# Negace v logickém programování

## Negativní znalost

- logické programy vyjadřují **pozitivní znalost**
- **negativní literály:** pozice určena definicí Hornových klauzulí
  ⇒ nelze vyvodit **negativní** informaci z logického programu
  - každý predikát definuje úplnou relaci
  - negativní literál **není** logickým důsledkem programu
- relace vyjádřeny explicitně v nejmenším Herbrandově modelu
  - $nad(X,Y) :- na(X,Y).$ $\quad na(c,b).$
    $nad(X,Y) :- na(X,Z), nad(Z,Y).$ $\quad na(b,a).$
  - nejmenší Herbrandův model: $\{na(b,a), na(c,b), nad(b,a), nad(c,b), nad(c,a)\}$
- ani program ani model nezahrnují negativní informaci
  - $a$ není nad $c$, $a$ není na $c$
  - i v realitě je negativní informace vyjadřena explicitně zřídka, např. jízdní řád



## Předpoklad uzavřeného světa

- neexistence informace chápána jako opak:
  **předpoklad uzavřeného světa (*closed world assumption*, CWA)**
- převzato z databází
- určitý vztah platí **pouze** když je vyvoditelný z programu.
- „odvozovací pravidlo" ($A$ je (uzavřený) term): $\dfrac{\mathrm{P} \not\models A}{\neg A}$ (CWA)
- pro SLD-rezoluci: $P \not\models nad(a,c)$, tedy lze podle CWA odvodit $\neg nad(a,c)$
- problém: není rozhodnutelné, zda daná atomická formule je logickým důsledkem daného logického programu.
  - nelze tedy určit, zda pravidlo CWA je aplikovatelné nebo ne
- CWA v logickém programování obecně nepoužitelná.



## Negace jako neúspěch (*negation as failure*)

- slabší verze CWA: **definitivně neúspěšný (*finitely failed*) SLD-strom** cíle : $-A$

  $\dfrac{\text{: } -A \text{ má definitivně (konečně) neúspěšný SLD-strom}}{\neg A}$ **(*negation as failure*, NF)**
- **normální cíl**: cíl obsahující i negativní literály
  - : $-nad(c,a), \neg nad(b,c).$
- rozdíl mezi CWA a NF
  - program $nad(X,Y) :- nad(X,Y)$, cíl : $-\neg nad(b,c)$
  - neexistuje odvození cíle podle NF, protože SLD-strom : $-nad(b,c)$ je nekonečný
  - existuje odvození cíle podle CWA, protože neexistuje vyvrácení : $-nad(b,c)$
- CWA i NF jsou nekorektní: $A$ není logickým důsledkem programu $P$
- řešení: definovat programy tak, aby jejich důsledkem byly i negativní literály
  **zúplnění logického programu**

# Podstata zúplnění logického programu

- převod všech **if** příkazů v logickém programu na **iff**
  - $nad(X,Y) :- na(X,Y).$
    $nad(X,Y) :- na(X,Z), nad(Z,Y).$
  - lze psát jako: $nad(X,Y) :- (na(X,Y)) \vee (na(X,Z), nad(Z,Y)).$
  - zúplnění: $nad(X,Y) \leftrightarrow (na(X,Y)) \vee (na(X,Z), nad(Z,Y)).$
  - $X$ je nad $Y$ **právě tehdy, když alespoň jedna z podmínek platí**
  - tedy **pokud žádná z podmínek neplatí**, $X$ **není** nad $Y$
- kombinace klauzulí je možná pouze pokud mají identické hlavy
  - $na(c,b).$
    $na(b,a).$
  - lze psát jako: $na(X_1,X_2) :- X_1 = c, X_2 = b.$
    $na(X_1,X_2) :- X_1 = b, X_2 = a.$
  - zúplnění: $na(X_1,X_2) \leftrightarrow (X_1 = c, X_2 = b) \vee (X_1 = b, X_2 = a).$



# Zúplnění programu

- **Zúplnění programu** $P$ je: $\mathrm{comp}(P) := \mathrm{IFF}(P) \cup \mathrm{CET}$
- Základní vlastnosti:
  - $\mathrm{comp(P)} \models \mathrm{P}$
  - do programu je přidána pouze negativní informace
- $\mathbf{IFF(P)}$: spojka $:-$ v $\mathrm{IF(P)}$ je nahrazena spojkou $\leftrightarrow$
- $\mathbf{IF}(P)$: množina všech formulí $\mathbf{IF}(q,P)$
  pro všechny predikátové symboly $q$ v programu P
- Cíl: definovat $\mathbf{IF}(q,P)$
- $def(p/n)$ predikátu $p/n$ je množina všech klauzulí predikátu $p/n$



