Curs 2

Exemplu de program logic

$$
\begin{aligned}
\texttt{oslo} &\rightarrow \texttt{windy} \\
\texttt{oslo} &\rightarrow \texttt{norway} \\
\texttt{norway} &\rightarrow \texttt{cold} \\
\texttt{cold} \wedge \texttt{windy} &\rightarrow \texttt{winterIsComing} \\
& \ \texttt{oslo}
\end{aligned}
$$

Exemplu de întrebare

Este adevărat `winterIsComing`?

Putem să testăm în SWI-Prolog

Program:

```
windy :- oslo.
norway :- oslo.
cold :- norway.
winterIsComing :- windy, cold.
oslo.
```

Intrebare:

```
?- winterIsComing.
true
```

Vom prezenta teoria care stă la baza acestui program!

Cuprins

Logica propozițională PL

Logica propozițională PL

- O propoziție este un enunț care poate fi adevărat (1) sau fals (0).
- Propozițiile sunt notate simbolic (φ, ψ, χ, $\cdots$) și sunt combinate cu ajutorul conectorilor logici ($\neg$, $\rightarrow$, $\vee$, $\wedge$, $\leftrightarrow$).

Exemplu

Fie φ propoziția:

$$(\texttt{stark} \wedge \neg\ \texttt{dead}) \rightarrow (\texttt{sansa} \vee \texttt{arya} \vee \texttt{bran})$$

Cine este $\neg\varphi$? Propoziția $\neg\varphi$ este:

$$\texttt{stark} \wedge \neg\ \texttt{dead} \wedge \neg\texttt{sansa} \wedge \neg\texttt{arya} \wedge \neg\texttt{bran}$$

Limbajul și formulele PL

- Limbajul PL
  - variabile propoziționale: $Var = \{p, q, v, \ldots\}$
  - conectori logici: $\neg$ (unar), $\rightarrow$, $\wedge$, $\vee$, $\leftrightarrow$ (binari)
- Formulele PL

$$
\begin{array}{lcl}
var & ::= & p \mid q \mid v \mid \ldots \\
form & ::= & var \mid (\neg form) \mid form \wedge form \mid form \vee form \\
 & & \mid form \rightarrow form \mid form \leftrightarrow form
\end{array}
$$

Exemplu

- Nu sunt formule: $v_1 \neg \rightarrow (v_2)$, $\neg v_1 v_2$
- Sunt formule: $((v_1 \rightarrow v_2) \rightarrow (\neg v_1))$, $(\neg(v_1 \rightarrow v_2))$

- Notăm cu *Form* mulțimea formulelor.

Limbajul și formulele PL

- Limbajul PL
  - variabile propoziționale: $Var = \{p, q, v, \ldots\}$
  - conectori logici: $\neg$ (unar), $\rightarrow$, $\wedge$, $\vee$, $\leftrightarrow$ (binari)

- Formulele PL

$$
\begin{array}{lcl}
\mathit{var} & ::= & p \mid q \mid v \mid \ldots \\
\mathit{form} & ::= & \mathit{var} \mid (\neg \mathit{form}) \mid \mathit{form} \wedge \mathit{form} \mid \mathit{form} \vee \mathit{form} \\
 & & \mid \mathit{form} \rightarrow \mathit{form} \mid \mathit{form} \leftrightarrow \mathit{form}
\end{array}
$$

- Conectorii sunt împărțiți în conectori de bază și conectori derivați (în funcție de formalism).

- Dacă $\neg$ și $\rightarrow$ sunt conectori de bază, atunci:

$$
\begin{array}{rcl}
\varphi \vee \psi & := & \neg\varphi \rightarrow \psi \\
\varphi \wedge \psi & := & \neg(\varphi \rightarrow \neg\psi) \\
\varphi \leftrightarrow \psi & := & (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)
\end{array}
$$

Sintaxa și semantica

Un sistem logic are două componente:

- Sintaxa
  - noțiuni sintactice: demonstrație, teoremă
  - notăm prin $\vdash \varphi$ faptul că φ este teoremă
  - notăm prin $\Gamma \vdash \varphi$ faptul că formula φ este demonstrabilă din mulțimea de formule Γ
- Semantica
  - noțiuni semantice: adevăr, model, tautologie (formulă universal adevărată)
  - notăm prin $\models \varphi$ faptul că φ este tautologie
  - notăm prin $\Gamma \models \varphi$ faptul că formula φ este adevărată atunci când toate formulele din mulțimea Γ sunt adevărate

Exemplu

Formalizați următorul raționament:

`If winter is coming and Ned is not alive then Robb is lord of Winterfell. Winter is coming. Rob is not lord of Winterfell. Then Ned is alive.`

O posibilă formalizare este următoarea:

p = `winter is coming`
q = `Ned is alive`
r = `Robb is lord of Winterfel`

$\{(p \wedge \neg q) \rightarrow r, p, \neg r\} \models q$

Semantica PL

- Mulțimea valorilor de adevăr este $\{0, 1\}$ pe care considerăm următoarele operații:

x	$\neg x$
0	1
1	0

x	y	$x \to y$
0	0	1
0	1	1
1	0	0
1	1	1

$$x \vee y := max\{x, y\} \qquad x \wedge y := min\{x, y\}$$

Semantica PL

- o funcție $e : Var \to \{0, 1\}$ se numește evaluare (interpretare)
- pentru orice evaluare $e : Var \to \{0, 1\}$ există o unică funcție $e^+ : Form \to \{0, 1\}$ care verifică următoarele proprietăți:
  - $e^+(v) = e(v)$
  - $e^+(\neg\varphi) = \neg e^+(\varphi)$
  - $e^+(\varphi \to \psi) = e^+(\varphi) \to e^+(\psi)$
  - $e^+(\varphi \wedge \psi) = e^+(\varphi) \wedge e^+(\psi)$
  - $e^+(\varphi \vee \psi) = e^+(\varphi) \vee e^+(\psi)$

  oricare ar fi $v \in Var$ și φ, $\psi \in Form$.

