

Pátá přednáška

NAIL062 Výroková a predikátová logika

Jakub Bulín (KTIML MFF UK)

Zimní semestr 2024

Program

- rezoluční metoda
- korektnost a úplnost rezoluce
- úvod do predikátové logiky
- syntaxe predikátové logiky

Materiály

Zápisky z přednášky, Sekce 5.1-5.3 z Kapitoly 5 (Sekci 5.4 zatím přeskočíme), Sekce 6.1-6.3 z Kapitoly 6

KAPITOLA 5: REZOLUČNÍ METODA

Rezoluční metoda

- jiný důkazový systém než tablo metoda
- mnohem efektivnější implementace
- logické programování, automatické dokazování, SAT solvery (důkaz jako **certifikát** nesplnitelnosti)
- pracuje s CNF (každý výrok/teorii lze převést do CNF)
- jediné inferenční pravidlo: **rezoluční pravidlo**

$$\frac{\{p\} \cup C_1, \{\neg p\} \cup C_2}{C_1 \cup C_2}$$

- platí obecnější **pravidlo řezu**:

$$\frac{\varphi \vee \psi, \neg \varphi \vee \chi}{\psi \vee \chi}$$

5.1 Množinová reprezentace

Množinová reprezentace

- **literál** ℓ je p nebo $\neg p$ (pro $p \in \mathbb{P}$), $\bar{\ell}$ je **opačný literál** k ℓ
- **klauzule** C je konečná množina literálů
- **prázdná klauzule** \square je nespílitelná
- **CNF formule** S je množina klauzulí (může být i **nekonečná!**)
- **prázdná formule** \emptyset je vždy splněna

Modely reprezentujeme jako množiny literálů:

- **(částečné) ohodnocení** je libovolná **konzistentní** množina literálů (tj. nesmí obsahovat dvojici opačných literálů)
- **úplné ohodnocení** obsahuje p nebo $\neg p$ pro každý prvovýrok
- ohodnocení \mathcal{V} **splňuje** formuli S , píšeme $\mathcal{V} \models S$, pokud \mathcal{V} obsahuje nějaký literál z každé klauzule v S :

$$\mathcal{V} \cap C \neq \emptyset \text{ pro každou } C \in S$$

Množinová reprezentace: příklad

$$\varphi = (\neg p_1 \vee p_2) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_3 \vee \neg p_4) \wedge (\neg p_4 \vee \neg p_5) \wedge p_4$$

- v množinové reprezentaci:

$$S = \{\{\neg p_1, p_2\}, \{\neg p_1, \neg p_2, p_3\}, \{\neg p_3, \neg p_4\}, \{\neg p_4, \neg p_5\}, \{p_4\}\}$$

- ohodnocení $\mathcal{V} = \{\neg p_1, \neg p_3, p_4, \neg p_5\}$ splňuje S , $\mathcal{V} \models S$
- není úplné, můžeme rozšířit libovolným literálem pro p_2 , platí
 - $\mathcal{V} \cup \{p_2\} \models S$
 - $\mathcal{V} \cup \{\neg p_2\} \models S$
- tato dvě úplná ohodnocení odpovídají modelům
 - $(0, 1, 0, 1, 0)$
 - $(0, 0, 0, 1, 0)$

5.2 Rezoluční důkaz

Rezoluční pravidlo

Mějme klauzule C_1 a C_2 a literál ℓ takový, že $\ell \in C_1$ a $\bar{\ell} \in C_2$.
Potom **rezolventa** klauzulí C_1 a C_2 **přes literál** ℓ je klauzule:

$$C = (C_1 \setminus \{\ell\}) \cup (C_2 \setminus \{\bar{\ell}\})$$

tedy z první klauzule odstraníme ℓ a z druhé $\bar{\ell}$ (musely tam být!) a zbylé literály sjednotíme, mohli bychom také psát:

$$C'_1 \cup C'_2 \text{ je rezolventou klauzulí } C'_1 \dot{\cup} \{\ell\} \text{ a } C'_2 \dot{\cup} \{\bar{\ell}\}$$

- z klauzulí $C_1 = \{\neg q, r\}$ a $C_2 = \{\neg p, \neg q, \neg r\}$ odvodíme klauzuli $\{\neg p, \neg q\}$ přes literál r
- z $\{p, q\}$ a $\{\neg p, \neg q\}$ odvodíme $\{p, \neg p\}$ přes literál q , nebo $\{q, \neg q\}$ přes literál p (obojí jsou ale tautologie)
- nelze z nich ale odvodit \square “*rezolucí přes p a q najednou*”!
($S = \{\{p, q\}, \{\neg p, \neg q\}\}$ je splnitelná, např. $(1, 0)$ je model)

Rezoluční důkaz

Rezoluční pravidlo je **korektní**, tj. pro libovolné ohodnocení \mathcal{V} platí:

Pokud $\mathcal{V} \models C_1$ a $\mathcal{V} \models C_2$, potom $\mathcal{V} \models C$.

V rezolučním důkazu můžeme vždy napsat buď axiom, nebo rezolventu již napsaných klauzulí; tím zaručíme korektnost důkazů:

Rezoluční důkaz (odvození) klauzule C z formule S je konečná posloupnost klauzulí $C_0, C_1, \dots, C_n = C$ taková, že pro každé i :

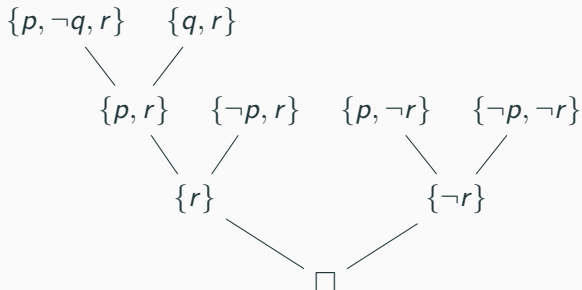
- $C_i \in S$, nebo
 - C_i je rezolventou nějakých C_j, C_k kde $j, k < i$
-
- existuje-li rez. důkaz, je C **rezolucí dokazatelná** z S , $S \vdash_R C$
 - **rezoluční zamítnutí** formule S je rezoluční důkaz \square z S
 - v tom případě je S **rezolucí zamítnutelná**

Příklad

Formule $S = \{\{p, \neg q, r\}, \{p, \neg r\}, \{\neg p, r\}, \{\neg p, \neg r\}, \{q, r\}\}$ je rezolucí zamítnutelná, jedno z možných zamítnutí je:

$$\{p, \neg q, r\}, \{q, r\}, \{p, r\}, \{\neg p, r\}, \{r\}, \{p, \neg r\}, \{\neg p, \neg r\}, \{\neg r\}, \square$$

Rezoluční důkaz má přirozeně stromovou strukturu, tzv. **rezoluční strom**: na listech jsou axiomy, vnitřní vrcholy jsou rezoluční kroky.



