Hatékony keresés a szemantikus világhálón

Lukácsy Gergely

Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

Magyarországi Web Konferencia 2008
W3C szekció

Motiváció

A Szemantikus Világháló két alapötlete

- adjunk metainformációkat (egységes alakban) erőforrásokról
  - adatszinten: adott képen egy delfin látható
  - sémaszinten: a delfinek állatok
  - (metainformációk eddig is voltak: oldalak fontossága, egyes kigyűjtött kifejezésekről azok helye/előfordulási gyakorisága, horgony, Word dokumentumok, képek)
- következtessünk a metainformációkon

Szemantikus keresés

- A keresés során vegyük figyelembe a dokumentumok és a felhasználói kérdés jelentését
- Információkeresés dokumentumok keresése helyett
- Kapcsolódó terület: szemantikus keresés meglévő információforrások (adatbázisok) felett
- Jellemzően: nagyon sok adat és kevés háttértudás

Az előadás vázlata

Az előadás vázlata

0. Bevezetés - matematikai alapok
   - Logikai programozás
   - Leíró logikák (Description Logics – DL)

1. Hatékony nyílt világ következtetés leíró logikákon
   - A PTTP specializálása DL klózokra
   - DL klózok fordítása Prolog programmá
   - A fordítás optimalizálása
   - A DLog rendszer megvalósítása

2. Összefoglalás

Az előadás vázlata

0. Bevezetés - matematikai alapok
   - Logikai programozás
   - Leíró logikák (Description Logics – DL)
1. Hatékony nyílt világ következtetés leíró logikákon
   - A PTTP specializálása DL klózokra
   - DL klózok fordítása Prolog programmá
   - A fordítás optimalizálása
   - A DLog rendszer megvalósítása
2. Összefoglalás

Logikai programozás

Prolog és kiterjesztései

- Logikai Programozás alapja: matematikai logika mint programozási nyelv
- A Prolog az első és máig a legelterjedtebb logikai programozási nyelv
- A logikai program végrehajtása elsőrendű következtetési folyamat
  - Prolog esetén ez a Horn-klózokon teljes SLD rezolúció
- Prolog végrehajtás kiterjeszthető teljes elsőrendű tételbizonyítóvá (PTTP)
  - kontrapozitív, előfordulás-ellenőrzés, ős-rezolúció, iteratívan mélyülő keresés

Prolog példaprogram

```
% valaki boldog, ha van okos és szép unokája is ugyanattól a gyermekétől
boldog(A) :- gyermeke(A, B), gyermeke(B, C), okos(C),
             gyermeke(B, D), szép(D).

okos(bence). szép(bence).
gyermeke(ági, kati). gyermeke(kati, bence).
```

Leíró logikai formalizmus

a $\mathcal{SHIQ}$ leíró logikai nyelv

- atomi fogalom: egyedek egy halmaza (unáris reláció)
  Könyv, Regény, Ember, $\top$, $\bot$
- atomi szerep: egyedek között fennálló bináris reláció
  címe, szerzője, gyermeke
- fogalomkonstruktorok: összetett fogalmak képzése atomiakból
  Könyv $\sqcap \neg$Regény, Okos $\sqcup$ Szép, $\{\exists, \forall\}$gyermeke.Okos, $(\geqslant 2\,$neje.Szép$)$
- szerepkonstruktor: inverzképzés
  szülője $\equiv$ gyermeke$^-$
- axiómák: fogalmi- és szerephierarchia, tranzitivitás

Leíró logikai tudásbázis = (T-doboz, A-doboz)

- T-doboz axiómák: Regény $\sqsubseteq$ Könyv, gyermeke $\sqsubseteq$ leszármazottja
- A-doboz axiómák: Ember(Kati), gyermeke(Kati, Bence)

Következtetés leíró logikákon

T-doboz következtetési feladatok

- fogalmak kielégíthetősége, alárendeltsége, ekvivalenciája, diszjunktsága
- a kielégíthetőséget hagyományosan tabló algoritmusokkal vizsgálják

A-doboz következtetési feladatok

- konzisztencia: van-e modellje egy $\mathcal{A}$ adatdoboznak egy $\mathcal{T}$ T-doboz felett?
- példányvizsgálat: igaz-e $\mathcal{A} \cup \mathcal{T} \models \alpha$, ahol α egy adatállítás?
  Következmény-e az, hogy $\forall$gyermeke.Okos(Kati)?
- példánykikeresés: meghatározandó a $\{i \mid \mathcal{A} \cup \mathcal{T} \models C(i)\}$ halmaz
  Mik a példányai a Ember $\sqcap \neg$Nőnemű fogalomnak?
- A C példánykikeresési feladat visszavezethető az összes $i_1, \ldots, i_n$ példány egyenkénti példányvizsgálatára
  1. hozzáadjuk az A-dobozhoz a $\neg C(i_k)$ adatállítást
  2. megvizsgáljuk az A-doboz konzisztenciáját
  3. inkonzisztencia esetén i_k biztosan a C fogalom példánya

