V1.1

Prof.dr.sc. Bojana Dalbelo Bašić

Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroelektroniku, računalne i inteligentne sustave

www.zemris.fer.hr/~bojana
bojana.dalbelo@fer.hr

UMJETNA INTELIGENCIJA

Zaključivanje uporabom predikatne logike (2)

ZAMJENA (SUPSTITUCIJA)

- Univerzalno kvantificirane varijable zamjenjivali smo konstantom.

Primjer

  - [1] $\forall x(\text{MUŽ}(x) \rightarrow \text{VOLI}(x, \text{ŽENA}(x)))$
  - [2] MUŽ(Marko)
  - [3] $(\text{MUŽ}(\text{Marko}) \rightarrow \text{VOLI}(\text{Marko}, \text{ŽENA}(\text{Marko})).$
- Općenito varijable mogu biti **zamijenjene (supstituirane)** čitavim **izrazima**

- Pojam <u>izraz u širem smislu</u> (*engl. expression*) označava: izraz (*engl. term*), atom, funkciju, predikat ili formulu
- **Supstitucija** je neophodna za primjenu rezolucijskog pravila

ZAMJENA (SUPSTITUCIJA)

Skup uređenih parova čini **zamjenu** (supstituciju)

α **= {t_1/y_1, t_2/y_2, ... , t_n/y_n,},**

gdje su t_i, $i = 1, 2, \ldots n$ izrazi u širem smislu,

y_i, $i = 1, 2, \ldots n$ su <u>različite</u> varijable, i vrijedi $t_i \neq y_i$

za bilo koji $i = 1, 2, \ldots n$.

- Kad se zamjena α primjeni na neki izraz K, tada se svako pojavljivanje y_i u K istodobno zamijeni sa t_i.
- Dobiveni izraz označavamo sa $K\alpha$ i kažemo da je $K\alpha$ **slučaj** ili primjer izraza K (*engl. instance*).
- **Prazna zamjena** ε je takva da je $K\varepsilon = K$

ZAMJENA (SUPSTITUCIJA)

Primjer:

- izraz GT(x, y),
  - supstitucija α = {prosinac/x, travanj/y},
  - rezultat Kα = GT(prosinac, travanj).

- izraz K = P(w, f(x), z),
  - supstitucija α = {a/x, g(u)/z},
  - rezultat Kα = P(w, f(a), g(u))

- Zamjena ili supstitucija u literaturi se još naziva i vezivanje *(engl. bindings)* u smislu da je varijabla vezana s vrijednošću kojom se zamjenjuje

KOMPOZICIJA ZAMJENA

- Neka su α i β dvije zamjene. **Kompozicija zamjena** je definirana kao nova zamjena $\alpha \circ \beta$ tako da je

  $K(\alpha \circ \beta) = (K\alpha)\beta$, za neki izraz K

- Uporaba zamjene $\alpha \circ \beta$ na izrazu K daje isti rezultat kao kad se prvo primjeni α na izrazu K, a onda se na tako dobiveni izraz primjeni β.

- Kompozicija zamjena $\alpha \circ \beta$ može biti izvedena na temelju sljedećeg <u>postupka u dva koraka</u>

KOMPOZICIJA ZAMJENA

- Neka su dane α i β zamjene

  $\alpha = \{t_1/y_1, t_2/y_2, \ldots, t_n/y_n\}$,

  $\beta = \{s_1/x_1, s_2/x_2, \ldots, s_m/x_m\}$

<u>1. korak</u>

- Iz α i β konstruiraj sljedeće skupove:
- $\lambda 1 = \{ t_1\beta/y_1, t_2\beta/y_2, \ldots, t_n\beta/y_n, s_1/x_1, s_2/x_2, \ldots, s_m/x_m\}$
- $\lambda 2 = \{ t_i\beta/y_i \mid t_i\beta/y_i \in \lambda 1$ i $t_i\beta = y_i$ za $1 \leq i \leq n\}$,
- $\lambda 3 = \{ s_i/x_i \mid s_i/x_i \in \lambda 1$ i $x_i \in \{ y_1, y_2, \ldots, y_n\}\}$;

<u>2. korak</u>

- Izvedi kompoziciju zamjena $\alpha \circ \beta$ uporabom formule

  $\boldsymbol{\alpha \circ \beta = \lambda 1 - \lambda 2 - \lambda 3}$

KOMPOZICIJA ZAMJENA - PRIMJER

Primjer

- Neka su zadane zamjene

  $\alpha = \{z/u, h(u)/w\}$

  $\beta = \{a/u, z/w, u/z\}$

- Izvedi kompoziciju ($\alpha \circ \beta$) na temelju postupka u dva koraka, a zatim pokaži da vrijedi $K(\alpha \circ \beta) = (K\alpha)\beta$ za izraz $K = P(u, w, f(z))$

<u>1. korak</u> – konstrukcija skupova $\lambda 1$, $\lambda 2$ i $\lambda 3$.

- $\lambda 1 = \{z\beta/u, h(u)\beta/w, a/u, z/w, u/z\}$

  $= \{u/u, h(a)/w, a/u, z/w, u/z\}$

- $\lambda 2 = \{u/u\}$
- $\lambda 3 = \{a/u, z/w\}$

<u>2. korak</u> – izvođenje kompozicije zamjena $\alpha \circ \beta$ uporabom formule

- $\alpha \circ \beta = \lambda 1 - \lambda 2 - \lambda 3 = \{h(a)/w, u/z\}$

KOMPOZICIJA ZAMJENA - PRIMJER

- Još treba pokazati da se isti rezultat dobije primjenom definicije zamjena tj. da je $K(\alpha \circ \beta) = (K\alpha)\beta$ za slučaj $K = P(u, w, f(z))$
- $(K\alpha)\beta = \mathbf{P(u, h(a), f(u))}$
- (i) prema definiciji:
  - $K\alpha = P(z, h(u), f(z))$
  - $(K\alpha)\beta = \mathbf{P(u, h(a), f(u))}$
- (ii) primjenom izvedene formule za $\alpha \circ \beta$:

  $$\alpha \circ \beta = \{h(a)/w, u/z\}$$

  $$K(\alpha \circ \beta) = \mathbf{P(u, h(a), f(u))}$$
- Dakle vrijedi $K(\alpha \circ \beta) = (K\alpha)\beta$

