

Logică computațională

Curs 3

Lector dr. Pop Andreea-Diana

Logica propozițiilor

Propozițiile logice sunt modele ale afirmațiilor propoziționale care sunt fie *adevărate*, fie *false*.



Sintaxa logicii propozițiilor

- alfabetul

- $\Sigma_P = \text{Var_propoz} \cup \text{Coneective} \cup \{ (,) \}$
- $\text{Var_propoz} = \{ p, q, r, p_1, p_2, \dots \}$
- $\text{Coneective} = \{ \neg, \wedge, \vee, \rightarrow, \leftrightarrow \}$

- regulile de formare a Formulelor propoziționale

- F_P = multimea formulelor propoziționale corect construite
= cea mai mică multime de formule ce se poate construi cu regulile:
 - *baza*: $p_i \in F_P$, $i = 1, 2, \dots$
 - *inducția*: dacă $U, V \in F_P$ atunci:
 $\neg U \in F_P, U \wedge V \in F_P, U \vee V \in F_P, U \rightarrow V \in F_P, U \leftrightarrow V \in F_P$
 - *închiderea*: toate formulele din F_P se obțin doar prin aplicarea regulilor precedente de un număr finit de ori.



Semantica logicii propoziționale

- Propozițiile logice sunt modele ale afirmațiilor propoziționale care sunt fie *adevărate*, fie *false*.
- **Scopul** definirii semanticăi logicii propoziționale este de a **atribui** un înțeles, **o valoare de adevăr**, formulelor propoziționale.
- Domeniul semantic:
 $\{ F(\text{fals}), T(\text{true, adevărat}) \}$ a.î. $\neg F = T$, $\neg T = F$

Semantica conectivelor

p	$\neg p$
T	F
F	T

\uparrow - nand $p \uparrow q := \neg(p \wedge q)$
 \downarrow - nor $p \downarrow q := \neg(p \vee q)$
 \oplus - xor $p \oplus q := \neg(p \leftrightarrow q)$

p	q	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$	$p \uparrow q$	$p \downarrow q$	$p \oplus q$
T	T	T	T	T	T	F	F	F
T	F	F	T	F	F	T	F	T
F	T	F	T	T	F	T	F	T
F	F	F	F	T	T	T	T	F

Interpretarea (Def.)

- O *interpretare* a formulei $U(p_1, p_2, \dots, p_n) \in F_P$ este o funcție $i: \{p_1, p_2, \dots, p_n\} \rightarrow \{T, F\}$ care asociază valori de adevăr variabilelor propoziționale și poate fi extinsă la o funcție $i: F_P \rightarrow \{T, F\}$ folosind relațiile:

$$\begin{array}{lll} i(\neg p) = \neg i(p) & i(p \wedge q) = i(p) \wedge i(q) \\ i(p \vee q) = i(p) \vee i(q) & i(p \rightarrow q) = i(p) \rightarrow i(q) & i(p \leftrightarrow q) = i(p) \leftrightarrow i(q) \end{array}$$

- Interpretările evaluatează formulele propoziționale conform semanticii conectivelor componente, atribuindu-le valori de adevăr.
- Tabela de adevăr a unei formule propoziționale $U(p_1, p_2, \dots, p_n) \in F_P$ corespunde evaluărilor formulei în toate cele 2^n interpretări.

Concepte semantice (Def.)

Fie formula propozițională $U(p_1, p_2, \dots, p_n) \in F_P$.

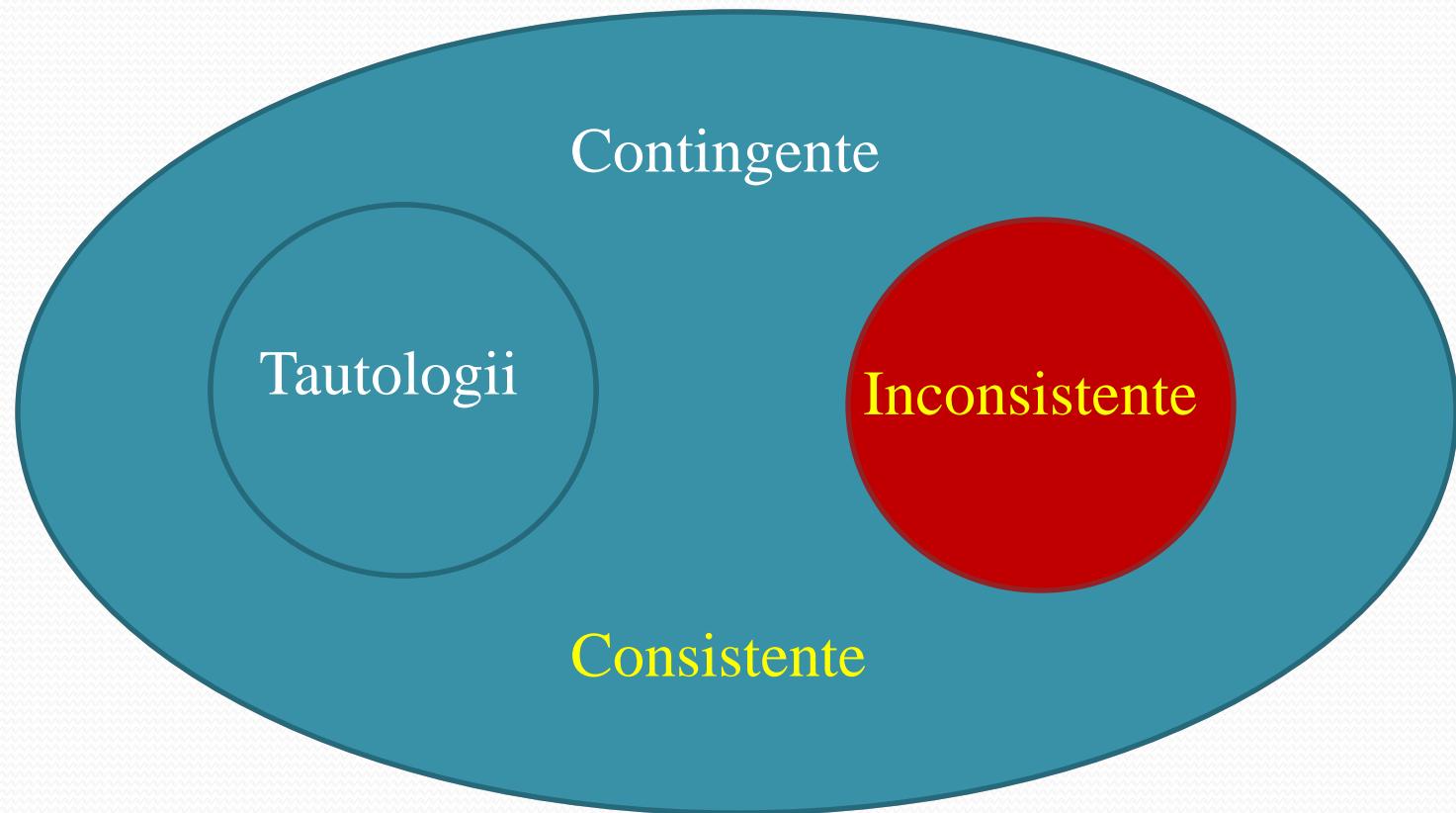
- O interpretare $i: \{p_1, p_2, \dots, p_n\} \rightarrow \{T, F\}$ care evaluează formula U ca adevărată, $i(U)=T$, se numește **model** al formulei.
- O interpretare $i: \{p_1, p_2, \dots, p_n\} \rightarrow \{T, F\}$ care evaluează formula U ca falsă, $i(U)=F$, se numește **anti-model** al formulei.

Concepte semantice (Def.) – cont.

Fie formula propozițională $U(p_1, p_2, \dots, p_n) \in F_P$.

- U se numește **consistentă (realizabilă)** dacă și numai dacă are cel puțin un model, deci poate fi evaluată ca adevărată:
 $\exists i: \{p_1, p_2, \dots, p_n\} \rightarrow \{\text{T}, \text{F}\}$ astfel încât $i(U)=\text{T}$.
- U se numește **validă (tautologie)**, notație: $\models U$, dacă și numai dacă U este evaluată ca adevărată în orice interpretare, adică:
 $\forall i: \{p_1, p_2, \dots, p_n\} \rightarrow \{\text{T}, \text{F}\}$, $i(U)=\text{T}$. Toate interpretările formulei U sunt modele ale formulei.
- Formula U se numește **inconsistentă (nerealizabilă)** dacă și numai dacă U nu are niciun model, adică U este interpretată totdeauna ca falsă: $\forall i: \{p_1, p_2, \dots, p_n\} \rightarrow \{\text{T}, \text{F}\}$, $i(U)=\text{F}$.
- Formula U se numește **contingentă** dacă și numai dacă este consistentă, dar nu este validă.

Tipuri de formule



Exemplu

- $U(p,q,r) = (\neg p \vee q) \wedge (r \vee p),$
- $V(p,q,r) = (\neg p \wedge r) \vee (q \wedge r) \vee (q \wedge p)$
- $p \uparrow \neg p$
- $p \downarrow \neg p$

Tabela de adevăr - completarea

	p	q	r	$\neg p \vee q$	$r \vee p$	$U(p,q,r)$	$V(p,q,r)$	$p \uparrow \neg p$	$p \downarrow \neg p$
i_1	T	T	T	T	T	T	T	T	F
i_2	T	T	F	T	T	T	T	T	F
i_3	T	F	T	F	T	F	F	T	F
i_4	T	F	F	F	T	F	F	T	F
i_5	F	T	T	T	T	T	T	T	F
i_6	F	T	F	T	F	F	F	T	F
i_7	F	F	T	T	T	T	T	T	F
i_8	F	F	F	T	F	F	F	T	F

$$U(p,q,r) = (\neg p \vee q) \wedge (r \vee p),$$

$$V(p,q,r) = (\neg p \wedge r) \vee (q \wedge r) \vee (q \wedge p)$$

Tabela de adevăr – interpretări

	p	q	r	$\neg p \vee q$	$r \vee p$	$U(p,q,r)$	$V(p,q,r)$	$p \uparrow \neg p$	$p \downarrow \neg p$
i_1	T	T	T	T	T	T	T	T	F
i_2	T	T	F	T	T	T	T	T	F
i_3	T	F	T	F	T	F	F	T	F
i_4	T	F	F	F	T	F	F	T	F
i_5	F	T	T	T	T	T	T	T	F
i_6	F	T	F	T	F	F	F	T	F
i_7	F	F	T	T	T	T	T	T	F
i_8	F	F	F	T	F	F	F	T	F

modele pt. U : i_1, i_2, i_5 și i_7

$i_1 : \{p, q, r\} \rightarrow \{\text{T}, \text{F}\}$, $i_1(p) = \text{T}$, $i_1(q) = \text{T}$, $i_1(r) = \text{T}$ și $i_1(U) = \text{T}$

anti-modele pt. U : i_3, i_4, i_6 și i_8

Tabela de adevăr – tip formule

	p	q	r	$\neg p \vee q$	$r \vee p$	$U(p,q,r)$	$V(p,q,r)$	$p \uparrow \neg p$	$p \downarrow \neg p$
i_1	T	T	T	T	T	T	T	T	F
i_2	T	T	F	T	T	T	T	T	F
i_3	T	F	T	F	T	F	F	T	F
i_4	T	F	F	F	T	F	F	T	F
i_5	F	T	T	T	T	T	T	T	F
i_6	F	T	F	T	F	F	F	T	F
i_7	F	F	T	T	T	T	T	T	F
i_8	F	F	F	T	F	F	F	T	F

$p \uparrow \neg p$ – tautologie

$p \downarrow \neg p$ – inconsistentă

U, V – contingente și consistente

Metasimboluri – relații semantice între formule

- Formula V este ***consecință logică*** a formulei U , notăție: $U \models V$, dacă și numai dacă $\forall i: F_P \rightarrow \{T, F\}$ astfel încât $i(U) = T$, are loc $i(V) = T$.
- Formulele $U(p_1, p_2, \dots, p_n) \in F_P$ și $V(p_1, p_2, \dots, p_n) \in F_P$ sunt ***logic echivalente***, notăție: $U \equiv V$, dacă și numai dacă tabelele lor de adevăr sunt identice, adică: $\forall i: F_P \rightarrow \{T, F\}$, $i(U) = i(V)$.

Tabela de adevăr – tip formule

	p	q	r	$\neg p \vee q$	$r \vee p$	$U(p,q,r)$	$V(p,q,r)$	$p \uparrow \neg p$	$p \downarrow \neg p$
i_1	T	T	T	T	T	T	T	T	F
i_2	T	T	F	T	T	T	T	T	F
i_3	T	F	T	F	T	F	F	T	F
i_4	T	F	F	F	T	F	F	T	F
i_5	F	T	T	T	T	T	T	T	F
i_6	F	T	F	T	F	F	F	T	F
i_7	F	F	T	T	T	T	T	T	F
i_8	F	F	F	T	F	F	F	T	F

$$U \equiv V$$

$$U \models \neg p \vee q$$

Concepțe semantice pentru mulțimi de formule

- O **mulțime** $\{U_1, U_2, \dots, U_n\}$ de formule se numește **consistentă (realizabilă)** dacă și numai dacă formula $U_1 \wedge U_2 \wedge \dots \wedge U_n$ este consistentă, adică: $\exists i: F_P \rightarrow \{T, F\}$ astfel încât $i(U_1 \wedge U_2 \wedge \dots \wedge U_n) = T$, i se numește **model** al mulțimii $\{U_1, U_2, \dots, U_n\}$.
- O **mulțime** $\{U_1, U_2, \dots, U_n\}$ de formule se numește **inconsistentă (nerealizabilă, contradicțorie)** dacă și numai dacă formula $U_1 \wedge U_2 \wedge \dots \wedge U_n$ este inconsistentă, adică, $\forall i: F_P \rightarrow \{T, F\}$ astfel încât $i(U_1 \wedge U_2 \wedge \dots \wedge U_n) = F$, i se numește **anti-model** al mulțimii $\{U_1, U_2, \dots, U_n\}$.
- Formula V este **consecință logică** a mulțimii de formule $\{U_1, U_2, \dots, U_n\}$ și se notează $U_1, U_2, \dots, U_n \models V$, dacă și numai $\forall i: F_P \rightarrow \{T, F\}$ astfel încât $i(U_1 \wedge U_2 \wedge \dots \wedge U_n) = T$, are loc $i(V) = T$. Formulele U_1, U_2, \dots, U_n se numesc *premize, ipoteze, fapte*, iar V se numește *concluzie*.

Teoremă

Fie $S = \{U_1, U_2, \dots, U_n\}$ o mulțime de formule propoziționale.

1. Dacă S este o mulțime consistentă, atunci
 $\forall j, 1 \leq j \leq n, S \setminus \{U_j\}$ este o mulțime consistentă.
2. Dacă S este o mulțime consistentă și V este o formulă validă, atunci mulțimea $S \cup \{V\}$ este consistentă.
3. Dacă S este o mulțime inconsistentă, atunci $\forall V \in F_p$ mulțimea $S \cup \{V\}$ este inconsistentă.
4. Dacă S este o mulțime inconsistentă și U_j este o formulă validă, unde $1 \leq j \leq n$, atunci mulțimea $S \setminus \{U_j\}$ este inconsistentă.

Teoremă

Fie $U_1, U_2, \dots, U_n, U, V$ formule propoziționale.

- $\models U$ dacă și numai dacă $\neg U$ este inconsistentă
(O formulă este tautologie dacă și numai dacă negația sa este o formulă inconsistentă).
- $U \models V$ dacă și numai dacă $\models U \rightarrow V$ dacă și numai dacă mulțimea $\{U, \neg V\}$ este inconsistentă.
- $U \equiv V$ dacă și numai dacă $\models U \leftrightarrow V$.
- $U_1, U_2, \dots, U_n \models V$ dacă și numai dacă $\models U_1 \wedge U_2 \wedge \dots \wedge U_n \rightarrow V$ dacă și numai dacă mulțimea $\{U_1, U_2, \dots, U_n, \neg V\}$ este inconsistentă.

Echivalențe logice în logica propozițională

- Legile lui DeMorgan

$$\neg(U \wedge V) \equiv \neg U \vee \neg V \quad \text{și} \quad \neg(U \vee V) \equiv \neg U \wedge \neg V$$

- Legile de absorbție

$$U \wedge (U \vee V) \equiv U \quad \text{și} \quad U \vee (U \wedge V) \equiv U$$

- Legile de comutativitate

$$U \wedge V \equiv V \wedge U \quad \text{și} \quad U \vee V \equiv V \vee U$$

- Legile de asociativitate

$$U \wedge (V \wedge Z) \equiv (U \wedge V) \wedge Z \quad \text{și} \quad U \vee (V \vee Z) \equiv (U \vee V) \vee Z$$

- Legile de distributivității

$$U \wedge (V \vee Z) \equiv (U \wedge V) \vee (U \wedge Z) \quad \text{și} \\ U \vee (V \wedge Z) \equiv (U \vee V) \wedge (U \vee Z)$$

- Legile de idempotență

$$U \wedge U \equiv U \quad \text{și} \quad U \vee U \equiv U$$

Alte echivalențe logice

- Definirea conectivelor

- Legile de simplificare

$$\neg \neg U \equiv U$$

$$U \rightarrow U \equiv T$$

$$U \wedge \neg U \equiv F$$

$$U \vee \neg U \equiv T$$

$$T \wedge U \equiv U$$

$$F \vee U \equiv U$$

$$U \rightarrow T \equiv T$$

$$U \rightarrow F \equiv \neg U$$

$$T \rightarrow U \equiv U$$

$$F \rightarrow U \equiv T$$

$$U \leftrightarrow T \equiv U$$

$$U \leftrightarrow F \equiv \neg U$$

$$U \oplus T \equiv \neg U$$

$$U \oplus F \equiv U$$

$$U \leftrightarrow U \equiv T$$

$$U \oplus U \equiv F$$

$$U \rightarrow V \equiv \neg U \vee V$$

$$U \rightarrow V \equiv \neg (U \wedge \neg V)$$

$$U \rightarrow V \equiv U \leftrightarrow (U \wedge V)$$

$$U \rightarrow V \equiv V \leftrightarrow (U \vee V)$$

$$U \leftrightarrow V \equiv (U \rightarrow V) \wedge (V \rightarrow U)$$

$$U \oplus V \equiv \neg (U \rightarrow V) \vee \neg (V \rightarrow U)$$

$$U \leftrightarrow V \equiv (U \vee V) \rightarrow (U \wedge V)$$

$$U \vee V \equiv \neg (\neg U \wedge \neg V)$$

$$U \wedge V \equiv \neg (\neg U \vee \neg V)$$

$$U \vee V \equiv \neg U \rightarrow V$$

$$U \wedge V \equiv \neg (U \rightarrow \neg V)$$

$$\neg U \equiv U \uparrow U \equiv U \downarrow U$$

$$U \vee V \equiv (U \uparrow U) \uparrow (V \uparrow V) \equiv (U \downarrow V) \downarrow (U \downarrow V)$$

$$U \wedge V \equiv (U \downarrow U) \downarrow (V \downarrow V) \equiv (U \uparrow V) \uparrow (U \uparrow V)$$

Principiul dualității

- Pentru orice echivalență logică $U \equiv V$ care conține doar conectivele $\neg, \wedge, \vee, \uparrow, \downarrow$ există o altă echivalență logică, $U' \equiv V'$, unde U', V' sunt formule obținute din U, V prin interschimbarea conectivelor logice duale: $(\wedge, \vee), (\uparrow, \downarrow)$ și a valorilor de adevăr: T, F.
- *Conective duale*: $(\wedge, \vee), (\uparrow, \downarrow), (\leftrightarrow, \oplus)$.
- *Valori de adevăr duale*: T și F.
- *Concepțe duale*: tautologie și formulă inconsistentă.

Forme normale în logica propozițiilor

1. Un **literal** este o variabilă propozițională sau negația sa.
2. O **clauză** este disjuncția unui număr finit de literali.
3. Un **cub** este conjuncția unui număr finit de literali.
4. **Clauza vidă**, simbolizată prin \square , este clauza fără literali, fiind singura clauză inconsistentă.
5. O formulă este în **formă normală disjunctivă (FND)**, dacă aceasta este scrisă ca o disjuncție de cuburi $\bigvee_{i=1}^P (\bigwedge_{j=1}^{q_i} l_{ij})$ unde l_{ij} sunt literali.
6. O formulă este în **formă normală conjunctivă (FNC)**, dacă aceasta este scrisă ca o conjuncție de clauze: $\bigwedge_{i=1}^n (\bigvee_{j=1}^{m_i} l_{ij})$ unde l_{ij} sunt literali.

Exemple FN în logica propozițiilor

- Un **literal** este o variabilă propozițională sau negația sa. $p, \neg q$
- O **clauză** este disjuncția unui număr finit de literali. $p \vee \neg q \vee r$
- Un **cub** este conjuncția unui număr finit de literali. $p \wedge \neg q \wedge \neg r$
- O formulă este în **formă normală disjunctivă (FND)**, dacă aceasta este scrisă ca o disjuncție de cuburi.

$$(\neg p \wedge q \wedge r) \vee (\neg p \wedge q \wedge \neg r) \vee (p \wedge q \wedge \neg r)$$

- O formulă este în **formă normală conjunctivă (FNC)**, dacă aceasta este scrisă ca o conjuncție de clauze.

$$(\neg p \vee q \vee r) \wedge (\neg p \vee \neg r) \wedge (p \vee q \vee \neg r) \wedge (p \vee q)$$

Sunt în FNC și/sau FND?

$(\neg p \vee q \vee r) \wedge (\neg p \vee q \vee \neg r) \wedge (p \vee q \vee \neg r)$ FNC, 3 clauze

$(\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg r) \vee (p \wedge q \wedge \neg r) \vee (p \wedge q)$ FND ,

p FNC, 1 clauză; FND , 1 cub 4 cuburi

$\neg q$ FNC, 1 clauză; FND , 1 cub

$\neg p \vee q \vee \neg r$ FND, 3 cuburi; FNC , 1 clauză

$p \wedge \neg q \wedge \neg r$ FNC, 3 clauze; FND , 1 cub

Proprietate

Fie multimea de literali $\{l_1, l_2, \dots, l_n\}$. Următoarele afirmații sunt echivalente:

- clauza $\vee_{i=1}^n l_i$ este validă;
- cubul $\wedge_{i=1}^n l_i$ este inconsistent;
- în mulțimea $\{l_1, l_2, \dots, l_n\}$ există cel puțin o pereche de literali opuși, adică: $\exists i, j \in \{1, \dots, n\}$ astfel încât $l_i = \neg l_j$.

Proprietate - exemple

Fie multimea de literali $\{l_1, l_2, \dots, l_n\}$. Următoarele afirmații sunt echivalente:

- clauza $\vee_{i=1}^n l_i$ este validă;
- cubul $\wedge_{i=1}^n l_i$ este inconsistent;
- în mulțimea $\{l_1, l_2, \dots, l_n\}$ există cel puțin o pereche de literali opuși, adică: $\exists i, j \in \{1, \dots, n\}$ astfel încât $l_i = \neg l_j$.

$$\neg p \vee q \vee p \equiv T$$

$$p \wedge r \wedge \neg r \equiv F$$

Teoremă

- *Orice formulă admite o formă normală conjunctivă și o formă normală disjunctivă logic echivalente cu ea.*

Algoritmul de normalizare

Pas1:

Înlocuirea formulelor de tip $U \rightarrow V$ cu forma echivalentă: $\neg U \vee V$

Înlocuirea formulelor de tip $U \leftrightarrow V$ cu forma echivalentă:

$$(\neg U \vee V) \wedge (\neg V \vee U).$$

Pas2:

Aplicarea legilor lui **DeMorgan** (se recomandă aplicarea dinspre exterior spre interior) ==> negația va preceda doar variabilele propoziționale.