Kérdés

Lehetséges-e olyan algoritmust készíteni, amely hatékonyan oldja meg a Leíró Logika $\mathcal{SHIQ}$ nyelvi példánykikeresési feladatát nagyméretű (azaz esetleg adatbázisban tárolt) A-doboz esetén?

Válasz

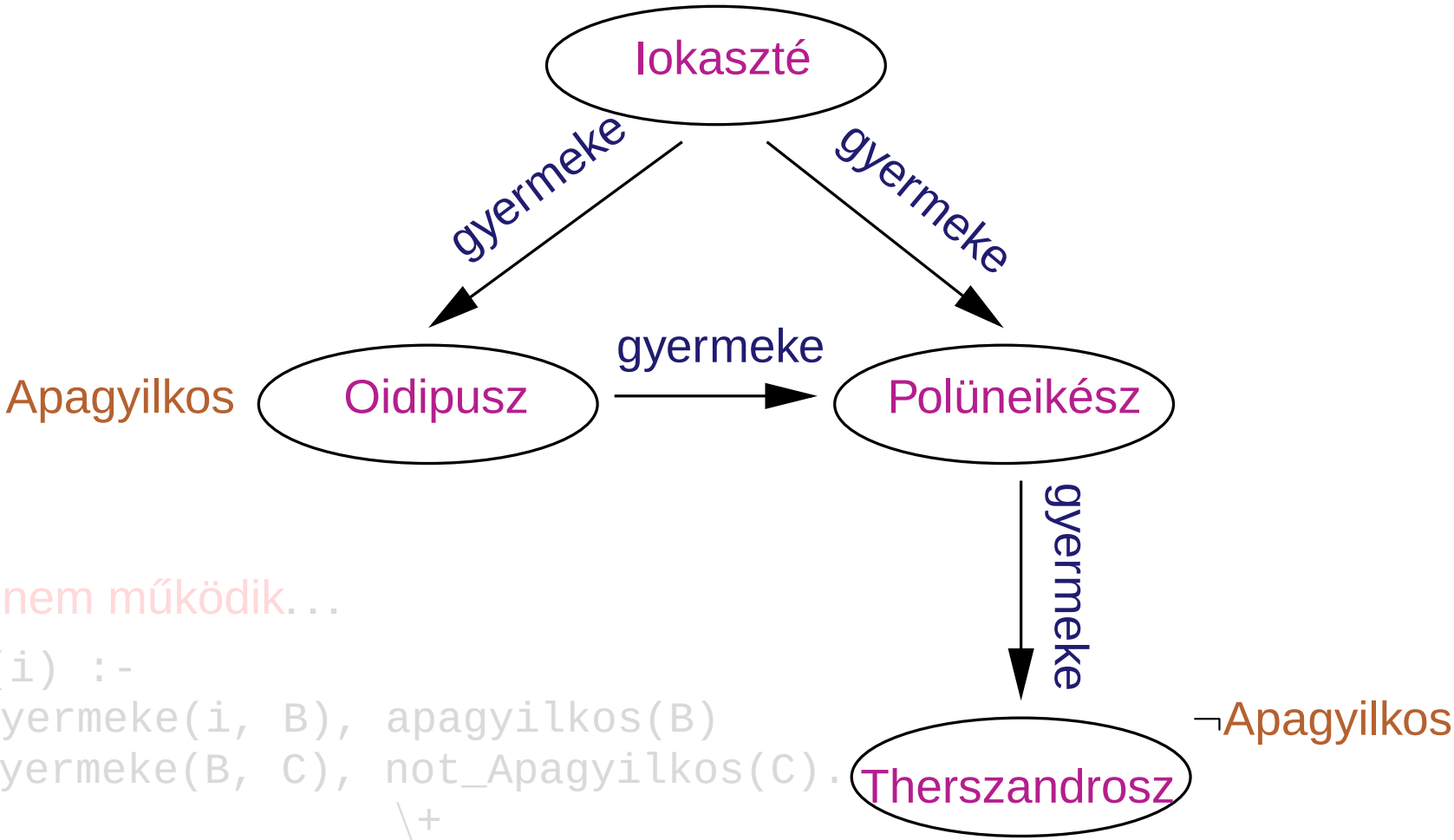
A DLog rendszer egy Prologban implementált rezolúciós alapú A-doboz következtető rendszer, amely a $\mathcal{SHIQ}$ Leíró Logikai nyelvre alkalmazható