# $\mathbf{IF}(q,P)$

$na(X_1,X_2) :- \underline{\exists Y(X_1 = c, X_2 = b, f(Y))} \vee (X_1 = b, X_2 = a, g).$

$\underline{na(c,b) :- f(Y).}$ $na(b,a) :- g.$

- $q/n$ predikátový symbol programu $P$
- $X_1,\ldots,X_n$ jsou „nové" proměnné, které se nevyskytují nikde v $P$
- Nechť $C$ je klauzule ve tvaru
  $q(t_1,\ldots,t_n) :- L_1,\ldots,L_m$
  kde $m \geq 0$, $t_1,\ldots,t_n$ jsou termy a $L_1,\ldots,L_m$ jsou literály.

  Pak označme $\mathbf{E}(C)$ výraz $\exists Y_1,\ldots,Y_k(X_1 = t_1,\ldots,X_n = t_n, L_1,\ldots,L_m)$
  kde $Y_1,\ldots,Y_k$ jsou všechny proměnné v $C$.
- Nechť $def(q/n) = \{C_1,\ldots,C_j\}$.

  Pak formuli $\mathbf{IF}(q,P)$ získáme následujícím postupem:

  $q(X_1,\ldots,X_n) :- \mathbf{E}(C_1) \vee \mathbf{E}(C_2) \vee \cdots \vee \mathbf{E}(C_j)$ pro $j > 0$ a
  $q(X_1,\ldots,X_n) :- \square$ pro $j = 0$ [$q/n$ není v programu $P$]



# Clarkova Teorie Rovnosti (CET)

všechny formule jsou univerzálně kvantifikovány:

1. $X = X$
2. $X = Y \rightarrow Y = X$
3. $X = Y \wedge Y = Z \rightarrow X = Z$
4. pro každý $f/m$: $X_1 = Y_1 \wedge \cdots \wedge X_m = Y_m \rightarrow f(X_1,\ldots,X_m) = f(Y_1,\ldots,Y_m)$
5. pro každý $p/m$: $X_1 = Y_1 \wedge \cdots \wedge X_m = Y_m \rightarrow (p(X_1,\ldots,X_m) \rightarrow p(Y_1,\ldots,Y_m))$
6. pro všechny různé $f/m$ a $g/n$, $(m,n \geq 0)$: $f(X_1,\ldots,X_m) \neq g(Y_1,\ldots,Y_n)$
7. pro každý $f/m$: $f(X_1,\ldots,X_m) = f(Y_1,\ldots,Y_m) \rightarrow X_1 = Y_1 \wedge \cdots \wedge X_m = Y_m$
8. pro každý term $t$ obsahující $X$ jako vlastní podterm: $t \neq X$

$X \neq Y$ je zkrácený zápis $\neg(X = Y)$

# Korektnost a úplnost NF pravidla

- **Korektnost NF pravidla**: Nechť $P$ logický program a $:-A$ cíl.
  Jestliže : $-A$ má definitivně neúspěšný SLD-strom,
  pak $\forall(\neg A)$ je logickým důsledkem $\mathrm{comp}(P)$ (nebo-li $\mathrm{comp}(P) \models \forall(\neg A)$)
- **Úplnost NF pravidla**: Nechť $P$ je logický program. Jestliže $\mathrm{comp}(P) \models \forall(\neg A)$, pak existuje definitivně neúspěšný SLD-strom : $-A$.
  - zůstává problém: není rozhodnutelné, zda daná atomická formule je logickým důsledkem daného logického programu.
  - teorém mluví pouze o **existenci** definitivně neúspěšného SLD-stromu
  - definitivně (konečně) neúspěšný SLD-strom sice existuje, ale nemusíme ho nalézt
    - např. v Prologu: může existovat konečné odvození, ale program přesto cyklí (Prolog nenajde definitivně neúspěšný strom)
- Odvození pomocí NF pouze **test**, nelze **konstruovat** výslednou substituci
  - v $(\mathrm{comp}(P) \models \forall(\neg A))$ je $A$ všeob. kvantifikováno, v $\forall(\neg A)$ nejsou volné proměnné