Exemplu

Dacă $e(p) = 0$ și $e(q) = 1$ atunci

$$e^+(p \vee (p \to q)) = e^+(p) \vee e^+(p \to q) = e(p) \vee (e(p) \to e(q)) = 1$$

Semantica PL

Considerăm $\Gamma \cup \{\varphi\} \subseteq Form$.

- O evaluare $e : Var \to \{0,1\}$ este model al formulei φ dacă $e^+(\varphi) = 1$. Evaluarea e este model al lui Γ dacă $e^+(\Gamma) = \{1\}$, i.e. $e^+(\gamma) = 1$ oricare $\gamma \in \Gamma$.

- O formulă φ este satisfiabilă dacă are un model. O mulțime Γ de formule este satisfiabilă dacă are un model.

- O formulă φ este tautologie (validă, universal adevarată) dacă $e^+(\varphi) = 1$ pentru orice evaluare $e : Var \to \{0,1\}$.
  Notăm prin $\models \varphi$ faptul că φ este o tautologie.

- O formulă φ este $\Gamma-$tautologie (consecință semantică a lui Γ) dacă orice model al lui Γ este și model pentru φ, i.e. $e^+(\Gamma) = \{1\}$ implică $e^+(\varphi) = 1$ pentru orice evaluare $e : Var \to \{0,1\}$.
  Notăm prin $\Gamma \models \varphi$ faptul că φ este o Γ-tautologie.

Semantica PL

Cum verificăm că o formulă este tautologie: $\models \varphi$?

- Fie $v_1, \ldots, v_n$ variabilele care apar în φ.
- Cele 2^n evaluări posibile $e_1, \ldots, e_{2^n}$ pot fi scrise într-un tabel:

v_1	v_2	$\ldots$	v_n	φ
$e_1(v_1)$	$e_1(v_2)$	$\ldots$	$e_1(v_n)$	$e_1^+(\varphi)$
$e_2(v_1)$	$e_2(v_2)$	$\ldots$	$e_2(v_n)$	$e_2^+(\varphi)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$e_{2^n}(v_1)$	$e_{2^n}(v_2)$	$\ldots$	$e_{2^n}(v_n)$	$e_{2^n}^+(\varphi)$

Fiecare evaluare corespunde unei linii din tabel!

- $\models \varphi$ dacă și numai dacă $e_1^+(\varphi) = \cdots = e_{2^n}^+(\varphi) = 1$

Verificarea problemei consecinței logice

- În principiu, putem verifica problema consecinței logice construind un tabel de adevăr, cu câte o linie pentru fiecare interpretare posibilă.
- În cazul în care formula conțin n variabile, tabelul de adevăr are 2^n rânduri. Această metodă este atât de costisitoare computațional (timp exponențial).
- Problemă deschisă de un milion de dolari:

  Este posibil să decidem problema consecinței logice în cazul propozițional printr-un algoritm care să funcționeze în timp polinomial?

  Echivalent, este adevărată $P = NP$?
  (Institutul de Matematica Clay – Millennium Prize Problems)
- SAT este problema satisfiabilității în calculul propozițional clasic. SAT-solverele sunt bazate pe metode sintactice.

Sintaxa PL

Sisteme deductive pentru calculul propozițional clasic:

- Sistemul Hilbert
- Rezoluție
- Deducția naturală
- Calculul cu secvenți

Sistemul Hilbert

- Oricare ar fi $\varphi, \psi, \chi \in Form$ următoarele formule sunt axiome:
  - (A1) $\varphi \to (\psi \to \varphi)$
  - (A2) $(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$
  - (A3) $(\neg\psi \to \neg\varphi) \to (\varphi \to \psi)$.
- Regula de deducție este modus ponens: $\dfrac{\varphi, \ \varphi \to \psi}{\psi}\,\textsf{MP}$
- O demonstrație pentru φ este o secvență de formule $\gamma_1, \ldots, \gamma_n$ astfel încât $\gamma_n = \varphi$ și, pentru fiecare $i \in \{1, \ldots, n\}$, una din următoarele condiții este satisfăcută:
  - γ_i este axiomă,
  - γ_i se obține din formulele anterioare prin MP:
    există $j, k < i$ astfel încât $\gamma_j = \gamma_k \to \gamma_i$
- O formulă φ este teoremă dacă are o demonstrație.
  Notăm prin $\vdash \varphi$ faptul că φ este teoremă.

Sistemul Hilbert

Fie $\Gamma \cup \{\varphi\} \subseteq Form$.

- O demonstrație din ipotezele Γ (sau Γ-demonstrație) pentru φ este o secvență de formule γ_1, ..., γ_n astfel încât $\gamma_n = \varphi$ și, pentru fiecare $i \in \{1, \dots, n\}$, una din următoarele condiții este satisfăcută:
  - γ_i este axiomă,
  - $\gamma_i \in \Gamma$
  - γ_i se obține din formulele anterioare prin MP:
    există j, $k < i$ astfel încât $\gamma_j = \gamma_k \to \gamma_i$
- O formulă φ este Γ-teoremă dacă are o Γ-demonstrație.
  Notăm prin $\Gamma \vdash \varphi$ faptul că φ este o Γ-teoremă

Sistemul Hilbert

Teorema deducției TD (Herbrand, 1930)

Fie $\Gamma \cup \{\varphi\} \subseteq Form$. Atunci

$$\Gamma \vdash \varphi \to \psi \text{ dacă și numai dacă } \Gamma \cup \{\varphi\} \vdash \psi$$

Exemplu

Arătați că $\vdash (\varphi \to \psi) \to ((\psi \to \chi) \to (\varphi \to \chi))$

