Logické systémy

Proč mluvit o logických systémech

* Co je logické uvažování, matematicky?
  Kde jsou jeho limit?
  Jak jej mechanizovat?
* ***Logický systém*** matematicky přesně definuje
  - co je logických výrok, jaký je jeho význam (syntax a sémantika),
  - co je logická argumentace (důkaz).

Odvozovací / důkazový systém

* Definuje, co je logická argumentace, důkaz.
* Skládá se z
  - axiomů, základních zjevný pravd, a
  - odvozovacích pravidel.
* Důkaz je sekvencí použití odvozovacích pravidel, vycházející z axiomů.

Část I

Hilbertovský důkaz ve výrokové logice

Axiomy a pravidla VL

* ***Schémata výrokových axiomů***

$$\begin{array}{ll} \text{(A1)} & A \to (B \to A) \\ \text{(A2)} & (A \to (B \to C)) \to ((A \to B) \to (A \to C)) \\ \text{(A3)} & (\neg B \to \neg A) \to ((\neg B \to A) \to B) \end{array}$$

kde A, B, C jsou libovolné výrokové formule.

* ***Odvozovací pravidlo modus ponens (pravidlo odloučení)***

(MP) Z *předpokladů* A a $(A \to B)$ odvodíme *závěr* B.

* (A1): Pokud A určitě platí a dovím se B, tak A bude pořád platit.
* (A2): V podstatě tranzitivita implikace.
* (A3): Pokud z výroku $(\neg B)$ plyne spor, pak neplatí. Tedy platí opak, B.
* (MP): Z (*prší* $\to$ *nemám deštník*) a *prší* odvodím, že *nemám deštník*.

Důkaz

Důkaz formule φ
je sekvence formulí $\varphi_1, \ldots, \varphi_n$, kde $\varphi_n = \varphi$ a pro každé $i : 1 \leq i \leq n$,
formule φ_i je axiomem nebo vznikla z $\varphi_1, \ldots \varphi_{i-1}$ aplikací odvozovacích pravidel.

Pokud existuje důkaz φ, tak je φ ***dokazatelná***, psáno $\vdash \boldsymbol{\varphi}$.

Důkaz z předpokladů

Důkaz formule φ ***z množiny předpokladů*** P,
kde P je množina formulí,
je sekvencí formulí $\varphi_1, \ldots, \varphi_n$, kde $\varphi_n = \varphi$ a pro každé $i : 1 \leq i \leq n$,
formule φ_i je axiomem ***nebo prvkem*** $\boldsymbol{P}$
nebo vznikla z $\varphi_1, \ldots \varphi_{i-1}$ aplikací odvozovacích pravidel. Píšeme $\boldsymbol{P \vdash \varphi}$.

Příklad důkazu: $\vdash A \to A$

(1)	$A \to ((A \to A) \to A)$	(A1)
(2)	$(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$	(A2)
(3)	$(A \to (A \to A)) \to (A \to A)$	(1),(2) MP
(4)	$A \to (A \to A)$	(A1)
(5)	$A \to A$	(3),(4) MP

Pozn.:
(1) je (A1) pro volbu A, $A \to A$ za A, B
(2) je (A2) pro volbu A, $A \to A$, A za A, B, C
(3) je závěr MP z předpokladů (1), (2)

Část II

Hilbertovský důkaz v predikátové logice

Axiomy PL

* ***Schémata výrokových axiomů***

$$\begin{array}{ll} (\mathsf{A1}) & \varphi \to (\psi \to \varphi) \\ (\mathsf{A2}) & (\varphi \to (\psi \to \eta)) \to ((\varphi \to \psi) \to (\varphi \to \eta)) \\ (\mathsf{A3}) & (\neg\psi \to \neg\varphi) \to ((\neg\psi \to \varphi) \to \psi) \end{array}$$

  kde φ, ψ, η jsou formule PL.

* ***Schéma axiomů kvantifikátoru***: Není-li x volné ve φ, pak

$$(\forall x(\varphi \to \psi)) \to (\varphi \to (\forall x \psi))$$

* ***Schéma axiomů substituce***:

$$(\forall x\, \varphi) \to \varphi[x/t]$$

  kde t je term substituovatelný za x do φ (proměnné t nejsou volné ve φ).