- a DLog a $\mathcal{SHIQ}$ tudásbázisból optimalizált Prolog programot készít
- a DLog lekérdezések fókuszáltak és a következtetés tisztán kétfázísú
- a DLog lényegesen gyorsabb a létező következtetőknél
- a DLog szabadon letölthető és használható

Egy példa: a fordítás már T-doboz nélkül sem könnyű

Igaz-e, hogy

$\exists$gyermeke.(Apagyilkos $\sqcap$ $\exists$gyermeke.$\neg$Apagyilkos)(Iokaszté)



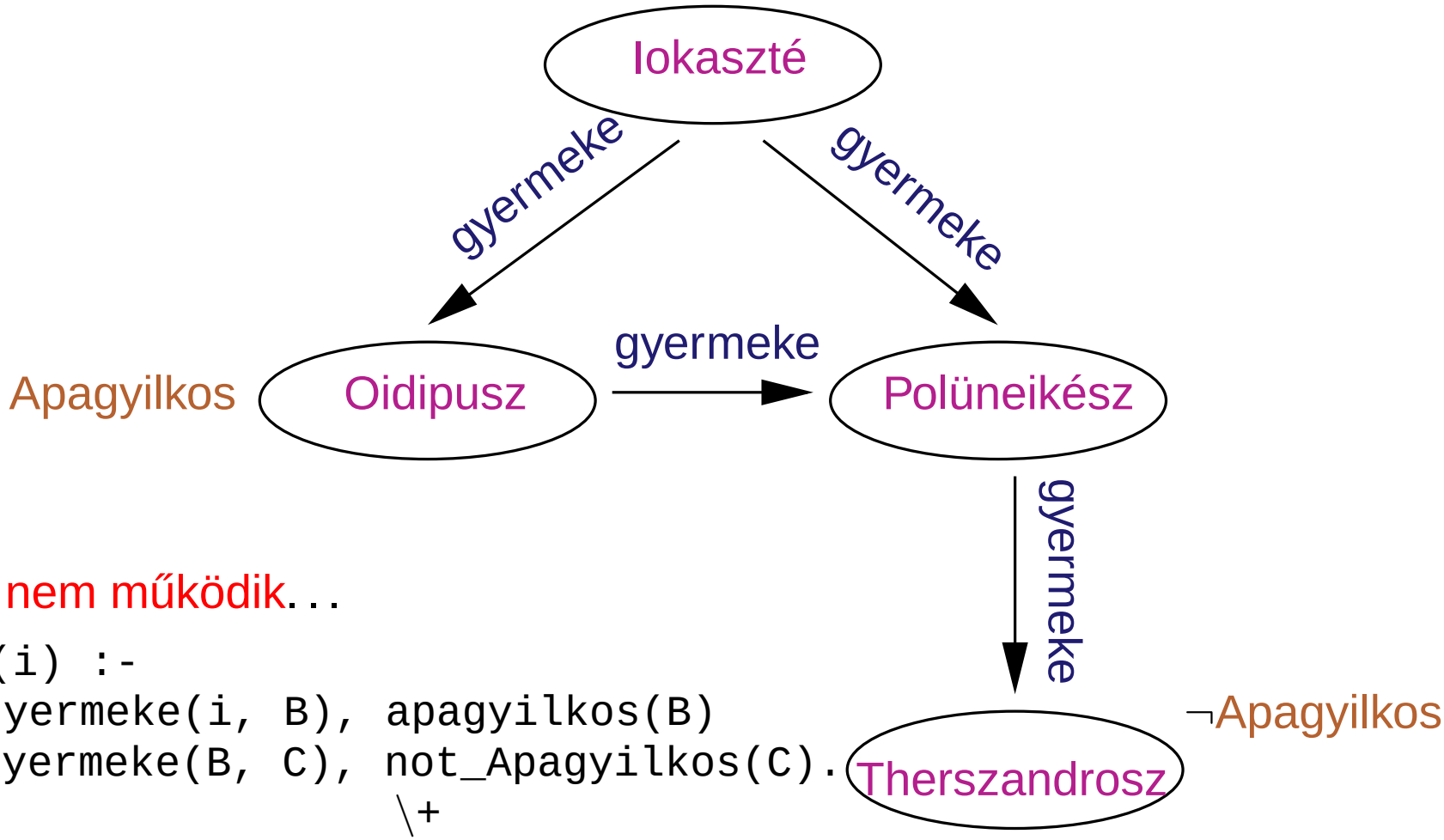
Ami nem működik...

```
Ans(i) :-
   gyermeke(i, B), apagyilkos(B)
   gyermeke(B, C), not_Apagyilkos(C).
                  \+
```

