

Korektnosť prvorádových tabiel.

Explicitné definície. Unifikácia

10. prednáška

Logika pre informatikov a Úvod do matematickej logiky

Ján Klúka, Ján Mazák, Jozef Šiška

Letný semester 2024/2025

Univerzita Komenského v Bratislave

Fakulta matematiky, fyziky a informatiky

Korektnosť tablového kalkulu
pre logiku prvého rádu

Vlastnosti ohodnotení a substitúcie

Korektnosť tabiel

Ďalšie korektné pravidlá

Explicitné definície

Unifikácia termov

Korektnosť tablového kalkulu pre logiku prvého rádu

Korektnosť tablového kalkulu pre logiku prvého rádu

Vlastnosti ohodnotení a substitúcie

Tvrdenie 13.1

Nech \mathcal{M} je štruktúra pre \mathcal{L} , nech e_1 a e_2 sú ohodnotenia, nech t je term, A je formula a S je množina formúl jazyka \mathcal{L} .

- Ak sa ohodnotenia e_1 a e_2 zhodujú na (voľných) premenných termu t (teda $e_1(x) = e_2(x)$ pre každú $x \in \text{free}(t)$), tak $t^{\mathcal{M}}[e_1] = t^{\mathcal{M}}[e_2]$.
- Ak sa ohodnotenia e_1 a e_2 zhodujú na voľných premenných formuly X , tak $\mathcal{M} \models A[e_1]$ vtt $\mathcal{M} \models A[e_2]$.
- Ak sa ohodnotenia e_1 a e_2 zhodujú na voľných premenných všetkých formúl z S , tak $\mathcal{M} \models S[e_1]$ vtt $\mathcal{M} \models S[e_2]$.

Substitúcia a hodnota termu

Ako súvisí **hodnota** termu po substitúcii s **hodnotou** termu, do ktorého sa substituuje?

Príklad 13.2

Zoberme štruktúru $\mathcal{M} = (D, i)$, kde

$$D = \{1, 2, 3, 4, 5\},$$

$$i(c) = 3, \quad i(d) = 4$$

$$i(f) = \{1 \mapsto 2, 2 \mapsto 5, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 5\}$$

Nech $e = \{x \mapsto 3, y \mapsto 4\}$.

$$\begin{aligned} ((f(x))\{x \mapsto f(y)\})^{\mathcal{M}}[e] &= (f(f(y)))^{\mathcal{M}}[e] \\ &= i(f)(i(f)(4)) = i(f)(1) = 2 \\ &= (f(x))^{\mathcal{M}}[e(x/1)] \\ &= (f(x))^{\mathcal{M}}[e(x / (f(y))^{\mathcal{M}}[e])] \end{aligned}$$

Substitúcia vs. hodnota termu a splnenie formuly

Hodnota termu $t\sigma$ /splnenie formuly $A\sigma$ po substitúcii

$\sigma = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ pri ohodnotení e

sa rovná hodnote termu t /splneniu formuly A pri ohodnotení e' , ktoré

- každej substituovanej premennej x_i
priradí hodnotu za ňu substituovaného termu t_i pri ohodnotení e ,
- ostatným premenným priradí rovnaké hodnoty ako e .

Tvrdenie 13.3

Nech \mathcal{M} je štruktúra pre jazyk \mathcal{L} a e je ohodnotenie ind. premenných a nech $\sigma = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ je substitúcia.

- *Nech t je term jazyka \mathcal{L} . Potom*
$$(t\sigma)^{\mathcal{M}}[e] = t^{\mathcal{M}}[e(x_1/t_1^{\mathcal{M}}[e]) \cdots (x_n/t_n^{\mathcal{M}}[e])].$$
- *Nech A je formula jazyka \mathcal{L} a σ je aplikovateľná na A . Potom*
$$\mathcal{M} \models A\sigma[e] \text{ vtt } \mathcal{M} \models A[e(x_1/t_1^{\mathcal{M}}[e]) \cdots (x_n/t_n^{\mathcal{M}}[e])].$$

Korektnosť tablového kalkulu pre logiku prvého rádu

Korektnosť tabiel

Tvrdenie 13.4

Nech S^+ je množina označených formúl v jazyku \mathcal{L} , nech x a y sú premenné, nech s, t sú termy, nech $\alpha, \beta, \gamma, \delta$ sú ozn. formuly príslušného typu, A je ozn. formula.

- Ak $\alpha \in S^+$, tak S^+ je splniteľná vtt $S^+ \cup \{\alpha_1, \alpha_2\}$ je splniteľná.
- Ak $\beta \in S^+$, tak S^+ je splniteľná vtt $S^+ \cup \{\beta_1\}$ je splniteľná **alebo** $S^+ \cup \{\beta_2\}$ je splniteľná.
- Ak $\gamma(x) \in S^+$ a t je term substituovateľný za x v $\gamma_1(x)$, tak S^+ je splniteľná vtt $S^+ \cup \{\gamma_1(t)\}$ je splniteľná.
- Ak $\delta(x) \in S^+$, y je substituovateľná za x v $\delta_1(x)$ a y nemá voľný výskyt v S^+ , tak S^+ je splniteľná vtt $S^+ \cup \{\delta_1(y)\}$ je splniteľná.
- S^+ je splniteľná vtt $S^+ \cup \{\mathbf{T} t \doteq t\}$ je splniteľná.
- Ak $\{\mathbf{T} s \doteq t, A^+\{x \mapsto s\}\} \subseteq S^+$, s a t sú substituovateľné za x v A^+ , tak S^+ je splniteľná vtt $S^+ \cup \{A^+\{x \mapsto t\}\}$ je splniteľná.