Rezoluční strom

Rezoluční strom klauzule C z formule S je konečný binární strom s vrcholy označenými klauzulemi, kde

- v kořeni je C ,
- v listech jsou klauzule z S ,
- v každém vnitřním vrcholu je rezolventa klauzulí ze synů tohoto vrcholu.

Pozorování: C má rezoluční strom z S , právě když $S \vdash_R C$.
(Důkaz snadno indukcí dle hloubky stromu a délky důkazu.)

- rezolučnímu důkazu odpovídá **jednoznačný** rezoluční strom
- z rezolučního stromu můžeme získat více důkazů (jsou dané libovolnou procházkou po vrcholech, která navštíví vnitřní vrchol až poté, co navštívila oba jeho syny)

Rezoluční uzávěr

jaké všechny klauzule se můžeme rezolucí 'naučit' z dané formule?
(není praktické je všechny najít, jde o užitečný teoretický pohled)

Rezoluční uzávěr $\mathcal{R}(S)$ formule S je definován induktivně jako nejmenší množina klauzulí splňující:

- $C \in \mathcal{R}(S)$ pro všechna $C \in S$,
- jsou-li $C_1, C_2 \in \mathcal{R}(S)$ a C jejich rezolventa, potom i $C \in \mathcal{R}(S)$

Pro $S = \{\{p, \neg q, r\}, \{p, \neg r\}, \{\neg p, r\}, \{\neg p, \neg r\}, \{q, r\}\}$ máme:

$$\begin{aligned}\mathcal{R}(S) = \{ & \{p, \neg q, r\}, \{p, \neg r\}, \{\neg p, r\}, \{p, s\}, \{q, r\}, \\ & \{p, \neg q\}, \{\neg q, r\}, \{r, \neg r\}, \{p, \neg p\}, \{r, s\}, \\ & \{p, r\}, \{p, q\}, \{r\}, \{p\}\}\end{aligned}$$

5.3 Korektnost a úplnost rezoluční metody

Korektnost rezoluce

Korektnost dokážeme snadno indukcí podle délky důkazu (nebo alternativně indukcí dle hloubky rezolučního stromu).

Věta (O korektnosti rezoluce): Je-li CNF formule S rezolucí zamítnutelná, potom je S nesplnitelná.

Důkaz: Nechť $S \vdash_R \square$, a vezměme nějaký rezoluční důkaz $C_0, C_1, \dots, C_n = \square$. **Sporem:** nechť existuje ohodnocení $\mathcal{V} \models S$. Indukcí podle i dokážeme, že $\mathcal{V} \models C_i$. Potom i $\mathcal{V} \models \square$, což je spor.

Pro $i = 0$ to platí, neboť $C_0 \in S$. Pro $i > 0$ máme dva případy:

- $C_i \in S$: v tom případě $\mathcal{V} \models C_i$ plyne z předpokladu, že $\mathcal{V} \models S$,
- C_i je rezolventou C_j, C_k , kde $j, k < i$: z indukčního předpokladu víme $\mathcal{V} \models C_j$ a $\mathcal{V} \models C_k$, $\mathcal{V} \models C_i$ plyne z korektnosti rezolučního pravidla □

Je-li S CNF formule a ℓ literál, potom **dosazení** ℓ do S je formule

$$S^\ell = \{C \setminus \{\bar{\ell}\} \mid \ell \notin C \in S\}$$

- S^ℓ je výsledkem **jednotkové propagace** aplikované na $S \cup \{\{\ell\}\}$.
- S^ℓ neobsahuje v žádné klauzuli literál ℓ ani $\bar{\ell}$ (vůbec tedy neobsahuje prvovýrok z ℓ)
- Pokud S neobsahovala literál ℓ ani $\bar{\ell}$, potom $S^\ell = S$.
- Pokud S obsahovala jednotkovou klauzuli $\{\bar{\ell}\}$, potom $\square \in S^\ell$, tedy S^ℓ je sporná.

Lemma: S je splnitelná, právě když je splnitelná S^ℓ nebo $S^{\bar{\ell}}$.

Důkaz: \Rightarrow Ohodnocení $\mathcal{V} \models S$ nemůže obsahovat ℓ i $\bar{\ell}$; BÚNO $\bar{\ell} \notin \mathcal{V}$. Ukážeme, že potom $\mathcal{V} \models S^\ell$.

Vezměme libovolnou klauzuli v S^ℓ . Ta je tvaru $C \setminus \{\bar{\ell}\}$ pro klauzuli $C \in S$ (neobsahující literál ℓ). Víme, že $\mathcal{V} \models C$, protože ale \mathcal{V} neobsahuje $\bar{\ell}$, muselo ohodnocení \mathcal{V} splnit nějaký jiný literál v C , takže platí i $\mathcal{V} \models C \setminus \{\bar{\ell}\}$.

\Leftarrow BÚNO mějme ohodnocení $\mathcal{V} \models S^\ell$. Protože se $\bar{\ell}$ (ani ℓ) nevyskytuje v S^ℓ , platí také $\mathcal{V} \setminus \{\bar{\ell}\} \models S^\ell$. Ohodnocení $\mathcal{V}' = (\mathcal{V} \setminus \{\bar{\ell}\}) \cup \{\ell\}$ potom splňuje všechny $C \in S$, tedy $\mathcal{V}' \models S$:

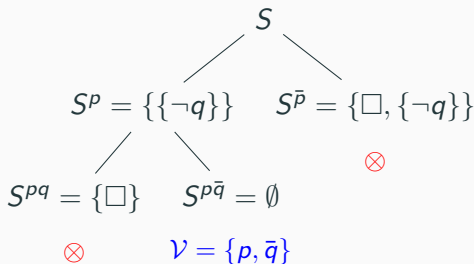
- pokud $\ell \in C$, potom $\ell \in C \cap \mathcal{V}'$ a $C \cap \mathcal{V}' \neq \emptyset$
- jinak $C \cap \mathcal{V}' = C \cap (\mathcal{V} \setminus \{\bar{\ell}\}) = (C \setminus \{\bar{\ell}\}) \cap (\mathcal{V} \setminus \{\bar{\ell}\}) \neq \emptyset$
neboť $\mathcal{V} \setminus \{\bar{\ell}\} \models C \setminus \{\bar{\ell}\} \in S^\ell$



Strom dosazení

Zda je *konečná* formule S splnitelná můžeme zjišťovat rekurzivně, dosazením obou literálů pro některý prvovýrok p , a rozvětvením na $S^p, S^{\bar{p}}$ (jako v DPLL). Výslednému stromu říkáme **strom dosazení**.

