

Operační a deklarativní semantika

Operační semantika

- **Operační semantikou** logického programu P rozumíme množinu $O(P)$ všech atomických formulí bez proměnných, které lze pro nějaký cíl G^1 odvodit nějakým rezolučním důkazem ze vstupní množiny $P \cup \{G\}$.
¹tímto výrazem jsou míněny všechny cíle, pro něž zmíněný rezoluční důkaz existuje.
- **Deklarativní semantika** logického programu P ???

Opakování: interpretace

- **Interpretace** \mathcal{I} jazyka \mathcal{L} je dána univerzem \mathcal{D} a zobrazením, které přiřadí konstantě c prvek \mathcal{D} , funkčnímu symbolu f/n n -ární operaci v \mathcal{D} a predikátovému symbolu p/n n -ární relaci.
 - příklad: $F = \{\{f(a, b) = f(b, a)\}, \{f(f(a, a), b) = a\}\}$
interpretace \mathcal{I}_1 : $\mathcal{D} = \mathbb{Z}, a := 1, b := -1, f := "+"$
- Interpretace se nazývá **modelem** formule, je-li v ní tato formule pravdivá
 - interpretace množiny \mathbb{N} s obvyklými operacemi je modelem formule $(0 + s(0) = s(0))$

Herbrandovy interpretace

- Omezení na obor skládající se ze **symbolických výrazů tvořených z predikátových a funkčních symbolů daného jazyka**
 - při zkoumání pravdivosti není nutné uvažovat modely nad všemi interpretacemi
- **Herbrandovo univerzum**: množina všech termů bez proměnných, které mohou být tvořeny funkčními symboly a konstantami daného jazyka
- **Herbrandova interpretace**: libovolná interpretace, která přiřazuje
 - proměnným prvky Herbrandova univerza
 - konstantám sebe samé
 - funkčním symbolům funkce, které symbolu f pro argumenty t_1, \dots, t_n přiřadí term $f(t_1, \dots, t_n)$
 - predikátovým symbolům libovolnou funkci z Herbrand. univerza do pravdivostních hodnot
- **Herbrandův model** množiny uzavřených formulí \mathcal{P} :
Herbrandova interpretace taková, že každá formule z \mathcal{P} je v ní pravdivá.

Specifikace Herbrandova modelu

- Herbrandovy interpretace mají předdefinovaný význam funktorů a konstant
- Pro specifikaci Herbrandovy interpretace tedy stačí zadat relace pro každý predikátový symbol
- Příklad: Herbrandova interpretace a Herbrandův model množiny formulí

$\text{lichy}(s(0)).$ % (1)

$\text{lichy}(s(s(X))) \text{ :- lichy}(X).$ % (2)

- $\mathcal{I}_1 = \emptyset$ není model (1)
- $\mathcal{I}_2 = \{\text{lichy}(s(0))\}$ není model (2)
- $\mathcal{I}_3 = \{\text{lichy}(s(0)), \text{lichy}(s(s(0)))\}$ není model (2)
- $\mathcal{I}_4 = \{\text{lichy}(s^n(0)) \mid n \in \{1, 3, 5, 7, \dots\}\}$ Herbrandův model (1) i (2)
- $\mathcal{I}_5 = \{\text{lichy}(s^n(0)) \mid n \in \mathbb{N}\}$ Herbrandův model (1) i (2)

Příklad: Herbrandovy interpretace

$\text{rodic}(a, b).$

$\text{rodic}(b, c).$

$\text{predek}(X, Y) \text{ :- rodic}(X, Y).$

$\text{predek}(X, Z) \text{ :- rodic}(X, Y), \text{predek}(Y, Z).$

$\mathcal{I}_1 = \{\text{rodic}(a, b), \text{rodic}(b, c), \text{predek}(a, b), \text{predek}(b, c), \text{predek}(a, c)\}$

$\mathcal{I}_2 = \{\text{rodic}(a, b), \text{rodic}(b, c),$
 $\text{predek}(a, b), \text{predek}(b, c), \text{predek}(a, c), \text{predek}(a, a)\}$

\mathcal{I}_1 i \mathcal{I}_2 jsou Herbrandovy modely klauzulí

Deklarativní a operační sémantika

- Je-li S množina programových klauzulí a M libovolná množina Herbrandových modelů S , pak **průnik těchto modelů** je opět Herbrandův model množiny S .
- Důsledek:**
Existuje **nejmenší Herbrandův model** množiny S , který značíme $M(S)$.
- Deklarativní sémantikou** logického programu P rozumíme jeho minimální Herbrandův model $M(P)$.
- Operační sémantikou** logického programu P rozumíme množinu $O(P)$ všech atomických formulí bez proměnných, které lze pro nějaký cíl G^1 odvodit nějakým rezolučním důkazem ze vstupní množiny $P \cup \{G\}$.
¹tímto výrazem jsou míněny všechny cíle, pro něž zmíněný rezoluční důkaz existuje.
- Pro libovolný logický program P platí $M(P) = O(P)$