Dôkaz (čiastočný, pre pravidlo δ v smere \Rightarrow).

Zoberme ľubovoľné S^+ , x , y a $\delta(x)$ spĺňajúce predpoklady tvrdenia.

Nech S^+ je splniteľná,

teda existuje štruktúra $\mathcal{M} = (D, i)$ a ohodnotenie e také, že $\mathcal{M} \models S^+[e]$.

Preto aj $\mathcal{M} \models \delta(x)[e]$.

Podľa tvaru $\delta(x)$ môžu nastať nasledujúce dva prípady:

- Ak $\delta(x) = \mathbf{T} \exists x A$ pre nejakú formulu A , tak podľa def. splnenia ozn. formuly $\mathcal{M} \models \exists x A[e]$ a podľa def. splnenia formuly máme nejakého svedka $m \in D$ takého, že $\mathcal{M} \models A[e(x/m)]$.
Podľa tvr. 13.3 potom $\mathcal{M} \models A\{x \mapsto y\}[e(x/m)(y/m)]$.
Prem. x nie je voľná v $A\{x \mapsto y\}$,
preto podľa tvr. 13.1 $\mathcal{M} \models A\{x \mapsto y\}[e(y/m)]$,
teda $\mathcal{M} \models \mathbf{T} A\{x \mapsto y\}[e(y/m)]$, teda $\mathcal{M} \models \delta_1(y)[e(y/m)]$.

Dôkaz (čiastočný, pre pravidlo δ v smere \Rightarrow , pokračovanie).

- Ak $\delta(x) = \mathbf{F} \forall x A$ pre nejakú formulu A , tak podľa def. splnenia ozn. formuly $\mathcal{M} \not\models \forall x A[e]$ a podľa def. splnenia formuly neplatí, že $\mathcal{M} \models A[e(x/m)]$ pre každé $m \in D$.

Preto máme nejaký *kontrapríklad* $m \in D$ taký, že $\mathcal{M} \not\models A[e(x/m)]$.

Podľa tvr. 13.3 potom $\mathcal{M} \not\models A\{x \mapsto y\}[e(x/m)(y/m)]$.

Prem. x nie je voľná v $A\{x \mapsto y\}$,

preto podľa tvr. 13.1 $\mathcal{M} \not\models A\{x \mapsto y\}[e(y/m)]$,

teda $\mathcal{M} \models \mathbf{F} A\{x \mapsto y\}[e(y/m)]$, čiže $\mathcal{M} \models \delta_1(y)[e(y/m)]$.

Navyše y nie je voľná v žiadnej formule z S^+ , preto $\mathcal{M} \models S^+[e(y/m)]$.

Teda $\mathcal{M} \models (S^+ \cup \{\delta_1(y)\})[e(y/m)]$.

Preto je $S^+ \cup \{\delta_1(y)\}$ splniteľná. □

Korektnosť — pravdivosť priameho rozšírenia tabla

Vetva sa správa ako konjunkcia svojich označených formúl — všetky musia byť naraz splnené.

Tablo sa správa ako disjunkcia vetiev — niektorá musí byť splnená.

Definícia 13.5

Nech S^+ je množina označených formúl v jazyku \mathcal{L} , nech \mathcal{T} je tablo pre S^+ , nech π je vetva tabla \mathcal{T} . Nech \mathcal{M} je štruktúra pre \mathcal{L} a e je ohodnotenie individuových premenných. Potom:

- *štruktúra \mathcal{M} spĺňa vetvu π pri e* vtt \mathcal{M} spĺňa **všetky** označené formuly vyskytujúce sa na vetve π pri e .
- *štruktúra \mathcal{M} spĺňa tablo \mathcal{T} pri e* vtt \mathcal{M} spĺňa **niektorú** vetvu v table \mathcal{T} pri e .

Pomocné tvrdenia pre korektnosť prvorádových tabiel

Lema 13.6 (K1)

Nech S^+ je množina ozn. formúl v jazyku \mathcal{L} , nech \mathcal{T} je tablo pre S^+ .
Nech \mathcal{M} je štruktúra pre \mathcal{L} a e je ohodnotenie ind. premenných.
Ak \mathcal{T} a S^+ sú splnené štruktúrou \mathcal{M} pri e ,
tak aj každé priame rozšírenie \mathcal{T} a S^+ sú splnené štruktúrou \mathcal{M} pri nejakom ohodnotení e' .

Definícia 13.7

Nech \mathcal{T} je tablo pre nejakú množinu označených formúl.
Tablo \mathcal{T} je **splniteľné** vtt existuje štruktúra, ktorá splňa \mathcal{T} pri nejakom ohodnotení individuových premenných.

Lema 13.8 (K2)

Nech S^+ je množina ozn. formúl v jazyku \mathcal{L} , nech \mathcal{T} je tablo pre S^+ .
Ak S^+ je splniteľná, tak aj \mathcal{T} je splniteľné.

Korektnosť prvorádových tabiel

Otvorené a uzavreté vetvy a tablá sú definované rovnako ako pri tabľách pre výrokovú logiku.

Veta 13.9 (Korektnosť tablového kalkulu)

Nech S^+ je množina označených formúl.

Ak existuje uzavreté tablo \mathcal{T} pre S^+ , tak je množina S^+ nesplniteľná.

Dôkaz (sporom).

Nech S^+ je množina označených formúl.