Eliminarea negațiilor multiple folosind echivalența logică: $\neg \neg U \equiv U$.

Pas3: Aplicarea legilor **distributivității**.

Pentru FND

FNC

$$U \wedge (V \vee Z) \equiv (U \wedge V) \vee (U \wedge Z) \text{ respectiv } U \vee (V \wedge Z) \equiv (U \vee V) \wedge (U \vee Z)$$

Pas4: Simplificarea formei obținute folosind alte echivalențe logice: legile de simplificare, legile absorbției, legile de idempotență.

Exemplu de aducere la FNC & FND

$$U = p \rightarrow r \vee \neg(\neg q \vee r)$$

$$U = p \rightarrow (r \vee \neg(\neg q \vee r))$$

Pas1: Înlocuirea formulelor de tip $U \leftrightarrow V$, $U \rightarrow V$ cu forma echivalentă: ..., $\neg U \vee V$

$$U \equiv \neg p \vee (r \vee \neg(\neg q \vee r))$$

Pas2: Aplicarea legilor lui **DeMorgan**

$$U \equiv \neg p \vee (r \vee (\neg \neg q \wedge \neg r))$$

$$U \equiv \neg p \vee (r \vee (q \wedge \neg r))$$

$$U \equiv \neg p \vee r \vee (q \wedge \neg r) \quad - FND$$

Pas3: Aplicarea legilor **distributivității**

$$U \equiv (\neg p \vee r \vee q) \wedge (\neg p \vee \underline{r} \vee \underline{\neg r}) \quad - FNC$$

Pas4: Simplificarea formei obținute folosind alte echivalențe logice

$$U \equiv \neg p \vee r \vee q \quad - FND \& FNC$$

Teoremă

- O formulă în *forma normală conjunctivă (FNC)* este tautologie dacă și numai dacă toate clauzele sale sunt *valide*.
- O formulă în *forma normală disjunctivă (FND)* este inconsistentă dacă și numai dacă toate cuburile sale sunt *inconsistente*.

Observații

- Prima parte a teoremei furnizează o ***metodă directă*** de rezolvare a problemei decizionale (verificarea dacă o formulă este tautologie) în logica propozițiilor.
- FND a unei formule propoziționale furnizează toate modelele formulei inițiale, prin găsirea interpretărilor care evaluatează cuburile componente ca adevărate.
- FNC a unei formule propoziționale furnizează toate anti-modelele formulei inițiale prin găsirea interpretărilor care evaluatează clauzele componente ca false.
- Cele două formulări din teorema precedentă sunt afirmații duale.

Logică computațională

Curs 4

Lector dr. Pop Andreea-Diana

Sisteme axiomatice

- Axiomele geometriei
- Axiomele aritmeticii

Sistemul axiomatic al calculului propozițional

- propus de Hilbert; deductiv, formal
- $P = (\Sigma_P, F_P, A_P, R_P)$
 - $\Sigma_P = Var_propoz \cup Conective \cup \{ (,) \}$
 - $Var_propoz = \{ p, q, r, p_1, p_2, \dots \}$
 - $Conective = \{ \neg, \wedge, \vee, \rightarrow, \leftrightarrow \}$
 - F_P = mulțimea formulelor propoziționale corect construite
 - - *baza*: $p_i \in F_P$, $i=1,2,\dots$
 - - *inducția*: dacă $U, V \in F_P$ atunci:
 $\neg U \in F_P$, $U \wedge V \in F_P$, $U \vee V \in F_P$, $U \rightarrow V \in F_P$, $U \leftrightarrow V \in F_P$
 - - *închiderea*: toate formulele din F_P se obțin doar prin aplicarea regulilor precedente de un număr finit de ori.

Axiome și reguli de inferență

- $A_P = \{A_1, A_2, A_3\}$ scheme axiomatice
 - $A_1: U \rightarrow (V \rightarrow U)$
 - $A_2: (U \rightarrow (V \rightarrow Z)) \rightarrow ((U \rightarrow V) \rightarrow (U \rightarrow Z))$
 - $A_3: (U \rightarrow V) \rightarrow (\neg V \rightarrow \neg U)$
- $R_P = \{m_P\}$ – o singură regulă de inferență „*modus ponens*”
 - $U, U \rightarrow V \vdash V$
„din faptele U și $U \rightarrow V$ se deduce (inferă) V ”

Definiția deducției

- Fie formulele U_1, U_2, \dots, U_n numite ipoteze și V formulă propozițională. Spunem că V este **deductibilă din U_1, U_2, \dots, U_n** și notăm $U_1, U_2, \dots, U_{n-1}, U_n \vdash V$, dacă există o secvență de formule (f_1, f_2, \dots, f_m) astfel încât $f_m = V$ și $\forall i \in \{1, \dots, m\}$ avem:
 - $f_i \in A_p$;
 - $f_i \in \{U_1, U_2, \dots, U_n\}$;
 - $f_j, f_k \vdash_{m_p} f_i$, $j < i$ și $k < i$
- Secvența (f_1, f_2, \dots, f_m) se numește **deducția lui V din U_1, U_2, \dots, U_n** .

Noțiunea de teoremă

- **Definiția 1.8.** O formulă $U \in F_P$, astfel încât $\emptyset \vdash U$ (sau $\vdash U$) se numește *teoremă*.
- **Observație:** Teoremele sunt formule care sunt deductibile doar din axiome și folosind regula modus ponens.

Teorema de deducție și inversa sa

- **Teorema de deducție**

Dacă $U_1, U_2, \dots, U_{n-1}, U_n \vdash V$, atunci $U_1, U_2, \dots, U_{n-1} \vdash U_n \rightarrow V$.

- **Inversa teoremei de deducție**

Dacă $U_1, U_2, \dots, U_{n-1} \vdash U_n \rightarrow V$ atunci $U_1, U_2, \dots, U_{n-1}, U_n \vdash V$.

Generalizarea

$U_1, U_2, \dots, U_{n-1}, U_n \vdash V$ dacă și numai dacă

$U_1, U_2, \dots, U_{n-1} \vdash U_n \rightarrow V$ dacă și numai dacă

$U_1, U_2, \dots, U_{n-2} \vdash U_{n-1} \rightarrow (U_n \rightarrow V)$ dacă și numai dacă

...

$U_1 \vdash U_2 \rightarrow (\dots U_{n-1} \rightarrow (U_n \rightarrow V) \dots)$ dacă și numai dacă

$\vdash U_1 \rightarrow (U_2 \rightarrow (\dots U_{n-1} \rightarrow (U_n \rightarrow V) \underbrace{\dots}_{n-1}))$

Consecințele teoremei de deducție

- $\vdash U \rightarrow ((U \rightarrow V) \rightarrow V)$
- $\vdash (U \rightarrow V) \rightarrow ((V \rightarrow Z) \rightarrow (U \rightarrow Z))$ legea silogismului
- $\vdash (U \rightarrow (V \rightarrow Z)) \rightarrow (V \rightarrow (U \rightarrow Z))$ legea permutării premizelor
- $\vdash (U \rightarrow (V \rightarrow Z)) \rightarrow (U \wedge V \rightarrow Z)$ legea reuniunii premizelor
- $\vdash (U \wedge V \rightarrow Z) \rightarrow (U \rightarrow (V \rightarrow Z))$ legea separării premizelor

Proprietățile logicii propozițiilor

- **Problemele decizionale** în logica propozițiilor:
 - „Este o formulă propozițională o teoremă sau nu?”
 - „Este o formulă deductibilă dintr-o mulțime de formule?”
- **Teorema de corectitudine**

Dacă $\vdash U$ atunci $\models U$. (Validitatea sintactică implică validitatea semantică)
- **Teorema de completitudine**

Dacă $\models U$ atunci $\vdash U$. (Validitatea semantică implică validitatea sintactică)
- **Teorema de corectitudine și completitudine**

$\vdash U$ dacă și numai dacă $\models U$.

Consecințele teoremei de corectitudine și completitudine

1) *Logica propozițiilor este necontradicțorie:*

nu pot avea loc simultan $\vdash U$ și $\vdash \neg U$.

2) *Logica propozițiilor este coerentă:*

nu orice formulă propozițională este teoremă.

3) *Logica propozițiilor este decidabilă:*

se poate decide dacă o formulă propozițională este sau nu teoremă.



Logică computațională

Curs 5

Lector dr. Pop Andreea-Diana

Metoda tabelelor semantice

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

Clase de formule

- clasa α - formule de tip conjunctiv
- clasa β - formule de tip disjunctiv

$$A \wedge B$$

$$A \vee B$$

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

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

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

$$A \rightarrow B$$

Reguli de descompunere a formulelor

- regula α

$$\begin{array}{ccc} A \wedge B & \neg(A \vee B) & \neg(A \rightarrow B) \\ | & | & | \\ A & \neg A & A \\ | & | & | \\ B & \neg B & \neg B \end{array}$$

- regula β

$$\begin{array}{c} A \vee B \\ \swarrow \quad \searrow \\ A & B \\ \neg(A \wedge B) \\ \swarrow \quad \searrow \\ \neg A & \neg B \end{array}$$

$$\begin{array}{c} A \rightarrow B \\ \swarrow \quad \searrow \\ \neg A & B \end{array}$$

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

Tipuri de ramuri

- O *ramură* a tabelei se numește *închisă* (simbolizată prin \otimes) dacă ea conține o formulă și negația ei, în caz contrar *ramura* se numește *deschisă* (simbolizată prin \odot).
- O *ramură* a tabelei 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.

Exemplul (2)

$$U = (p \vee q) \wedge \neg(q \rightarrow r) \wedge (p \rightarrow q \wedge r) \quad (1)$$

| – regula α pentru (1)

$$p \vee q \quad (2)$$

$$\neg(q \rightarrow r) \quad (3)$$

$$p \rightarrow q \wedge r \quad (4)$$

– regula β pentru (2)

$$\text{regula } \alpha \text{ pentru (3)} - \boxed{p}$$

$$q$$

$$q \quad | - \text{regula } \alpha \text{ pentru (3)}$$

$$\text{regula } \beta \text{ pentru (4)} - \boxed{\neg r}$$

$$\neg r$$

$$q \wedge r \quad (5)$$

$$\text{regula } \alpha \text{ pentru (5)} - \otimes \boxed{\neg p}$$

$$q$$

$$r$$

$$\otimes$$

$$\neg r \quad | - \text{regula } \beta \text{ pentru (4)}$$

$$\neg r$$

$$q \wedge r \quad (6)$$

$$\text{regula } \alpha \text{ pentru (6)} - \otimes \boxed{q}$$

$$q$$

$$r$$

$$\otimes$$

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;

Observații (2):

- formulele de pe aceeași ramură a unei tabele 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 *formei sale normale disjunctive*. Fiecare ramură reprezintă un *cub* (conjuncția tuturor literalilor de pe acea ramură), iar arborele este *disjuncția* tuturor *ramurilor* sale.
- Unei formule *consistentă* i se asociază o *tabelă completă deschisă*, iar fiecare *ramură deschisă* a tabelei furnizează cel puțin un *model* pentru formula respectivă.
- O *tabelă semantică închisă* asociată unei formule indică faptul că formula este *inconsistenta*, 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 $U_1 \wedge U_2 \wedge \dots \wedge U_n \wedge \neg Y$.

Logică computațională

Curs 6

Lector dr. Pop Andreea-Diana

Metoda rezoluției (Robinson, 1965)

- metodă de demonstrare automată *sintactică*, prin *respingere*
- este o metodă corectă și completă de demonstrare automată
- verificarea *consistenței/inconsistenței* unei multimi de clauze (scop)

Sistem formal (axiomatic) asociat Rezoluției propoziționale

- $\text{Res} = (\Sigma_{\text{Res}}, F_{\text{Res}}, A_{\text{Res}}, R_{\text{Res}})$
 - $\Sigma_{\text{Res}} = \Sigma_P \setminus \{\wedge, \rightarrow, \leftrightarrow\}$ – **alfabetul**
 - $F_{\text{Res}} \cup \{\square\}$ – **mulțimea formulelor bine-formate**
 - F_{Res} mulțimea tuturor clauzelor ce se pot forma folosind alfabetul Σ_{Res}
 - \square - **clauza vidă** care nu conține nici un literal, simbolizează inconsistență
 - $A_{\text{Res}} = \emptyset$ – **mulțimea axiomelor**
 - $R_{\text{Res}} = \{res\}$ **mulțimea regulilor de inferență** care conține doar
 - **regula rezoluției:** $A \vee l, B \vee \neg l \vdash_{\text{res}} A \vee B$, unde l este un literal, iar $A, B \in F_{\text{Res}}$

Terminologie

- clauzele $C_1 = A \vee l$, $C_2 = B \vee \neg l$ **rezolvă** deoarece conțin doi literali opuși (complementari)
- **Notație:** $C_3 = \text{Res}_l(C_1, C_2)$
- C_3 **rezolventul** clauzelor C_1 și C_2
- clauzele C_1 , C_2 **clauze părinte**
- caz particular: $C_1 = l$, $C_2 = \neg l$, $\text{Res}_l(C_1, C_2) = \square$ - inconsistentă

Observație:

- Rezoluția ca și regulă de inferență este o generalizare a regulilor *modus ponens*, *modus tollens* și a *silogismului*.
- *mp*: $U, U \rightarrow V \vdash V \Rightarrow U, \neg U \vee V \vdash V$
- *mt*: $U \rightarrow V \vdash \neg V \rightarrow \neg U \stackrel{\text{ITD}}{\Rightarrow} U \rightarrow V, \neg V \vdash \neg U \Rightarrow \neg U \vee V, \neg V \vdash \neg U$
- *sil.*: $U \rightarrow V, V \rightarrow Z \vdash U \rightarrow Z \Rightarrow \neg U \vee V, \neg V \vee Z \vdash \neg U \vee Z$

Algoritmul rezoluției propoziționale:

Date de intrare: S – o mulțime de clauze

Date de ieșire: S consistentă sau inconsistentă

$$S_0 = S$$

$$i = 0$$

Repetă

@ se aleg două clauze $C_1, C_2 \in S$ care rezolvă

$$C_3 = \text{Res}(C_1, C_2)$$

$$S_{i+1} = S_i \cup \{C_3\}$$

Dacă $C_3 = \square$

Atunci Scrie ” S este inconsistentă”; **STOP**

Altfel $i = i + 1$

Sfârșit_dacă

Până când $S_i = S_{i-1}$ //nu se mai pot deriva clauze noi

Scrie ” S este consistentă”

Sfârșit algoritm

Notație:

- $S \vdash_{\text{Res}} \square$ ”din mulțimea S de clauze s-a derivat clauza vidă prin aplicarea algoritmului rezoluției propoziționale”

Teorema de corectitudine și completitudine

- Teorema de corectitudine

Dacă $S \vdash_{\text{Res}} \square$ atunci S este inconsistentă.

- Teorema de completitudine

Dacă S este inconsistentă atunci $S \vdash_{\text{Res}} \square$.

- Teorema de corectitudine și completitudine

Mulțimea S este inconsistentă dacă și numai dacă $S \vdash_{\text{Res}} \square$.

Teoreme

- U este tautologie dacă și numai dacă $FNC(\neg U) \vdash_{\text{Res}} \square$
- $U_1, U_2, \dots, U_n \models V$ dacă și numai dacă
 $U_1, U_2, \dots, U_n \not\models V$ dacă și numai dacă
 $FNC(U_1 \wedge U_2 \wedge \dots \wedge U_n \wedge \neg V) \vdash_{\text{Res}} \square$

$$S_i \stackrel{\text{not.}}{=} FNC(U_i), i = \overline{1, n}$$

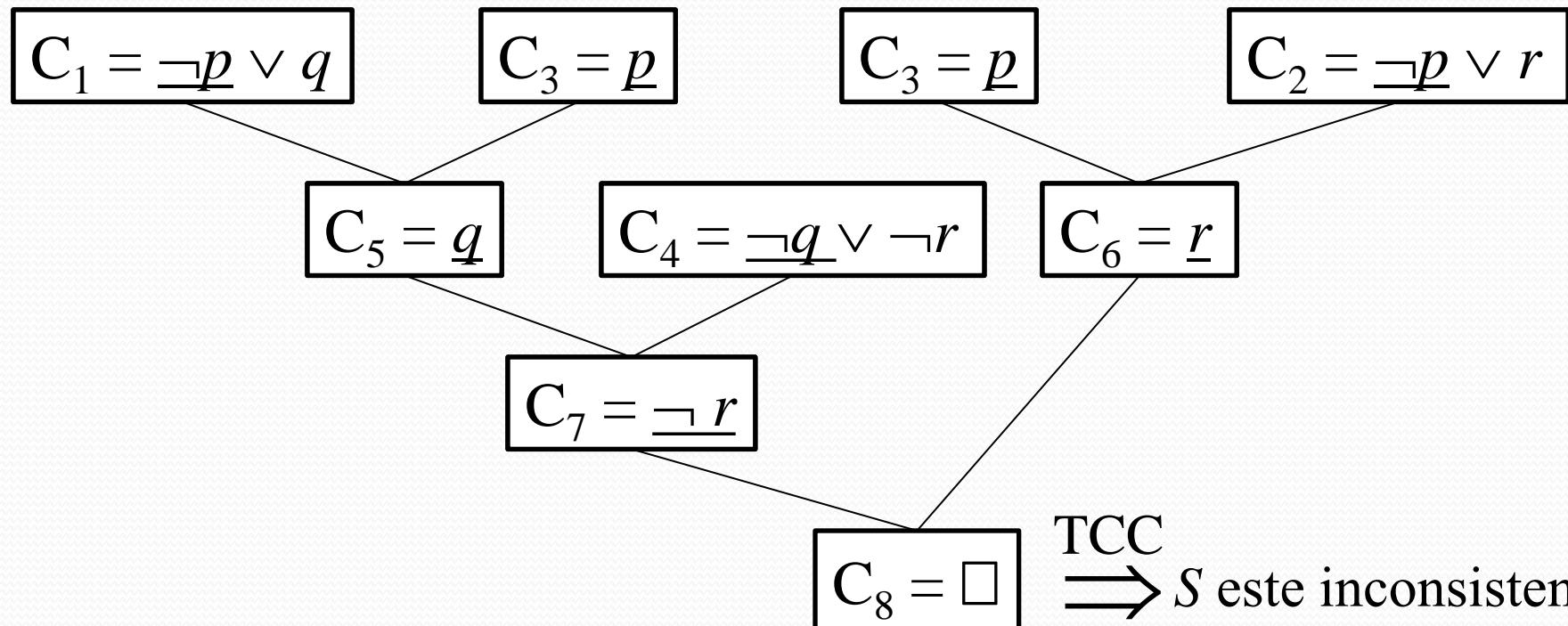
$$S_{n+1} \stackrel{\text{not.}}{=} FNC(\neg V)$$

$$S_1 \cup S_2 \cup \dots \cup S_n \cup S_{n+1} \vdash_{\text{Res}} \square$$

Exemplu

$$S = \{\neg p \vee q, \neg p \vee r, p, \neg q \vee \neg r\}$$

$$C_1 \stackrel{\text{not.}}{=} \neg p \vee q, \quad C_2 \stackrel{\text{not.}}{=} \neg p \vee r, \quad C_3 \stackrel{\text{not.}}{=} p, \quad C_4 \stackrel{\text{not.}}{=} \neg q \vee \neg r$$



Problemă propusă

Ion, Vasile și Gheorghe sunt studenți la Informatică. Ei au făcut următorul pact:

- Dacă Ion își face laboratorul atunci Ion sau Vasile au laboratorul făcut.
 - Dacă Ion își face laboratorul, atunci și Vasile își face laboratorul.
 - Dacă Vasile nu-și face laboratorul, atunci Gheorghe îl va face.
 - Dacă Gheorghe nu își face laboratorul, atunci nu își face nici Ion laboratorul sau Vasile îl face.
 - Știm că Gheorghe nu și-a făcut laboratorul.
1. Să se verifice dacă pactul nu este contradictoriu
 2. Să se verifice dacă din cele de mai sus se poate deduce că Vasile a făcut laboratorul.

Notății

Ion și-a făcut lab. $\stackrel{\text{not.}}{=} p$,

Vasile și-a făcut lab. $\stackrel{\text{not.}}{=} q$,

Gheorghe și-a făcut lab. $\stackrel{\text{not.}}{=} r$.

- Dacă Ion își face laboratorul atunci Ion sau Vasile au laboratorul făcut.
- Dacă Ion își face laboratorul, atunci și Vasile își face laboratorul.
- Dacă Vasile nu-și face laboratorul, atunci Gheorghe îl va face.
- Dacă Gheorghe nu își face laboratorul, atunci nu își face nici Ion laboratorul sau Vasile îl face.
- Știm că Gheorghe nu și-a făcut laboratorul.
? Vasile a făcut laboratorul.

Transformare limbaj natural în limbaj logic

Ion și-a făcut lab. $\stackrel{\text{not.}}{=} p$,

Vasile și-a făcut lab. $\stackrel{\text{not.}}{=} q$,

Gheorghe și-a făcut lab. $\stackrel{\text{not.}}{=} r$.

- Dacă Ion își face laboratorul atunci Ion sau Vasile au laboratorul făcut. $p \rightarrow p \vee q$
- Dacă Ion își face laboratorul, atunci și Vasile își face laboratorul $p \rightarrow q$
- Dacă Vasile nu-și face laboratorul, atunci Gheorghe îl va face.

$$\neg q \rightarrow r$$

- Dacă Gheorghe nu își face laboratorul, atunci nu își face nici Ion laboratorul sau Vasile îl face. $\neg r \rightarrow \neg p \vee q$
- Știm că Gheorghe nu și-a făcut laboratorul. $\neg r$

$$? \text{ Vasile a făcut laboratorul. } q$$

În limbaj logic

- $p \rightarrow p \vee q$
- $p \rightarrow q$
- $\neg q \rightarrow r$
- $\neg r \rightarrow \neg p \vee q$
- $\neg r$

? q

Obținerea multimilor de clauze

- $p \rightarrow p \vee q \equiv \neg p \vee p \vee q \stackrel{\text{not.}}{=} C_1$
- $p \rightarrow q \equiv \neg p \vee q \stackrel{\text{not.}}{=} C_2$
- $\neg q \rightarrow r \equiv q \vee r \stackrel{\text{not.}}{=} C_3$
- $\neg r \rightarrow \neg p \vee q \equiv r \vee \neg p \vee q \stackrel{\text{not.}}{=} C_4$
- $\neg r \stackrel{\text{not.}}{=} C_5$
 $? q \qquad \qquad \qquad \Rightarrow \qquad \qquad \qquad \neg q \stackrel{\text{not.}}{=} C_6$
se neagă concluzia

$$S_1 = \{C_1, C_2, C_3, C_4, C_5\}$$

$$S_2 = \{C_1, C_2, C_3, C_4, C_5, C_6\}$$

Aplicarea metodei rezoluției

$$\begin{aligned} S_1 &= \{ C_1, C_2, C_3, C_4, C_5 \} = \\ &= \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r \} \end{aligned}$$

$$\begin{aligned} S_2 &= \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ &= \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \} \end{aligned}$$

Aplicarea metodei rezoluției pt. S_1 :

$$S_1 = \{ C_1, C_2, C_3, C_4, C_5 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r \}$$

$$C_6 = \text{Res}_r(C_3, C_5) = q$$

$$C_2 = \text{Res}_p(\overline{C_4}, \overline{C_2}) = \neg p \vee \neg q \text{ nu s-a obținut o clauză nouă}$$

$$C_4 = \text{Res}_p(\overline{C_4}, \overline{C_4}) = \neg p \vee p \vee q \text{ nu s-a obținut o clauză nouă}$$

$$C_2 = \text{Res}_r(\overline{C_4}, \overline{C_5}) = \neg p \vee \neg q \text{ nu s-a obținut o clauză nouă}$$

Abandonăm fără a ajunge la o concluzie, am constatat că avem nevoie de o strategie.