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
 - $nad(X, Y) : \neg na(X, Y). \quad na(c, b).$
 $nad(X, Y) : \neg na(X, Z), nad(Z, Y). \quad na(b, a).$
 - 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 vyjádřena explicitně zřídka, např. jízdní řád

Negace jako neúspěch (*negation as failure*)

- slabší verze CWA: **definitivně neúspěšný (*finitely failed*) SLD-strom** cíle : $\neg A$
: $\neg A$ má definitivně (konečně) neúspěšný SLD-strom (*negation as failure, NF*)
 $\neg A$
- **normální cíl**: cíl obsahující i negativní literály
 - : $\neg nad(c, a), \neg nad(b, c).$
- rozdíl mezi CWA a NF
 - program $nad(X, Y) : \neg nad(X, Y)$, cíl : $\neg \neg nad(b, c)$
 - neexistuje odvození cíle podle NF, protože SLD-strom : $\neg nad(b, c)$ je nekonečný
 - existuje odvození cíle podle CWA, protože neexistuje vyvrácení : $\neg nad(b, c)$
- 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

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): $\frac{P \neq A}{\neg A}$ (CWA)
- pro SLD-rezoluci: $P \neq nad(a, c)$, 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.

Podstata zúplnění logického programu

- převod všech **if** příkazů v logickém programu na **iff**
 - $nad(X, Y) : \neg na(X, Y).$
 $nad(X, Y) : \neg na(X, Z), nad(Z, Y).$
 - lze psát jako: $nad(X, Y) : \neg (na(X, Y)) \vee (na(X, Z), nad(Z, Y)).$
 - 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: $na(X_1, X_2) : \neg X_1 = c, X_2 = b.$
 $na(X_1, X_2) : \neg X_1 = b, X_2 = a.$
 - zúplnění: $na(X_1, X_2) : \neg (X_1 = c, X_2 = b) \vee (X_1 = b, X_2 = a).$

Zúplnění programu

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

IF(q, P)

$$na(X_1, X_2) : \neg \exists Y (X_1 = c, X_2 = b, f(Y)) \vee (X_1 = b, X_2 = a, g).$$

- q/n predikátový symbol programu P $na(c, b) : \neg f(Y)$. $na(b, a) : \neg g$.
- X_1, \dots, X_n jsou „nové“ proměnné, které se nevyskytují nikde v P
- Necht' C je klauzule ve tvaru
 $q(t_1, \dots, t_n) : \neg L_1, \dots, L_m$
kde $m \geq 0$, t_1, \dots, t_n jsou termy a L_1, \dots, L_m jsou literály.
Pak označme $\text{E}(C)$ výraz $\exists Y_1, \dots, Y_k (X_1 = t_1, \dots, X_n = t_n, L_1, \dots, L_m)$
kde Y_1, \dots, Y_k jsou všechny proměnné v C .
- Necht' $\text{def}(q/n) = \{C_1, \dots, C_n\}$.
Pak formuli $\text{IF}(q, P)$ získáme následujícím postupem:
 $q(X_1, \dots, X_n) : \neg \text{E}(C_1) \vee \text{E}(C_2) \vee \dots \vee \text{E}(C_j)$ pro $j > 0$ a
 $q(X_1, \dots, X_n) : \neg \square$ 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 : $X_1 = Y_1 \wedge \dots \wedge X_m = Y_m \rightarrow f(X_1, \dots, X_m) = f(Y_1, \dots, Y_m)$
5. pro každý p/m : $X_1 = Y_1 \wedge \dots \wedge X_m = Y_m \rightarrow (p(X_1, \dots, X_m) \rightarrow p(Y_1, \dots, Y_m))$
6. pro všechny různé f/m a g/n , ($m, n \geq 0$): $f(X_1, \dots, X_m) \neq g(Y_1, \dots, Y_n)$
7. pro každý f/m : $f(X_1, \dots, X_m) = f(Y_1, \dots, Y_m) \rightarrow X_1 = Y_1 \wedge \dots \wedge X_m = Y_m$
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**: Necht' P logický program a $\neg A$ cíl.
Jestliže $\neg A$ má definitivně neúspěšný SLD-strom,
pak $\forall (\neg A)$ je logickým důsledkem $\text{comp}(P)$ (nebo-li $\text{comp}(P) \models \forall (\neg A)$)
- **Úplnost NF pravidla**: Necht' P je logický program. Jestliže $\text{comp}(P) \models \forall (\neg A)$,
pak existuje definitivně neúspěšný SLD-strom $\neg A$.
 - 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 $(\text{comp}(P) \models \forall (\neg A))$ 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, \dots, S_m ($S_i \equiv \text{stratum}$)
 - $p(\dots) : \dots, q(\dots), \dots \in P, p \in S_k \implies q \in S_0 \cup \dots \cup S_k$
 - $p(\dots) : \dots, \neg q(\dots), \dots \in P, p \in S_k \implies q \in S_0 \cup \dots \cup S_{k-1}$

