Negácia v databázach a logickom programovaní

2. časť úvodnej prednášky

Negácia

- Ako zistiť, že niečo neplatí ako to dokázať?
- Ako zistiť, že teória je nekonzistentná?
- Bertrand Russell: Keď pripustíme jednu nepravdu dá sa dokázať všetko.
- Predikátový kalkul: Z množiny pozitívnych faktov a implikácii nevyplýva žiaden negatívny výrok.
- Databázisti a umelí inteligenti sú s matamatickou logikou nespokojní.

O logike

- Logika je len jedna, kto myslí inak, myslí nelogicky.
  – Aristoteles
- Matematická logika je od Boha a pre Boha.
  - Tretej cesty niet: $A \vee \neg A = true$, Boh je vševedúci.
- ***Neviem, možno, pravdepodobne*** to sú výmysly diabla.
- Svetonázorové otázky
  1. Čo existuje ?
  2. Čo platí (je pravda) ?
  3. Vzťah k protirečeniam: $A \wedge \neg A = false.$
- Rôzne oblasti ľudskej vedy sa stavajú k svetonázorovým otázkam rôzne.

Krátky prehľad

- Empirické vedy (fyzika, medicína)
  - Každá hypotéza je pravdivá, pokiaľ ju niekto nevyvrátí (neobjaví, fakty ktoré jej protirečia).
- Právo
  - Všetko platí, pokiaľ zákon neurčí inak.
  - Zákony si môžu protirečiť, pravidlá na riešenie protirečení
    - Lex superior derogat legi inferiori.
    - Lex posterior derogat legi priori.
    - Lex specialis derogat legi generali.
  - Ius summum saepe summa est iniuria. (Cicero)

Formálne systémy

- Propozicionálny kalkul
  - Dvojhodnotový
  - Trojhodnotový a viachodnotové
- Predikátový kalkul
  - prvého rádu (používame variantu s rovnosťou)
  - prvý a druhý rád nepodstatné rozšírenia I. rádu
  - vyšších rádov
- Lambda kalkul
- Všeliké „zvrhlé“ logiky (vyskytujú sa v UI).

Teórie a modely

- Teória = množina formúl (tvrdení).
- Modelom teórie nazývame akýkoľvek objekt, pre ktorý platia aspoň všetky tvrdenia teórie.
- Matematický model obsahuje doménu (obor, z ktorého premenné nadobúdajú hodnoty), relácie, ktoré sú priradené predikátovým symbolom a funkcie priradené funkčným symbolom
- Matematické modely teórie bez predikátových symbolov sú algebry
- Ak teória obsahuje aj predikátové symboly hovoríme o algebraických štruktúrach, alebo jednoducho o štrukturách

Doménove nezávislé formuly

- Formula $\phi(\overline{X})$ je doménovo nezávislá, ak pre dve štruktúry $\mathfrak{I}1 = <D1, \mathcal{F}, \mathcal{R}>$ a $\mathfrak{I}2 = <D2, \mathcal{F}, \mathcal{R}>$, odlišujúce sa iba v doméne je množina $\{\bar{x}: \phi(\bar{x})\}$ rovnaká.
- Je nerozhodnuteľné, či formula ϕ je doménovo nezávislá.
- Ak formula ϕ jedoménovo nezávislá, potom bezpečná formula $\Delta \wedge \phi$, je jej ekvivalentná.

Modely, algebry a databázy

- Vždy popísané formulami predikátového kalkulu
- Všeobecné modely (záujem matematikov)
  - veľké modely (ultra produkt)
  - spočitateľné modely (zostrojené z termov, Herbrandovské)
- Algebry sú o formuliach, ktoré sú splnené pre špecifické domény (grupa, pole, zväz, ...). Vlastne, axiomy algebry definujú tieto domény.
- Spočitateľná alebo konečná doména, s rovnosťou a prípadne usporiadaním. Doménovo nezávislé formuly vypovedajú o reláciach.