Axiomy PL

* Pokud pracujeme s jazykem s rovností, přidáváme ***axiomy rovnosti***:
  Pro libovolné funkční a predikátové symboly f/n a p/n a proměnné $x, x_1, \ldots, x_n, y_1, \ldots, y_n$

$$x = x$$

$$x_1 = y_1 \to (x_2 = y_2 \to (\ldots (x_n = y_n \to f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)) \ldots))$$

$$x_1 = y_1 \to (x_2 = y_2 \to (\ldots (x_n = y_n \to p(x_1, \ldots, x_n) \to p(y_1, \ldots, y_n)) \ldots))$$

Příklady axiomů PL

* axiom kvantifikátoru: $(\forall x(\varphi \to \psi)) \to (\varphi \to (\forall x \psi))$

$$(\forall \textit{Brňák}\ (\textit{je hezky} \to \textit{Brňák je na přehradě})) \to (\textit{je hezky} \to (\forall \textit{Brňák}(\textit{Brňák je na přehradě})))$$

* axiom substituce: $(\forall x\, \varphi) \to \varphi[x/t]$

$$(\forall x\, x < \infty) \to (5 + 5 < \infty)$$

* ax. rovnosti: $x_1 = y_1 \to (\ldots (x_n = y_n \to p(x_1, \ldots, x_n) \to p(y_1, \ldots, y_n)) \ldots)$

$$(\textit{pivo} = \textit{chleba} \to (\textit{snídaně}(\textit{chleba}, \textit{máslo}) \to \textit{snídaně}(\textit{pivo}, \textit{máslo}))$$

Odvozovací pravidla predikátové logiky

* ***Modus ponens***:

  Z předpokladů φ a $(\varphi \to \psi)$ odvodíme závěr ψ.

* ***Pravidlo zobecnění (generalizace)***:

  Z předpokladu φ odvodíme závěr $(\forall x \varphi)$.

Např., platí-li $x < \infty$, potom platí $(\forall x\, x < \infty)$.

Příklad důkazu v PL: $p(x, y) \vdash p(y, x)$

(1)	$p(x, y)$	(předpoklad)
(2)	$\forall y\, p(x, y)$	(pravidlo zobecnění)
(3)	$\forall x \forall y\, p(x, y)$	(pravidlo zobecnění)
(4)	$(\forall x \forall y\, p(x, y)) \to (\forall y\, p(z, y))$	(axiom substituce, x za z)
(5)	$\forall y\, p(z, y)$	(MP)
(6)	$(\forall y\, p(z,y)) \to p(z,x)$	(axiom substituce, y za x)
(7)	$p(z, x)$	(MP)
(8)	$(\forall z\, p(z, x))$	(pravidlo zobecnění)
(9)	$(\forall z\, p(z, x)) \to p(y, x)$	(axiom substituce, z za y)
(10)	$p(y, x)$	(MP)

Pozn.:
(1): předpoklad; (2): p. zobec. pro $\varphi = (1)$, x = y; (3): p. zobec. pro $\varphi = (2)$;
(4): ax. subst. pro $\varphi = (3)$ a $t = z$; (5): MP pro $\varphi = (3)$ a $\varphi \to \psi = (4)$;
(6): ax. subst. pro $\varphi = (5)$, $x = y$ a $t = x$; (7): MP pro $\varphi = (5)$ a $\varphi \to \psi = (6)$;
(8): ax. subst. pro $\varphi = (7)$, $x = z$; (9): ax. subst. pro $\varphi = (8)$, $x = z$ a $t = y$;
(10): MP pro $\varphi = (8)$ a $\varphi \to \psi = (9)$;

Formlání důkaz versus běžný důkaz

* Formální logický systém je jakýsi „asembler“ logického odvozování.
  - Důkaz je absolutně nezpochybnitelný, mechanicky zkontrolovatelný.
  - Proto je formální odvozování extrémně detailní, mechanicky jednoduché, a systém je minimalistický.
  - Formální systém je stavěn tak, aby se dobře přemýšlelo *o něm*, ne *v něm*.
* Běžný důkaz je poloformální.
  - Čtenář by měl snadno pochopit a zároveň nepochybovat.
  - Dělají se větší kroky, spoléhá se na čtenářovu intuici a znalosti.
  - Ideálně je expandovatelný na plně formální důkaz.
* Hilbertovský důkaz je snadno, mechanicky, ověřitelný,
  ale není snadno vymyslitelný, a to ani pomocí počítače.

