

Logică computațională

Curs 9

Lector dr. Pop Andreea-Diana

Metoda tabelelor semantice în calculul predicatelor

- introdusă de Smullyan
- se bazează pe considerații semantice
- încearcă să construiască modelele unei formule date (FND)
- $\vdash U$ prin respingere, $\neg U$ nu are modele
- ideea:
 - descompunerea formulei inițiale în subformule
 - până la nivel de literali

Clase de formule (1)

- clase α - formule de tip conjunctiv

$$A \wedge B$$

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

$$\neg (A \rightarrow B)$$

- clase β - formule de tip disjunctiv

$$A \vee B$$

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

$$A \rightarrow B$$

Clase de formule (2)

- clase γ - formule cuantificate universale
- clase δ - formule cuantificate existențiale

$$(\forall x) A(x)$$

$$(\exists x) A(x)$$

$$\neg (\exists x) A(x)$$

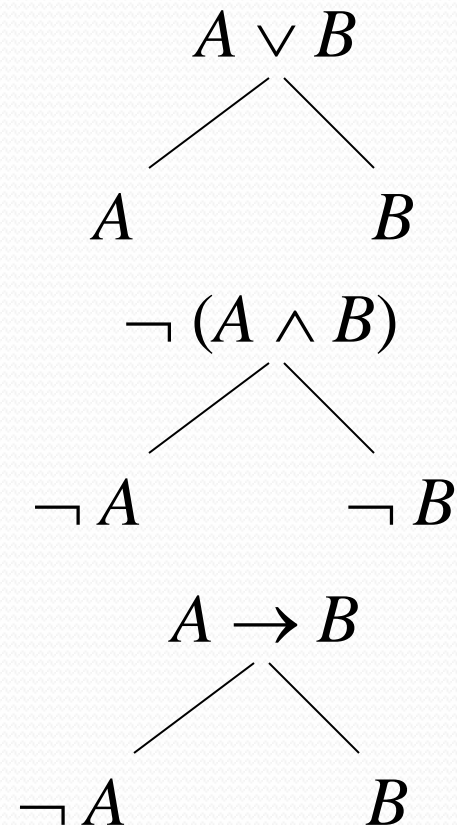
$$\neg (\forall x) A(x)$$

Reguli de descompunere a formulelor (1)

- regula α

$A \wedge B$	$\neg (A \vee B)$	$\neg (A \rightarrow B)$
/	/	/
A	$\neg A$	A
/	/	/
B	$\neg B$	$\neg B$

- regula β



Reguli de descompunere a formulelor (2)

$(\forall x) A(x)$ • regula γ

/ c_1, c_2, \dots, c_n – toate constantele existente pe ramură

$A(c_1)$ $\neg (\exists x) A(x)$

/

/

$A(c_2)$

$\neg A(c_1)$

|

/

\vdots

$\neg A(c_2)$

|

|

$A(c_n)$

\vdots

|

$\neg A(c_n)$

$(\forall x) A(x)$

|

copie formulă $\neg (\exists x) A(x)$ – copie formulă

• regula δ

$(\exists x) A(x)$

/ a – constantă nou introdusă

$A(a)$

$\neg (\forall x) A(x)$

/

$\neg A(a)$

Arborele binar de descompunere a unei formule

Având o formulă U , ei i se poate asocia o tabelă semantică, care este de fapt un arbore binar ce conține în nodurile sale formule și se construiește astfel:

- rădăcina arborelui este etichetată cu formula U ;
- fiecare ramură a arborelui care conține o formulă va fi extinsă cu subarborele corespunzător regulii de descompunere care se aplică formulei;
- extinderea unei ramuri se *încheie* în două situații:
 - a) dacă pe ramură apare o formulă și negația sa;
 - b) dacă au fost descompuse toate formulele de pe acea ramură sau prin aplicarea regulilor de descompunere nu se mai obțin formule noi pe acea ramură

Tipuri de ramuri

- O *ramură* a tabelii se numește *închisă* (simbolizată prin \otimes) dacă ea conține o formulă și negația ei, în caz contrar, dacă este completă, *ramura* se numește *deschisă* (simbolizată prin \odot).
- O *ramură* a tabelii se numește *completă* dacă ea este fie *închisă*, fie *toate formulele* de pe acea ramură au fost *descompuse*.