Ako je to v databázach

- V databázach:
  - teória = EDB + program
  - model = EDB + IDB
- Ak program neobsahuje negácie, počítame najmenší model naívnou (seminaívnou) iteráciou. Tarského veta zaručuje, že výpočet úspešne skončí.
- Sémantika najmenšieho modelu súhlasí s matematickou sémantikou.
  - V najmenšom modeli datalógového programu bez negácie platí práve to, čo vo všetkých modeloch tohto programu.

Tarského veta o pevnom bode

Úplný zväz: $S = < D, \sqsubseteq, \sqcup, \sqcap, \bot, \top >$

$\sqsubseteq$ čiastočné usporiadanie

$\sqcup$ l.u.b., sup, join $\quad\quad$ $\bot$ dolník

$\sqcap$ g.l.b., inf, meet $\quad\quad$ $\top$ horník

Každá neprázdna množina má l.u.b. (suprémum).

Veta: Nech ***F*** je zobrazenie z úplného zväzu *S* do *S* také, že $x \sqsubseteq y \Rightarrow F(x) \sqsubseteq F(y)$. Potom ***F*** má aspoň jeden pevný bod.

Dôkaz: Nech $U = \{x: x \sqsubseteq F(x)\}$. Množina U je neprázdna, $\bot \in U$.

Označme: $x_0 = \bigsqcup_{x \in U} x$. Pre každé $x \in U$ platí $x \sqsubseteq F(x) \sqsubseteq x_0$.

Preto aj $x_0 \sqsubseteq F(x_0)$ t.j. $x_0 \in U$. Preto aj $F(x0) \sqsubseteq F(F(x_0))$ a $F(x_0) \in U$. Z toho ale plynie $F(x_0) \sqsubseteq x_0$. Teda $x_0 = F(x_0)$.

Výpočet rekurzívnych programov bez negácie

Systém rovníc: $\vec{P} = \vec{\mathrm{E}}(\vec{P}, \vec{R})$, kde $\vec{P}$ sú intencionálne a $\vec{R}$ extenzionálne predikáty.

Riešenie: naivnou iteráciou. Začnememe $\vec{P_0} = \vec{0}$. Postupne počítame postupne P_1, P_2, … , podľa vzorca:

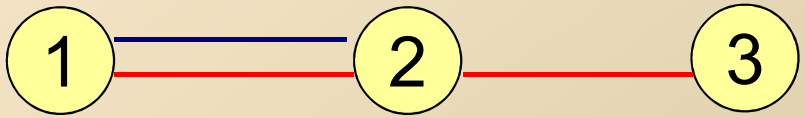
$$\vec{P}_i = \vec{\mathrm{E}}\,(\vec{P}_{i-1}, \vec{R}\,),$$

pokiaľ $\vec{P_i} \neq \vec{P_{i+1}}$.

Podľa Tarského vety o pevnom bode tento proces vždy skončí (konverguje). Stačí overiť, že prirodzené spojenie, zjednotenie, projekcia a premenovanie sú „neklesajúce“ operácie. Vypočítané riešenie je najmenší pevný bod uvedeného systému rovníc. Z vlastnosti implikácií plynie, že každé riešenie uvedeného systému rovníc musí obsahovať vety vypočítané našim algoritmom.

Datalógové programy s negáciou

- Program s negáciou nemusí mať najmenší pevný bod.
  p ← q.
  p ← p,¬q.
- Eventuálne môže mať niekoľko minimálnych pevných bodov.
  modrácesta(x, y) ← modrá(x, y)
  modrácesta(x, y) ← modrá(x, z), modrácesta(z, y)
  červenýmonopol(x,y) ← červená(x, y),¬modrácesta(x, y).
- EDB



- Minimálne pevné body:

  EDB + modrácesta(1,2) + modrácesta(2,3)