K zapamatování

* Logický systém
* Odvozovací, důkazový systém.
* Axiom
* Odvozovací pravidlo
* Důkaz
* Důkaz z předpokladů

Část III

Rezoluční důkaz ve výrokové logice

Co je Rezoluce

* Rezoluce je důkazový systém, se kterým je *vymýšlení důkazu jednoduché a mechanizovatelné*.
* Je *základem řady technologií*, SAT-solverů, SMT-solverů, theorem-prooverů, programovacího jazyka Prolog, typové inference v programovacích jazycích,...
* Dá se chápat jako *strategie a systém maker* pro tvorbu Hilbertovského důkazu.

Rezoluční odvozovací pravidlo

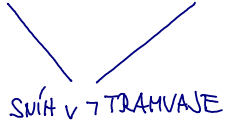
Rezoluční pravidlo

$$(\ell \vee A), (\neg\ell \vee B) \vdash A \vee B$$

* kde $(\ell \vee A), (\neg\ell \vee B)$ jsou ***konfliktní klauzule***
* $A \vee B$ je ***resolvent(a)***.

Například:

$$(\text{déšť} \vee \text{sníh}), (\neg\text{sníh} \vee \neg\text{tramvaje})$$



Rezoluční důkaz

Rezoluční důkaz φ ***z množiny předpokladů*** P je sekvence formulí patřících buď do P nebo odvozených z předchozích rezolučním pravidlem. Píšeme $P \vdash \varphi$. Rezoluční důkaz 0 z P je ***rezoluční vyvrácení*** P.

Rezoluční vyvrácení φ je ***důkazem nesplnitelnosti*** φ.

Platnost formule φ pomocí rezoluce dokážeme tak, že:

1. převedeme $\neg\varphi$ do ekvivalentní formule v CNF s množinou klauzulí C, a
2. rezolučně vyvrátíme C.

Příklad rezolučního důkazu

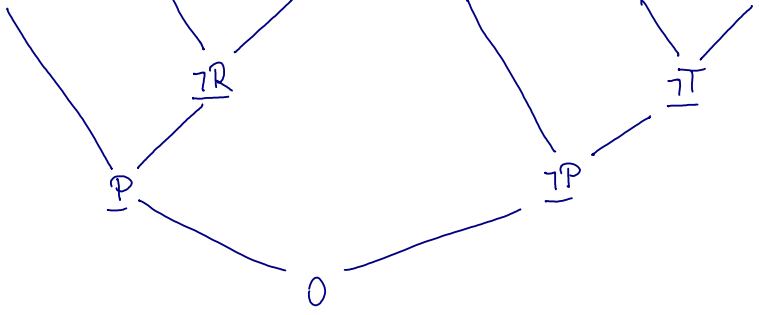
Dokazujeme rezolucí formuli φ:

$$\varphi: \quad (P \vee R) \to S \vee (\neg Q \wedge R) \vee (P \wedge \neg T) \vee ((S \to Q) \wedge (\neg T \to Q))$$

$$\neg[\quad (P \vee R) \to S \vee (\neg Q \wedge R) \vee (P \wedge \neg T) \vee ((S \to Q) \wedge (\neg T \to Q)) \quad] \quad \textbf{(neguj)}$$

$$\neg[\neg(P \vee R) \vee S \vee (\neg Q \wedge R) \vee (P \wedge \neg T) \vee ((\neg S \vee Q) \wedge (T \vee Q))] \quad \text{(impl.)}$$
$$(P \vee R) \wedge \neg S \wedge (Q \vee \neg R) \wedge (\neg P \vee T) \wedge ((S \wedge \neg Q) \vee (\neg T \wedge \neg Q)) \quad \text{(NNF)}$$
$$(P \vee R) \wedge \neg S \wedge (Q \vee \neg R) \wedge (\neg P \vee T) \wedge \neg Q \wedge (S \vee \neg T) \quad \text{(distrib.)}$$

$$(P \vee R) \wedge (Q \vee \neg R) \wedge \neg Q \wedge (\neg P \vee T) \wedge (S \vee \neg T) \wedge \neg S \quad \textbf{(CNF)}$$

