Deduktívne databáze (DDB)

- DB na úrovni FOL - First order logic (predikátová logika)
- systém, kt. poskytne matematickú logiku v spojení s relačnými DB
- potrebujeme ukladať veľa dát, preto ich nestačí mať v pamäti, ale nutne na disku => DB systém

- DB obsahuje fakty, ktoré odrážajú nejakú skutočnosť
- my potrebujeme ale určitý druh informácií **odvodiť z dát** na základe nejakých <u>odvodzovacích pravidiel</u>
  ↓
  pomocou určitej podmnožiny predikátovej logiky

napr. PROLOG používa backtracking a vracia 1 výsledok (po každom stisknutí klávesy ENTER↵)

- Pri DB očakávame nejaký agregovaný výsledok napr. celý zoznam zamestnancov

Riešenie

1.) Vložiť odvodzovacie pravidlá do aplikácie
   ↳ špatne čitateľné
   ↳ také apply sú veľmi náročné, navyše každá zmena je <u>drahá</u>

2.) Použiť **deduktívny** DB systém, kde samotné dáta rozdelíme na:
   - <u>fakty</u> (explicitne uložené predikáty)
   - <u>odvodzovacie pravidlá</u>
   - <u>odvodené fakty</u> (odvoditeľne uložené predikáty)
     ↳ vznikajú z explicitných faktov
     ↳ užívateľ ich zväčšinou nemá rozlíšiť od explicitných faktov

Rysy DDB systému

- spracováva veľké množstvo dát (viac ako PROLOG)
- umožňuje aplikovať deduktívne pravidlá nad dátami (logické, odvodz.)
  ↳ nie je to bežné v DB systémoch
- má jednoduchší spôsob výpočtu ako PROLOG
- jednoduchšie operácie sa rozložia ~~na~~ v podstate SQL príkazy
- je možné spraviť rekurziu

Jazyk DDBS

↳ tvorí <u>množina správne definovaných formulí</u> (wffs → <u>w</u>ell-<u>f</u>ormed <u>f</u>ormula<u>s</u>)

↳ predikátová logika funguje na princípe:

wffs + modus ponens + zobecnenie

↳ dedukciu potom môžeme vykonávať na základe:

a.) sémantiky (platnosti formule) - na základe vlast. nesplniteľnosti, kde pravidlá definujú možné modely, kt. sú potom predikátovo ohodnotené → {true, false}

b.) syntaktiky (platnej konštrukcii formule) -
  ↳ snaží sa dokázať formule uplatňovaním pravidiel ~~sa sa~~ a dôjsť k axiómom všetkými možnými spôsobmi (↓, ↑)

Predikátová logika (FOL)

→ budeme chcieť **uzavreté formule** (bez voľných premenných) na rozdiel od temporálnych DB → tvar formulí bude veľmi obmedzený (podobne ako v PROLOG)

- používa sa transformácia na prenexný tvar

$$u\check{c}\acute{\imath}(x,y) \rightarrow \exists z.\,diplomka(x,z) \longrightarrow \exists z(diplomka(x,z) \vee \neg u\check{c}\acute{\imath}(x,y))$$

Modus ponens: ak $P \wedge (P \to Q) \Rightarrow Q$

Pravidlo zovšeobecnenia: z P odvoď $\forall x.P$

(Herbrandov) model

↳ ohodnotenie, kt. je pravdivé pre každú formulu – **model**

- ohodn. je nekonečne veľa – nie každé ohodn. je modelom
- niekedy žiadne nie je modelom

Logický dôsledok

- W : správne formulované formuly (wffs)
- K : množina wffs

$K \vdash W \iff$ W je pravdivé pre všetky modely K

- ak pre množinu K nájdeme model, potom automaticky platí W.

→ splniteľnosť – existencia modelu pre formulu

→ platnosť – splniteľnosť ~~na~~ pre všetky ohodnotenia alebo vo všetkých interpretáciách

Relačný model nad logikou

- relácia je formula z predikátovej logiky

napr. osoba(ID, meno, vek, plat)

↳ konkrétne osoba(123, "Karol", 35, 30000)

- je možné previesť na predikát:

"Je v tabuľke osoba ~~...~~ s id = 123, meno = karol, vek = 35, plat = 30000?

