Ontologie
Logiki deskrypcyjne

Joanna Józefowska

Definicja

- Ontologia jest to jawny opis pewnej dziedziny zawierający:
  - pojęcia,
  - własności i atrybuty pojęć,
  - ograniczenia własności i atrybutów,
  - instancje (nie koniecznie).
- Ontologia wprowadza:
  - jednolitą terminologię,
  - jednolitą interpretację (definicję) pojęć.

Przykłady ontologii

- Taksonomie w sieci www
  - kategorie wyszukiwarki Yahoo!
- Katalogi zakupów on-line
  - katalog produktów Amazon.com
- Standardowa terminologia specyficzna dla danej dziedziny
  - Unified Medical Language System (UMLS)
  - PKWiU – klasyfikacja wyrobów i usług

Wykorzystanie ontologii

Proces budowania ontologii

„Gotowe" ontologie

- **Biblioteki ontologii**
  - DAML ontology library (www.daml.org/ontologies)
  - Ontolingua ontology library (www.ksl.stanford.edu/software/ontolingua/)
  - Protégé ontology library (protege.stanford.edu/plugins.html)
- Ontologie nadrzędne
  - IEEE Standard Upper Ontology (suo.ieee.org)
  - Cyc (www.cyc.com)
- Ontologie ogólne
  - DMOZ (www.dmoz.org)
  - WordNet (www.cogsci.princeton.edu/~wn/)
- Ontologie specyficzne dla dziedziny
  - UMLS Semantic Net
  - GO (Gene Ontology) (www.geneontology.org)

Narzędzia

- Protégé-2000:
  - jest graficznym narzędziem do budowy ontologii,
  - wspiera wiele aspektów reprezentacji wiedzy,
  - jest oprogramowaniem open-source (http://protege.stanford.edu).
- Inne narzędzia:
  - Ontolingua i Chimaera,
  - OntoEdit,
  - OilEd.

Języki dla Semantic Web

- RDF Schema
- DAML+OIL (OWL)
- SHOE
- XOL
- XML Schema

Języki dla Semantic Web

- Języki różnią się
  - składnią
    - To nas nie dotyczy – ontologia jest reprezentacją na poziomie pojęć a nie struktur.
  - terminologią
    - klasa-pojęcie
    - instancja-obiekt
    - slot-własność
  - ekspresyjnością
    - To, co możemy wyrazić w jakimś języku, nie zawsze da się wyrazić w innym
  - semantyką
    - Te same zdania mogą oznaczać co innego w różnych językach.

RDF Schema

- Język reprezentacji wiedzy pozwalający reprezentować ontologie dane w postaci RDF
- RDF = **Resource Description Framework**
- Ogólna metoda modelowania informacji
- Opis RDF składa się z „trójek" postaci: obiekt-predykat-określenie (subject-predicate-object).
- Np. niebo-kolor-niebieski (niebo ma kolor niebieski).
- RDFS, OWL – jezyki operujące na modelu RDF.

Web ontology Language (OWL)

- **OWL** (ang. *Web Ontology Language*) jest językiem ze składnią opartą na XML, a semantyką opartą na tzw. logice deskrypcyjnej (ang. *description logics*).
- Stanowi on rozszerzenie RDF (ang. *Resource Description Framework*).
- Służy do reprezentacji i przetwarzania danych w sieci WWW.
- OWL umożliwia opisywanie danych w postaci ontologii i budowanie w ten sposób tzw. Semantycznego Internetu.
- Istnieją trzy odmiany języka OWL:
  - OWL Lite;
  - OWL DL (rozszerzenie OWL Lite);
  - OWL Full (rozszerzenie OWL DL).
- OWL został uznany za standard przez W3C w lutym 2004 roku.

Logiki deskrypcyjne

- Są językami reprezentacji wiedzy.
- Formalizują podstawową terminologię modelowanej dziedziny.
- Przechowują ją w ontologii/terminologii/TBox-ie dla celów wnioskowania.
- Semantyka jest definiowana przez odwzorowanie na rachunek predykatów, dzięki czemu możliwe jest wnioskowanie na podstawie tak reprezentowanej wiedzy.
- Są rozszerzeniem sieci semantycznych i ram o formalną semantykę bazującą na rachunku predykatów.