SVOJSTVA KOMPOZICIJE ZAMJENA

Asocijativnost

- $(\alpha \circ \beta) \circ \gamma = \alpha \circ (\beta \circ \gamma)$

ε je neutralni element s lijeva

- $\varepsilon \circ \alpha = \alpha$

ε je neutralni element s desna

- $\alpha \circ \varepsilon = \alpha$

Općenito ne vrijedi komutativnost

- $\alpha \circ \beta \neq \beta \circ \alpha$

UNIFIKACIJA

- Općeniti izrazi K1 i K2 mogu se svesti na isti oblik akko postoji supstitucija γ takva da vrijedi

$$\boxed{K_1 \gamma = K_2 \gamma}$$

- Supstitucija γ se naziva **unifikator** (*engl. unifier*)
- Izraz $K_i \gamma$, i =1, 2 naziva se **zajednički slučaj** ili primjer *(engl. common instance*)
- Kaže se da su opći izrazi K1 i K2 unificirani pomoću γ
- Izrazi mogu imati više unifikatora

UNIFIKACIJA - PRIMJER

- Atomi P(x) i P(y), gdje su x i y varijable, imaju unifikatore:
  - $\gamma_1 = \{b/x, b/y\}$ koji daje zajednički slučaj P(b) i
  - $\gamma_2 = \{z/x, z/y\}$ koji daje zajednički slučaj P(z).
- Ako je *b* konstanta, a *z* varijabla možemo reći da je P(z) općenitiji zajednički slučaj nego li je to P(b)
- Naime, P(b) je slučaj od P(z) uz supstituciju {b/z}, ali NE VRIJEDI OBRATNO. (**Zašto**?)
- Ako je u klauzuli P(z) istinito (podrazumijeva se da su sve varijable univerzalno kvantificirane), tada je i P(b) istinito (uporabom pravila univerzalne specifikacije).
- Ako je P(b) istinito, ne znači da je P(z) istinito – ne možemo slučaj zamijeniti varijablom!

NAJOPĆENITIJI UNIFIKATOR

- Za gornji primjer kažemo da je unifikator $\gamma 2$ **općenitiji** od unfikatora $\gamma 1$ za atome P(x) i P(y). Naime, tada postoji zamjena $\gamma 3$ takva da je

$$(P\ \gamma\mathbf{2})\ \gamma\mathbf{3} = P\ \gamma\mathbf{1}$$

- P(x) i P(y) mogu imati više unifikatora – zanima nas najopćenitiji unifikator
- Unifikator δ se naziva **najopćenitiji unifikator** akko za svaki unifikator γ od K_1 i K_2, zajednički primjer $K_i\delta$ je općenitiji od zajedničkog primjera $K_i\gamma$
- Izraz $K_i\delta$ se naziva **najopćenitiji zajednički slučaj**
- $K_i\gamma$ je primjer od $K_i\delta$

NAJOPĆENITIJI UNIFIKATOR

δ je **najopćenitiji unifikator** akko za svaki unifikator γ od K_i, $i=1,2$ postoji zamjena θ takva da je

$$K_i \gamma = (K_i \delta)\, \theta = K_i (\delta \circ \theta)$$

tj. $\gamma = \delta \circ \theta$.

- MGU se može tumačiti kao da ima za posljedicu najmanje moguće promjene da bi se unificirali izrazi
- Pretpostavnimo da tijekom postupka nalaženja zajedničkog primjera dva izraza postoji izbor zamijeniti varijablu konstantom ili zamijeniti varijablu drugom varijablom. Procedura nalaženja MGU će uvijek izabrati ovo drugo
- Dva izraza mogu imati više najopćenitijih unifikatora – u tom su slučaju svi najopćenitiji zajednički izrazi jednako općeniti

ALGORITAM ZA UNIFIKACIJU IZRAZA

- U literaturi postoji više poznatih algoritama za nalaženje najopćenitijeg unifikatora konačnog skupa izraza koji se mogu unificirati i koji dojavljuju pogrešku ako se izrazi ne mogu unificirati
- Razmotrit ćemo rekurzivni algoritam MGUNIFIER (Luger, Stubblefield, 1993; Shinghal, 1992)
- Algoritam koristi pojmove:
  - glava općenitog izraza
  - rep općenitog izraza
- Općeniti izraz možemo napisati kao **listu** gdje su elementi liste odvojeni prazninama, a svaki element liste je opet moguća lista nazvana podizraz

ALGORITAM ZA UNIFIKACIJU IZRAZA

Primjer

- P(x, y, f(b)) može biti napisano kao lista (P x y (f b)) gdje su element liste P, x, y, (f b). Zadnji element liste je i sam lista, sastavljen od elemenata f i b
- **Glava liste** (P x y (f b)) je P, dok je **rep liste** (x y (f b))
- Sintaksa lista je sintaksa programskog jezika LISP

Sintaksa FOPL	***Sintaksa liste***
P(a, b)	(P a b)
P(f(a), g(x,y))	(P (f a) (g x y))
EQUAL(Eva, MAJKA(Iva))	(EQUAL Eva (MAJKA Iva))
P(x) ∧ Q(y)	((P x) ∧ (Q y))

REKURZIVNI ALGORITAM ZA NALAŽENJE NAJOPĆENITIJEG ZAJEDNIČKOG UNIFIKATORA

procedura **MGUNIFIER (K_1, K_2)**

ako ili K_1 ili K_2 simbolizira konstantu, varijablu, funkciju ili predikat **tada**:

- **ako** su K_1 i K_2 identični, **tada** vrati praznu supstituciju {};
- **ako** K_1 predstavlja varijablu **tada**:
  - **ako** se K_1 pojavljuje u K_2 **tada** vrati *pogreška*;
  - **inače** vrati $\{K_2 / K_1\}$;
- **ako** K_2 predstavlja varijablu **tada**:
  - **ako** se K_2 pojavljuje u K_1 **tada** vrati *pogreška*;
  - **inače** vrati $\{K_1 / K_2\}$;
- **ako** niti K_1 niti K_2 ne predstavlja varijablu **tada** vrati *pogreška*;

α := MGUNIFIER(glava od K_1 , glava od K_2);

ako α = *pogreška* **tada** vrati *pogreška*;

...

REKURZIVNI ALGORITAM ZA NALAŽENJE NAJOPĆENITIJEG ZAJEDNIČKOG UNIFIKATORA

… *(nastavak)*

K_3 := rezultat primjene zamjene α na rep od K_1;

K_4 := rezultat primjene zamjene α na rep od K_2;

β := MGUNIFIER(K_3, K_4);

ako β = *pogreška* **tada** vrati *pogreška*;

vrati kompoziciju zamjena $\alpha \circ \beta$;

REKURZIVNI ALGORITAM ZA NALAŽENJE NAJOPĆENITIJEG ZAJEDNIČKOG UNIFIKATORA

- Procedura MGUNIFIER vraća pogrešku u slučajevima kada unifikacija nije moguća:
  - varijabla treba biti zamjenjena izrazom koji ju sadrži. Provjerava se pojavljuje li se varijabla u izrazu kojim se zamjenjuje. Takva zamjena vodila bi beskonačnoj petlji.
  - kada bi trebalo zamijeniti predikat ili funkciju – što je u suprotnosti s definicijom supstitucije

PRIMJER

- Dani su literali

  P(x, x) i P(f(z), z).