  EDB + modrácesta(1,2)+ červenýmonopol(2, 3)
  stratifikovaný pevný bod

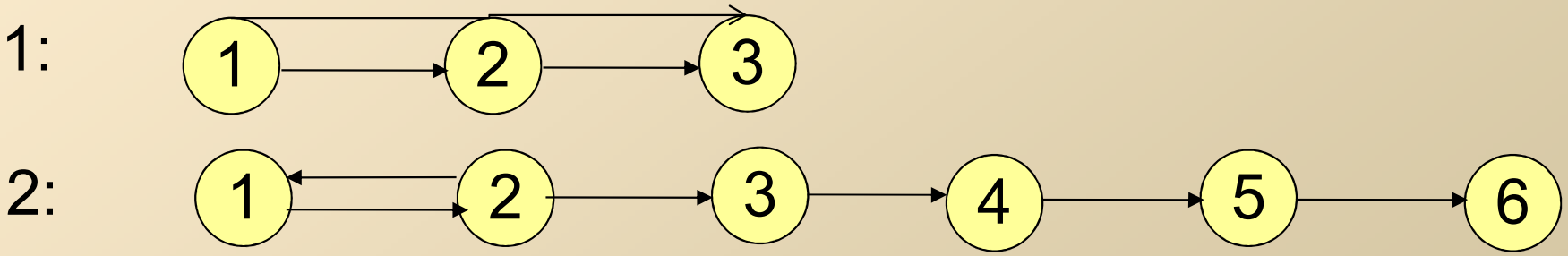
Zovšeobecnenia

Niekedy vieme vypočítať aj nestratifikované programy:

výhra(x) ← ťah(x, y), ¬výhra(y)

To môže byť pavidlo pre nejakú hru (odoberanie zápaliek, nim), kde prehráva hráč, ktorý nemá ťah.

Príklady rôzny naplnení tabuľky ťah:



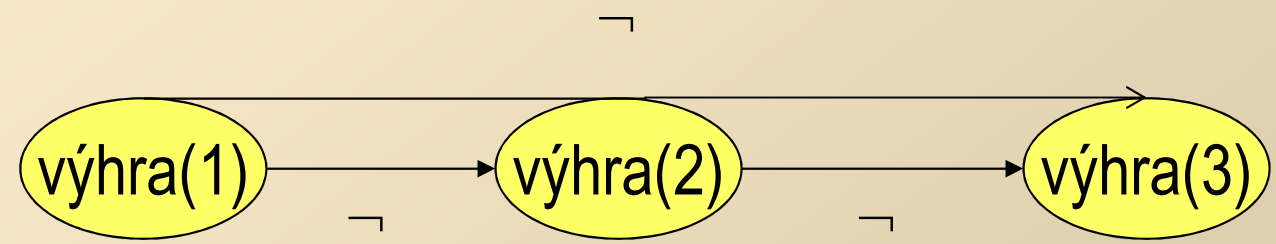
Oba prípady vieme vyriešiť. V každom prípade musíme pracovať s obsahom tabuliek. Odvoditeľnými uzavretými atómami (ground atoms).

Lokálne stratifikované modely

1. Vytvor graf závislostí IDB atómov:
   - Uzly sú relevatné IDB atómy.
   - Hrana $p \rightarrow q$ práve vtedy, keď q sa vyskytuje instanciovanom tele pravidla s hlavou p.
   - Značka (label) na hrane, ak q je negované.
2. Stratifikácia sa robí obvyklým spôsobom.
3. Model je lokálne stratifikovaný, ak počet úrovní (strata) je konečný.

Príklad

výhra(1) ← ťah(1,2), ¬výhra(2)
výhra(1) ← ťah(1,3), ¬výhra(3)
výhra(2) ← ťah(2,3), ¬výhra(3)



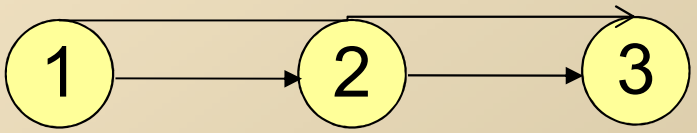
Stratum 2 Stratum 1 Stratum 0

Stabílne modely (stable models)

- Hrubá intuícia: model je stabílny, keď aplikáciou programu sa nepodarí odvodiť nič nové, smie sa však využívať len negatívna informácia.
- Formálne: model M je stabílny, ak M = GLT(M), kde GLT je Gelfond Lifschitzova transformácia.

