MODELOVÁNÍ ZNALOSTÍ

Zdeňka Telnarová

Ostravská univerzita, Přírodovědecká fakulta, 30. dubna 22, Ostrava,
Zdenka.Telnarova@osu.cz

Abstrakt
Příspěvek se věnuje modelování znalostí v konkrétním prostředí objektově deduktivního systému řízení báze dat ConceptBase, jehož součástí je jazyk pro modelování znalostí Telos. Je stručně popsána znalostní báze v Telosu, včetně jednoduchého příkladu, dále pak deduktivní pravidla v Telos a definice integritních omezení, opět s jednoduchými příklady implementace.

1. Úvod

Ve vývoji databázových systémů se setkáváme s permanentní snahou o dosažení co nejvyšší nezávislosti dat na aplikacích, které s nimi manipulují. Jako příklad můžeme uvést zásadní osamostatnění popisů dat od aplikací nebo s objektovými paradigmaty spojené začlenění popisu chování objektu, o němž se data v databázi udržují, do samotné databáze. Deduktivní databáze rozšiřují nezávislost dat na aplikacích v dalším smyslu a to v tzv. znalostní nezávislosti. Znalostní nezávislost je v deduktivních databázích realizována deduktivními pravidly, která slouží zejména k odvozování dat z dat uložených v databázi a k definování integritních omezení. Deduktivní pravidla jsou součástí databáze a jsou nezávislá na aplikacích, které nad databází pracují. Příspěvek se věnuje reprezentaci znalostí v prostředí objektově deduktivního systému řízení báze dat ConceptBase, jehož jazyk Telos mimo jiné disponuje prostředky pro modelování a reprezentaci znalostí.

2. Jazyk Telos v conceptbase

Model vytvořený v jazyce Telos zevšeobecňuje dřívější datové modely a znalostní reprezentace, jako E-R diagram nebo sémantické sítě a spojuje je s predikátovými tvrzeními a dočasnými informacemi. Telos podporuje tři různé formáty pro zápis příkazů. Základním formátem je formát logický, který umožňuje začlenit jazyk predikátových tvrzení pro deduktivní pravidla, dotazy a integritní omezení. Dalšími formáty je formát grafický (sémantická síť) a formát rámcový (frame). Oba formáty jsou založeny na formátu logickém a jsou z něho odvozeny.

2.1 Logická interpretace znalostní báze

Znalostní báze v Telosu je založena na konečné množině vzájemně propojených tvrzení, která jsou zde chápána jako objekty.

$$\mathbf{KB = \{P(oid, x, l, y, tt) |\ oid, x, y, tt \in ID, l \in LABEL\ \},}$$

kde:
oid je identifikátor, který je klíčem v znalostní bázi,
ID je neprázdná množina identifikátorů,
LABEL je neprázdná podmnožina jmen,

x je zdroj,
l je jméno,
y je cíl,
tt je doba trvání.

Objek **x** má vztah s objektem **y,** kde jméno vztahu je **l**. Vztah se předpokládá do doby **tt**.

2.2 Vzory tvrzení

V ConceptBase jsou rozeznávány čtyři vzory tvrzení a jsou jim dávána následující jména:

1) Individuals
   P(oid, oid, l, oid, tt)
   Toto tvrzení vytváří objekt **oid** se jménem **l**, jehož trvání se předpokládá do doby **tt**.

2) InstanceOf relationship (instantiations)
   P(oid, x, *instanceof, y, tt)
   Toto tvrzení vytváří instanci **x** třídy **y** do doby **tt.**

3) IsA relationships (specializations)
   P(oid, x, *isa, y, tt)
   Toto tvrzení vytváří třídu **x** jako specializaci třídy **y** do doby **tt**.

4) Attribute
   Všechna další tvrzení.

2.3 Předdefinované třídy Telosovské znalostní báze

Telos využívá strukturální axiomy, jejichž úplný výčet zde neuvádím. Pro zájemce odkazuji na literaturu [1]. Tyto axiomy jsou spojeny s předdefinovanými třídami, které jsou automatickou součástí každé Telosovské znalostní báze.

Individual	obsahuje všechny individualy jako instance
InstanceOf	obsahuje všechny konkretizace jako instance
IsA	obsahuje všechny specializace jako instance
Attribute	obsahuje všechny atributy jako instance
Proposition	obsahuje všechna tvrzení jako instance
Class	obsahuje všechny třídy (včetně sebe) jako instance
Token	obsahuje ty individualy, které již nemohou mít instance
SimpleClass	obsahuje individualy, které mohou mít instance, které jsou Token
MetaClass	obsahuje individualy, které mohou mít instance, které jsou SimpleClass
Metameta Class	obsahuje individualy, které mohou mít instance, které jsou MetaClass

Navíc Telos využívá vestavěných tříd jako Integer, Real a String.