- MGUNIFIER će prvo naći supstituciju {f(z)/x} koja daje

  P(f(z), f(z)) i P(f(z), z).

- <u>Kada ne bi bilo provjere uvjeta</u>, sljedeća supstitucija {f(z)/z}

  bi dala P(f(f(z)), f(f(z))) i P(f(f(z)), f(z))

  i tako u beskonačnost

PRIMJER

- Primjeri izraza koji se ne mogu unificirati

K1	**K2**	**Uzrok pogrešci kod unifikacije**
P(x)	P(f(x))	Vidi prethodni primjer (beskonačna petlja)
P(f(x))	P(x)	
P(x)	Q(x)	Predikat se ne može zamijeniti drugim predikatom.
P(f(x))	P(g(x))	Funkcija se ne može zamijeniti drugom funkcijom.

PRIMJER

- Nađi MGU izraza:

  K1 = P(g(u), z, f(z)) i K2 = P(x,y,f(b))

{g(u)/x} unificira prve podizraze od K1 i K2 koji se ne slažu

K1{g(u)/x} = P(g(u), z, f(z))

K2{g(u)/x} = P(**g(u)**, y, f(b))

{y/z} unificira sljedeće izraze koji se ne slažu

Kompozicija {g(u)/x} ○ {y/z} = {g(u)/x, y/z}

K1{g(u)/x, y/z} = P(**g(u), y**, f(y))

K2{g(u)/x, y/z} = P(**g(u), y** ,f(b))

{b/y} unificira sljedeće izraze koji se ne slažu

Kompozicija {g(u)/x, y/z} ○ { b/y} = {g(u)/x, b/z, b/y}

je **MGU**, koji označavamo s δ.

- Primjenom mgu δ = {g(u)/x, b/z, b/y} na K1 i K2 dobivamo najopćenitiji zajednički primjer

  P(g(u), b, f(b))

PRIMJER

- Ako za neka 2 izraza MGU nije jedinstven, tj. postoji više najopćenitijih unifikatora, odgovarajući najopćenitiji zajednički izrazi su **alfabetske varijante** – razlikuju se samo po *imenima* varijabli
- *Primjer*

Skup izraza	Najopćenitiji zajednički primjer
{P(x), P(a)}	P(a)
{P(f(x), y, g(y)), P(f(x), z, g(x)}	P(f(x), x, g(x))
{P(f(x, g(a,y)), g(a, y)), P(f(x,z), z)}	P(f(x, g(a, y)), g(a, y))

UNIFIKACIJA LITERALA

Definicija

- Dva literala mogu se **unificirati** akko
  1. oba označavaju negirane atome ili oba označavaju afirmativne atome
  2. ti atomi se mogu unificirati.

Primjer

P(x) i P(y) ili ~P(x) i ~P(y)

Definicija

- Dva literala mogu se **komplementarno unificirati** akko
  1. jedan od njih je negirani atom, a drugi je afirmativni atom
  2. ti se atomi mogu unificirati.

Primjer

P(x) i ~P(y) ili ~P(x) i P(y)

REZOLUCIJA

- Rezolucija u FOPL slična je rezoluciji u propozicijskoj logici
- Neka su L_k, $k = 1, \ldots, i$ i M_l, $l = 1, \ldots, j$ literali (neki negirani, a neki afirmativni)
- Neka su dane dvije klauzule

  (I1) $L_1 \vee L_2 \vee \ldots \vee L_i$

  (I2) $M_1 \vee M_2 \vee \ldots \vee M_j$
- Prije primjene rezolucje klauzule trebaju biti standardizirane (preimenovati varijable)
- Pretpostavimo da u (I1) i (I2) postoje literali koji se mogu komplemenarno unificirati. Pretpostavimo (bez gubitka općenitosti – vrijedi komutativnost) da su to literali L_1 i M_1
- Pretpostavimo da se unifikacija L_1 i M_1 može obaviti sa **MGU** δ.

REZOLUCIJA

- Sada možemo razriješiti *roditeljske klauzule* (I1) i (I2) po L_1 i M_1 i izvesti *resolventnu klauzulu*

$$(I3) \qquad \underbrace{L_2\delta \vee L_3\delta \vee \ldots\ldots \vee L_i\delta}_{\text{Ostatak (I1)}} \vee \underbrace{M_2\delta \vee M_3\delta \vee \ldots\ldots \vee M_j\delta}_{\text{Ostatak (I2)}}$$

- Literali u resolventi dobiju se primjenom $\vee$ na uniju literala u roditeljskim klauzulama izuzevši literale sa kojima se razrješavanje obavlja

Napomena

- Razrješavanjem dviju jediničnih klauzula izvodi se prazna klauzula *NIL*

PRIMJER REZOLUCIJE

Nađi resolventu za sljedeće klauzule

- $P(g(y), x, f(z)) \lor Q(z, b) \lor R(x)$
- $S(x, y) \lor \sim P(x, y, f(a))$
- Klauzule nisu standadizirane! (simboli za varijable x i y pojavljuju se u obje klauzule)

Potrebno je preimenovati varijable u prvoj klauzuli:

x postaje varijabla w,

y postaje varijabla u.

Standardizirane klauzule:

- **(I1) $P(g(u), w, f(z)) \lor Q(z, b) \lor R(w)$**
- **(I2) $S(x, y) \lor \sim P(x, y, f(a))$**

Nađi mgu!