Např. pro $S = \{\{p\}, \{\neg q\}, \{\neg p, \neg q\}\}$:



- jakmile větev obsahuje \square , je nesplnitelná a nepokračujeme v ní
- listy jsou buď nesplnitelné, nebo prázdné teorie: v tom případě z posloupnosti dosazení získáme splňující ohodnocení.

Strom dosazení a nesplnitelnost

Důsledek: CNF formule S (ve spočetném jazyce, může být i nekonečná) je nesplnitelná, právě když každá větev stromu dosazení obsahuje \square .

Důkaz: Pro konečnou S snadno dokážeme indukcí dle $|\text{Var}(S)|$:

- Je-li $|\text{Var}(S)| = 0$, máme $S = \emptyset$ nebo $S = \{\square\}$, v obou případech je strom dosazení jednoprvkový a tvrzení platí.
- V indukčním kroku vybereme libovolný literál $\ell \in \text{Var}(S)$ a aplikujeme Lemma.

Je-li S nekonečná a splnitelná, má splňující ohodnocení, to se 'shoduje' s odpovídající (nekonečnou) větví ve stromu dosazení.

Je-li nekonečná a nesplnitelná, dle Věty o kompaktnosti existuje konečná $S' \subseteq S$, která je také nesplnitelná. Po dosazení pro všechny proměnné z $\text{Var}(S')$ bude v každé větvi \square , to nastane po konečně mnoha krocích.

Úplnost rezoluce

Věta (O úplnosti rezoluce): Je-li CNF formule S nesplnitelná, je rezolucí zamítnutelná (tj. $S \vdash_R \square$).

Důkaz: Je-li S nekonečná, má z kompaktnosti konečnou nesplnitelnou část, její rezoluční zamítnutí je také zamítnutí S .

Je-li S konečná, ukážeme indukcí dle počtu proměnných: Je-li $|\text{Var}(S)| = 0$, jediná možná nesplnitelná formule bez proměnných je $S = \{\square\}$, a máme jednokrokový důkaz $S \vdash_R \square$.

Jinak vyberme $p \in \text{Var}(S)$. Podle Lemmatu jsou S^p i $S^{\bar{p}}$ nesplnitelné. Mají o proměnnou méně, tedy dle ind. předpokladu existují rezoluční stromy T pro $S^p \vdash_R \square$ a T' pro $S^{\bar{p}} \vdash_R \square$.

Ukážeme, jak z T vyrobit rezoluční strom \hat{T} pro $S \vdash_R \neg p$. Analogicky \hat{T}' pro $S \vdash_R p$ a potom už snadno vyrobíme rezoluční strom pro $S \vdash_R \square$: ke kořeni \square připojíme kořeny stromů \hat{T} a \hat{T}' jako levého a pravého syna (tj. získáme \square rezolucí z $\{\neg p\}$ a $\{p\}$).

Dokončení důkazu

Rezoluční strom T pro $S^p \vdash_R \Box \rightsquigarrow \hat{T}$ pro $S \vdash_R \neg p$:

Vrcholy i uspořádání jsou stejné, jen do některých klauzulí ve vrcholech přidáme literál $\neg p$.

Na každém listu stromu T je nějaká klauzule $C \in S^p$, a

- buď $C \in S$,
- nebo $C \notin S$, ale $C \cup \{\neg p\} \in S$

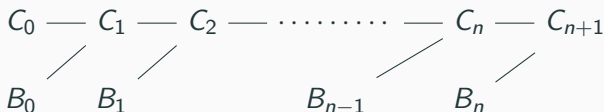
V prvním případě necháme label stejný. Ve druhém případě přidáme do C a do všech klauzulí nad tímto listem literál $\neg p$.

Listy jsou nyní klauzule z S , a každý vnitřní vrchol je nadále rezolventou svých synů. V kořeni jsme \Box změnili na $\neg p$ (leďže každý list T už byl klauzule z S , to ale už T dává $S \vdash_R \Box$). \square

**Zatím přeskočíme: 5.4 LI-rezoluce a
Horn-SAT**

Lineární důkaz: neformálně

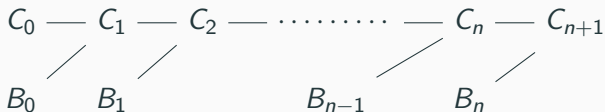
Rezoluční důkaz můžeme kromě rezolučního stromu zorganizovat i jinak, jako tzv. **lineární důkaz**:



- v každém kroku máme jednu **centrální** klauzuli
- tu resolvujeme s **boční** ('side') klauzulí
- boční klauzule je buď axiom z S , nebo některá z předchozích centrálních (jako bychom odvozené klauzule přidávali k axiomům)
- výsledná **rezolventa** je novou **centrální klauzulí**

(Tento pohled lépe odpovídá procedurálnímu výpočtu, jde jen o to, jak vybírat vhodné boční klauzule.)

Lineární důkaz: formálně



Lineární důkaz klauzule C z formule S je konečná posloupnost

$$\left[\begin{array}{c} C_0 \\ B_0 \end{array} \right], \left[\begin{array}{c} C_1 \\ B_1 \end{array} \right], \dots, \left[\begin{array}{c} C_n \\ B_n \end{array} \right], C_{n+1}$$

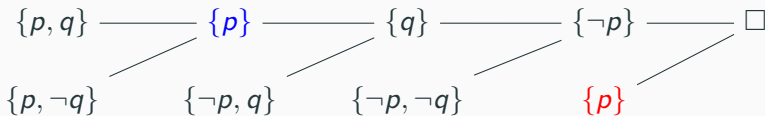
kde C_i říkáme **centrální** klauzule, C_0 je **počáteční**, $C_{n+1} = C$ je **koncová**, B_i jsou **boční** klauzule, a platí:

- $C_0 \in S$, pro $i \leq n$ je C_{i+1} rezolventou C_i a B_i ,
- $B_0 \in S$, pro $i \leq n$ je $B_i \in S$ nebo $B_i = C_j$ pro nějaké $j < i$.

Lineární zamítnutí S je lineární důkaz \square z S .

Příklad a ekvivalence s rezolučním důkazem

Lineární zamítnutí $S = \{\{p, q\}, \{p, \neg q\}, \{\neg p, q\}, \{\neg p, \neg q\}\}$:



Poslední boční klauzule $\{p\}$ není z S , ale je rovna předchozí centrální klauzuli.