Tipuri de tabele semantice

- O *tabelă* se numește *închisă* dacă toate ramurile sale sunt închise. Dacă o tabelă are cel puțin o ramură deschisă, atunci ea se numește *deschisă*.
- O *tabelă* se numește *completă* dacă toate ramurile ei sunt complete.

Observații:

- Procesul de construire a unei tabele semantice este unul *nedeterminist* deoarece regulile de descompunere se pot aplica în orice ordine și la un moment dat se pot alege mai multe ramuri pentru extindere. Astfel unei formule i se pot asocia mai multe tabele semantice, dar acestea sunt echivalente.
- Pentru a obține tabele semantice *cât mai simple* (mai puțin ramificate) se recomandă:
 - utilizarea regulilor de tip α înaintea regulilor de tip β care realizează o ramificare;
 - utilizarea regulilor de tip δ (care introduc constante noi) înaintea regulilor de tip γ care utilizează toate constantele de pe ramura respectivă;

Observații (2):

- formulele de pe aceeași ramură a unei tablele semantice sunt *legate* între ele prin conectiva logică \wedge , iar *ramificarea* corespunde conectivei logice \vee .
- tabela semantică asociată unei formule propoziționale este o reprezentare grafică a *forme* sale *normale disjunctive*. Fiecare ramură reprezintă un *cub* (conjuncția tuturor literalilor de pe acea ramură), iar arborele este *disjuncția* tuturor *ramurilor* sale.
- Unei formule *consistente* i se asociază o *tabelă completă deschisă*, iar fiecare *ramură deschisă* a tablei furnizează cel puțin un *model* pentru formula respectivă.
- O *tabelă semantică închisă* asociată unei formule indică faptul că formula este *inconsistentă*, adică nu există nicio interpretare în care formula să fie adevărată

Teorema de corectitudine și completitudine a metodei tabelelor semantice

- O formulă U este teoremă (tautologie) dacă și numai dacă există o tabelă semantică închisă pentru formula $\neg U$.

Teoremă

- $U_1, U_2, \dots, U_n \vdash Y$ (echivalent cu $U_1, U_2, \dots, U_n \models Y$) dacă și numai dacă există o tabelă semantică închisă pentru formula $U_1 \wedge U_2 \wedge \dots \wedge U_n \wedge \neg Y$.

Exemple

$$\not\models^? (\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)$$



$$\neg((\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)) (1) \checkmark$$

$\propto (1)$

$$(\exists x) (A(x) \wedge B(x)) (2) \checkmark$$

$$\neg((\exists x) A(x) \wedge (\exists x) B(x)) (3) \checkmark$$

$\delta (2), a - \text{constantă nouă}$

$$A(a) \wedge B(a) (4) \checkmark$$

$\propto (4)$

$$A(a)$$

$$B(a)$$

$\beta (3)$

$$\neg(\exists x) A(x) (5) \checkmark$$

$$\neg(\exists x) B(x) (6) \checkmark$$

$$\neg A(a)$$

$\gamma (5), a - \text{constantă existentă}$

$$\neg B(a)$$

$\gamma (6), a - \text{constantă existentă}$

$$\neg(\exists x) A(x) (5') (\text{copia})$$

$$\neg(\exists x) B(x) (6') (\text{copia})$$

\otimes

TCC

\otimes

Deci tabela semntică este închisă \Rightarrow formula este tautologie

Example

$$\stackrel{?}{\neq} (\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)$$

$$\stackrel{?}{\neq} (\forall x) (A(x) \vee B(x)) \rightarrow (\forall x) A(x) \vee (\forall x) B(x)$$



$$\neg((\forall x) (A(x) \vee B(x)) \rightarrow (\forall x) A(x) \vee (\forall x) B(x)) (1)\checkmark$$