PRIMJER REZOLUCIJE

- (I1) $P(g(u), y, f(a)) \lor Q(a, b) \lor R(y)$
- (I2) $S(g(u), y) \lor \sim P(g(u), y, f(a))$

- Razrješavanjem po $P(g(u), y, f(a))$ i $\sim P(g(u), y, f(a))$ dobiva se resolventna klauzula:

(I3) $S(g(u), y) \lor Q(a, b) \lor R(y)$

Rezolucijsko pravilo je ispravno (zdravo)

- Prema pravilu univerzalne specijalizacije $l_1\delta$ je logička posljedica od l_1. Slično, $l_2\delta$ je logička posljedica od l_2
- Zaključivanjem istim kao i u propozicijskoj logici zaključujemo da je l_3 je logička posljedica $l_1\delta$ **i** $l_2\delta$.
- Dakle **l_3 je logička posljedica l_1 i l_2** te je **rezolucijsko pravilo u predikatnoj logici je ispravno**.
- Literali L_1 i M_1 su unificirani pomoću mgu pa je resolventa l_3 u najopćenitijem mogućem obliku

Rezolucija opovrgavanjem je potpuna

- Kao i u propozicijskoj logici tako i u predikatnoj logici vrijedi da je rezolucija opovrgavanjem potpuna. (Robinson, 1965)
- Formula G je logička posljedica premisa F1, F2, .. , Fn akko možemo pokazati da je ulazni skup F1, F2, …, Fn, ~G nekonzistentan izvođenjem resolventne klauzule NIL.
- Svojstvo da je rezolucija zdrava i da je rezolucija opovrgavanjem potpuna osigurava da možemo uvijek dokazati teorem G, ako je G teorem.
- Međutim ako G nije teorem postupak dokazivanja teorema rezolucijom opovrgavanjem može nikad ne završiti – u tom je smislu predikatna logika **poluodlučljiva**

TEMELJNE KLAUZULE I ODLUČLJIVOST

- Literali koji sadrže samo konstante (bez varijabli) nazivaju se **temeljni literali** (*engl. ground literals*), a njihova disjunkcija naziva se **temeljna klauzula** (*engl. ground clause*)

- Podrazred predikatne logike koji dopušta samo pojavljivanje konstanti, funkcija i predikata (a ne varijabli) je **ODLUČLJIV**

- Ta potklasa ima istu moć zaključivanja kao i propozicijska logika

PRIMJER

Primjer rezolucije opovrgavanjem u podklasi predikatne logike koja ne dozvoljava pojavljivanje varijabli.

- Za isti ovaj primjer, cilj "Majka je zadovoljna" dokazali smo u propozicijskoj logici:
  1. prirodnim zaključivanjem i
  2. rezolucijom opovrgavanjem.

[i] Ivan se probudio

[ii] Ivan nosi pribor za čišćenje

[iii] Majka je zadovoljna ako se Ivan probudi i čisti svoju sobu

[iv] Ako Ivan nosi pribor za čišćenje, tada on čisti svoju sobu

- Dokažite rezolucijom opovrgavanjem cilj: *Majka je zadovoljna*

PRIMJER

- Ulazni skup tvore premise i negacija cilja pretvorene u klauzalnu formu

[I1] BUDAN(Ivan)

[I2] NOSI(Ivan, pribor)

[I3] ~BUDAN(Ivan) ∨ ~ČISTI(Ivan, soba) ∨ ZADOVOLJNA(majka)

[I4] ~NOSI(Ivan, pribor) ∨ ČISTI (Ivan, soba)

[I5] ~ZADOVOLJNA(majka) **CILJ**

- Iz ulaznog skupa izvodimo slijedeće resolvente:

[I3] ~BUDAN(Ivan) ∨ ~ČISTI(Ivan, soba) ∨ ZADOVOLJNA(majka)

[I5] ~ZADOVOLJNA(majka)

[I6] ~BUDAN(Ivan) ∨ ~ČISTI(Ivan, soba)

PRIMJER

[I6] ~BUDAN(Ivan) ∨ ~ČISTI(Ivan, soba)

[I1] BUDAN(Ivan)

    [I7] ~ČISTI(Ivan, soba)

[I4] ~NOSI(Ivan, pribor) ∨ ČISTI (Ivan, soba)

    [I8] ~NOSI(Ivan, pribor)

[I2] NOSI(Ivan, pribor)

    [I9] *NIL*

VAŽNOST FAKTORIZIRANJA KLAUZULA U PREDIKATNOJ LOGICI

- Slično kao i u propozicijskoj logici, klauzule u predikatnoj logici moraju biti faktorizirane kako bi se sačuvala potpunost postupka opovrgavanjem

Primjer kako rezolucija gubi svoju potpunost ako nema faktorizacije:

[I1] $P(u) \vee P(w)$

[I2] $\sim P(x) \vee \sim P(y)$

- Uz uporabu mgu = $\{u/x\}$ i razrješavanjem po $P(u)$ i $\sim P(u)$ dobivamo resolventu

[I3] $P(w) \vee \sim P(y)$

(Razrješavanjem po drugim literalima dobivamo alfabetske varijante od I3)

FAKTORIZACIJA U PREDIKATNOJ LOGICI

- Klauzula u predikatnoj logici može biti faktorizirana akko sadrži literale koji se mogu unificirati

[I] $L1 \vee L2 \vee L3 \vee \ldots \vee Ln$

- Pretpostavimo (bez gubitka općenitosti) da se literali L1 i L2 mogu unificirati sa MGU δ. Gornja klauzula tada može biti faktorizirana dajući klauzulu I’

[I’] $L2\delta \vee L3\delta \vee \ldots \vee Ln\delta$

- Klauzula I’ se naziva **faktor klauzula** od I
- Ako skup literala ima MGU δ tada je $I\delta$ faktor klauzula od I, gdje su višestruka pojavljivanja literala u $I\delta$ zamijenjena jednim pojavljivanjem literala

PRIMJER

Primjer:

[I1] $P(u) \vee P(w)$

[I2] $\sim P(x) \vee \sim P(y)$

- Uz faktorizaciju:
- Faktoriziramo I1 uporabom mgu={w/u} dobivamo

[I1’] P(w)

- Faktoriziramo I2 uporabom mgu={y/x} dobivamo

[I2’] ~P(y)

- Razrješavanjem jediničnih klauzula P(w) i ~P(y) uz uporabu mgu = {w/y} izvodimo praznu klauzulu

[I3’] NIL

PRIMJER

Primjer

$$P(x, y, f(b)) \vee S(x, y) \vee P(g(u), w, f(z))$$

gdje su: P i S - predikatini simboli,

f i g - funkcijski simboli,

x, y, u, w, z – varijable,

b - konstanta.

