Magické transformácie

Ján Šturc

Jar, 2014

Magická transformácia

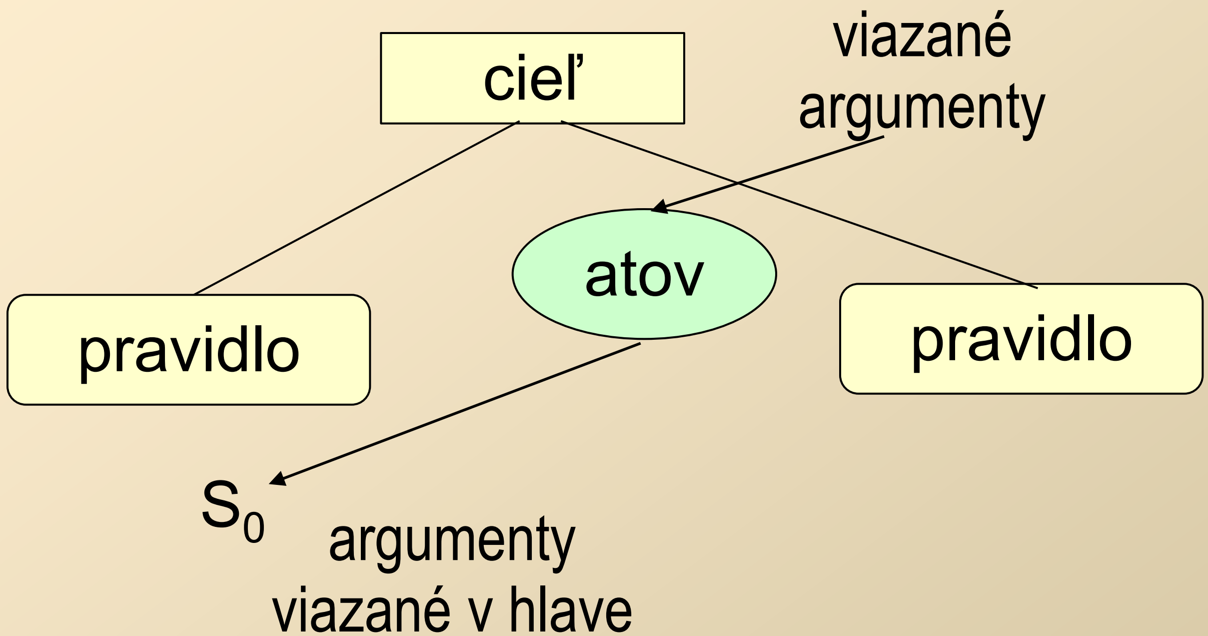
- Magická transformácia je optimalizačná technika, ktorá transformuje datalógový program na iný datalógový program s nasledujúcimi vlastnosťami:

1. Počíta tú istú odpoveď ako pôvodný datalógový program. Nie je to celkom pravda, keď pôvodný program neskončí.
2. Keď ho vypočítavame seminaívnou (naívnou) evaluáciou, do výpočtu vchádza tá istá množina faktov (n-tíc) ako pri výpočte zhora nadol (SLD-rezolúciou alebo RGT, či QRGT).
3. Selekcie (väzby argumentov) sa posúvajú od cieľov k pocieľom. Z hlavy pravidla postupne (z ľava do prava) k jednotlivým pocieľom v tele.

Od realizovateľného RGG k „mágii“

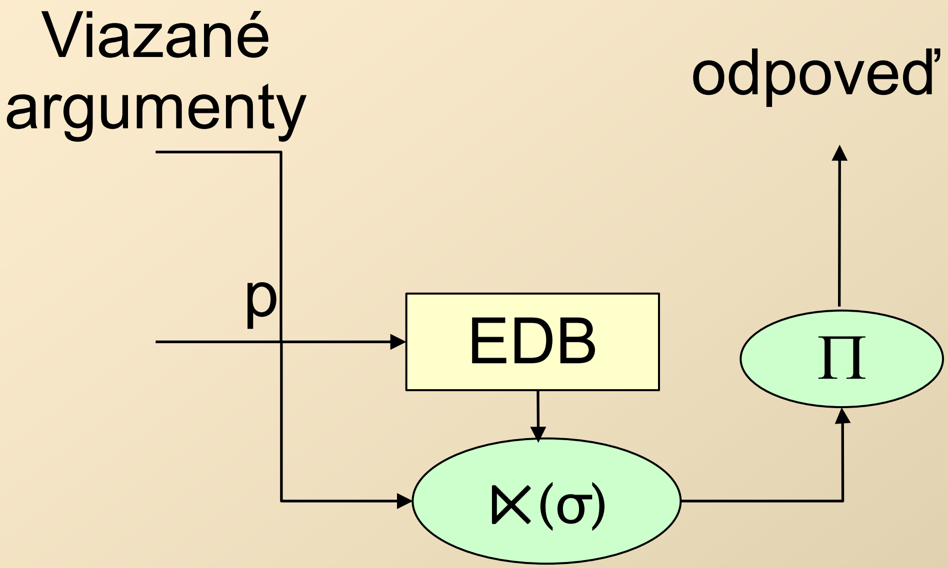
- Nech je daný realizovateľný RGG
- Každému IDB predikátu p priradíme magický predikát m_p.
- Argumenty predikátu m_p zodpovedajú viazaným argumentom predikátu p. Preddpokladáme, že p má uniformné ozdoby (unique binding pattern).
- m_p bude pravdivý práve pre tie n-tice, ktoré sa vyskytnú v niektorom uzle pre cieľ p pri „rozvíjaní“ zhora nadol.
- Pravidlovému uzlu r_{ij} priradíme pomocný predikát sup_{ij}.
- Argumentami sup_{ij} sú *viazané argumenty* pravidla („dosiahnuté“, v ozdobe pravidla sa nachádzajú [tu | …].) a *zaujímavé* premenné na dannon mieste pravidla.
- Premenná je zaujímava na mieste j ak sa vyskytuje v hlave, alebo v podcieľoch napravo miesta j (s indexami väčšími ako j).

Vzory toku dát (widgets) 1. od cieľa k hlave pravidla



Odovzdanie viazaných argumentov cieľa hlave pravidla.