Deci nu știm încă dacă ipotezele sunt contradictorii.

Aplicarea metodei rezoluției pt. S_2 :

$$S_2 = \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \}$$

$$C_7 = \text{Res}_r(C_3, C_5) = q_{\text{TCC}}$$

$C_8 = \text{Res}_r(C_7, C_6) = \square \Rightarrow S_2$ e inconsistentă adică, Vasile își face laboratorul

Automatizarea procesului rezolutiv

- prin intermediul unor *strategii*
 - asigură exploatarea tuturor modurilor posibile de derivare a clauzei vide
 - evitarea deducerii unor clauze redundante sau irelevante pentru obținerea \square

Strategia eliminării

- inspirată din procedura Davis-Putman
- O mulțime S de clauze poate fi simplificată, păstrând consistența/inconsistența ei prin aplicarea următoarelor transformări:
 - **Eliminarea clauzelor tautologice** (nu pot contribui la derivarea clauzei vide): $\neg p \vee q \vee p \vee \neg r$
 - **Eliminarea clauzelor subsumate de alte clauze din S** : clauza C_1 este subsumată de C_2 dacă există o clauză C_3 astfel încât $C_1 = C_2 \vee C_3$:
 $\neg p \vee q \vee r$ este subsumată de $\neg p \vee q$
 - **Eliminarea clauzelor care conțin literali puri în S** : Un literal este pur dacă negația sa nu apare în nici o clauză din S :
 $\{\neg p \vee q, \neg p \vee r, p, \neg q \vee r\}$
- Dacă $C=l$ este o **clauză unitate** din S , se șterg **toate clauzele care-l conțin pe l și $\neg l$ din clauzele rămase.**
 $\{\neg p \vee q, \neg p \vee r, \neg \neg q \vee \neg r, p \vee \neg r\} \rightarrow \{q, r, \neg q \vee \neg r\}$
sau $\{\square\}$

Strategia eliminării aplicată la S_1 :

$$S_1 = \{\neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r\}$$

- **Eliminarea clauzelor tautologice** (nu pot contribui la derivarea clauzei vide):

$$S_1 = \{\neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r\}$$

- **Eliminarea clauzelor subsumate de alte clauze din S :** clauza C_1 este subsumată de C_2 dacă există o clauză C_3 astfel încât $C_1 = C_2 \vee C_3$:

$$S_1 = \{\neg p \vee q, q \vee r, \neg r\}$$

- **Eliminarea clauzelor care conțin literali puri în S :** Un literal este pur dacă negația sa nu apare în nici o clauză din S :

$$S_1 = \{q \vee r, \neg r\}$$

$$S_1 = \{\neg r\}$$

$S_1 = \{\} = \emptyset$ - consistentă, deci pactul studenților este necontradictoriu

Strategia eliminării aplicată la S_2 :

$$S_2 = \{\neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q\}$$

- **Eliminarea clauzelor tautologice** (nu pot contribui la derivarea clauzei vide):

$$S_2 = \{\neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q\}$$

- **Eliminarea clauzelor subsumate de alte clauze din S :** clauza C_1 este subsumată de C_2 dacă există o clauză C_3 astfel încât $C_1 = C_2 \vee C_3$:

$$S_2 = \{\neg p \vee q, q \vee r, \neg r, \neg q\}$$

- **Eliminarea clauzelor care conțin literali puri în S :** Un literal este pur dacă negația sa nu apare în nici o clauză din S :

$$S_2 = \{q \vee r, \neg r, \neg q\}$$

- Dacă $C=l$ este o clauză unitate din S , se șterg toate clauzele care-l conțin pe l și $\neg l$ din clauzele rămase.

$$S_2 = \{\textcolor{red}{q}, \neg \textcolor{green}{q} \vee \square\}$$

$S_2 = \{ \square \}$ – inconsistentă, deci Vasile și-a făcut laboratorul

Strategia saturării pe nivele (algoritmul)

Date de intrare: S – o mulțime de clauze

Date de ieșire: S consistentă sau inconsistentă

//Se generează mulțimile de clauze S^0, S^1, \dots, S^k ce reprezintă nivelele

$$S^0 = S$$

$$k = 0$$

Repetă

$$k = k + 1$$

$$S^k = \{ \text{Res} (C_1, C_2) \mid C_1 \in S^0 \cup S^1 \cup \dots \cup S^{k-1}, C_2 \in S^{k-1} \}$$

$$S^k = S^k \setminus (S^0 \cup S^1 \cup \dots \cup S^{k-1})$$

Până când $\square \in S^k$ sau $S^k = \emptyset$

Dacă $\square \in S^k$

Atunci Scrie ” S este inconsistentă”;

Altfel Scrie ” S este consistentă”

Sfârșit_dacă

Sfârșit algoritm

Strategia saturării pe nivele aplicată la S_1 (1):

$$S_1^0 = \{ C_1, C_2, C_3, C_4, C_5 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r \}$$

$$C_2 = \text{Res}_p(C_1, C_2) = \neg p \vee q$$

$$\text{Res}_?(C_1, C_3) \text{ NU}$$

$$C_4 = \text{Res}_p(C_1, C_4) = \neg p \vee q \vee r$$

$$\text{Res}_?(C_1, C_5) \text{ NU}$$

$$\text{Res}_?(C_2, C_3) \text{ NU}$$

$$\text{Res}_?(C_2, C_4) \text{ NU}$$

$$\text{Res}_?(C_2, C_5) \text{ NU}$$

$$\text{Res}_?(C_3, C_4) \text{ NU}$$

$$C_6 = \text{Res}_r(C_3, C_5) = q$$

$$C_2 = \text{Res}_r(C_4, C_5) = \neg p \vee q$$

$$S_1^1 = \{ C_6 \} = \{ q \}$$

Strategia saturării pe nivele aplicată la S_1 (2):

$$S_1^0 = \{ C_1, C_2, C_3, C_4, C_5 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r \}$$

$$S_1^1 = \{ C_6 \} = \{ q \}$$

$$\text{Res}_? (C_6, C_1) \text{ NU}$$

$$\text{Res}_? (C_6, C_2) \text{ NU}$$

$$\text{Res}_? (C_6, C_3) \text{ NU}$$

$$\text{Res}_? (C_6, C_4) \text{ NU}$$

$$\text{Res}_? (C_6, C_5) \text{ NU}$$

$$S_1^2 = \{ \} = \emptyset \Rightarrow S_1 \text{ este consistent}$$

Strategia saturării pe nivele aplicată la S_2 (1):

$$S_2 = \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \}$$

$$C_2 = \text{Res}_p(C_1, C_2) = \neg p \vee q$$

$$\text{Res}_?(C_1, C_3) \text{ NU}$$

$$C_4 = \text{Res}_p(C_1, C_4) = \neg p \vee q \vee r$$

$$\text{Res}_?(C_1, C_5) \text{ NU}$$

$$C_7 = \text{Res}_q(C_1, C_6) = \neg p \vee p$$

$$\text{Res}_?(C_2, C_3) \text{ NU}$$

$$\text{Res}_?(C_2, C_4) \text{ NU}$$

$$\text{Res}_?(C_2, C_5) \text{ NU}$$

$$C_8 = \text{Res}_q(C_2, C_6) = \neg p$$

$$\text{Res}_?(C_3, C_4) \text{ NU}$$

$$C_9 = \text{Res}_r(C_3, C_5) = q$$

$$C_{10} = \text{Res}_q(C_3, C_6) = r$$

$$C_{12} = \text{Res}_r(C_4, C_5) = \neg p \vee q$$

$$C_{11} = \text{Res}_q(C_4, C_6) = r \vee \neg p$$

$$\text{Res}_?(C_5, C_6) \text{ NU}$$

$$S_2^1 = \{ C_7, C_8, C_9, C_{10}, C_{11} \} = \{ \neg p \vee p, \neg p, q, r, r \vee \neg p \}$$

Strategia saturării pe nivele aplicată la S_2 (1):

$$S_2 = \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \}$$

$$C_2 = \text{Res}_p(C_1, C_2) = \neg p \vee q$$

$$\text{Res}_?(C_1, C_3) \text{ NU}$$

$$C_4 = \text{Res}_p(C_1, C_4) = \neg p \vee q \vee r$$

$$\text{Res}_?(C_1, C_5) \text{ NU}$$

$$C_7 = \text{Res}_q(C_1, C_6) = \neg p \vee p$$

$$\text{Res}_?(C_2, C_3) \text{ NU}$$

$$\text{Res}_?(C_2, C_4) \text{ NU}$$

$$\text{Res}_?(C_2, C_5) \text{ NU}$$

$$C_8 = \text{Res}_q(C_2, C_6) = \neg p$$

$$\text{Res}_?(C_3, C_4) \text{ NU}$$

$$C_9 = \text{Res}_r(C_3, C_5) = q$$

$$C_{10} = \text{Res}_q(C_3, C_6) = r$$

$$C_{12} = \text{Res}_r(C_4, C_5) = \neg p \vee q$$

$$C_{11} = \text{Res}_q(C_4, C_6) = r \vee \neg p$$

$$\text{Res}_?(C_5, C_6) \text{ NU}$$

$$S_2^1 = \{ C_7, C_8, C_9, C_{10}, C_{11} \} = \{ \neg p \vee p, \neg p, q, r, r \vee \neg p \}$$

Strategia saturării pe nivele aplicată la $S_2(2)$:

$$S_2^0 = \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \}$$

$$S_2^1 = \{ C_7, C_8, C_9, C_{10}, C_{11} \} = \\ = \{ \neg p \vee p, \neg p, q, r, r \vee \neg p \}$$

$$C_1 = \text{Res}_p(C_7, C_1) = \neg p \vee p \vee q$$

$$C_2 = \text{Res}_p(C_7, C_2) = \neg p \vee q$$

$$\text{Res}_? (C_7, C_3) \text{ NU}$$

$$C_4 = \text{Res}_p(C_7, C_4) = r \vee \neg p \vee q$$

$$\text{Res}_? (C_7, C_5) \text{ NU}$$

$$\text{Res}_? (C_7, C_6) \text{ NU}$$

$$C_8 = \text{Res}_p(C_7, C_8) = \neg p$$

$$\text{Res}_? (C_7, C_9) \text{ NU}$$

$$\text{Res}_? (C_7, C_{10}) \text{ NU}$$

$$C_{11} = \text{Res}_p(C_7, C_{11}) = r \vee \neg p$$

$$C_2 = \text{Res}_p(C_8, C_1) = \neg p \vee q$$

$$\text{Res}_? (C_8, C_2) \text{ NU}$$

$$\text{Res}_? (C_8, C_3) \text{ NU}$$

$$\text{Res}_? (C_8, C_4) \text{ NU}$$

$$\text{Res}_? (C_8, C_3) \text{ NU}$$

$$\text{Res}_? (C_8, C_4) \text{ NU}$$

$$\text{Res}_? (C_8, C_5) \text{ NU}$$

$$\text{Res}_? (C_8, C_6) \text{ NU}$$

$$\text{Res}_? (C_8, C_9) \text{ NU}$$

$$\text{Res}_? (C_8, C_{10}) \text{ NU}$$

$$\text{Res}_? (C_8, C_{11}) \text{ NU}$$

$$\text{Res}_? (C_9, C_1) \text{ NU}$$

$$\text{Res}_? (C_9, C_2) \text{ NU}$$

$$\text{Res}_? (C_9, C_3) \text{ NU}$$

$$\text{Res}_? (C_9, C_4) \text{ NU}$$

$$\text{Res}_? (C_9, C_5) \text{ NU}$$

$$C_{12} = \text{Res}_p(C_9, C_6) = \square$$

$\Rightarrow S_2$ este inconsistentă

Strategia mulțimii suport

- se evită aplicarea regulii de rezoluție asupra unor clauze dintr-o submulțime *consistentă* a mulțimii inițiale de clauze, deoarece rezolvenții obținuți sunt *irrelevanti* în procesul de derivare a \square
- Această strategie a fost inspirată din faptul următor: în general mulțimea *premizelor* (faptelor) unei deducții este *consistentă*, deci rezolvarea unor clauze din această mulțime consistentă nu poate duce la derivarea clauzei vide (inconsistență)
- **Definiție:** Fie S o mulțime de clauze. O submulțime Y a lui S se numește *mulțime suport* a lui S , dacă $S \setminus Y$ este consistentă.
Rezoluția mulțimii suport este rezoluția a două clauze care nu aparțin ambele mulțimii $S \setminus Y$.

Aplicarea strategiei multimii suport pt. S_2 :

$$S_2 = \{ C_1, C_2, C_3, C_4, C_5, C_6 \} = \\ = \{ \neg p \vee p \vee q, \neg p \vee q, q \vee r, r \vee \neg p \vee q, \neg r, \neg q \}$$

$Y = \{C_6\}$ mulțimea suport a lui S_2 . este formată din clauzele obținute din concluzie.

$S_2 \setminus Y = S_1$ și am demonstrat deja că S_1 este consistentă.

Așadar nu vom rezolva clauzele S_1 din între ele.

Vom începe de la o clauză din Y și nu vom rezolva deloc primele 5 clauze între ele:

$$C_7 = \text{Res}_r(C_5, C_3) = q_{\text{TCC}}$$

$C_8 = \text{Res}_r(C_6, C_7) = \square \Rightarrow S_2$ e inconsistentă adică, Vasile își face laboratorul

Logică computațională

Curs 7

Lector dr. Pop Andreea-Diana

Rafinările rezoluției

- impun restricții asupra clauzelor care rezolvă, pentru a eficientiza procesul rezolutiv

Notație

- $S \vdash_{\text{Res}}^{st} \square$ ”din multimea S de clauze s-a derivat clauza vidă prin aplicarea strategiei st a rezoluției propoziționale”

Compleitudinea și corectitudinea

- Toate rafinările și strategiile rezolutive păstrează completitudinea și corectitudinea.
- Combinarea lor poate impune prea multe restricții și deși multimea inițială de clauze este inconsistentă, s-ar putea să nu se poată deriva clauza vidă.
- sunt complete:
 - rezoluția generală + strategia eliminării
 - rezoluția generală + strategia mulțimii suport
 - rezoluția generală + strategia mulțimii suport + strategia eliminării
 - rezoluția liniară + strategia eliminării
 - rezoluția liniară + strategia mulțimii suport
- nu sunt complete:
 - rezoluția blocării + strategia eliminării
 - rezoluția blocării + strategia mulțimii suport
 - rezoluția blocării + rezoluția liniară
 - rezoluția unitară
 - rezoluția de intrare

Rezoluția blocării (lock resolution)

- introdusă de Boyer în 1971
- fiecare apariție de literal din multimea de clauze este indexat arbitrar cu un întreg
- *restricția*: literalii care rezolvă din clauzele părinți trebuie să aibă **cei mai mici indici** din aceste clauze
- literalii din rezolvenți moștenesc indicii de la clauzele părinți, iar în cazul moștenirii a doi literali identici, se păstrează cel cu indicele mai mic
- este foarte eficientă și ușor de implementat, se recomandă combinarea ei cu strategia saturării pe nivele

Teorema de corectitudine și completitudine

- Teorema de completitudine

Fie S o mulțime de clauze în care fiecare literal este indexat în mod arbitrar cu un întreg. Dacă S este inconsistentă, atunci există o deducție din mulțimea S a clauzei vide prin rezoluția blocării.

- Teorema de corectitudine

Fie S o mulțime de clauze în care fiecare literal este indexat în mod arbitrar cu un întreg. Dacă din S se deduce prin rezoluția blocării clauza vidă, atunci S este inconsistentă.

Exemplu (1)

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția blocării:

- $S = \{ \neg r, p \vee \neg q, r \vee p \vee q, \neg q \vee \neg p \}$

Rezolvare

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția blocării:

$$S = \{(1) \neg r, (6) p \vee (2) \neg q, (8) r \vee (3) p \vee (5) q, (4) \neg q \vee (7) \neg p\}$$

$$C_1 \stackrel{\text{not.}}{=} (1) \neg r$$

$$C_2 \stackrel{\text{not.}}{=} (2) \neg q \vee (6) p$$

$$C_3 \stackrel{\text{not.}}{=} (3) p \vee (5) q \vee (8) r$$

$$C_4 \stackrel{\text{not.}}{=} (4) \neg q \vee (7) \neg p$$

$\xrightarrow{\text{TCC}}$ S este consistentă

Exemplu (2)

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția blocării:

- $S = \{ \neg r, p \vee \neg q, r \vee p \vee q, \neg q \vee \neg p \}$
- $S = \{ p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q \}$

Rezolvare

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția blocării:

$$S = \{ {}_{(5)} p \vee {}_{(4)} q, {}_{(1)} \neg p \vee {}_{(6)} q, {}_{(7)} p \vee {}_{(2)} \neg q, {}_{(8)} \neg p \vee {}_{(3)} \neg q \}$$

$$C_1 \stackrel{\text{not.}}{=} {}_{(4)} q \vee {}_{(5)} p$$

$$C_5 = \text{Res}_q^{\text{lock}}(C_1, C_3) = {}_{(5)} p$$

$$C_2 \stackrel{\text{not.}}{=} {}_{(1)} \neg p \vee {}_{(6)} q$$

$$C_6 = \text{Res}_p^{\text{lock}}(C_2, C_5) = {}_{(6)} q$$

$$C_3 \stackrel{\text{not.}}{=} {}_{(2)} \neg q \vee {}_{(7)} p$$

$$C_7 = \text{Res}_q^{\text{lock}}(C_4, C_6) = {}_{(8)} \neg p$$

$$C_4 \stackrel{\text{not.}}{=} {}_{(3)} \neg q \vee {}_{(8)} \neg p$$

$$C_8 = \text{Res}_p^{\text{lock}}(C_5, C_7) = \square$$

$\xrightarrow{\text{TCC}}$ S este inconsistentă

Observație!!!!

$$C_1 \stackrel{\text{not.}}{=} q \vee p$$

$$C_4 \stackrel{\text{not.}}{=} \neg q \vee \neg p$$

$$A \vee l, B \vee \neg l \vdash_{res} A \vee B$$

$$\text{Res}_p(C_1, C_4) = q \vee \neg q \equiv \text{T}$$

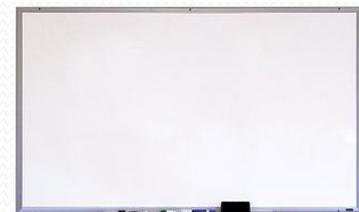
$$\text{Res}_q(C_1, C_4) = p \vee \neg p \equiv \text{T}$$

$$\square \equiv \mathbf{F}$$

$$\mathbf{U} \wedge \neg \mathbf{U} \equiv \mathbf{F} \quad \neg(q \vee p) \equiv \neg q \wedge \neg p \neq \neg q \vee \neg p$$

Exerciții (1)

- rezoluția blocării + strategia eliminării nu e completă
 - $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$
 - $C_1 =_{(2)} p \vee_{(1)} q, C_2 =_{(3)} \neg p \vee_{(4)} q, C_3 =_{(5)} p \vee_{(6)} \neg q, C_4 =_{(8)} \neg p \vee_{(7)} \neg q$



Exerciții (1)

- rezoluția blocării + strategia eliminării nu e completă

- $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$

$$C_1 = {}_{(2)}p \vee {}_{(1)}q, C_2 = {}_{(3)}\neg p \vee {}_{(4)}q, C_3 = {}_{(5)}p \vee {}_{(6)}\neg q, C_4 = {}_{(8)}\neg p \vee {}_{(7)}\neg q$$

~~$C_?$~~ = $\text{Res}_{p}^{lock}(C_2, C_3) = {}_{(4)}q \vee {}_{(6)}\neg q$ conform str.
eliminării este o clauză tautologică, deci se va elimina

~~$C_?$~~ = $\text{Res}_{q}^{lock}(C_1, C_4) = {}_{(2)}p \vee {}_{(8)}\neg p$ conform str.
eliminării este o clauză tautologică, deci se va elimina

Nu se mai rezolvă clauze noi, deci nu putem ajunge la \square , deci, am ajunge la concluzia greșită că S nu e inconsistentă.

Exerciții (1)

- rezoluția blocării fără strategia eliminării e completă

- $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$

$$C_1 =_{(2)} p \vee_{(1)} \textcolor{red}{q}, C_2 =_{(3)} \neg p \vee_{(4)} q, C_3 =_{(5)} \textcolor{red}{p} \vee_{(6)} \neg q, C_4 =_{(8)} \neg p \vee_{(7)} \textcolor{red}{q}$$

$$C_5 = \text{Res}_{p^{lock}}(C_2, C_3) =_{(4)} \textcolor{red}{q} \vee_{(6)} \neg q$$

$$C_6 = \text{Res}_{q^{lock}}(C_4, C_5) =_{(6)} \neg \textcolor{red}{q} \vee_{(8)} \neg p$$

$$C_8 = \text{Res}_{q^{lock}}(C_6, C_1) =_{(2)} \textcolor{red}{p} \vee_{(8)} \neg p$$

$$C_9 = \text{Res}_{p^{lock}}(C_8, C_2) =_{(4)} \textcolor{red}{q} \vee_{(8)} \neg p$$

$$C_{10} = \text{Res}_{q^{lock}}(C_9, C_4) =_{(8)} \neg p$$

$$C_{11} = \text{Res}_{p^{lock}}(C_{10}, C_3) =_{(6)} \neg q$$

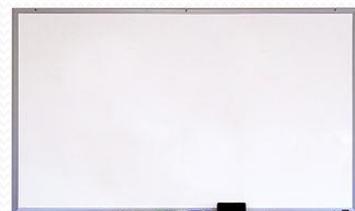
$\xrightarrow{\text{TCC}}$ S este inconsistentă

$$C_{12} = \text{Res}_{q^{lock}}(C_{11}, C_1) =_{(2)} \textcolor{red}{p}$$

$$C_{13} = \text{Res}_{p^{lock}}(C_{12}, C_{10}) = \square$$

Exerciții (1)

- rezoluția blocării + strategia eliminării nu e completă
 - $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$
 $C_1 = {}_{(2)}p \vee {}_{(1)}q, C_2 = {}_{(3)}\neg p \vee {}_{(4)}q, C_3 = {}_{(5)}p \vee {}_{(6)}\neg q, C_4 = {}_{(8)}\neg p \vee {}_{(7)}\neg q$
- rezoluția blocării + strategia mulțimii suport nu e completă
 - $p \rightarrow (q \rightarrow r), r \wedge s \rightarrow t, u \rightarrow s \wedge \neg t \vdash p \wedge q \rightarrow \neg u$
 $C_1 = {}_{(3)}\neg p \vee {}_{(2)}\neg q \vee {}_{(1)}r, C_2 = {}_{(6)}\neg r \vee {}_{(5)}\neg s \vee {}_{(4)}t, C_3 = {}_{(8)}\neg u \vee {}_{(7)}s,$
 $C_4 = {}_{(10)}\neg u \vee {}_{(9)}\neg t, C_5 = {}_{(11)}p, C_6 = {}_{(12)}q, C_7 = {}_{(13)}u$
 $Y = \{C_5, C_6, C_7\}$



Exerciții (1)

- rezoluția blocării + strategia mulțimii suport nu e completă

- $p \rightarrow (q \rightarrow r), r \wedge s \rightarrow t, u \rightarrow s \wedge \neg t \vdash p \wedge q \rightarrow \neg u$

$$C_1 =_{(3)} \neg p \vee_{(2)} \neg q \vee_{(1)} \textcolor{red}{r}, C_2 =_{(6)} \neg r \vee_{(5)} \neg s \vee_{(4)} \textcolor{red}{t}, C_3 =_{(8)} \neg u \vee_{(7)} \textcolor{red}{s},$$

$$C_4 =_{(10)} \neg u \vee_{(9)} \neg \textcolor{red}{t}, C_5 =_{(11)} \textcolor{red}{p}, C_6 =_{(12)} \textcolor{red}{q}, C_7 =_{(13)} \textcolor{red}{u}$$

$$Y = \{C_5, C_6, C_7\}$$

În această indexare, nu rezolvă nici o clauză urmând strategia mulțimii suport, deci am ajunge la falsa concluzie că S este consistentă.