Egy példa: a fordítás már T-doboz nélkül sem könnyű

Igaz-e, hogy

$\exists$gyermeke.(Apagyilkos $\sqcap$ $\exists$gyermeke.$\neg$Apagyilkos)(Iokaszté)



Ami nem működik...

```
Ans(i) :-
   gyermeke(i, B), apagyilkos(B)
   gyermeke(B, C), not_Apagyilkos(C).
                   \+
```

Egy másik példa esetszétválasztásra: alkoholisták

Legyenek adottak a következő axiómák

$\exists$barátja.Alkoholista $\sqsubseteq \neg$Alkoholista (ha valakinek van alkoholista barátja, akkor ő nem alkoholista)

$\exists$szülője.$\neg$Alkoholista $\sqsubseteq \neg$Alkoholista (ha valakinek van nem alkoholista szülője akkor ő sem alkoholista)

szülője(i, sz1).
szülője(i, sz2).
barátja(sz1, sz2).

Igaz-e az alábbi?

$\neg$Alkoholista(i)

A DLog és a PTTP technológia

A DLog megközelítés dióhéjban

1/A a T-doboz → elsőrendű klózok [Motik06, Zombori07] ⊆ DL-klózok

1/B specializált PTTP technológia alkalmazása a DL-klózokra

2 kérdések futtatása hagyományos Prolog végrehajtással

A PTTP technológia specializálása DL-klózokra

	PTTP	DLog
előfordulás ellenőrzés	szükséges	nem szükséges (n.sz.)
keresés	iteratívan mélyülő	beépített Prolog + cikluseliminálás
kontrapozitívok képzése	összes	negált bináris fejűek esetén n.sz.
ős-rezolúció	minden predikátumra	bináris predikátumokra n.sz.
	nemdeterminisztikus	determinisztikus

DL klózok fordítása Prolog programmá

DL-klózok fordítási sémája

- Bemenet: DL klózok egy tetszőleges halmaza
- Kimenet: hagyományos módon végrehajtható Prolog program
- Tétel: az eredeti klózhalmaz és a kapott program példánykikeresési szempontból ekvivalens

Példa: $\exists$gyermeke.($\exists$gyermeke.szép $\sqcap$ $\exists$gyermeke.okos) $\sqsubseteq$ boldog

```
boldog(A, L)     :- member(C, L), C == boldog(A), !, fail.
boldog(A, L)     :- memberchk(not_boldog(A), L).
boldog(A, L)     :- gyermeke(A, B), gyermeke(B, C), okos(C),
                    gyermeke(B, D), szép(D).
...
not_okos(A, L) :- F = [not_okos(A)|L], gyermeke(C, A),
                    gyermeke(D, C), gyermeke(C, E), szép(E),
                    not_boldog(D, F).
```

A fordítás optimalizálása

A DLog 8(+2) optimalizálási transzformációt alkalmaz

1. szűrés: felesleges klózok eltávolítása (hamisan-, kétszeresen árva)
2. osztályozás: atomi-, kérdés-, árva-, általános klózok $\rightarrow$ egyedi fordítás
3. rendezés: klóztörzsek sorrendezése heurisztika alapján
4. indexelés: kétargumentumú Prolog indexelés
5. tömörítés: tömör cél igazságértékének hatékony meghatározása
6. dekompozíció: klóztörzs független komponensekre bontása
7. projekció: szuperhalmazok keresése $\rightarrow$ egyenkénti példányvizsgálat
8. szerepfordítás: cikluseliminálás kiküszöbölése szerepeknél
9. (parciális evaluálás): bejárati eljárások speciális fordítása
10. (speciális táblázás): már kiszámított ágak újrafelhasználása

Tétel: a fent definiált transzformációk helyesek és teljesek

Megvalósítás

Megvalósítottuk az előzőekben megfogalmazott módszereket a DLog rendszerben és elvégeztük a DLog részletes hatékonyságvizsgálatát.

