6. Logika a UI, rezoluční metoda a její využití při řešení úloh

Výroková logika

Formální systém výrokové logiky tvoří tři složky

- Jazyk výrokové logiky,
- Axiómy výrokové logiky (A a B jsou formule):

  $A \Rightarrow (B \Rightarrow A)$,

  $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$,

  $(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$.
- Odvozovací pravidlo modus ponens:

  Z formulí A, $A \Rightarrow B$ odvoď formuli B.

Jazyk výrokové logiky:

- Množina prvotních formulí (výroků).
- Symboly pro logické spojky: $\neg$, $\Rightarrow$, $\vee$, $\wedge$, $\Leftrightarrow$.
- Pomocné symboly: kulaté závorky a čárka.

Prvotní formule (atom): Je buď pravdivá – true,
nebo nepravdivá – false.

Formule:

- Každá prvotní formule je formule.
- Jsou-li A a B formule,
  pak i $\neg A$, $A \vee B$, $A \wedge B$, $A \Rightarrow B$ a $A \Leftrightarrow B$ jsou formule.

A	B	$\neg$A	$\neg$B	A $\vee$ B	A $\wedge$ B	A $\Rightarrow$ B	A $\Leftrightarrow$ B
F	F	T	T	F	F	T	T
F	T	T	F	T	F	T	F
T	F	F	T	T	F	F	F
T	T	F	F	T	T	T	T

Zákony výrokové logiky:

1. Zákon ekvivalence

$$A \Leftrightarrow B \quad \Leftrightarrow \quad (A \Rightarrow B) \wedge (B \Rightarrow A)$$

2. Zákon implikace

$$A \Rightarrow B \quad \Leftrightarrow \quad \neg A \vee B$$

3. Zákony komutativní

$$A \vee B \quad \Leftrightarrow \quad B \vee A$$
$$A \wedge B \quad \Leftrightarrow \quad B \wedge A$$

4. Zákony asociativní

$$A \wedge (B \wedge C) \quad \Leftrightarrow \quad (A \wedge B) \wedge C$$
$$A \vee (B \vee C) \quad \Leftrightarrow \quad (A \vee B) \vee C$$

5. Zákony distributivní

$$A \vee (B \wedge C) \quad \Leftrightarrow \quad (A \vee B) \wedge (A \vee C)$$
$$A \wedge (B \vee C) \quad \Leftrightarrow \quad (A \wedge B) \vee (A \wedge C)$$

6. Zákony zjednodušení disjunkce

$A \vee T \Leftrightarrow T$

$A \vee F \Leftrightarrow A$

$A \vee A \Leftrightarrow A$

$A \vee (A \wedge B) \Leftrightarrow A$

7. Zákony zjednodušení konjunkce

$A \wedge T \Leftrightarrow A$

$A \wedge F \Leftrightarrow F$

$A \wedge A \Leftrightarrow A$

$A \wedge (A \vee B) \Leftrightarrow A$

8. Zákon vyloučeného třetího

$A \vee \neg A \Leftrightarrow T$

9. Zákon sporu

$\neg(A \vee \neg A) \Leftrightarrow F$

Pozn.: Formálně je symbolem T (true) označena nějaká logicky pravdivá formule (tautologie), například $A \Rightarrow A$, a symbolem F (false) pak nějaká nesplnitelná formule (kontradikce), například $A \wedge \neg A$.

10. Zákon identity

$$A \Leftrightarrow A$$

11. Zákon negace

$$\neg(\neg A) \Leftrightarrow A$$

12. Zákony De Morganovy

$$\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$$

$$\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$$

Eliminace AND

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \Rightarrow A_i \qquad i \in \langle 1,n \rangle$$

Zavedení OR

$$A_i \Rightarrow A_1 \vee A_2 \vee \ldots \vee A_n \qquad i \in \langle 1,n \rangle$$

Pravdivostní ohodnocení prvotních formulí v dané formuli se nazývá interpretací této formule.

Základní pojmy:

- Tautologie (logicky pravdivá formule, je pravdivá v každé interpretaci).
- Kontradikce (nesplnitelná formule, není pravdivá v žádné interpretaci).
- Splnitelná formule (je splnitelná alespoň v jedné interpretaci).
- Ekvivalentní formule (jsou splnitelné ve stejných interpretacích).
- Literál (prvotní formule, nebo její negace).
- Klauzule (disjunkce literálů ($L_1 \vee L_2 \vee ... \vee L_n$)).
- Prázdná klauzule (označuje se prázdným čtverečkem $\square$ a je nesplnitelnou formulí).
- Disjunktivní normální forma (DNF – Disjunctive Normal Form): Disjunkce formulí, které jsou konjunkcemi literálů.
- Konjunktivní normální forma (CNF – Conjunctive Normal Form): Konjunkce formulí, které jsou disjunkcemi literálů.

Příklady: formule $(A \vee \neg A)$ je tautologií,
formule $(A \wedge \neg A)$ je kontradikcí,
formule $(A \vee \neg B)$ je splnitelnou formulí,
formule $\neg(A \vee B)$ a $(\neg A \wedge \neg B)$ jsou ekvivalentní.

Příklad na převedení formule $(A \vee \neg B) \Rightarrow C$ do

DNF: $(A \vee \neg B) \Rightarrow C \Leftrightarrow \neg(A \vee \neg B) \vee C \Leftrightarrow (\neg A \wedge B) \vee C$

CNF: $(A \vee \neg B) \Rightarrow C \Leftrightarrow (\neg A \wedge B) \vee C \Leftrightarrow (\neg A \vee C) \wedge (B \vee C)$

Konjunktivní normální forma je tedy konjunkcí klauzulí a často se zapisuje v množinovém tvaru:

$$(C_1 \wedge C_2 \wedge \ldots \wedge C_n) \equiv \{C_1, C_2, \ldots, C_n\}$$

Formule G je logickým důsledkem (logicky vyplývá z) formulí $F_1, F_2, ..., F_n$, je-li pravdivá v každé interpretaci, v níž je pravdivá formule $(F_1 \wedge F_2 \wedge ,..., \wedge F_n)$, tj. pokud je formule $((F_1 \wedge F_2 \wedge ... \wedge F_n) \Rightarrow G)$ tautologií.