Logiki deskrypcyjne - początki

- Semantyka rachunku predykatów może być zastosowana w systemach ram (Hayes, 1979).
- Na tej podstawie można budować użyteczne systemy o łatwej do zrozumienia semantyce.
- Opracowanie języka KL-ONE (Brachman & Leveque, 1985).
- Wcześniejsze nazwy: terminological systems, concept languages.

Systemy reprezentacji wiedzy oparte na logikach deskrypcyjnych

- KL-ONE (Brachman and Schmolze, 1985).
- LOOM (1987),
- BACK (1988),
- KRIS (1991),
- CLASSIC (1991),
- FaCT (1998),
- RACER (2001),
- CEL (2005),
- KAON 2 (2005).

Logiki deskrypcyjne

Podstawową składową każdej DL jest: język pojęć (concept language)

- Nazwy pojęć – są przyporządkowane grupom obiektów
- Nazwy ról – są przyporządkowane relacjom między obiektami
- Konstruktory – sposoby łączenia nazw pojęć i nazw ról

Symbol języka DL (KL-ONE, OWL)

Person ⊓ ∃ enrolled_at.University ⊓ ∀ attends.UnderGradCourse

Różne zbiory konstruktorów definiują różne języki pojęć.

Języki pojęć (TBox)

Pojęcia – oznaczają *obiekty*

(predykaty jednoargumentowe, klasy)

Przykład:

`Student`

`{x | Student(x)}`

`Żonaty`

`{x | Married(x)}`

Role – oznaczają *własności*

(predykaty dwuargumentowe, relacje)

Przykład:

`FRIEND`

`{<x; y> | FRIEND(x; y)}`

`LOVES`

`{<x; y> | LOVES(x; y)}`

Podstawowa logika $\mathcal{AL}$

$C, D \rightarrow A \mid \top \mid \bot \mid \neg A \mid C \sqcap D \mid \forall R.C \mid \exists R.\top$

Konstruktory	
$\neg A$	negacja formuły atomowej
$C \sqcap D$	koniunkcja
$\forall R.C$	ograniczenie wartości (value restriction)
$\exists R.\top$	ograniczenie istnienia (existencial restriction)

Podstawowa logika $\mathcal{AL}$

Attributive language

pojęcie

rola atomowa

$C, D \rightarrow A \mid \top \mid \bot \mid \neg A \mid C \sqcap D \mid \forall R.C \mid \exists R.\top$

pojęcie atomowe

pojęcie uniwersalne (top concept)

pojęcie „najniższe” (bottom concept)

pojęcie ::= <pojęcie atomowe> |
$\top$ |
$\bot$ |
$\neg$ *<pojęcie atomowe>* |
<pojęcie> $\sqcap$ *<pojęcie>* |
$\forall$*<rola atomowa>.<pojęcie>* |
$\exists$ *<rola atomowa>.*$\top$

Intuicyjna semantyka

Pojęcia reprezentują klasy, czyli zbiory obiektów.

Role reprezentują relacje między parami obiektów.

Pojęcia atomowe są nazwami elementarnych (niedefiniowanych) pojęć, a konstrukcje reprezentują obiekty złożone.

Dzięki temu można użyć kilku własności (tzn. pojęć nadrzędnych lub ograniczeń atrybutów) równocześnie w definicji pojęcia.

Semantyka - formalnie

Interpretacja $\mathcal{I} = (\Delta^{\mathcal{I}}; \bullet^{\mathcal{I}})$ składa się z:

- niepustego zbioru $\Delta^{\mathcal{I}}$ (dziedziny)
- funkcji $\bullet^{\mathcal{I}}$ (funkcji interpretacji)

odwzorowującej

- każde pojęcie na podzbiór $\Delta^{\mathcal{I}}$
- każdą rolę na podzbiór $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Interpretacja

$\top = \Delta^{\mathcal{I}}$

$\bot = \varnothing$

$(\neg A)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}$

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$

$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid (\forall y.\ (x; y) \in R^{\mathcal{I}}) \rightarrow (y \in C^{\mathcal{I}})\}$

$(\exists R.\top)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y.\ (x; y) \in R^{\mathcal{I}}\}$

Intuicja: interpretacja jest zupełnym opisem świata.

Technicznie: interpretacja jest strukturą pierwszego rzędu zawierającą jedynie predykaty jedno- i dwuargumentowe.