$\infty (1)$

$$(\forall x) (A(x) \vee B(x)) (2)\checkmark$$

$$\neg((\forall x) A(x) \vee (\forall x) B(x)) (3)\checkmark$$

$\infty (3)$

$$\neg(\forall x) A(x) (4)\checkmark$$

$$\neg(\forall x) B(x) (5)\checkmark$$

$\delta (4)$, a- constantă nouă

$$\neg A(a)$$

$\delta (5)$, b - constantă nouă

$$\neg B(b)$$

$\gamma (2)$, a,b- constante existente

$$A(a) \vee B(a) (6)\checkmark$$

$$A(b) \vee B(b) (7)\checkmark$$

$$(\forall x) (A(x) \vee B(x)) (2') \beta (6)$$

$$A(a)$$

\otimes

$$B(a)$$

$$A(b)$$

$\beta (7)$

$$B(b)$$

\otimes

TCC

\odot

Deci tabela semntică este deschisă \Rightarrow formula nu este tautologie

Example

$$\not\models^? (\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)$$

$$\not\models^? (\forall x) (A(x) \vee B(x)) \rightarrow (\forall x) A(x) \vee (\forall x) B(x)$$

$$\not\models^? (\exists y) (\forall x) P(x, y)$$



$$\neg(\exists y) (\forall x) P(x, y) (1)\checkmark$$

$\gamma (1)$, a - constantă implicită

$$\neg (\forall x) P(x, a) (2)\checkmark$$

$$\neg(\exists y) (\forall x) P(x, y) (1')\checkmark$$

$\delta (2)$, b - constantă nouă

$$\neg P(b, a)$$

$\gamma (1')$, b - constantă existentă

$$\neg (\forall x) P(x, b) (3)\checkmark$$

$$\neg(\exists y) (\forall x) P(x, y) (1'')$$

$\delta (3)$, c - constantă nouă

$$\neg P(c, b)$$

$\gamma (1'')$, c - constantă existentă

...

Deci am intrat în ciclu infinit, deci nu putem decide tipul formulei (pp. nu are loc – identificăm un anti-model)

Semi-decidabilitatea calcului predicativ

- Pentru cazul logicii predicatelor de ordinul I, arborele poate fi infinit datorită combinării regulilor de tip γ și δ .
- Dacă arborele asociat negației unei formule predicative este *finit*, atunci *se poate* decide dacă formula respectivă este o tautologie sau nu, dar dacă arborele este *infinit*, *nu* se poate decide nimic asupra validității formulei.

Substituții

Definiție: O *substituție* este o funcție definită pe mulțimea variabilelor, Var cu valori în mulțimea termenilor, $TERM$.

Se notează cu $\theta = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k]$, reprezentând o mulțime finită de înlocuiri de variabile cu termeni. x_1, \dots, x_k sunt variabile distincte, iar t_1, \dots, t_k sunt termeni, astfel încât $\forall i = 1, \dots, k, t_i \neq x_i$ și x_i nu este *subtermen* al lui t_i .

- $dom(\theta) = \{x_1, \dots, x_k\}$ se numește domeniul substituției θ .
- ε – substituția vidă
- $\varphi, \xi, \psi, \eta, \theta, \lambda$

Aplicarea substituției

$\theta = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k]$ asupra formulei U se definește recursiv:

- $\theta(x_i) = t_i, x_i \in \text{dom}(\theta); \theta(x) = x, x \notin \text{dom}(\theta);$
- $\theta(c) = c, c - \text{constantă};$
- $\theta(f(t_1, \dots, t_n)) = f(\theta(t_1), \dots, \theta(t_n)), f \in \mathcal{F}_n;$
- $\theta(P(t_1, \dots, t_n)) = P(\theta(t_1), \dots, \theta(t_n)), P \in \mathcal{P}_n;$
- $\theta(\neg U) = \neg \theta(U);$
- $\theta(U \wedge V) = \theta(U) \wedge \theta(V);$
- $\theta(U \vee V) = \theta(U) \vee \theta(V);$
- $\theta(U \rightarrow V) = \theta(U) \rightarrow \theta(V);$
- $\theta(U \leftrightarrow V) = \theta(U) \leftrightarrow \theta(V).$

Compunerea substituțiilor

$$\theta_1 = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k] \text{ și } \theta_2 = [y_1 \leftarrow s_1, \dots, y_k \leftarrow s_k]$$

$$\theta = \theta_1 \circ \theta_2 = [x_i \leftarrow \theta_2(t_i) \mid x_i \in \text{dom}(\theta_1), x_i \neq \theta_2(t_i)] \cup [y_i \leftarrow s_j \mid y_i \in \text{dom}(\theta_2) \setminus \text{dom}(\theta_1)]$$

- Obs.: Nu întotdeauna compunerea unor substituții este o substituție.