Formule $((F_1 \wedge F_2 \wedge ,..., \wedge F_n) \Rightarrow G)$ se pak nazývá teorémem, formule $(F_1 \wedge F_2 \wedge ... \wedge F_n)$ premisami (postuláty) a formule G tvrzením tohoto teorému.

Je zřejmé, že pokud je nějaká formule G logickým důsledkem formulí $F_1 \wedge F_2 \wedge ... \wedge F_n$, tj. je-li formule $((F_1 \wedge F_2 \wedge ... \wedge F_n) \Rightarrow G)$ tautologií, musí být formule $\neg((F_1 \wedge F_2 \wedge ... \wedge F_n) \Rightarrow G)$ kontradikcí.

Úprava formule $\neg((F_1 \wedge F_2 \wedge ... \wedge F_n) \Rightarrow G)$:

$$\neg((F_1 \wedge F_2 \wedge ... \wedge F_n) \Rightarrow G) \quad \Leftrightarrow$$
$$\neg(\neg(F_1 \wedge F_2 \wedge ... \wedge F_n) \vee G) \quad \Leftrightarrow$$
$$((F_1 \wedge F_2 \wedge ... \wedge F_n) \wedge \neg G) \quad \Leftrightarrow$$
$$(F_1 \wedge F_2 \wedge ... \wedge F_n \wedge \neg G)$$

pak vede k velmi důležitému závěru:

Formule G je logickým důsledkem formulí $(F_1 \wedge F_2 \wedge ... \wedge F_n)$, když a jen když je formule $(F_1 \wedge F_2 \wedge ... \wedge F_n \wedge \neg G)$ nesplnitelná.

Příklad: Postup důkazu, že formule C je logickým důsledkem formulí $(A \Rightarrow B), (B \Rightarrow C), A$

a) Postup při důkazu, že formule $((A \Rightarrow B) \wedge (B \Rightarrow C) \wedge A) \Rightarrow C$ je tautologií:

$$
\begin{aligned}
&((A \Rightarrow B) \wedge (B \Rightarrow C) \wedge A) \Rightarrow C && \Leftrightarrow \\
&\neg((\neg A \vee B) \wedge (\neg B \vee C) \wedge A) \vee C && \Leftrightarrow \\
&(\neg (\neg A \vee B) \vee \neg(\neg B \vee C) \vee \neg A) \vee C && \Leftrightarrow \\
&(A \wedge \neg B) \vee (B \wedge \neg C) \vee \neg A \vee C && \Leftrightarrow \\
&((A \wedge \neg B) \vee \neg A) \vee ((B \wedge \neg C) \vee C) && \Leftrightarrow \\
&((A \vee \neg A) \wedge (\neg B \vee \neg A)) \vee ((B \vee C) \wedge (\neg C \vee C)) && \Leftrightarrow \\
&(T \wedge (\neg B \vee \neg A)) \vee ((B \vee C) \wedge T) && \Leftrightarrow \\
&\neg B \vee \neg A \vee B \vee C && \Leftrightarrow \\
&T
\end{aligned}
$$

b) Postup důkazu, že formule $\neg(((A \Rightarrow B) \wedge (B \Rightarrow C) \wedge A) \Rightarrow C)$ je kontradikcí:

$$\begin{aligned}
&\neg(((A \Rightarrow B) \wedge (B \Rightarrow C) \wedge A) \Rightarrow C) && \Leftrightarrow \\
&\neg(\neg((A \Rightarrow B) \wedge (B \Rightarrow C) \wedge A) \vee C) && \Leftrightarrow \\
&\neg(\neg((\neg A \vee B) \wedge (\neg B \vee C) \wedge A) \vee C) && \Leftrightarrow \\
&(\neg A \vee B) \wedge (\neg B \vee C) \wedge A \wedge \neg C && \Leftrightarrow \\
&((\neg A \vee B) \wedge A) \wedge ((\neg B \vee C) \wedge \neg C) && \Leftrightarrow \\
&((\neg A \wedge A) \vee (B \wedge A)) \wedge ((\neg B \wedge \neg C) \vee (C \wedge \neg C)) && \Leftrightarrow \\
&((F \vee (B \wedge A)) \wedge ((\neg B \wedge \neg C) \vee F)) && \Leftrightarrow \\
&B \wedge A \wedge \neg B \wedge \neg C && \Leftrightarrow \\
&F
\end{aligned}$$

Predikátová logika (1. řádu)

Formální systém predikátové logiky tvoří tři složky:

- Jazyk predikátové logiky (JPL),
- Axiómy predikátové logiky:
  - Axiómy výrokové logiky,
  - schéma specifikace $\forall x(\mathrm{A}) \Rightarrow \mathrm{A}_x[t]$, kde $\mathrm{A}_x[t]$ je formule vzniklá substitucí termu t za proměnnou x ve formuli A,
  - schéma kvantifikace implikace $\forall x(\mathrm{A} \Rightarrow \mathrm{B}) \Rightarrow (\mathrm{A} \Rightarrow \forall x(\mathrm{B}))$, kde proměnná x nesmí být volná ve formuli A.
- Odvozovací pravidla:
  - Modus ponens,
  - pravidlo generalizace: Pro libovolnou proměnnou x odvoď z formule A formuli $\forall x(\mathrm{A})$.