(1) $\{\varphi \to \psi, \psi \to \chi, \varphi\} \vdash \varphi$ (ipoteza)
(2) $\{\varphi \to \psi, \psi \to \chi, \varphi\} \vdash \varphi \to \psi$ (ipoteza)
(3) $\{\varphi \to \psi, \psi \to \chi, \varphi\} \vdash \psi$ (1),(2), MP
(4) $\{\varphi \to \psi, \psi \to \chi, \varphi\} \vdash \psi \to \chi$ (ipoteza)
(5) $\{\varphi \to \psi, \psi \to \chi, \varphi\} \vdash \chi$ (3),(4), MP
(6) $\{\varphi \to \psi, \psi \to \chi\} \vdash \varphi \to \chi$ TD
(7) $\{\varphi \to \psi\} \vdash (\psi \to \chi) \to (\varphi \to \chi)$ TD
(8) $\vdash (\varphi \to \psi) \to ((\psi \to \chi) \to (\varphi \to \chi))$ TD

Sistemul Hilbert

- Oricare ar fi $\varphi, \psi, \chi \in Form$ următoarele formule sunt axiome:

  (A1) $\varphi \to (\psi \to \varphi)$
  (A2) $(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$
  (A3) $(\neg\psi \to \neg\varphi) \to (\varphi \to \psi)$.

- Regula de deducție este modus ponens: $\dfrac{\varphi, \ \varphi \to \psi}{\psi}\,\mathbf{MP}$

Teorema de completitudine

Γ-teoremele și Γ-tautologiile coincid, i.e.

$$\Gamma \vdash \varphi \text{ dacă și numai dacă } \Gamma \models \varphi$$

oricare are fi $\Gamma \cup \{\varphi\} \in Form$.
În particular, $\vdash \varphi$ dacă și numai dacă $\models \varphi$.

($\Rightarrow$) Corectitudine (exercițiu)
($\Leftarrow$) Completitudine

Reguli de deducție pentru PL

O regula de deducție are forma

$$\frac{\Gamma_1 \vdash \varphi_1, \ldots, \Gamma_n \vdash \varphi_n}{\Gamma \vdash \varphi}$$

A demonstra o regulă de deducție derivată revine la a deduce concluzia $\Gamma \vdash \varphi$ din premisele $\Gamma_1 \vdash \varphi_1$, ..., $\Gamma_n \vdash \varphi_n$.

Exemplu

Folosind teorema deducției se demonstrează regula:

$$\frac{\Gamma \cup \{\varphi\} \vdash \psi}{\Gamma \vdash \varphi \to \psi}$$

Forme normale în calculul propozițional

Formule și funcții

Arătați că $\models v_1 \to (v_2 \to (v_1 \wedge v_2))$.

v_1	v_2	$v_1 \to (v_2 \to (v_1 \wedge v_2))$
0	0	1
0	1	1
1	0	1
1	1	1

Acest tabel definește o funcție $F : \{0,1\}^2 \to \{0,1\}$

x_1	x_2	$F(x_1, x_2)$
0	0	1
0	1	1
1	0	1
1	1	1

Funcția asociată unei formule

Fie φ o formulă v_1, ..., v_n variabilele care apar în φ, e_1, ..., e_{2^n} evaluările posibile. Tabelul asociat

v_1	v_2	$\ldots$	v_n	φ
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$e_k(v_1)$	$e_k(v_2)$	$\ldots$	$e_k(v_n)$	$f_{e_k}(\varphi)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

definește funcția $F_\varphi : \{0,1\}^n \to \{0,1\}$

x_1	x_2	$\ldots$	x_n	$F_\varphi(x_1,\ldots,x_n)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$e_k(v_1)$	$e_k(v_2)$	$\ldots$	$e_k(v_n)$	$f_{e_k}(\varphi)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Funcția asociată unei formule

Fie φ o formulă cu variabilele $v_1, \ldots, v_n$.

- Funcția asociată lui φ este $F_\varphi : \{0,1\}^n \to \{0,1\}$ definită prin
$$F_\varphi(x_1, \ldots, x_n) = f_e(\varphi),$$
unde $e(v_1) = x_1$, $e(v_2) = x_2$, ..., $e(v_n) = x_n$
- O **funcție Booleană** în n variabile este o funcție $F : \{0,1\}^n \to \{0,1\}$.
- F_φ este o funcție Booleană pentru orice φ.

Teoremă

Dacă φ și ψ sunt formule cu variabilele $v_1, \ldots, v_n$, atunci
$$\models \varphi \leftrightarrow \psi \Leftrightarrow F_\varphi = F_\psi$$

Dem. exercițiu

Caracterizarea funcțiilor Booleene

Teorema FB

Pentru orice funcție Booleană $H : \{0,1\}^n \to \{0,1\}$ există o formulă φ cu n variabile astfel încât $H = F_\varphi$.

Dem. Dacă $x \in \{0,1\}^n$ atunci $x = (x_1, \ldots, x_n)$. Definim
$T = \{x \in \{0,1\}^n | H(x) = 1\}$ si $F = \{x \in \{0,1\}^n | H(x) = 0\}$,

$$\theta_1 := \bigvee_{x \in T} \left(\bigwedge_{x_i = 1} v_i \wedge \bigwedge_{x_i = 0} \neg v_i \right),$$

$$\theta_2 := \bigwedge_{x \in F} \left(\bigvee_{x_i = 1} \neg v_i \wedge \bigvee_{x_i = 0} v_i \right).$$

$F_{\theta_1}(y_1, \ldots, y_n) = 1 \Leftrightarrow$
există $x \in T$ astfel încât $\bigwedge_{x_i=1} y_i \wedge \bigwedge_{x_i=0} \neg y_i = 1 \Leftrightarrow$
$\bigwedge_{x_i=1} y_i = 1$ si $\bigwedge_{x_i=0} \neg y_i = 1 \Leftrightarrow$
$x_i = y_i$ oricare $i \in \{1, \ldots, n\} \Leftrightarrow$
$(y_1, \ldots, y_n) = x \in T \Leftrightarrow H(y_1, \ldots, y_n) = 1$.