Exerciții (1)

- rezoluția blocării fără strategia multimii suport:

$$p \rightarrow (q \rightarrow r), r \wedge s \rightarrow t, u \rightarrow s \wedge \neg t \vdash p \wedge q \rightarrow \neg u$$

$$\begin{aligned} C_1 &=_{(3)} \neg p \vee_{(2)} \neg q \vee_{(1)} \textcolor{red}{r}, \quad C_2 =_{(6)} \neg r \vee_{(5)} \neg s \vee_{(4)} \textcolor{red}{t}, \quad C_3 =_{(8)} \neg u \vee_{(7)} \textcolor{red}{s}, \\ C_4 &=_{(10)} \neg u \vee_{(9)} \neg \textcolor{red}{t}, \quad C_5 =_{(11)} \textcolor{red}{p}, \quad C_6 =_{(12)} \textcolor{red}{q}, \quad C_7 =_{(13)} \textcolor{red}{u} \end{aligned}$$

$$C_8 = \text{Res}_{t^{\text{lock}}}(C_2, C_4) =_{(5)} \neg \textcolor{red}{s} \vee_{(6)} \neg r \vee_{(10)} \neg u$$

$$C_9 = \text{Res}_{s^{\text{lock}}}(C_8, C_3) =_{(6)} \neg \textcolor{red}{r} \vee_{(8)} \neg u$$

$$C_{10} = \text{Res}_{r^{\text{lock}}}(C_1, C_9) =_{(2)} \neg \textcolor{red}{q} \vee_{(3)} \neg p \vee_{(8)} \neg u$$

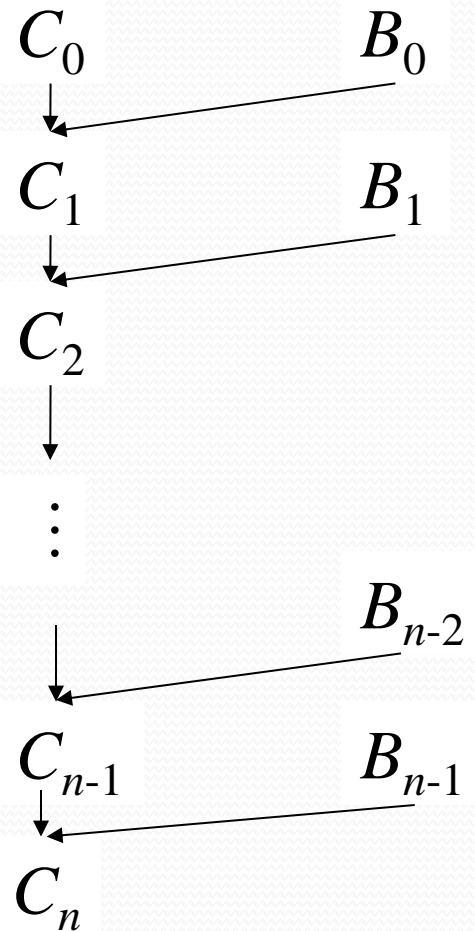
$$C_{11} = \text{Res}_{q^{\text{lock}}}(C_{10}, C_6) =_{(3)} \neg \textcolor{red}{p} \vee_{(8)} \neg u$$

$$C_{12} = \text{Res}_{p^{\text{lock}}}(C_{11}, C_5) =_{(8)} \neg \textcolor{red}{u}$$

$$C_{13} = \text{Res}_{u^{\text{lock}}}(C_{12}, C_7) =_{\square} \xrightarrow{\text{TCC}} S \text{ este inconsistentă}$$

Rezoluția liniară

- Loveland 1970
- procesul rezolutiv este liniar: la fiecare pas una dintre clauzele părinte este rezolventul obținut la pasul anterior
- Arborele de derivare corespunzător procesului rezolutiv liniar are forma:
 - C_0 clauză vârf
 - C_1, C_2, \dots, C_n clauze centrale
 - B_0, B_1, \dots, B_{n-1} clauze laterale
 - $\forall i=1,2,\dots,n$, are loc: $C_i = \text{Res} (C_{i-1}, B_{i-1})$



Teorema de corectitudine și completitudine

- Multimea S de clauze este inconsistentă, dacă și numai dacă $S \vdash_{\text{Res}}^{\text{lin}} \square$.

Exerciții (2)

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția liniară:

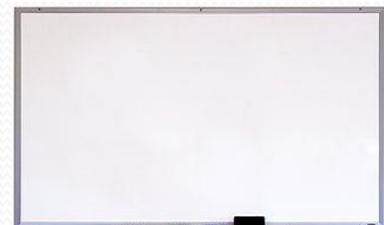
- $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$

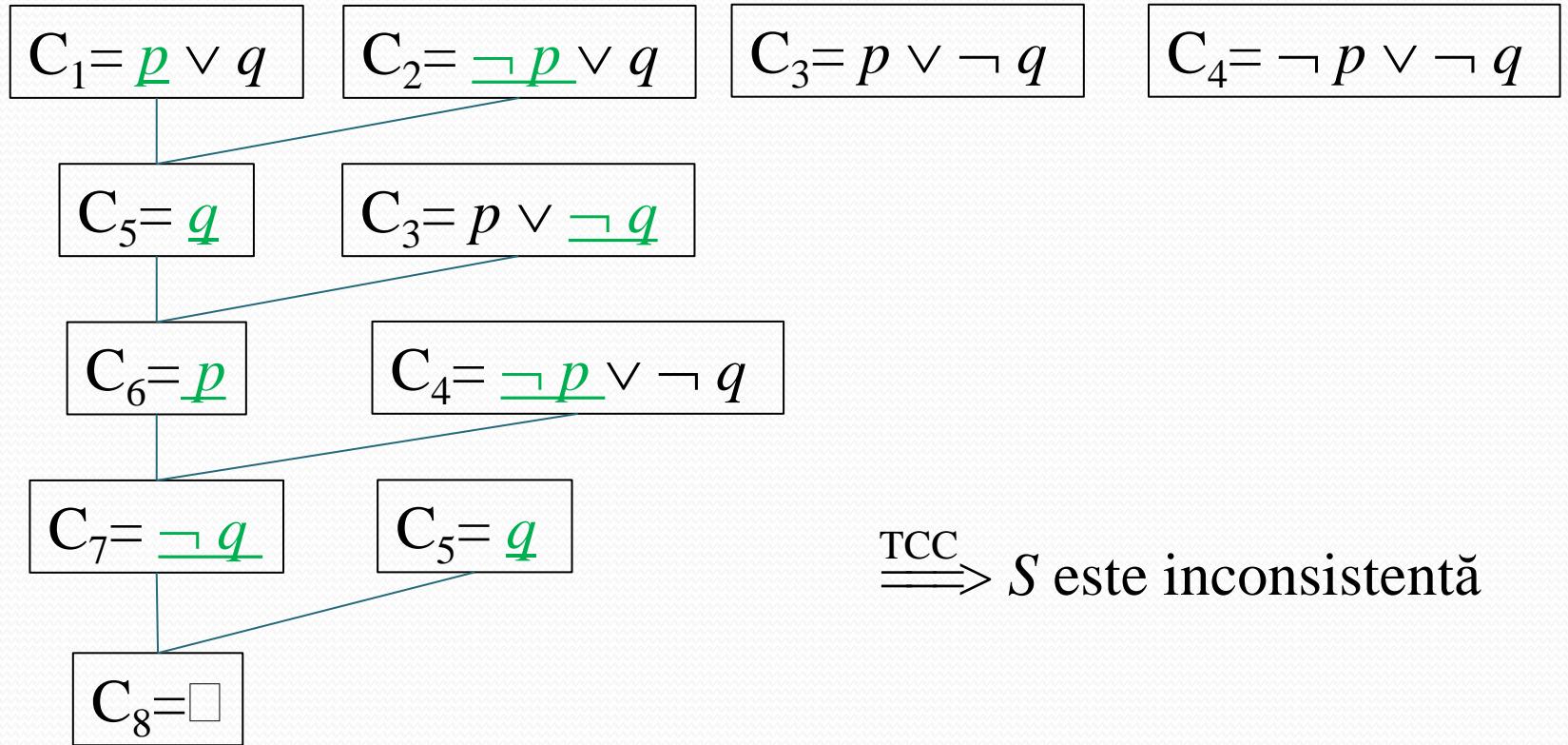
$$C_1 = p \vee q$$

$$C_2 = \neg p \vee q$$

$$C_3 = p \vee \neg q$$

$$C_4 = \neg p \vee \neg q$$





$\xrightarrow{\text{TCC}} S$ este inconsistentă

Exerciții (2)

Verificați inconsistența mulțimilor următoare de clauze utilizând rezoluția liniară:

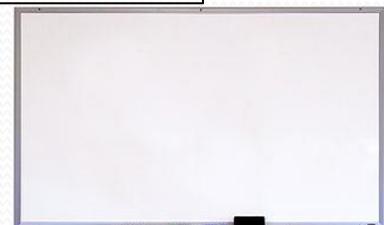
- $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$
- $S = \{\neg r, p \vee \neg q, r \vee p \vee q, \neg q \vee \neg p\}$ (+strategia eliminării)

$$C_1 = \neg r$$

$$C_2 = p \vee \neg q$$

$$C_3 = r \vee p \vee q$$

$$C_4 = \neg q \vee \neg p$$



$$C_1 = \neg r$$

$$C_2 = p \vee \neg q$$

$$C_3 = r \vee p \vee q$$

$$C_4 = \neg q \vee \neg p$$

3. Mai sus...

Începem cu alte 2 clauze care revolvă
Și reîncercăm (backtracking)...
Tot nu ajungem la clauza vidă

$\xrightarrow{\text{TCC}}$ S este consistentă

$$C_1 = \underline{\neg r}$$

$$C_3 = \underline{r} \vee p \vee q$$

$$C_5 = p \vee \underline{q}$$

$$C_2 = p \vee \underline{\neg q}$$

$$C_6 = \underline{p}$$

$$C_4 = \neg q \vee \underline{\neg p}$$

$$C_7 = \underline{\neg q}$$

$$C_3 = r \vee p \vee \underline{q}$$

$$C_4 = \neg q \vee \underline{\neg p}$$

$$C_8 = \underline{r} \vee p$$

$$C_1 = \underline{\neg r}$$

$$C_1 = \underline{\neg r}$$

$$C_9 = \underline{\neg q} \vee \underline{r}$$

$$C_6 = \underline{p}$$

$$C_3 = r \vee p \vee \underline{q}$$

$$C_7 = \underline{\neg q}$$

$$C_5 = p \vee \underline{q}$$

$$C_8 = r \vee p$$

$$C_6 = p$$

1. Nu îl mai luăm tot pe C_4 , nu avem altă opțiune, revenim un pas mai sus

2. nu avem altă opțiune, revenim mai sus

Observație:

- rezoluția liniară furnizează o strategie la nivel de implementare: *căutarea cu revenire*
 - la fiecare iterație, pentru clauza centrală pot exista mai multe posibile clauze laterale
 - după ce au fost utilizate toate posibilele clauze laterale, dar nu s-a obținut clauza vidă, se revine la iterarea precedentă
 - consistența mulțimii de clauze este demonstrată după o căutare completă fără derivarea clauzei vide

Cazuri particolare ale rezoluției liniare

- **Rezoluția unitară (unit)**: clauzele centrale au *cel puțin* o clauză *părinte unitară* (conține un singur literal)
- **Rezoluția de intrare (input)**: clauzele *laterale* sunt clauze *inițiale* (de intrare)

Teorema de echivalență dintre rezoluția unit și cea input

- Fie mulțimea S de clauze. $S \vdash_{\text{Res}}^{\text{input}} \square$ dacă și numai dacă $S \vdash_{\text{Res}}^{\text{unit}} \square$.
- **corectitudinea**: Dacă $S \vdash_{\text{Res}}^{\text{input/unit}} \square$ atunci S este inconsistentă
- **incompletitudinea**: există mulțimi inconsistente de clauze din care nu se poate deriva clauza vidă folosind rezoluția input sau rezoluția unit.

Exerciții (3)

Verificați inconsistența mulțimilor următoare de clauze utilizând rafinările input și unit:

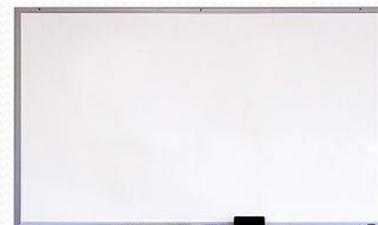
- $S = \{ \neg r, p \vee \neg q, r \vee q, \neg q \vee \neg p \}$

$$C_1 = \neg r$$

$$C_2 = p \vee \neg q$$

$$C_3 = r \vee q$$

$$C_4 = \neg q \vee \neg p$$



Input (cl. lat. sunt din mult. init.) și unit
(C_5, \dots, C_8 au cel puțin un părinte unit.)

$$C_1 = \neg r$$

$$C_2 = p \vee \neg q$$

$$C_3 = r \vee q$$

$$C_4 = \neg q \vee \neg p$$

$$C_1 = \neg r$$

$$C_3 = r \vee q$$

$$C_5 = q$$

$$C_2 = p \vee \neg q$$

$$C_6 = p$$

$$C_4 = \neg q \vee \neg p$$

$$C_7 = \neg q$$

$$C_3 = r \vee q$$

$$C_8 = r$$

$$C_1 = \neg r$$

$$C_9 = \square$$

$\xrightarrow{\text{TCC}}$ S este inconsistentă

Tipuri de metode

	Semantice	Sintactice
Directe	Tabela de adevăr FNC	Deductia (<i>mp</i>)
prin Respingere	FND Tabele semantică	Rezoluția (generală, strategia eliminării, strategia saturării pe nivele, strategia mulțimii suport, rafinarea rezoluției blocării, rafinarea rezoluției liniare, cazuri particulare: input și unit)

Logică computațională

Curs 8

Lector dr. Pop Andreea-Diana

Logica predicatelor de ordinul I

- ”Afară plouă” • $p \equiv P(a)$ (a – azi)
- ”În fiecare zi plouă afară” • $(\forall x) P(x)$ (x – ziua)
- ”Mâine va ploua” • $P(b)$ (b – mâine)
- ”Acum o lună a plouat” • $P(f(a))$ ($f(x) := x - 30$)



Sistemul axiomatic al logicii predicatelor de ordinul I - Alfabetul

- $P = (\Sigma_{Pr}, F_{Pr}, A_{Pr}, R_{Pr})$
- $\Sigma_{Pr} = Var \cup Const \cup (\cup_{j=1}^n \mathcal{F}_j) \cup (\cup_{j=1}^n \mathcal{P}_j) \cup \mathcal{P}_0 \cup Conective \cup Cuantif$
 - $Var = \{x, y, z, \dots\}$ – mulțimea *simbolurilor de variabile*
 - $Const = \{a, b, c, \dots\}$ – mulțimea *constantelor*
 - $\mathcal{F}_j = \{f \mid f: D^j \rightarrow D\}$ – mulțimea *simbolurilor de funcții* de aritate “ j ”
 - $\mathcal{P}_j = \{P \mid P: D^j \rightarrow \{T, F\}\}$ – mulțimea *simbolurilor de predicate* de aritate “ j ”
 - $\mathcal{P}_0 = \{p, q, r, \dots\} \cup \{T, F\}$ – mulțimea *variabilelor propoziționale* și a *valorilor de adevăr*
 - $Conective = \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$
 - $Cuantif = \{\forall \text{ (cuantificatorul universal)}, \exists \text{ (cuantificatorul existențial)}\}$

TERM, ATOM, Literal

- **$TERM$** = mulțimea *termenilor*:
 - $Var \subset TERM$
 - $Const \subset TERM$
 - dacă $f \in \mathcal{F}_k$ și $t_1, \dots, t_k \in TERM$ atunci $f(t_1, \dots, t_k) \in TERM$
- **$ATOM$** = mulțimea *formulelor atomice (atomilor)*:
 - $T, F \in ATOM$
 - dacă $P \in \mathcal{P}_k$ și $t_1, \dots, t_k \in TERM$ atunci $P(t_1, \dots, t_k) \in ATOM$
- ***Literal*** = un atom sau negația sa

Formule corect construite

- $F_{Pr} =$ mulțimea formulelor predicative bine formate
 - $ATOM \subset F_{Pr}$
 - dacă $U \in F_{Pr}$ și $x \in Var$ astfel încât x nu se află deja sub incidentă unui cuantificator (nu este legat), atunci:
 $(\forall x) U(x) \in F_{Pr}$ și $(\exists x) U(x) \in F_{Pr}$
 - dacă $U, V \in F_{Pr}$ astfel încât U și V nu conțin aceeași variabilă atât liberă cât și legată, atunci:
 $\neg U \in F_{Pr}, U \wedge V \in F_{Pr}, U \vee V \in F_{Pr}, U \rightarrow V \in F_{Pr}, U \leftrightarrow V \in F_{Pr}$

Axiome

- $A_{\text{Pr}} = \{A_1, A_2, A_3, A_4, A_5\}$ scheme axiomatice
 - $A_1: U \rightarrow (V \rightarrow U)$
 - $A_2: (U \rightarrow (V \rightarrow Z)) \rightarrow ((U \rightarrow V) \rightarrow (U \rightarrow Z))$
 - $A_3: (U \rightarrow V) \rightarrow (\neg V \rightarrow \neg U)$
 - $A_4: (\forall x) U(x) \rightarrow U(t)$, unde t este un termen arbitrar
 - $A_5: (U \rightarrow V(y)) \rightarrow (U \rightarrow (\forall x) V(x))$, unde y este o variabilă liberă
în V care nu apare în U , iar x nu este variabilă liberă nici în U , nici în V

Reguli de inferență

- $R_{\text{Pr}} = \{mp, gen\}$
- *modus ponens*: $U, U \rightarrow V \vdash_{mp} V$
- *regula generalizării*: $U(x) \vdash_{gen} (\forall x) U(x)$
 $(x \text{ era o variabilă liberă în } U)$

Definiții

- Variabilele din formulele predicative care se află sub incidența unui cuantificator se numesc *variabile legate*, în caz contrar ele se numesc *variabile libere*.
- O formulă predicativă se numește *închisă*, dacă toate variabilele sale sunt legate, iar în caz contrar se numește *deschisă*.

Exemplu – Transformarea unor afirmații din limbaj natural în logica predicatelor

- Dacă x și y sunt întregi nenegativi și x este mai mare decât y , atunci x^2 este mai mare decât y^2 .

$D = \mathbf{N}$

Variabilele (din D): x, y

Constantele (din D): 2

Simboluri de Funcții (definite pe $D^n \rightarrow D$):

$f: \mathbf{N}^2 \rightarrow \mathbf{N}, f(x,y) = x^y$

Simboluri de Predicate (definite pe $D^n \rightarrow \{\text{T,F}\}$):

$P: \mathbf{N}^2 \rightarrow \{\text{T,F}\}, P(x,y) = "x > y"$

Formula Predicativă: $(\forall x) (\forall y) (P(x, y) \rightarrow P(f(x,2), f(y,2)))$

Definiția deducției

- Fie formulele U_1, U_2, \dots, U_n numite ipoteze și V formulă propozițională. Spunem că V este deductibilă din U_1, U_2, \dots, U_n și notăm $U_1, U_2, \dots, U_{n-1}, U_n \vdash V$, dacă există o secvență de formule (f_1, f_2, \dots, f_m) astfel încât $f_m = V$ și $\forall i \in \{1, \dots, m\}$ avem:
 - $f_i \in A_{Pr}$;
 - $f_i \in \{U_1, U_2, \dots, U_n\}$;
 - $f_j, f_k \vdash_{mp} f_i$, $j < i$ și $k < i$
 - $f_j \vdash_{gen} f_i$, $j < i$
- Secvența (f_1, f_2, \dots, f_m) se numește *deducția* lui V din U_1, U_2, \dots, U_n .

Definiția teoremei

- O formulă $U \in F_{\text{Pr}}$, astfel încât $\emptyset \vdash U$ (sau $\vdash U$) se numește **teoremă**.

Exercițiu

$$A_1: U \rightarrow (V \rightarrow U)$$

$$A_2: (U \rightarrow (V \rightarrow Z)) \rightarrow ((U \rightarrow V) \rightarrow (U \rightarrow Z))$$

$$A_3: (U \rightarrow V) \rightarrow (\neg V \rightarrow \neg U)$$

$$A_4: (\forall x) U(x) \rightarrow U(t), \dots$$

$$A_5: (U \rightarrow V(y)) \rightarrow (U \rightarrow (\forall x) V(x)), \dots$$

$$\bullet (\forall y) P(y) \vdash (\forall x) (Q(x) \rightarrow P(x)) \quad U, U \rightarrow V \vdash_{mp} V$$

$$f_1: (\forall y) P(y) \text{ (ip)} \quad U(x) \vdash_{gen} (\forall x) U(x)$$

$$f_2: (\forall y) P(y) \rightarrow P(x) \text{ (A}_4\text{)}$$

$$f_1, f_2 \vdash_{mp} f_3: P(x)$$

$$f_4: P(x) \rightarrow (Q(x) \rightarrow P(x)) \text{ (A}_1\text{)}$$

$$f_3, f_4 \vdash_{mp} f_5: Q(x) \rightarrow P(x)$$

$$f_5 \vdash_{gen} f_6: (\forall x) (Q(x) \rightarrow P(x))$$

(f_1, f_2, \dots, f_6) este deducția lui $(\forall x) (Q(x) \rightarrow P(x))$ din $(\forall y) P(y)$, deci $(\forall y) P(y) \vdash (\forall x) (Q(x) \rightarrow P(x))$

Semantica logicii predicatorilor de ordinul I

- realizează legătura dintre
 - constantele,
 - simbolurile de funcții,
 - simbolurile de predicate respectiv constantele, funcțiile și predicatele din conceptualizarea universului modelat
- este furnizat un înțeles în termenii universului modelat pentru orice formulă din limbaj

Definiția interpretării

- O *interpretare* pentru un limbaj L al calculului predicatorilor este o pereche $I = \langle D, m \rangle$, unde :
 - D este o mulțime nevidă numită *domeniu al interpretării*.
 - m este o funcție care asociază:
 - o valoare fixă $m(c)$ din domeniul D unei constante c .
 - o funcție $m(f) : D^n \rightarrow D$ fiecărui simbol de funcție f de aritate n ;
 - un predicat $m(P) : D^n \rightarrow \{T, F\}$ fiecărui simbol de predicat P de aritate n .

Notări

pentru interpretarea $I = \langle D, m \rangle$:

- $|I| = D$ este domeniul interpretării I
- $I|x| = m(x)$ unde x este constantă, simbol de funcție sau simbol de predicat.
- $\text{As}(I)$ mulțimea funcțiilor de asignare de variabile peste domeniul interpretării I .
O funcție $a \in \text{As}(I)$ este definită astfel $a: \text{Var} \rightarrow |I|$.
 $[a]_x = \{a' \mid a' \in \text{As}(I) \text{ și } a'(y) = a(x), \text{ pentru orice } y \neq x\}$.