Príklad: *výhra(x) ← ťah(x, y), ¬výhra(y)*

s EDB ťah:



M = EDB ∪ {výhra(1), výhra(2)}

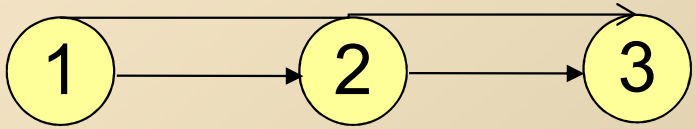
je stabílny model. Žiadnou substitúciou za x, y neodvodíme *výhra(3)*.

Gelfond Lifschitzova transformácia (GLT)

1. Inštancijuj pravidlá všetkymi možnými spôsobmi.
2. Vynechaj inštanciované pravidlá s nepravdivým EDB alebo zabudovaným podcieľom (včítane negácie, ¬).
3. Vynechaj inštanciované pravidlá s IDB podcieľom ¬p(…), kde p(…) je v *M*.
4. Vynechaj podcieľ ¬p(…), ak p(…) nie je v *M*.
5. Vynechaj pravdivé EDB a zabudované podciele.
6. Použi zvyšné inštanciované pravidlá a EDB na odvodenie všetkých IDB uzavretých atómov.
7. Na záver pridaj EDB.

Príklad – znovu ten istý

výhra(x) ← ťah(x, y), ¬výhra(y)
s EDB:
M = EDB ∪ {výhra(1), výhra(2)}



1. krok (všetky inštanciované pravidlá)
2. krok (nemá efekt na dané pravidlá)

~~*výhra(1) ← ťah(1, 2), ¬výhra(2)*~~ 3. krok *výhra(2)* $\in$ *M*

výhra(1) ← ~~ťah(1, 3)~~, ~~¬výhra(3)~~ 4. krok výhra(3) $\notin$ M

výhra(2) ← ~~ťah(2, 3)~~, ~~¬výhra(3)~~ 5. krok pravdivý EDB predikát

6. krok zostali nám dve pravidlá s prázdnym (true) telom. Teda odvodime *{výhra(1), výhra(2)}.*
7. krok pridáme EDB. *GLT(M) = EDB∪ {výhra(1), výhra(2)}*

Trojhodnotové modely

- Trojhodnotový model pozostáva z:
  1. Z množiny pravdivých EDB faktov.
  2. Všetky ostatné EDB fakty sú považované za nepravdivé.
  3. Z množiny pravdivých IDB faktov.
  4. Z množiny nepravdivých IDB faktov.
  5. Pravdivostná hodnota zvyšných IDB faktov je neznáma (unknown).

Well-founded modely

- Začni s inštanciovanými pravidlami.
- Vyčisti „clean“ pravidlá t.j. vynechaj pravidlá s známými ne-pravdivými podcieľmi a vynechaj známe pravdivé podciele.
- Dva spôsoby odvodenia:
  1. „Obyčajný“: z pravdivého tela vyplýva hlava.
  2. „Unfounded sets“: predpokladá, že každý prvok unfounded set je nepravdivý.
- *U* je „unfounded set“ (pozitívnych uzavretých IDB atomov), ak každé zostávajúce inštanciované pravidlo s hlavou v *U,* má aj v tele prvok z *U*.
- Vlastnosť byť „unfounded set“ je uzavretá vzhľadom na zjednotenie. Teda existuje najväčšia unfounded set.
- Nikdy nedokážeme odvodiť pravdivosť prvku U, ale môžeme predpokladať jeho nepravdivosť.

Presnejšie „Unfounded set w.r.t. partial model“

Def: Nech M je čiastočný model pre program P. Množinu U inštaciovaných (uzavretých) atómov nazývame unfounded set pre P vzhľadom k M. Ak pre každé $A \in U$ a pre každé inštanciované pravidlo $A \leftarrow$ body z P platí:

i.) body obsahuje podcieľ nepravdivý v M alebo

ii.) nejaký pozitívny podcieľ z body je v U.