# Normální a stratifikované programy

- **normální program:** obsahuje negativní literály v pravidlech
- problém: existence zúplnění, která nemají žádný model
  - $p :- \neg p.$ zúplnění: $p \leftrightarrow \neg p$
- rozdělení programu na vrstvy
  - vynucují použití negace relace pouze tehdy pokud je relace úplně definovaná
  - $a.$ $a.$
    $a :- \neg b, a.$ $a :- \neg b, a.$
    $b.$ $b :- \neg a.$
    stratifikovaný není stratifikovaný
- normální program $P$ je **stratifikovaný**: množina predikátových symbolů programu lze rozdělit do disjunktních množin $S_0, \ldots, S_m$ ($S_i \equiv$ **stratum**)
  - $p(\ldots) :- \ldots, q(\ldots), \ldots \in P,\ p \in S_k \Longrightarrow q \in S_0 \cup \ldots \cup S_k$
  - $p(\ldots) :- \ldots, \neg q(\ldots), \ldots \in P,\ p \in S_k \Longrightarrow q \in S_0 \cup \ldots \cup S_{k-1}$



# Stratifikované programy II

- program je $m$-**stratifikovaný** $\Longleftrightarrow$ $m$ je nejmenší index takový, že $S_0 \cup \ldots \cup S_m$ je množina všech predikátových symbolů z $P$
- **Věta:** Zúplnění každého stratifikovaného programu má Herbrandův model.
  - $p :- \neg p.$ nemá Herbrandův model
  - $p :- \neg p.$ ale není stratifikovaný
- stratifikované programy nemusí mít **jedinečný** minimální Herbrandův model
  - $cykli :- \neg zastavi.$
  - dva minimální Herbrandovy modely: $\{cykli\}, \{zastavi\}$
  - důsledek toho, že $cykli :- \neg zastavi.$ je ekvivalentní $cykli \vee zastavi$

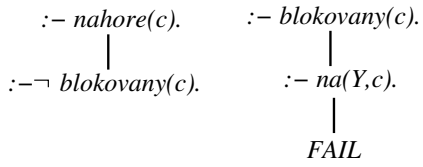


# SLDNF rezoluce: úspěšné odvození

- NF pravidlo: $\dfrac{:- C. \text{ má konečně neúspěšný SLD-strom}}{\neg C}$
- Pokud máme negativní podcíl $\neg C$ v dotazu $G$, pak hledáme důkaz pro $C$
- **Pokud odvození $C$ selže (strom pro $C$ je konečně neúspěšný), pak je odvození $G$ (i $\neg C$) celkově úspěšné**

$nahore(X) :- \neg blokovany(X).$
$blokovany(X) :- na(Y, X).$
$na(a, b).$

$:- nahore(c).$
$yes$



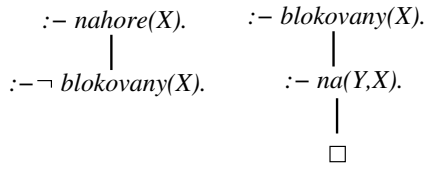

⇒ **uspěšné odvození**

# SLDNF rezoluce: neúspěšné odvození

- NF pravidlo: $\dfrac{:- C.\ \text{má konečně neúspěšný SLD-strom}}{\neg C}$
- Pokud máme negativní podcíl $\neg C$ v dotazu $G$, pak hledáme důkaz pro $C$
- **Pokud existuje vyvrácení $C$ s prázdnou substitucí (strom pro $C$ je konečně úspěšný), pak je odvození $G$ (i $\neg C$) celkově neúspěšné**

$nahore(X) :- \neg blokovany(X).$
$blokovany(X) :- na(Y,X).$
$na(_,_).$

$:- nahore(X).$
*no*




⇒ **neúspěšné odvození**

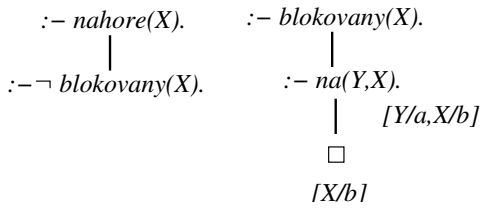


# SLDNF rezoluce: uvázlé odvození

- NF pravidlo: $\dfrac{:- C.\ \text{má konečně neúspěšný SLD-strom}}{\neg C}$
- Pokud máme negativní podcíl $\neg C$ v dotazu $G$, pak hledáme důkaz pro $C$
- **Pokud existuje vyvrácení $C$ s neprázdnou substitucí (strom pro $C$ je konečně úspěšný), pak je odvození $G$ (i $\neg C$) uvázlé**