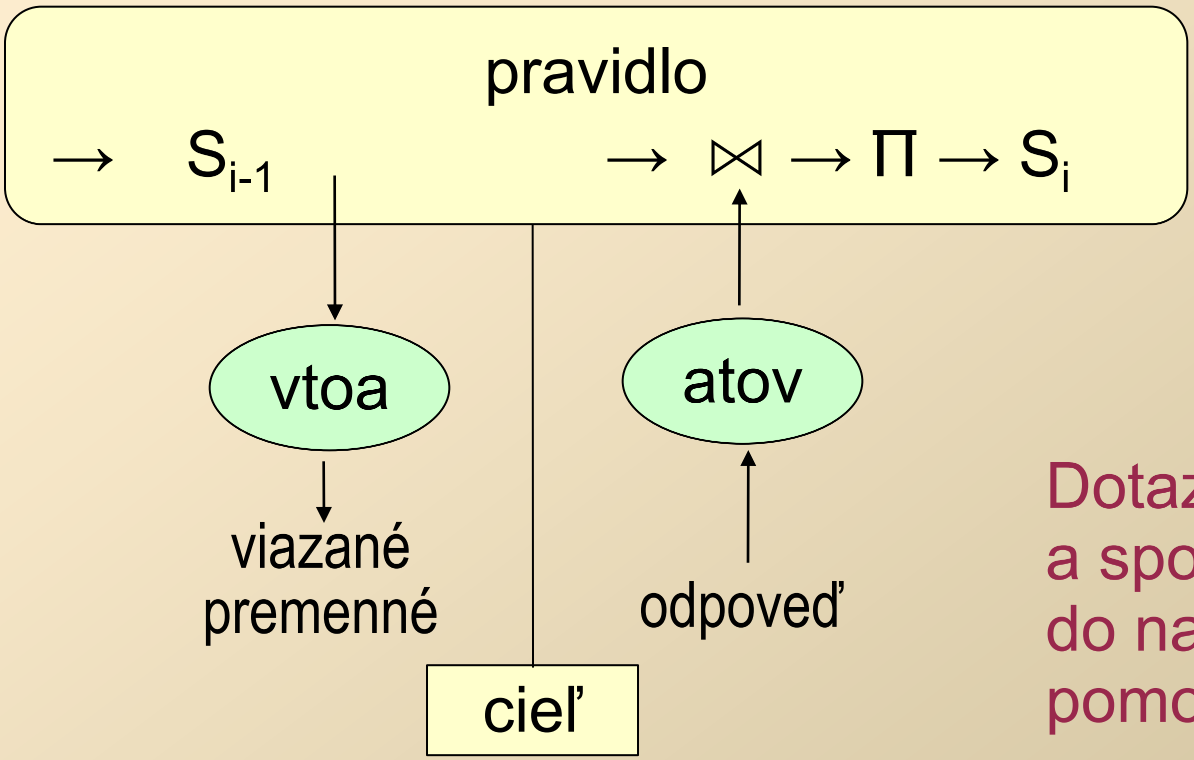
Vzory toku dát (widgets)
2. dotazy na EDB



Eventuálna projekcia na „non singleton“ premenné.

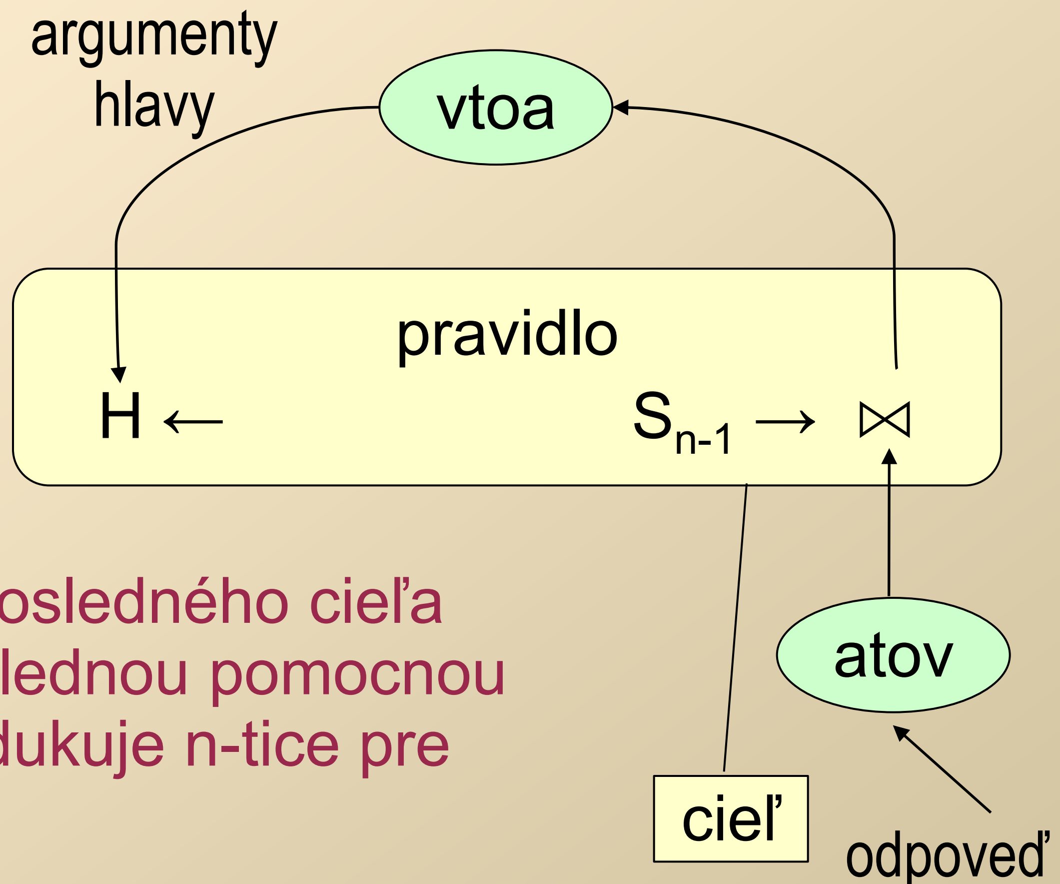
Detail dotazu na EDB predikát.

