí i výrazem na pravé straně



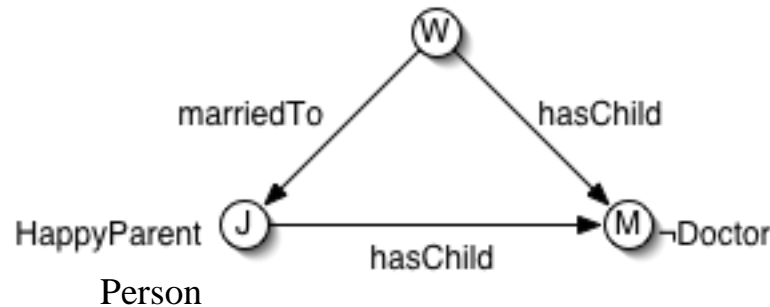
Person \sqcup \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor)

Příklad odvození tabla

- Teorie: T-box plus A-box

{HappyParent \sqsubseteq Person \sqcap \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor),
John:HappyParent, John hasChild Mary, Mary:: Doctor
Wendy hasChild Mary, Wendy marriedTo John }

- Pomocí pravidla pro konjunktci se uzlu J přiřadí oba konjunktvy definující třídu HappyParent



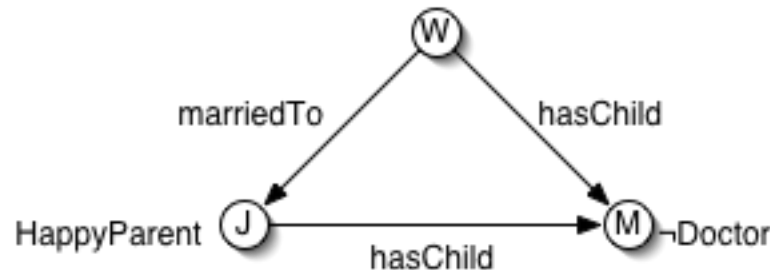
\exists hasChild.(Doctor \sqcap \exists hasChild.Doctor)

Příklad odvození tabla

- Teorie: T-box plus A-box

{HappyParent \sqsubseteq Person \sqcup \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor),
John:HappyParent, John hasChild Mary, Mary:: Doctor
Wendy hasChild Mary, Wendy marriedTo John }

- Pomocí pravidla pro univerzální restrikcí se odvodí, že všechny uzly, ke kterým od uzlu J vede hrana hasChild, musí mít přiřazený koncept Doctor \sqcap \exists hasChild.Doctor, tj. toto platí i pro M



HappyParent
Person
 \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor)

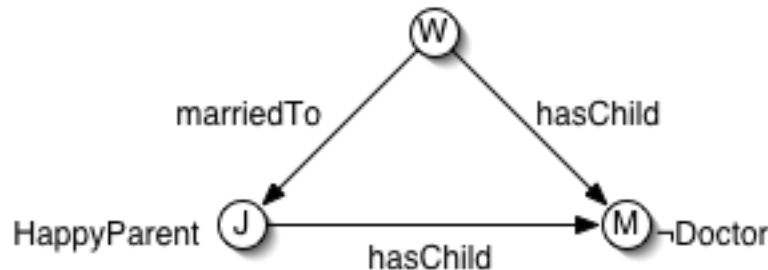
Doctor \sqcap \exists hasChild.Doctor

Příklad odvození tabla

- Teorie: T-box plus A-box

{HappyParent \sqsubset Person \sqcup \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor),
John:HappyParent, John hasChild Mary, Mary:: Doctor
Wendy hasChild Mary, Wendy marriedTo John }

- Protože koncept Doctor nelze pro uzel M použít
(v nově založeném „alternativním“ grafu by byl okamžitě spor),
je nutno odvozovat přes druhý disjunkt, \exists hasChild.Doctor



HappyParent
Person
 \exists hasChild.(Doctor \sqcap \exists hasChild.Doctor)

