1 Úvod

Při formálním odvozování formulí v logice 1. řádu narážíme na dva zásadní problémy:

1. Máme-li dokázat domněnku C z množiny předpokladů $\mathbf{A}$, jak vést odvozování z množiny $\mathbf{A}$ směrem k nalezení odvození (důkazu) C?

2. V logice 1. řádu lze dosazením za proměnné (substitucí) odvodit z jedné formule nekonečně mnoho důsledků. Například důsledkem formule $p(X)$ jsou všechny formule $p(f(X))$, $p(f(f(X)))$, ..., podle toho, co dosadíme za X.

První problém lze vyřešit snadno převedením problému na důkaz sporem: Snažíme se dokázat spor z množiny $\mathbf{A} \cup \{\neg C\}$.

Druhý problém vyřešil v r. 1965 John Alan Robinson ve své práci *A Machine-Oriented Logic Based on the Resolution Principle*, navazujíc na odvozovací pravidlo *condensed detachment* formulované irským matematikem Carew Meredithem a ještě dřívější práci polského matematika Jana Łukasiewicze.

Základem jeho rezolučního kalkulu pro logiku 1. řádu je tzv. *unifikace*. Její myšlenkou je, že používáme pouze takové substituce, které nám umožní provést odvození s nějakou jinou formulí, a dále že volíme ty nejobecnější možné. Odvozovací pravidlo rezoluce přesně určuje, jaké substituce lze provést.

2 Substituce, unifikátor, nejobecnější unifikátor

Definice 1. *Substituce* je zobrazení, které každé proměnné přiřazuje term. Proměnných, které nejsou zobrazeny samy na sebe, je jen konečně mnoho.

Obvykle značíme řeckými písmeny θ, σ, ... Při explicitním zápisu vypisujeme zobrazení jednotlivých proměnných, např. $\{X \mapsto f(c,Y), Y \mapsto f(g(X), f(Y,Z))\}$. Aplikaci substituce píšeme zprava, tedy například $X\theta$.

Substituce jakožto zobrazení přirozeně rozšiřujeme i na termy a formule:

- Při aplikací substituce na term nahradíme všechny proměnné v daném termu termy předepsanými substitucí.

- Při aplikací substituce na formuli nahradíme všechny *volné* proměnné v dané formuli termy předepsanými substitucí.

  Vylučujeme takové substituce, při kterých bychom některá proměnná v dosazovaném termu měla v daném místě vázaný výskyt. Například nelze provést $((\exists X)p(X,Y))\{Y \mapsto f(X)\}$, protože bychom dosazovali za Y term $f(X)$, ale X je v tom místě vázaná kvantifikátorem. Dostali bychom formuli $(\exists X)p(X, f(X))$, jenž není sémantickým důsledkem původní. V takovém případě nejprve přejmenujeme vázanou proměnnou, a již je korektní provést $((\exists Z)p(Z,Y))\{Y \mapsto f(X)\} \equiv (\exists Z)p(Z, f(X))$.

Poznámka 1. Identická či prázdná substituce zobrazuje každou proměnnou samu na sebe. U substitucí v zápisu popisujeme pouze ty proměnné, které nejsou zobrazeny samy na sebe. Substitucím, které za všechny proměnné dosazují opět jen proměnné, někdy říkáme *přejmenování proměnných*[1].

Příklad 1. Buď $\phi \equiv (\exists Y)p(f(X,Y), f(c,Y))$ a $\theta \equiv \{X \mapsto f(c,Z),\ Y \mapsto d\}$. Pak $\phi\theta \equiv (\exists Y)p(f(f(c,Z),Y), f(c,Y))$.

Definice 2 (Skládání substitucí)**.** Substituce skládáme stejným způsobem jako funkce. Složení dvou substitucí θ a σ je substituce $\theta \circ \sigma$ definovaná vztahem $X_i(\theta \circ \sigma) \equiv (X_i\theta)\sigma$ pro všechny proměnné X_i.

Poznámka 2. Aplikaci substitucí píšeme zprava právě proto, abychom v zápisu $(t\theta)\sigma \equiv t(\theta \circ \sigma)$ zůstalo zachováno pořadí symbolů.

Lemma 1. *Skládání substitucí je asociativní*[2]*. Substituce tedy tvoří vzhledem ke skládání monoid, prázdná substituce je neutrální prvek tohoto monoidu.*

Definice 3. *Unifikátor* termů $t_1, \ldots, t_n$ je každá substituce θ, pro kterou platí $t_1\theta \equiv \ldots \equiv t_n\theta$. Stejně definujeme unifikátor atomických formulí. Pro $n \leq 1$ je tedy unifikátorem každá substituce.

Definice 4. *Nejobecnější unifikátor*[3] termů $t_1, \ldots, t_n$ je takový jejich unifikátor θ, pro který platí, že je-li σ rovněž unifikátorem termů $t_1, \ldots, t_n$, lze σ rozložit na θ a nějakou další substituci ρ, tedy $\sigma \equiv \theta \circ \rho$.

Poznámka 3. Je-li $n \leq 1$ (tedy máme jen jeden nebo žádný term), je jejich nejobecnějším unifikátorem identická substituce.