Similar $F_{\theta_2}(y_1, \ldots, y_n) = 0 \Leftrightarrow H(y_1, \ldots, y_n) = 0$.
Am demonstrat că $H = F_{\theta_1} = F_{\theta_2}$.

Exemplu

Fie $H : \{0,1\}^3 \to \{0,1\}$ descrisă prin tabelul:

x	y	z	$H(x,y,z)$	
0	0	0	0	$M_1 = x \vee y \vee z$
0	0	1	0	$M_2 = x \vee y \vee \neg z$
0	1	0	1	$m_1 = \neg x \wedge y \wedge \neg z$
0	1	1	0	$M_3 = x \vee \neg y \vee \neg z$
1	0	0	1	$m_2 = x \wedge \neg y \wedge \neg z$
1	0	1	1	$m_3 = x \wedge \neg y \wedge z$
1	1	0	1	$m_4 = x \wedge y \wedge \neg z$
1	1	1	1	$m_5 = x \wedge y \wedge z$

$H(x,y,z) = m_1 \vee m_2 \vee m_3 \vee m_4 \vee m_5$

$H(x,y,z) = M_1 \wedge M_2 \wedge M_3$

FND si FNC

Un literal este o variabilă sau negația unei variabile.

- O formă normală disjunctivă (FND) este o disjuncție de conjuncții de literali.
- O formă normală conjuctivă (FNC) este o conjuncție de disjuncții de literali.
  - clauza este o disjunctie de literali
  - multime de clauze este o FNC

Teoremă

Pentru orice formulă φ există o FND θ_1 si o FNC θ_2 astfel încât $\models \varphi \leftrightarrow \theta_1$ si $\models \varphi \leftrightarrow \theta_2$

FNC

Orice formulă poate fi adusa la FNC prin urmatoarele transformări:

1. înlocuirea implicațiilor și echivalențelor

$$\varphi \to \psi \sim \neg\varphi \vee \psi,$$
$$\varphi \leftrightarrow \psi \sim (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$$

2. regulile De Morgan

$$\neg(\varphi \vee \psi) \sim \neg\varphi \wedge \neg\psi,$$
$$\neg(\varphi \wedge \psi) \sim \neg\varphi \vee \neg\psi,$$

3. principiului dublei negații

$$\neg\neg\psi \sim \psi$$

4. distributivitatea

$$\varphi \vee (\psi \wedge \chi) \sim (\varphi \vee \psi) \wedge (\varphi \vee \chi),$$
$$(\psi \wedge \chi) \vee \varphi \sim (\psi \vee \varphi) \wedge (\chi \vee \varphi)$$

Exemple

1. Determinați FNC pentru formula
$(\neg p \rightarrow \neg q) \rightarrow (p \rightarrow q)$
$\sim \neg(\neg p \rightarrow \neg q) \vee (p \rightarrow q)$
$\sim \neg(p \vee \neg q) \vee (\neg p \vee q)$
$\sim (\neg p \wedge q) \vee (\neg p \vee q)$
$\sim (\neg p \vee \neg p \vee q) \wedge (q \vee \neg p \vee q)$

2. Determinați FNC pentru formula
$\neg((p \wedge q) \rightarrow q)$
$\sim \neg(\neg(p \wedge q) \vee q)$
$\sim p \wedge q \wedge \neg q$

Rezoluția în calculul propozițional

Clauze

Definitii

Un **literal** este o variabila sau negatia unei variabile.
O **clauza** este o multime finita de literali:
$C = \{L_1, \ldots, L_n\}$, unde L_1, ..., L_n sunt literali.
C este **triviala** daca exista $p \in Var$ astfel incat $p, \neg p \in C$.

Daca $e : Var \to \{0, 1\}$ este o evaluare și $C = \{L_1, \ldots, L_n\}$, vom nota $e(C) := f_e(L_1 \vee \ldots \vee L_n)$.

$C = \{L_1, \ldots, L_n\}$ este **satisfabila** daca formula $L_1 \vee \ldots \vee L_n$ este satisfiabila, adica exista o evaluare $e : Var \to \{0, 1\}$ cu $e(C) = 1$.

Clauza vida $\square := \{\}$ nu este satisfiabila (disjunctie indexata de $\emptyset$).

O multime de clauze $\mathcal{S} = \{C_1, \ldots, C_m\}$ este satisfiabila daca exista o evaluare $e : Var \to \{0, 1\}$ astfel incat $e(C_i) = 1$ oricare $i \in \{1, \ldots, m\}$.

Clauze

Observatie

Putem identifica
clauza $C = \{L_1, \ldots, L_n\}$ cu $L_1 \vee \ldots \vee L_n$,
multimea de clauze $\mathcal{S} = \{C_1, \ldots, C_m\}$ cu $C_1 \wedge \ldots \wedge C_m$.

Definitie

C clauza, $\mathcal{S}$ multime de clauze
$Var(C) = \{p \in Var | p \in C \text{ sau } \neg p \in C\}$,
$Var(\mathcal{S}) = \bigcup\{Var(C) | C \in \mathcal{S}\}$.

Exemple

1. p, $\neg r$, q sunt literali.
2. $\{p, \neg r\}$, $\{\neg r, r\}$, $\{q, p\}$, $\{q, \neg p, r\}$ sunt clauze.
3. $\mathcal{S} = \{\{p, \neg r\}, \{\neg r, r\}, \{q, p\}, \{q, \neg p, r\}\}$ este satisfiabila.
Dem. Consideram $e(p) = e(q) = 1$.
4. $\mathcal{S} = \{\{\neg p, q\}, \{\neg r, \neg q\}, \{p\}, \{r\}\}$ nu este satisfiabila.
Dem. Daca exista o evaluare e care satisface $\mathcal{C}$, atunci $e(p) = e(r) = 1$. Rezulta $e(q) = 0$, deci $f_e(\{\neg p, q\}) = f_e(\neg p \vee q) = 0$. In consecinta, $\{\neg p, q\}$ nu e satisfacuta de e.
5. O multime de clauze triviale este intotdeauna satisfiabila.