Definiția funcției de evaluare

Fie o interpretare I și $a \in \text{As}(I)$. Se definește inductiv **funcția de evaluare** v^I_a :

- $v^I_a(x) = a(x)$, $x \in \text{Var}$;
- $v^I_a(c) = I|c|$, $c \in \text{Const}$;
- $v^I_a(f(t_1, \dots, t_n)) = I|f|/(v^I_a(t_1), \dots, v^I_a(t_n))$, $f \in \mathcal{F}_k$, $n > 0$;
- $v^I_a(P(t_1, \dots, t_n)) = I|P|/(v^I_a(t_1), \dots, v^I_a(t_n))$, $P \in \mathcal{P}_k$, $n > 0$;
- $v^I_a(\neg A) = \neg v^I_a(A)$; $v^I_a(A \wedge B) = v^I_a(A) \wedge v^I_a(B)$
- $v^I_a(A \vee B) = v^I_a(A) \vee v^I_a(B)$; $v^I_a(A \rightarrow B) = v^I_a(A) \rightarrow v^I_a(B)$
- $v^I_a((\exists x)A(x)) = T$ dacă și numai dacă $v^I_{a'}(A(x)) = T$ pentru o funcție $a' \in [a]_x$
- $v^I_a((\forall x)A(x)) = T$ dacă și numai dacă $v^I_{a'}(A(x)) = T$ pentru orice funcție $a' \in [a]_x$

Concepțe semantice

- O formulă A este *realizabilă (consistentă)* dacă și numai dacă există o interpretare I și o funcție $a \in \text{As}(I)$ astfel încât $v^I_a(A)=T$. În caz contrar formula se numește *nerealizabilă (inconsistentă)*.
- Formula A este *adevărată* în interpretarea I dacă și numai dacă pentru orice funcție $a \in \text{As}(I)$ de asignare avem $v^I_a(A)=T$ și notăm $\models_I A$, iar I se numește *model* al lui A .
- Interpretarea I se numește *anti-model* al formulei predicative A dacă A este evaluată ca falsă în I , adică: $\forall a \in \text{As}(I)$ are loc $v^I_a(A)=F$.
- Formula A este *validă (tautologie)* dacă și numai dacă A este adevărată în orice interpretare și se notează: $\vdash A$.
- Două formule A și B sunt *logic echivalente* dacă $v^I_a(A)=v^I_a(B)$ pentru orice interpretare I și funcție a de asignare. Notație $A \equiv B$.
- O mulțime S de formule *implică logic* o formulă A dacă toate modelele mulțimii (adică modelele conjuncției formulelor din S) sunt modele ale formulei. Spunem că A este o *consecință logică* a mulțimii de formule S și notăm $S \models A$.
- O mulțime de formule predicative este *consistentă* dacă formula obținută prin conjuncția elementelor sale este consistentă, adică are cel puțin un model.
- O mulțime de formule este *inconsistentă* dacă nu există nici un model pentru formula obținută prin conjuncția elementelor sale.

Observații

- Evaluarea unei formule A închise depinde doar de interpretarea în care se evaluează formula, notându-se $v^I(A)$.
- Dacă interpretarea are domeniu finit:
 - o formulă cuantificată *universal* este înlocuită cu *conjuncția* instanțelor acesteia folosind toate elementele domeniului de interpretare
 - o formulă cuantificată *existențial* este înlocuită cu *disjuncția* instanțelor acesteia folosind toate elementele domeniului de interpretare

Exercițiu:

Evaluăți într-o interpretare cu domeniu finit și una cu domeniu infinit:

- $U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$



Exercițiu (domeniu infinit):

$$U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$$

$$I = \langle D, m \rangle$$

$$D = \mathbf{R}$$

Sunt constante (a,b,c)? Nu

Sunt simboluri de funcții? Nu

Sunt simboluri de Predicate? Da: P, Q – ambele au aritatea 1

$$m(P): \mathbf{R} \rightarrow \{\text{T}, \text{F}\}, m(P)(x) = "x < 0"$$

$$m(Q): \mathbf{R} \rightarrow \{\text{T}, \text{F}\}, m(Q)(x) = "x > 0"$$

$$\begin{aligned} v^I(U) &= v^I((\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)) = \\ &= v^I((\forall x)(P(x) \vee Q(x))) \rightarrow v^I((\forall x)P(x) \vee (\forall x)Q(x)) = \\ &= \color{red}{v^I((\forall x)(P(x) \vee Q(x)))} \rightarrow \color{green}{v^I((\forall x)P(x))} \vee \color{blue}{v^I((\forall x)Q(x))} = \\ &= \color{red}{= "(\forall x \in \mathbf{R}, x < 0 \text{ sau } x > 0)"} \rightarrow \color{green}{("(\forall x \in \mathbf{R}, x < 0)")} \vee \color{blue}{("(\forall x \in \mathbf{R}, x > 0)")} = \\ &= \color{red}{= F} \rightarrow \color{green}{F} \vee \color{blue}{F} = F \rightarrow F = T, \text{ deci } I \text{ este un model} \end{aligned}$$

Exercițiu (domeniu finit):

$$U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$$

$$I = \langle D, m \rangle$$

$$D = \{\text{România, Norvegia}\}$$

Sunt constante (a,b,c)? Nu

Sunt simboluri de funcții? Nu

Sunt simboluri de Predicate? Da: P, Q – ambele au aritatea 1

$m(P): \{\text{România, Norvegia}\} \rightarrow \{\text{T,F}\}$, $m(P)(x) = \text{"x se califică la Mondială"}$

$m(Q): \{\text{România, Norvegia}\} \rightarrow \{\text{T,F}\}$, $m(Q)(\text{România}) = \text{T}$,
 $m(Q)(\text{Norvegia}) = \text{F}$

$$\begin{aligned} v^I(U) &= v^I((\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)) = \\ &= v^I((\forall x)(P(x) \vee Q(x))) \rightarrow v^I((\forall x)P(x) \vee (\forall x)Q(x)) = \\ &= \color{red}{v^I((\forall x)(P(x) \vee Q(x)))} \rightarrow \color{green}{v^I((\forall x)P(x))} \vee \color{blue}{v^I((\forall x)Q(x))} = \\ &= \color{red}{(m(P)(\text{România}) \vee m(Q)(\text{România}))} \wedge \color{red}{(m(P)(\text{Norvegia}) \vee} \\ &\quad \color{red}{m(Q)(\text{Norvegia}))} \rightarrow \color{green}{m(P)(\text{România})} \wedge \color{green}{m(P)(\text{Norvegia})} \vee \color{blue}{m(Q)(\text{România})} \wedge \\ &\quad \color{blue}{m(Q)(\text{Norvegia})} = \color{red}{(\text{T} \vee \text{T})} \wedge \color{red}{(\text{T} \vee \text{F})} \rightarrow \color{green}{(\text{T} \wedge \text{T})} \vee \color{blue}{(\text{T} \wedge \text{F})} = \text{T} \wedge \text{T} \rightarrow \text{T} \vee \text{F} = \text{T} \rightarrow \text{T} = \text{T} \end{aligned}$$

Deci și I este model

Echivalențe logice în calculul predicatorilor

- Legile de expansiune

$$(\forall x) A(x) \equiv (\forall x) A(x) \wedge A(t), t - \text{termen oarecare } t \neq x$$

$$(\exists x) A(x) \equiv (\exists x) A(x) \vee A(t), t - \text{termen oarecare } t \neq x$$

- Legile infinite ale lui DeMorgan

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

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

- Legile de interschimbare a cuantificatorilor

$$(\exists x) (\exists y) A(x, y) \equiv (\exists y) (\exists x) A(x, y)$$

$$(\forall x) (\forall y) A(x, y) \equiv (\forall y) (\forall x) A(x, y)$$

Legi de extragere

- Legile de extragere a cuantificatorilor în fața formulei

$$A \vee (\exists x) B(x) \equiv (\exists x) (A \vee B(x))$$

$$A \vee (\forall x) B(x) \equiv (\forall x) (A \vee B(x))$$

$$A \wedge (\exists x) B(x) \equiv (\exists x) (A \wedge B(x))$$

$$A \wedge (\forall x) B(x) \equiv (\forall x) (A \wedge B(x))$$

unde formula A nu conține pe x ca variabilă liberă sau legată

$$(\exists x) A(x) \vee B \equiv (\exists x) (A(x) \vee B)$$

$$(\forall x) A(x) \vee B \equiv (\forall x) (A(x) \vee B)$$

$$(\exists x) A(x) \wedge B \equiv (\exists x) (A(x) \wedge B)$$

$$(\forall x) A(x) \wedge B \equiv (\forall x) (A(x) \wedge B)$$

unde formula B nu conține pe x ca variabilă liberă sau legată

Legile distributivității

- \exists față de \vee :

$$(\exists x) (A(x) \vee B(x)) \equiv (\exists x) A(x) \vee (\exists x) B(x)$$

- \forall față de \wedge :

$$(\forall x) (A(x) \wedge B(x)) \equiv (\forall x) A(x) \wedge (\forall x) B(x)$$

Semidistributivități

- \exists față de \wedge :

$$\models (\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)$$

- \forall față de \vee :

$$\models (\forall x) A(x) \vee (\forall x) B(x) \rightarrow (\forall x) (A(x) \vee B(x))$$

Forme normale ale formulelor predicative

- O formulă U predicativă este în *forma normală prenexă* dacă ea este de forma $(Q_1x_1)\dots(Q_nx_n) M$, unde $Q_i, i=1,\dots,n$ sunt cuantificatori logici, iar M nu conține cuantificatori. Secvența se numește *prefixul formulei* U , iar M este *matricea formulei* U .
- O formulă U predicativă este în *forma normală prenexă conjunctivă* dacă ea este în formă normală prenexă, iar matricea este în FNC.

• Teoremă:

Orice formulă din calculul predicatorilor poate fi transformată într-o formă normală prenexă logic echivalentă cu ea.

Algoritmul de aducere la forma normală prenexă

Pas 1: Se înlocuiesc conectivele \rightarrow și \leftrightarrow folosind \neg , \wedge , \vee .

Pas 2: Se aplică legile finite și infinite ale lui DeMorgan astfel încât cuantificatorii să nu fie precedați de negație.

Pas 3: Se redenumesc variabilele legate astfel încât ele să fie distințe.

Pas 4: Se utilizează echivalențele logice care reprezintă *legile de extragere a cuantificatorilor în fața formulei*.

!!! Ordinea de extragere a cuantificatorilor este arbitrară.

Forma normală Skolem (Pas 5)

- Fie U o formulă predicativă, iar $U^P = (Q_1x_1)\dots(Q_nx_n)M$ una dintre formele sale normale prenexe.
- Formulei U îi corespunde o formulă în *forma normală Skolem* notată U^S care se obține astfel: pentru fiecare cuantificator existențial Q_r din prefix se aplică următoarea transformare:
 - dacă înaintea simbolului Q_r nu apare niciun cuantificator universal, atunci se alege o constantă notată **a**, diferită de toate constantele care apar în M și se înlocuiesc toate aparițiile variabilei x_r în M cu **a**. Se șterge $(Q_r x_r)$ din prefixul formulei.
 - dacă înaintea simbolului Q_r apar cuantificatorii universali Q_{s_1}, \dots, Q_{s_m} , unde $1 \leq s_1 < \dots < s_m < r$, atunci alegem un simbol f de funcție de m variabile, diferit de celealte simboluri de funcții și se înlocuiește fiecare apariție a variabilei în M cu $f(x_{s_1}, \dots, x_{s_m})$. Se șterge $(Q_r x_r)$ din prefixul formulei.
- Constantele și funcțiile folosite pentru a înlocui variabilele existențiale se numesc *constante Skolem* și *funcții Skolem*.

Exemplu

$(\exists x) (\exists y) (\forall z) (\exists t) (\forall s) (\exists u) (\forall w) P(x, y, z, t, s, u, w)$

$x \leftarrow a$

$y \leftarrow b$

$t \leftarrow f(z)$

$u \leftarrow g(z, s)$

Forma Skolem: $(\forall z) (\forall s) (\forall w) P(a, b, z, f(z), s, g(z, s), w)$

Forma normală clauzală

- Formulei U îi corespunde o formulă în *forma normală Skolem fără cuantificatori* notată U^{Sq} care se obține prin eliminarea cuantificatorilor universali din U^S . (**Pas 6**)
- Formulei U îi corespunde o formulă în *forma normală clauzală* notată U^C care se obține din U^{Sq} prin aducerea la FNC. (**Pas 7**)
- Obs.: Transformările utilizate în procesul de Skolemizare nu păstrează echivalența logică, dar păstrează inconsistența

Teoremă

Fie U_1, U_2, \dots, U_n, V formule predicative.

- V inconsistentă **dacă** V^P inconsistentă **dacă** V^S inconsistentă **dacă** V^{Sq} inconsistentă **dacă** V^C inconsistentă.
- $\{U_1, U_2, \dots, U_n\}$ inconsistentă **dacă** $\{U_1^C, U_2^C, \dots, U_n^C\}$ inconsistentă.

Exercițiu: Aduceți la Forma normală Clauzală

- $U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$

Pas 1: Se înlocuiesc conectivele \rightarrow și \leftrightarrow folosind \neg , \wedge , \vee .

$$U \equiv \neg (\forall x)(P(x) \vee Q(x)) \vee (\forall x)P(x) \vee (\forall x)Q(x)$$

Pas 2: Se aplică legile finite și infinite ale lui DeMorgan astfel încât cuantificatorii să nu fie precedați de negație.

$$U \equiv (\exists x) (\neg P(x) \wedge \neg Q(x)) \vee (\forall x)P(x) \vee (\forall x)Q(x)$$

Pas 3: Se redenumesc variabilele legate astfel încât ele să fie distincte

$$U \equiv (\exists x) (\neg P(x) \wedge \neg Q(x)) \vee (\forall y)P(y) \vee (\forall z)Q(z)$$

Pas 4: Se utilizează echivalențele logice care reprezintă *legile de extragere a cuantificatorilor în fața formulei*. (Forma Normală Prenexă)

Exercițiu: Aduceți la Forma normală Clauzală

- $U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$

$$U \equiv (\exists x) (\neg P(x) \wedge \neg Q(x)) \vee (\forall y)P(y) \vee (\forall z)Q(z)$$

Pas 4: Se utilizează echivalențele logice care reprezintă *legile de extragere a cuantificatorilor în fața formulei*. (Forma Normală Prenexă)

$$U \equiv U^{P_1} = (\exists x) (\forall y) (\forall z) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

$$U \equiv U^{P_2} = (\exists x) (\forall z) (\forall y) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

$$U \equiv U^{P_3} = (\forall y) (\exists x) (\forall z) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

$$U \equiv U^{P_4} = (\forall z) (\exists x) (\forall y) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

$$U \equiv U^{P_5} = (\forall y) (\forall z) (\exists x) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

$$U \equiv U^{P_6} = (\forall z) (\forall y) (\exists x) ((\neg P(x) \wedge \neg Q(x)) \vee P(y) \vee Q(z))$$

Exercițiu: Aduceți la Forma normală Clauzală

- $U = (\forall x)(P(x) \vee Q(x)) \rightarrow (\forall x)P(x) \vee (\forall x)Q(x)$

$$U \equiv U^{P_3} \equiv (\forall y) (\exists x)(\forall z) ((\neg P(\textcolor{red}{x}) \wedge \neg Q(\textcolor{red}{x})) \vee P(y) \vee Q(z))$$

Pas 5: Eliminarea cuantificatorilor \exists (Forma normală Skolem)
(se păstrează doar inconsistență)

$$x \leftarrow f(y)$$

$$U \not\equiv U^{S_3} \equiv (\forall y) (\forall z) ((\neg P(\textcolor{red}{f}(y)) \wedge \neg Q(\textcolor{red}{f}(y))) \vee P(y) \vee Q(z))$$

Pas 6: Eliminarea cuantificatorilor \forall (Forma normală Skolem
fără cuantificatori)

$$U \not\equiv U^{Sq_3} \equiv (\neg P(f(y)) \wedge \neg Q(f(y))) \vee P(y) \vee Q(z)$$

Pas 7: Aducerea la Forma Normală Clauzală (distributivitatea \vee
față de \wedge)

$$U \not\equiv U^{C_3} \equiv (\neg P(f(y)) \vee P(y) \vee Q(z)) \wedge (\neg Q(f(y)) \vee P(y) \vee Q(z))$$

Sumar

Pas 1: Se înlocuiesc conectivele \rightarrow și \leftrightarrow folosind \neg , \wedge , \vee .

Pas 2: Se aplică legile finite și infinite ale lui DeMorgan astfel încât cuantificatorii să nu fie precedați de negație.

Pas 3: Se redenumesc variabilele legate astfel încât ele să fie distincte.

Pas 4: Se utilizează echivalențele logice care reprezintă *legile de extragere a cuantificatorilor în fața formulei*.

Pas 5: Eliminarea cuantificatorilor \exists (Forma normală Skolem)

Pas 6: Eliminarea cuantificatorilor \forall (Forma normală Skolem fără cuantificatori)

Pas 7: Aducerea la Forma Normală Clauzală (distributivitatea \vee față de \wedge)

Proprietățile logicii predicatelor de ordinul I

- **Teorema de completitudine și corectitudine**

Fie S o mulțime de formule predicative, iar A o formulă predicativă.

- **completitudinea:** dacă $S \models A$ atunci $S \vdash A$.
- **corectitudinea:** dacă $S \vdash A$ atunci $S \models A$.

- **Teorema respingerii**

Fie S o mulțime de formule predicative, iar A o formulă predicativă. Dacă $S \cup \{\neg A\}$ este inconsistentă atunci $S \vdash A$.

- **Teorema de deducție**

Fie S o mulțime de formule predicative, iar A o formulă predicativă. Dacă $S \cup \{A\} \vdash B$ atunci $S \vdash A \rightarrow B$.

Teorema lui Church 1936

- Problema validității unei formule în calculul predicatelor de ordinul I este *nedecidabilă*. Multimea formulelor valide din acest sistem logic este recursiv numărabilă, adică există o procedură P care, având ca intrare o formulă A din limbaj, are următorul comportament:
 - dacă formula A este validă, P se termină și furnizează răspunsul corespunzător;
 - dacă formula A nu este validă, P se termină cu răspunsul corespunzător sau execuția procedurii nu se încheie niciodată.
- Calculul predicatelor este **semi-decidabil**.

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)

- clasa α - formule de tip conjunctiv
- clasa β - formule de tip disjunctiv

$$A \wedge B$$

$$A \vee B$$

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

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

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

$$A \rightarrow B$$

Clase de formule (2)

- clasa γ - formule cuantificate universale
- clasa δ - formule cuantificate existențiale

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

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

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

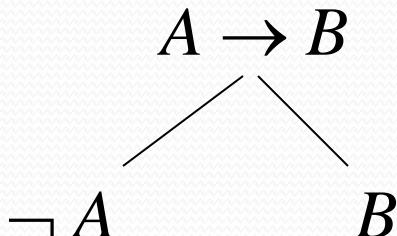
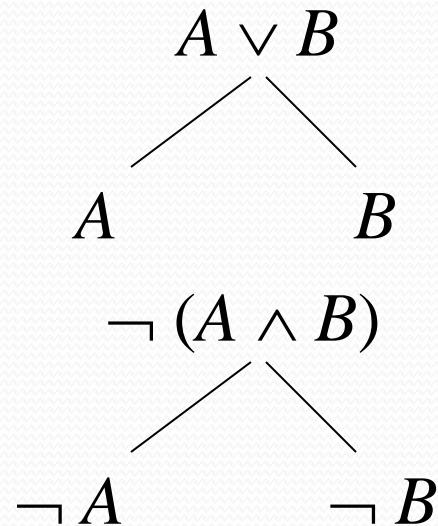
$$\neg (\forall x) A(x)$$

Reguli de descompunere a formulelor (1)

- regula α

$$\begin{array}{ccc} A \wedge B & \neg(A \vee B) & \neg(A \rightarrow B) \\ / & / & / \\ A & \neg A & A \\ / & / & / \\ B & \neg B & \neg B \end{array}$$

- 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

|

|

$(\forall x) A(x)$

$\neg A(c_n)$

|

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 î 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 tabelei 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 tabelei 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 tabele 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 *formei sale normale disjunctive*. Fiecare ramură reprezintă un *cub* (conjuncția tuturor literalilor de pe acea ramură), iar arborele este *disjuncția* tuturor *ramurilor* sale.
- Unei formule *consistentă* i se asociază o *tabelă completă deschisă*, iar fiecare *ramură deschisă* a tabelei furnizează cel puțin un *model* pentru formula respectivă.
- O *tabelă semantică închisă* asociată unei formule indică faptul că formula este *inconsistenta*, 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

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



$\neg((\exists x) (A(x) \wedge B(x)) \rightarrow (\exists x) A(x) \wedge (\exists x) B(x)) \ (1) \ \checkmark$ $\infty (1)$ $(\exists x) (A(x) \wedge B(x)) \ (2) \ \checkmark$ $\neg((\exists x) A(x) \wedge (\exists x) B(x)) \ (3) \checkmark$ $\delta (2), a - \text{constantă nouă}$ $A(a) \wedge B(a) \ (4) \ \checkmark$ $\infty (4)$ $A(a)$ $B(a)$ $\beta (3)$ $\neg(\exists x) A(x) \ (5) \checkmark$ $\neg A(a)$ $\gamma (5), a - \text{constantă existentă}$ $\neg(\exists x) A(x) \ (5') \ (\text{copia})$ $\neg(\exists x) B(x) \ (6) \checkmark$ $\neg B(a)$ $\gamma (6), a - \text{constantă existentă}$ $\neg(\exists x) B(x) \ (6') \ (\text{copia})$ \otimes TCC \otimes

Deci tabela semnatică este închisă \Rightarrow formula este tautologie

Exemple

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

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



$\neg((\forall x) (A(x) \vee B(x)) \rightarrow (\forall x) A(x) \vee (\forall x) B(x)) (1)\checkmark$ $\propto (1)$ $(\forall x) (A(x) \vee B(x)) (2)\checkmark$ \vdash $\neg((\forall x) A(x) \vee (\forall x) B(x)) (3)\checkmark$ $\propto (3)$ $\neg(\forall x) A(x) (4)\checkmark$ \vdash $\neg(\forall x) B(x) (5)\checkmark$ $\neg A(a)$ $\delta (4), a - \text{constantă nouă}$ $\neg B(b)$ $\delta (5), b - \text{constantă nouă}$ $A(a) \vee B(a)(6)\checkmark$ \vdash $A(b) \vee B(b)(7)\checkmark$ $(\forall x) (A(x) \vee B(x)) (2')\beta (6)$ $A(a)$ $B(a)$ $\beta (7)$ \otimes TCC \odot \otimes

Deci tabela semantică este deschisă \Rightarrow formula nu este tautologie

Exemple

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

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

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



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

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

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

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

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

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

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

$\gamma(1'')$, 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 *infinite*, *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 distințe, 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 – 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

$\theta_1 = [x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k]$ 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.