Lema: Ak U1 a U2 sú unfounded, potom aj $U1 \cup U2$ je unfounded.

Dôsledok: Existuje najväčšia unfounded set.

Konštrukcia well-founded modelu

Podciele z unfounded set sa nedajú dokázať (ani vyvrátiť). Podľa princípu byrokracie chceme, aby ich negácia bola pravdivá.

repeat

“clean” instantiated rules;

make all ordinary inferences;

“clean” instantiated rules;

find the largest unfounded set and make its atoms false;

until no changes;

make all remaining IDB atoms “unknown”;

Alternujúci pevný bod
Alternating fixedpoint

1. Inštancijuj a „vyčisti“ pravidlá.
2. Na počiatku sú všetky IDB uzavreté atomy nepravdivé.
3. V každom ďalšom kole aplikuj GLT na EDB a na v predošlom odvodené pravdivé IDB uzavreté atomy.
4. Proces konverguje k alternácii dvoch množín pravdivých IDB faktov.
5. Nepárne kolá len zväčšujú; párne kolá len zmenšujú množinu pravdivých faktov.
6. V limite: pravdivé fakty sú stále pravdivé; nepravdivé fakty sú stalé nepravdivé a fakty s pravdivostnou hodnotou „unknown“ alternujú.

Príklad – priama konštrukcia

výhra(x) ← ťah(x, y), ¬výhra(y)

S EDB:

Inštanciované vyčistené pravidlá:

výhra(1) ← ¬výhra(2)
výhra(2) ← ¬výhra(1)
výhra(2) ← ¬výhra(3)
výhra(3) ← ~~¬výhra(4)~~
~~výhra(4) ← ¬výhra(5)~~
výhra(5) ← ~~¬výhra(6)~~

Kolo	G.U.S	pozitívne	negatívne
1.	{výhra(6)}		¬výhra(6)
2.	{výhra(4)}	výhra(5)	¬výhra(4)
3.	Ø	výhra(3)	

Príklad – alternujúci pevný bod

výhra(x) ← ťah(x, y), ¬výhra(y)

S EDB:

Inštanciované vyčistené pravidlá:

výhra(1) ← ¬výhra(2)
výhra(2) ← ¬výhra(1)
výhra(2) ← ¬výhra(3)
výhra(3) ← ¬výhra(4)
výhra(4) ← ¬výhra(5)
výhra(5) ← ¬výhra(6)

kolo	0	1	2	3	4	5
výhra(1)	F	T	F	T	F	T
výhra(2)	F	T	F	T	F	T
výhra(3)	F	T	F	T	T	T
výhra(4)	F	T	F	F	F	F
výhra(5)	F	T	T	T	T	T
výhra(6)	F	F	F	F	F	F

Príklad – alternujúci pevný bod

výhra(x) ← ťah(x, y), ¬výhra(y)