Interpretacja: ograniczenie wartości

Konstruktor $\forall$ nakłada ograniczenie na wartości atrybutu.

$$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid (\forall y.\ (x; y) \in R^{\mathcal{I}}) \rightarrow (y \in C^{\mathcal{I}})\}$$

Np. ($\forall$ `Ma_Dziecko.Lekarz`) określa zbiór obiektów, których każde dziecko jest lekarzem.

Komentarz: Jest to sposób na ograniczenie wartości szczeliny w ramie.

Interpretacja: ograniczenie istnienia

Konstruktor $\exists$ gwarantuje, że istnieje co najmniej jedna wartość dla danej nazwy atrybutu.

$$(\exists R)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y.\ (x; y) \in R^{\mathcal{I}}\}$$

Np. ($\exists$ `Ma_Dziecko.`$\top$) reprezentuje obiekt: rodzic.

Komentarz: W ten sposób w ramie wprowadzamy szczelinę.

Logika $\mathcal{FL}^-$

Frame Language

$C, D \rightarrow A \mid \top \mid \bot \mid \cancel{\neg A} \mid C \sqcap D \mid \forall R.C \mid \exists R.\top$

Logika $\mathcal{FL}_0$

$C, D \rightarrow A \mid \top \mid \bot \mid \cancel{\neg A} \mid C \sqcap D \mid \forall R.C \mid \cancel{\exists R.\top}$

Logika $\mathcal{ALC}$

C, D → A | ¬ C | C ⊓ D | C ⊔ D | ∀ R.C | ∃R.C

$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y \in C\ (x; y) \in R\}$

∃ Ma_Dziecko.Lekarz

Obiekt: rodzic, którego co najmniej jedno dziecko jest lekarzem.

Logika $\mathcal{ALC}$

Konstruktory

¬C	negacja
C ⊓ D	koniunkcja
C ⊔ D	dysjunkcja
∃R.C	ograniczenie istnienia (existencial restriction)
∀R.C	ograniczenie wartości (value restriction)

Skróty

C → D = ¬C ⊔ D	implikacja
C ↔ D = C → D ⊓ D → C	bi-implikacja

Przykłady pojęć złożonych

Person ⊓ Female

Person ⊓ ∃ attends.Course

Person ⊓ ∀ attends.(Course → ¬Easy)

Person ⊓ ∃ teaches.(Course ⊓ ∀ attended_by.(Bored ⊔ Sleeping))

Semantyka

Logiki deskrypcyjne organizują informację w klasach (pojęciach) budując zbiory instancji o wspólnych własnościach.

Przykład:

`Student ⊓ ∃FRIEND.Married`

`{x|Student(x)∧ ∃y FRIEND(x;y) ∧ Married(y)}`

Semantyka

$(\text{Student} \sqcap \exists\ \text{FRIEND.Married})^{\mathcal{I}}$

=

$(\text{Student})^{\mathcal{I}} \cap (\exists\ \text{FRIEND.Married})^{\mathcal{I}}$

=

{x | Student(x)} ∩

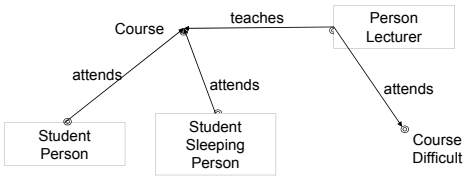
{x | ∃y. FRIEND(x; y) ∧ Married(y)}

=

{x | Student(x) ∧

∃y. FRIEND(x; y) ∧ Married(y)}

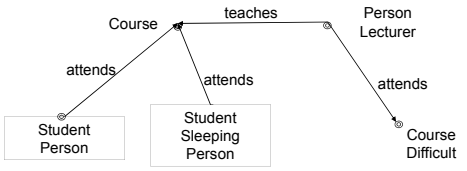
Przykład



Person ⊓ ∃ attends.Course

Person ⊓ ∀ attends.(¬Course ⊔ Difficult)

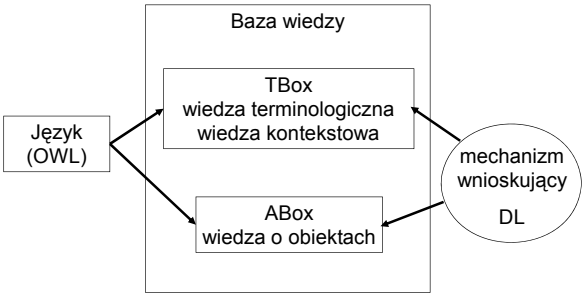
Przykład



Person ⊓ ∃attends.Course

Person ⊓∀ attends.(¬Course ⊔ Difficult)

Architektura systemu reprezentacji wiedzy opartego na logice deskrypcyjnej



TBox

TBox przechowuje definicje pojęć.