Exemple

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

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

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

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

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

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

$= [x \leftarrow a, z \leftarrow a, u \leftarrow f(g(u)), y \leftarrow a, t \leftarrow g(u), v \leftarrow w]$ – nu este 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 *instanță 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: $l_1 = P_1(t_{11}, t_{12}, \dots, t_{1n})$ și $l_2 = P_2(t_{21}, t_{22}, \dots, t_{2k})$ 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, $mgu(l_1, l_2) = \theta$ ”

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
- $P(\cancel{a}, x, g(f(y)))$ și $P(\cancel{f}(y), f(z), g(z))$ $[a \leftarrow \cancel{f}(y)]$
- $P(x, g(f(a)), x)$ și $P(f(y), z, h(y, f(y)))$



$A_1 = P(x, g(f(a)), x)$ 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
- ~~$P(\textcolor{red}{a}, x, g(f(y)))$ și $P(\textcolor{red}{f}(y), f(z), g(z))$~~ $[a \leftarrow \cancel{f(y)}]$
- ~~$P(x, g(f(a)), x)$ și $P(f(y), z, h(y, f(y)))$~~
- $P(a, h(x, u), f(g(y)))$ și $P(z, h(z, u), f(u))$



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

Logică computațională

Curs 10

Lector dr. Pop Andreea-Diana

Sistem formal (axiomatic) asociat Rezoluției predicative

- $\text{Res}^{\text{Pr}} = (\Sigma_{\text{Res}}^{\text{Pr}}, F_{\text{Res}}^{\text{Pr}}, A_{\text{Res}}^{\text{Pr}}, R_{\text{Res}}^{\text{Pr}})$
 - $\Sigma_{\text{Res}}^{\text{Pr}} = \Sigma_{\text{Pr}} \setminus \{ \forall, \exists, \wedge, \rightarrow, \leftrightarrow \}$ – **alfabetul**
 - $F_{\text{Res}}^{\text{Pr}} \cup \{ \square \}$ – **mulțimea formulelor bine-formate**
 - $F_{\text{Res}}^{\text{Pr}}$ mulțimea tuturor clauzelor ce se pot forma folosind alfabetul $\Sigma_{\text{Res}}^{\text{Pr}}$
 - \square - **clauza vidă** care nu conține nici un literal, simbolizează inconsistență
- $A_{\text{Res}}^{\text{Pr}} = \emptyset$ – **mulțimea axiomelor**
- $R_{\text{Res}}^{\text{Pr}} = \{ \text{res}^{\text{Pr}}, \text{fact} \}$ **mulțimea regulilor de inferență** care conține:

Reguli de inferență predicative

- **regula rezoluției predicative:**

$$A \vee l_1, B \vee \neg l_2 \vdash_{res}^{\text{Pr}} \theta(A) \vee \theta(B),$$

unde $\theta = mgu(l_1, l_2)$ și $A, B \in F_{\text{Res}}^{\text{Pr}}$

- $C_1 = A \vee l_1, C_2 = B \vee \neg l_2$ clauzele care rezolvă,

dacă literalii l_1 și l_2 sunt unificabili

- Rezolventul binar $C_3 = \text{Res}_{\theta}^{\text{Pr}}(C_1, C_2) = \theta(A) \vee \theta(B)$

- **regula factorizării:**

$$C \vdash_{fact} C', C' - \text{factor al lui } C$$

unde $C = l_1 \vee l_2 \vee \dots \vee l_k \vee l_{k+1} \vee \dots \vee l_n$,

$\lambda = mgu(l_1, l_2, \dots, l_k)$

$$C' = \lambda(l_k) \vee \lambda(l_{k+1}) \vee \dots \vee \lambda(l_n)$$

Exemple

$$C_1 = P(x) \vee \underline{Q(y, b)}$$

$$C_2 = \underline{\neg Q(z, z)} \vee R(g(z))$$

$$\text{Rez}_{\theta}^{\text{Pr}}(C_1, C_2) = \theta(P(x)) \vee \theta(R(g(z))) = P(x) \vee R(g(b))$$

$$\theta = [y \leftarrow z] \circ [z \leftarrow b] = [y \leftarrow b, z \leftarrow b]$$

$$C_3 = \underline{P(x)} \vee \underline{P(y)} \vee \underline{P(a)} \vee R(g(z), x)$$

$$\lambda = [x \leftarrow a, y \leftarrow a]$$

$$\text{Fact}_{\lambda}(C_3) = \lambda(\underline{P(a)}) \vee \lambda(R(g(z), a)) = \underline{P(a)} \vee R(g(z), a)$$

Teoremă

Fie U_1, U_2, \dots, U_n și V formule predicative.

- $\vdash V$ dacă și numai dacă $(\neg V)^C \vdash_{res}^{\Pr} \square$
- $U_1, U_2, \dots, U_n \vdash V$ dacă și numai dacă
$$\{U_1^C, U_2^C, \dots, U_n^C, (\neg V)^C\} \vdash_{res} \square$$

Observație: Variabilele din clauze distințe se recomandă să fie distințe.

Algoritmul rezoluției predicative:

Date de intrare: U_1, U_2, \dots, U_n , V – formule predicative

Date de ieșire: ”are loc $U_1, U_2, \dots, U_n \vdash V$ ” sau ”nu are loc $U_1, U_2, \dots, U_n \vdash V$ ”

Se construiește $S = \{ U_1^C, U_2^C, \dots, U_n^C, (\neg V)^C \}$

Repetă

Se selectează literalii l_1, l_2 și clauzele C_1, C_2 astfel încât sunt clauze sau factori ai unor clauze din S

Fie $l_1 \in C_1$ și $\neg l_2 \in C_2$, respectiv $l_1, l_2 \in C_1$

Dacă l_1 și l_2 sunt unificabili cu $\theta = mgu(l_1, l_2)$

Atunci

$C = \text{Res}_{\theta}^{\text{Pr}}(C_1, C_2)$ respectiv $C = \text{fact}_{\theta}(C_1)$

Algoritmul rezoluției predicative - continuare

Dacă $C = \square$

Atunci Scrie ”are loc $U_1, U_2, \dots, U_n \vdash V$ ”; **STOP**

Altfel $S = S \cup \{C\}$

Sfârșit_dacă

Sfârșit_dacă

Până când nu se mai pot deriva noi rezolvenți **sau** un număr fixat de iterații au fost executate

Dacă nu se mai pot deriva noi rezolvenți

Atunci Scrie ”nu are loc $U_1, U_2, \dots, U_n \vdash V$ ”

Altfel Scrie ”nu se poate decide dacă are loc sau nu $U_1, U_2, \dots, U_n \vdash V$ ”

Sfârșit_dacă

Sfârșit algoritm

Strategii și rafinări ale rezoluției predicative

- Strategii:
 - Strategia eliminării !unificarea, factorizarea
 - Strategia saturării pe nivele
 - Strategia mulțimii suport
- Rafinări:
 - Rezoluția blocării
 - Rezoluția liniară
 - input
 - unit

Exemple

$\vdash (\exists x)(\forall y) (P(x,y) \rightarrow R(x)) \rightarrow ((\forall x)(\forall y) P(x,y) \rightarrow (\exists z)R(z))$

$\vdash (\exists y) (\exists x) (P(x, y) \wedge P(y, y)) \vee (\exists y) (\exists x) (\neg P(y, x) \wedge \neg P(y, y))$

$\vdash (\exists x) (\forall y) (P(x) \wedge \neg P(y)) \vee \neg(\exists z)P(z)$



Exemplu (1)

obținerea multimii de clauze

$$\begin{aligned} & \vdash^? (\exists x)(\forall y) (P(x,y) \rightarrow R(x)) \rightarrow ((\forall x)(\forall y) P(x,y) \rightarrow (\exists z) R(z)) \xrightarrow{\text{ITD}} \\ & (\exists x)(\forall y) (P(x,y) \rightarrow R(x)) \vdash^? (\forall x)(\forall y) P(x,y) \rightarrow (\exists z) R(z) \xrightarrow{\text{ITD}} \\ & (\exists x)(\forall y) (P(x,y) \rightarrow R(x)), (\forall x)(\forall y) P(x,y) \vdash^? (\exists z) R(z) \end{aligned}$$

$$U_1 = (\exists x)(\forall y) (P(x,y) \rightarrow R(x)) \equiv (\exists x)(\forall y) (\neg P(x,y) \vee R(x)) = U_1^P$$

$$x \leftarrow a \quad U_1^S = (\forall y) (\neg P(a,y) \vee R(a))$$

$$U_1^{\text{Sq}} = \neg P(a,y) \vee R(a) = U_1^C = C_1$$

$$U_2 = (\forall x)(\forall y) P(x,y)$$

$$U_2^C = P(x,y) = U_2^S = C_2$$

$$\neg V = \neg (\exists z) R(z)$$

$$\neg V \equiv (\forall z) \neg R(z)$$

$$(\neg V)^{\text{Sq}} = \neg R(z) = (\neg V)^C = C_3$$

$$S = \{\neg P(a,y) \vee R(a), P(x,y), \neg R(z)\}$$

Exemplu (1)

aplicarea metodei rezoluției

$$S = \{\neg P(a,y) \vee R(a), P(x,y), \neg R(z)\}$$

$$\text{Rez}_{\theta}^{Pr}(C_1, C_2) = R(a) = C_4$$

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

$$\text{Rez}_{\lambda}^{Pr}(C_4, C_3) = \square$$

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

($\stackrel{\text{TCC}}{\Rightarrow}$ S este inconsistentă)

\Rightarrow (pe baza teoremei din curs) că $U_1, U_2 \vdash V$, și din rationamentul prin respingere (sau aplicând de 2 ori TD)

$$\vdash (\exists x)(\forall y) (P(x,y) \rightarrow R(x)) \rightarrow ((\forall x)(\forall y) P(x,y) \rightarrow (\exists z)R(z))$$

Exemplu (2)

obținerea multimii de clauze

$$\stackrel{?}{\vdash} (\exists y) (\exists x) (P(x, y) \wedge P(y, y)) \vee (\exists y) (\exists x) (\neg P(y, x) \wedge \neg P(y, y)) = U^{\text{not.}}$$

$$\neg U \equiv \neg ((\exists y) (\exists x) (P(x, y) \wedge P(y, y)) \vee (\exists y) (\exists x) (\neg P(y, x) \wedge \neg P(y, y)))$$

$$\neg U \equiv (\forall y) (\forall x) (\neg P(x, y) \vee \neg P(y, y)) \wedge (\forall y) (\forall x) (P(y, x) \vee P(y, y))$$

$$\neg U \equiv (\forall y) (\forall x) (\neg P(x, y) \vee \neg P(y, y)) \wedge (\forall t) (\forall z) (P(t, z) \vee P(t, t))$$

$$\neg U \equiv (\forall y) (\forall x) (\forall t) (\forall z) ((\neg P(x, y) \vee \neg P(y, y)) \wedge (P(t, z) \vee P(t, t)))$$

$$\neg U^{Sq} = (\neg P(x, y) \vee \neg P(y, y)) \wedge (P(t, z) \vee P(t, t)) = \neg U^C$$

$$C_1 = \neg P(x, y) \vee \neg P(y, y)$$

$$C_2 = P(t, z) \vee P(t, t)$$

$$S = \{ C_1, C_2 \}$$

Exemplu (2)

aplicarea metodei rezoluției

$$S = \{ C_1, C_2 \}$$

$$\text{Rez}_{\theta}^{\text{Pr}}(C_1, C_2) = \neg P(x, t) \vee P(t, z) = C_3$$

$$\theta = [y \leftarrow t]$$

$$\text{Fact}_{\lambda}^{\text{Pr}}(C_2) = P(t, t) = C_4$$

$$\lambda = [z \leftarrow t]$$

$$\text{Fact}_{\xi}^{\text{Pr}}(C_1) = \neg P(y, y) = C_5$$

$$\xi = [x \leftarrow y]$$

$$\text{Rez}_{\theta}^{\text{Pr}}(C_4, C_5) = \square, \text{ deci, pe baza teoremei din curs} \Rightarrow \vdash U$$

Exemplu (3)

obținerea multimii de clauze

$$\stackrel{?}{\vdash} (\exists x) (\forall y) (P(x) \wedge \neg P(y)) \vee \neg(\exists z) P(z) \stackrel{\text{not.}}{=} U$$

$$\neg U \equiv \neg ((\exists x) (\forall y) (P(x) \wedge \neg P(y)) \vee \neg(\exists z) P(z))$$

$$\neg U \equiv (\forall x) (\exists y) (\neg P(x) \vee P(y)) \wedge (\exists z) P(z)$$

$$\neg U \equiv (\exists z) (\forall x) (\exists y) ((\neg P(x) \vee P(y)) \wedge P(z)) = (\neg U)^P$$

$$z \leftarrow a, y \leftarrow f(x)$$

$$(\neg U)^S = (\forall x) ((\neg P(x) \vee P(f(x))) \wedge P(a))$$

$$(\neg U)^{Sq} = (\neg P(x) \vee P(f(x))) \wedge P(a) = (\neg U)^C$$

$$C_1 = \neg P(x) \vee P(f(x))$$

$$C_2 = P(a)$$

$$S = \{ C_1, C_2 \}$$

Exemplu (3)

aplicarea metodei rezoluției

$$S = \{ C_1, C_2 \}$$

$$\text{Rez}_{\theta}^{\text{Pr}}(C_1, C_2) = P(f(a)) = C_3$$

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

$$\text{Rez}_{\lambda}^{\text{Pr}}(C_1, C_3) = P(f(f(a))) = C_4$$

$$\lambda = [x \leftarrow f(a)]$$

Ciclu infinit, deci nu putem decide dacă formula este sau nu teoremă.

Compleitudinea și corectitudinea

- Toate rafinările și strategiile rezolutive păstrează completitudinea și corectitudinea.
- Combinarea lor poate impune prea multe restricții și deși multimea inițială de clauze este inconsistentă, s-ar putea să nu se poată deriva clauza vidă.
- sunt complete:
 - rezoluția generală + strategia eliminării
 - rezoluția generală + strategia mulțimii suport
 - rezoluția generală + strategia mulțimii suport + strategia eliminării
 - rezoluția liniară + strategia eliminării
 - rezoluția liniară + strategia mulțimii suport
- nu sunt complete:
 - rezoluția blocării + strategia eliminării
 - rezoluția blocării + strategia mulțimii suport
 - rezoluția blocării + rezoluția liniară
 - rezoluția unitară
 - rezoluția de intrare

Completitudinea rezoluției de intrare

Definiții:

- O clauză se numește *pozitivă* dacă aceasta conține literali pozitivi.
- O clauză se numește *negativă* dacă aceasta conține doar literali negativi.
- O clauză se numește *clauză Horn* dacă aceasta conține un singur literal pozitiv, ceilalți fiind negativi.

Teoremă:

- Rezoluția de intrare este completă pe o mulțime de cluze Horn, cu o clauză negativă ca și clauză de vârf (PROLOG).

- $H_i: l_1 \wedge l_2 \wedge \dots \wedge l_n \rightarrow v$, $i \in \{1, \dots, k\}$
- $C: n_1 \wedge n_2 \wedge \dots \wedge n_m ?$

Limbajul Prolog - "o privire pe furiș"

- tată(B,C):-bărbat(B),părinte(B,C).
 - mamă(M,C):-femeie(M),părinte(M,C).
 - fiu(P,S):-bărbat(S),părinte(P,S).
 - Fiică(P,D):-femeie(D),părinte(P,D).
 - frate(A,F):-părinte(P,A),părinte(P,F).
-
- părinte("Petru","Vlad")
 - părinte("Petru","Bogdan")
-
- ? frate("Vlad","Bogdan")?

Problemă de modelare

- Toate păsările colibri sunt bogat colorate.
 - Păsările mari nu se hrănesc cu miere.
 - Păsările care nu se hrănesc cu miere nu sunt bogat colorate.
 - Picky este o pasăre colibri.
-
- Toate păsările colibri sunt păsări mici.



Transformarea din limbaj natural în limbaj logic

”pasărea x este bogat colorată” $\stackrel{\text{not.}}{=} B(x)$

”pasărea x se hrănește cu miere” $\stackrel{\text{not.}}{=} H(x)$

”pasărea x este o parsăre colibri” $\stackrel{\text{not.}}{=} C(x)$

”pasărea x este mică” $\stackrel{\text{not.}}{=} M(x)$

Constantă: Picky $\stackrel{\text{not.}}{=} a$

$$U_1 = (\forall x) (C(x) \rightarrow B(x))$$

$$U_2 = (\forall x) (\neg M(x) \rightarrow \neg H(x))$$

$$U_3 = (\forall x) (\neg H(x) \rightarrow \neg B(x))$$

$$U_4 = C(a)$$

$$V = (\forall x) (C(x) \rightarrow M(x))$$



Logică computațională

Curs 11

Lector dr. Pop Andreea-Diana

Algebrelor booleene

- introduse de George Boole (1815-1864)
- stau la baza definirii funcțiilor booleene
 - care sunt utilizate în realizarea circuitelor logice

Definirea axiomatică (1)

O **algebră booleană** este o structură $(A, \wedge, \vee, \bar{}, 0, 1)$, unde:

- $|A| \geq 2$, A conținând cel puțin 2 elemente diferite, 0 și 1, $0 \neq 1$
- \wedge, \vee sunt operații binare
- $\bar{}$ este operator unar
- există elementul unic 0 – elementul zero, cu proprietățile:
 $x \wedge 0 = 0 \wedge x = 0$ și $x \vee 0 = 0 \vee x = x$, $\forall x \in A$
- există elementul unic 1 – elementul unitate, cu proprietățile:
 $x \wedge 1 = 1 \wedge x = x$ și $x \vee 1 = 1 \vee x = 1$, $\forall x \in A$
- elementul zero, 0 și cel unitate, 1 sunt primul respectiv ultimul element, iar \bar{x} este complementul lui x :
 $x \wedge \bar{x} = 0$ și $x \vee \bar{x} = 1$, $\forall x \in A$
- dubla negație:
 $\bar{\bar{x}} = x$, $\forall x \in A$

Definirea axiomatică (2)

- operațiile \wedge , \vee sunt comutative:

$$x \wedge y = y \wedge x \text{ și } x \vee y = y \vee x, \forall x, y \in A$$

- operațiile \wedge , \vee sunt asociative:

$$x \wedge (y \wedge z) = (x \wedge y) \wedge z (= x \wedge y \wedge z) \text{ și}$$

$$x \vee (y \vee z) = (x \vee y) \vee z (= x \vee y \vee z), \forall x, y, z \in A$$

- au loc proprietățile de distributivitate ale operatorilor \wedge și \vee :

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \text{ și}$$

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z), \forall x, y, z \in A$$

- are loc proprietatea de idempotență pentru ambele operații:

$$x \wedge x = x \text{ și } x \vee x = x, \forall x \in A$$

- au loc legile lui De Morgan:

$$\overline{x \wedge y} = \overline{x} \vee \overline{y} \text{ și } \overline{x \vee y} = \overline{x} \wedge \overline{y}, \forall x, y \in A$$

- au loc proprietățile de absorbție:

$$x \wedge (x \vee y) = x \text{ și } x \vee (x \wedge y) = x, \forall x, y \in A$$

Observații

- Într-o algebră booleană are loc principiul dualității:
„Pentru orice egalitate între două expresii booleene $U = V$, există o nouă egalitate $U' = V'$, obținută prin interschimbarea operațiilor \wedge , \vee și a elementelor: 0, 1”.
- majoritatea axiomelor algebrei booleene, sunt perechi de axiome duale

Algebra booleană binară

$$B = (B_2 = \{0,1\}, \wedge, \vee, \neg, 0, 1)$$

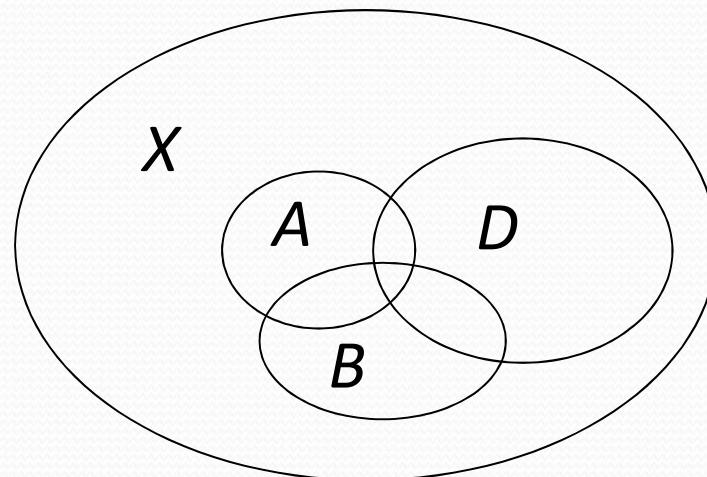
\vee	0	1
0	0	1
1	1	1

\wedge	0	1
0	0	0
1	0	1

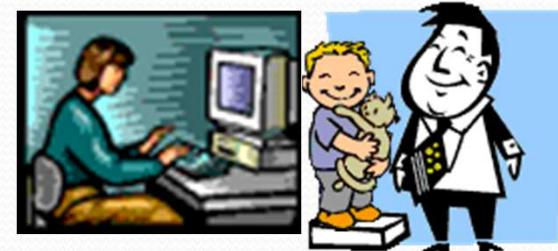
x	\bar{x}
0	1
1	0

Alte exemple de algebre booleene

- $(F_P, \wedge, \vee, \neg, F, T)$
- $(\mathcal{P}(X), \cap, \cup, C, \emptyset, X)$



Problemă practică



- Cine face tortul?

Soluțiile posibile

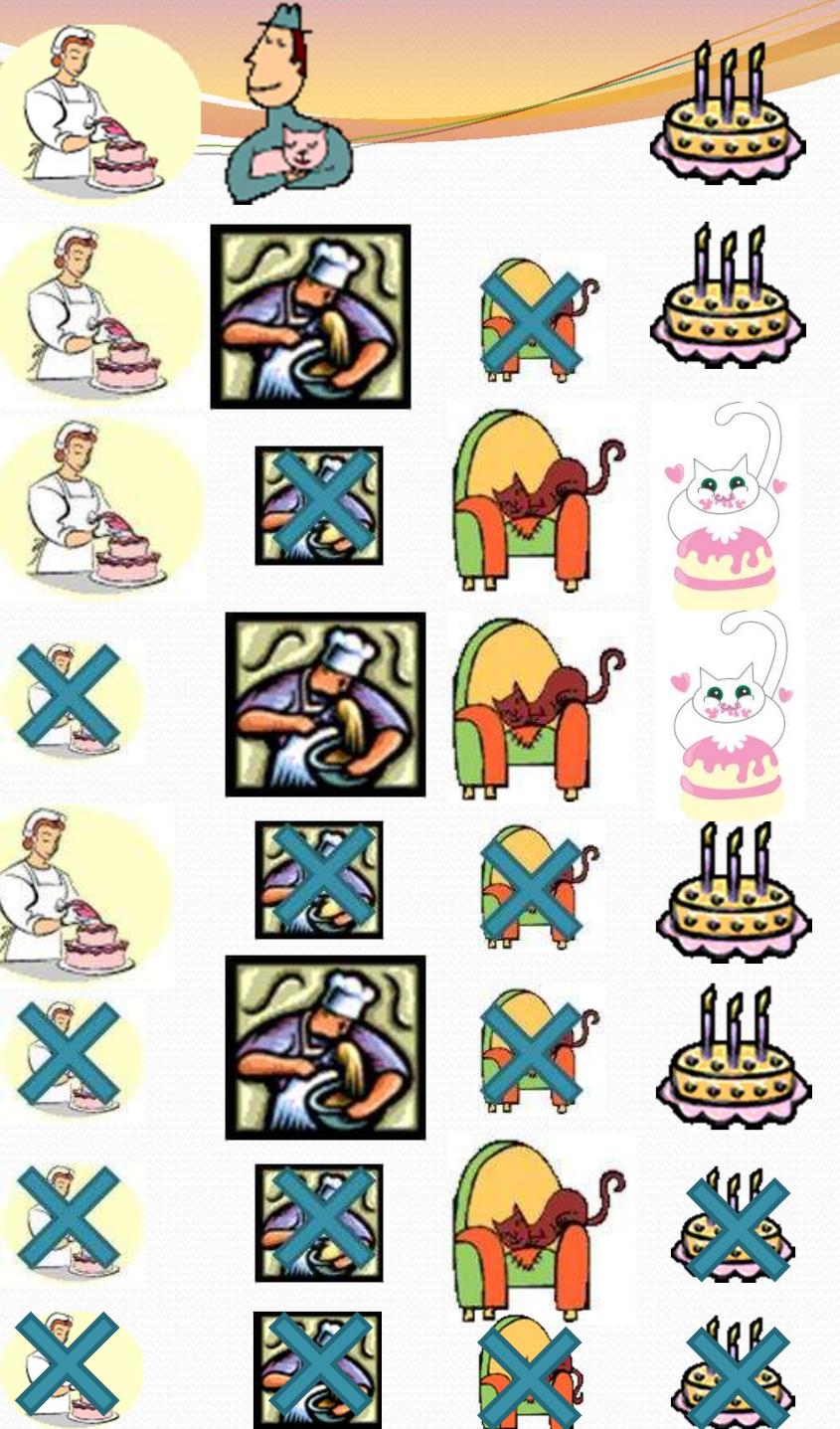


$x,$

$y,$

z

$$f(x, y, z) =$$



Funcții booleene

Fie $B = (B_2, \wedge, \vee, \neg, 0, 1)$ algebra booleană binară, $B_2 = \{0, 1\}$ și $n \in N^*$. O *funcție booleană de n variabile* este o funcție definită recursiv astfel:

1. Funcția *proiecție*: $P_i : B_2^n \rightarrow B_2$, $P_i(x_1, \dots, x_i, \dots, x_n) = x_i$, care păstrează doar variabila x_i , este o funcție booleană.
 2. Dacă avem două funcții booleene $f, g : B_2^n \rightarrow B_2$, atunci $f \wedge g, f \vee g, \bar{f}$ sunt funcții booleene, unde:
$$(f \wedge g)(x_1, \dots, x_n) = f(x_1, \dots, x_n) \wedge g(x_1, \dots, x_n)$$
$$(f \vee g)(x_1, \dots, x_n) = f(x_1, \dots, x_n) \vee g(x_1, \dots, x_n)$$
$$\bar{f}(x_1, \dots, x_n) = \overline{f(x_1, \dots, x_n)}$$
- Orice funcție booleană este obținută prin aplicarea de un număr finit de ori a regulilor 1 și 2 de mai sus.