Example

$$\theta_1 = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k] \text{ și } \theta_2 = [y_1 \leftarrow s_1, \dots, y_k \leftarrow s_k]$$

$$\theta = \theta_1 \circ \theta_2 = [x_i \leftarrow \theta_2(t_i) \mid x_i \in \text{dom}(\theta_1), x_i \neq \theta_2(t_i)] \cup \\ [y_i \leftarrow s_j \mid y_i \in \text{dom}(\theta_2) \setminus \text{dom}(\theta_1)]$$

$$\theta_1 = [x \leftarrow y, z \leftarrow a, u \leftarrow f(y)] \text{ și } \theta_2 = [y \leftarrow a, t \leftarrow g(b), v \leftarrow w]$$

$$\theta = \theta_1 \circ \theta_2 = [x \leftarrow a, z \leftarrow a, u \leftarrow f(a), y \leftarrow a, t \leftarrow g(b), v \leftarrow w]$$

$$\theta_3 = [x \leftarrow y, z \leftarrow a, u \leftarrow f(t)] \text{ și } \theta_4 = [y \leftarrow a, t \leftarrow g(u), v \leftarrow w]$$

$$\theta = \theta_3 \circ \theta_4 =$$

$$= [x \leftarrow a, z \leftarrow a, u \leftarrow f(g(u)), y \leftarrow a, t \leftarrow g(u), v \leftarrow w] - \text{nu este} \\ \text{o substituție}$$

Proprietăți ale operației de compunere

- Element neutru: ε – substituția vidă:

$$\varepsilon \theta = \theta \varepsilon = \theta, \forall \theta - \text{substituție}$$

- Asociativitatea: $\theta_1(\theta_2 \theta_3) = (\theta_1 \theta_2) \theta_3 = \theta_1 \theta_2 \theta_3$
- În general compunerea nu este comutativă

Unificatori

- O substituție θ se numește *unificator* al termenilor t_1 și t_2 dacă $\theta(t_1) = \theta(t_2)$. Termenul $\theta(t_1)$ se numește *instanța comună* a termenilor unificați.
- Un *unificator al mulțimii* de formule $\{U_1, U_2, \dots, U_n\}$ este o substituție θ cu proprietatea: $\theta(U_1) = \dots = \theta(U_n)$.
- *Cel mai general unificator (mgu)* este un unificator μ cu proprietatea că orice alt unificator θ se obține din compunerea lui μ cu o altă substituție λ : $\theta = \mu \lambda$.

Algoritm pentru determinarea celui mai general unificator a doi literali (1)

Date de intrare: $l_1 = P_1(t_{1_1}, t_{1_2}, \dots, t_{1_n})$ și $l_2 = P_2(t_{2_1}, t_{2_2}, \dots, t_{2_k})$ doi literali

Date de ieșire: $mgu(l_1, l_2)$ sau “ l_1, l_2 nu sunt unificabili”

dacă $(P_1 \neq P_2)$ // simbolurile predicative sunt diferite

atunci *scrie* “ l_1, l_2 nu sunt unificabili”; STOP;

sf_dacă

dacă $(n \neq k)$ // aritate diferită pentru același simbol predicativ

atunci *scrie* “ l_1, l_2 nu sunt unificabili”; STOP;

sf_dacă

$\theta \leftarrow \varepsilon$; // inițializare cu substituția vidă

Algoritm pentru determinarea celui mai general unificator a doi literali (2)

cât_timp ($\theta(l_1) \neq \theta(l_2)$)

Din $\theta(l_1), \theta(l_2)$ se determină cele mai din stânga simbol de funcție, constantă sau variabilă diferite și notăm cu t_1 și t_2 termenii lor corespunzători.