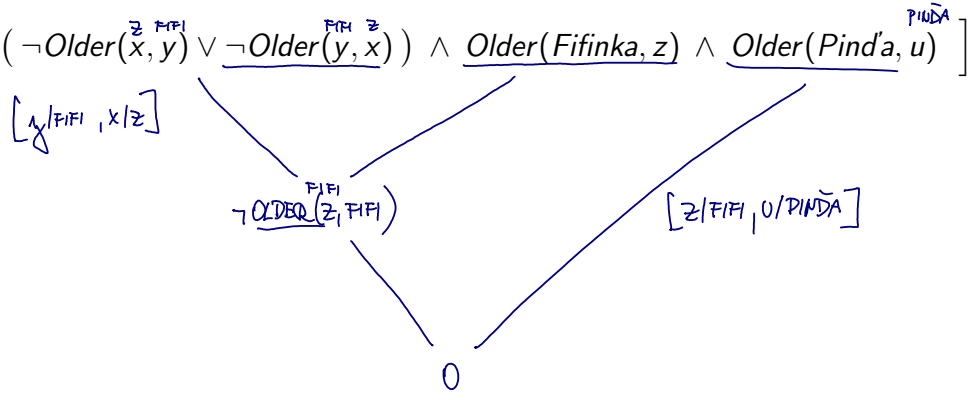


Část IV

Rezoluční důkaz v predikátové logice

Rezoluční inference v predikátové logice, jednoduchý příklad

$$\forall u \forall x \forall y \forall z \left[\left(\neg Older(x, y) \vee \neg Older(y, x) \right) \wedge Older(Fifinka, z) \wedge Older(Pinďa, u) \right]$$



Rezoluční inference v predikátové logice

Jak funguje rezoluční pravidlo $(\ell \vee A), (\neg\ell \vee B) \vdash A \vee B$ v PL?

$$\forall \boldsymbol{x}\,\forall \boldsymbol{y}\,\forall \boldsymbol{z}\,\Big[\;\begin{array}{rcl} \big(\neg\mathrm{eats}(x,y) \vee \mathrm{fears}(y,x)\big) & , & \big(\mathrm{eats}(x,\mathrm{prey}(x))\big) \\ \big(\neg\mathrm{eats}(x,y) \vee \mathrm{fears}(y,x)\big) & \wedge & \forall \boldsymbol{x}\,\big(\mathrm{eats}(x,\mathrm{prey}(x))\big) \\ \big(\neg\mathrm{eats}(x,y) \vee \mathrm{fears}(y,x)\big) & \wedge & \big(\mathrm{eats}(\boldsymbol{z},\mathrm{prey}(\boldsymbol{z}))\big) \end{array}\;\Big]$$

Vyvracíme platnost, t.j., ukázujeme existenci protipříkladu, ***nějakých hodnot proměnných***, pro které formule uvnitř kvant. ***neplatí***.
Můžeme tedy jejich hodnoty zkusit omezit tak, aby bylo potom možné použít predikátové rezoluční pravidlo.
Zde konkrétně pomůže omezit se na x stejné jako z a y stejné jako $\mathrm{prey}(z)$.

$$\forall \boldsymbol{z}\,\Big[\;\big(\underline{\neg\mathrm{eats}(\boldsymbol{z},\mathbf{prey}(\boldsymbol{z}))} \vee \mathrm{fears}(\mathbf{prey}(\boldsymbol{z}),\boldsymbol{z})\big) \;\wedge\; \big(\underline{\mathrm{eats}(z,\mathrm{prey}(z))}\big)\;\Big]$$

$$\forall z\,\Big[\;\big(\,\mathrm{fears}(\mathrm{prey}(z),z)\,\big)\;\Big]$$

Substituce

Substituce je zobrazení σ z proměnných do termů.
$[x_1/t_1, \ldots, x_n/t_n]$ značí substituci σ, kde $\sigma(x_i) = t_i$ pro $1 \leq i \leq n$ a $\sigma(x) = x$ pro ostatní proměnné (množina proměnných bude jasná z kontextu).