JPL:

- Individuové konstanty: a, b, c, ...,
- individuové proměnné: x, y, z, ...,
- funkční symboly: f, g, h, ...,
- predikátové symboly: P, Q, R, ...,
- logické spojky: $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$,
- kvantifikátory (univerzální a existenční): $\forall$, $\exists$,
- pomocné symboly (kulaté závorky, hranaté závorky a čárka).

Individuová proměnná se nazývá vázaná, je-li v oblasti působnosti některého z kvantifikátorů, jinak se nazývá volná.

Term je individuová konstanta, nebo individuová proměnná, nebo struktura $f(t_1, t_2, ..., t_n)$, kde f je n-místný funkční symbol a t_i jsou termy.

Atomická formule je struktura $P(t_1, t_2, ..., t_n)$, kde P je *n*-místný predikátový symbol a t_i jsou termy.

Formule:

- Každá atomická formule je formule,
- jsou-li A a B formule, pak i $\neg A$, $A \vee B$, $A \wedge B$, $A \Rightarrow B$ a $A \Leftrightarrow B$ jsou formule,
- je-li *x* individuová proměnná a A formule, pak i $\forall x(A)$ a $\exists x(A)$ jsou formule.

Literál je atomická formule nebo její negace.

Formule $\forall x_1 \forall x_2 ... \forall x_n(A)$ se nazývá univerzálním uzávěrem formule A, jsou-li všechny volné proměnné ve formuli A z množiny $\{x_1, x_2, ..., x_n\}$.

Obdobně je definován i existenční uzávěr formule A.

Prenexní normální forma formule A má tvar $Q_1x_1...Q_nx_n(M)$, kde x_i jsou navzájem různé proměnné, Q_i jsou univerzální, nebo existenční kvantifikátory a M je otevřená formule (tzv. jádro formule A) vzniklá z atomických formulí použitím logických spojek. Posloupnost kvantifikátorů $Q_1x_1...Q_nx_n$ … se nazývá prefixem formule A.

Interpretací **I** formule A nad libovolnou neprázdnou množinou D (oborem interpretace - univerzem) se rozumí:

- Přiřazení pevně zvoleného prvku z D každé individuové konstantě,
- přiřazení libovolného prvku z D každé individuové proměnné,
- přiřazení *n*-ární operace každému *n*-místnému funkčnímu symbolu $D^n \rightarrow D$,
- přiřazení *n*-ární relace každému predikátovému symbolu $D^n \rightarrow \{T, F\}$.

Příklad: Důkaz, že formule $\forall x(P(a, x) \Rightarrow Q(f(x)))$ je pravdivá v interpretaci **I**:

D = {1,2,3}	a = 2	f(x) = 4 - *x*
P(1, 1) = F	P(1, 2) = T	P(1, 3) = T
P(2, 1) = T	P(2, 2) = F	P(2, 3) = F
P(3, 1) = T	P(3, 2) = T	P(3, 3) = F
Q(1) = T	Q(2) = F	Q(3) = T

Postup při odvozování důkazu:

$\forall x(P(a, x) \Rightarrow Q(f(x)))$ $\Leftrightarrow$

$(P(2, 1) \Rightarrow Q(f(1))) \wedge (P(2, 2) \Rightarrow Q(f(2))) \wedge (P(2, 3) \Rightarrow Q(f(3)))$ $\Leftrightarrow$

$(T \Rightarrow T) \wedge (F \Rightarrow F) \wedge (F \Rightarrow T)$ $\Leftrightarrow$

$T \wedge T \wedge T$ $\Leftrightarrow$

T

Zákony výrokové logiky jsou v JPL doplněny o zákony:

13. Přesun kvantifikátorů doleva (*x* se nevyskytuje v B!)

$$Qx(A) \vee B \quad \Leftrightarrow \quad Qx(A \vee B)$$
$$Qx(A) \wedge B \quad \Leftrightarrow \quad Qx(A \wedge B)$$

14. Přesun negace dovnitř

$$\neg(\forall x(A)) \quad \Leftrightarrow \quad \exists x(\neg A)$$
$$\neg(\exists x(A)) \quad \Leftrightarrow \quad \forall x(\neg A)$$

15. Distributivní pro kvantifikátory

$$\forall x(A) \wedge \forall x(B) \quad \Leftrightarrow \quad \forall x(A \wedge B)$$
$$\exists x(A) \vee \exists x(B) \quad \Leftrightarrow \quad \exists x(A \vee B)$$

16. Přejmenování vázaných proměnných

$$\forall x(A(x)) \vee \forall x(B(x)) \quad \Leftrightarrow \quad \forall x \forall y(A(x) \vee B(y))$$
$$\exists x(A(x)) \wedge \exists x(B(x)) \quad \Leftrightarrow \quad \exists x \exists y(A(x) \wedge B(y))$$

Příklad: Převedení formule

$\forall x \forall y(\exists z(\mathrm{A}(x, z) \wedge \mathrm{B}(y, \mathrm{z})) \Rightarrow \exists v(\mathrm{C}(v, x, y)))$

do prenexní normální formy:

$$\forall x \forall y(\exists z(\mathrm{A}(x, z) \wedge \mathrm{B}(y, z)) \Rightarrow \exists v(\mathrm{C}(v, x, y))) \quad \Leftrightarrow$$

$$\forall x \forall y(\neg \exists z(\mathrm{A}(x, z) \wedge \mathrm{B}(y, z)) \vee \exists v(\mathrm{C}(v, x, y))) \quad \Leftrightarrow$$

$$\forall x \forall y(\forall z(\neg(\mathrm{A}(x, z) \wedge \mathrm{B}(y, z))) \vee \exists v(\mathrm{C}(v, x, y))) \quad \Leftrightarrow$$

$$\forall x \forall y \forall z(\neg \mathrm{A}(x, z) \vee \neg \mathrm{B}(y, z) \vee \exists \mathrm{v}(\mathrm{C}(v, x, y))) \quad \Leftrightarrow$$

$$\forall x \forall y \forall z \exists v(\neg \mathrm{A}(x, z) \vee \neg \mathrm{B}(y, z) \vee \mathrm{C}(v, x, y))$$