Vzory toku dát (widgets)
3. spracovanie v pravidle



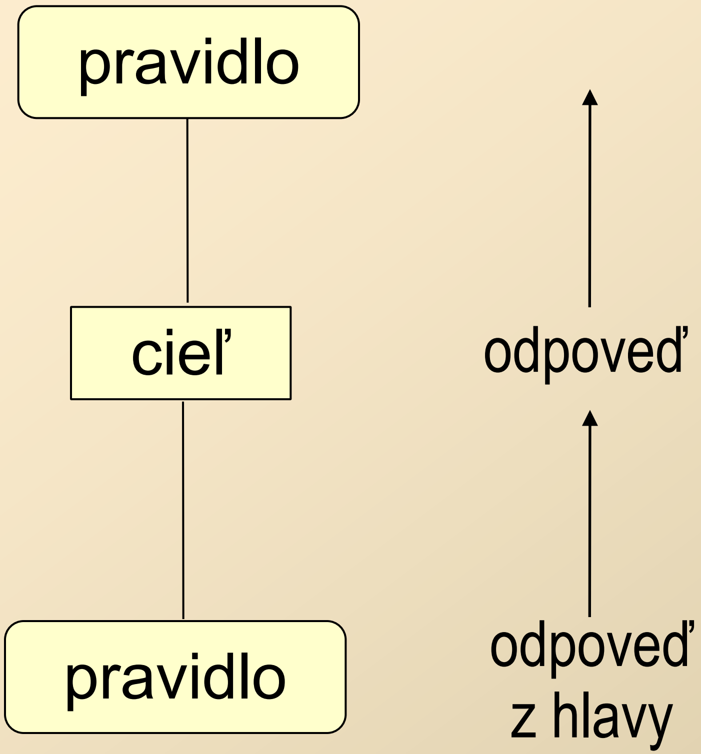
Dotaz na podcieľ
a spojenie výsledkov
do nasledujúcej
pomocnej relácie.

Vzory toku dát (widgets) 4. od posledného cieľa k hlave



Odpoveď od posledného cieľa sa spája s poslednou pomocnou reláciou a produkuje n-tice pre hlavu pravidla.

Vzory toku dát (widgets)
5. odpoveď od hlavy pravidla k cieľu



Odpovede sa posúvajú od hlavy pravidla syna k otcovi cieľu a od cieľa syna k pravidlu otcovi. Pokiaľ nedospejú do koreňa.

Päť skupín (groups) pravidiel pre magickú transformáciu

Pravidlá sú tvaru:

$H \leftarrow G_1, G_2, \ldots, G_n$

Dotaz: G(viazané argumenty, voľné argumenty)

Základnou myšlienkou magických transformácii je simulácia toku dát v grafe cieľov a pravidiel.

- Celý výpočet prebieha v pomocných predikátoch
- Magické predikáty slúžia ako filtre, ktoré pri výpočte zdola nahor nepustia do výpočtu n-tice, ktoré nie sú v súlade s ozdobou v hlave (slide 5)
- Skupiny pravidiel 1 – 4 približne zodpovedajú vzorom toku dát
- Semeno sú viazané premenne v koreni (dotaze)

Magické transformácie 1

1. skupina
Pomocný predikát určuje magický pre nasledujúci podcieľ.
Ak G_i má IDB predikát p: m_p := atov(G_i, $sup_{r,i-1}$)
m_p(viazané argumenty G_i) $\leftarrow$ $sup_{r,i-1}$(premenné)

2. skupina
Magický predikát určuje nultý pomocný.
Ak hlava pravidla má predikát q: $sup_{r,0}$:= vtoa(q, m_q)
$sup_{r,0}$(premenné) $\leftarrow$ m_q(viazané argumenty)

3. skupina
i -1. pomocný predikát a G_i určujú i. Pomocný.
$sup_{r,i} := sup_{r,i-1} \bowtie G_i$
$sup_{r,i}$(premenné) $\leftarrow$ $sup_{r,i-1}$(premenné), G_i(argumenty)

Magické transformácie 2

4.skupina
Posledný pomocný predikát a posledný podcieľ určuje hlavu. $H := atov(H, \ sup_{r,n-1} \bowtie G_n)$

$H(argumenty) \leftarrow \ sup_{r,n-1}(variables), G_n(argumenty)$

5. skupina
Inicializácia (semeno, seed). Ak dotaz je na predikát p, pridáme pravidlo:

m_p(viazané argumenty dotazu).

Je to jediné pravidlo, ktoré závisí na dotaze (the actual query). Ostatné pravidlá závisia len od pôvodných pravidiel (rules) a väzieb (binding patterns).

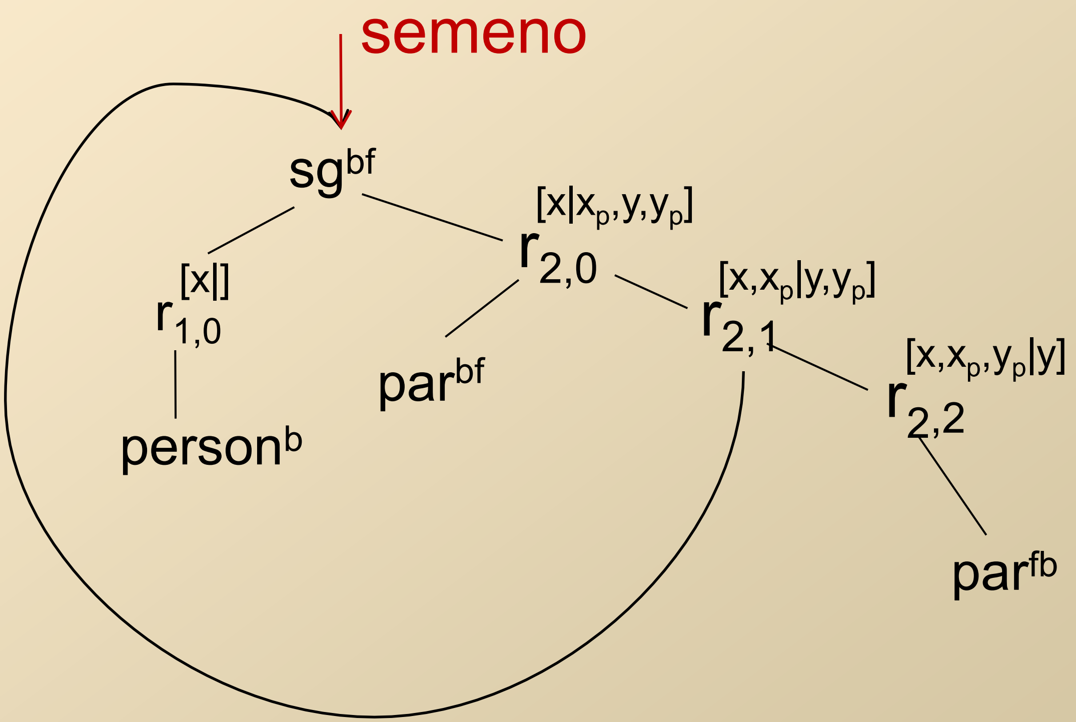
Možno názornejšie je

1. Začať semenom (5. skupina)
2. Priradiť hranám postupujúcim od cieľa k pravidlu pravidlá pre nulté pomocné predikáty (2. skupina)
3. Priradiť vnútropravidlovým hranám postupové pravidlá $sup_{r,i} \leftarrow sup_{r,i-1}, G_i$ (3. skupina)
4. Spracovať konce pravidiel – posledné pravidlové uzly (3. skupina)
5. Nakoniec definovať magické predikáty pre hrany vedúce do intencionálnych predikátov (v rekurzívnych pravidlách sú to spätné hrany, môžu to byť aj dopredné hrany, ak pravidlo obsahuje „cudzí intencionálny predikát“. mag_p ← sup... (1. skupina).