Proprietati

Propozitie

Fie C, D clauze si $\mathcal{T}$, $\mathcal{S}$, $\mathcal{U}$ multimi de clauze. Urmatoarele implicatii sunt adevarate.

(p1) $C \subseteq D$, C satisfiabila $\Rightarrow$ D satisfiabila

(p2) $C \cup D$ satisfiabila $\Rightarrow$ C satisfiabila sau D satisfiabila

(p3) $p \notin Var(C) \Rightarrow C \cup \{p\}$, $C \cup \{\neg p\}$ satisfiabile

(p4) $\mathcal{S} \subseteq \mathcal{T}$, $\mathcal{T}$ satisfiabila $\Rightarrow$ $\mathcal{S}$ satisfiabila

(p5) Fie $p \in Var$ si $\mathcal{U}$, $\mathcal{T}$, $\mathcal{S}$ multimi de clauze astfel incat

$$p \notin Var(\mathcal{U}),$$
$$\text{or. } T \in \mathcal{T} \ (p \in T \text{ si } \neg p \notin T),$$
$$\text{or. } S \in \mathcal{S} \ (p \notin S \text{ si } \neg p \in S).$$

Atunci $\mathcal{U}$ satisfiabila $\Rightarrow \mathcal{U} \cup \mathcal{T}$, $\mathcal{U} \cup \mathcal{S}$ satisfiabile

Dem. exercitiu

Regula Rezolutiei

Regula Rezolutiei

$$Rez \quad \frac{C_1 \cup \{p\}, C_2 \cup \{\neg p\}}{C_1 \cup C_2}$$

unde $\{p, \neg p\} \cap C_1 = \emptyset$ si $\{p, \neg p\} \cap C_2 = \emptyset$.

Propozitie

Regula Rezolutiei pastreaza satsifiabilitatea, i.e.
$\{C_1 \cup \{p\}, C_2 \cup \{\neg p\}\}$ satisfiabila $\Leftrightarrow$ $C_1 \cup C_2$ satisfiabila.

Regula Rezolutiei

Propozitie

Regula Rezolutiei pastreaza satsifiabilitatea, i.e.
$\{C_1 \cup \{p\}, C_2 \cup \{\neg p\}\}$ satisfiabila $\Leftrightarrow$ $C_1 \cup C_2$ satisfiabila.

Dem. Fie $e : Var \to \{0, 1\}$ este o evaluare astfel incat $e(C_1 \cup \{p\}) = e(C_2 \cup \{\neg p\}) = 1$.Daca $e(p) = 1$ atunci $e(C_2) = 1$, deci $e(C_1 \cup C_2) = 1$. Similar pentru $e(p) = 0$.

Invers, presupunem ca $e(C_1 \cup C_2) = 1$, deci $e(C_1) = 1$ sau $e(C_2) = 1$. Daca $e(C_1) = 1$ consideram $e' : Var \to \{0, 1\}$ o evaluare cu $e'(v) = e(v)$ daca $v \in Var(C_1)$ si $e'(p) = 0$. Atunci $e'(C_1) = e(C_1) = 1$, deci $e'(C_1 \cup \{p\}) = 1$.De asemenea $e'(C_2 \cup \{\neg p\}) = e'(C_2) \vee e'(\neg p) = 1$, deci e' este model pentru multimea de clauze $\{C_1 \cup \{p\}, C_2 \cup \{\neg p\}\}$.

Exemple

$$\frac{\{p\}, \{\neg p\}}{\square} \quad \frac{\{p\}, \{\neg p, q\}}{\{q\}}$$

Atentie

Aplicarea **simultana** a regulii pentru doua variabile diferit este gresita. De exemplu

$$\frac{\{p, \neg q\}, \{\neg p, q\}}{\square}$$

contrazice rezultatul anterior, deoarece $\{\{p, \neg q\}, \{\neg p, q\}\}$ este satisfiabila ($e(p) = e(q) = 1$).
Aplicarea corecta a Regulii Rezolutiei este

$$\frac{\{p, \neg q\}, \{\neg p, q\}}{\{q, \neg q\}}$$

.

Derivare prin rezolutie

Definitie

Fie $\mathcal{S}$ o multime de clauze. O *derivare prin rezolutie* din $\mathcal{S}$ este o secventa finita de clauze astfel incat fiecare clauza este din $\mathcal{S}$ sau rezulta din clauze anterioare prin rezolutie.

Exemplu $\mathcal{S} = \{\{\neg p, q\}, \{\neg q, \neg r, s\}, \{p\}, \{r\}, \{\neg s\}\}$
O derivare prin rezolutie pentru $\square$ din $\mathcal{S}$ este
$C_1 = \{\neg s\}$ $(\in \mathcal{S})$
$C_2 = \{\neg q, \neg r, s\}$ $(\in \mathcal{S})$
$C_3 = \{\neg q, \neg r\}$ $(C_1,\ C_2,\ Rez)$
$C_4 = \{r\}$ $(\in \mathcal{S})$
$C_5 = \{\neg q\}$ $(C_3,\ C_4,\ Rez)$
$C_6 = \{\neg p, q\}$ $(\in \mathcal{S})$
$C_7 = \{\neg p\}$ $(C_5,\ C_6,\ Rez)$
$C_8 = \{p\}$ $(\in \mathcal{S})$
$C_9 = \square$ $(C_7,\ C_8,\ Rez)$

Procedura Davis-Putnam DPP (informal)

Intrare: o mulțime $\mathcal{C}$ de clauze

Se repetă următorii pași:

- se elimină clauzele triviale
- se alege o variabilă p
- se adaugă la mulțimea de clauze toți rezolvenții obținuti prin aplicarea Rez pe variabila p
- se șterg toate clauzele care conțin p sau $\neg p$

Ieșire: dacă la un pas s-a obținut $\square$, mulțimea $\mathcal{C}$ nu este satisfiabilă; altfel $\mathcal{C}$ este satisfiabilă.