→ vráť všetky osoby: osoba(_, name, age, wage)

→ vráť všetkých mladších zamestnancov: osoba(_, name, age > 14 && age < 45, wage)

↳ je nutné obmedzenie, pretože potenciálne možno dostať nekonečný výsledok – vek môže ísť do nekonečna

- funkcie sú špeciálne prípady relácií

$$y = f(x) \iff F(x,y)$$

$$F(x,y) \land F(x,z) \rightarrow (y = z)$$

Rôzny druh informácií

- explicitne uložené predikáty (EUP)
  - explicitné dáta, fakty
  - napr. osoba(1132, "Pepa", 45, 34500)
- odvoditeľne uložené predikáty (OUP)
  - odvodené dáta
  - napr. ak vek > 34 potom plat = 45000

    osoba(x,y,z,45000) :- osoba(x,y,z,w), z > 34.

Význam pravidiel → spôsob vyhodnotenia - ako získam výsledok?

1.) Dôkazná interpretácia (syntaktická)

- existencia axiómov v DB → samotné fakty/dáta (EUP)

  napr. vek(Pepa, 35)
- patria sem aj implicitné informácie, tzn. sú odvoditeľné z EUP a OUP
- negácia: pozitívne / negatívne predikáty

Dôkaz

↳ všetky fakty odvodené pomocou OUP sú odvodené pomocou modus ponens:

- osoba(1234, Pepa, 44, x)
- osoba(x,y,z,45000) :- osoba(x,y,z,w), z > 34.
- osoba(1234, Pepa, 44, 45000)

- uplatňuje sa dopredné odvodenie → od axiómu k teorému

2.) Modelová interpretácia (sémantická)

$\boxed{\text{interpretácia = ohodnotenie}}$

- pravidlá definujú možné <u>modely</u>
- <u>interpretácia/ohodnocovanie</u> – predikát priradzuje TRUE / FALSE každej možnej inštancii
- <u>modelom</u> množiny pravidiel je také **ohodnotenie**, v ktorom sú **všetky pravidlá pravdivé** – nezáleží na priradení hodnôt premenným

$\boxed{\text{Pr}}$

Minimálny model

Nech $M_1, \dots, M_n$ sú modely množiny S wffs.
Minimálny model je taký model M_k, ktorý:

$$M_k \subseteq M_j, \quad j \in \{1, \dots, k-1, k+1, \dots, n\}$$

$$\neg \exists M.\ M \text{ je modelom z } S \text{ a } M \subset M_k$$

- neexistuje menší model než minimálny
- môže ich byť viac, potom sú vzájomne neporovnateľné

Usporiadanie modelov

- hľadáme takú permutáciu, že

$$M_k = M_{i_1} \wedge M_{i_1} \subseteq M_{i_2} \subseteq \dots \subseteq M_{i_n}$$

- výhodou nájdenia minimálneho modelu je, že z neho dokážeme odvodiť všetky ostatné modely
- M_k obsahuje riešenie

Problém – negácia

$p(x) :- r(x) \land \neg q(x)$
$q(x) :- r(x) \land \neg p(x)$

DB: $\{r(1)\}$

$\xrightarrow[\text{2 minimálne modely}]{}$ $S_1 = \{q(1), r(1)\}$
$S_2 = \{p(1), r(1)\}$

- pre odvodenie všetkých výsledkov potrebujeme všetky minimálne modely, lenže my nevieme koľko ich je.
- ak nemáme negáciu, tak minimálny model je iba 1 a stáva sa aj najmenším. Potom je to jednoduché, lebo nájdeme deterministický postup, ktorým získame výsledné množinu.

3.) Výpočetná interpretácia

↳ algoritmus na "vykonanie" TRUE/FALSE pravidiel

↳ PROLOG

↳ algoritmus na <u>vyhľadanie dôkazu potenciálneho faktu</u>

↳ to, čo PROLOG považuje za pravdivé fakty nemusia byť vždy aj modely

↳ ale vo veľa prípadoch nám poskytne minimálny model

- PROLOG integrujeme s DB
- ak integrujeme <u>bez negácií</u> ⇒ výsledkom je jediný minimálny model a množina faktov odvoditeľných z DB
- ak integrujeme <u>s negáciou</u> (obmedzené možnosti)
  - 1 z veľa minimálnych modelov

Vnútorný model - možná varianta

- modelová interpretácia
- prologovské príkazy (atom. formule, pred. symboly, funkcie počít. hodnoty
- prologovské konvence

- logické príkazy (výrazy)
  ↳ Hornovy klauzule → $B :- A_1, A_2, \dots, A_n$

Pr

boss (M, E) :- riadi (M, E)

boss(M, E) :- boss(N, E), riadi (M, N).