Aplikací substituce σ na výraz (formuli nebo term) E dostaneme výraz $E\sigma$, a to *současným* nahrazením každého volného výskytu proměnné x termem $\sigma(x)$.

Např. pro $E = p(x) \vee p(f(y))$ a $\sigma = [x/y, y/f(a)]$ dostaneme $E\sigma = p(y) \vee p(f(f(a)))$.
Slovem „současně“ v definici aplikace je myšleno, že nenahrazujeme postupně x za y a potom y za $f(a)$, to bychom dostali ~~$p(f(a)) \vee p(f(f(a)))$~~.

Unifikace

Unifikátor dvou výrazů E a E' je substituce μ taková, že $E\mu = E'\mu$.
Pokud existuje, potom jsou E a E' ***unifikovatelné***.

Například, nechť a, b, c jsou konstanty, x, y, z proměnné.
Unifikátor $p(x, c)$ a $p(b, c)$ je $[x/b]$.
Unifikátor $p(f(x), y)$ a $p(f(a), z)$ je $[x/a, y/z]$, $[x/a, y/a, z/a]$, $[x/a, y/f(b), z/f(b)]$ ...
$p(x, a)$ a $p(b, c)$ nejsou unifikovatelné, stejně jako $p(f(x), z)$ a $p(a, y)$.

Maximálně obecný unifikátor (mgu) výrazů E a E' je unifikátor μ takový, že každý unifikátor E a E' dostaneme jeho složením $\sigma \circ \mu$ s nějakou další substitucí σ.

Např. mgu $p(f(x), y)$ a $p(f(a), z)$ je $\mu = [x/a, y/z]$,
ale ne $\mu_1 = [x/a, y/a, z/a]$ ani $\mu_2 = [x/a, y/f(b), z/f(b)]$.
Opravdu, $\mu_1 = [z/a] \circ \mu$ a $\mu_2 = [z/f(b)] \circ \mu$.

Unifkátor množiny M výrazů je unifikátor každých dvou výrazů v M.
Mgu množiny je takový unifikátor, jehož složením se substitucí dostaneme kterýkoliv jiný.

Jednoduchý algoritmus pro nalezení mgu

Input: množina M výrazů
Output: mgu μ pro M, pokud existuje, jinak M není unifikovatelná

1 $\mu = \emptyset$
2 **while** *existují výrazy* $E, E \in M$ *a nejlevější pozice* i*, kde se* $E[i]$ *liší od* $E'[i]$ **do**
3 **if** $E[i]$ *ani* $E'[i]$ *není proměnná* **then**
4 **return** M *není unifikovatelná*
5 Nechť, w.l.o.g., $E[i]$ je proměnná x a na $E'[i]$ je term t.
6 **if** x se vyskytuje v t **then**
7 **return** M *není unifikovatelná*
8 **else**
9 $\mu = [x/t] \circ \mu$
10 Všechny výrazy $E'' \in M$ nahraď za $E''[x/t]$
11 **return** μ (nejobecnější unifikátor, mgu, množiny M)

Výpočet mgu

$$M = \{\ P(z, w, a)\ ,\quad P(u, g(y), y)\ ,\quad P(f(x), x, v)\ \}$$

$P(\ \boldsymbol{z}\ ,\ w\ ,\ a\ )$	$P(\ \boldsymbol{u}\ ,\ g(y)\ ,\ y)$	$P(\ f(x)\ ,\ x\ ,\ v\ )$	$[z/u]$
$P(\ u\ ,\ \boldsymbol{w}\ ,\ a\ )$	$P(\ u\ ,\ \boldsymbol{g(y)}\ ,\ y)$	$P(\ f(x)\ ,\ x\ ,\ v\ )$	$[w/g(y)]$
$P(\ u\ ,\ g(y)\ ,\ \boldsymbol{a}\ )$	$P(\ u\ ,\ g(y)\ ,\ \boldsymbol{y})$	$P(\ f(x)\ ,\ x\ ,\ v\ )$	$[y/a]$
$P(\ u\ ,\ g(a)\ ,\ a\ )$	$P(\ \boldsymbol{u}\ ,\ g(a)\ ,\ a)$	$P(\ \boldsymbol{f(x)}\ ,\ x\ ,\ v\ )$	$[u/f(x)]$
$P(\ f(x)\ ,\ g(a)\ ,\ a\ )$	$P(\ f(x)\ ,\ \boldsymbol{g(a)}\ ,\ a)$	$P(\ f(x)\ ,\ \boldsymbol{x}\ ,\ v\ )$	$[x/g(a)]$
$P(\ f(g(a))\ ,\ g(a)\ ,\ a\ )$	$P(\ f(g(a))\ ,\ g(a)\ ,\ \boldsymbol{a})$	$P(\ f(g(a))\ ,\ g(a)\ ,\ \boldsymbol{v}\ )$	$[v/a]$
$P(\ f(g(a))\ ,\ g(a)\ ,\ a\ )$	$P(\ f(g(a))\ ,\ g(a)\ ,\ a)$	$P(\ f(g(a))\ ,\ g(a)\ ,\ a\ )$	