Skolemizace (odstranění existenčních kvantifikátorů):

1) Nechť $\exists x_1$ je prvním kvantifikátorem v prefixu formule A zleva ($\exists x_1\, Q_2 x_2 \ldots Q_n x_n(M)$). Pak se vybere libovolná individuová konstanta, která se dosud v matici M nevyskytuje, tzv. Skolemova konstanta, nahradí se jí všechny výskyty proměnné x_1 v matici M a kvantifikátor $\exists x_1$ se z prefixu formule odstraní.
2) Nechť $\exists x_{m+1}$ je prvním existenčním kvantifikátorem v prefixu formule A zleva ($\forall_1 x_1 \ldots \forall_m x_m \exists x_{m+1}\, Q_{m+2} x_{m+2} \ldots Q_n x_n(M)$). Pak se vybere m-místná struktura $f(x_1, \ldots, x_m)$, kde f je funkční symbol, který se dosud v matici M nevyskytuje, tzv. Skolemova funkce, nahradí se jí všechny výskyty proměnné x_{m+1} v matici M a kvantifikátor $\exists x_{m+1}$ se z prefixu formule odstraní.

Skolemova normální forma je prenexní normální forma (obvykle CNF) s výhradně univerzálními kvantifikátory.

Příklad: Převedení formule $\exists x \forall y \exists z(\neg A(x, y) \wedge (B(x, z) \vee C(x, y, z)))$ do Skolemovy normální formy:

$$\exists x \forall y \exists z(\neg A(x, y) \wedge (B(x, z) \vee C(x, y, z))) \quad \Leftrightarrow$$
$$\forall y \exists z(\neg A(a, y) \wedge (B(a, z) \vee C(a, y, z))) \quad \Leftrightarrow$$
$$\forall y(\neg A(a, y) \wedge (B(a, f(y)) \vee C(a, y, f(y))))$$

Odpovídající množina klauzulí:

$$S = \{\neg A(a, y), (B(a, f(y)) \vee C(a, y, f(y)))\}$$

Rezoluční metoda - pravidlo základní rezoluce

Nechť C_1 a C_2 jsou klauzule a nechť první z nich obsahuje literál L a druhá negaci tohoto literálu $\neg L$.

Pak rezolventou klauzulí C_1 a C_2 je klauzule

$$C = (C_1 - L) \vee (C_2 - \neg L),$$

která je logickým důsledkem klauzulí C_1 a C_2.

Robinsonův rezoluční princip:

> Výchozí množina klauzulí je nesplnitelná právě tehdy, když se z ní podaří odvodit prázdnou klauzuli $\square$.

Rezoluční metoda - důkaz:

Nechť C_1 a C_2 jsou logicky pravdivé klauzule:

$C_1 = A_1 \vee A_2 \vee \ldots \vee A_n \vee L$
$C_2 = B_1 \vee B_2 \vee \ldots \vee B_m \vee \neg L$

Je-li pravdivý literál L, pak jeho negace je nepravdivá, a proto musí být pravdivá formule $B_1 \vee B_2 \vee \ldots \vee B_m$, neboli formule $(C_2 - \neg L)$,

Je-li naopak literál L nepravdivý, musí být pravdivá formule $A_1 \vee A_2 \vee \ldots \vee A_n$, neboli formule $(C_1 - L)$.

V každém případě tedy musí být pravdivá formule

$$C = (C_1 - L) \vee (C_2 - \neg L)$$

Příklad: Důkaz nesplnitelnosti množiny klauzulí.

1)	$P(a,b)$	
2)	$\neg P(a,b) \vee \neg Q \vee R(c)$	
3)	$\neg R(c)$	
4)	$\neg T(a,c) \vee Q$	
5)	$T(a,c)$	

Postup při důkazu:

6)	$\neg Q \vee R(c)$	rezolventa z 1, 2
7)	$\neg Q$	rezolventa z 6, 3
8)	$\neg T(a,c)$	rezolventa z 7, 4
9)	$\square$	rezolventa z 8, 5

V případě množiny klauzulí s proměnnými se nejprve musí provést tzv. unifikace - substituce termů za proměnné:

- Substitucí σ je konečná množina dvojic $\{x_1/t_1, ..., x_n/t_n\}$, kde x_i jsou proměnné a t_i jsou termy.
- Instancí klauzule C je klauzule $C\sigma$ získaná z klauzule C postupnou náhradou každého výskytu proměnné x_i termem t_i ze substituce σ.
- Unifikátorem λ množiny literálů $\{L_1, L_2, ..., L_k\}$ je substituce, pro kterou platí $L_1\lambda = L_2\lambda = ... = L_k\lambda = L\lambda$.

Jestliže unifikátor λ existuje, nazývá se množina klauzulí unifikovatelnou.

Nejobecnějším unifikátorem (MGU – Most General Unifier) je pak takový unifikátor, z něhož lze každý jiný unifikátor odvodit.