Syntaktyka

Skończony zbiór definicji pojęć: A = C
gdzie A – nazwa pojęcia.
Lewa strona musi być unikalna!

Semantyka

Interpretacja $\mathcal{I}$ spełnia definicję A = C wtw. $A^{\mathcal{I}} = C^{\mathcal{I}}$

$\mathcal{I}$ jest modelem $\mathcal{T}$ gdy spełnia wszystkie definicje w $\mathcal{T}$

Dwa rodzaje pojęć: zdefiniowane i pierwotne.

Lecturer = Person ⊓ ∃teaches.Course

Obiekty TBoxa: klasy

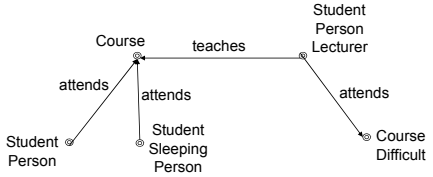
Student
Osoba
nazwisko: [String]
adres: [String]
zarejestrowany: [Specjalność]

{x | Student(x)} = {x | Osoba(x) ∧
(∃y NAZWISKO(x.y) ∧ String(y)) ∧
(∃z ADRES(x.z) ∧ String(z)) ∧
(∃w ZAPISANY(x.w) ∧ Specjalność(w))}

Student := Osoba ⊓ ∃NAZWISKO.String ⊓
∃ADRES.String ⊓ ∃ZAPISANY.Specjalność

TBox

Tbox wyznacza zbiór dopuszczalnych interpretacji.



Lecturer = Person ⊓ ∃teaches.Course

Student = Person ⊓ ∃attends.Course

ABox

ABox jest skończonym zbiorem stwierdzeń

C(a)	(a – nazwa indywiduowa, C – pojęcie)
R(a,b)	(a, b – nazwy indywiduowe, R – nazwy ról)

Np. $\mathcal{A}$ = {student(peter), tought-by(ai-course, jozefowska) }

Interpretacje $\mathcal{I}$ odwzorowują każdą nazwę indywiduową na element $\Delta^{\mathcal{I}}$.

$\mathcal{I}$ spełnia stwierdzenie

C(a)	wtw	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
R(a,b)	wtw	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

$\mathcal{I}$ jest modelem Abox $\mathcal{A}$ gdy $\mathcal{I}$ spełnia wszystkie stwierdzenia w $\mathcal{A}$.

Obiekty ABoxa: instancje

s1: Student
nazwisko: „Nowak”
adres: „Jana Pawła II ...”
zarejestrowany: SIZ

Student(s1) ∧ NAME(s1.„Nowak”) ∧ String(„Nowak”) ∧ ADRES(s1.„Jana Pawła II”) ∧ String(„Jana Pawła II”) ∧ ZAPISANY(s1.SIZ) ∧ Specjalność(SIZ)

ABox

- Interpretacje opisują stan świata w sposób zupełny.
- ABox opisuje stan świata w sposób niezupełny.
- ABox ma wiele modeli.
- ABox ogranicza zbiór dopuszczalnych modeli podobnie, jak TBox.

Wnioskowanie w TBox

- Spełnialność
- Subsumcja
- Równoważność
- Rozłączność

Wnioskowanie (TBox): spełnialność

C jest spełnialne ze względu na $\mathcal{T}$ wtw. istnieje model $\mathcal{T}$ taki, że $C^I \neq \emptyset$.

Intuicja: Jeżeli pojęcie nie jest spełnialne, to zawiera sprzeczność.

TBox $\mathcal{T}$

Woman = Person ⊓ Female

Man = Person ⊓¬Female

C = ∀ sibling.Woman ⊓ ∃ sibling.Man

jest niespełnialne ze względu na $\mathcal{T}$ ($C^I = \emptyset$).

$(\forall \text{ sibling.Woman})^I$ = {x: if (x,y) ∈ sibling then y ∈ Woman}

$(\exists \text{ sibling.Man})^I$ = {x: ∃y (x,y) ∈ sibling and y ∈ Man}

Wnioskowanie (TBox): spełnialność

Woman = Person ⊓ Female

Man = Person ⊓¬Female

C = ∀ sibling.Woman ⊓ ∃ sibling.Man

jest niespełnialne ze względu na $\mathcal{T}$ ($C^I = \emptyset$).