Stratifikované programy II

- program je **m -stratifikovaný** $\iff m$ je nejmenší index takový, že $S_0 \cup \dots \cup S_m$ 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: $\frac{:- C. \text{ má konečně neúspěšný SLD-strom}}{\neg C}$
- 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

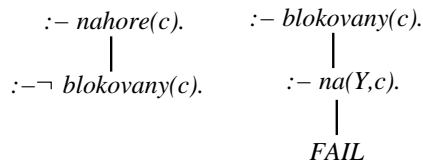
$nahore(X) : \neg blokovany(X).$

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

$na(a, b).$

$:- nahore(c).$

yes



\Rightarrow úspěšné odvození

SLDNF rezoluce: neúspěšné odvození

- NF pravidlo: $\frac{:- C. \text{ má konečně neúspěšný SLD-strom}}{\neg C}$
- 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

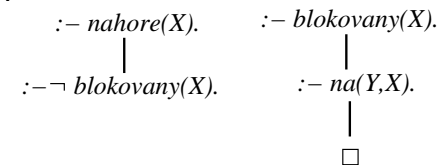
$nahore(X) : \neg blokovany(X).$

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

$na(_, _).$

$:- nahore(X).$

no



\Rightarrow neúspěšné odvození

SLDNF rezoluce: uvázné odvození

- NF pravidlo: $\frac{:- C. \text{ má konečně neúspěšný SLD-strom}}{\neg C}$
- 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 $\neg C$) uvázné**

$nahore(X) : \neg \neg blokovany(X).$

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

$na(a, b).$

$:- nahore(X).$ $:- nahore(X).$ $:- blokovany(X).$
 $:- \neg blokovany(X).$ $:- na(Y, X).$ $[Y/a, X/b]$
 \square
 $[X/b]$ \Rightarrow **uvázné odvození**

SLD⁺ odvození

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

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

nebo nekonečná posloupnost

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

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. **blokováné:** 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
 - $0 \iff$ žádné SLD⁺-odvození s pravidlem R není blokováno
 - $k + 1 \iff$ maximální úroveň cílů: $\neg A$,
které ve tvaru $\neg A$ blokují SLD⁺-odvození G_0 , je k
 - nekonečná úroveň cíle: **blokováné SLDNF odvození**
- **Množina SLDNF odvození** = $\{(\text{SLDNF odvození } G_0) \cup (\text{SLDNF odvození } : \neg A)\}$
 - 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 $\langle G_0; C_0 \rangle, \dots, G_i$ **blokováno na $\neg A$**
 - tj. G_i je tvaru: $\neg L_1, \dots, L_{m-1}, \neg A, L_{m+1}, \dots, L_n$
- pak

- **existuje-li SLDNF odvození: $\neg A$ (pod R) s prázdnou cílovou substitucí**, pak $\langle G_0; C_0 \rangle, \dots, G_i$ je **neúspěšné SLDNF odvození**
- **je-li každé úplné SLDNF odvození: $\neg A$ (pod R) neúspěšné** pak $\langle G_0; C_0 \rangle, \dots, \langle G_i; \epsilon \rangle, (: \neg L_1, \dots, L_{m-1}, L_{m+1}, \dots, L_n)$ je **(úspěšné) SLDNF odvození cíle G_0**
 - ϵ 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 : $\neg A$ je úspěšné s **neprázdnou cílovou substitucí**
4. **blokové:** G_i je negativní ($\neg A$) a : $\neg A$ nemá konečnou úroveň.

Korektnost a úplnost SLDNF odvození

- **korektnost SLDNF-odvození:**
 P normální program, : $\neg G$ normální cíl a R je selekční pravidlo:
je-li θ cílová substituce SLDNF-odvození cíle : $\neg G$, pak
 $G\theta$ je logickým důsledkem $\text{comp}(P)$
 - 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 : $\neg A$, pak $\neg A$ platí
ale místo toho se odvozování : $\neg A$ může zacyklit, tj. SLDNF rezoluce $\neg A$ neodvodí
 $\Rightarrow \neg A$ tedy sice platí, ale SLDNF rezoluce ho nedokáže odvodit