Nech existuje uzavreté tablo \mathcal{T} pre S^+ , ale S^+ je splniteľná. Pretože \mathcal{T} je uzavreté, pre každú jeho vetvu π existuje formula X taká, že $\mathbf{T}X$ a $\mathbf{F}X$ sa vyskytuje na π , a teda π je nesplniteľná. Preto \mathcal{T} je nesplniteľné. To je v spore s lemov K2, podľa ktorej je \mathcal{T} splniteľné, pretože S^+ je splniteľná. □

Korektnosť tablového kalkulu pre logiku prvého rádu

Ďalšie korektné pravidlá

Tvrdenie 13.10

Nasledujúce pravidlá sú korektné:

$$\begin{array}{c} \gamma^* \quad \frac{\mathbf{T} \forall x_1 \dots \forall x_n A}{\mathbf{T} A\{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}} \quad \frac{\mathbf{F} \exists x_1 \dots \exists x_n A}{\mathbf{F} A\{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}} \\[2ex] \delta^* \quad \frac{\mathbf{F} \forall x_1 \dots \forall x_n A}{\mathbf{F} A\{x_1 \mapsto y_1, \dots, x_n \mapsto y_n\}} \quad \frac{\mathbf{T} \exists x_1 \dots \exists x_n A}{\mathbf{T} A\{x_1 \mapsto y_1, \dots, x_n \mapsto y_n\}} \end{array}$$

kde A je formula, x_1, \dots, x_n sú premenné, t_1, \dots, t_n sú termy,
 y_1, \dots, y_n sú navzájom rôzne premenné,
ktoré sa **nevyskytujú voľne** vo vetve, v liste ktorej je pravidlo použité,
pričom pre každé $i \in \{1, \dots, n\}$ je term t_i **substituovateľný** za x_i v A
a premenná y_i je **substituovateľná** za x_i v A .

Tvrdenie 13.11

Nasledujúce pravidlá sú korektné:

$$ESTT \quad \frac{\mathbf{T}(A_1 \leftrightarrow A_2) \quad \mathbf{T} A_i}{\mathbf{T} A_{3-i}}$$

$$ESTF \quad \frac{\mathbf{T}(A_1 \leftrightarrow A_2) \quad \mathbf{F} A_i}{\mathbf{F} A_{3-i}}$$

$$ESFT \quad \frac{\mathbf{F}(A_1 \leftrightarrow A_2) \quad \mathbf{T} A_i}{\mathbf{F} A_{3-i}}$$

$$ESFF \quad \frac{\mathbf{F}(A_1 \leftrightarrow A_2) \quad \mathbf{F} A_i}{\mathbf{T} A_{3-i}}$$

kde A_1 a A_2 sú formuly, $i \in \{1, 2\}$.

Všimnite si: $3 - 1 = 2$ a $3 - 2 = 1$.

Explicitné definície

V mnohých doménach sú zaujímavé komplikovanejšie
kombinácie základných vlastností alebo vzťahov:

- *x má spoločného rodiča s y, ale x je rôzne od y*
 $\exists z(\text{rodič}(z, x) \wedge \text{rodič}(z, y)) \wedge \neg x \doteq y$
- *x je živočích, ktorý konzumuje iba rastliny*
 $\text{živočích}(x) \wedge \forall y(\text{konzumuje}(x, y) \rightarrow \text{rastlina}(y))$

Často sa vyskytujúce kombinácie vzťahov a vlastností je výhodné:

- **pomenovať**
- a jasne **vyjadriť význam** nového mena
pomocou doteraz známych vlastností a vzťahov,

teda **zadefinovať pojem**.

Definícia je tvrdenie, ktoré vyjadruje význam pojmu.

Explicitná definícia (najčastejší druh definície) je
ekvivalencia medzi pojmom a opisom jeho významu,
v ktorom sa definovaný pojem sám nevyskytuje.

Príklad 14.1

- Objekt x je **súrodencom** objektu y práve vtedy,
keď x nie je y a x má spoločného rodiča s y .

$$\forall x \forall y (\text{súrodenec}(x, y) \leftrightarrow \\ (x \neq y \wedge \exists z (\text{rodič}(z, x) \wedge \text{rodič}(z, y))))$$

- Objekt x je **bylinožravec** vtedy a len vtedy,
keď x je živočích, ktorý konzumuje iba rastliny.

$$\forall x (\text{bylinožravec}(x) \leftrightarrow \\ (\text{živočích}(x) \wedge \forall y (\text{konzumuje}(x, y) \rightarrow \text{rastlina}(y))))$$

Explicitná def. a nutná a postačujúca podmienka

Všimnite si:

Definícia pojmu *súrodenec* vyjadruje **nutnú aj postačujúcu** podmienku toho, aby medzi dvoma objektmi bol súrodenecký vzťah.

- Pre každú dvojicu objektov x a y , ktoré označíme za súrodencov, **musí** existovať ich spoločný rodič a musia byť navzájom rôzne.
- ← Každé dva navzájom rôzne objekty x a y , ktoré majú spoločného rodiča, **musia** byť súrodenci.

Podobne pre iné definície.

Použitie pojmov

Využitím definovaného pojmu

- skracujeme tvrdenia: *Škrečky sú bylinožravce.*
 $\forall x(\text{škrekok}(x) \rightarrow \text{bylinožravec}(x))$
- jednoduchšie definujeme ďalšie pojmy:
*Objekt x je **sestrou** objektu y práve vtedy, keď x je žena, ktorá je súrodencom y .*
 $\forall x \forall y(\text{sestra}(x, y) \leftrightarrow (\text{žena}(x) \wedge \text{súrodenec}(x, y)))$
- potenciálne skracujeme dôkazy (napr. nájdeme spor vyjadrený novým pojmom a nemusíme analyzovať celý podstrom zodpovedajúci rozvinutiu pojmu cez jeho definíciu)

Vyskúšajte si 14.1

Zadefinujte pojem *teta* (chápaný ako vzťah dvoch ľudí)
neformálne (v slovenčine)
aj formálne (formulou logiky prvého rádu).