Algoritmul Davis-Putnam(DP)

Intrare: $\mathcal{S}$ multime nevida de clauze netriviale.
$\mathcal{S}_1 := \mathcal{S}$; $i := 1$;

P1. $v_i \in Var(\mathcal{C}_i)$;
$\mathcal{T}_i^1 := \{C \in \mathcal{S}_i | v_i \in C\}$;
$\mathcal{T}_i^0 := \{C \in \mathcal{S}_i | \neg v_i \in C\}$;
$\mathcal{T}_i := \mathcal{T}_i^0 \cup \mathcal{T}_i^1$;
$\mathcal{U}_i := \emptyset$;

P2. if $\mathcal{T}_i^0 \neq \emptyset$ and $\mathcal{T}_i^1 \neq \emptyset$ then
$\mathcal{U}_i := \{C_1 \setminus \{v_i\} \cup C_0 \setminus \{\neg v_i\} | C_1 \in \mathcal{T}_i^1, C_0 \in \mathcal{T}_i^0\}$;

P3. $\mathcal{S}_{i+1} = (\mathcal{S}_i \setminus \mathcal{T}_i) \cup \mathcal{U}_i$; $\mathcal{S}_{i+1} = \mathcal{S}_{i+1} \setminus \{C \in \mathcal{S}_{i+1} | C \text{ triviala}\}$;

P4. if $\mathcal{S}_{i+1} = \emptyset$ then *SAT*
else if $\square \in \mathcal{S}_{i+1}$ then *NESAT*
else $\{i := i + 1$; go to P1$\}$.

Run DP

Exemplu

$\mathcal{S}_1 := \mathcal{S} = \{\{\neg p, q, \neg s\}, \{\neg r, \neg q\}, \{p, r\}, \{p\}, \{r\}, \{s\}\}$

$v_1 := p$; $T_1^1 := \{\{p, r\}, \{p\}\}$; $T_1^0 := \{\{\neg p, q, \neg s\}\}$;
$U_1 := \{\{r, q, \neg s\}, \{q, \neg s\}\}$;
$\mathcal{S}_2 := \{\{\neg r, \neg q\}, \{r\}, \{s\}, \{r, q, \neg s\}, \{q, \neg s\}\}$; $i := 2$;

$v_2 := q$; $T_2^1 := \{\{r, q, \neg s\}, \{q, \neg s\}\}$; $T_2^0 := \{\{\neg r, \neg q\}\}$;
$U_2 := \{\{r, \neg s, \neg r\}, \{\neg s, \neg r\}\}$;
$\mathcal{S}_3 := \{\{r\}, \{s\}, \{\neg s, \neg r\}\}$; $i := 3$;

$v_3 := r$; $T_3^1 := \{\{r\}\}$; $T_3^0 := \{\{\neg s, \neg r\}\}$; $U_3 := \{\{\neg s\}\}$;
$\mathcal{S}_4 := \{\{s\}, \{\neg s\}\}$; $i := 4$;

$v_4 := s$; $T_4^1 := \{\{s\}\}$; $T_4^0 := \{\{\neg s\}\}$; $U_4 := \{\square\}$;
$\mathcal{S}_5 := \{\square\}$
In consecinta, $\mathcal{S}$ nu este satisfiabila

Exemplu

$\mathcal{S}_1 := \mathcal{S} = \{\{p, \neg r\}, \{q, p\}, \{q, \neg p, r\}\}$

$v_1 := r;\ T_1^1 := \{\{q, \neg p, r\}\};\ T_1^0 := \{\{p, \neg r\}\};$
$U_1 := \{\{q, \neg p, p\}\};$
$\mathcal{S}_2 := \{\{q, p\}\};\ i := 2;$

$v_2 := q;\ T_2^1 := \{\{q, p\}\};\ T_2^0 := \emptyset;$
$U_2 := \emptyset;$
$\mathcal{S}_3 := \emptyset$
In consecinta, $\mathcal{S}$ este satisfiabila

Algoritmul DP (facultativ)

Fie $\mathcal{S}$ o multime finita de clauze si n este numarul de variabile care apar in $\mathcal{S}$. Algoritmul DP se opreste deoarece $Var(\mathcal{S}_{i+1}) \subset Var(\mathcal{S}_i)$. Daca n este numarul de variabile care apar in $\mathcal{S}$, atunci exista un $n_0 \leq n$ astfel incat $Var(\mathcal{S}_{n_0+1}) = \emptyset$, deci $\mathcal{S}_{n_0+1} = \emptyset$ sau $\mathcal{S}_{n_0+1} = \{\square\}$. Pentru simplitate, vom presupune in continuare ca $n_0 = n$.

Propozitie

$\mathcal{S}_i$ satisfiabila $\Leftrightarrow$ $\mathcal{S}_{i+1}$ satisfiabila
oricare $i \in \{1, \ldots, n-1\}$.

Dem. Fie $i \in \{1, \ldots, n-1\}$. Stim ca $\mathcal{S}_{i+1} = (\mathcal{S}_i \setminus \mathcal{T}_i) \cup \mathcal{U}_i$. Consideram cazurile $U_i = \emptyset$ si $U_i \neq \emptyset$.
Daca $U_i = \emptyset$, atunci $\mathcal{T}_i^0 = \emptyset$ sau $\mathcal{T}_i^1 = \emptyset$ si $\mathcal{S}_i = (\mathcal{S}_{i+1} \cup \mathcal{T}_i)$. Ne aflam in ipotezele proprietatii (p5) . Din (p4) si (p5) obtinem concluzia dorita.