	Kérdés	$\exists$gyermeke.(Apagyilkos $\sqcap$ $\exists$gyermeke.$\neg$Apagyilkos)(X)							
	Tesztfájl	c10	c20	c100	c1000	c10000	n1	n2	n3
DLog	betöltés	0.07	0.08	0.15	0.33	1.47	0.24	0.38	1.99
	futási idő	0.00	0.00	0.00	0.01	0.11	0.00	0.00	0.02
	szumma	**0.07**	**0.08**	**0.15**	**0.34**	**1.58**	**0.24**	**0.38**	**2.01**
KAON2	betöltés	0.45	-	-	-	-	0.60	0.97	2.36
	futási idő	0.72	-	-	-	-	4.72	63.60	425.17
	szumma	1.17	-	-	-	-	5.32	64.57	427.53
Racer	betöltés	0.01	0.01	0.03	0.51	4.68	0.10	0.68	6.04
	futási idő	0.07	0.09	0.15	1.68	79.91	0.47	1.76	23.25
	szumma	0.08	0.10	0.18	2.19	84.59	0.57	2.44	29.29
Pellet	betöltés	1.27	1.35	1.44	2.19	-	1.53	2.36	5.92
	futási idő	0.19	0.32	1.31	456.40	-	0.80	2.48	23.95
	szumma	1.46	1.68	2.76	458.58	-	2.33	4.84	29.87

Megvalósítás

Megvalósítottuk az előzőekben megfogalmazott módszereket a DLog rendszerben és elvégeztük a DLog részletes hatékonyságvizsgálatát.

	Kérdés	LQ_1				LQ_2			
	Tesztfile	f1	f2	f3	f4	f1	f2	f3	f4
DLog	betöltés	6.96	11.83	15.79	21.34	6.96	11.83	15.79	21.34
	futási idő	0.26	0.63	0.92	1.32	0.00	0.00	0.00	0.00
	szumma	**7.22**	**12.46**	**16.71**	**22.66**	**6.96**	**11.83**	**15.79**	**21.34**
KAON2	betöltés	6.56	13.56	20.66	28.73	6.46	13.56	20.66	28.73
	futási idő	0.70	0.99	1.33	1.69	0.66	0.93	1.27	1.62
	szumma	7.26	14.55	21.99	30.42	7.12	14.49	21.93	30.35
Racer	betöltés	24.84	91.57	X	X	24.84	91.57	X	X
	setup	29.41	112.29	X	X	29.41	112.29	X	X
	futási idő	2.69	5.89	X	X	4.07	7.49	X	X
	szumma	56.94	209.75	X	X	58.32	211.35	X	X
Pellet	beöltés	16.76	-	-	-	16.76	-	-	-
	setup	4.84	-	-	-	4.84	-	-	-
	futási idő	27.09	-	-	-	27.19	-	-	-
	szumma	48.69	-	-	-	48.79	-	-	-

Összefoglalás

- Fontos feladat, hogy meglévő információforrások felett legyünk képesek következtetéseket végezni
- A hagyományos megoldások nagyszámú példány esetén nem skálázódnak megfelelően $\rightarrow$ a weben nem igazán használhatók
- A DLog a leíró logikai T-dobozból Prolog programot készít; az adatokat a Prolog futás során éri el fókuszált módon
- A DLog rendszer hatékonyan képes következtetni nagyméretű adatdobozokon
- A DLog rendszer általánosan használható szemantikus alkalmazások következtető-motorjaként
  - letölthető a `http://dlog-reasoner.sourceforge.net` webcímen
  - használható a Protege ontológiaszerkesztővel
  - folyamatosan fejlődik (újabb optimalizálások, Java API, . . .)

Kérdések?

A DLog rendszerhez kapcsolódó publikációk

Gergely Lukácsy, Péter Szeredi.
Efficient description logic reasoning in Prolog: the DLog system.
Bírálat alatt a Theory and Practice of Logic Programming folyóiratnál.

Gergely Lukácsy, Zsolt Nagy, Péter Szeredi.
Using logic programs for description logic reasoning.
In Proceedings of Semantics 2006 - From Visions to Applications, pp. 113–125, Vienna, Austria, November 2006.

Zsolt Nagy, Gergely Lukácsy, Péter Szeredi.
Description logic reasoning using the PTTP approach.
In Proceedings of the 2006 International Workshop on Description Logics (DL2006), volume 189 of *CEUR*, pp. 183–191, Windermere, U.K., May 2006.

Zsolt Nagy, Gergely Lukácsy, Péter Szeredi.
Translating description logic queries to Prolog.
In Proceedings of the Practical Aspects of Declarative Languages, PADL 2006, volume 3819 of *Springer LNCS*, pp. 168–182, Charleston, USA, January 2006.