Řešení: Nejobecnější unifikátor M (mgu) je substituce

$$[v/a] \circ [x/g(a)] \circ [u/f(x)] \circ [y/a] \circ [w/g(y)] \circ [z/u] =$$
$$[z/f(g(a)), w/g(a), u/f(g(a)), y/a, x/g(a), v/a]$$

Rezoluční odvozovací pravidlo predikátové logiky, základní

Rezoluční pravidlo

$$(\ell \vee \varphi), (\neg \ell' \vee \psi) \vdash \varphi\mu \vee \psi\mu$$

pokud $\texttt{Free}(\ell \vee \varphi) \cap \texttt{Free}(\neg \ell' \vee \psi) = \emptyset$ a μ je mgu pro ℓ a ℓ'.

* kde $(\ell \vee \varphi), (\neg \ell \vee \psi)$ jsou ***konfliktní klauzule***,
* $\varphi \vee \psi$ je ***resolvent(a)***.

! Pokud není splněna podmínka na prázdný průnik proměnných, je třeba před použitím rezolučního pravidla ***přejmenovat proměnné***!

$$\big(\neg(x = 0)\big) \ , \quad \big(z = 0 \vee z > (x - x)\big)$$

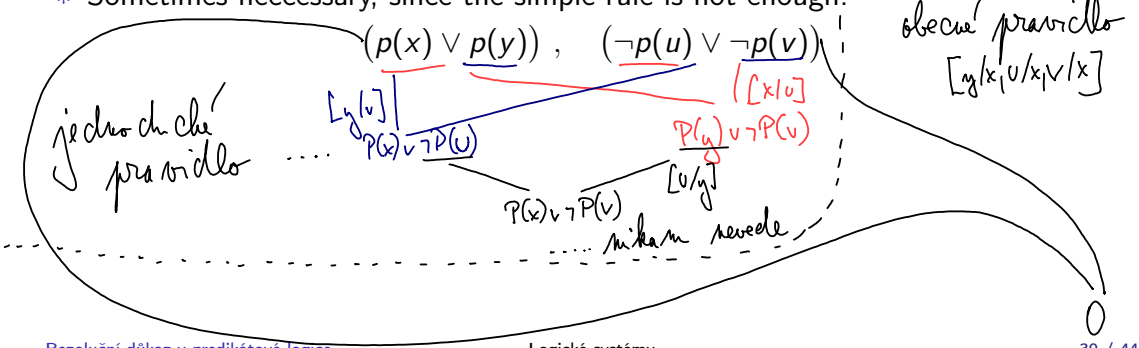
Rezoluční odvozovací pravidlo predikátové logiky, obecné