Bezpečná DB - nie sú vidno citlivé informácie

Algoritmy

- ako mapovať odvodzovanie na sériu dotazov v SQL (rel. DB)?
- relácia je daná predikátovým symbolom ~~p~~ p: $p(A_1, \dots, A_n)$

premenná/konštanta   arita n

- potrebujeme si pamätať pozíciu premenných/konštánt a ich význam v p

Bezpečné pravidlá

- obsahujú na pravej strane premenné z ľavej strany

$$p(X, Y) :- q(X, U), r(U, Y).$$

- potrebujeme ich ~~...~~ pre tvorbu SQL dotazu
- nemôžme špecifikovať pravidlá, kde na ľavej strane je premenná, ktorá nie je na pravej strane
- všetky premenné z ľava sa MUSIA vyskytnúť niekde vpravo.

Logický program

- tvorený súborom Hornových klauzulí
- Pr X a Y sú súrodenci, ak existuje také Z, že je rodičom oboch X aj Y a zároveň X a Y sú rôzne individuá

súrodenci(X,Y) :- rodič(Z,X), rodič(Z,Y), X != Y.

Konverzia logických programov

- je treba zvoliť vhodné DB štruktúry pre reprezentáciu rôznych informácií:
  - EUP → v pohode
  - OUP → ?
  - závislosť medzi predikátmi
    ↳ určujú celý algoritmus

↳ Závislosť

- orientovaný graf závislostí G je dvojica G = (V,E), kde
  - V je množina predikátov (uzly)
  - E: (p,q) ∈ E ak existuje pravidlo r také, že q je hlava pravidla a p sú podciele
- takýto graf zostavíme ľahko
- graf G určuje:
  ↳ či je program rekurzívny (obsahuje aspoň 1 cyklus)
  ↳ ktoré predikáty a ako sú rekurzívne
  ↳ EUP sú nerekurzívne

Príklad

- doplníme o ďalšie funkcie (či čo sú to) ← sú to (bezpečné) predikáty

$\boxed{\text{b\_s = bratranec\_sesternica}}$

b_s(X,Y) :- rodič(Xp,X), rodič(Yp,Y), súrodenci(Xp,Yp).

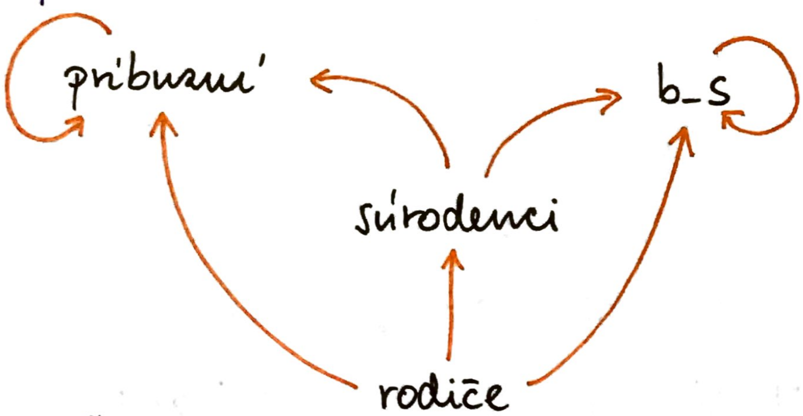
b_s(X,Y) :- rodič(Xp,X), rodič(Yp,Y), b_s(Xp,Yp)

príbuzní(X,Y) :- súrodenci(X,Y).

príbuzní(X,Y) :- príbuzní(X,Z), rodič(Z,Y).

príbuzní(X,Y) :- príbuzní(Z,Y), rodič(Z,X).

Graf závislostí:



- rodič je nutný výpočet príbuzných, súrodencov a aj pre bratrancov a sesterníc
  ↳ smyčka = rekurzia

napísať

Stačia bezpečné predikáty?

- odstraňujú problém voľných premenných
  ↳ p(X,Y) :- p(Y).
  ↳ neviem, čo dať za X aby to dávalo zmysel

- neriešia problém so vstavanými predikátmi, kt. môžu byť zdrojom nekonečnosti:

  väčší_než(X,Y) :- X >= Y.