Pri správnom usporiadaní podcieľov a trochu praxe sa to dá robiť aj bez kreslenia RGG.

Príklad tá istá generácia RGG

r_1: $sg(x,x) \leftarrow person(x)$
r_2: $sg(x,y) \leftarrow par(x,x_p),sg(x_p,y_p),par(y,y_p)$
?- $sg(j,y)$



Príklad tá istá generácia „mágia“

1. skupina
$m\_sg(x_p) \leftarrow sup_{2,1}(x,x_p)$
2. skupina
$sup_{1,0}(x) \leftarrow m\_sg(x)$
$sup_{2,0}(x) \leftarrow m\_sg(x)$
3. skupina
$sup_{2,1}(x,x_p) \leftarrow sup_{2,0}(x), par(x,x_p)$
$sup_{2,2}(x,y_p) \leftarrow sup_{2,1}(x,x_p), sg(y,y_p)$
4. skupina
$sg(x,x) \leftarrow sup_{1,0}(x), person(x)$
$sg(x,y) \leftarrow sup_{2,2}(x,y_p), par(y, y_p)$
5. semeno
$m\_sg(j).$

Inak

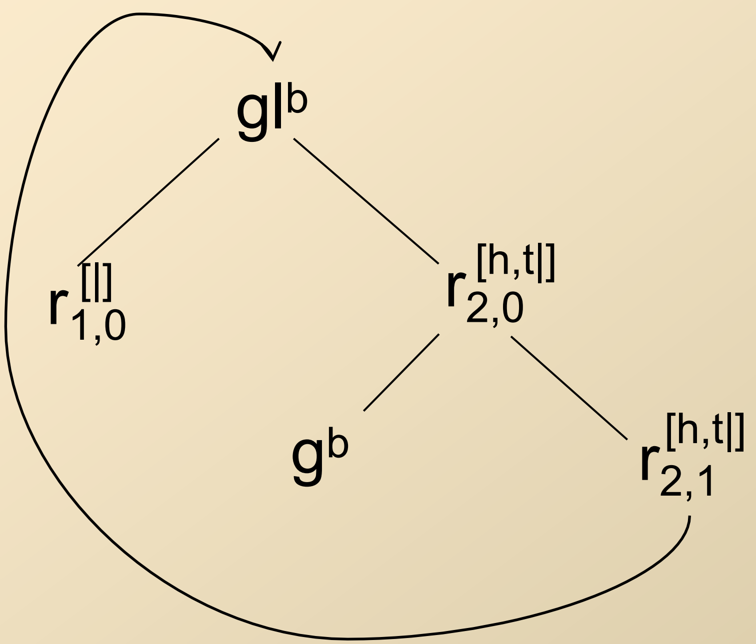
Dôležité sú nulté pomocné predikáty, ktoré filtrujú výpočty pravidieľ.

r_1: $sg(x,x) \leftarrow sup_0(x), person(x)$
r_2: $sg(x,y) \leftarrow sup_0(x), par(x,x_p), sg(x_p,y_p), par(y,y_p)$
r_3: $sup_0(x) \leftarrow x=j$
r_4: $sup_0(x) \leftarrow sup_0(z), par(z,x)$

Kúzla, čary, mágia? Nie! Len jednoduchá optimalizácia pomocou filtra sup_0, ktorý riadi, ktoré hodnoty x pustíme do výpočtu.

Príklad „good list“

r_1: gl(nil)
r_2: gl(cons(h,t)) $\leftarrow$ g(h), gl(t).
?- gl(l_0)



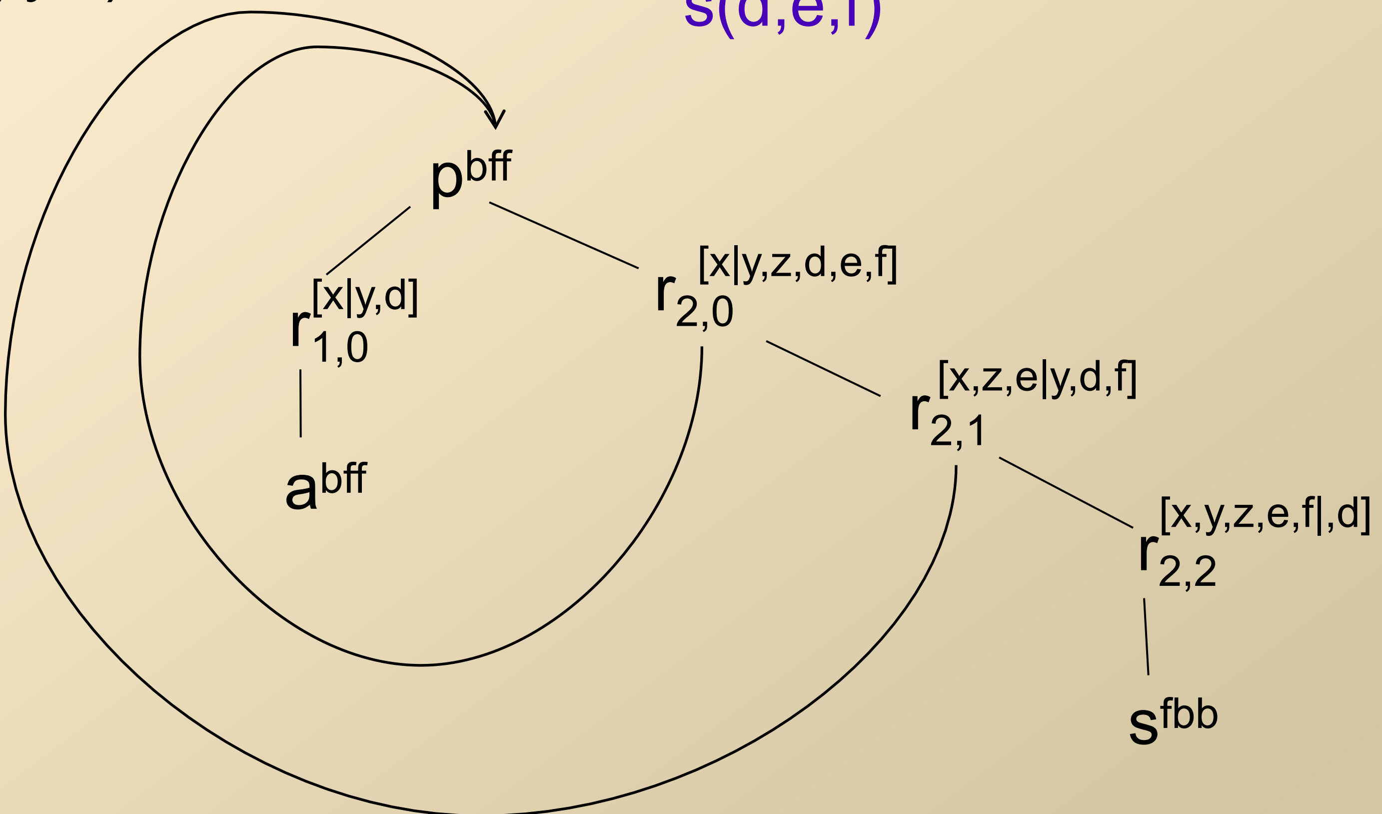
1. skupina
   m_gl(t) $\leftarrow$ $sup_{2,1}$(h,t).
2. skupina
   $sup_{1,0}$ $\leftarrow$ m_gl(nil)
   $sup_{2,0}$(h,t) $\leftarrow$ m_gl(cons(h,t)).
3. skupina
   $sup_{2,1}$(h,t) $\leftarrow$ $sup_{2,0}$(h,t), g(h).
4. skupina
   gl(nil) $\leftarrow$ $sup_{1,0}$
   gl(cons(h,t)) $\leftarrow$ $sup_{2,1}$(h,t), gl(t).
5. semeno
   m_gl(l_0).