[1] Pozor, substituce vždy pracují s volnými proměnnými, tedy taková přejmenovávací substituce je to něco jiného, než přejmenování vázaných proměnných.

[2] Ale není komutativní! Např. $\{X \mapsto Y\} \circ \{Y \mapsto Z\} \equiv \{X \mapsto Z, Y \mapsto Z\}$, ale $\{Y \mapsto Z\} \circ \{X \mapsto Y\} \equiv \{X \mapsto Y, Y \mapsto Z\}$

[3] Angl. most general unifier – MGU.

Poznámka 4. Nejobecnějších unifikátorů dané množiny termů může být obecně nekonečně mnoho, ale všechny jsou stejné až na přejemování proměnných. Jsou-li totiž θ_1 a θ_2 dva nejobecnější unifikátory termů $t_1, \ldots, t_n$, pak lze rozložit $\theta_2 \equiv \theta_1 \circ \rho_1$ a rovněž $\theta_1 \equiv \theta_2 \circ \rho_2$ pro nějaké substituce ρ_1 a ρ_2. Tedy $\theta_1 \equiv \theta_1 \circ (\rho_1 \circ \rho_2)$ a $\theta_2 \equiv \theta_2 \circ (\rho_2 \circ \rho_1)$, což je možné jen v případě, že ρ_1 a ρ_2 jsou vzájemně inverzní přejmenování proměnných.

Příklad 2. Substituce $\sigma \equiv \{X \mapsto f(c), Y \mapsto f(c)\}$ je unifikátorem termů $f(X,Y)$ a $f(Y,X)$, ale není nejobecnější. Substituce $\theta \equiv \{X \mapsto Y\}$ je nejobecnějšám unifikátorem. Unikátor σ lze rozložit na $\sigma \equiv \theta \circ \{X \mapsto f(c)\}$.

Příklad 3. Některé další nejobecnější unifikátory termů $f(X,Y)$ a $f(Y,X)$ jsou $\{Y \mapsto X\}$ nebo $\{X \mapsto Z, Y \mapsto Z\}$.

3 Existence nejobecnějšího unifikátoru

Věta 1. *Existuje-li unifikátor termů $t_1, \ldots, t_n$, existuje i jejich nejobecnější unifikátor, a existuje algoritmus pro jeho nalezení.*

Popíšeme algoritmus pro hledání nejobecnějšího unifikátoru.

Algoritmus $\mathcal{U}$ pro nalezení nejobecnějšího unifikátoru.

Vstup: Množina neuspořádaných dvojic termů $\mathbf{S} = \{(t_1, s_1), \ldots, (t_n, s_n)\}$.

Výstup: Buďto nejobecnější unifikátor θ, který unifikuje všechny dané dvojice termů ($t_i\theta \equiv s_i\theta$), a nebo zpráva o tom, že takový unifikátor neexistuje.

Postup: Z množiny $\mathbf{S}$ vybereme vždy libovolnou dvojici termů (je-li neprázdná) a zpracujeme jí podle následujících pravidel.

$$
\begin{aligned}
\mathcal{U}(\{\}) &= \text{identická substituce} \\
\mathcal{U}(\mathbf{S}' \cup \{(t,t)\}) &= \mathcal{U}(\mathbf{S}') \\
\mathcal{U}(\mathbf{S}' \cup \{(f(s_1,\ldots,s_k), f(t_1,\ldots,t_k))\}) &= \mathcal{U}(\mathbf{S}' \cup \{(s_1,t_1),\ldots,(s_k,t_k)\}) \\
\mathcal{U}(\mathbf{S}' \cup \{(f(s_1,\ldots,s_k), g(t_1,\ldots,t_l))\}) & \quad [\text{je-li } k \neq l \text{ nebo } f \not\equiv g] \\
& = \text{nelze unifikovat} \\
\mathcal{U}(\mathbf{S}' \cup \{(X,t)\}) & \quad [X \text{ je volná v } t] \\
& = \text{nelze unifikovat,} \\
& \quad X\theta \text{ má vždy méně symbolů než } t\theta \\
\mathcal{U}(\mathbf{S}' \cup \{(X,t)\}) & \quad [X \text{ není volná v } t] \\
& = \theta \circ \sigma, \text{ kde } \sigma \equiv \{X \mapsto t\} \text{ a } \theta \equiv \mathcal{U}(\mathbf{S}'\sigma) \\
& = \theta \cup \{X \mapsto t\theta\}
\end{aligned}
$$

Lemma 2. *Algoritmus vždy skončí.*

Důkaz. Přiřaďme každému kroku algoritmu trojici (v, f, d) kde v je celkový počet různých proměnných v $\mathbf{S}$, f je celkový počet funkčních symbolů v $\mathbf{S}$ a $d = |\mathbf{S}|$ velikost $\mathbf{S}$. Snadno lze ukázat, že posloupnost těchto trojic je klesající vzhledem k lexikografickému uspořádání, a tedy je kroků jen konečně mnoho. □

Důkaz věty 1. Indukcí podle počtu kroků algoritmu, že algoritmus vrací nejobecnější unifikátor $\mathbf{S}$ právě když $\mathbf{S}$ má unifikátor. □

4 Další aplikace unifikátoru

4.1 Prolog

4.2 Hindley-Milnerův algoritmus pro odvozování typů ve funkcionálních jazycích