- Prvi i treći literal mogu se unificirati pomoću mgu

  $\delta = \{g(u)/x, y/w, b/z\}$. Faktor klauzula je:

$$P(g(u), y, f(b)) \vee S(g(u), y)$$

FAKTORIZACIJA U PREDIKATNOJ LOGICI

- Faktor koji se sastoji samo od jednog literala naziva se **jedinični faktor**.
- Od sada pa na dalje će se smatrati da je resolventa I3 izvedena iz kluazula I1 i I2 akko dobivamo I3 na bilo koji od sljedećih načina:
  1) razrješavanjem I1 i I2
  2) razrješavanjem I1 i faktora od I2
  3) razrješavanjem faktora I1 i I2
  4) razrješavajnjem faktora I1 i faktora I2

REZOLUCIJSKI POSTUPAK OPOVRGAVANJEM (ALGORITAM)

Da bi se dokazalo da je ciljna formula G deduktivna posljedica od F1, F2, …, Fn, potrebno je primijeniti sljedeće korake:

1. Pretvori F1, F2, F3, …, Fn, ~G u **klauzalni oblik**.
2. Izaberi iz skupa klauzula dvije klauzule koje su razrješive. Ako je potrebno, standardiziraj te klauzule. **Izvedi resolventu i dodaj je skupu klauzula**. Ponavljaj ovaj korak sve dok se ne ispuni jedan od uvjeta:
   - Izvedena je prazna klauzula *NIL*. (Dokazano je da je cilj G teorem, ulazni skup je nekonzistentan.)
   - Niti jedan par klauzula ne može se razriješiti ili ne može se izvesti niti jedna nova klauzula (Dokazano je da cilj G nije teorem)
   - Neki unaprijed zadani resursi računala su iscrpljeni. (Nema odluke o G – predikatna logika je poluodlučljiva)

PRIMJER

Primjer

[1] ∀x(MUŽ(x) → VOLI(x, žena(x)))

[2] MUŽ(Marko)

Dokaži uporabom rezolucije opovrgavanjem:

VOLI(Marko, žena(Marko)).

Premise i negacija ciljne formule pretvore se u klauzalni oblik

[I1] ~MUŽ(x) ∨ VOLI(x, žena(x))

[I2] MUŽ(Marko)

[I3] ~VOLI(Marko, žena(Marko))

Iz ulaznog skupa rezolucijom izvodimo:

[I4] ~MUŽ(Marko) - uporabom **mgu**={Marko/x} iz [I1] i [I3].

[I5] *NIL* - uporabom {} iz [I2] i [I4].

Time je dokazano da je VOLI(Marko, žena(Marko)) teorem

PRIMJER

Primjer

- Robot dostavlja pakete.
- Robot zna da su svi paketi u sobi 27 manji od svakog paketa u sobi 28.
- A i B su paketi.
- Paket A je u sobi 27 ili u sobi 28, ali robot ne zna u kojoj.
- Paket B je u sobi 27 i nije manji od paketa A.

Uporabom rezolucije opovrgavanjem pokaži kako robot može zaključiti da je paket A u sobi 27.

- **KONSTANTE**

  27, 28, A, B

- **PREDIKATI**

  PAKET(x) - x je paket - skraćeno P(x)

  U_SOBI(x, y) - x je u sobi y - skraćeno U(x, y)

  MANJI(x, y) - x je manji od y - M(x,y)

PRIMJER

- **BAZA ZNANJA**

$\forall x\ \forall y\ \big((P(x) \land P(y) \land U(x, 27) \land U(y, 28)) \rightarrow M(x, y) \big)$

$P(A)$

$P(B)$

$U(A, 27) \lor U(A, 28)$

$U(B, 27) \land \sim M(B, A)$

PRIMJER

- **BAZA ZNANJA**

  $\forall x\ \forall y\ (\ P(x) \land P(y) \land U(x, 27) \land U(y, 28)\ ) \rightarrow M(x, y)$