Príklad – ohodnotené cesty v grafe

r1: $p(x,y,d) \leftarrow a(x,y,d)$

r2: $p(x,y,d) \leftarrow p(x,z,e), p(z,y,f), \underbrace{d=e+f}_{s(d,e,f)}$

?- $p(x_0,y,d)$



Ohodnotené cesty v grafe - „mágia“

1. skupina
$m\_p(x) \leftarrow sup_{2,0}(x)$
$m\_p(z) \leftarrow sup_{2,1}(x,z,e)$
2. skupina
$sup_{1,0}(x) \leftarrow m\_p(x)$
$sup_{2,0}(x) \leftarrow m\_p(x)$
3. skupina
$sup_{2,1}(x,z,e) \leftarrow sup_{2,0}(x), p(x,z, e)$
$sup_{2,2}(x,y,e,f) \leftarrow sup_{2,1}(x,z,e), p(z,y,f)$
4. skupina
$p(x,y,d) \leftarrow sup_{1,0}(x), a(x,y,d)$
$p(x,y,d) \leftarrow sup_{2,2}(x,y,e,f), d=e+f$
5. semeno: $m\_p(x_0)$

Nerekurzívny príklad v SQL.

select Z1.meno
from Zamestnanci Z1
where Z1.funkcia = 'IT špecialista' **and**
Z1.plat < (**select** AVG(Z2.plat)
from Zamestnanci Z2
WHERE Z2.oddelenie = Z1.oddelenie);

Prepis do datalógu:
EDB: Z(m,o,f,p)
r_1: ppo(o,a) ← subtotal(Z(m,o,f,p); o; a=avg(plat))
r_2: ppz(m,f) ← Z(m,o,f,p), ppo(o,a), p<a
?- ppz(m,'IT')

Pokračovanie RGG

ppz^{fb}

$r_{2,0}^{[f|m,o,p,a]}$

Z^{ffbf}

$r_{2,1}^{[f,m,o,p|a]}$

ppo^{bf}

$r_{2,2}^{[f,m,o,pa|]}$

$r_{1,0}^{[o|m',p',a]}$

$<^{bb}$

Z^{fbff}

$r_{1,1}^{[o,m',p'|a]}$

subtotal

Z^{bbbb}

Pokračovanie – magický program

1. skupina
$m\_ppo(o) \leftarrow sub_{2,1}(m,f,o,p)$
2. skupina
$sub_{1,0}(o) \leftarrow m\_ppo(o)$
$sub_{2,0}(f) \leftarrow m\_ppz(f)$
3. skupina
$sub_{1,1}(m',o,p') \leftarrow sub_{1,0}(o), Z(m',f',o,p')$
$sub_{2,1}(m,f,o,p) \leftarrow sub_{2,0}(o), Z(m,f,o,p)$
$sub_{2,2}(m,f,p,a) \leftarrow sub_{2,1}(m,f,o,p), ppo(o,a)$
4. skupina
$ppo(o,a) \leftarrow subtotal(sub_{1,1}(m',o,p'); o; a= avg(p'))$
$ppz(m,f) \leftarrow sub_{2,2}(m,f,p,a), p<a$
5. semeno: m_ppz('IT').

Zovšeobecnená magická transformácia

V niektorých prípadoch je nevhodné obmedziť magické predikáty len na viazané argumenty cieľov. Informácia v ostatných premenných sa dá pri výpočte využiť. Ponecháme preto magickým predikátom tie isté argumenty ako majú príslušné IDB predikáty.
Tým sa zmenia pravidlá prvej a druhej skupiny. V ostatných skupinách pribudnú potrebné premenné. Semeno sa zmení tak, že obsahuje všetky argumenty dotazu.

1. skupina: $m\_p(t_1, \ldots, t_n) \leftarrow sup_{j,i-1}(x_1, \ldots, x_m)$
2. skupina: $sup_{j,0}(x_1, \ldots, x_m) \leftarrow m\_p(t_1, \ldots, t_n)$
5. semeno: $m\_p(t_1, \ldots, t_n)$

Vypočet zovšeobecnených magických programov

- Semeno obsahuje neuzavreté termy pre volné argumenty. Programy a dáta môžu obsahovať termy
- Algebraický stroj (db-engine) vyžaduje miernú modifikáciu. Joiny a antijoiny sa robia podľa unifikácie a nie podľa rovnosti.
- Odstránenie duplikátov sa robí na základe subsumcie. Možno je dobré „duplikáty“ vôbec nevyrábať.
  – Vyhľadanie podľa rovnosti dokážeme robiť pomocou B-stromov, či hašovania.
  – Zovšeobecnenie pre unifikáciu a subsumciu je netriviálne. Asi všetky programy budú v najhoršom prípade kvadratické (nested loops).