Algoritmul DP (facultativ)

Dem. continuare

Daca $U_i \neq \emptyset$, notam $\mathcal{C}_i := \mathcal{S}_i \setminus \mathcal{T}_i$. In consecinta $\mathcal{S}_i = \mathcal{C}_i \cup \mathcal{T}_i$ si $\mathcal{S}_{i+1} = \mathcal{C}_i \cup \mathcal{U}_i$. Observam ca fiecare clauza din $\mathcal{U}_i$ se obtine aplicand Regula Rezolutiei pentru doua clauze din $\mathcal{T}_i$. Am demonstrat ca Regula Rezolutiei pastreaza satisfiabilitatea:

$$\mathcal{U}_i \text{ satisfiabila} \Leftrightarrow \mathcal{T}_i \text{ satisfiabila.}$$

Au loc urmatoarele echivalente:
$\mathcal{S}_i = \mathcal{C}_i \cup \mathcal{T}_i$ satisfiabila $\Leftrightarrow \mathcal{C}_i$, $\mathcal{T}_i$ satisfiabile
$\Leftrightarrow \mathcal{C}_i, \mathcal{U}_i$ satisfiabile $\Leftrightarrow \mathcal{U}_i \cup \mathcal{T}_i = \mathcal{S}_{i+1}$ satisfiabila.

Algoritmul DP (facultativ)

Teorema DP

Algoritmul DP este corect si complet, i.e. $\mathcal{S}$ nu este satisfiabila daca si numai daca iesirea algoritmului DP este $\{\square\}$.

Dem. Din propozitia anterioara, $\mathcal{S}$ nu este satisfiabila daca si numai daca $\mathcal{S}_n$ nu este satisfiabila. Fie p unica variabila propozitionala care apare in $\mathcal{S}_n$.

Daca $\mathcal{S}_n = \{\{p\}\}$ atunci $\mathcal{S}_n$ este satisfiabila deoarece $e(p) = 1$ este un model. Daca $\mathcal{S}_n = \{\{\neg p\}\}$ atunci $\mathcal{S}_n$ este satisfiabila deoarece $e(p) = 0$ este un model. In ambele cazuri $\mathcal{S}_{n+1} = \{\}$.
Daca $\mathcal{S}_n = \{\{p\}, \square\}$ sau $\mathcal{S}_n = \{\{\neg p\}, \square\}$ atunci $\mathcal{S}_n$ nu este satisfiabila si $\mathcal{S}_{n+1} = \{\square\}$.
Daca $\mathcal{S}_n = \{\{p\}, \{\neg p\}\}$ sau $\mathcal{S}_n = \{\{p\}, \{\neg p\}, \square\}$ atunci $\mathcal{S}_n$ nu este satisfiabila, putem aplica Regula Rezolutiei si obtinem $\mathcal{S}_{n+1} = \{\square\}$.

Se observa ca S_n nu este satisfiabila daca si numai daca $S_{n+1} = \{\square\}$.

Rezolutia

Observatie

Algoritmul DP cu intrarea $\mathcal{S}$ se termina cu $\{\square\}$ daca si numai daca exista o derivare prin rezolutie a clauzei vide $\square$ din $\mathcal{S}$.

Teorema

Daca $\mathcal{S}$ este o multime finita nevida de clauze, sunt echivalente:

- $\mathcal{S}$ nu este satisfiabila,
- exista o derivare prin rezolutie a clauzei vide $\square$ din $\mathcal{S}$.

Forma clauzala

Fie φ o formula si $C_1 \wedge \cdots \wedge C_n$ o FNC astfel incat $\models \varphi \leftrightarrow C_1 \wedge \cdots \wedge C_n$. Spunem ca multimea de clauze $\{C_1, \cdots, C_n\}$ este **forma clauzala** a lui φ.

Lema

Sunt echivalente:

- φ satisfiabila,
- $C_1 \wedge \cdots \wedge C_n$ satisfiabila,
- $\{C_1, \cdots, C_n\}$ satisfiabila.

Dem. exercitiu

Daca $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ este o multime de formule atunci o forma clauzala pentru Γ este $\mathcal{S} := \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_n$ unde $\mathcal{S}_i$ este forma clauzala pentru γ_i oricare i. Observăm că Γ este satisfiabila dacă și numai dacă $\mathcal{S}$ satisfiabila

Demonstratii prin rezoluție

Teorema

Fie Γ o multime finită de formule, φ o formula și $\mathcal{S}$ o forma clauzală pentru $\Gamma \cup \{\neg\varphi\}$. Sunt echivalente:

- $\Gamma \models \varphi$,
- $\Gamma \cup \{\neg\varphi\}$ nu e satisfiabilă,
- $\mathcal{S}$ nu este satisfiabilă,
- exista o derivare prin rezolutie a clauzei vide $\square$ din $\mathcal{S}$.

Observatie

Teorema este adevarata si pentru Γ multime oarecare. Demonstratia foloseste *compacitatea* calculului propozitional:
o multime de formule este satisfiabila daca si numai daca orice submultime finita a sa este satisfiabila.

Exemplu

A demonstra ca $\models p \to (q \to p)$ revine la a demonstra ca $\{\neg(p \to (q \to p))\}$ nu e satisfiabila. Pentru aceasta:
(1) determinam o forma clauzala pentru $\neg(p \to (q \to p))$,
(2) aplicam DP.

(1) Determinam FNC pentru $\neg(p \to (q \to p))$:
$\neg(p \to (q \to p)) \sim \neg(\neg p \vee \neg q \vee p) \sim p \wedge q \wedge \neg p$
Forma clauzala este $\mathcal{S} = \{\{p\}, \{q\}, \{\neg p\}\}$.

(2) Aplicam DP:
$\{\{p\}, \{q\}, \{\neg p\}\}$
$\{\{p\}, \{\neg p\}\}$
$\{\square\}$

$\mathcal{S}$ nu este satisfiabila deoarece exista o derivare prin rezolutie a clauzei vide din $\mathcal{S}$. In consecinta, $\models p \to (q \to p)$.

Exemplu

Cercetati daca $\{p \vee q\} \models p \wedge q$. Aceasta revine cerceta satisfiabilitatea multimii $\Delta = \{p \vee q, \neg(p \wedge q)\}$.