  $P(A)$

  $P(B)$

  $U(A, 27) \lor U(A, 28)$

  $U(B, 27) \land \sim M(B, A)$

- **BAZA ZNANJA u klauzalnoj formi**

  $\sim P(x) \lor \sim P(y) \lor \sim U(x, 27) \lor \sim U(y, 28) \lor M(x, y)$

  $P(A)$

  $P(B)$

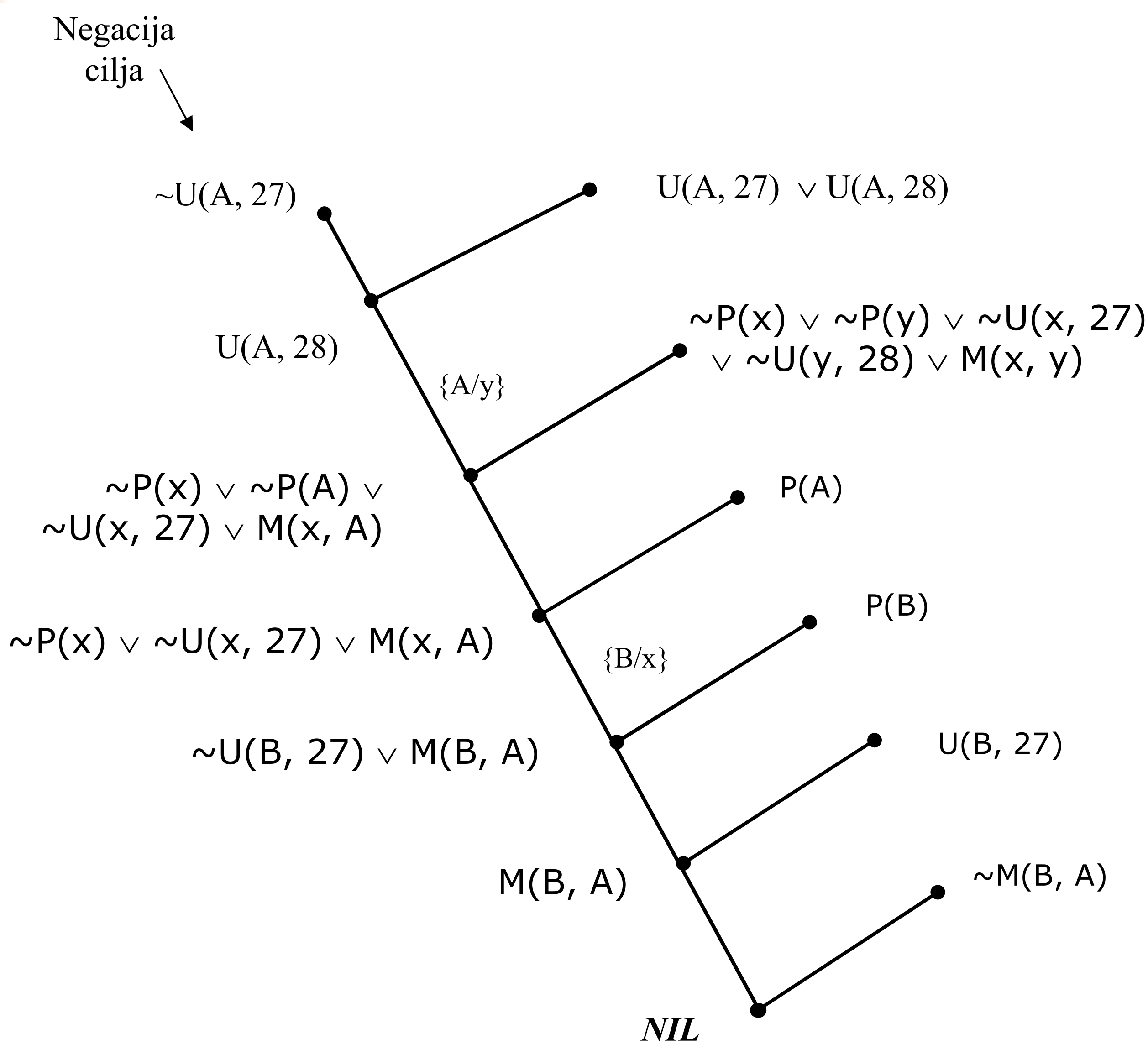
  $U(A, 27) \lor U(A, 28)$

  $U(B, 27)$

  $\sim M(B, A)$

  $\sim U(A, 27)$ (negacija cilja)

STABLO DOKAZA



VRSTE REZOLUCIJA

- **Binarna rezolucija** je kombiniranje dvije klauzule koje sadrže komplementarne literale

Primjer

$Q(x) \vee \sim P(x, a)$

$\sim Q(b) \vee R(x)$

Uz unifikaciju {b/x} resolventna klauzula je

$\sim P(b, a) \vee R(b)$

- **Rezolucija s rezultirajućom jediničnom klauzulom** (engl. *unit resulting resolution*)

  Istodobno razrješavanje više klauzula kako bi se izvela jedinična klauzula. Sve roditeljske klauzule osim jedne su jedinične, a ta ima točno jedan literal više od ukupnog broja jediničnih klauzula

VRSTE REZOLUCIJA

Primjer

VJENČANI(Ana, Marko)

~OTAC(Marko, Ivan)

~VJENČANI(x, y) ∨ ~MAJKA(x, z) ∨ OTAC(y, z)

(uz supstituciju {Ana/x, Marko/y, Ivan/z} resolventa je jedinična klauzula)

~MAJKA(Ana, Ivan)

- **Linearna rezolucija**

  Ako je uvijek jedna od roditeljskih klauzula izvedena u prethodnom koraku

- **Linearna rezolucija na ulaznom skupu**

  Ako je jedna od roditeljskih klauzula uvijek iz izvornog ulaznog skupa klauzula

STRATEGIJE REZOLUCIJE

Unifikacija + rezolucija → **automatsko zaključivanje**

ALI

- Rezolucija je nedjelotvorna bez daljnjih razrada!
- Razrješavanje slučajno odabranih klauzula → kombinatorna eksplozija
- Važna uporaba metoda koje ograničavaju pretraživanje

Odabir redoslijeda razrješavanja klauzula kako bi **postupak rezolucije bio djelotvorniji** zove se **STRATEGIJA**

STRATEGIJE REZOLUCIJE

1. **Uređajne strategije** (engl. *ordering strategies*) – određuju slijed (poredak) kojim će se klauzule razrješavati

2. **Strategije skraćivanja** (engl. *pruning strategies*) – uklanjanje iz skupa klauzula onih klauzula i literala koji nisu neophodni za dokaz

3. **Strategije ograničavanja** (engl. *restriction strategies*) – propisuju primjenu rezolucije samo na one klauzule koje se smatraju vitalne za dokaz

STRATEGIJA SKUPA POTPORE

- Jedna od najvažnijih strategija (iz skupa strategija ograničavanja)
- Neka je S kontradiktoran skup klauzula i neka je T podskup od S. Tada je T **skup potpore** od S ako je S – T konzistentan.
- Rezolucija skupa potpore je rezolucija u kojoj nikad nisu obje klauzule iz S – T. Za svaku resolventu barem je jedna roditeljska klauzula iz T.
- Ako je S konzistentan, tada je svaki skup koji sadrži negaciju cilja kontradiktoran i može biti skup potpore.
  - pretpostavka: baza znanja je konzistentna, jer inače nam ni dokazivanje cilja ne znači puno
  - dodatna prednost – stabla dokaza su razumljiva jer su usmjerena cilju

EKSTRAKCIJA ODGOVORA POMOĆU REZOLUCIJE OPOVRGAVANJEM

- Ako je neka formula oblika $\exists$**xG(x)** logička posljedica nekih premisa tada to možemo dokazati rezolucijom opovrgavanjem. *Za koje vrijednosti x je G(x) logička posljedica premisa?*

- Rezolucijom opovrgavanjem možemo i više od toga tj. odgovoriti na pitanje:

> *Za koje vrijednosti x*
> *je G(x) logička posljedica premisa?*

EKSTRAKCIJA ODGOVORA POMOĆU REZOLUCIJE OPOVRGAVANJEM

Primjer:

- Pas Fido je svagdje gdje je njegov gazda Ivan.
- Ivan je u knjižnici.

Gdje je Fido?

Aksiomi:

$\forall x$ JE_U(Ivan, x) $\rightarrow$ JE_U(Fido, x)

JE_U(Ivan, knjižnica)

Klauzalna forma:

~JE_U(Ivan, x) $\vee$ JE_U(Fido, x)

JE_U(Ivan, knjižnica)

Hipoteza:

$\exists z$ JE_U(Fido, z) (z?)

Negacija hipoteze:

$\forall z$ ~JE_U(Fido, z) …… (Fido je nigdje)

~JE_U(Fido, z) (klauzalna forma negacije hipoteze)

EKSTRAKCIJA ODGOVORA POMOĆU REZOLUCIJE OPOVRGAVANJEM

~JE_U(Fido, z)   ~JE_U(Ivan, x) ∨ JE_U(Fido, x)

{x/z}

~JE_U(Ivan, x)   JE_U(Ivan, knjižnica)

{knjižnica/x}

NIL

EKSTRAKCIJA ODGOVORA

- Supstitucije pod kojima je nađena kontradikcija jesu supstitucije pod kojima je pretpostavka (hipoteza) istinita.
- Čuvanje informacije o unifikacijama tijekom dokaza omogućava odgovor na upit

Odgovor:

hipoteza	supstitucije		odgovor na pitanje
∃z JE_U(Fido, z)	{x/z, knjižnica/x}	=	**JE_U(Fido, knjižnica)**

Napomena: Nađemo li kompoziciju ovih dviju suspstitucija:

α ={x/z} i β ={knjižnica/x} (postupkom u dva koraka) dobit ćemo $\alpha \circ \beta$ = {knjižnica/z, knjižnica/x}

Primjer

1. Ankica je mama od Branke	MAJKA(Ankica, Branka)
2. Za sve x i y vrijedi: *Ako je x kćerka od y onda je y majka od x*	$\forall x \forall y$ (KĆERKA(x, y) $\rightarrow$ MAJKA(y, x))
3. Zorica je kćerka od Branke.	KĆERKA (Zorica, Branka)
4. Za sve x, y, z vrijedi: *Ako je x majka od y, i y majka od z, tada je x baka od z.*	$\forall x \forall y \forall z$(MAJKA(x, y) $\wedge$ MAJKA(y, z)) $\rightarrow$ BAKA(x, z))

Dokaži rezolucijom opovrgavanjem: Zorica ima baku tj.