Poznámka: C má lineární důkaz z S , právě když $S \vdash_B C$.

⇒ Z lineárního důkazu snadno vyrobíme rezoluční strom. Indukcí dle délky důkazu: máme-li boční klauzuli $B_i \notin S$, potom $B_i = C_j$ pro nějaké $j < i$: místo B_i připojíme rezoluční strom pro C_j z S .

⇒ Plyne z úplnosti lineární rezoluce, důkaz najdete v učebnici.

- **lineární důkaz**: boční klauzule je **axiom nebo dřívější centrální**
- co když požadujeme, aby boční klauzule byly **pouze axiomy**?
⇒ **LI-rezolute (linear-input)**

LI-důkaz (rezolucí) klauzule C z formule S je lineární důkaz

$$\left[\begin{array}{c} C_0 \\ B_0 \end{array} \right], \left[\begin{array}{c} C_1 \\ B_1 \end{array} \right], \dots, \left[\begin{array}{c} C_n \\ B_n \end{array} \right], C$$

ve kterém je každá boční klauzule B_i axiom z S . Pokud LI-důkaz existuje, říkáme, že je C **LI-dokazatelná** z S , a píšeme $S \vdash_{LI} C$. Pokud $S \vdash_{LI} \square$, je S **LI-zamítnutelná**.

- LI-důkaz odpovídá rezolučnímu stromu tvaru “*chlupaté cesty*”
- z toho plyne korektnost
- ztrácíme úplnost, ale hledání důkazů je snazší
- ukážeme **úplnost pro Hornovy formule**, je základem Prologu

Hornovy formule

- **Hornova klauzule** má nejvýše jeden pozitivní literál
- **Hornova formule** je množina Hornových klauzulí (i nekonečná)
- **Fakt** je pozitivní jednotková klauzule, např. $\{p\}$
- **Pravidlo** je klauzule s právě jedním pozitivním a alespoň jedním negativním literálem
- Pravidlům a faktům říkáme **programové klauzule**
- **Cíl** je neprázdná klauzule bez pozitivního literálu
- dokazujeme sporem: **cíl** je negací toho, co chceme dokázat (konjunkce faktů)

Pozorování: Je-li Hornova formule S nesplnitelná a $\square \notin S$, potom obsahuje fakt i cíl.

Důkaz: Neobsahuje-li fakt, ohodnotíme všechny proměnné 0; neobsahuje-li cíl, ohodnotíme 1. □

Příklad konstrukce LI-zamítnutí

Ukážeme: $T = \{\{p, \neg r, \neg s\}, \{\neg q, r\}, \{q, \neg s\}, \{s\}\} \models p \wedge q$

Sestrojíme LI-zamítnutí $T \cup \{G\} \vdash_{LI} \square$ pro cíl $G = \{\neg p, \neg q\}$.

V T najdeme fakt, a provedeme jednotkovou propagaci v $T \cup \{G\}$.

Opakujeme, dokud není formule prázdná:

- $T = \{\{p, \neg r, \neg s\}, \{\neg q, r\}, \{q, \neg s\}, \{s\}\}, G = \{\neg p, \neg q\}$
- $T^s = \{\{p, \neg r\}, \{\neg q, r\}, \{q\}\}, G^s = \{\neg p, \neg q\}$
- $T^{sq} = \{\{p, \neg r\}, \{r\}\}, G^{sq} = \{\neg p\}$
- $T^{sqr} = \{\{p\}\}, G^{sqr} = \{\neg p\}$
- $T^{sqrp} = \emptyset, G^{sqrp} = \square$

To, že vždy najdeme fakt, plyne z Pozorování pro $T \cup \{G\}$.

Nyní zpětným postupem sestrojíme LI-zamítnutí, podobně jako v důkazu úplnosti rezoluce.

Konstrukce zamítnutí zpětným postupem

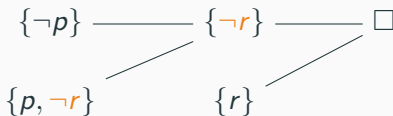
- $T^{sqr p}, G^{sqr p} \vdash_{LI} \square$:

\square

- $T^{sqr}, G^{sqr} \vdash_{LI} \square$:

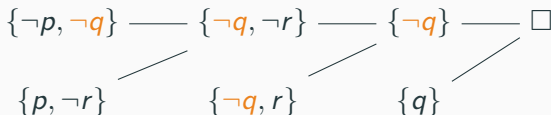


- $T^{sq}, G^{sq} \vdash_{LI} \square$:



Konstrukce zamítnutí zpětným postupem – pokračování

- $T^s, G^s \vdash_{LI} \square$:



- $T, G \vdash_{LI} \square$



Úplnost pro Hornovy formule

Věta (O úplnosti LI-rezoluce pro Hornovy formule): Je-li Hornova formule T splnitelná, a $T \cup \{G\}$ je nespjitelná pro cíl G , potom $T \cup \{G\} \vdash_{LI} \square$, a to LI-zamítnutím, které začíná cílem G .

Důkaz: Opět lze díky Větě o kompaktnosti předpokládat, že T je konečná. LI-zamítnutí sestojíme indukcí podle počtu proměnných.

Z Pozorování víme, že T obsahuje fakt $\{p\}$. Protože $T \cup \{G\}$ je nespjitelná, je dle Lemmatu o dosazení **nespjitelná i** $(T \cup \{G\})^p = T^p \cup \{G^p\}$, kde $G^p = G \setminus \{\neg p\}$.

Všimněte si, že **T^p je splnitelná**. (Stejným ohodnocením jako T , neboť to musí obsahovat p kvůli faktu $\{p\}$, tedy neobsahuje $\neg p$.)

Zároveň má T^p méně proměnných než T . **Je-li G^p cíl**, využijeme indukčního předpokladu (následující slide). Co když ale G^p není cíl?

Není-li G^P cíl, nutně $G^P = \square$ a $G = \{\neg p\}$. Potom je \square rezolventou G a faktu $\{p\} \in T$, a máme jednokrokové LI-zamítnutí $T \cup \{G\}$. (To dává i bázi indukce.)

Je-li G^P cíl, dle indukčního předpokladu existuje LI-odvození \square z $T^P \cup \{G^P\}$ začínající $G^P = G \setminus \{\neg p\}$.