(1) O forma clauzala pentru $p \vee q$ este $\{\{p, q\}\}$. O forma clauzala pentru $\neg(p \wedge q)$ este $\{\{\neg p, \neg q\}\}$.
Forma clauzala pentru Δ este $\mathcal{S} = \{\{p, q\}, \{\neg p, \neg q\}\}$.

(2) Aplicam DP:
$\{\{p, q\}, \{\neg p, \neg q\}\}$
$\{\{q, \neg q\}\}$
$\{\}$ (multimea vida)

In acest caz $\mathcal{S}$ este satisfiabila, deci $\{p \vee q\} \not\models p \wedge q$.

Exemplu

Cercetati daca $\{p, p \to (q \vee r)\} \models \neg p \to (\neg p \wedge q \wedge \neg r)$.
(1) Determinam forma clauzala pentru
$\{p, p \to (q \vee r), \neg(\neg p \to (\neg p \wedge q \wedge \neg r))\}$.

Forma clauzala a lui p este $\{\{p\}\}$.

$p \to (q \vee r) \sim \neg p \vee q \vee r$ (FNC)
Forma clauzala a lui $p \to (q \vee r)$ este $\{\{\neg p, q, r\}\}$.

$\neg(\neg p \to (\neg p \wedge q \wedge \neg r)) \sim \neg(p \vee (\neg p \wedge q \wedge \neg r)) \sim \neg p \wedge (p \vee \neg q \vee r))$
Forma clauzala a lui $\neg(\neg p \to (\neg p \wedge q \wedge \neg r))$ este $\{\{\neg p\}, \{p, \neg q, r\}\}$.

(2) Aplicam DP:
$\{\{p\}, \{\neg p, q, r\}, \{\neg p\}, \{p, \neg q, r\}\}$
$\{\{q, r\}, \square, \{\neg q, r\}\}$
$\{\{r\}, \square\}$
$\{\square\}$
Este adevarat ca $\{p, p \to (q \vee r)\} \models \neg p \to (\neg p \wedge q \wedge \neg r)$

Concluzie

Fie Γ o mulțime de formule și φ o formulă.
Notăm prin $\Gamma \vdash_{Rez} \varphi$ faptul că există o derivare prin rezoluție a clauzei vide $\square$ dintr-o formă clauzală a lui $\Gamma \cup \{\neg\varphi\}$. În acest caz spunem că φ se demostrează prin rezoluție din Γ.

Teorema

Sunt echivalente:

- $\Gamma \models \varphi$,
- $\Gamma \vdash_{Rez} \varphi$.

În consecință, regula Rezolutiei este corecta si completa pentru calculul propozițional.

Folosind rezolutia (fara alte axiome si reguli de deductie) se poate construi un demonstrator automat pentru calculul propozițional.

Clauze propoziționale definite. Rezoluția SLD

Clauze propoziționale definite

- O clauză propozițională definită este o formulă care poate avea una din formele:
  1. q (un fapt în Prolog `q.`)
  2. $p_1 \wedge \ldots \wedge p_k \to q$ (o regulă în Prolog `q :- p1,...,pk`)

  unde $q, p_1, \ldots, p_n$ sunt variabile propoziționale

- Numim variabilele propoziționale atomi.

Programare logică – cazul logicii propoziționale

- Un "program logic" este o listă $Cd_1, \ldots, Cd_n$ de clauze definite.
- O întrebare este o listă $q_1, \ldots, q_m$ de atomi.
- Sarcina sistemului este să stabilească:

$$Cd_1, \ldots, Cd_n \models q_1 \wedge \ldots \wedge q_m.$$

Vom studia metode sintactice pentru a rezolva această problemă!

Rezoluția SLD (cazul propozițional)

Fie S o mulțime de clauze definite.

$$\text{SLD}\ \boxed{\ \frac{\neg p_1 \vee \cdots \vee \neg q \vee \cdots \vee \neg p_n}{\neg p_1 \vee \cdots \vee \neg q_1 \vee \cdots \vee \neg q_m \vee \cdots \vee \neg p_n}\ }$$

unde $q \vee \neg q_1 \vee \cdots \vee \neg q_m$ este o clauză definită din S.

Rezoluția SLD

Fie S o mulțime de clauze definite și q o întrebare.

O derivare din S prin rezoluție SLD este o secvență

$$G_0 := \neg q, \quad G_1, \quad \ldots, \quad G_k, \ldots$$

în care G_{i+1} se obține din G_i prin regula SLD.

Dacă există un k cu $G_k = \square$ (clauza vidă), atunci derivarea se numește SLD-respingere.

Rezoluția SLD

Baza de cunoștințe KB:

```
oslo .
windy :- oslo.
norway :- oslo.
cold :- norway.
winter :- cold, windy.
```

Întrebarea:

```
-? winter.
```

- Formă clauzală:

  $$\mathrm{KB} = \{\{oslo\}, \{\neg oslo, windy\}, \{\neg oslo, norway\}, \{\neg norway, cold\}, \{\neg cold, \neg windy, winter\}\}$$

- KB $\vdash$ *winter* dacă și numai dacă $KB \cup \{\neg winter\}$ este satisfiabilă.

- Satisfiabilitatea este verificată prin rezoluție

  SLD = Linear resolution with Selected literal for Definite clauses

Clause Horn propoziționale - rezoluția SLD

Exemplu

Demonstrăm KB $\vdash$ *winter* prin rezoluție SLD:

$\{\neg winter\}$	$\{\neg cold, \neg windy, winter\}$
$\{\neg cold, \neg windy\}$	$\{\neg norway, cold\}$
$\{\neg norway, \neg windy\}$	$\{\neg oslo, norway\}$
$\{\neg oslo, \neg windy\}$	$\{oslo\}$
$\{\neg windy\}$	$\{\neg oslo, windy\}$
$\{\neg oslo\}$	$\{oslo\}$
$\square$	

Pe săptămâna viitoare!