Obecná verze rezolučního pravidla

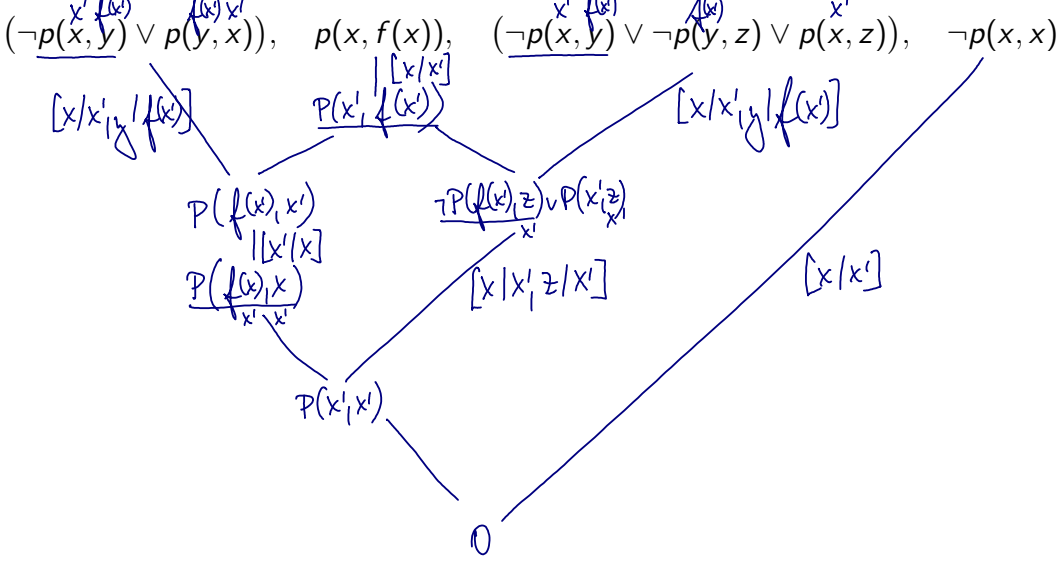
$$(\ell_1 \vee \ldots \vee \ell_n \vee \varphi), (\neg\ell_{n+1} \vee \ldots \vee \neg\ell_m \vee \psi) \vdash \varphi\mu \vee \psi\mu$$

pokud $1 \leq n < m$, $\texttt{Free}(\ell_1 \vee \ldots \vee \ell_n \vee \varphi) \cap \texttt{Free}(\neg\ell_{n+1} \vee \ldots \vee \neg\ell_m \vee \psi) = \emptyset$ a μ je mgu pro $\{\ell_1, \ldots, \ell_m\}$.

* Sometimes neccessary, since the simple rule is not enough.



Rezoluční vyvrácení formule v CNF



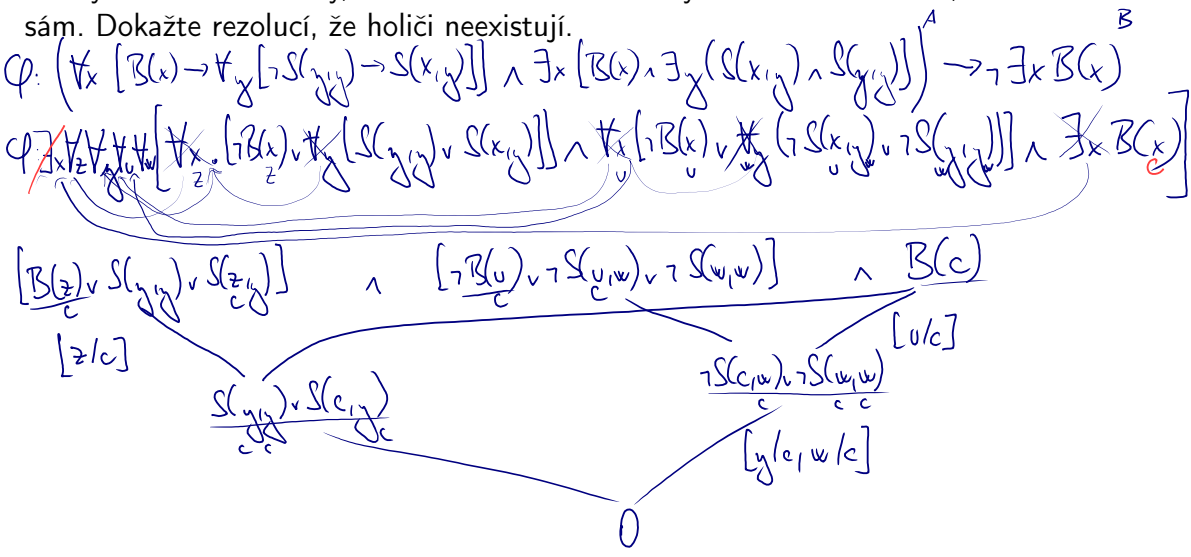
Rezoluční důkaz

Rezoluční důkaz z předpokladů a rezoluční vyvrácení definujeme stejně, jako v PL.
Platnost formule φ pomocí rezoluce dokážeme těmito kroky:

1. převod $\neg\varphi$ do CNF (klauzulární n.f.) s množinou klauzulí C:
   i. převod do NNF
   ii. převod do PNF (kvantifikátory ven)
   iii. skolemizace (nahrazení existenciálně kvant. proměnných skolemovými funkcemi)
   iv. smazání kvantifikátorů (univerzálních)
   v. převod do CNF (konjunktivní n.f.) roznásobením (distributivita)
2. rezoluční vyvrácení C

Rezoluční důkaz, slovní úloha

Každý holič holí všechny, kdo se neholí sami. Žádný holič neholí nikoho, kdo se holí sám. Dokažte rezolucí, že holiči neexistují.



Rezoluce v praxi

Viděli jsme jen základní princip rezoluce.
V praxi se používají efektivnější varianty, např. Predikátová SLD-rezoluce v Prologu, s obrovským množstvím optimalizací, podobně výroková rezoluce v SAT a SMT-solverech.
Existují další důkazové metody, tabla, přirozená dedukce, sekventový kalkul, ...

Část V

Vlastnosti logických systémů

Ideál logických systémů

* V ideálním logickém systému

  existuje mechanicky ověřitelný důkaz/vyvrácení všech zajímavých tvrzení.

  Chtěli bych také ***mechanizovat generování důkazů***.


HILBERT
PODME NA TO!

Efektivnost (mechanická ověřitelnost)

Logický systém je efektivní, pokud můžeme efektivně (=mechanicky) ověřit korektnost logického argumentu, důkazu. Můžeme pro něj napsat program Ověřovač, který ověří, že daný řetězec symbolů je důkaz.

Výroková i predikátová logika je efektivní. (Rezoluce i Hilbertovský důkaz)

* Protože můžeme algoritmicky ověřit
  1. co je dobře formulovaná formule,
  2. co je axiom,
  3. že formule v důkazu byla odvozena odvozovacími pravidly z předchozích.

Dokazatelnost versus Platnost

$\vdash \varphi$	$\models \varphi$
Dokazatelná (z axiomů pomocí odvozovacích pravidel) Čistě syntaktická manipulace.	Platná ve všech interpretacích. Rozhoduje sémantika (význam).

$\vdash \varphi$ je ověřením $\models \varphi$
Důkaz je ověřením platnosti.

Mělo byt tedy platit, že
$\vdash \varphi \iff \models \varphi.$

Korektnost

Systém je ***korektní*** pokud

$$\vdash \varphi \implies \models \varphi$$

Co je dokazatelné, to je platné.
Nemůžeme dokázat nesmysly.

Úplnost (sémantická)

Systém je ***úplný*** pokud

$$\models \varphi \implies \vdash \varphi$$

Vše platné můžeme dokázat.

Korektnost a sém. úplnost Hilbertovského systému pro VL a PL

Výroková i predikátová logika je korektní a úplná. (Post, Gödel)
Pro libovolnou formuli VL nebo PL platí $\models \varphi \iff \vdash \varphi$.

Pro PL je to Gödelova věta o úplnosti.

Kurt Gödel

Brno, 1906 - 1978

Korektnost a sém. úplnost rezoluce pro VL a PL

Rezoluce ve VL i v PL je ***korektní*** a ***refutačně úplná***.

Refutační úplnost rezoluce znamená, že pro každou nesplnitelnou formuli existuje rezoluční vyvrácení, t.j., pokud $\not\models \varphi$, potom existuje rezoluční vyvrácení (refutace) φ ($\varphi \vdash 0$, důkaz nesplnitelnosti φ).

To je ekvivalentní klasické úplnosti: platná formule má nesplnitelnou negaci, rezoluční vyvrácení negace formule je důkazem platnosti formule.

K zapamatování

* Efektivnost
* Korektnost
* Sémantická úplnost
* Gödelova věta o úplnosti PL (dokazatelné je platné) pro Hilbertovský systém
* Rezoluce je refutačně úplná, a vlastně také „normálně“ úplná.