dacă (*niciunul dintre t_1 și t_2 nu este variabilă sau unul este subtermenul celuilalt*)

atunci *scrie “ l_1, l_2 nu sunt unificabili”*; STOP;

sf_dacă

dacă (t_1 *este variabilă*)

atunci $\lambda = [t_1 \leftarrow t_2]$;

altfel $\lambda = [t_2 \leftarrow t_1]$;

sf_dacă

$\theta \leftarrow \theta \lambda$;

dacă (θ *nu este substituție*)

atunci *scrie “ l_1, l_2 nu sunt unificabili”*; STOP;

sf_dacă

sf_cât_timp

scrie “ l_1 și l_2 sunt unificabili, $mgu(l_1, l_2) =$ ” θ

Sf_algoritm

Exerciții

- ~~$P(a, x, g(f(y)))$ și $Q(f(y), f(z), g(z))$~~ Nu au același simbol de
predicat
- ~~$P(a, x, g(f(y)), b)$ și $P(f(y), f(z), g(z))$~~ Nu au aceeași aritate
- ~~$P(a, x, g(f(y)))$ și $P(f(y), f(z), g(z))$~~ [~~$a \leftarrow f(y)$~~]
- $P(x, g(f(a)), x)$ și $P(f(y), z, h(y, f(y)))$



$$A_1 = P(x, g(f(a)), x) \text{ și } A_2 = P(f(y), z, h(y, f(y)))$$

$$\theta_1 = [x \leftarrow f(y)]$$

$$\theta_1(A_1) = P(f(y), g(f(a)), f(y))$$

$$\theta_1(A_2) = P(f(y), z, h(y, f(y)))$$

$$\theta_2 = [z \leftarrow g(f(a))]$$

$$\theta_2(\theta_1(A_1)) = P(f(y), g(f(a)), \textcolor{red}{f}(y))$$

$$\theta_2(\theta_1(A_2)) = P(f(y), g(f(a)), \textcolor{red}{h}(y, f(y)))$$

Nu sunt unificabili pentru că nu avem variabilă care să se substituie (f și h sunt ambele simboluri de funcții)

Exerciții

- ~~$P(a, x, g(f(y)))$ și $Q(f(y), f(z), g(z))$~~ Nu au același simbol de
predicat
- ~~$P(a, x, g(f(y)), b)$ și $P(f(y), f(z), g(z))$~~ Nu au aceeași aritate
- ~~$P(a, x, g(f(y)))$ și $P(f(y), f(z), g(z))$~~ [~~$a \leftarrow f(y)$~~]
- ~~$P(x, g(f(a)), x)$ și $P(f(y), z, h(y, f(y)))$~~
- $P(a, h(x, u), f(g(y)))$ și $P(z, h(z, u), f(u))$



$$A_3 = P(a, h(x, u), f(g(y))) \text{ și } A_4 = P(z, h(z, u), f(u))$$

$$\theta_1 = [z \leftarrow a]$$

$$\theta_1(A_3) = P(a, h(x, u), f(g(y)))$$

$$\theta_1(A_4) = P(a, h(a, u), f(u))$$

$$\theta_2 = [x \leftarrow a]$$

$$\theta_2(\theta_1(A_3)) = P(a, h(a, u), f(g(y)))$$

$$\theta_2(\theta_1(A_4)) = P(a, h(a, u), f(u))$$

$$\theta_3 = [u \leftarrow g(y)]$$

$$\theta_3(\theta_2(\theta_1(A_3))) = P(a, h(a, g(y)), f(g(y)))$$

$$\theta_3(\theta_2(\theta_1(A_4))) = P(a, h(a, g(y)), f(g(y)))$$

Deci $\theta_3(\theta_2(\theta_1(A_3))) = \theta_3(\theta_2(\theta_1(A_4)))$, deci A_3 și A_4 sunt unificabile și

$$mgu(A_3, A_4) = \theta_1 \circ \theta_2 \circ \theta_3 = [z \leftarrow a, x \leftarrow a, u \leftarrow g(y)]$$