Hledané LI-zamítnutí $T \cup \{G\}$ začínající G zkonstruujeme (podobně jako v důkazu Věty o úplnosti rezoluce):

- Přidáme literál $\neg p$ do všech listů, které už nejsou v $T \cup \{G\}$ (vznikly odebráním $\neg p$), a do všech vrcholů nad nimi.
- Tím získáme $T \cup \{G\} \vdash_{LI} \neg p$.
- Na závěr přidáme boční klauzuli $\{p\}$ a odvodíme \square . \square

Program v Prologu

síla Prologu vychází z **unifikace** a rezoluce v predikátové logice, nyní si ale ukážeme příklad **výrokového** programu:

- **program** v Prologu je Hornova formule obsahující pouze **programové klauzule**, tj. **fakta** nebo **pravidla**
- **dotaz** je konjunkce faktů, negace dotazu je **cíl**

Např. program $\{\{p, \neg r, \neg s\}, \{\neg q, r\}, \{q, \neg s\}, \{s\}\}$, dotaz $p \wedge q$

- klauzule $\{p, \neg r, \neg s\}$ je ekvivalentní $r \wedge s \rightarrow p$, píšeme $p:-r,s$.
- výsledný program a dotaz:

$p:-r,s.$

$r:-q.$

$q:-s.$

$s.$

$?-p,q.$

Důsledek: Mějme program P a dotaz $Q = p_1 \wedge \dots \wedge p_n$, a označme $G = \{\neg p_1, \dots, \neg p_n\}$ (tj. $G \sim \neg Q$). Následující podmínky jsou ekvivalentní:

- (i) $P \models Q$,
- (ii) $P \cup \{G\}$ je nespílitelná,
- (iii) $P \cup \{G\} \vdash_{LI} \square$, a existuje LI-zamítnutí začínající cílem G .

Důkaz:

- (i) \Leftrightarrow (ii) Věta o důkazu sporem
- (ii) \Leftrightarrow (iii) Věta o úplnosti LI-rezoluce pro Hornovy formule
(Program je vždy splnitelný) □

ČÁST II – PREDIKÁTOVÁ LOGIKA

KAPITOLA 6: SYNTAXE A SÉMANTIKA PREDIKÁTOVÉ LOGIKY

6.1 Úvod

Výroková logika: popis světa pomocí **výroků** složených z **prvovýroků** (**výrokových proměnných**) – bitů informace

Predikátová logika [prvního řádu]:

- základní stavební kámen jsou **proměnné** reprezentující **individa** – nedělitelné objekty z nějaké množiny (např. přirozená čísla, vrcholy grafu, stavy mikroprocesoru)
- tato individua mají určité vlastnosti a vzájemné vztahy (**relace**), kterým říkáme **predikáty**
 - $\text{Leaf}(x)$ nebo $\text{Edge}(x, y)$ mluvíme-li o grafu
 - $x \leq y$ v přirozených číslech
- a mohou vstupovat do **funkcí**
 - $\text{lowest_common_ancestor}(x, y)$ v zakořeněném stromu
 - $\text{succ}(x)$ nebo $x + y$ v přirozených číslech
- a mohou být **konstantami** se speciálním významem, např. **root** v zakořeněném stromu, **0** v tělese.

Syntaxe neformálně

- **atomické formule**: predikát (včetně **rovnosti** $=$) o proměnných nebo o **termech** ('výrazy' složené z funkcí popř. konstant)
- **formule** jsou složené z atomických formulí pomocí logických spojek, a dvou **kvantifikátorů**:

$\forall x$ "pro všechna individua (reprezentovaná proměnnou x)"

$\exists x$ "existuje individuum (reprezentované proměnnou x)"

Např. "*Každý, kdo má dítě, je rodič.*" lze formalizovat takto:

$$(\forall x)((\exists y)\text{child_of}(y, x) \rightarrow \text{is_parent}(x))$$

- **child_of**(y, x) je binární predikát vyjadřující, že individuum reprezentované proměnnou y je dítětem individua reprezentovaného proměnnou x
- **is_parent**(x) je unární predikát vyjadřující, že individuum reprezentované x je rodič

$$(\forall x)((\exists y)\text{child_of}(y, x) \rightarrow \text{is_parent}(x))$$

Platnost? Záleží na **modelu** světa/systému, který nás zajímá:

Model je...

- (neprázdná) množina individuí, spolu
- s binární relací **interpretující** binární relační symbol **child_of**, a
- s unární relací (tj. podmnožinou) interpretující unární relační symbol **is_parent**

Obecně mohou být relace jakékoliv, snadno sestojíme model, ve kterém formule neplatí, např.

$$\mathcal{A} = \langle \{0, 1\}, \{(0, 0), (0, 1), (1, 0), (1, 1)\}, \emptyset \rangle$$

Příklad s funkcemi a konstantami

“Je-li $x_1 \leq y_1$ a $x_2 \leq y_2$, potom platí $(y_1 \cdot y_2) - (x_1 \cdot x_2) \geq 0$.”

$$\varphi = (x_1 \leq y_1) \wedge (x_2 \leq y_2) \rightarrow ((y_1 \cdot y_2) + (-(x_1 \cdot x_2))) \geq 0$$

- dva binární relační symboly (\leq, \geq), binární funkční symbol $+$, unární funkční symbol $-$, a konstantní symbol 0
- **model, ve kterém φ platí:** \mathbb{N} s binárními relacemi $\leq^{\mathbb{N}}, \geq^{\mathbb{N}}$, bin. funkcemi $+^{\mathbb{N}}, \cdot^{\mathbb{N}}$, unární funkcí $-^{\mathbb{N}}$, a konstantou $0^{\mathbb{N}} = 0$
- vezmeme-li ale podobně množinu \mathbb{Z} , φ už platit nebude

Poznámky:

- mohli bychom chápat ‘ $-$ ’ jako binární, obvykle ale bývá unární
- pro **konstantní symbol** 0 používáme (jak je zvykem) stejný symbol, jako pro přirozené číslo 0 . Ale pozor, v našem modelu může být **symbol** 0 interpretován jako **jiné číslo**, nebo náš model vůbec nemusí sestávat z čísel!

$$\varphi = (x_1 \leq y_1) \wedge (x_2 \leq y_2) \rightarrow ((y_1 \cdot y_2) + (-(x_1 \cdot x_2)) \geq 0)$$

- φ nemá žádné kvantifikátory, tj. je **otevřená**
- x_1, x_2, y_1, y_2 jsou **volné proměnné** této formule (nejsou **vázané** žádným kvantifikátorem), píšeme $\varphi(x_1, x_2, y_1, y_2)$
- sémantiku φ chápeme stejně jako $(\forall x_1)(\forall x_2)(\forall y_1)(\forall y_2)\varphi$
- používáme **konvence** (infixový zápis, vynechání závorek), jinak:

$$\varphi = (((\leq(x_1, y_1) \wedge \leq(x_2, y_2)) \rightarrow \leq(+(\cdot(y_1, y_2), -(\cdot(x_1, x_2)))), 0))$$