$(\forall \text{ sibling.Woman})^I$ = {x: if (x,y) ∈ sibling then y ∈ Woman}

$(\exists \text{ sibling.Man})^I$ = {x: ∃y (x,y) ∈ sibling and y ∈ Man}

Zbiór osób, które mają co najmniej jednego brata

Zbiór osób, które mają same siostry

Te zbiory są rozłączne

Wnioskowanie (TBox): spełnialność

C jest spełnialne ze względu na $\mathcal{T}$ wtw. istnieje model $\mathcal{T}$ taki, że $C^I \neq \emptyset$.

Spełnialność może być zredukowana do (nie) subsumcji i odwrotnie.

$C \sqsubseteq_{\mathcal{T}} D$ wtw $C \sqcap \neg D$ nie jest spełnialne ze względu na $\mathcal{T}$

C jest spełnialne ze względu na $\mathcal{T}$ wtw $\neg C \sqsubseteq_{\mathcal{T}} \bot$

Wnioskowanie (TBox): subsumcja (subsumtion)

$C \sqsubseteq_{\mathcal{T}} D$

C jest uogólnione (subsumed) przez D ze wzg[illegible] na $\mathcal{T}$

wtw

$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ zachodzi dla wszystkich modeli $\mathcal{I}$ TB

Intuicja

Jeżeli $C \sqsubseteq_{\mathcal{T}} D$ to D jest ogólni

Lecturer = Person

Student = Person

Lecturer ⊓ ∃ attends

subsumcja (*sub-* + *sumere* 'brać') *log.* proces wynajdowania dla danego pojęcia innego pojęcia, bardziej ogólnego.

Wg.„Słownika Wyrazów Obcych" Wydawnictwa Europa, pod redakcją naukową prof. Ireny Kamińskiej-Szmaj, autorzy: Mirosław Jarosz i zespół. ISBN 83-87977-08-X. Rok wydania 2001.

Wnioskowanie (TBox): subsumcja

$C \sqsubseteq_{\mathcal{T}} D$

C jest uogólnione (subsumed) przez D ze względu na $\mathcal{T}$

wtw

$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ zachodzi dla wszystkich modeli $\mathcal{I}$ TBoxa $\mathcal{T}$

Intuicja

Jeżeli $C \sqsubseteq_{\mathcal{T}} D$ to D jest ogólniejsze niż C.

Lecturer = Person ⊓ ∃ teaches.Course

Student = Person ⊓ ∃ attends.Course

Lecturer ⊓ ∃ attends.Course $\sqsubseteq_{\mathcal{T}}$ Student

Ogólne zawieranie pojęć

Ogólny TBox: skończony zbiór zależności (implikacji) (GCIs)

$C \sqsubseteq D$

gdzie C i D mogą być złożone.

Course ⊓ ∀ attended-by.Sleeping $\sqsubseteq$ Boring

Uwaga: $C \sqsubseteq D$ jest równoważne $T = C \rightarrow D$

(w sensie modelu $\mathcal{I}$)

Problem zawierania

$C \sqsubseteq D$

C *jest zawarte w* D wtedy i tylko wtedy gdy dla dowolnej dziedziny $\Delta^{\mathcal{I}}$ i dowolnej funkcji rozszerzającej $\bullet^{\mathcal{I}}$ nad $\Delta^{\mathcal{I}}$ zachodzi:

$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

czyli

$\forall x.\ C(x) \rightarrow D(x)$

Złożoność obliczeniowa

Zawieranie w $\mathcal{FL}^-$ jest:

- *rozstrzygalne*
- *należy do P*

Algorytm obliczania zawierania opiera się na porównaniu struktur wyrażeń opisujących pojęcia.

Algorytm SUBS

Niech $C = C_1 \sqcap \ldots \sqcap C_n$ and $D = D_1 \sqcap \ldots \sqcap D_m$ (postać normalna).

Wtedy SUBS?[C,D] kończy się wynikiem TRUE wtedy i tylko wtedy gdy dla wszystkich C_i:

1. Jeżeli C_i pojęciem atomowym lub pojęciem postaci ∃R, to istnieje D_j takie, że $C_i = D_j$;

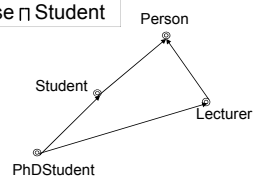
2. Jeżeli C_i jest pojęciem postaci ∀R.C', to istnieje D_j postaci ∀R.D' (ta sama rola atomowa R) takie, że SUBS?[C',D'].