$\exists$v(BAKA(v, Zorica))

PRIMJER

Premise u klauzalnom obliku + negacija cilja:

[I1] MAJKA(Ankica, Branka)

[I2] ~KĆERKA(u, w) ∨ MAJKA(w, u)

[I3] KĆERKA (Zorica, Branka)

[I4] ~MAJKA(x, y) ∨ ~MAJKA(y, z) ∨ BAKA(x, z)

[I5] ~BAKA(v, Zorica)

Izvodimo

[I6] ~MAJKA(x, y) ∨ ~MAJKA(y, Zorica)	mgu={x/v, Zorica/z}, [I4] i [I5]
[I7] ~KĆERKA(Zorica, y) ∨ ~MAJKA(x, y)	mgu={y/w, Zorica/u}, [I2] i [I6]
[I8] ~MAJKA (x, Branka)	mgu={Branka/y} [I3] i [I7]
[I9] **NIL**	mgu={Ankica/x} [I1] i [I8]

PRIMJER

- Uporaba rezolucije opovrgavanjem za odgovor na pitanje iz skupa premisa [I1]-[I4]

Tko je Zoričina baka?

- Ne zanima nas samo odgovor postoji li *v* tako da je *v* baka od Zorice tj. $\exists v$ (BAKA(v, Zorica)), nego nas zanima vrijednost varijable *v*

Postupak:

- Svaka klauzula koja se dobije kao negacija cilja pretvara se u tautologiju. To se radi tako da klauzuli dodajemo negaciju svakog literala koji ona sadrži. (Ako je G(x) negacija cilja, tada je $G(x) \vee \sim G(x)$ tautologija)

PRIMJER

[J1] MAJKA(Ankica, Branka)
[J2] ~KĆERKA(u, w) ∨ MAJKA(w, u)
[J3] KĆERKA (Zorica, Branka)
[J4] ~MAJKA(x, y) ∨ ~MAJKA(y, z) ∨ BAKA(x, z)
[J5] ~BAKA(v, Zorica) ∨ BAKA(v, Zorica)

- Sada ponavljamo potpuno isti postupak kao u dokazivanju formule ∃v(BAKA(v, Zorica)).

- Rezultat: umjesto *NIL* - odgovor na pitanje *tko je Zoričina baka*

PRIMJER

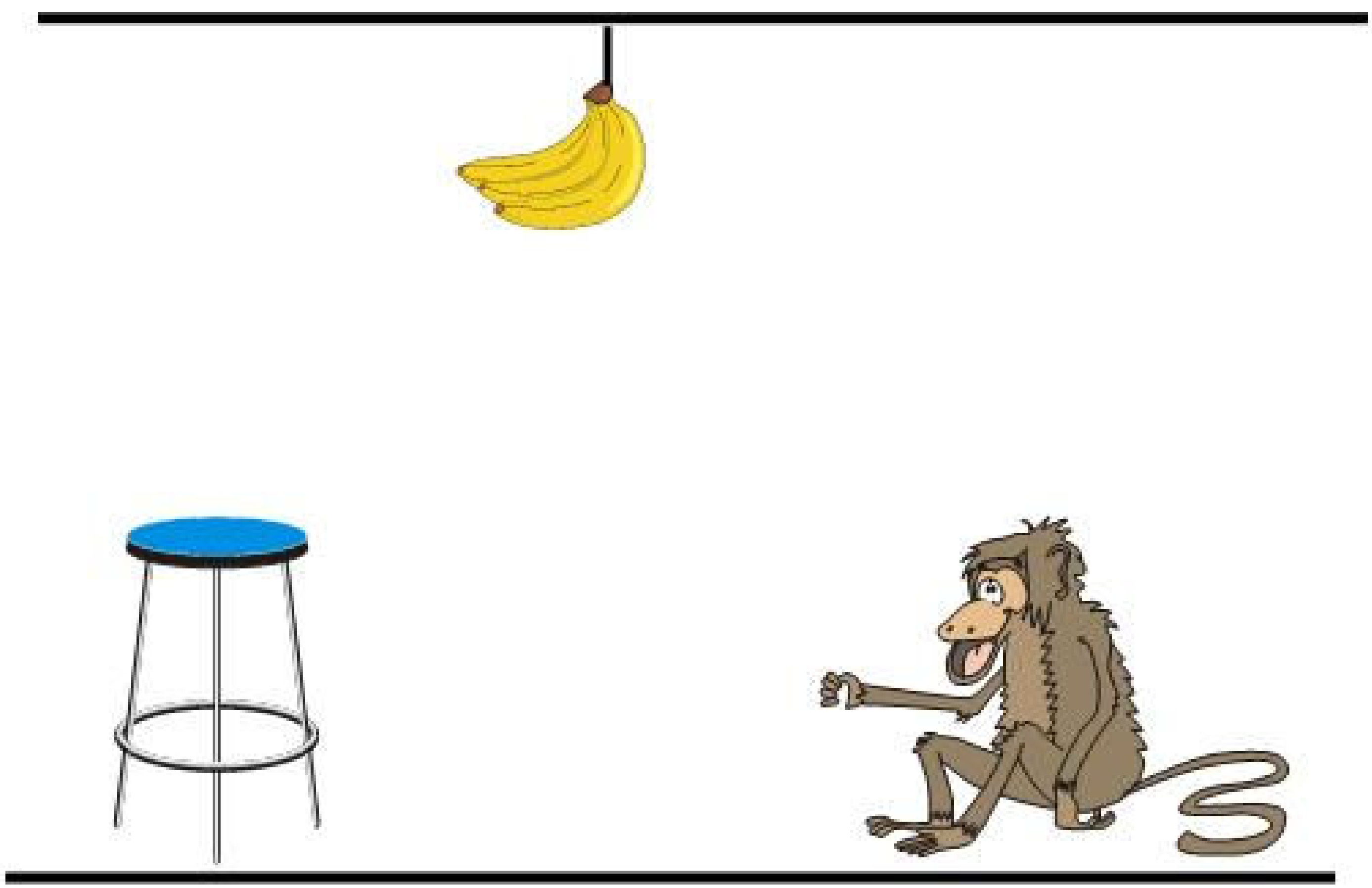
1. ~MAJKA(x, y) ∨ ~MAJKA(y, Zorica) ∨ BAKA(x, Zorica)	mgu={x/v, Zorica/z}, [J4] i [J5]
2. ~KĆERKA(Zorica, y) ∨ ~MAJKA(x, y) ∨ BAKA(x, Zorica)	mgu={y/w, Zorica/u}, [J2] i [J6]
3. ~MAJKA (x, Branka) ∨ BAKA(x, Zorica)	mgu={Branka/y} [J3] i [J7]
4. BAKA(Ankica, Zorica)	mgu={Ankica/x} [J1] i [J8]