Podmienené definície

Niekedy má pojem význam iba pre niektoré druhy objektov,
alebo má ten istý pojem rôzne významy pre rôzne druhy objektov.

Vtedy môžeme definície **podmieniť** druhmi:

- *Študent absolvuje predmet vtt*
je z neho hodnotený inou známkou ako Fx.
$$\forall x \forall y (\text{študent}(x) \wedge \text{predmet}(y) \rightarrow$$
$$(\text{absolvuje}(x, y) \leftrightarrow$$
$$\exists z (\text{hodnotený}(x, y) \doteq z \wedge \text{známka}(z) \wedge z \neq Fx)))$$
- *Študent absolvuje študijný program vtt*
absolvuje každý jeho povinný predmet.
$$\forall x \forall y (\text{študent}(x) \wedge \text{št_prog}(y) \rightarrow$$
$$(\text{absolvuje}(x, y) \leftrightarrow$$
$$\forall z (\text{pov_predmet_prog}(z, y) \rightarrow \text{absolvuje}(x, z))))$$

Definícia 14.2

Nech \mathcal{L} a \mathcal{L}_1 sú jazyky logiky prvého rádu.

Jazyk \mathcal{L}_1 je **rozšírením** jazyka \mathcal{L} vtt $\mathcal{V}_{\mathcal{L}_1} = \mathcal{V}_{\mathcal{L}}$, $\mathcal{C}_{\mathcal{L}} \subseteq \mathcal{C}_{\mathcal{L}_1}$,

$\mathcal{P}_{\mathcal{L}} \subseteq \mathcal{P}_{\mathcal{L}_1}$, $\mathcal{F}_{\mathcal{L}} \subseteq \mathcal{F}_{\mathcal{L}_1}$.

Definícia 14.3

Nech \mathcal{L} je jazyk logiky prvého rádu, T je teória v jazyku \mathcal{L} , a \mathcal{L}_P je rozšírenie jazyka o predikátový symbol P je s aritou n , ktorý sa nevyskytuje v \mathcal{L} . Teóriu v jazyku \mathcal{L}_P

$$T \cup \{\forall x_1 \dots \forall x_n (P(x_1, \dots, x_n) \leftrightarrow A)\},$$

kde A je formula, v ktorej sa nevyskytuje P , nazývame **rozšírením teórie T explicitnou definíciou** $\forall x_1 \dots \forall x_n (P(x_1, \dots, x_n) \leftrightarrow A)$ predikátového symbolu P .

Jednoznačnosť interpretácie definovaného predikátu

Význam explicitne definovaného predikátu je jednoznačne určený.

Príklad 14.4

Majme nejakú teóriu T v jazyku \mathcal{L} s $\mathcal{P}_{\mathcal{L}} = \{\text{rodič}^2\}$.

Rozšírme T o $X = \forall x \forall y (\text{súrodenec}(x, y) \leftrightarrow (x \neq y \wedge \exists z (\text{rodič}(z, x) \wedge \text{rodič}(z, y))))$.

Nech $\mathcal{M} = (\{\mathbf{i}_I, \mathbf{c}_J, \mathbf{i}_K, \mathbf{i}_L, \mathbf{i}_M, \mathbf{i}_N, \mathbf{i}_O\}, i)$ je model T , kde

$$i(\text{rodič}) = \{(\mathbf{i}_I, \mathbf{i}_M), (\mathbf{i}_L, \mathbf{i}_M), (\mathbf{i}_I, \mathbf{i}_N), (\mathbf{i}_O, \mathbf{i}_N), (\mathbf{i}_M, \mathbf{i}_K), (\mathbf{i}_M, \mathbf{c}_J)\}$$

Potom sa \mathcal{M} dá **jednoznačne** rozšíriť na model $T \cup \{X\}$:

$\mathcal{M}_1 = (\{\mathbf{i}_I, \mathbf{c}_J, \mathbf{i}_K, \mathbf{i}_L, \mathbf{i}_M, \mathbf{i}_N, \mathbf{i}_O\}, i_1)$, $i_1(\text{rodič}) = i(\text{rodič})$,

$$i_1(\text{súrodenec}) =$$

Jednoznačnosť interpretácie definovaného predikátu

Význam explicitne definovaného predikátu je jednoznačne určený.

Príklad 14.4

Majme nejakú teóriu T v jazyku \mathcal{L} s $\mathcal{P}_{\mathcal{L}} = \{\text{rodič}^2\}$.

Rozšírme T o $X = \forall x \forall y (\text{súrodenec}(x, y) \leftrightarrow (x \neq y \wedge \exists z (\text{rodič}(z, x) \wedge \text{rodič}(z, y))))$.