Relation algebra revisited I.

- Storing relations
  - We tacitly assume that EDB relations contains ground terms only.
  - The IDB relations consists of three sections:
    - **Variable** a boolean value indicating if the relation contains a tuple consisting of mutually distinct variables.
    - **Non-ground terms**, where all tuples containing a non-ground term are stored. The names of variables in the particular tuple are considerable.
    - **Rest**, this is a ordinary relation containing only ground terms.
  - **Duplicates** are tuples subsumed by some other tuple.
  - Maybe,a good idea is to use **special function symbols $\mathfrak{f}^n$, which bind n arguments and to deal with unary relations only.**

Relation algebra revisited II.

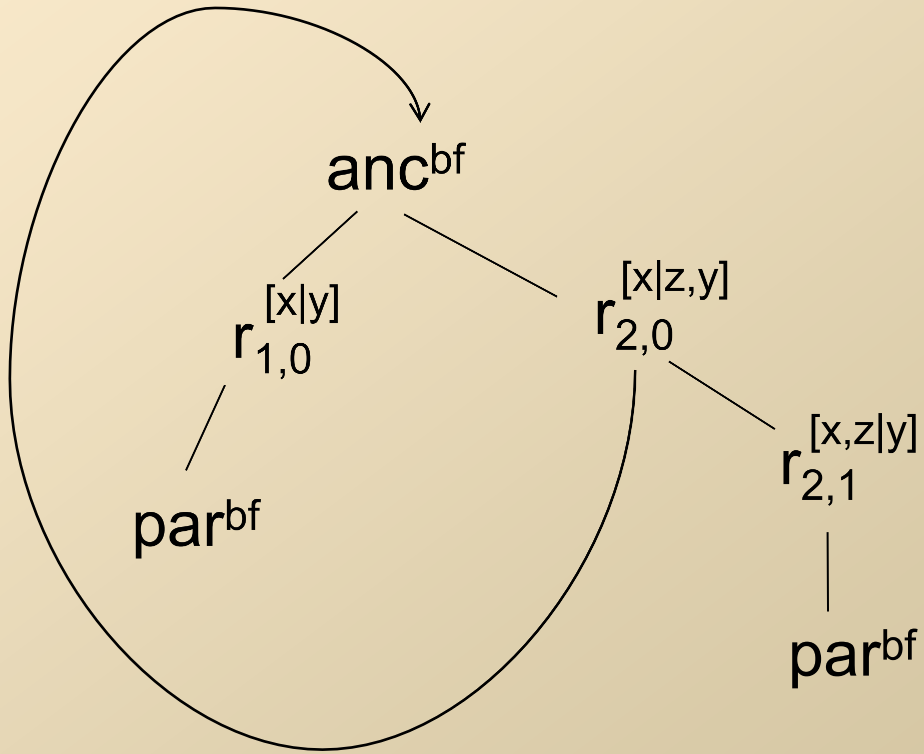
- Projection $\mathfrak{f}^k$ of the tuple $\mathfrak{f}^n$ on its $k \leq n$ arguments i_1, ..., i_k.
  $$\mathfrak{f}^k(X_{i1},..., X_{ik}) \leftarrow \mathfrak{f}^n(X_1, ..., X_n).$$
  Note, that this "rule" define only one tuple.
- Join and semijoin of two tuples $\mathfrak{f}^m(X_1, ..., X_m)$ and $\mathfrak{f}^n(Y_1, ..., Y_m)$ on $i_1=j_1$, ..., $i_k=j_k$.
  1. Compute projections $\mathfrak{f}^k(X_{i1},..., X_{ik})$ and $\mathfrak{f}^k(Y_{i1},..., Y_{ik})$.
  2. Unify these tuples as terms using week unification. Let τ be mgu if exists.
  3. If the tuples do not unify their join and semijoin is empty set. Else $\mathfrak{f}^m \bowtie \mathfrak{f}^n = \{\mathfrak{f}^{m+n-k}(X_1, ..., X_m, \mathbf{Y}) \circ \tau\}$ and $\mathfrak{f}^m \bowtie \mathfrak{f}^n = \{\mathfrak{f}^{m+n-k}(X_1, ..., X_m) \circ \tau\}$, where $\mathbf{Y}$ are arguments of the second tuple except selected k ones.

Zjednodušenia a drobné optimalizácie

- Pravidlá druhej skupiny nepotrebujeme. Namiesto nultých pomocných predikátov môžeme rovno použiť magické predikáty.
- Pravidlá, kde ľavá strana subsumuje pravú možno vynechať.
- Vo vnútri postupnosti po sebe idúcich EDB predikátov sú pomocné predikáty zbytočné (na celú konjunkciu sa môžeme dívať ako na jeden EDB predikát) a netreba ich generovať (t.j. ani pravidlá pre nich).
- Ak sa pomocný predikát nachádza pred EDB predikátom, vyskytuje sa iba raz, môžeme ho eliminovať (vynechať pravidlo, kde je na ľavej strane a jeho výskyt nahradiť pravou stranou tohto pravidla).

Príklad – predkovia Johna

r_1: anc(x,y) $\leftarrow$ par(x,y) /* par(x,y): y je rodič x, anc – predok */
r_2: anc(x,y) $\leftarrow$ anc(x,z), par(z,y)
?- anc('j', y)



Predkovia Johna – magická a zjednodušená magická transformácia

(1) $\cancel{m\_anc(x) \leftarrow sup_{2.0}(x)}$

(2) $\cancel{sup_{1.0}(x) \leftarrow m\_anc(x)}$

$\cancel{sup_{2.0}(x) \leftarrow m\_anc(x)}$

(3) $\cancel{sup_{2.1}(x,z) \leftarrow sup_{2.0}(x), anc(x,z)}$

(4) $anc(x,y) \leftarrow sup_{1.0}(x), par(x,y)$

$anc(x,y) \leftarrow sup_{2.1}(x,z), par(z,y)$

(5) $m\_anc('j').$

m_anc(x)

Po zjednodušení:

$anc(x,y) \leftarrow m\_anc(x), par(x,y).$

$anc(x,y) \leftarrow m\_anc(x), anc(x,z), par(z,y).$

$m\_anc('j').$

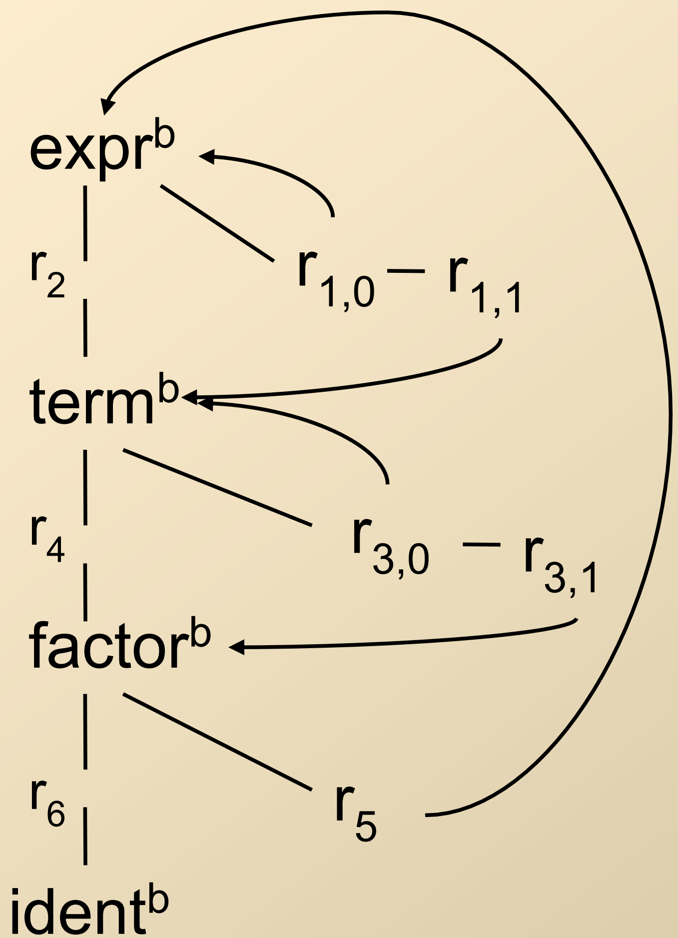
Príklad – syntaktická analýza

Pravidlá:

r_1: expr(plus(E,T)) $\leftarrow$ expr(E), term(T).
r_2: expr(T) $\leftarrow$ term(T).
r_3: term(mult(T,F)) $\leftarrow$ term(T), factor(F).
r_4: term(F) $\leftarrow$ factor(F).
r_5: factor(bra(E)) $\leftarrow$ expr(E).
r_6: factor(F) $\leftarrow$ ident(F).

Cieľ: $expr^b$ *rozpoznanie konkrétneho výrazu* ***e****.*

Príklad – syntaktická analýza (graf cieľov a pravidiel)



Pretože $expr^b$ znamená, že dekomponujeme známy výraz. Všetky zaujimavé premenné v pravidlách sú viazané. Ozdoba párnych pravidiel je [x|] a ozdoba pravidlových uzlov s nepárnym prvým indexom [x,y|].

Príklad – syntaktická analýza (magické pravidlá)

1. skupina:

$m\_expr(E) \leftarrow sup_{1,0}(E,T).$
$m\_expr(E) \leftarrow sup_{5}(E).$
$m\_term(T) \leftarrow sup_{2}(T).$
$m\_term(T) \leftarrow sup_{1,1}(E,T).$
$m\_term(T) \leftarrow sup_{3,0}(T,F).$
$m\_factor(F) \leftarrow sup_{4}(F).$
$m\_factor(F) \leftarrow sup_{3,1}(T,F).$
$m\_ident(I) \leftarrow sup_{6}(I).$

2. skupina:

$sup_{1,0}(E,T) \leftarrow m\_expr(plus(E,T)).$
$sup_{3,0}(T,F) \leftarrow m\_term(mult(T,F)).$
$sup_{5}(E) \leftarrow m\_factor(bra(E)).$
$sup_{2}(T) \leftarrow m\_expr(T).$
$sup_{4}(F) \leftarrow m\_term(F).$
$sup_{6}(I) \leftarrow m\_factor(I).$

Príklad – syntaktická analýza
(magické pravidlá – pokračovanie)

3. skupina: $sup_{1,1}(E,T) \leftarrow sup_{1,0}(E,T), expr(E).$
$sup_{3,1}(T,F) \leftarrow sup_{3,0}(T,F), term(T).$

4. skupina: $expr(E) \leftarrow sup_{1,1}(E,T), term(T).$
$expr(E) \leftarrow sup_{2}(E), term(E).$
$term(T) \leftarrow sup_{3,1}(T,F), factor(F).$
$term(T) \leftarrow sup_{4}(T), factor(T).$
$factor(F) \leftarrow sup_{5}(F), expr(F).$
$factor(F) \leftarrow sup_{6}(F), ident(F).$

5. semeno: m_expr(**e**).

Pozn.: **e** je zadaný výraz.

Príklad – syntaktická analýza (zovšeobecnená mágia)

```
m_expr(E)    ← m_expr(plus(E,T)).
m_expr(E)    ← m_factor(bra(E)).
m_term(T)    ← m_expr(T).
m_term(T)    ← m_expr(plus(E,T)), expr(E).
m_term(T)    ← m_term(mult(T,F)).
m_factor(F)  ← m_term(F).
m_factor(F)  ← m_term(mult(T,F)),term(T).
expr(E)      ← m_expr(plus(E,T)), expr(E), term(T).
expr(E)      ← m_expr(E), term(E).
term(T)      ← m_term(mult(T,F)), term(T), factor(F).
term(T)      ← m_term(T), factor(T).
factor(F)    ← m_factor(bra(F)), expr(F).
factor(F)    ← m_factor(F), ident(F).
m_expr(e).
```

Príklad – syntaktická analýza (O čom to vlastne je ?)

- Magické predikáty zariadia, že do výpočtu môžu vstupovať len podvýrazy zadaného výrazu **e**.
- Jediný extenzionálny predikát ident slúži na kontrolu, či identifikátor bol deklarovaný.
- Oproti programu syntaktickej analýzy je tu len malá neefektívnosť v tom, že do m_term a m_factor sa dostávajú aj podvýrazy **e**, ktoré nie sú termy resp. faktory.
- Všimnite si, že netransformovaný program a program pre $expr^f$ sa môže eventuálne zacykliť. Vyžaduje dodatočnú inteligenciu na to, aby v konečnom čase vygeneroval všetky výrazy kratšie než zadaná dĺžka.