2.4 Některé Telosovské axiomy

Uživatelé nepracují přímo s tvrzeními, ale s jejich textovou (frame) nebo grafickou (sémantická síť) podobou. Tyto formáty nejsou založeny na **oid** objektů, ale na jejich komponentě **label**. Aby bylo garantováno jednoznačné mapování, musí platit axiom pojmenování.

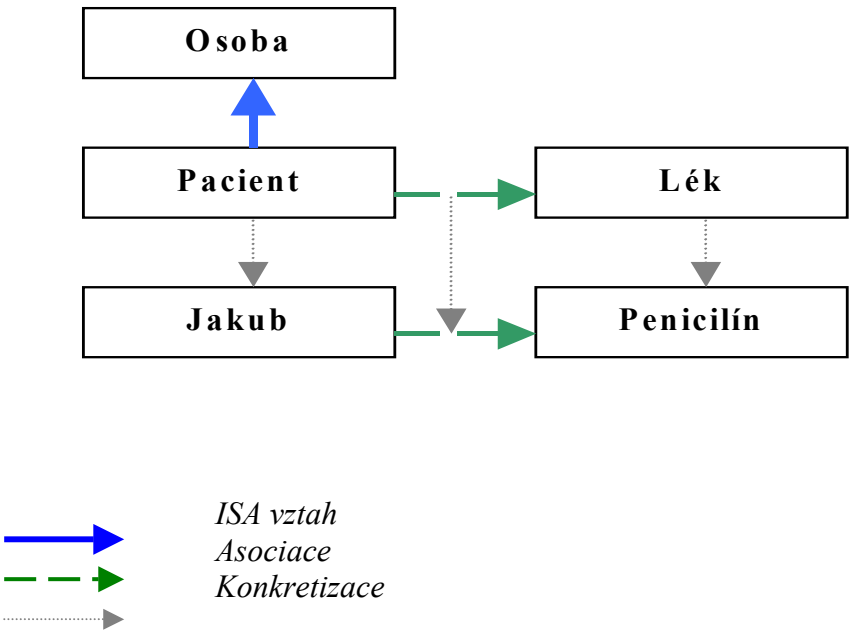
Axiom pojmenování
Jméno individualu musí být jednoznačné, jména všech atributů společného zdrojového objektu musí být rovněž jednoznačná.

Specializační axiom
Cílový objekt specializace dědí všechny instance svého zdroje.

Konkretizační axiom
Jestliže **p** je tvrzení, které je instancí tvrzení **P**, pak zdroj **p** musí být instancí zdroje **P** a cíl **p** musí být instancí cíle **P**.

2.5 Jednoduchý model v Telos

Nechť existuje následující jednoduchý model:



Pak tvrzení, korespondující s uvedenými objekty a vztahy, jsou následující:

Proposition(Osoba, Osoba, -, Osoba)
Proposition(Pacient, Pacient, -, Pacient)
Proposition(#1, Pacient, *isa, Osoba)
Proposition(Lék, Lék, -, Lék)
Proposition(#2, Pacient, bere, Lék)
Proposition(Penicilín, Penicilín, -, Penicilín)
Proposition(#3, Penicilin, *instanceof, Lék)
Proposition(Jakub, Jakub, -, Jakub)
Proposition(#4, Jakub, *instanceof, Pacient)
Proposition(#5, Jakub, lék1, Penicilín)
Proposition(#6, #5, *instanceof, #2)

Oid, výše označováno #, je generováno systémem. Objekty, korespondující s tvrzeními 1), 2) a 4), jsou individualy. Ostatní objekty se nazývají atributy a to atributy instanční, které mají třetí komponentu ***instanceof*** a atributy specializační, které mají třetí komponentu ***isa***.

Je-li tvrzení ve formě **Proposition(oid, x, *instanceof, c)**, pak **x** je instancí **c** (**c** je třídou **x**). Je-li tvrzení ve formě **Proposition(oid, c, *isa, d)**, pak **c** je subclass **d** (**d** je superclass **c**).

3. Predikátový jazyk CBL jako součást TELOS

Predikátový jazyk CBL se využívá k vyjádření integritních omezení, deduktivních pravidel a dotazů. Proměnné, použité ve formulích, musí být kvantifikovány a spojeny s typem, který limituje rozsah možných instancí z množiny instancí dané třídy. Obvykle je znalostní báze omezena do doby rollbacku, která závisí na typu formule.

$$\mathbf{KB_{rbt} = \{P(oid, x, l, y, tt) \text{ in } KB \mid rbt \text{ during } tt\}}$$

Integritní omezení je vyhodnocováno v aktuální **KB** (rbt představuje aktuální čas prováděné transakce). Rollback pro dotaz je dán dobou vyhodnocení dotazu. Význam použitých symbolů je shodný s významem, popsaným v době 2.1.