- cvičení: definujte **strom formule**, nakreslete ho pro φ

Termy vs. atomické formule

$$\varphi = (x_1 \leq y_1) \wedge (x_2 \leq y_2) \rightarrow ((y_1 \cdot y_2) + (-(x_1 \cdot x_2)) \geq 0)$$

- výraz $(y_1 \cdot y_2) + (-(x_1 \cdot x_2))$ je **term**
- výrazy $(x_1 \leq y_1)$, $(x_2 \leq y_2)$ a $((y_1 \cdot y_2) + (-(x_1 \cdot x_2)) \geq 0)$ jsou (všechny) **atomické (pod)formule** φ

V čem je rozdíl? Máme-li konkrétní model, a konkrétní **ohodnocení proměnných** individui (prvky) tohoto modelu:

- výsledkem termu (při daném ohodnocení proměnných) je konkrétní **individuum z modelu**, zatímco
- atomickým formulí lze přiřadit **pravdivostní hodnotu** (a tedy kombinovat je logickými spojkami)

6.2 Struktury

- specifikuje jakého **typu** bude daná struktura, tj. jaké má relace, funkce (jakých arit) a konstanty, a symboly pro ně
- **konstanty** lze chápat jako funkce arity 0, tj. funkce bez vstupů

Signatura je dvojice $\langle \mathcal{R}, \mathcal{F} \rangle$, kde \mathcal{R}, \mathcal{F} jsou disjunktní množiny symbolů (**relační** a **funkční**, ty zahrnují **konstantní**) spolu s danými aritami (tj. danými funkcí $ar: \mathcal{R} \cup \mathcal{F} \rightarrow \mathbb{N}$) a neobsahující symbol '=' (ten je rezervovaný pro **rovnost**).

- často zapíšeme jen výčet symbolů, jsou-li arity a zda jsou relační nebo funkční zřejmé
- kromě běžně používaných symbolů typicky používáme:
 - pro relační symboly P, Q, R, \dots
 - pro funkční (nekonstantní) symboly f, g, h, \dots
 - pro konstantní symboly c, d, a, b, \dots

Příklady signatur

- $\langle E \rangle$ signatura **grafů**: E je binární relační symbol (strukтуры jsou uspořádané grafy)
- $\langle \leq \rangle$ signatura **částečných uspořádání**: stejná jako signatura grafů, jen jiný symbol (ne každá struktura v této signatuře je částečné uspořádání! k tomu musí splňovat příslušné **axiomy**)
- $\langle +, -, 0 \rangle$ signatura **grup**: $+$ je binární funkční, $-$ unární funkční, 0 konstantní symbol
- $\langle +, -, 0, \cdot, 1 \rangle$ signatura **těles**: \cdot je binární funkční, 1 konstantní symbol
- $\langle +, -, 0, \cdot, 1, \leq \rangle$ signatura **uspořádaných těles**: \leq je binární relační symbol
- $\langle -, \wedge, \vee, \perp, \top \rangle$ signatura **Booleových algeber**: \wedge, \vee jsou binární funkční, \perp, \top jsou konstantní symboly
- $\langle S, +, \cdot, 0, \leq \rangle$ signatura **aritmetiky**: S je unární funkční symbol

Strukturu dané signatury získáme tak, že:

- zvolíme neprázdnou **doménu**, a na ní
- zvolíme **realizace** (také říkáme **interpretace**) všech relačních a funkčních symbolů (včetně konstantních)
- to znamená **konkrétní** relace resp. funkce příslušných arit
- realizací konstantního symbolu je zvolený prvek z domény
- na tom, jaké konkrétní symboly jsou v signatuře nezáleží (např. $+$ neznamená, že realizace musí souviset se sčítáním)

- Struktura v **prázdné signatuře** $\langle \rangle$ je libovolná neprázdná množina. (Nemusí být konečná, ani spočetná! Formálně to bude trojice $\langle A, \emptyset, \emptyset \rangle$, ale rozdíl zanedbáme.)
- Struktura v **signatuře grafů** je $\mathcal{G} = \langle V, E \rangle$, kde $V \neq \emptyset$ a $E \subseteq V^2$, říkáme jí **orientovaný graf**.
 - je-li E ireflexivní a symetrická, je to **jednoduchý graf**
 - je-li E reflexivní, tranzitivní, a antisymetrická, jde o **částečné uspořádání**
 - je-li E reflexivní, tranzitivní, a symetrická, je to **ekvivalence**
- Struktury v **signatuře částečných uspořádání** jsou tytéž, jako v signatuře grafů, signatury se liší jen symbolem. (Ne každá struktura v signatuře částečných uspořádání je č. uspořádání!)

Struktury v signatuře grup jsou například následující grupy:

- $\underline{\mathbb{Z}}_n = \langle \mathbb{Z}_n, +, -, 0 \rangle$, aditivní grupa celých čísel modulo n (operace jsou modulo n).

Poznámka: $\underline{\mathbb{Z}}_n$ znamená strukturu, zatímco \mathbb{Z}_n jen její doménu. Často se to ale nerozlišuje a \mathbb{Z}_n se používá i pro strukturu. Podobně $+$, $-$, 0 jsou jak symboly, tak interpretace.

- $\mathcal{S}_n = \langle \text{Sym}_n, \circ, {}^{-1}, \text{id} \rangle$ je symetrická grupa (grupa všech permutací) na n prvcích.
- $\underline{\mathbb{Q}}^* = \langle \mathbb{Q} \setminus \{0\}, \cdot, {}^{-1}, 1 \rangle$ je multiplikativní grupa (nenulových) racionálních čísel. (Interpretací symbolu 0 je číslo $1!$)

Všechny tyto struktury splňují axiomy teorie grup, snadno ale najdeme jiné, které axiomy nesplňují, nejsou tedy grupami.

- Struktury $\underline{\mathbb{Q}} = \langle \mathbb{Q}, +, -, 0, \cdot, 1, \leq \rangle$ a $\underline{\mathbb{Z}} = \langle \mathbb{Z}, +, -, 0, \cdot, 1, \leq \rangle$ (se standardními operacemi a uspořádáním) jsou v signatuře uspořádaných těles (ale jen první z nich je uspořádané těleso).
- $\underline{\mathcal{P}(X)} = \langle \mathcal{P}(X), \neg, \cap, \cup, \emptyset, X \rangle$, tzv. potenční algebra nad množinou X , je struktura v signatuře Booleových algeber. (Booleova algebra je to pokud $X \neq \emptyset$.)
- $\underline{\mathbb{N}} = \langle \mathbb{N}, S, +, \cdot, 0, \leq \rangle$, kde $S(x) = x + 1$, a ostatní symboly jsou realizovány standardně, je standardní model aritmetiky.