Wnioskowanie (TBox): klasyfikacja

Uporządkowanie wszystkich pojęć zdefiniowanych w TBox w hierarchię ze względu na ogólność.

Lecturer = Person ⊓ ∃ teaches.Course
Student = Person ⊓ ∃ attends.Course
PhDStudent = ∃ teaches.Course ⊓ Student

Można wyznaczyć za pomocą wielokrotnych testów subsumcji.



Wnioskowanie (ABox)

ABox zgodność

$\mathcal{A}$ jest zgodne z $\mathcal{T}$ wtw istnieje interpretacja, która jest modelem $\mathcal{A}$ i $\mathcal{T}$.

Dany jest ABox $\mathcal{A}$ i TBox $\mathcal{T}$. Czy mają one wspólny model?

Sprawdzanie instancji $\mathcal{A}, \mathcal{T} \models C(a)$

Dane są ABox $\mathcal{A}$, TBox $\mathcal{T}$, nazwa indywiduowa a, i pojęcie C. Czy $a^I \in C^I$ zachodzi we wszystkich modelach $\mathcal{A}$ i $\mathcal{T}$?

Poniższe dwa zdania są równoważne:

- $\mathcal{A}$ jest zgodne z $\mathcal{T}$ wtw $\mathcal{A}, \mathcal{T} \not\models \bot(a)$
- $\mathcal{A}, \mathcal{T} \models C(a)$ wtw $\mathcal{A} \cup \{C(a)\}$ jest zgodne z $\mathcal{T}$.

Przykład

ABox

Mammal(dumbo)
Trunk(t14)
bodypart(dumbo, t14)
Darkgrey(g23)
~~color(dumbo, g23)~~
color(dumbo, Lightgrey)

TBox

Elephant = Mammal ⊓ ∃ bodypart.Trunk ⊓ ∀ color.Grey
Grey = Lightgrey ⊔ Darkgrey
⊥ = Lightgrey ⊓ Darkgrey

- ABox nie jest zgodny z TBox.
- dumbo jest instancją Elephant.

System wnioskowania (Tabele sematyczne)

Cel: algorytm, który dla dowolnego pojęcia C_0 z $\mathcal{ALC}$:

1. Daje wynik „spełnialne" wtw C_0 jest spełnialne.
2. Zatrzymuje się dla każdego wejścia.

tj. określa spełnialność pojęć $\mathcal{ALC}$

Uwaga: taki algorytm nie może istnieć dla rachunku predykatów gdyż RP jest nierozstrzygalny!

Postać normalna Negation normal form (NNF)

Negacja występuje tylko bezpośrednio przed nazwą pojęcia.

¬ ¬C	↔	C
¬ (C ⊓ D)	↔	¬ C ⊔ ¬ D
¬ (C ⊔ D)	↔	¬ C ⊓ ¬ D
¬ ∃R.C	↔	∀R. ¬ C
¬ ∀R.C	↔	∃R. ¬ C

Intuicja

Czy A ⊓ ∃R.B ⊓ ∀R. ¬B jest spełnialne?

Algorytm pracuje na drzewie, które

- reprezentuje pewien model $\mathcal{I}$:
  - wierzchołki odpowiadają elementom Δ^I
    każdy wierzchołek x jest etykietowany pojęciami $\mathcal{L}(x) \subseteq sub(C_0)$, $C \in \mathcal{L}(x)$ czyta się jako „x powinno być instancją C"
  - łuki reprezentują związki wynikające z ról
    każdy łuk $\langle x,y \rangle$ jest etykietowany nazwą roli C_0, $R \in \mathcal{L}(\langle x,y \rangle)$ czyta się „(x,y) powinno być elementem R"
- zaczyna się wierzchołkiem x_0 z etykietą $\mathcal{L}(x_0) = \{C_0\}$
- rozwija się stosując odpowiednie reguły

Reguły

Stosujemy reguły tylko wtedy, gdy wnoszą „coś nowego".