$nahore(X) :- \neg blokovany(X).$
$blokovany(X) :- na(Y,X).$
$na(a,b).$

$:- nahore(X).$




⇒ **uvázlé odvození**



# SLD$^+$ odvození

- $P$ je normální program, $G_0$ normální cíl, $R$ selekční pravidlo:
  **SLD$^+$-odvození** $G_0$ je buď konečná posloupnost

  $$\langle G_0; C_0\rangle, \ldots, \langle G_{i-1}; C_{i-1}\rangle, G_i$$

  nebo nekonečná posloupnost

  $$\langle G_0; C_0\rangle, \langle G_1; C_1\rangle, \langle G_2; C_2\rangle, \ldots$$

  kde v každém kroku $m+1 (m \geq 0)$, $R$ vybírá **pozitivní literál** v $G_m$ a dospívá k $G_{m+1}$ obvyklým způsobem.

- konečné SLD$^+$-odvození může být:
  1. **úspěšné:** $G_i = \square$
  2. **neúspěšné**
  3. **blokované:** $G_i$ je negativní (např. $\neg A$)



# SLDNF rezoluce: pojmy

- **Úroveň cíle**
  - P normální program, $G_0$ normální cíl, $R$ selekční pravidlo:
    **úroveň cíle** $G_0$ se rovná
    - $\mathbf{0} \iff$ žádné SLD$^+$-odvození s pravidlem $R$ není blokováno
    - $k+\mathbf{1} \iff$ maximální úroveň cílů : $-A$,
      které ve tvaru $\neg A$ blokují SLD$^+$-odvození $G_0$, je $k$
  - nekonečná úroveň cíle: **blokované SLDNF odvození**
- **Množina SLDNF odvození** = {(SLDNF odvození $G_0$) $\cup$ (SLDNF odvození : $-A$)}
  - při odvozování $G_0$ jsme se dostali k cíli $\neg A$
- SLDNF odvození cíle $G$ ?

# SLDNF rezoluce

$P$ normální program, $G_0$ normální cíl, $R$ selekční pravidlo:

**množina SLDNF odvození a podmnožina neúspěšných SLDNF odvození** cíle $G_0$ jsou takové nejmenší množiny, že:

- každé **SLD$^+$-odvození** $G_0$ je SLDNF odvození $G_0$
- je-li SLD$^+$-odvození $\langle G_0; C_0\rangle, \ldots, G_i$ **blokováno na** $\neg A$
  - tj. $G_i$ je tvaru : $- L_1, \ldots, L_{m-1}, \neg A, L_{m+1}, \ldots, L_n$

  pak
  - **existuje-li SLDNF odvození** : $-A$ (pod $R$) s prázdnou cílovou substitucí, pak $\langle G_0; C_0\rangle, \ldots, G_i$ je **neúspěšné SLDNF odvození**
  - je-li **každé úplné SLDNF odvození** : $-A$ **(pod $R$) neúspěšné** pak $\langle G_0; C_0\rangle, \ldots, \langle G_i, \epsilon\rangle, (: - L_1, \ldots, L_{m-1}, L_{m+1}, \ldots, L_n)$ je **(úspěšné) SLDNF odvození cíle** $G_0$
    - $\epsilon$ označuje prázdnou cílovou substituci



# Typy SLDNF odvození

Konečné SLDNF-odvození může být:

1. **úspěšné:** $G_i = \square$
2. **neúspěšné**
3. **uvázlé (*flounder*):**
   $G_i$ je negativní ($\neg A$) a : $-A$ je úspěšné **s neprázdnou cílovou substitucí**
4. **blokované:** $G_i$ je negativní ($\neg A$) a : $-A$ nemá konečnou úroveň.



# Korektnost a úplnost SLDNF odvození

- **korektnost SLDNF-odvození:**
  $P$ normální program, : $-G$ normální cíl a $R$ je selekční pravidlo:
  je-li $\theta$ cílová substituce SLDNF-odvození cíle : $-G$, pak
  $G\theta$ je logickým důsledkem comp(P)
  - implementace SLDNF v Prologu není korektní
  - Prolog neřeší uvázlé SLDNF-odvození (neprázdná substituce)
  - použití bezpečných cílů (negace neobsahuje proměnné)
- **úplnost SLDNF-odvození:** SLDNF-odvození **není** úplné
  - pokud existuje konečný neúspěšný strom : $-A$, pak $\neg A$ platí

    ale místo toho se odvozování : $-A$ může zacyklit, tj. SLDNF rezoluce $\neg A$ neodvodí

  $\Rightarrow$ $\neg A$ tedy sice platí, ale SLDNF rezoluce ho nedokáže odvodit