Zobecněná rezoluční metoda:

Nechť C_1 a C_2 jsou klauzule, které nemají společné proměnné. Nechť C_1 je klauzule (množina literálů) obsahující unifikovatelnou podmnožinu literálů M_1, kterou unifikátor λ_1 unifikuje do jediného literálu L_1, nechť C_2 je klauzule (množina literálů) obsahující unifikovatelnou podmnožinu literálů M_2, kterou unifikátor λ_2 unifikuje do jediného literálu L_2 a nechť tyto literály tvoří komplementární pár ($L_1 = \neg L_2$).

Pak rezolventou klauzulí C_1 a C_2 je klauzule

$$(C_1\lambda_1 - M_1\lambda_1) \vee (C_2\lambda_2 - M_2\lambda_2).$$

Nesplnitelná množina klauzulí je rezolucí zamítnutelná, což znamená, že z této množiny lze vždy pomocí rezoluce odvodit prázdnou klauzuli.

Postup odvozování však může být značně složitý (jde o přístup podobný prohledávání stavového prostoru).

Lineární rezoluce používá v každém kroku předchozí rezolventu, a je proto jednodušší − bylo dokázáno, že nesplnitelná množina klauzulí je lineární rezolucí také zamítnutelná (tzv. lineárně zamítnutelná).

Příklad: Postup důkazu nesplnitelnosti množiny klauzulí:

1)	$\neg A(x) \vee B(x) \vee C(x, f(x))$		
2)	$\neg A(x) \vee B(x) \vee D(f(x))$		
3)	$E(a)$		
4)	$A(a)$		
5)	$\neg C(a, y) \vee E(y)$		
6)	$\neg E(x) \vee \neg B(x)$		
7)	$\neg E(x) \vee \neg D(x)$		
8)	$B(a) \vee C(a, f(a))$	rezolventa z 1, 4	substituce $\{x/a\}$
9)	$C(a, f(a)) \vee \neg E(a)$	rezolventa z 8, 6	substituce $\{x/a\}$
10)	$\neg E(a) \vee E(f(a))$	rezolventa z 9, 5	substituce $\{y/f(a)\}$
11)	$E(f(a))$	rezolventa z 10, 3	
12)	$\neg D(f(a))$	rezolventa z 11, 7	substituce $\{x/f(a)\}$
13)	$\neg A(a) \vee B(a)$	rezolventa z 12, 2	substituce $\{x/a\}$
14)	$B(a)$	rezolventa z 13, 4	
15)	$\neg E(a)$	rezolventa z 14, 6	substituce $\{x/a\}$
16)	$\square$	rezolventa z 15, 3	

Příklad: Odvození rezolventy klauzulí

$$P(x) \vee P(f(y)) \vee A(c),\ \neg P(v) \vee \neg P(f(w)) \vee A(c)$$

Nalezení nejobecnějšího unifikátoru literálů první klauzule:

- unifikovatelnou množinu tvoří literály $P(x) \vee P(f(y))$; substitucí $\{x/f(y)\}$ přecházejí v jediný literál $P(f(y))$

Nalezení nejobecnějšího unifikátoru literálů druhé klauzule:

- unifikovatelnou množinu tvoří literály $\neg P(v) \vee \neg P(f(w))$; substitucí $\{v/f(w)\}$ přecházejí v jediný literál $\neg P(f(w))$

Původní klauzule přecházejí na tvar

$$P(f(y)) \vee A(c),\ \neg P(f(w)) \vee A(c),$$

a jejich rezolventou po substituci $\{w/y\}$, resp. $\{y/w\}$ je formule A(c).

Následující algoritmus/procedura pro unifikaci dvou klauzulí/výrazů předpokládá zápis klauzulí ve tvaru seznamů, jejichž prvky jsou predikátové symboly následované termy (a pro termy/funkce pak platí stejná konvence zápisu):

Klauzule:	Odpovídající seznam:
P(a, b)	(P a b)
P(f(a), g(x, y))	(P (f a) (g x y))
P(x) $\vee$ Q(y)	((P x) $\vee$ (Q y))

Tento algoritmus buď nalezne nejobecnější unifikátor obou výrazů, nebo zjistí, že dané výrazy nejsou unifikovatelné.

Procedura Unify(E1, E2)
{unifikuje výrazy E1 a E2, vrací nejobecnější unifikátor, nebo Fail}

1. E1 i E2 jsou buď konstanty nebo prázdné seznamy:
   - Jestliže E1 = E2, vraťte { } (tj. prázdnou substituci), jinak vraťte Fail (výrazy nelze unifikovat).
2. E1 je proměnná:
   - Jestliže E1 je proměnnou v E2, vraťte Fail, jinak vraťte substituci {E1/E2}.
3. E2 je proměnná:
   - Jestliže E2 je proměnnou v E1, vraťte Fail, jinak vraťte substituci {E2/E1}.

4. Jinak:

4.1. Nechť HE1 je první prvek E1 a HE2 první prvek E2.

4.2. Volejte proceduru Unify(HE1, HE2).

4.3. Vrátí-li procedura volaná v bodě 4.2 Fail, vraťte také Fail, jinak pokračujte.

4.4. Nechť SUBST1 je vrácená substituce procedurou volanou v bodě 4.2.

4.5. Nechť TE1 je zbytek E1 (bez prvku HE1) a TE2 zbytek E2 (bez prvku HE2), s respektováním substituce SUBST1.

4.6. Volejte proceduru Unify(TE1, TE2).

4.7. Vrátí-li procedura volaná v bodě 4.6 Fail, vraťte také Fail, jinak pokračujte.

4.8. Nechť SUBST2 je vrácená substituce procedurou volanou v bodě 4.6.

4.9. Vraťte kompozici substitucí SUBST1 a SUBST2.

Příklad:

Unify((parents *x* (father *x*) (mother john)), (parents john (father john) *y*))
    Unify(parents, parents)
    return { }
    Unify((*x* (father *x*) (mother john)), (john (father john) *y*))
        Unify(*x*, john)
        return {*x*/john}
        Unify(((father john) (mother john)), ((father john) *y*))
            Unify((father john), (father john))
                Unify(father ,father)
                return { }
                Unify((john), (john))
                    Unify(john, john)
                    return { }
                    Unify(( ),( ))
                    return { }
                return { }
            return { }

Unify(((mother john)), (*y*))
Unify((mother john), *y*)
return {*y*/(mother john)}
Unify(( ),( ))
return { }
return {*y*/(mother john)}
return {*y*/(mother john)}
return {*x*/john, *y*/(mother john)}
return {*x*/john, *y*/(mother john)}

Původní klauzule:	(parents *x* (father *x*) (mother john))
	(parents john (father john) *y*)
Vrácená substituce:	{*x*/john, *y*/(mother john)}
Výsledné klauzule:	(parents john (father john) (mother john))
	(parents john (father john) (mother john))

Příklad: Z následující množiny vět

1. Bohatí a zdraví lidé jsou šťastní.
2. Lidé, kteří sportují, jsou zdraví.
3. Emil sportuje a je bohatý.
4. Šťastní lidé mají spokojený život.

dokažte tvrzení "Někdo má spokojený život".

Postup při důkazu:

1. Bohatí a zdraví lidé jsou šťastní.

$$\forall x((\text{Bohatý}(x) \wedge \text{Zdravý}(x)) \Rightarrow \text{Šťastný}(x)) \quad \Leftrightarrow$$
$$\neg(\text{Bohatý}(x) \wedge \text{Zdravý}(x)) \vee \text{Šťastný}(x) \quad \Leftrightarrow$$
$$\neg\text{Bohatý}(x) \vee \neg\text{Zdravý}(x) \vee \text{Šťastný}(x) \qquad (1)$$

2. Lidé, kteří sportují, jsou zdraví.

$$\forall y(\text{Sportuje}(y) \Rightarrow \text{Zdravý}(y)) \quad \Leftrightarrow$$
$$\neg\text{Sportuje}(y) \vee \text{Zdravý}(y) \qquad (2)$$

3. Emil sportuje a je bohatý.

   $\text{Sportuje(emil)} \wedge \text{Bohatý(emil)} \Leftrightarrow$
   Sportuje(emil) (3a)
   Bohatý(emil) (3b)

4. Šťastní lidé mají spokojený život.

   $\forall z(\text{Šťastný}(z) \Rightarrow \text{Spokojený\_život}(z)) \Leftrightarrow$
   $\neg\text{Šťastný}(z) \vee \text{Spokojený\_život}(z)$ (4)

Přidá se klauzule vytvořená negací dokazovaného tvrzení „Někdo má spokojený život“:

$\neg(\exists v(\text{Spokojený\_život}(v)) \Leftrightarrow$
$\forall v(\neg\text{Spokojený\_život}(v)) \Leftrightarrow$
$\neg\text{Spokojený\_život}(v)$ (5)

a dokáže se nesplnitelnost této (rozšířené) množiny:

¬Bohatý(x) ∨ ¬Zdravý(x) ∨ Šťastný(x)			(1)
¬Sportuje(y) ∨ Zdravý(y)			(2)
Sportuje(emil)			(3a)
Bohatý(emil)			(3b)
¬Šťastný(z) ∨ Spokojený_život(z)			(4)
¬Spokojený_život(v)			(5)
¬Šťastný(v)	r. 5, 4	s. {z/v}	(6)
¬Bohatý(v) ∨ ¬Zdravý(v)	r. 6, 1	s. {x/v}	(7)
¬Zdravý(emil)	r. 7, 3b	s. {v/emil}	(8)
¬Sportuje(emil)	r. 8, 2	s. {y/emil}	(9)
□	r. 9, 3a		(10)

Pozn.: zkratka „r.“ znamená „rezolventa z klauzulí“ a „s.“ „substituce“.

Příklad: Z následující množiny vět

1. Každý má rodiče.
2. Prarodič je rodič rodiče.

dokažte tvrzení "Karel má prarodiče”.

Postup při důkazu:

1. Každý má rodiče.

$\forall x_1 \exists y_1(\text{rodič}(x_1, y_1)) \Leftrightarrow$
$\forall x_1(\text{rodič}(x_1, \text{přímý\_předek}(x_1))) \Leftrightarrow$
$\text{rodič}(x_1, \text{přímý\_předek}(x_1))$ (1)

2. Prarodič je rodič rodiče.

$\forall x_2 \forall y_2 \forall z_2(\text{rodič}(x_2, y_2) \wedge \text{rodič}(y_2, z_2) \Rightarrow \text{prarodič}(x_2, z_2)) \Leftrightarrow$
$\forall x_2 \forall y_2 \forall z_2(\neg\text{rodič}(x_2, y_2) \vee \neg\text{rodič}(y_2, z_2) \vee \text{prarodič}(x_2, z_2)) \Leftrightarrow$
$\neg\text{rodič}(x_2, y_2) \vee \neg\text{rodič}(y_2, z_2) \vee \text{prarodič}(x_2, z_2)$ (2)

Přidá se klauzule vytvořená negací dokazovaného tvrzení "Karel má prarodiče”:

$\neg$prarodič(karel, x_3) (3)

a dokáže se nesplnitelnost této (rozšířené) množiny:

$\neg$rodič(karel, y_2) $\vee$ $\neg$rodič(y_2, x_3) (4)

r. 3, 2

s. {x_2/karel/, z_2/x_3}

$\neg$rodič(přímý_předek(karel), x_3) (5)

r. 4, 1

s. {x_1/karel, y_2/přímý_předek(karel)}

□ (6)

r. 5, 1

s. {přímý_předek(karel)/x_1, přímý_předek(přímý_předek(karel))/x_3}

Pozn.: zkratka „r.“ opět znamená „rezolventa z klauzulí“ a „s.“ „substituce“.