⊓rule: if $(C_1 \sqcap C_2) \in \mathcal{L}(x)$ and $\{C_1, C_2\} \not\subset \mathcal{L}(x)$
then set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{C_1, C_2\}$

⊔rule: if $(C_1 \sqcup C_2) \in \mathcal{L}(x)$ and $\{C_1, C_2\} \neq \emptyset$
then set $\mathcal{L}(x) = \mathcal{L}(x) \cup C$ for some $C \in \{C_1, C_2\}$

∃rule: if $\exists S.C \in \mathcal{L}(x)$ and x has no S-successor y with $C \in \mathcal{L}(x)$
then create a new node y with $\mathcal{L}(\langle x, y \rangle) = \{S\}$ and
$\mathcal{L}(y) = \mathcal{L}(x) \cup \{C\}$

∀rule: if $\forall S.C \in \mathcal{L}(x)$ and there is an S-
with $C \notin \mathcal{L}(y)$
then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$

Reguła ⊔ jest niedeterministyczna

Sprzeczność

Drzewo c-tree zawiera sprzeczność, jeżeli ma wierzchołek x taki, że $\bot \in \mathcal{L}(x)$ lub $\{A, \neg A\} \subseteq \mathcal{L}(x)$ w przeciwnym razie jest niesprzeczne (clash-free)

C_0 jest spełnialne wtw gdy reguły uzupełniania mogą być zastosowane w taki sposób, aby otrzymać zupełne i niesprzeczne drzewo c-tree.

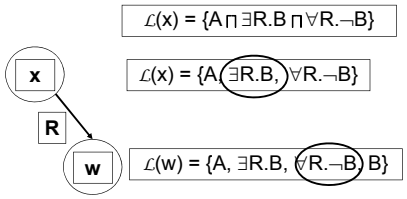
Ta procedura nie jest deterministyczna!

Własności tabel semantycznych

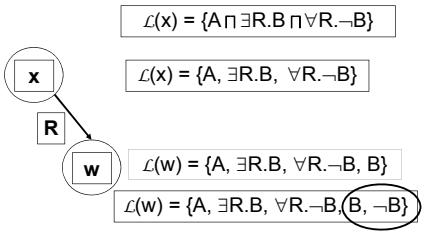
Niech C_0 będzie pojęciem $\mathcal{ALC}$ w NNF. Wtedy:

1. Algorytm kończy obliczenia dla danego C_0 i
2. Reguły mogą być zastosowane w taki sposób, że generują zupełne drzewo domknięte wtw, gdy C_0 jest niespełnialne.

Przykład



Przykład



$x \in A^I$	$x \in A^I$	**SPRZECZNOŚĆ!**
$x \in (\exists R.B)^I$	$\exists d: (x,d) \in R^I, d \in B^I$	
$x \in (\forall R.\neg B)^I$	$d \in (\neg B)^I$	

Logiki deskrypcyjne

- Formalizm uogólniający reprezentację wiedzy:
  - Systemy ram
  - Sieci semantyczne
  - Reprezentacje obiektowe
  - Modele semantyczne
  - Języki ontologii

  : : :
- Jest ustrukturalizowanym fragmentem rachunku predykatów.
- Jest teorią pozwalającą reprezentować ustrukturalizowaną informację i dostęp oraz wnioskowanie w tej wiedzy.

Ustrukturalizowana logika

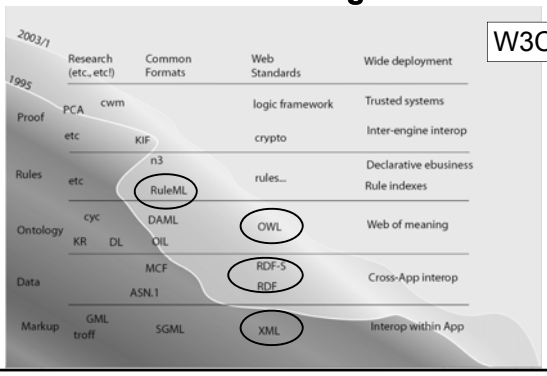
Każda logika deskrypcyjna jest fragmentem rachunku predykatów.

Ontologia w języku logiki deskrypcyjnej składa się z dwóch elementów:

- definicji predykatów (*TBox*)
- zbioru stwierdzeń na stałych (*ABox*)

Każda (podstawowa) logika deskrypcyjna jest podzbiorem *L3*, czyli rachunku predykatów bez funkcji z co najwyżej trzema nazwami zmiennych.

Standardy reprezentacji wiedzy dla web-mining



W3C