Obmedzované premenné

- každá premenná, ktorá je na pravej strane pravidla
- každá premenná, ktorá je porovnávaná na = s konštantou
  ↳ X = 5
- každá premenná, kt. je porovnávaná na = s obmedzovanou premennou

Redefinícia bezpečných pravidiel

- pravidlo je bezpečné, ak sú všetky jeho premenné obmedzované

Pr

p(X,Y) :- q(X,Z), W=a, Y=W ← je bezpečné

- počíta $\pi(q) \times \{a\}$ ← projekcia(q) na prvý stĺpec
- pravidlo je konečné, ak Q je konečné (ale Q je wat)

Vyhodnotenie nerekurzívnych pravidiel (v grafe)

- nemusí obsahovať negáciu
- programy sú preložené do relačnej algebry
- OUP sú modelom pre pravidlá
- pre nerekurzívny program je možné usporiadať ~~p~~ pravidlá tak, že ak v $p_1, \dots, p_n$ platí, že $p_i < p_j$ potom vedie cesta z i do j

Vyhodnotenie OUP

- pre každé pravidlo $\underline{r}$ s hlavičkou $\underline{p_i}$ je vypočítaná relácia tela predikátu. Ide o ~~f~~ operáciu spojenia výsledkov všetkých podcieľov

telo

$\underbrace{\text{súrodenci}(X,Y)}_{\text{hlavička } p_i} \text{ :- } \overline{\underline{\text{rodič}(Z,X)}, \underline{\text{rodič}(Z,Y)}, \underline{X\ != Y}}.$

podciele

Výpočet samotného p_i je projekciou relácie tela predikátu na premenné korešpondujúce s hlavičkou s tým, že dochádza k zjednoteniu výsledkov pre všetky kombinácie.

Pozn.:

Podciele máme buď z predchádzajúcich krokov alebo to máme v tabuľke explicitných faktov.

Relácia definovaná telom pravidla

↳ relácia $\underline{r}$ pre $q \text{ :- } p_1, \dots, p_n$

↳ $P_1, \dots, P_n$ sú relácie pre $p_1, \dots, p_n$

$\longrightarrow P_j = \{(a_1, \dots, a_n) \mid p_j(a_1, \dots, a_n) \text{ je splnené}\}$

Podcieľ S je splnený ak:

- ak je to bežný podcieľ (EUP či OUP), potom S je v tvare: $p(b_1, \dots, b_n)$, kde $(b_1, \dots, b_n)$ je n-tica v relácii P korešpond. k p

- ak je to uslanauj podciel potom S je aritmeticka' relácia, ktora' je splnena'

Príklady

1.) b_s(X,Y) :- rodič(Xp,X), rodič(Yp,Y), súrodenci(Xp,Yp).

- rodič a súrodenci sú už vypočítané
- $R(X, X_p, Y, Y_p) = P(X_p, X) \circledast P(Y_p, Y) \circledast S(X_p, Y_p)$ $\circledast$ = spojenie resp. join

nech n-tica R je tvaru (a,b,c,d), potom:

- (a,b) je v P
- (c,d) je v P
- (a,c) je v S

- výsledná relácia má tiež 4 atribúty.
- pre každý riadok výsl. n-tice, kt. vznikla vyhodnotením čiastoč. výsl.

2.) Sourozenci(X,Y) :- rodič(Z,X), rodič(Z,Y), X != Y.

→ $Q(X,Y,Z) = \sigma_{X != Y}(P(Z,X) \circledast P(Z,Y))$

select where X != Y

→ Q obsahuje n-tice (x,y,z), kde

- (z,x) je v P
- (z,y) je v P
- x != y

→ equijoin cez Z → vznikne 3-atribútová relácia → 1 rodič, 2 potomkovia

→ navyše ešte selekcia, že x != y (equijoin nevie rozlíšiť rovnosť potomkov

→ ak chceme len x,y, musíme ešte spraviť projekciu na x,y

3.) $p(X,Y) :- q(a,X), r(X,Z,X), s(Y,Z)$

konštanta

- relácia pre $q(a,X)$ je $T(X) = \Pi_2(\sigma_{\$1=a}(Q))$

$q(a,X)$ — projekcia len na X (selekcia kde $\$1=a$)

$\underset{1}{a}$, $\underset{2}{X}$

- relácia pre $r(\underset{1}{X},\underset{2}{Z},\underset{3}{X})$ je $U(X,Z) = \Pi_{1,2}(\sigma_{\$1=\$3}(R))$

- relácia pre $s(Y,Z)$ je $S(Y,Z)$

- dostaneme $T(X) \circledast U(X,Z) \circledast S(Y,Z)$, ktorá obsahuje n-tice (x,y,z) také, že:
  - (a,x) je v Q
  - (x,z,x) je v R
  - (y,z) je v S

Algoritmus

VSTUP: telo pravidla r s podcieľmi $S_1,\dots,S_n$, ktoré generujú premenné $X_1,\dots,X_m$. Každé $S_i = p_i(A_{i_1},\dots,A_{i_n})$ má už spočítanú svoju reláciu R_i a A sú jej parametre (prem., konšt.)