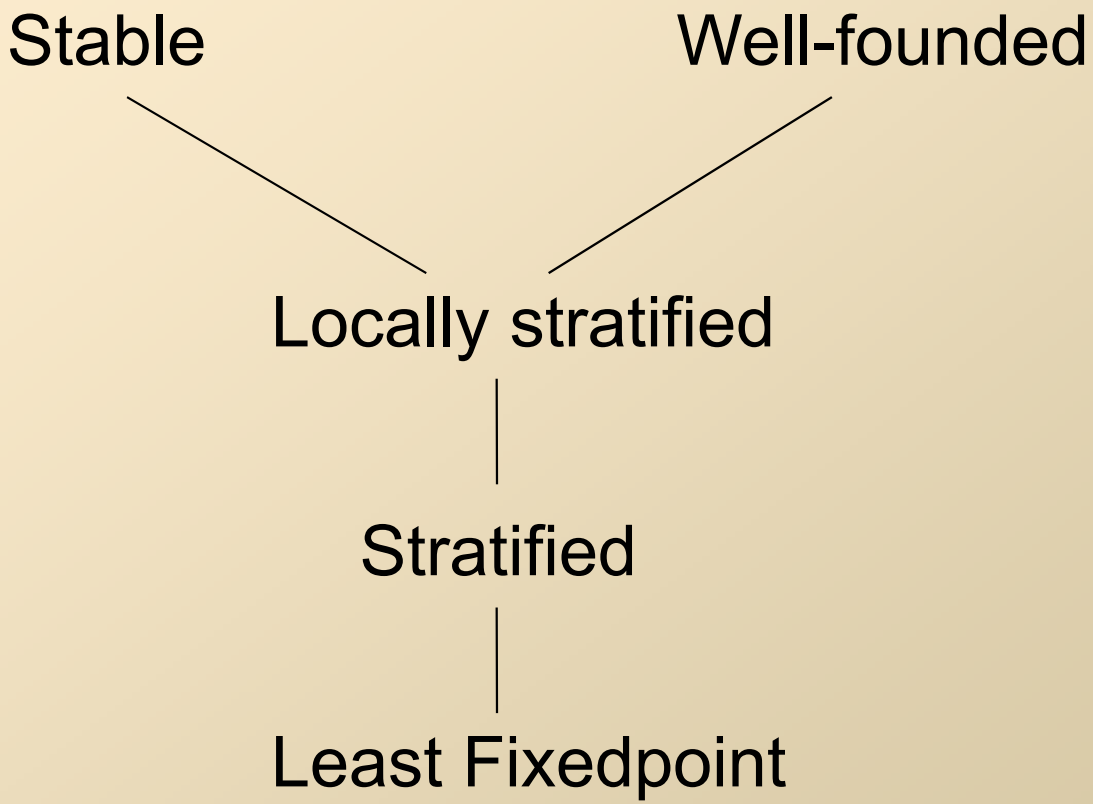
S EDB:

Inštanciované vyčistené pravidlá:

výhra(1) ← ¬výhra(2)
výhra(2) ← ¬výhra(1)
výhra(2) ← ¬výhra(3)
výhra(3) ← ¬výhra(4)
výhra(4) ← ¬výhra(5)
výhra(5) ← ¬výhra(6)

kolo	0	1	2	3	4	5
výhra(1)	F	T	F	T	F	T
výhra(2)	F	T	F	T	F	T
výhra(3)	F	T	F	T	T	T
výhra(4)	F	T	F	F	F	F
výhra(5)	F	T	T	T	T	T
výhra(6)	F	F	F	F	F	F

Porovnanie sémantík



Inflačná sémantika

1. Na začiatku predpoladáme, že všetky uzavreté IDB atómy sú nepravdivé.
2. Opakujeme následujúci proces:
   - Inštacijuj pravidlá použitím EDB a doteraz pravdivých IDB predikátov.
   - Ak je telo pravdivé (pritom predpokladáme, že všetky predikáty, ktoré nie sú pravdivé sú nepravdivé) je aj hlava pravdivá.
   - Pokiaľ sa niečo pridalo.

- Raz odvodený IDB fakt zostáva stále pravdivý,

Jednoduchý príklad

1. $p \leftarrow q.$
2. $q \leftarrow \neg r.$
3. $s \leftarrow \neg p.$

1. kolo: Všetky predikáty sú nepravdivé.
2. kolo: V dôsledku (2) q je pravdivé a v dôsledku (3) aj s je pravdivé.
3. kolo: Odvodíme p podľa (1).
4. kolo: Nič nové sa neodvodí. Koniec.

Inflačný model: $M = \{p, q, s\}$.

Negácia ako nekonzistencia
Negation as inconsistency

Daný je datalógový program P a jeho čiastočný model M (oboje chápeme ako množinu formúl). Formula $\neg\varphi$ platí, ak φ nie je konzistentné s $P \cup M$. (T.j. φ neplatí v žiadnom rozšírení $P \cup M$.)