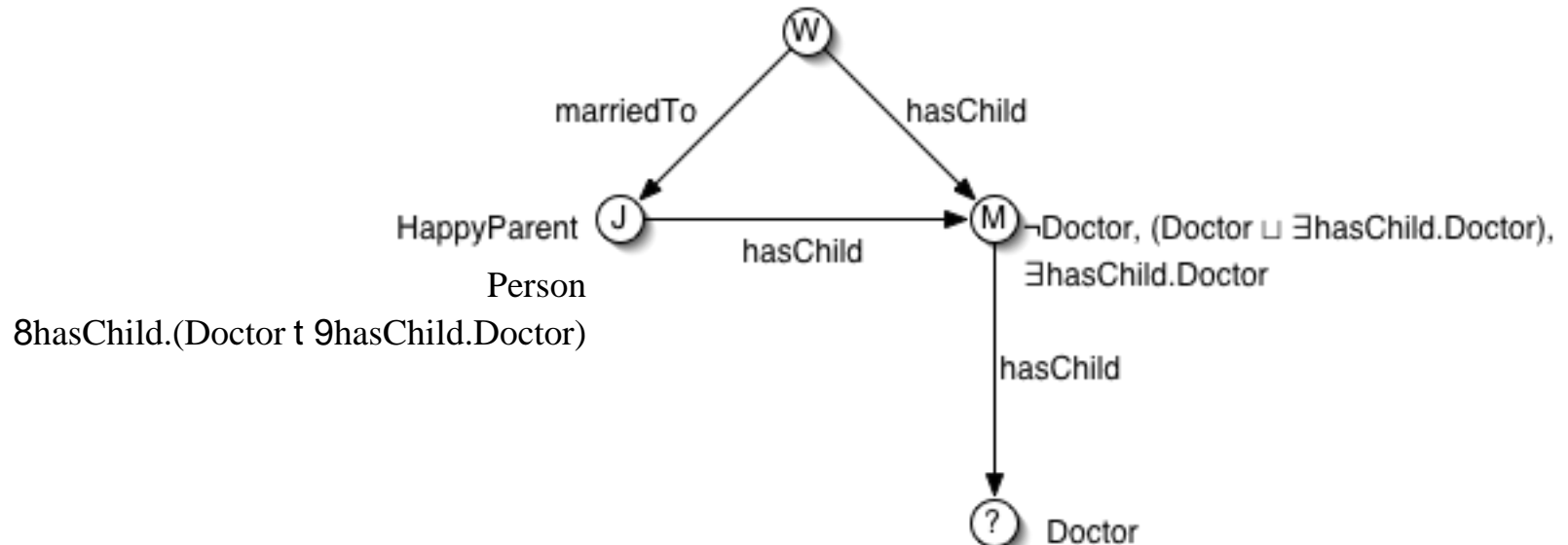
Doctor
Doctor \sqcap \exists hasChild.Doctor

Příklad odvození tabla

- Teorie: T-box plus A-box

$\{ \text{HappyParent} \sqsubseteq \text{Person} \sqcup \exists \text{hasChild} . (\text{Doctor} \sqcap \exists \text{hasChild} . \text{Doctor}),$
 $\text{John} : \text{HappyParent}, \text{John hasChild Mary}, \text{Mary} :: \text{Doctor}$
 $\text{Wendy hasChild Mary}, \text{Wendy marriedTo John} \}$

- Pomocí pravidla pro existenční restrikcí se odvodí, že od uzlu M musí vést hrana hasChild k nějakému dalšímu uzlu třídy Doctor



Příklad odvození tabla

- Teorie: T-box plus A-box

$\{ \text{HappyParent} \sqsubseteq \text{Person} \sqcup \exists \text{hasChild} . (\text{Doctor} \sqcap \exists \text{hasChild} . \text{Doctor}),$
 $\text{John} : \text{HappyParent}, \text{John} \text{ hasChild } \text{Mary}, \text{Mary} :: \text{Doctor}$
 $\text{Wendy} \text{ hasChild } \text{Mary}, \text{Wendy} \text{ marriedTo } \text{John} \}$

- Neexistuje další aktivovatelné pravidlo, takže výpočet končí, všechny třídy jsou splnitelné a teorie je konzistentní