Nech $\mathcal{M} = (\{\mathbf{i}_I, \mathbf{z}_J, \mathbf{i}_K, \mathbf{i}_L, \mathbf{i}_M, \mathbf{i}_N, \mathbf{i}_O\}, i)$ je model T , kde

$$i(\text{rodič}) = \{(\mathbf{i}_I, \mathbf{i}_M), (\mathbf{i}_L, \mathbf{i}_M), (\mathbf{i}_I, \mathbf{i}_N), (\mathbf{i}_O, \mathbf{i}_N), (\mathbf{i}_M, \mathbf{i}_K), (\mathbf{i}_M, \mathbf{z}_J)\}$$

Potom sa \mathcal{M} dá **jednoznačne** rozšíriť na model $T \cup \{X\}$:













$\mathcal{M}_1 = (\{\mathbf{i}_I, \mathbf{z}_J, \mathbf{i}_K, \mathbf{i}_L, \mathbf{i}_M, \mathbf{i}_N, \mathbf{i}_O\}, i_1)$, $i_1(\text{rodič}) = i(\text{rodič})$,

$$i_1(\text{súrodenec}) = \{(\mathbf{i}_M, \mathbf{i}_N), (\mathbf{i}_N, \mathbf{i}_M), (\mathbf{i}_K, \mathbf{z}_J), (\mathbf{z}_J, \mathbf{i}_K)\}$$

Definícia ako dopyt









Explicitne definovaný predikát sa správa ako **dopyt** alebo **pohľad** nad ostatnými predikátmi.

Príklad 14.5

rodič	
r	d
 _I	 _M
 _L	 _M
 _I	 _N
 _O	 _N
 _M	 _K
 _M	 _J

```
CREATE VIEW súrodenec AS
SELECT r1.d AS d1, r2.d AS d2
FROM rodič AS r1
JOIN rodič AS r2 ON r1.r = r2.r
WHERE r1.d <> r2.d
```

$$\forall x \forall y$$
$$(\text{súrodenec}(x, y) \leftrightarrow$$
$$(x \neq y \wedge$$
$$\exists z(\text{rodič}(z, x) \wedge \text{rodič}(z, y))))$$

súrodenec	
d1	d2
 _M	 _N
 _N	 _M
 _K	 _J
 _J	 _K

Definícia 14.6

Nech \mathcal{L}_2 je rozšírenie jazyka \mathcal{L}_1 . Nech $\mathcal{M}_1 = (D_1, i_1)$ je štruktúra pre \mathcal{L}_1 a $\mathcal{M}_2 = (D_2, i_2)$ je štruktúra pre \mathcal{L}_2 .

Potom \mathcal{M}_2 je **rozšírením** (**expanziou**) \mathcal{M}_1 vtt $D_2 = D_1$ a $i_2(s) = i_1(s)$ pre každý mimologický symbol s jazyka \mathcal{L}_1 .

Tvrdenie 14.7

Nech

- T je teória v jazyku \mathcal{L} ,
- T' je rozšírenie T explicitnou definíciou predikátového symbolu.

Potom

- pre každý model teórie T existuje práve jedno jeho rozšírenie, ktoré je modelom teórie T' ,
- každý model teórie T' je rozšírením práve jedného modelu teórie T .

Konzervatívnosť spočíva v tom, že pridávaním nepokazíme význam existujúcich vecí.

Tvrdenie 14.8

Nech T je teória v jazyku \mathcal{L} a T' je rozšírenie T explicitnou definíciou nejakého predikátového symbolu.

Nech X je uzavretá formula jazyka \mathcal{L} .

Potom $T \models X$ vtt $T' \models X$.

Kontextová definícia funkčného symbolu

Nech $A(x, y)$ je formula s voľnými premennými x, y . Táto formula popisuje akýsi vzťah medzi x a y (a mohli by sme pridať predikát, ktorým tento vzťah pomenujeme). Ak tento vzťah je funkcia, t. j.

$$T \vdash \forall x \exists y (A(x, y) \wedge \forall y_2 (A(x, y_2) \rightarrow y_2 \doteq y)),$$

môžeme jazyk rozšíriť o nový funkčný symbol f^1 a teóriu T o *kontextovú definíciu funkcie f* :

$$\forall x \forall y (f(x) \doteq y \leftrightarrow A(x, y))$$

Príklady:

- Do jazyka teórie grúp pridáme unárny funkčný symbol $^{-1}$ označujúci inverzný prvok (v grupe existuje práve jeden).
- Do teórie popisujúcej rodinné vzťahy pridáme funkčný symbol na označenie matky (z teórie však musí vyplývať, že matka je len jedna).

Kontextová definícia individuovej konštanty

Nech $A(y)$ je formula s voľnou premennou y . Táto formula popisuje akúsi vlastnosť prvku domény (a mohli by sme pridať predikát, ktorým túto vlastnosť pomenujeme). Ak je takýto prvok jediný, t. j.

$$T \vdash \exists y(A(y) \wedge \forall y_2(A(y_2) \rightarrow y_2 \doteq y)),$$

môžeme jazyk rozšíriť o novú individuovú konštantu a a teóriu T

o *kontextovú definíciu konštanty a*

$$\forall x(a \doteq x \leftrightarrow A(x))$$

Napr. pre jazyk popisujúci (matematické) polia pridáme symboly 0 a 1;
do jazyka teórie množín pridáme konštantu pre prázdnu množinu.

Pridávanie mimologických symbolov

Rozširovanie existujúceho jazyka (resp. teórie) o nové predikáty, konštanty a funkčné symboly explicitnými/kontextovými definíciami **nezvyšuje** vyjadrovaciu silu jazyka:

- nové symboly možno vnímať ako pohodlné skratky,
- nevieme však dokázať nič viac, ako bez nich.

Toto zachytáva tvrdenie 14.8; podobné tvrdenia možno sformulovať aj pre pridané konštanty a funkčné symboly.