Toate funcțiile booleene definite pe
 $B_2^n \rightarrow B_2$
pentru $n=1$

x	$f_0(x)$	$f_1(x)$	$f_2(x)$	$f_3(x)$
0	0	0	1	1
1	0	1	0	1

Teoreme

- $\forall n \in N^*$, există 2^{2^n} funcții booleene de n variabile.
- Mulțimea tuturor funcțiilor booleene de $n \in N^*$ variabile formează o algebră booleană: $(FB(n), \wedge, \vee, \neg, f_0, f_{2^{2^n}-1})$, unde $f_0(x_1, x_2, \dots, x_n) = 0$ și $f_{2^{2^n}-1}(x_1, \dots, x_n) = 1$ funcțiile constante 0, respectiv 1.

Notații

- $x^\alpha = \begin{cases} x, & \text{dacă } \alpha = 1 \\ \bar{x}, & \text{dacă } \alpha = 0 \end{cases}, x \in \{0, 1\}$
- Pentru $x, \alpha \in \{0, 1\}$, au loc: $x^0 = \bar{x}$, $x^1 = x$ și
 - $0^0 = \bar{0} = 1$; $0^1 = 0$;
 - $1^0 = \bar{1} = 0$; $1^1 = 1$
- Astfel, se obține: $x^\alpha = \begin{cases} 1, & \text{dacă } x = \alpha \\ 0, & \text{dacă } x \neq \alpha \end{cases}, x, \alpha \in \{0, 1\}$

Formele canonice ale funcțiilor booleene (1)

O funcție booleană $f: (B_2)^n \rightarrow B_2$, $n \in N^*$ poate fi transformată în cele două forme echivalente:

- *forma canonică disjunctivă* (FCD):

$$(1) f(x_1, \dots, x_n) = \bigvee_{(\alpha_1, \dots, \alpha_n) \in (B_2)^n} (f(\alpha_1, \dots, \alpha_n) \wedge x_1^{\alpha_1} \wedge \dots \wedge x_n^{\alpha_n})$$

- *forma canonică conjunctivă* (FCC):

$$(2) f(x_1, \dots, x_n) = \bigwedge_{(\alpha_1, \dots, \alpha_n) \in (B_2)^n} (f(\alpha_1, \dots, \alpha_n) \vee x_1^{\overline{\alpha_1}} \vee \dots \vee x_n^{\overline{\alpha_n}})$$

Formele canonice ale funcțiilor booleene (2)

O funcție booleană $f: (B_2)^n \rightarrow B_2$, $n \in N^*$ este unic determinată de valorile sale $f(\alpha_1, \alpha_2, \dots, \alpha_n)$, unde $(\alpha_1, \alpha_2, \dots, \alpha_n) \in (B_2)^n$:

- forma canonică disjunctivă:

$$(1) \Leftrightarrow (1') f(x_1, \dots, x_n) = \bigvee_{(\alpha_1, \dots, \alpha_n) \in (B_2)^n \text{ și } f(\alpha_1, \dots, \alpha_n) = 1} (x_1^{\alpha_1} \wedge \dots \wedge x_n^{\alpha_n})$$

- forma canonică conjunctivă:

$$(2) \Leftrightarrow (2') f(x_1, \dots, x_n) = \bigwedge_{(\alpha_1, \dots, \alpha_n) \in (B_2)^n \text{ și } f(\alpha_1, \dots, \alpha_n) = 0} (x_1^{\overline{\alpha_1}} \vee \dots \vee x_n^{\overline{\alpha_n}})$$

Observație

- Forma canonica *disjunctivă* este recomandată în caz contrar, când există un *număr mare de zerouri* ale funcției și un număr mic de valori 1.
- Forma canonica *conjunctivă* este utilă când există un număr mic de zerouri și un *număr mare de valori 1* (realizări).
- Funcția booleană f_0 nu poate fi scrisă în forma canonica disjunctivă, deoarece nu ia valoarea 1 pentru niciun argument.
- Funcția booleană $f_{2^{2^n}-1}$ nu poate fi scrisă în forma canonica conjunctivă, deoarece nu ia valoarea 0 pentru niciun argument.

Definiție

Fie $f: (B_2)^n \rightarrow B_2$, $n \in N^*$ o funcție booleană cu n variabile.

- O conjuncție de variabile se numește **monom**.
- Un monom care conține toate cele n variabile se numește **monom canonic** sau **minterm** de n variabile. Are forma:

$$x_1^{\alpha_1} \wedge \dots \wedge x_n^{\alpha_n}, \alpha_i \in B_2$$

- Disjuncția care conține toate cele n variabile, având forma:

$$x_1^{\alpha_1} \vee \dots \vee x_n^{\alpha_n}, \alpha_i \in B_2$$
 se numește **maxterm** de n variabile.

Proprietăți

- $\forall n \in \mathbf{N}^*$, există exact 2^n maxtermi, notați cu $M_0, M_1, \dots, M_{2^n - 1}$ și exact 2^n mintermi, notați cu $m_0, m_1, \dots, m_{2^n - 1}$.
- Un *maxterm* este o funcție booleană care ia valoarea 0 doar pentru un argument.
- Un *minterm* este o funcție booleană care ia valoarea 1 doar pentru un argument.

Observație

- Indicele unui minterm de n variabile este obținut prin conversia în zecimal a numărului binar format cu cifrele ce reprezintă puterile celor n variabile ale expresiei acestuia.
- Indicele unui maxterm de n variabile este obținut prin conversia în zecimal a numărului binar, format cu dualele cifrelor ce reprezintă puterile celor n variabile ale expresiei acestuia.

Propoziție

- Conjuncția a doi mintermi distincți este 0:

$$m_i \wedge m_j = 0, \forall i \neq j, \quad i, j = 0, \dots, 2^{n-1}$$

- Disjuncția a doi maxtermi distincți este 1:

$$M_i \vee M_j = 1, \forall i \neq j, \quad i, j = 0, \dots, 2^{n-1}$$

- Un minterm și un maxterm cu același indice sunt funcții duale.

$$M_i = \overline{m_i}, \overline{M}_i = m_i, \forall i = 0, \dots, 2^{n-1}$$

Observație

- Forma canonică conjunctivă, FCC , este conjuncția maxtermilor corespunzători argumentelor pentru care funcția ia valoarea 0.
- Forma canonică disjunctivă, FCD , este disjuncția mintermilor corespunzători argumentelor pentru care funcția ia valoarea 1.

Soluțiile posibile



$x,$



$y,$

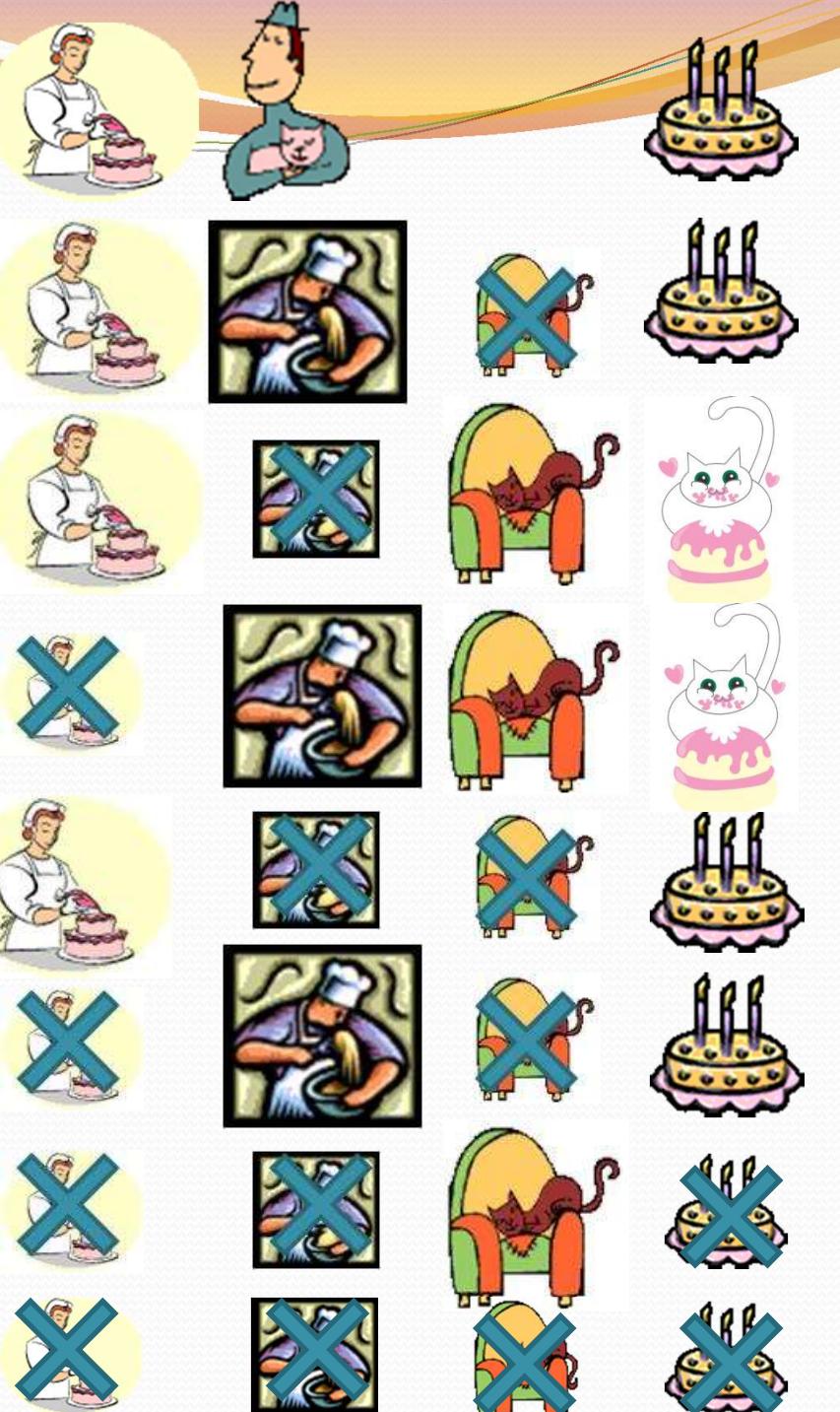


z

$$f(x, y, z) =$$



- Putem să le „condensăm”?



Suportul funcției

- suportul funcției booleene f



$$S_f = \{ (x_1, x_2, \dots, x_n) \in B_2^n \mid f(x_1, x_2, \dots, x_n) = 1 \}$$



$$(1, 1, 0) \in S_f$$



$$(1, 0, 1) \notin S_f$$

Relația mai mic sau egal

- Relația *mai mic sau egal* $f \leq g \Leftrightarrow S_f \subseteq S_g$

$$m = x \wedge y \wedge \overline{z}$$

și

$$m' = x \wedge \overline{z}$$



$$S_m = \{(1, 1, 0)\} \quad \subset \quad S_{m'} = \{(1, 1, 0), (1, 0, 0)\}$$

$$m \leq m'$$

Factorizarea

- *monoame adiacente (vecine)* dacă ele diferă doar prin puterea (semnul: negație sau nu) variabilei cu indicele „ k_i ”:

$$m = x_{k_1}^{\alpha_{k_1}} \wedge \dots \wedge x_{k_j}^{\alpha_{k_j}} \wedge x_{k_i} \wedge x_{k_l}^{\alpha_{k_l}} \wedge \dots \wedge x_{k_s}^{\alpha_{k_s}}$$

$$m' = x_{k_1}^{\alpha_{k_1}} \wedge \dots \wedge x_{k_j}^{\alpha_{k_j}} \wedge \overline{x_{k_i}} \wedge x_{k_l}^{\alpha_{k_l}} \wedge \dots \wedge x_{k_s}^{\alpha_{k_s}}$$

- *factorizarea monoamelor* m și m' este operația prin care se obține, eliminând variabila cu indicele „ k_i ”, monomul

$$m \vee m' = x_{k_1}^{\alpha_{k_1}} \wedge \dots \wedge x_{k_j}^{\alpha_{k_j}} \wedge x_{k_l}^{\alpha_{k_l}} \wedge \dots \wedge x_{k_s}^{\alpha_{k_s}}$$

mai mare decât m și m' .

Simplificarea

- A *simplifica* o funcție booleană, înseamnă a obține o formă echivalentă a sa cu un număr cât mai mic de apariții de variabile.
- Dacă se pornește de la FCD(f), prin factorizări repetate, într-un număr finit de pași se poate obține o formă simplificată a funcției.

Mulțimea monoamelor maximale

- Mulțimea $M(f)$ se numește *mulțimea monoamelor maximale* ale funcției booleene $f : B_2^n \rightarrow B_2$ dacă:

$$\forall m \in M(f), m \in FB(n), m \leq f;$$

$$\forall m \in M(f), \exists m' \in FB(n), \text{ astfel încât } m < m' \leq f$$

Mulțimea monoamelor centrale

- Mulțimea $C(f)$ se numește *mulțimea monoamelor centrale* ale funcției booleane $f : B_2^n \rightarrow B_2$ dacă:
$$\forall m \in C(f), m \in M(f)$$
$$\forall m \in C(f), \text{nu are loc } m \leq \vee m', \text{ unde } m' \in M(f) - \{m\}$$

Algoritmul de simplificare a funcțiilor booleene

Date de intrare: f – o funcție booleană în formă canonica disjunctivă

Date de ieșire: $f'_1, f'_2, \dots, f'_{k'}$ - variantele simplificate ale funcției f

Se determină $M(f)$ și $C(f)$.

dacă $M(f) = C(f)$

atunci $f'_1 = \bigvee_{m \in M(f)} m$ STOP1 // caz 1--- soluție unică
altfel

dacă $C(f) \neq \emptyset$ *not.*

atunci $g = \bigvee_{m \in C(f)} m$
 $f'_i = g \vee h_i, i = \overline{1, k}$, unde h_i este disjuncția unui număr cât mai mic de monoame maximale astfel încât $S_{h_i} = S_f - S_g$
STOP2 // caz 2 --- k soluții

altfel // nu există monoame centrale

$f'_i = h_i, i = \overline{1, k}$, unde h_i este disjuncția unui număr cât mai mic de monoame maximale astfel încât $S_{h_i} = S_f$
STOP3 // caz 3 --- k soluții

sf_dacă

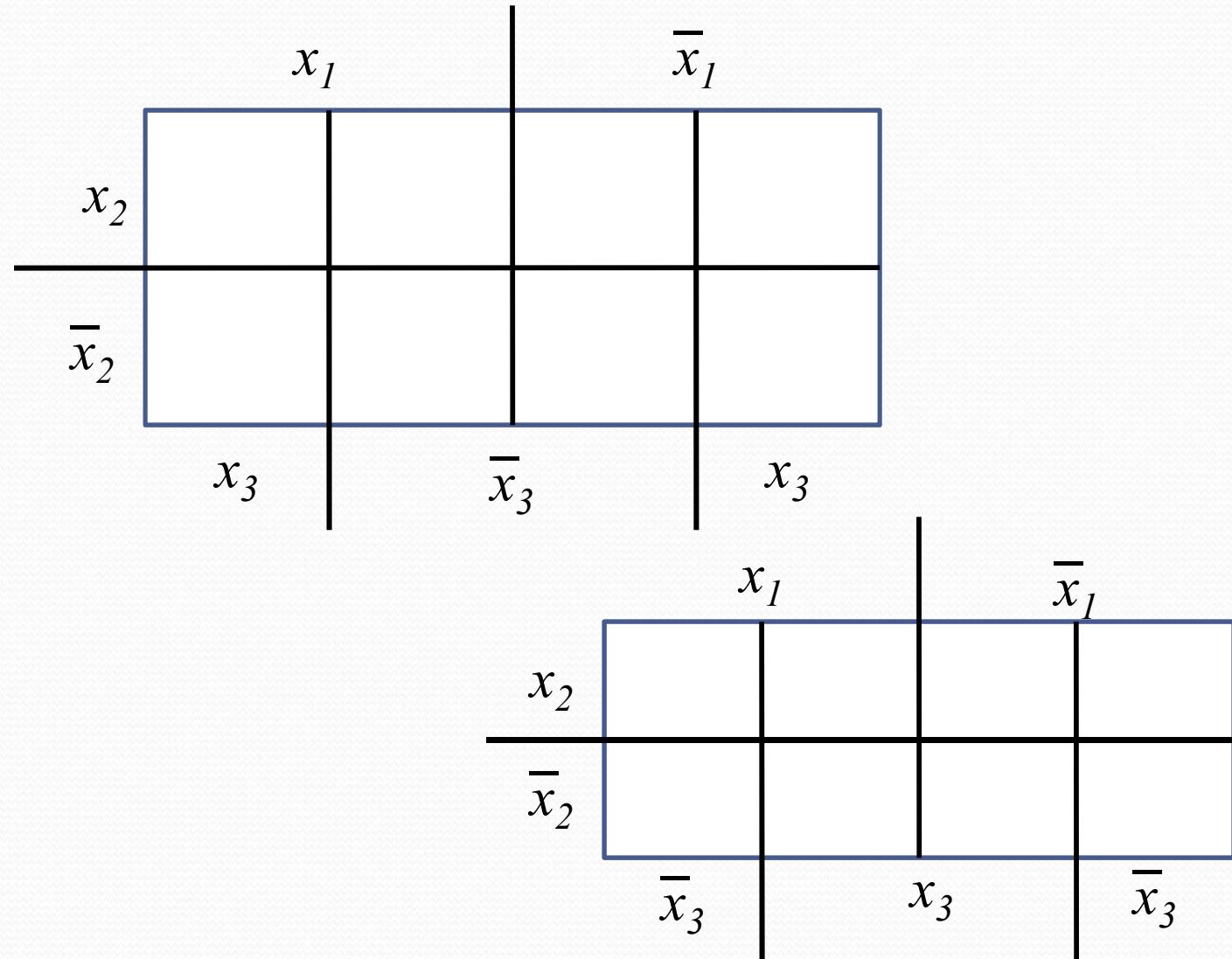
sf_dacă

Sf_algoritm

Metoda diagramelor Veitch

- Această metodă se bazează pe reprezentarea grafică, sub forma unei diagrame, a suportului funcției.
- Este foarte utilă pentru funcții cu un număr mic de variabile: 2, 3 sau 4.

Construirea diagramei Veitch



Completarea diagramei

	x_1		\bar{x}_1	
x_2		m_{13}		x_4
\bar{x}_2				\bar{x}_4
		m_0		
x_3			\bar{x}_3	x_4

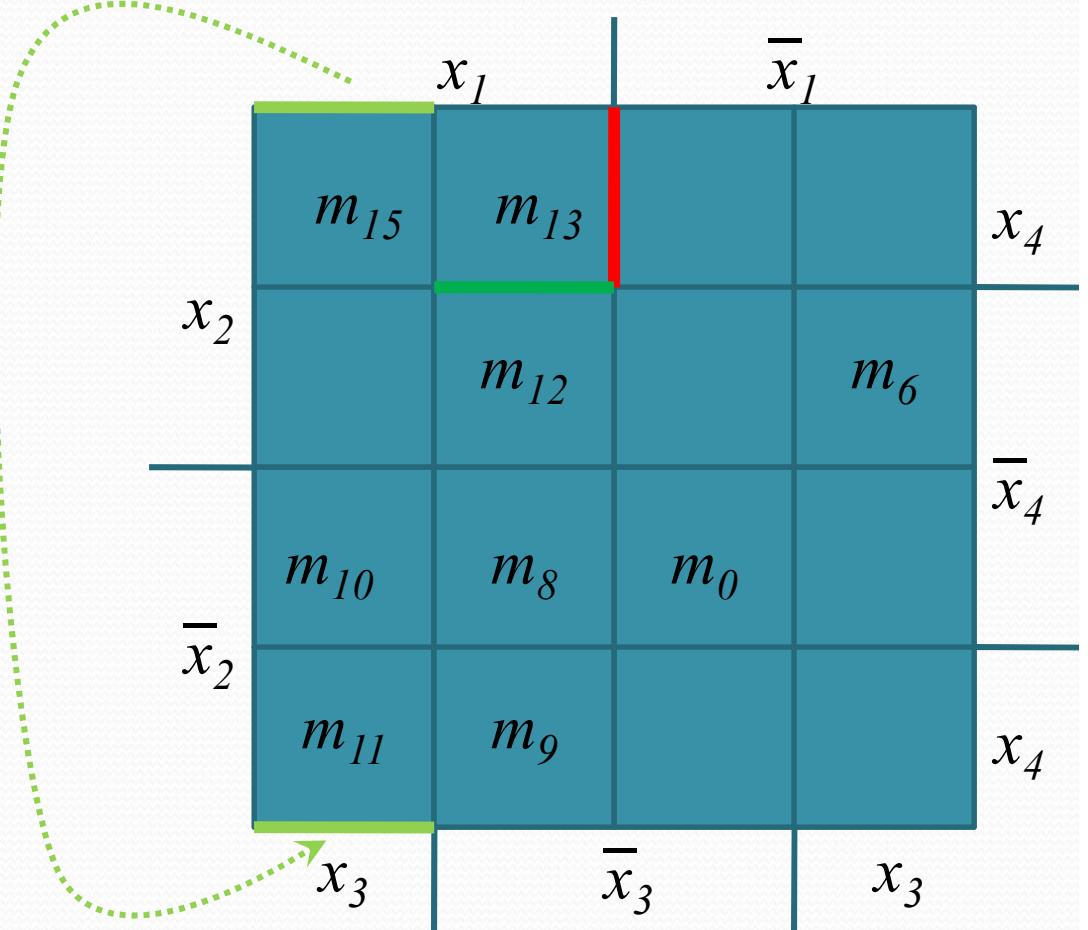
$m_{13} = x_1 x_2 \bar{x}_3 x_4$

Exercițiu

$$f(x_1, x_2, x_3, x_4) = x_1 x_2 \bar{x}_3 x_4 \vee \bar{x}_1 \bar{x}_2 \bar{x}_3 \bar{x}_4 \vee x_1 x_2 \bar{x}_3 \bar{x}_4 \vee x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4$$
$$\vee x_1 \bar{x}_2 \bar{x}_3 x_4$$
$$\vee x_1 x_2 x_3 x_4$$
$$\vee \bar{x}_1 x_2 x_3 \bar{x}_4$$
$$\vee x_1 \bar{x}_2 x_3 \bar{x}_4$$
$$\vee x_1 \bar{x}_2 \bar{x}_3 x_4$$

		x_1		\bar{x}_1	
x_2	m_{15}		m_{13}		
			m_{12}		m_6
\bar{x}_2	m_{10}		m_8	m_0	
	m_{11}		m_9		
x_3			\bar{x}_3		x_3
\bar{x}_3					
x_4					x_4
\bar{x}_4					