VÝSTUP: výraz relačnej algebry, ktorý vypočíta reláciu $R(X_1,\dots,X_m)$ s n-ticami $(a_1,\dots,a_m)$, kde pri substitúcii a_j za X_j budú všetky podciele $S_1,\dots,S_n$ splnené.

Metóda

- Pre každé bežné S_i (podcieľ) je Q_i v tvare $\Pi_{V_i}(\sigma_{F_i}(R))$, kde
  - ↳ V_i je množina premenných z S_i
  - ↳ F_i je konjunkcia ($\wedge$)
    - ↳ ak je na pozícii k v podcieli S_i konštanta a, potom F_i obsahuje ($\$k=a$)
    - ↳ ak sú na pozíciách k a l v podcieli S_i stejné premenné, potom F_i obsahuje ($\$k=\$l$)
- Pre každú premennú X, kt. sa nenachádza v bežných podcieľoch S_i, urči doménu D_X, ktorá definuje unárnu reláciu obsahujúce všetky hodnoty, kt. môže X nadobúdať.
  - ak je r bezpečná, potom musí existovať také Y, že $X=Y$ je aspoň 1 z podcieľov a zároveň Y je buď konštanta alebo parameter bežného podcieľa
    - ↳ $Y=a$, potom $D_X=\{a\}$
    - ↳ ak je Y j-tý argument podcieľa S_i, potom $D_X=\Pi_j(R_i)$
- Nech E je normálne spojenie všetkých Q_i z prvého bodu a všetkých D_X z druhého bodu.
  - existencia príslušných atribútov je daná príslušnými premennými z podcieľov
- Vyhodnotenie relácie r je potom $\sigma_F(E)$, kde
  - ↳ F je konjunkcia aritmetických podcieľov pre každý podcieľ v r
  - ↳ E je výraz z 3. bodu. ak nie sú ostatné predikáty, potom zostáva iba E.

Príklad

$$p(X,Y) :- q(a,X), r(X,Z,X), s(Y,Z).$$

↳ podciele:

- S_1: $q(a,X)$
  ↳ $Q_1 = \pi_2(\sigma_{\$1=a}(Q))$
- S_2: $r(X,Z,X)$
  ↳ $Q_2 = \pi_{1,2}(\sigma_{\$1=\$3}(Q))$
- S_3: $s(Y,Z)$
  ↳ $Q_3 = S(Y,Z)$

→ spojenie nad Q_1, Q_2, Q_3 + projekcia na X, Y

Veta: Výsledný algoritmus je korektný v tom, že výsledná relácia R obsahuje iba také n-tice, ktoré substitúcie za premenné v podcieli vedú vždy k splneniu týchto podcieľov

→ ∃ dôkaz

Pravidlá s viacerými pravými stranami

- vezmi do úvahy všetky predikáty s p v hlavičke
- vypočítaj ich telá
- vykonaj projekciu na premenné z hlavičky
- zjednoť výsledky

↓

Problém

- konštanty v hlavičke – $q(a,z)$
- opakované premenné – $q(X,Z,A,X)$

Príklad

- upravíme na iné tvary:

konšt.

$$p(a,X,Y) \text{ :- } r(X,Y). \xrightarrow{\text{úprava}} p(U,V,W) \text{ :- } r(V,W),\ U=a.$$

$$p(X,Y,X) \text{ :- } r(Y,X). \qquad p(U,V,W) \text{ :- } r(V,U),\ \underline{W=U}.$$

rovnaké premenné

Lemma: • ak je $\underline{r}$ bezpečné, potom aj upravené $\underline{r}$ je bezpečné

• pravidlá $\underline{r}$ a upravené $\underline{r}$ sú splniteľné pre rovnaké hodnoty argumentov.

DDB s negáciou

Negácia je problém: máme viac minimálnych modelov a nájdeme len 1 z nich, ak ho nájdeme, výpočet je problematický

- nehľadáme všetky modely, len minimálny, z ktorého je možné odvodiť všetky ostatné.