Niekedy môže byť výhodnejšie ako definičné axiómy funkcií a konštánt použiť:

$$\forall x A(x, f(x))$$
$$A(a)$$

Dokazovanie s explicitnými definíciami a rovnosťou

Využime nové pravidlá na dôkaz vyplývania z teórie s definíciou:

Príklad 14.9

Dokážme tablom, že $T \models X$ pre

$$\begin{aligned} T = & \{ \forall x \forall y (\text{študent}(x) \wedge \text{predmet}(y) \rightarrow \\ & \quad (\text{absolvuje}(x, y) \leftrightarrow \\ & \quad \exists z (\text{známka}(z) \wedge \text{hodnotený}(x, y) \doteq z \wedge z \neq Fx))), \\ & \forall x \forall y (\text{študent}(x) \wedge \text{št_prog}(y) \rightarrow \\ & \quad (\text{absolvuje}(x, y) \leftrightarrow \\ & \quad \forall z (\text{pov_predmet_prog}(z, y) \rightarrow \text{absolvuje}(x, z)))), \\ & \forall x (\text{št_prog}(x) \rightarrow \exists y \text{pov_predmet_prog}(z, x)), \\ & \forall x (\exists y \text{pov_predmet_prog}(x, y) \rightarrow \text{predmet}(x)) \} \\ X = & \forall x \forall y (\text{študent}(x) \wedge \text{št_prog}(y) \wedge \text{absolvuje}(x, y) \rightarrow \\ & \quad \exists y \text{hodnotený}(x, y) \neq Fx) \end{aligned}$$

Unifikácia termov

Dosádzanie termov za premenné

Pri kvantifikovaných formulách s funkčnými symbolmi môže byť ťažké povedať, aké termy dosádzať za všeobecne kvantifikované premenné.

Čo možno usúdiť z nasledujúcich dvoch formúl?

$$\begin{aligned}\forall y \quad & P(f(y), y) \\ \forall x (\neg P(x, d) \vee R(x))\end{aligned}$$

Ak by sme vhodným dosadením termov dosiahli totožnosť $f(y)$ s x a y s d , možno usúdiť $R(x)$.

Dosádzanie termov za premenné

$$\forall y \ P(f(y), y)$$
$$\forall x (\neg P(x, d) \vee R(x))$$

Dosadenie popisujeme pomocou substitúcie. V našom prípade zjavne za y musíme substituovať d a za x ...

Dosádzanie termov za premenné

$$\begin{aligned}\forall y \quad & P(f(y), y) \\ \forall x (\neg P(x, d) \vee R(x))\end{aligned}$$

Dosadenie popisujeme pomocou substitúcie. V našom prípade zjavne za y musíme substituovať d a za x ...

$$\sigma = \{x \mapsto f(d), y \mapsto d\}$$

Po substitúcii σ majú komplementárne literály rovnaké argumenty predikátu (preto σ nazývame *unifikátor*):

$$\begin{aligned}P(f(y), y)\sigma &= P(f(d), d) \\ \neg P(x, d)\sigma &= \neg P(f(d), d)\end{aligned}$$

Jedným z dôsledkov uvedených dvoch formúl je teda $R(f(d))$.
(Aké iné dôsledky z uvedených formúl vyplývajú?)

Definícia 15.1

Nech A, B sú postupnosti symbolov, σ je substitúcia.

Substitúcia σ je **unifikátorom** A a B vtt $A\sigma = B\sigma$.

Príklad 15.2

- $A_1 = R(\text{filantrop}, y), B_1 = R(x, d),$
 $\sigma_1 = \{x \mapsto \text{filantrop}, y \mapsto d\}$
- $A_2 = R(\text{nk}(y), y), B_2 = R(x, d),$

Definícia 15.1

Nech A, B sú postupnosti symbolov, σ je substitúcia.

Substitúcia σ je **unifikátorom** A a B vtt $A\sigma = B\sigma$.

Príklad 15.2

- $A_1 = R(\text{filantrop}, y), B_1 = R(x, d),$
 $\sigma_1 = \{x \mapsto \text{filantrop}, y \mapsto d\}$
- $A_2 = R(\text{nk}(y), y), B_2 = R(x, d),$
 $\sigma_2 = \{x \mapsto \text{nk}(d), y \mapsto d\}$
- $A_3 = R(\text{nk}(y), y), B_3 = R(e, x),$

Definícia 15.1

Nech A, B sú postupnosti symbolov, σ je substitúcia.

Substitúcia σ je **unifikátorom** A a B vtt $A\sigma = B\sigma$.

Príklad 15.2

- $A_1 = R(\text{filantrop}, y), B_1 = R(x, d),$
 $\sigma_1 = \{x \mapsto \text{filantrop}, y \mapsto d\}$
- $A_2 = R(\text{nk}(y), y), B_2 = R(x, d),$
 $\sigma_2 = \{x \mapsto \text{nk}(d), y \mapsto d\}$
- $A_3 = R(\text{nk}(y), y), B_3 = R(e, x), \quad \sigma_3 = ???$ **neexistuje!**
- $A_4 = R(\text{nk}(y), y), B_4 = R(x, x),$

Definícia 15.1

Nech A, B sú postupnosti symbolov, σ je substitúcia.

Substitúcia σ je **unifikátorom** A a B vtt $A\sigma = B\sigma$.

Príklad 15.2

- $A_1 = R(\text{filantrop}, y), B_1 = R(x, d),$
 $\sigma_1 = \{x \mapsto \text{filantrop}, y \mapsto d\}$
- $A_2 = R(\text{nk}(y), y), B_2 = R(x, d),$
 $\sigma_2 = \{x \mapsto \text{nk}(d), y \mapsto d\}$
- $A_3 = R(\text{nk}(y), y), B_3 = R(e, x), \quad \sigma_3 = ??? \text{ neexistuje!}$
- $A_4 = R(\text{nk}(y), y), B_4 = R(x, x), \quad \sigma_4 = ??? \text{ neexistuje!}$
- $A_5 = R(f(y)), B_5 = R(x),$

Definícia 15.1

Nech A, B sú postupnosti symbolov, σ je substitúcia.