Špecializácia rektifikáciou

- Niekedy netreba mágia stačí dôsledná rektifikácia programu včítane dotazu.

r_1: anc(X,Y) ← par(X,Y) /*par(x,y): y je rodič x, anc – predok*/

r_2: anc(X,Y) ← anc(X,Z), par(Z,Y)

?- anc('j', Y)

r_3: ancj(Y) ← par(j,Y).

r_4: ancj(Y) ← ancj(Z), par(Z,Y)

?-ancj(Y)

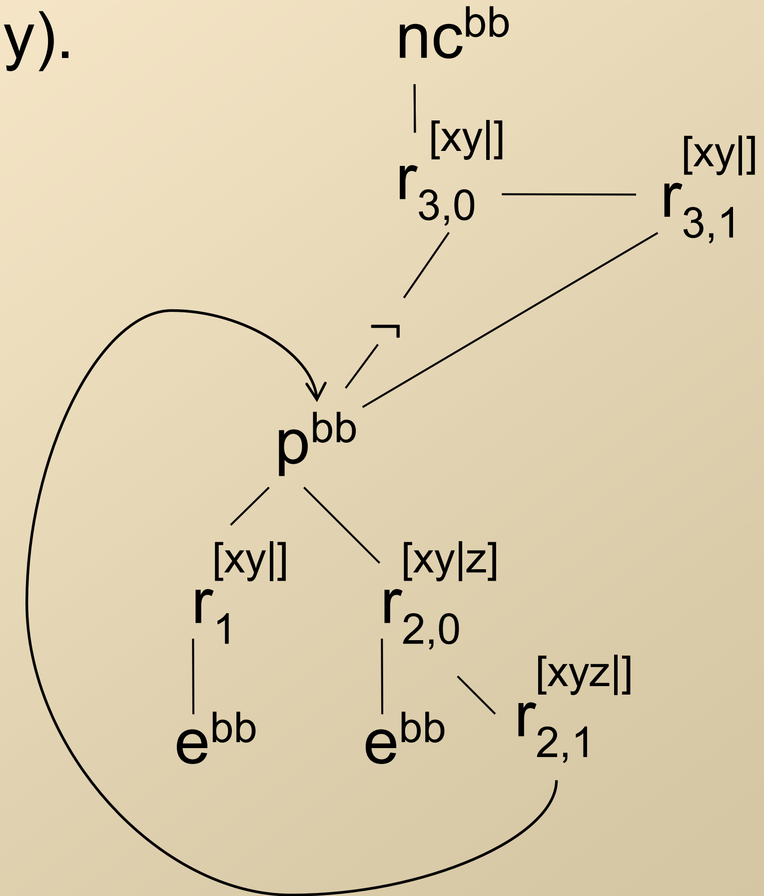
- Zistite, ktoré z programov tejto prednášky sa dajú optimalizovať rektifikáciou!

Ako je to s negáciou ?

r_1: $p(x,y) \leftarrow e(x,y)$.
r_2: $p(x,y) \leftarrow e(x,z), p(z,y)$.
r_3: $nc(x,y) \leftarrow \neg p(y,x), p(x,y)$.

?- nc(a,d)

stratifikovaný program



Po magickej transformácii

$p^{bb}(x, y) \leftarrow m\_p^{bb}(x,y),e(x,y).$
$p^{bb}(x, y) \leftarrow m\_p^{bb}(x,y),e(x,z),p^{bb}(z,y).$
$nc^{bb}(x,y) \leftarrow m\_nc^{bb}(x,y),\neg p^{bb}(y,x),p^{bb}(x,y).$
$m\_p^{bb}(y,x) \leftarrow m\_nc^{bb}(x,y).$
$m\_p^{bb}(x,y) \leftarrow m\_nc^{bb}(x,y),\neg p^{bb}(y,x).$
$m\_p^{bb}(z,y) \leftarrow m\_p^{bb}(x,y),e(x,z).$
$m\_nc(a,d).$

Program nie je stratifikovaný: $p^{bb} \rightarrow m\_p^{bb} \rightarrow \neg p^{bb}$.

Všimnite si, že ak dodržite pravidlo, písať negované predikáty až po všetkých pozitívnych predikátoch, sa môže podariť konštrukcia programu, kde magická transformácia nezachováva stratifkovanosť, iba ak program obsahuje pravidlo s aspoň dvomi negovanými predikátmi.

Mágia a „well-founded“ sémantika

$p(a) \leftarrow q(a), \neg r(a).$
$q(a) \leftarrow \neg p(a).$ EDB = Ø.
$r(a).$ wf. model $M=\{r(a), q(a), \neg p(a)\}$
? - q(a)

Po magickej transformácii:

$p(a) \leftarrow m\_p(a), q(a), \neg r(a).$
$q(a) \leftarrow m\_q(a), \neg p(a).$
$r(a) \leftarrow m\_r(a).$
$m\_p(a) \leftarrow m\_q(a).$
$m\_q(a) \leftarrow m\_p(a).$
$m\_r(a) \leftarrow m\_p(a), q(a).$
$m\_q(a).$ wf. model $M'=\{m\_q(a), m\_p(a)\}$

Nezachováva ani M, ani správnu odpoveď na dotaz.

Diskusia

- Príklad je značne umelý
  - Použitie magickej transformácie na program, ktorý neobsahuje premenné.
  - Fakt r(a) je postulovaný v programe namiesto, aby bol uložený v EDB.
- Naznačuje to, že pri použití magickej transformácie na programy s negáciou, musíme byť opatrní a najprv zhodnotiť výsledok, či je korektný.
- Možný jej aj taký prístup, že vyšpecifikujeme dostatočne širokú podtriedu datalógových programov, pre ktoré magickou transformáciou nedochádza k zmene sémantiky, alebo aspoň odpoveď sa zachová.

Literatúra

S. Abiteboul, R. Hull, V. Vianu: *Foundations of Databases.* Adison Wesley, Reading MA,1995

J. D. Ullman: *Database and Knowledge-Base Systems*, Vol. II. Computer Science Press, New York, 1988

Website www-db.stanford.edu/~ullman/cs345-notes.html

C. Beeri, R. Ramakrishnan: *On the Power of magic.* In PODS, 1987.

D. B. Kemp, D. Srivastava, P. J. Stuckey: Bottom-up Evaluation and Query Optimization of Well-Founded Models. Theoretical Computer Science (1995) .

Niektoré projekty

Name	Evaluation	Syntactic Restrictions	Negation	Data Requirements
CORAL (Wisconsin)	Magic Templates	First-order	Modularly Stratified	Main Memory
LDL (Texas)	Magic Sets	First-order	Modularly Stratified	Main Memory
Glue-Nail (Stanford)	Magic Rewriting	Restricted HiLog Extensions	Well-founded	Main Memory
XSB	SLG	HiLog with F-O Optimizations	Well-founded	Main Memory