Vyžaduje evidovať pozitívne aj negatívne atómy.

- Matematicky je to korektné v praxi slabé. (Chceme, aby platilo viac negatívnych faktov, ako sa dá dokázať.)
- D.M. Gabbay, H. J. Sergot: Journal of logic programming. Vol. 1, pp. 1-35 (1986).
- Podobá sa to na intuicionisticku logiku, alebo finite forcing (A. Robinson cca 1970).

DLV (TUV Vienna + University of Calabria)

- Dva druhy negácie

~ normálna databázova negácia

$\neg p$ intenzionálny predikát pre negáciu p

p^- EDB relácia pre negáciu p danú explicité.

Pravidlá: $\neg p \leftarrow p^-$.

$\neg p \leftarrow \sim p$. (princíp byrokracie)

Nemusíme posledné pravidlo použiť. Môžeme pridať aj iné pravidlá.

Požaduje sa aby platilo: $(\neg\exists \mathbf{x})(p(\mathbf{x}) \wedge \neg p(\mathbf{x}))$.

DLV (http://www.dlvsystem.com/)

- Disjunktívny datalóg v hlave pravidla môžu byť aj disjunkcie.
  - Napr. $p \vee \neg p \leftarrow$. Princíp vylúčenia tretieho.
- DLV počíta koherentné stabílne modely.
  - Model je koherentný, ak pre relácie zodpovedajúce všetkým predikátom platí: $p \cap p^- = \emptyset$.
- *Brave reasoning* - platí v aspoň jednom modeli.
- *Cautios reasoning* - platí vo všetkých modeloch.

O čom to je ? – Riešenie hry nim.

stav(x) ← ťah(x,y)
← ťah(z,x)
prehra(x) ← stav(x),¬ťah(x, _) /* mat */
← stav(x),¬dobrýťah(x, _) /*nemá neprehrávajúci ťah*/
dobrýťah(x,y) ← ťah(x,y), ¬výhra(y)
výhra(x) ← ťah(x, y), prehra(y)
remíza(x) ← stav(x),¬výhra(x),¬prehra(x)

Predikáty výhra, prehra a dobrýťah sú zacyklené.
Potrebujeme aspoň lokálne stratifikovaný pevný bod.
Aby sme sa zaobišli so straifikovaným pevným bodom, museli by sme využiť nejaké špecifické znalosti o hre.

Trade off: Zložitosť sémantiky (well-founded) za pojmovú zložitosť programu.

Semijoin a antijoin

Definicie:
Semijoin $R \ltimes S = \Pi_R(R \bowtie S)$
Antijoin $R \triangleleft\triangleright S = R - (R \ltimes S) = R - \Pi_R(R \bowtie S)$

Obe operácie sú odvodené. Dajú sa využiť na optimalizáciu a pri výpočte datalógových programov s negáciou.

Antijoin je zovšeobecnenie rozdielu aj na relácie s rôznými schémami.

Zápis v datalógu:
semijoin(X,Y,Z) ← r(X,Y,Z), s(Y,_).
antijoin(X,Y,Z) ← r(X,Y,Z), ¬ s(Y,_).

Vstupné množiny a pseudokľúče

Hoci domény môžu byť nekonečné spočitateľné množiny.
Databázové relácie sú vždy konečné - tabuľky s konečným počtom riadkov.
Zaujíma nás, či dokážeme tabuľku vypísať.
Príklady nekonečných tabuliek (matematických predikátov):

- =(x,y), <(x,y), y = sin(x)
- ≠(x,y), y = arcsin(x)
- perverse(a,b,c,n) $\Leftrightarrow$ $a^n + b^n = c^n$ (a, b, c, n prirodzené čísla).

Definícia: Vstupnou množinou pre reláciu R, nazývame takú, množinu atribútov relácie R, že existuje len konečný počet riadkov R z danými hodnotami vo vstupnej množine.
Minimálna vstupná množina relácie R sa nazýva *pseudokľúč* relácie R.