Substitúcia σ je **unifikátorom** A a B vtt $A\sigma = B\sigma$.

Príklad 15.2

- $A_1 = R(\text{filantrop}, y), B_1 = R(x, d),$
 $\sigma_1 = \{x \mapsto \text{filantrop}, y \mapsto d\}$
- $A_2 = R(\text{nk}(y), y), B_2 = R(x, d),$
 $\sigma_2 = \{x \mapsto \text{nk}(d), y \mapsto d\}$
- $A_3 = R(\text{nk}(y), y), B_3 = R(e, x), \quad \sigma_3 = ??? \text{ neexistuje!}$
- $A_4 = R(\text{nk}(y), y), B_4 = R(x, x), \quad \sigma_4 = ??? \text{ neexistuje!}$
- $A_5 = R(f(y)), B_5 = R(x),$
 $\sigma_5 = \{x \mapsto f(d), y \mapsto d\} \quad / \quad \{x \mapsto f(f(d)), y \mapsto f(d)\} \quad / \quad \dots$

Definícia 15.3

Nech $\sigma = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ a $\theta = \{y_1 \mapsto s_1, \dots, y_m \mapsto s_m\}$ sú substitúcie.

Zložením (kompozíciou) substitúcií σ a θ je substitúcia

$$\sigma\theta = \{x_1 \mapsto t_1\theta, \dots, x_n \mapsto t_n\theta, y_{i_1} \mapsto s_{i_1}, \dots, y_{i_k} \mapsto s_{i_k}\},$$

kde $\{y_{i_1}, \dots, y_{i_k}\} = \{y_1, \dots, y_m\} \setminus \{x_1, \dots, x_n\}$.

Príklad 15.4

$$\sigma = \{x \mapsto \text{nk}(y), z \mapsto y\}$$

$$\theta = \{y \mapsto d\}$$

$$\sigma\theta = \{x \mapsto \text{nk}(d), \\ z \mapsto d, y \mapsto d\}$$

Je pravda, že pre ľubovoľné substitúcie α, β, γ platí $(\alpha\beta)\gamma = \alpha(\beta\gamma)$?

Definícia 15.5

Nech A, B sú postupnosti symbolov, σ a θ sú substitúcie.

σ je **všeobecnejšia** ako θ vtt existuje subst. γ taká, že $\theta = \sigma\gamma$.

σ je **najvšeobecnejším unifikátorom** A a B vtt

- σ je unifikátorom A a B a zároveň
- pre každý unifikátor θ A a B je σ všeobecnejšia ako θ .

Príklad 15.6

$A = R(nk(x), y), B = R(u, v)$

- $\sigma_1 = \{u \mapsto nk(d), v \mapsto y, x \mapsto d\}$
 $\theta_1 = \{u \mapsto nk(d), v \mapsto Biba, x \mapsto d, y \mapsto Biba\}$
 $\gamma_1 = \{y \mapsto Biba\}$
- $\sigma_2 = \{u \mapsto nk(x), v \mapsto y\}$
 $\theta_2 = \{u \mapsto nk(d), v \mapsto y, x \mapsto d\}$
 $\gamma_2 = \{x \mapsto d\}$

Unifikácia má mnohoraké využitie:

- rezolvencia v prvorádovej logike
- inferencia typov kompilátormi (typy sú vlastne termy)
- niektoré druhy parserov (o. i. pattern matching)
- spracovanie prirodzeného jazyka (Prolog)
- deduktívne databázy
- expertné systémy, automatizované usudzovanie

Ukážeme si základný algoritmus na hľadanie najvšeobecnejšieho unifikátora (z r. 1965).

<http://web.stanford.edu/class/linguist289/robinson65.pdf>

<https://eli.thegreenplace.net/2018/unification/>

<https://github.com/eliben/code-for-blog/blob/master/2018/unif/unifier.py>

Unifikácia: typy

```
class Term:
```

```
    pass
```

```
class Const(Term):
```

```
    """Constant"""
```

```
    def __init__(self, name):
```

```
        self.name = name
```

```
class Var(Term):
```

```
    """Variable"""
```

```
    def __init__(self, name):
```

```
        self.name = name
```

```
class App(Term):
```

```
    """Application of a function symbol"""
```

```
    def __init__(self, fname, args=()):
```

```
        self.fname = fname
```

```
        self.args = args
```

```
Subst = dict[Var: Term]
```

Unifikácia: unify

```
def unify(s: Term, t: Term, sigma: Subst | None) -> Subst | None:
    """Unifies terms s and t, given an initial substitution."""
    if sigma is None:
        return None
    elif s == t:
        return sigma
    elif isinstance(s, Var):
        return unify_variable(s, t, sigma)
    elif isinstance(t, Var):
        return unify_variable(t, s, sigma)
    elif isinstance(s, App) and isinstance(t, App):
        if s.fname != t.fname:
            return None
        else:
            for i in range(len(s.args)):
                sigma = unify(s.args[i], t.args[i], sigma)
            return sigma
    else:
        # includes the case where s, t are different constants
        return None
```

Unifikácia: unify_variable

```
def unify_variable(x: Var, t: Term, sigma: Subst) -> Subst | None:
    """Unifies variable x with term t, using sigma.

    Returns updated sigma or None if impossible.
    """
    if x.name in sigma:
        return unify(sigma[x.name], t, sigma)
    elif isinstance(t, Var) and t.name in sigma:
        return unify(x, sigma[t.name], sigma)
    elif occurs_check(x, t, sigma):
        return None
    else:
        # x is not yet in sigma and can't simplify t. Extend sigma.
        return {**sigma, x.name: t}
```


Unifikácia: occurs_check

```
def occurs_check(v: Var, t: Term, sigma: Subst) -> bool:
    """Does the variable v occur anywhere inside t?

    Variables in t are looked up in sigma and the check is applied
    recursively.
    """
    if v == t:
        return True
    elif isinstance(t, Var) and t.name in sigma:
        return occurs_check(v, sigma[t.name], sigma)
    elif isinstance(t, App):
        return any(occurs_check(v, arg, sigma) for arg in t.args)
    else:
        return False
```

Korektný algoritmus: skončí a dá správny výsledok.