Struktura v signatuře $\langle \mathcal{R}, \mathcal{F} \rangle$ je trojice $\mathcal{A} = \langle A, \mathcal{R}^{\mathcal{A}}, \mathcal{F}^{\mathcal{A}} \rangle$, kde

- A je neprázdná množina, říkáme jí **doména** (také **univerzum**),
- $\mathcal{R}^{\mathcal{A}} = \{R^{\mathcal{A}} \mid R \in \mathcal{R}\}$ kde $R^{\mathcal{A}} \subseteq A^{\text{ar}(R)}$ je **interpretace** relačního symbolu R ,
- $\mathcal{F}^{\mathcal{A}} = \{f^{\mathcal{A}} \mid f \in \mathcal{F}\}$ kde $f^{\mathcal{A}}: A^{\text{ar}(f)} \rightarrow A$ je **interpretace** funkčního symbolu f (speciálně pro konstantní symbol $c \in \mathcal{F}$ máme $c^{\mathcal{A}} \in A$).

Příklad: rozmyslete si, jak vypadají struktury v **signatuře** n konstant $\langle c_1, c_2, \dots, c_n \rangle$? Popište všechny 5-prvkové v signatuře 3 konstant.

6.3 Syntaxe

Jazyk je daný **signaturou** a informací, zda je **s rovností** nebo ne.

Tj. specifikujeme 'typ' modelů a zda můžeme používat symbol '=' interpretovaný jako **identita** prvků z domény; většinou to dovolíme. (Je-li jazyk bez rovnosti, musí mít signatura relační symbol. Proč?)

Do jazyka patří:

- spočetně mnoho **proměnných** x_0, x_1, x_2, \dots (píšeme také x, y, z, \dots ; množinu všech proměnných označíme **Var**)
- **relační, funkční a konstantní symboly** ze signatury, symbol = jde-li o jazyk s rovností (to jsou '**mimologické**' symboly)
- **univerzální a existenční kvantifikátory** $(\forall x), (\exists x)$ pro každou proměnnou $x \in \text{Var}$ (kvantifikátor ' $(\forall x)$ ' chápeme jako jediný symbol, tj. **neobsahuje** proměnnou x)
- symboly pro log. spojky $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$, závorky $(,)$, a čárka ','

- Jazyk $L = \langle \rangle$ s rovností je jazyk **čisté rovnosti**
- jazyk $L = \langle c_0, c_1, c_2, \dots \rangle$ s rovností je jazyk **spočetně mnoha konstant**
- jazyk **uspořádání** je $\langle \leq \rangle$ s rovností
- jazyk **teorie grafů** je $\langle E \rangle$ s rovností
- jazyky **teorie grup, teorie těles, teorie uspořádaných těles, Booleových algeber, aritmetiky** jsou jazyky s rovností odpovídající daným signaturám

čistě syntaktické 'výrazy' z proměnných, konstantních symbolů, funkčních symbolů, závorek a čárek

Termy jazyka L jsou konečné nápisy definované induktivně:

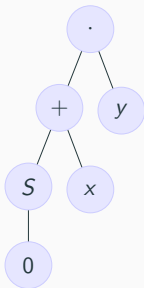
- každá proměnná a každý konstantní symbol z L je term,
- je-li f funkční symbol z L arity n a jsou-li t_1, \dots, t_n termy, potom nápis $f(t_1, t_2, \dots, t_n)$ je také term.

Množinu všech **termů** jazyka L označíme Term_L .

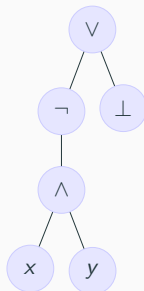
- **podterm** je podřetězec, který je sám termem
- term bez proměnných je **konstantní (ground)**, např. $((S(0) + S(0)) \cdot S(S(0)))$ v jazyce aritmetiky
- termy nesmí obsahovat prvky struktury, jen symboly z jazyka
- $(1 + 1) \cdot 2$ **není** term, ledaže rozšíříme jazyk o **symboly** 1 a 2
- jako lidé můžeme použít **infixový** zápis, např. $(t_1 + t_2)$ místo $+(t_1, t_2)$, vynechat závorky je-li struktura termu zřejmá

Strom termu

Strom termu t , $\text{Tree}(t)$: v listech proměnné nebo konst. symboly, ve vnitřních vrcholech funkční symboly (arita je rovna počtu synů)



(a) $(S(0) + x) \cdot y$ v jazyce aritmetiky



(b) $\neg(x \wedge y) \vee \perp$ v jazyce Booleových algeber

- symboly \neg, \wedge, \vee nejsou logické, ale mimologické ze signatury
- **sémantika**: proměnné ohodnotíme prvky, konst. a funkční symboly nahradíme interpretacemi, výsledek je prvek z domény

Atomické formule

Termům nelze přiřadit **pravdivostní hodnotu**, potřebujeme **predikát** (relační symbol nebo $=$), který mluví o **'vztahu' termů**: v dané struktuře při ohodnocení proměnných prvky je buď splněn, nebo ne.

Formule ('tvrzení o strukturách') skládáme z **atomických formulí** pomocí logických spojek a kvantifikátorů:

Atomická formule jazyka L je nápis $R(t_1, \dots, t_n)$, kde R je n -ární relační symbol z L (včetně $=$ jde-li o jazyk s rovností) a $t_i \in \text{Term}_L$.

- $R(f(f(x)), c, f(d))$ kde R je ternární relační, f unární funkční, c, d konstantní symboly
- **infixový zápis** $\leq(x, y), = (t_1, t_2)$ píšeme jako $x \leq y, t_1 = t_2$
- $(x \cdot x) + (y \cdot y) \leq (x + y) \cdot (x + y)$ v jazyce uspořádaných těles
- $x \cdot y \leq (S(0) + x) \cdot y$ v jazyce aritmetiky
- $\neg(x \wedge y) \vee \perp = \perp$ v jazyce Booleových algeber

Formule jazyka L jsou konečné nápisy definované induktivně:

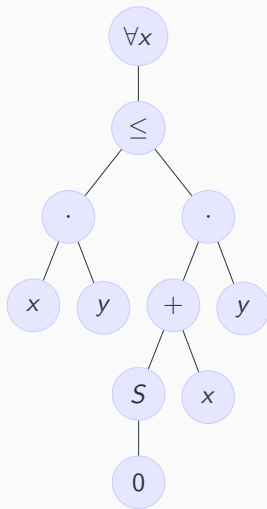
- každá **atomická formule** jazyka L je formule,
 - je-li φ formule, potom $(\neg\varphi)$ je také formule
 - jsou-li φ, ψ formule, potom $(\varphi \square \psi)$ pro $\square \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ jsou také formule
 - je-li φ formule a x proměnná, potom $((Qx)\varphi)$ pro $Q \in \{\forall, \exists\}$ jsou také formule
-
- **podformule** je podřetězec, který je sám formulí
 - při zápisu formulí jako lidé používáme obvyklé konvence
 - kvantifikátory mají stejnou prioritu jako \neg , vyšší než ostatní logické spojky! místo $((\forall x)\varphi)$ píšeme $(\forall x)\varphi$
 - **pozor, $(\forall x)\varphi \wedge \psi$ neznamená totéž, co $(\forall x)(\varphi \wedge \psi)$!**
 - někde uvidíte $\forall x\varphi$ nebo $\forall_x\varphi$, my ale budeme psát jen $(\forall x)\varphi$

Strom formule

Příklad: $(\forall x)(x \cdot y \leq (S(0) + x) \cdot y)$

Strom formule, $\text{Tree}(\varphi)$:

- strom atomické formule $R(t_1, \dots, t_n)$:
v kořeni R , připojíme stromy $\text{Tree}(t_i)$
- pro složené formule podobně jako ve
výrokové logice
- kvantifikátory mají jediného syna



Volné a vázané proměnné

Význam formule (**pravdivostní hodnota**) může/nemusí záviset na proměnných v ní: $x \leq 0$ vs. $(\exists x)(x \leq 0)$ vs. $x \leq 0 \vee (\exists x)(x \leq 0)$

- **výskyt x ve φ** : list $\text{Tree}(\varphi)$ označený x [$\vee (Qx)$ nemá výskyt!]
- **vázaný**: součástí podformule začínající (Qx) , jinak **volný**
- x je **volná** ve φ má-li volný výskyt, **vázaná** má-li vázaný výskyt
- zápis $\varphi(x_1, \dots, x_n)$ znamená, že mezi x_1, \dots, x_n jsou všechny volné proměnné ve formuli φ

Proměnná může být **volná i vázaná**, např.:

$$\varphi = (\forall x)(\exists y)(x \leq y) \vee x \leq z$$

- první výskyt x je vázaný a druhý volný (nakreslete si strom!)
- y je vázaná a z je volná, můžeme tedy psát $\varphi(x, z)$

Otevřené a uzavřené formule

otevřená formule: nemá žádný kvantifikátor

uzavřená formule (sentence): nemá žádnou volnou proměnnou

- $x + y \leq 0$ je otevřená formule
- $(\forall x)(\forall y)(x + y \leq 0)$ je uzavřená formule neboli sentence
- $(\forall x)(x + y \leq 0)$ není ani otevřená, ani uzavřená
- $(0 + 1 = 1) \wedge (1 + 1 = 0)$ je otevřená i uzavřená
- atomické formule je otevřená, otevřené formule jsou kombinace atomických pomocí logických spojek
- je-li formule otevřená i uzavřená potom nemá žádné proměnné (všechny termy v ní jsou konstantní)
- formule bez vázané proměnné není nutně otevřená! $(\forall x)0 = 1$

Uvidíme, že **pravdivostní hodnota** závisí jen na ohodnocení volných proměnných; **sentence** mají ve struktuře pravdiv. hodnotu 0 nebo 1

Instance a varianty: neformálně

- proměnná může hrát různé 'role' ('lokální' vs. 'globální')
- **instance**: 'dosazení' do 'globální' proměnné (lépe 'nahrazení' proměnné nějakým termem, který ji počítá, čistě syntaktické!)
- **varianta**: 'přejmenování' 'lokální' proměnné

$$P(x) \wedge (\forall x)(Q(x) \wedge (\exists x)R(x))$$

- první výskyt x je volný, 2. je vázaný ($\forall x$), 3. je vázaný ($\exists x$)
- pokud **substituujeme** za proměnnou x term $t = 1 + 1$, dostáváme **instanci** formule φ , kterou označíme $\varphi(x/t)$:

$$P(1 + 1) \wedge (\forall x)(Q(x) \wedge (\exists x)R(x))$$

- přejmenujeme-li kvantifikátory, získáme **variantu** formule φ :

$$P(x) \wedge (\forall y)(Q(y) \wedge (\exists z)R(z))$$

Kdy a jak to lze, aby instance byla **důsledek** a varianta **ekvivalentní**?

Substituujeme-li do φ za x term t , chceme aby výsledná formule 'říkala o t totéž, co φ o x '. Např. $\varphi(x) = (\exists y)(x + y = 1)$

- říká o x , že 'existuje $1 - x$ '
- term $t = 1$ lze: $\varphi(x/t) = (\exists y)(1 + y = 1)$ říká 'existuje $1 - 1$ '
- term $t = y$ nelze: $(\exists y)(y + y = 1)$ říká '1 je dělitelné 2'

problém: obsahuje y , po nahrazení bude nově vázané $(\exists y)$

Term t je **substituovatelný** za proměnnou x ve formuli φ , pokud po simultánním nahrazení všech volných výskytů x za t nevznikne žádný vázaný výskyt proměnné z t . Potom je vzniklá formule **instance** φ vzniklá substitucí t za x , $\varphi(x/t)$.

- t **není** substituovatelný za x do φ , právě když x má volný výskyt v nějaké podformuli φ tvaru $(Qy)\psi$ a y se vyskytuje v t
- speciálně: konstantní termy jsou vždy substituovatelné

Substituovat t můžeme vždy do **varianty** φ , ve které přejmenujeme všechny kvantifikované proměnné na nové (které nejsou v t ani φ)

Má-li formule φ podformuli tvaru $(Qx)\psi$ a je-li y proměnná, že

- (i) y je substituovatelná za x do ψ , a
- (ii) y nemá volný výskyt v ψ .

Varianta φ vznikne nahrazením $(Qx)\psi$ formulí $(Qy)\psi(x/y)$, říkáme tak i výsledku postupné variace ve více podformulích.

Mějme $\varphi = (\exists x)(\forall y)(x \leq y)$:

- $(\exists u)(\forall v)(u \leq v)$ je varianta φ
- $(\exists y)(\forall y)(y \leq y)$ není varianta kvůli (i): y není substituovatelná za x do $\psi = (\forall y)(y \leq y)$
- $(\exists x)(\forall x)(x \leq x)$ není varianta kvůli (ii): x má volný výskyt v $\psi = (x \leq y)$