Ankica je Zoričina baka

Primjer: Majmun i banane

PRIMJER REZOLUCIJE

- Majmun i banane



PRIMJER REZOLUCIJE

U sobi se nalaze:

- majmun,
- stolica,
- banane koje vise sa sredine stropa, ali na visini koja nije na dohvat ruke majmunu.

Ako je majmun dovoljno bistar, on može:

- postaviti stolicu ispod snopa banana,
- popesti se na stolicu i
- dohvatiti banane.

Zadatak:

1. Uporabom FOPL prikaži činjenice iz svijeta “majmun-banane”
2. Uporabom rezolucije opovrgavanjem dokaži da majmun može dohvatiti banane

PRIMJER REZOLUCIJE

- U kreiranju baze znanja važno je odrediti:
  - sve relevantne objekte (majmun, banane, stolica, pod)
  - odnose među njima (npr. banane nisu blizu poda, stolica se može pomaknuti ispod banana).
- Sve nevažno treba izostaviti (npr. prozori, vrata…)

KONSTANTE

pod, stolica, banana, majmun

VARIJABLE

x, y, z

PREDIKATI

MOŽE_DOHVATITI(x, y)	*x može dohvatiti y*
SPRETAN(x)	*x je spretan*
BLIZU(x, y)	*x je blizu y*
JE_NA(x, y)	*x je na y*

PRIMJER REZOLUCIJE

ISPOD(x, y)	*x je ispod y*
VISOKA(x)	*x je visok(a)*
U_SOBI(x)	*x je u sobi*
MOŽE_POMAKNUTI_BLIZU(x, y, z)	*x može pomaknuti y blizu z*
MOŽE_POPETI_NA(x, y)	*x se može popeti na y*

AKSIOMI BAZE ZNANJA

U_SOBI(banane)
U_SOBI(stolica)
U_SOBI(majmun)
VISOKA(stolica)
SPRETAN(majmun)
MOŽE_POMAKNUTI_BLIZU(majmun, stolica, banane)
MOŽE_POPETI_NA(majmun, stolica)
~BLIZU(banane, pod)

PRIMJER REZOLUCIJE

U_SOBI(x) ∧ U_SOBI(y) ∧ U_SOBI(z) ∧ MOŽE_POMAKNUTI_BLIZU(x, y, z) → BLIZU(z, pod) ∨ ISPOD(y, z)

MOŽE_POPETI_NA(x, y) → JE_NA(x, y)

JE_NA(x, y) ∧ ISPOD(y, banane) ∧ VISOK(y) → BLIZU(x, banane)

SPRETAN(x) ∧ BLIZU(x, y) → MOŽE_DOHVATITI(x, y)

- **Aksiomi se mogu pretvoriti u klauzalnu formu (uporabom De Morganovih zakona i ekvivalencije P → Q ≡ ~P ∨ Q)**

PRIMJER REZOLUCIJE

Klauzalni oblik baze znanja

1. U_SOBI(banane)
2. U_SOBI(stolica)
3. U_SOBI(majmun)
4. VISOKA(stolica)
5. SPRETAN(majmun)
6. MOŽE_POMAKNUTI_BLIZU(majmun, stolica, banane)
7. MOŽE_POPETI_NA(majmun, stolica)
8. ~BLIZU(banane, pod)
9. ~MOŽE_POPETI_NA(x, y) ∨ JE_NA(x, y)
10. ~SPRETAN(x) ∨ ~BLIZU(x, y) ∨ MOŽE_DOHVATITI(x, y)
11. ~JE_NA(x, y) ∨ ~ISPOD(y, banane) ∨ ~VISOK(y) ∨ BLIZU(x, banane)
12. ~U_SOBI(x) ∨ ~U_SOBI(y) ∨ ~U_SOBI(z) ∨ ~MOŽE_POMAKNUTI_BLIZU(x, y, z) ∨ BLIZU(z, pod) ∨ ISPOD(y, z)
13. ~ MOŽE_DOHVATITI(majmun, banane) (negacija cilja)

PRIMJER REZOLUCIJE

Dokaz rezolucijom

14. ~MOŽE_POMAKNUTI_BLIZU(majmun, stolica, banane) ∨ BLIZU(banane, pod) ∨ ISPOD(stolica, banane)

    (resolventa od [1], [2], [3] i [12] uz supstituciju {majmun/x, stolica/y, banane/z})

15. BLIZU(banane, pod) ∨ ISPOD(stolica, banane)

    (resolventa od [6], [14])

16. ISPOD(stolica, banane) *([8] i [15])*

17. ~JE_NA(x, stolica) ∨ ~VISOK(stolica) ∨ BLIZU(x,banane)

    (supstitucija {stolica/y} i [16] i [11])

18. ~JE_NA(x, stolica) ∨ BLIZU(x, banane) *([4] i [17])*

PRIMJER REZOLUCIJE

19. ~JE_NA(majmun, stolica) *([7] i [9])*
20. BLIZU(majmun, banane)
    ([18] i [19] supstitucija {majmun/x})
21. ~BLIZU(majmun, y) ∨ MOŽE_DOHVATITI(majmun, y)
    ([10] i [5] supstitucija {majmun/x})
22. MOŽE_DOHVATITI(majmun, y)
    ([20] i [21] supstitucija {banane/y})
23. **{}** *([13] i [22])*

PRIMJER REZOLUCIJE

- Ovaj dokaz nije vođen niti jednom posebnom strategijom iako se vodilo računa o odabiru roditeljskih klauzula. U suprotnom, puno nepotrebnih koraka može biti učinjeno.

Zadatak

- *Možete li komentirati koji su svi oblici rezolucije učinjeni kroz korake od [14] do [23]?*