- Vďaka `occurs_check` algoritmus nikdy za premennú nedosadí term, ktorý ju obsahuje.
- Ak sme raz za premennú niečo dosadili, nedosadíme za ňu nič iné, a pri jej unifikovaní vždy použijeme existujúce dosadenie.
- `unify_variable` znižuje počet premenných. (Môžeme si predstaviť, že všetky termy pri každom dosadení prepíšeme už bez premennej, za ktorú sme dosadzovali.)
- `unify` zjednodušuje termy (postupne ubúdajú funkčné symboly).
- Algoritmus je preto konečný a nájde nejaký unifikátor (ak existuje).

Nájdenny unifikátor je najvšeobecnejší kvôli tomu, že algoritmus rozširuje substitúciu len vtedy, keď musí, a najvšeobecnejšie, ako sa dá (nepridáva zbytočné funkčné symboly).

(Toto by si zaslúžilo podrobný dôkaz, o. i. pretože najvšeobecnejší unifikátor nie je celkom jednoznačný – hoci ak ich existuje viac, líšiť sa môžu len označením premenných. Na tomto predmete ho však robiť nebudeme.)

Zaujímavosť: v r. 1991 bola objavená chyba v 7 rôznych seriózných knihách prezentujúcich tento algoritmus¹.

¹<http://norvig.com/unify-bug.pdf>

Názorná predstava, ako unifikácia prebieha: máme sústavu rovností termov, ktorú upravujeme a postupne rozširujeme substitúciu o dosadenia za nové premenné. Povolené úpravy:

- Miesto rovnice, ktorá porovnáva dva rovnaké funkčné symboly s aritou k , zapíš k rovníc pre rovnosť jednotlivých argumentov.
- Ak je na niektorej strane rovnice osamotená premenná, dosad' za ňu term predpísaný rovnicou a prepíš všetky výskyty tejto premennej.
- Zmaž triviálne splnenú rovnicu.

Každá z operácií niečo znižuje (počet funkčných symbolov na ľavej strane, počet premenných, počet rovníc).

Príklad 15.7 (Úspešný beh unifikačného algoritmu)

$$\underline{f(X, h(X), Y, g(Y)) = f(g(Z), W, Z, X)}$$

Príklad 15.7 (Úspešný beh unifikačného algoritmu)

$$f(X, h(X), Y, g(Y)) = f(g(Z), W, Z, X)$$

$$X = g(Z) \quad \{X \mapsto g(Z)\}$$

$$h(X) = W$$

$$Y = Z$$

$$g(Y) = X$$

Príklad 15.7 (Úspešný beh unifikačného algoritmu)

$$f(X, h(X), Y, g(Y)) = f(g(Z), W, Z, X)$$

$$X = g(Z) \quad \{X \mapsto g(Z)\}$$

$$h(X) = W$$

$$Y = Z$$

$$g(Y) = X$$

$$h(g(Z)) = W \quad \{W \mapsto h(g(Z))\}$$

$$Y = Z$$

$$g(Y) = g(Z)$$

Príklad 15.7 (Úspešný beh unifikačného algoritmu)

$$f(X, h(X), Y, g(Y)) = f(g(Z), W, Z, X)$$

$$X = g(Z) \quad \{X \mapsto g(Z)\}$$

$$h(X) = W$$

$$Y = Z$$

$$g(Y) = X$$

$$h(g(Z)) = W \quad \{W \mapsto h(g(Z))\}$$

$$Y = Z$$

$$g(Y) = g(Z)$$

$$Y = Z \quad \{Y \mapsto Z\}$$

$$g(Y) = g(Z)$$

Príklad 15.7 (Úspešný beh unifikačného algoritmu)

$$f(X, h(X), Y, g(Y)) = f(g(Z), W, Z, X)$$

$$X = g(Z) \quad \{X \mapsto g(Z)\}$$

$$h(X) = W$$

$$Y = Z$$

$$g(Y) = X$$

$$h(g(Z)) = W \quad \{W \mapsto h(g(Z))\}$$

$$Y = Z$$

$$g(Y) = g(Z)$$

$$Y = Z \quad \{Y \mapsto Z\}$$

$$g(Y) = g(Z)$$

$$g(Z) = g(Z)$$

Uvedený algoritmus nie je veľmi efektívny (napr. ho `occurs_check` spomaľuje natoľko, že sa v niektorých implementáciách vynecháva²). Existujú teoreticky lepšie algoritmy (zhruba lineárne), ale tie zase na mnohých praktických vstupoch bežia prídlho, preto sa veľmi nepoužívajú.

Poznámka: Pre účely tohto predmetu je najdôležitejšie, aby ste plne rozumeli, o čo pri unifikácii ide, a vedeli nájsť najvšeobecnejší unifikátor v konkrétnom prípade. Úplná znalosť všeobecného algoritmu či zdôvodnenie jeho vlastností sú menej podstatné.

²https://en.wikipedia.org/wiki/Occurs_check