Plánování pomocí logiky (nalezení posloupnosti operátorů, kterými lze vyřešit zadanou úlohu)

Nejznámějším plánovacím systémem, původně určeným pro řešení úloh ze světa kostek, je systém STRIPS (STanford Research Institute Problem Solver). Tento systém k plánování používá:

- Databázi DB, která obsahuje aktuální stav řešeného problému popsaný pomocí predikátů.
- Zásobník, který obsahuje cílové predikáty i predikáty dílčích cílů.
- Operátory, které umožňují měnit stav DB. Každý operátor je přitom popsán třemi seznamy: Seznam COND obsahuje predikáty popisující použitelnost operátoru, seznam DEL obsahuje predikáty, které se při použití operátoru z DB odstraní a seznam ADD, který popisuje predikáty, které se při použití operátoru do DB přidají.
- Seznam/frontu použitých operátorů (tj. plán řešení úlohy).

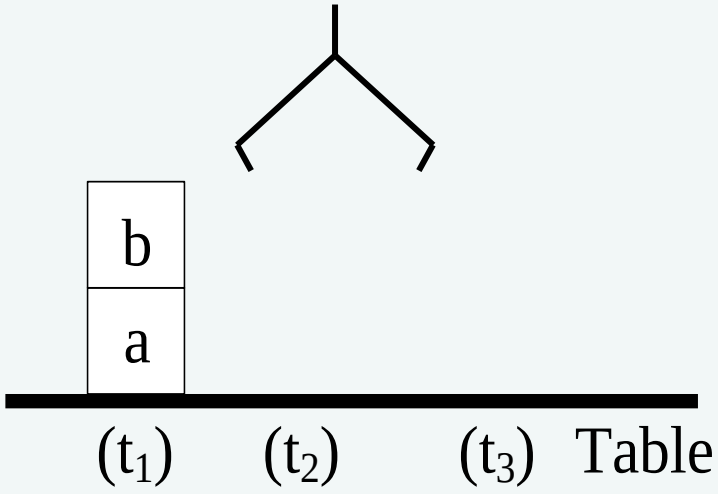
SRIPS - princip činnosti:

1. Aktuální stav řešeného problému se uloží do DB, konjunkce predikátů popisujících požadovaný cílový stav se uloží do zásobníku a vytvoří se prázdná fronta pro použité operátory
2. Pokud je na vrcholu zásobníku konjunkce predikátů, porovnává se postupně každý predikát s predikáty v DB. Pokud se některý predikát v aktuální DB nenachází, uloží se do zásobníku (nový vrchol zásobníku).
3. Pokud se predikát na vrcholu zásobníku v aktuální DB nachází, tak se pouze ze zásobníku odstraní.
4. Pokud predikát na vrcholu zásobníku v aktuální DB není, tak se hledá operace, která ho tam může uložit. Do zásobníku se pak kromě vlastního operátoru přidá i konjunkce predikátů COND popisujících podmínky použitelnosti tohoto operátoru.

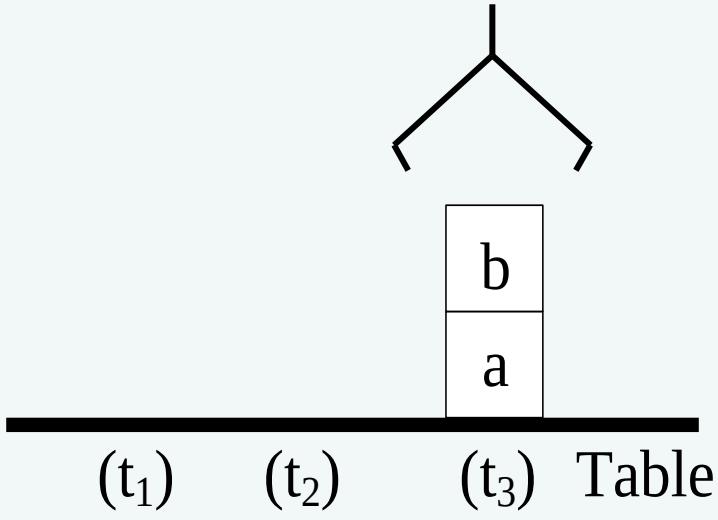
5. Pokud je na vrcholu zásobníku predikát operátoru pak se operace provede:
   - z DB se odstraní predikáty, které má operátor v seznamu DEL a do DB se naopak přidají predikáty, které má operátor v seznamu ADD.
   - Operátor se z vrcholu zásobníku odstraní a uloží se do fronty.
6. Pokud není zásobník prázdný, pak se postup opakuje od bodu 2, jinak je úloha vyřešena – fronta pak obsahuje pořadí operátorů – plán, kterým se úloha vyřeší.

Příklad na použití STRIPS:

Initial state:



Goal state:



Initial state: on(a, t_1), on(b, a), clear(b), clear(t_2), clear(t_3), armempty.

Goal: on(a, t_3), on(b, a).

