Jednoduchá mágia je naozaj jednoduchá

Vstup: datalogový program + dotaz

Kuchársky recept:

1. Rektifikovať program
2. (Toto nie je nutné.) Premenovať premenné v hlavách pravidiel tak, aby hlavy boli rovnaké
3. **Generovať pravidlá jednoduchej magickej transformácie**
4. (Toto nie je nutné.) Zjednodušiť vygenerovaný program

RGG netreba ani kresliť, pravidlá magického programu sa dajú jednoducho generovať:

- Každému IDB predikátu je priradený magický predikát
- Každému podcieľu je priradený supplementary predikát, ktorý obsahuje premenné, ktoré sú viazané pri vstupe do toho podcieľa

Príklad: cesty v grafe, variant 1

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow e(X, Z), p(Z, Y).$

Dotaz: ? $p(a, Y).$

Pravidlá 1. skupiny: volania IDB predikátov (nielen priamo rekurzívne) na pravých stranách definujú magický predikát pre daný podcieľ

- Magický predikát obsahuje premenné viazané pri volaní (t.j. pri vstupe do) IDB podcieľa (tieto pravidlá zodpovedajú hranám v RGG grafe, ktoré končia v IDB predikáte; pre každú hranu 1 pravidlo). V tomto prípade pri volaní druhého podcieľa v r2 je viazané to Z. Iné IDB podciele tam nie sú

m_p(Z) ← sup2.1(X, Z).

Príklad: cesty v grafe, variant 1

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow e(X, Z), p(Z, Y).$

Dotaz: ? $p(a, Y).$

Pravidlá 2. skupiny: vstupy do pravidiel

- Tieto pravidlá sú dosť nudné, lebo vyzerajú vždy rovnako. Hovoria, že pri vstupe do ktoréhokoľvek pravidla sa najskôr aplikuje magický filter (ktorý prináleží IDB predikátu, ktorý je definovaný tým pravidlom)

sup1.0(X) ← m_p(X).

sup2.0(X) ← m_p(X).

Príklad: cesty v grafe, variant 1

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow e(X, Z), p(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Pravidlá 3. skupiny: bublanie cez podciele v pravidlách

- sup predikáty obsahujú premenné, ktoré sú pri vstupe do toho podcieľa viazané

sup2.1(X, Z) ← sup2.0(X), e(X, Z).

Príklad: cesty v grafe, variant 1

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← e(X, Z), p(Z, Y).

Dotaz: ? p(a, Y).

Pravidlá 4. skupiny: prebublanie cez posledný podcieľ pravidla určuje hlavu toho pravidla

p(X, Y) ← sup1.0(X), e(X, Y).

p(X, Y) ← sup2.1(X, Z), p(Z, Y).

Príklad: cesty v grafe, variant 1

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow e(X, Z), p(Z, Y).$

Dotaz: ? p(a, Y).

Pravidlá 5. skupiny: inicializácia (konštanty v dotaze určujú počiatočný filter)

m_p(a).

Príklad: cesty v grafe, variant 1

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow e(X, Z), p(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Zjednodušený magický program (na zbavenie sa sup predikátov stačí urobiť zopár textových substitúcií):

m_p(a).

m_p(Z) ← m_p(X), e(X, Z).

p(X, Y) ← m_p(X), e(X, Y).

p(X, Y) ← m_p(X), e(X, Z), p(Z, Y).

Všimnite si podobnosť s pôvodným programom!

Príklad: cesty v grafe, variant 2

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow p(X, Z), p(Z, Y).$

Dotaz: ? $p(a, Y).$

Pravidlá 1. skupiny: volania IDB predikátov (nielen priamo rekurzívne) na pravých stranách definujú magický predikát pre daný podcieľ

- Magický predikát obsahuje viazané premenné IDB podcieľa, z ktorého v RGG vychádza hrana (nie nutne spätná hrana) do nejakého IDB predikátu. V tomto prípade pri volaní prvého podcieľa v r2 je viazané X. Pri volaní druhého podcieľa v r2 je viazané Z. Iné IDB podciele tam nie sú

m_p(X) ← sup2.0(X).

m_p(Z) ← sup2.1(X, Z).

Príklad: cesty v grafe, variant 2

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← p(X, Z), p(Z, Y).

Dotaz: ? p(a, Y).

Pravidlá 2. skupiny: vstupy do pravidiel

- Tieto pravidlá sú dosť nudné, lebo vyzerajú vždy rovnako. Hovoria, že pri vstupe do ktoréhokoľvek pravidla sa najskôr aplikuje magický filter (ktorý prináleží IDB predikátu, ktorý je definovaný tým pravidlom)

sup1.0(X) ← m_p(X).

sup2.0(X) ← m_p(X).

Príklad: cesty v grafe, variant 2

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow p(X, Z), p(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Pravidlá 3. skupiny: bublanie cez podciele v pravidlách

- sup predikáty obsahujú premenné, ktoré sú pri vstupe do toho podcieľa viazané

sup2.1(X, Z) ← sup2.0(X), p(X, Z).