3.1 Literály, umožňující přístup k Telos objektům

(x in c) resp. **in(x,c)** infixová, resp. prefixová notace
Objekt **x** je instancí třídy **c**.
(c isA d) resp. **IsA(c,d)**
Objekt **c** je specializací třídy **d.**
(x m y) resp. **A(x,m,y)**
Objekt **x** má atribut, jehož hodnoty jsou instance objektu **y**. Takto definovaný vztah je instancí kategorie atributů s označením **m.**
From(p, x)
Objekt **p** má zdroj v objektu **x**.
To(p, y)
Objekt **p** má cíl v objektu **y**.
Label(p, l)
Objekt **p** má jméno **l**.

Následující literály definují druhou třídu formulí:
x < y, x > y, x <= y, x >= y, x = y, x <> y,
kde **x**,**y** musí být instance třídy Integer nebo Real.
x = = y
Objekty **x** a **y** jsou totožné.

3.2 Kvantifikovatelné proměnné

Formule s univerzálním kvantifikátorem má tvar:

Forall x | c F

nebo

forall x(x in c) → F

Pro všechny instance **x** třídy **c** platí formule **F.**

Formule s existenčním kvantifikátorem má tvar:

Exists x | c F
nebo
exists x (x in c) and F
Existuje instance **x** třídy **c**, pro kterou platí formule **F.**

3.3 Restrikce pro formule

Každá konstanta, obsažená ve formuli **F,** musí být jméno existujícího objektu v Telosovské znalostní bázi nebo je to konstanta vestavěné třídy Integer, Real nebo String. Každý atribut **(x m y)** obsažený ve formuli musí mít unikátní jméno **m** ve znalostní bázi.

Definice 3-1 podle[2]

Legální integritní omezení je CBL formule, která splňuje uvedené restrikce.

Příklad:
Zapišme integritní omezení pro plat zaměstnance/manažéra, když platí, že plat zaměstnance/manažéra nesmí být větší než 10.000,- Kč
Forall x | Zaměstnanec y | Integer (x plat y) → y <= 10000
Forall x | Manažér y | Integer (x plat y) → y <= 10000

Definice 3-2 podle[2]
Legální deduktivní pravidlo je CBL formule splňující uvedené restrikce a mající formát:

$$\textbf{forall } x_1 \mid c_1 \ldots \textbf{forall } x_n \mid c_n \; R \rightarrow \textbf{lit}(a_1, \ldots, a_m)$$

kde:
lit je literál typu 1) 2) nebo 3),
proměnné $a_1, \ldots, a_m$ odpovídají proměnným $x_1, \ldots x_n$, které jsou tříd $c_1, \ldots c_a$,
R je formule.
V Telosu jsou pravidla a integritní omezení definována jako atributy tříd. Text formule je uzavřen do znaků "$". Následující formule definuje šéfa zaměstnance jako deduktivní pravidlo a stanovuje integritní omezení pro plat zaměstnance, kdy plat zaměstnance nesmí být větší než plat jeho šéfa.

```
Zaměstnanec with
Rule
        ŠéfRule:        $forall z | Zaměstnanec m | Manažér
                        (exists o | Oddělení (z oddel o) and (o vedoucí m)) → ( z šéf m) $
constraint
        PlatHranice:    $ forall  z | Zaměstnanec b | Manažér x,y | Integer
                        (z šéf b) and (z plat x) and (b plat y) → x < y $
End
```

4. Závěr

Modelování znalostí pomocí deduktivních pravidel je základní ideou deduktivních databází. Deduktivní databáze, které jsou založeny na deduktivních pravidlech, spolu s objektovými, aktivními, temporálními, textovými a multimediálními databázemi zaznamenávají v současné době rychlý pokrok a to především proto, že jsou schopny věrněji modelovat realitu, efektivněji zpracovávat komplexní data a lépe zajišťovat nezávislost dat jak na uživatelských

aplikacích, tak na jejich implementaci. V souvislosti s deduktivními databázemi, potažmo s deduktivními pravidly, hovoříme o tzv. znalostní nezávislosti, která je dalším krokem k flexibilitě databází a jejich aplikací. Proto je problematika modelování znalostí s využitím deduktivních přístupů jistě aktuálním tématem.

Literatura:

1. Jeusfeld, M.A.: Update control in deductive object bases, Infix-Verlag, St. Augustin, Germany
2. Jarke, M., Gallersdörfer, R., Jeusfeld, M.A, Staudt, M., Eherer, S.: ConceptBase - a deductive object base for meta data management. In Journal of Intelligent Information Systems, Special Issue on Advances in Deductive Object-Oriented Databases, Vol. 4, No. 2, 167-192, 1995; draft appeared as technical report Aachener Informatik-Berichte 93-14
3. Jeusfeld, M.A., Krüger, E.: Deductive integrity maintenance in an object-oriented setting. Technical report MIP-9013, Universität Passau.
4. Telnarová, Z.: Metodická doporučení k integraci deduktivních pravidel do analýzy a návrhu informačního systému s databází. Rigorózní práce: VŠB - TU. 2001, 133 s.