Akce robota (zjednodušené)

pickup(X, Y):

COND:	$on(X, Y) \wedge clear(X) \wedge armempty$
DEL:	$on(X, Y) \wedge clear(X) \wedge armempty$
ADD:	$holding(X) \wedge clear(Y)$

puton(X, Y):

COND:	$holding(X) \wedge clear(Y)$
DEL:	$holding(X) \wedge clear(Y)$
ADD:	$on(X, Y) \wedge clear(X) \wedge armempty$

Akce robota v originálním systému

Stack(X, Y) / Unstack(X, Y) (X on/from a block Y)
Puton(X) / Pickup(X) (X on/from the table)

Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$on(a, t_1)$	
$on(a, t_3)$	$on(b, a)$	
$puton(a, t_3)$	$clear(b)$	
$holding(a) \wedge clear(t_3)$	$clear(t_2)$	
$holding(a)$	$clear(t_3)$	
$pickup(a, Y)$	$armempty$	
$on(a,Y) \wedge clear(a) \wedge armempty$		
$clear(a)$		
$pickup(X, a)$		
$on(X, a) \wedge clear(X) \wedge armempty$ $\|\{X/b\}$		

Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$on(a, t_1)$	pickup(b, a)
$on(a, t_3)$	$clear(t_2)$	
$puton(a, t_3)$	$clear(t_3)$	
$holding(a) \wedge clear(t_3)$	holding(b)	
holding(a)	clear(a)	
pickup(a, Y)		
$on(a,Y) \wedge clear(a) \wedge armempty$		
armempty		
puton(X,Y)		
$holding(X) \wedge clear(Y) \mid \{X/b, Y/t_2\}$		

Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$on(a, t_1)$	pickup(b, a)
$on(a, t_3)$	$clear(t_3)$	puton(b, t_2)
$puton(a, t_3)$	clear(a)	
$holding(a) \wedge clear(t_3)$	$on(b, t_2)$	
holding(a)	clear(b)	
pickup(a, Y)	armempty	
$on(a,Y) \wedge clear(a) \wedge armempty$		

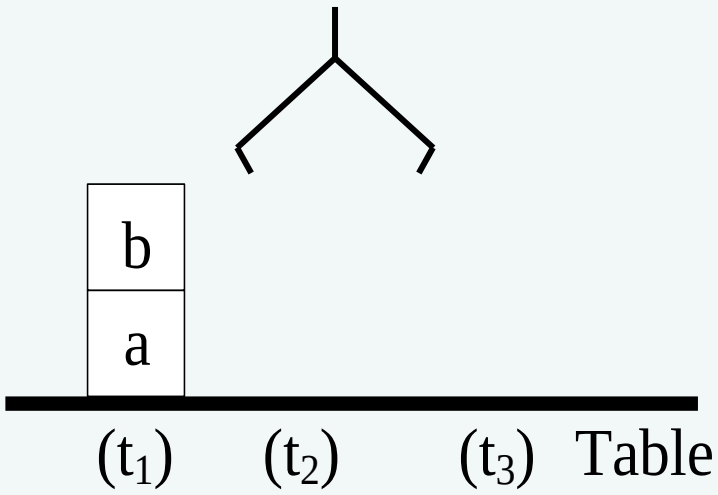
Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$clear(t_3)$	$pickup(b, a)$
$on(a, t_3)$	$on(b, t_2)$	$puton(b, t_2)$
$puton(a, t_3)$	$clear(b)$	$pickup(a, t_1)$
$holding(a) \wedge clear(t_3)$	$holding(a)$	
	$clear(t_1)$	

Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$on(b, t_2)$	pickup(b, a)
on(b, a)	clear(b)	puton(b, t_2)
puton(b, a)	clear(t_1)	pickup(a, t_1)
$holding(b) \wedge clear(a)$	on(a, t_3)	puton(a, t_3)
holding(b)	clear(a)	
pickup(b, Y)	armempty	
$on(b, Y) \wedge clear(b) \wedge armempty) \mid \{Y/t_2\}$		

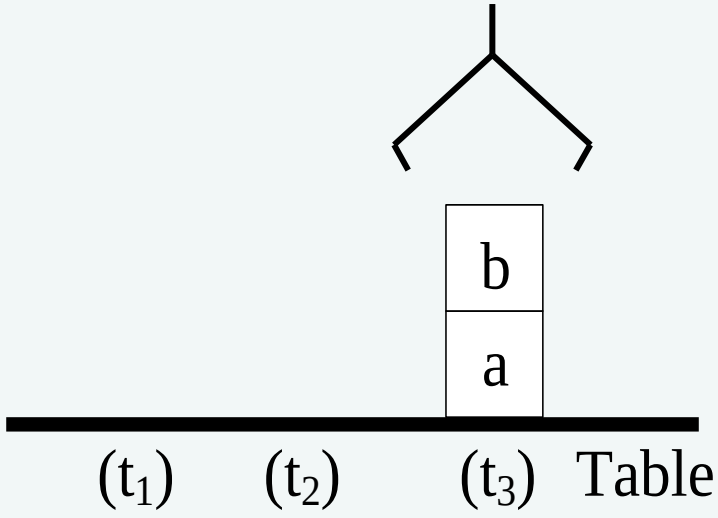
Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$clear(t_1)$	$pickup(b, a)$
$on(b, a)$	$on(a, t_3)$	$puton(b, t_2)$
$puton(b, a)$	$clear(a)$	$pickup(a, t_1)$
$holding(b) \wedge clear(a)$	$holding(b)$	$puton(a, t_3)$
	$clear(t_2)$	$pickup(b, t_2)$

Zásobník cílů:	DB:	Fronta operátorů:
$on(a, t_3) \wedge on(b, a)$	$clear(t_1)$	pickup(b, a)
	$on(a, t_3)$	puton(b, t_2)
	$clear(t_2)$	pickup(a, t_1)
	on(b, a)	puton(a, t_3)
	clear(b)	pickup(b, t_2)
	armempty	puton(b, a)

Initial state:



Goal state:



Plan (queue of operators):

1) pickup(b, a)
2) puton(b, t_2)
3) pickup(a, t_1)
4) puton(a, t_3)
5) pickup(b, t_2)
6) puton(b, a)