Príklad: cesty v grafe, variant 2

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow p(X, Z), p(Z, Y).$

Dotaz: ? $p(a, Y).$

Pravidlá 4. skupiny: prebublanie cez posledný podcieľ pravidla určuje hlavu toho pravidla

$p(X, Y) \leftarrow sup1.0(X), e(X, Y).$

$p(X, Y) \leftarrow sup2.1(X, Z), p(Z, Y).$

Príklad: cesty v grafe, variant 2

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← p(X, Z), p(Z, Y).

Dotaz: ? p(a, Y).

Pravidlá 5. skupiny: inicializácia (konštanty v dotaze určujú počiatočný filter)

m_p(a).

Príklad: cesty v grafe, variant 2

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← p(X, Z), p(Z, Y).

Dotaz: ? p(a, Y).

Zjednodušený magický program (na zbavenie sa sup predikátov stačí urobiť zopár textových substitúcií):

m_p(a).

m_p(Z) ← m_p(X), p(X, Z).

p(X, Y) ← m_p(X), e(X, Y).

p(X, Y) ← m_p(X), p(X, Z), p(Z, Y).

Všimnite si podobnosť s pôvodným programom!

Príklad: cesty v grafe, variant 3

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← p(X, Z), e(Z, Y).

Dotaz: ? p(a, Y).

Pravidlá 1. skupiny: volania IDB predikátov (nielen priamo rekurzívne) na pravých stranách definujú magický predikát pre daný podcieľ

- Magický predikát obsahuje viazané premenné IDB podcieľa, z ktorého v RGG vychádza hrana (nie nutne spätná hrana) do nejakého IDB predikátu. V tomto prípade prvý podcieľ v r2 má viazanú premennú X. Iné IDB podciele tam nie sú

m_p(X) ← sup2.0(X).

Príklad: cesty v grafe, variant 3

Program:

r1: p(X, Y) ← e(X, Y).

r2: p(X, Y) ← p(X, Z), e(Z, Y).

Dotaz: ? p(a, Y).

Pravidlá 2. skupiny: vstupy do pravidiel

- Tieto pravidlá sú dosť nudné, lebo vyzerajú vždy rovnako. Hovoria, že pri vstupe do ktoréhokoľvek pravidla sa najskôr aplikuje magický filter (ktorý prináleží IDB predikátu, ktorý je definovaný tým pravidlom)

sup1.0(X) ← m_p(X).

sup2.0(X) ← m_p(X).

Príklad: cesty v grafe, variant 3

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow p(X, Z), e(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Pravidlá 3. skupiny: bublanie cez podciele v pravidlách

- sup predikáty obsahujú premenné, ktoré sú pri vstupe do toho podcieľa viazané

sup2.1(X, Z) ← sup2.0(X), p(X, Z).

Príklad: cesty v grafe, variant 3

Program:

r1: $p(X, Y) \leftarrow e(X, Y).$

r2: $p(X, Y) \leftarrow p(X, Z), e(Z, Y).$

Dotaz: ? $p(a, Y).$

Pravidlá 4. skupiny: prebublanie cez posledný podcieľ pravidla určuje hlavu toho pravidla

$p(X, Y) \leftarrow sup1.0(X), e(X, Y).$

$p(X, Y) \leftarrow sup2.1(X, Z), e(Z, Y).$

Príklad: cesty v grafe, variant 3

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow p(X, Z), e(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Pravidlá 5. skupiny: inicializácia (konštanty v dotaze určujú počiatočný filter)

m_p(a).

Príklad: cesty v grafe, variant 3

Program:

r1: $p(X, Y) \leftarrow e(X, Y)$.

r2: $p(X, Y) \leftarrow p(X, Z), e(Z, Y)$.

Dotaz: ? $p(a, Y)$.

Zjednodušený magický program (na zbavenie sa sup predikátov stačí urobiť zopár textových substitúcií):

m_p(a).

p(X, Y) ← m_p(X), e(X, Y).

p(X, Y) ← m_p(X), p(X, Z), e(Z, Y).

Vyzerá to možno zvláštne, ale magický filter m_p je vskutku nerekurzívny. Obsahuje len konštantu *a*.

Všimnite si podobnosť s pôvodným programom!

Príklad: cesty v grafe, rekapitulácia

? p(a, Y).

Program 1:
p(X, Y) ← e(X, Y).
p(X, Y) ← e(X, Z), p(Z, Y).

Program 2:
p(X, Y) ← e(X, Y).
p(X, Y) ← p(X, Z), p(Z, Y).

Program 3:
p(X, Y) ← e(X, Y).
p(X, Y) ← p(X, Z), e(Z, Y).

Magické programy:

m_p(a).
m_p(Z) ← **m_p(X), e(X, Z).**
p(X, Y) ← **m_p(X),** e(X, Y).
p(X, Y) ← **m_p(X),** e(X, Z), p(Z, Y).

m_p(a).
m_p(Z) ← **m_p(X), p(X, Z).**
p(X, Y) ← **m_p(X),** e(X, Y).
p(X, Y) ← **m_p(X),** p(X, Z), p(Z, Y).

m_p(a).
p(X, Y) ← **m_p(X),** e(X, Y).
p(X, Y) ← **m_p(X),** p(X, Z), e(Z, Y).