Krajné prípady

- Pseudokľúč je prázdna množina (generátor)
  - databázové tabuľky
  - konečné matematické relácie
- Podmienka použiteľnosti je, že db-systém má k dispozícii program, ktorý postupne generuje (číta) prvky relácie.
- Pseudokľúč tvoria iba všetky atribúty relácie (rozpoznávač). Znovu db-systém musí disponovať programom, ktorý zistí, či daná n-tica patrí do relácie.

Dôsledok: Použiteľné sú len rekurzívne relácie.

Bezpečnosť programov s matematickými predikátmi

- Join a antijoin databazových relácii sú bezpečné operácie. Výsledok je bezpečná relácia.
- Join a antijoin bezpečnej relácie s nekonečnou reláciou je bezpečný pokiaľ sa „joinuje“ podľa vstupnej množiny nekonečnej relácie. Výsledok je bezpečná relácia.
- Projekcia, selekcia a agregácia aplikovaná na bezpečnú reláciu je bezpečná. Výsledok je bezpečná relácia.

Výpočet datalógových programov s negáciou (stratifikovaný pevný bod)

- Všetko robíme rovnako ako bez negácie jedine v kroku 2 pre výpočet pravidla v prípade nenegogovaného predikátu generujeme join a v prípade negovaného predikátu antijoin.
- Anonýmne premenné (podčiarovníky) postupne nahradíme čerstvými nikde sa nevyskytujúcimi premennými. Striktne vzaté nemali by sa vyskytovať v negovaných predikátoch (pravidlo nie je bezpečné), ale nevadia.
- Poradie negovaných a nenegovaných predikátov je dôležité antijoin sa nesmie použiť, pokiaľ v pozitívnej časti nie sú inštanciované spoločné premenné.

Výpočet zhora dole – SLD rezolúcia

Jedná sa o teoretický model (pure prolog).
Je to nedeterministický výpočet.
Dokazujeme konjunkciu cieľov G1, …, Gn

1. Nedeterministicky vyber podcieľ (intenzionálny predikát napr. Gi).
2. Nedeterministicky pravidlo s hlavou vybraného podcieľu Gi
3. Unifikuj hlavu pravidla s vybraným podcieľom, pričom sa uprednostňujú premenné podcieľa.
4. Nahraď prislušný podcieľ telom pravidla.
5. Na novú konjunkciou aplikuj vypočítanú substitúciu (mgu).
6. Ak sú už všetky podciele extezionálne, over v databáze, či ich konjunkcia (kombinácia joinov a antijoinov) je neprázdna.

Skutočný prológ

- Relatívne nekorektná implementácia predošlej metódy.
- Nedeterminizmus je nahradený prehladávaním do hĺbky (backtracking)
- Z unifikácie je vynechaná kontrola na výskyt
- Databáza je obmedzená na množinu extenziónálnych atómov zapamätateľnú v operačnej pamäti.
- Obsahuje rozšírenia skôr smerom k programovaniu než smerom k bázam dát.
- Vyššie uvedená metóda sa dá implmentovať aj korektne, ale prológ si zo svojch chýb už urobil features.

Porovnanie metód zdola nahor (forward chaining) a zhora nadol (backward chaining)

Uvažujme program pre výpočet ciest v grafe.
$p(x, y) \leftarrow e(x, y).$

$p(x, y) \leftarrow e(x, z), p(z, y).$

Pokiaľ je našim cieľom vypočítať celý tranzitívny uzáver grafu, sa to zdá byť skoro jedno. Matching je jednoduchší ako unifikácia. Asi by som uprednostnil výpočet zdola nahor.

Ak však chceme zistiť, či existuje cesta p(a, b), je výpočet zhora nadol evidentne výhodnejší a to aj v prípadoch, keď prehľadáme všetky cesty vychádzajúce z a (vedúce do b).

Vďaka nekorektnej implementácii, prológ sa môže zacykliť a nenájsť riešenie dokonca z dvoch dôvodov:
Ľavá rekurzia (dá sa odstrániť), cyklus v dátach (v grafe).