Mintermi adiacenți



Factorizarea (1)

Se grupează 2^k , $k \in \mathbf{N}$ mintermi adiacenți, **k – cât mai mare**

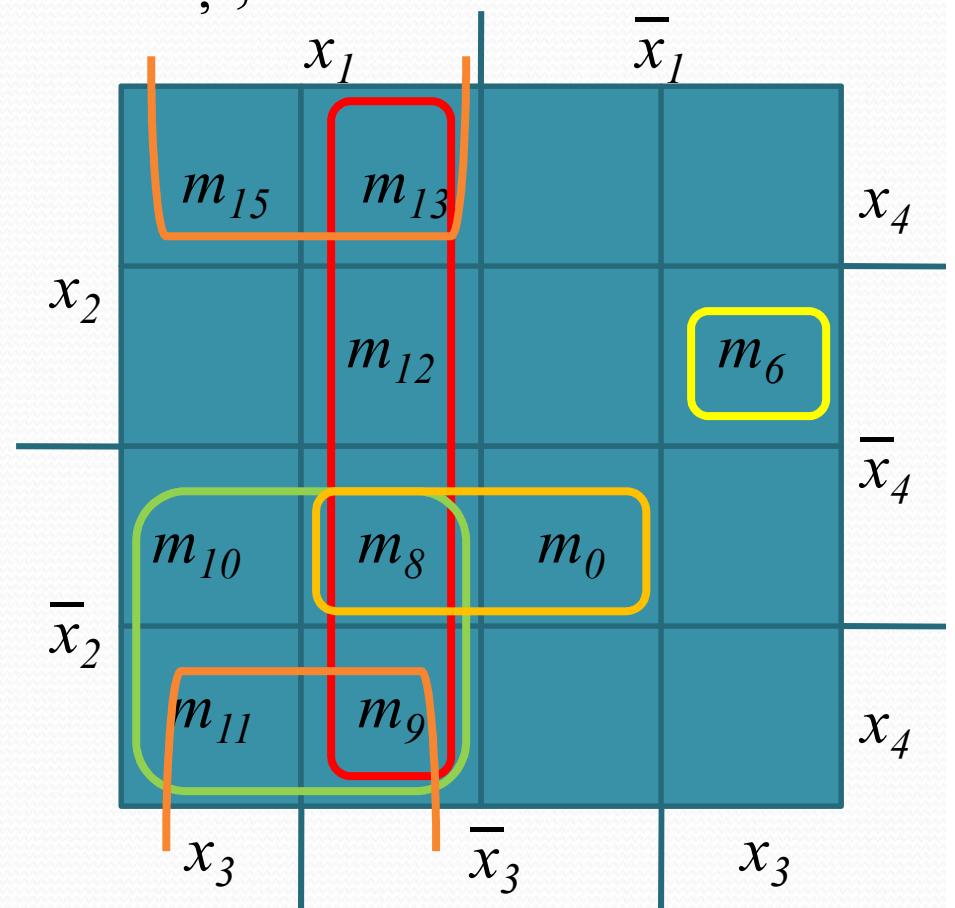
$$max_1 = m_{13} \vee m_{12} \vee m_8 \vee m_9$$

$$max_2 = m_{10} \vee m_{11} \vee m_8 \vee m_9$$

$$max_3 = m_{11} \vee m_9 \vee m_{15} \vee m_{13}$$

$$max_4 = m_8 \vee m_0$$

$$max_5 = m_6$$



Factorizarea (2)

- se grupează 2^k , $k \in \mathbf{N}$ mintermi adiacenți
- ! prima și ultima linie, prima și ultima coloană sunt vecine!
- același minterm poate fi utilizat de mai multe ori
- ! se negligează grupurile incluse!
- nu se lasă mintermi neîncercuiți

Formulele monoamelor maximale

$$max_1 = m_{13} \vee m_{12} \vee m_8 \vee m_9 = x_1 \bar{x}_3$$

$$max_2 = m_{10} \vee m_{11} \vee m_8 \vee m_9$$

$$= x_1 \bar{x}_2$$

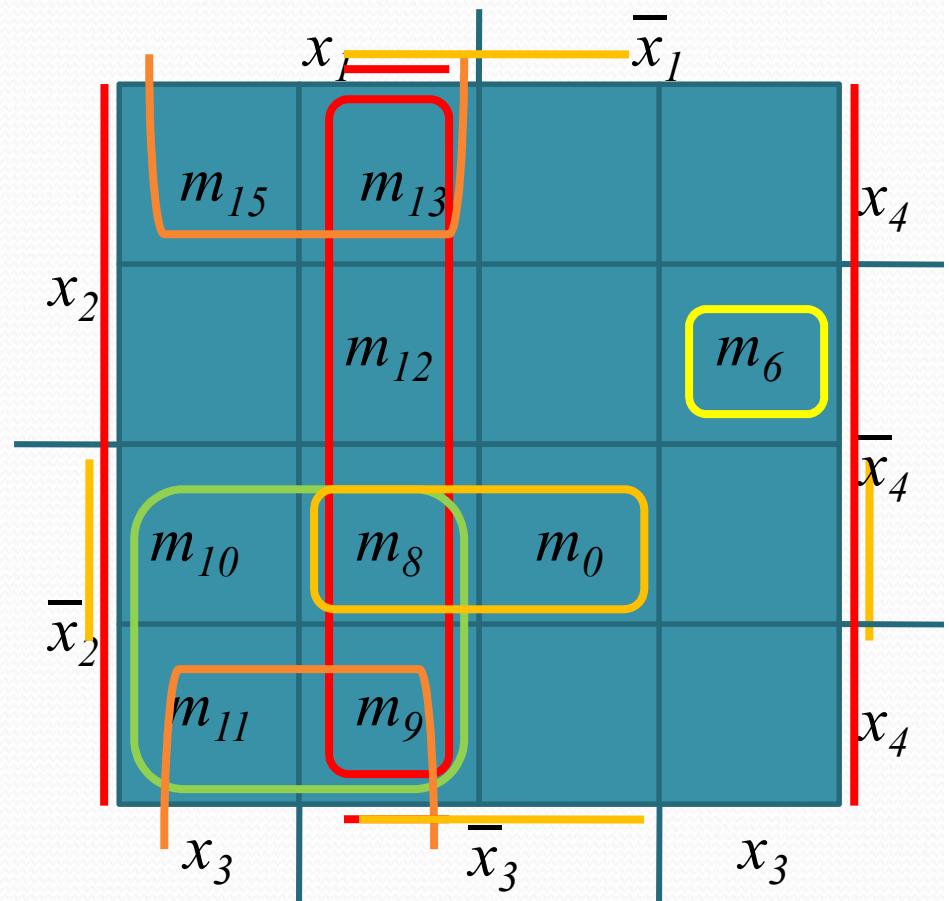
$$max_3 = m_{11} \vee m_9 \vee m_{15} \vee m_{13}$$

$$= x_1 x_4$$

$$max_4 = m_8 \vee m_0$$

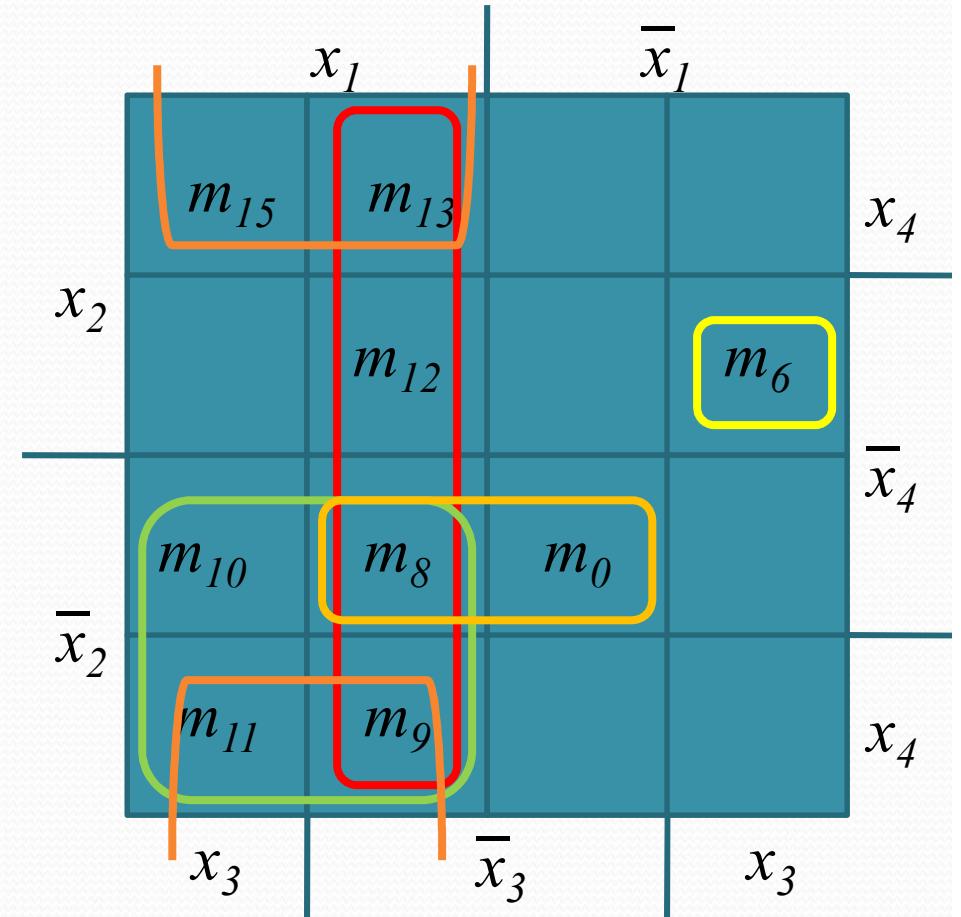
$$= \bar{x}_2 \bar{x}_3 \bar{x}_4$$

$$max_5 = m_6 = \bar{x}_1 x_2 x_3 \bar{x}_4$$



Mulțimea monoamelor maximale și mulțimea monoamelor centrale

$$M(f) = \{max_1, max_2, max_3, max_4, max_5\}$$
$$M(f) = C(f)$$

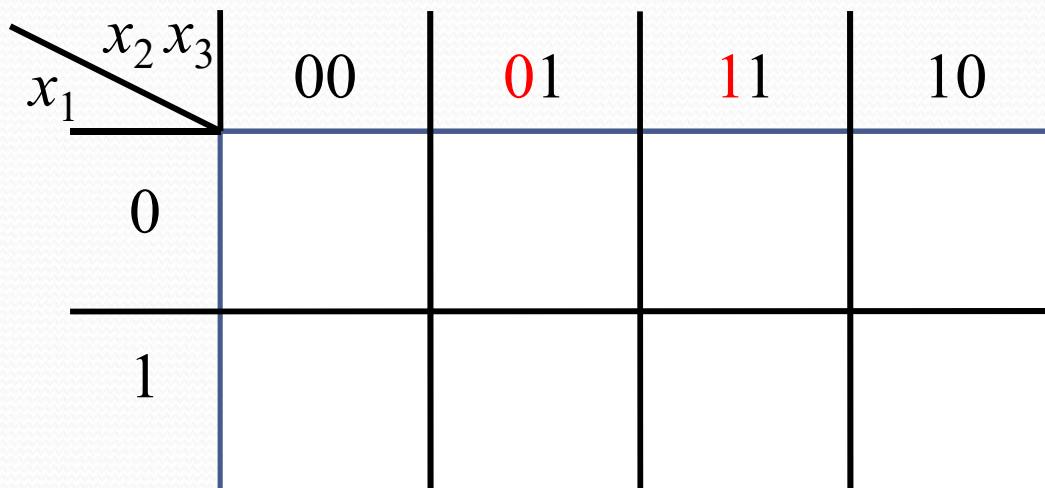


Forma simplificată a funcției

- Cazul I al algoritmului de simplificare
- O singură formă simplificată a funcției:

$$f^s(x_1, x_2, x_3, x_4) = x_1\bar{x}_3 \vee x_1\bar{x}_2 \vee x_1x_4 \vee \bar{x}_2\bar{x}_3\bar{x}_4 \vee \bar{x}_1x_2x_3\bar{x}_4$$

Construirea diagramei Karnaugh



- ! O singură cifră **distinctă**!



Logică computațională

Curs 12

Lector dr. Pop Andreea-Diana

Metoda analitică a lui Quine-Mc'Clusky

- se bazează pe completarea a două tabele ajutătoare
 - unul pentru factorizare, utilizat la calcularea mulțimii monoamelor maximale
 - unul pentru identificarea mulțimii monoamelor centrale
- se aplică formei canonice disjunctive a funcției
- poate fi utilizată pentru oricâte variabile

Aplicarea metodei

1. ordonarea mulțimii suport a funcției cu n variabile, crescător sau descrescător după numărul de valori 1 conținut de fiecare n -uplu

$$f(x_1, x_2, x_3, x_4) = m_4 \vee m_6 \vee m_7 \vee m_8 \vee m_9 \vee m_{10} \vee m_{11} \vee m_{12}$$

$$S_f = \{(0,1,0,0), (0,1,1,0), (0,1,1,1), (1,0,0,0), (1,0,0,1), \\ (1,0,1,0), (1,0,1,1), (1,1,0,0)\}$$

$$S_f = \{(0,1,1,1), (1,0,1,1), \\ (0,1,1,0), (1,0,0,1), (1,0,1,0), (1,1,0,0), \\ (0,1,0,0), (1,0,0,0)\}$$

Construirea primei tabele

- Capul de tabel conține numele variabilelor funcției.
- Pe fiecare linie se va completa suportul câte unui minterm din expresia funcției, grup după grup, în ordine crescătoare/descrescătoare a numărului valorilor de 1 din n -upluri.
- Grupurile se delimită prin linii orizontale. După completarea întregii multimi suport a funcției, se va trasa o linie dublă.
- Doar două grupuri vecine pot conține suporturi ale monoame adiacente, astfel că factorizarea se poate efectua doar între n -upluri din grupuri vecine.

Prima tabelă

Grupul

I

II

III

	x_1	x_2	x_3	x_4
	0	1	1	1
	1	0	1	1
	0	1	1	0
	1	0	0	1
	1	0	1	0
	1	1	0	0
	0	1	0	0
	1	0	0	0

$$S_f = \{(0,1,1,1), (1,0,1,1), (0,1,1,0), (1,0,0,1), (1,0,1,0), (1,1,0,0), (0,1,0,0), (1,0,0,0)\}$$

m_7

m_{11}

m_6

m_9

m_{10}

m_{12}

m_4

m_8

Operația de factorizare

- constă în introducerea unei linii noi în tabel, care conține aceleași valori (0, 1 sau –) pe coloanele variabilelor comune celor două monoame care participă la factorizare și simbolul „–” pe coloana corespunzătoare variabilei eliminate (o singură variabilă se va simplifica la un moment dat).
- Monoamele care participă la factorizare vor fi bifate deoarece acestea nu sunt maximale.
- Rezultatele factorizării a două grupuri de n -upluri vecine vor forma un nou grup (delimitat printr-o linie orizontală) de n -upluri în tabel, care poate fi utilizat mai departe în alte factorizări de ordin superior.
- Sfârșitul tuturor factorizărilor dintre două grupuri vecine de monoame cu același număr de simboluri „–” se marchează cu o dublă linie orizontală, pentru a sugera grafic faptul că s-a încheiat factorizarea de un anumit ordin și se trece la factorizare de ordin mai mare.
- Acest proces continuă până când nu se mai pot face factorizări între două grupuri vecine delimitate printr-o singură bară orizontală. Tabelul se încheie cu o triplă bară orizontală.

Factorizarea

Grupul

I

\checkmark

x_1	x_2	x_3	x_4	
0	1	1	1	m_7
1	0	1	1	m_{11}
0	1	1	0	m_6
1	0	0	1	m_9
1	0	1	0	m_{10}
1	1	0	0	m_{12}
0	1	0	0	m_4
1	0	0	0	m_8
0	1	1	-	$m_7 \vee m_6$

II

III

$IV=I+II$

Factorizare
simplă

Factorizarea

	Grupul	x_1	x_2	x_3	x_4	
I	✓	0	1	1	1	m_7
	✓	1	0	1	1	m_{11}
	✓	0	1	1	0	m_6
II	✓	1	0	0	1	m_9
	✓	1	0	1	0	m_{10}
	✓	1	1	0	0	m_{12}
	✓	0	1	0	0	m_4
III	✓	1	0	0	0	m_8
		0	1	1	—	$m_7 \vee m_6$
Factorizare simplă	$IV=I+II$	1	0	—	1	$m_{11} \vee m_9$
		1	0	1	—	$m_{11} \vee m_{10}$
		0	1	—	0	$m_6 \vee m_4$
$V=II+III$	✓	1	0	0	—	$m_9 \vee m_8$
	✓	1	0	—	0	$m_{10} \vee m_8$
		—	1	0	0	$m_{12} \vee m_4$
		1	—	0	0	$m_{12} \vee m_8$
		1	0	—	—	$m_{11} \vee m_9 \vee m_{10} \vee m_8$
Factorizare dublă	$VI=IV+V$					

Identificarea monoamelor maximale

- Multimea **monoamelor maximale** este formată din toate monoamele corespunzătoare liniilor nebificate din tabel.

Monoamele maximale

Grupul		x_1	x_2	x_3	x_4	
I	✓	0	1	1	1	m_7
	✓	1	0	1	1	m_{11}
	✓	0	1	1	0	m_6
II	✓	1	0	0	1	m_9
	✓	1	0	1	0	m_{10}
	✓	1	1	0	0	m_{12}
	✓	0	1	0	0	m_4
III	✓	1	0	0	0	m_8
		0	1	1	—	$m_7 \vee m_6 = \bar{x}_1 x_2 x_3 = max_1$
Factorizare simplă	$IV=I+II$	1	0	—	1	$m_{11} \vee m_9$
		1	0	1	—	$m_{11} \vee m_{10}$
		0	1	—	0	$m_6 \vee m_4 = \bar{x}_1 x_2 \bar{x}_4 = max_2$
Factorizare dublă	$V=II+III$	✓	1	0	—	$m_9 \vee m_8$
		1	0	—	0	$m_{10} \vee m_8$
		—	1	0	0	$m_{12} \vee m_4 = x_2 \bar{x}_3 \bar{x}_4 = max_3$
		1	—	0	0	$m_{12} \vee m_8 = x_1 \bar{x}_3 \bar{x}_4 = max_4$
	$VI=IV+V$	1	0	—	—	$m_{11} \vee m_9 \vee m_{10} \vee m_8 = x_1 \bar{x}_2 = max_5$

Mulțimea monoamelor maximale

- $M(f) = \{\bar{x}_1x_2x_3, \bar{x}_1x_2\bar{x}_4, x_2\bar{x}_3\bar{x}_4, x_1\bar{x}_3\bar{x}_4, x_1\bar{x}_2\}$
 $= \{max_1, max_2, max_3, max_4, max_5\}$

Identificarea monoamelor centrale

- se face utilizând un tabel care arată corespondența dintre monoamele maximale (pe coloane) și mintermii (pe linii) care au contribuit prin factorizare la obținerea acestora. O căsuță din tabel se marchează cu o steluță dacă mintermul corespunzător liniei a fost utilizat pentru obținerea monomului maximal de pe coloană.

se încercuiesc * singure pe linie

Monoamele centrale – conțin pe coloană *

monoame maximale mintermi	max_1	max_2	max_3	max_4	max_5
m_7	*				
m_{11}					*
m_6	*	*			
m_9					*
m_{10}					*
m_{12}			*	*	
m_4		*	*		
m_8				*	*

Mulțimea monoamelor centrale

- $C(f) = \{max_1, max_5\}$
- Cazul II

Identificarea formelor simplificate

- $g(x_1, x_2, x_3, x_4) = max_1 \vee max_5 = \bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2$

Se hașurează coloanele ce conțin $*$

monoame maximale mintermi	max_1	max_2	max_3	max_4	max_5
m_7	\otimes				
m_{11}					\otimes
m_6	*	*			
m_9					\otimes
m_{10}					\otimes
m_{12}			*	*	
m_4		*	*		
m_8				*	*

Se hașurează liniile ce conțin * hașurate

monoame maximale mintermi	max_1	max_2	max_3	max_4	max_5
m_7	*				
m_{11}					*
m_6	*	*			
m_9					*
m_{10}					*
m_{12}			*	*	
m_4		*	*		
m_8				*	*

Forma simplificată

- Se observă că cel mai simplu mod de a acoperi mintermii neacoperiți de funcția g (liniile nehașurate) este:
 - $h(x_1, x_2, x_3, x_4) = \max_3$
- $f'(x_1, x_2, x_3, x_4) = g(x_1, x_2, x_3, x_4) \vee h(x_1, x_2, x_3, x_4) =$
 $= \bar{x}_1 x_2 x_3 \vee x_1 \bar{x}_2 \vee x_2 \bar{x}_3 \bar{x}_4$

Logică computațională

Curs 13

Lector dr. Pop Andreea-Diana

Circuite logice

- circuite electronice simple

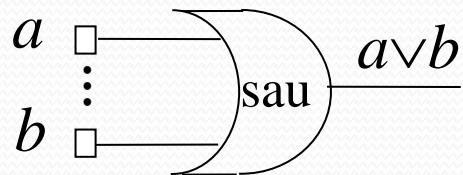
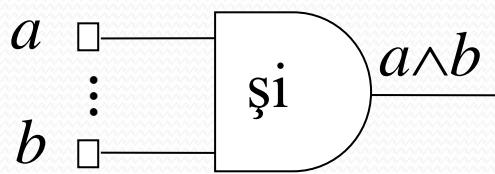


- modelarea – se face cu ajutorul *funcțiilor booleene* și a *circuitelor logice* care descriu algebric și grafic funcționarea acestora.

Portile logice

- sunt elementele de bază ale unui circuit logic
- sunt utilizate pentru modelarea circuitelor
- **Definiție:** O *poartă* este un minicircuit logic care realizează una dintre operațiile logice de bază: \wedge , \vee , \neg .

Porțile logice – conform standardelor IEEE



$$\frac{a \quad b}{a \wedge b}$$

legare în serie

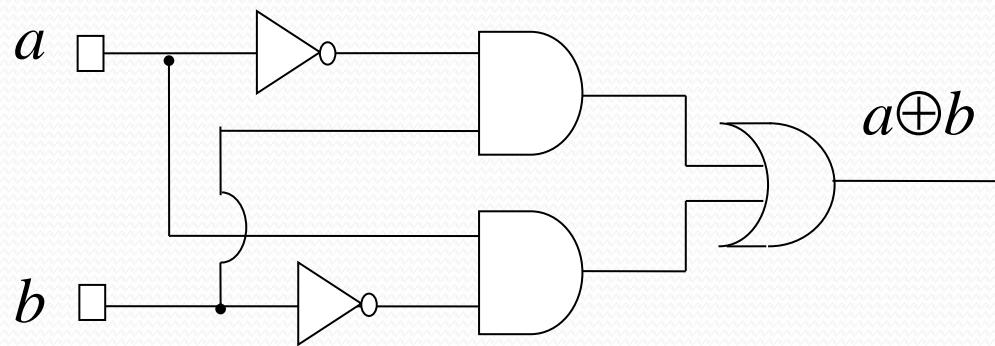
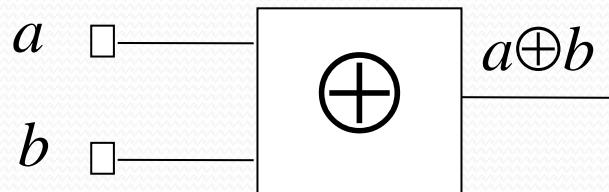
$$\frac{a \quad b}{a \vee b}$$

legare în paralel

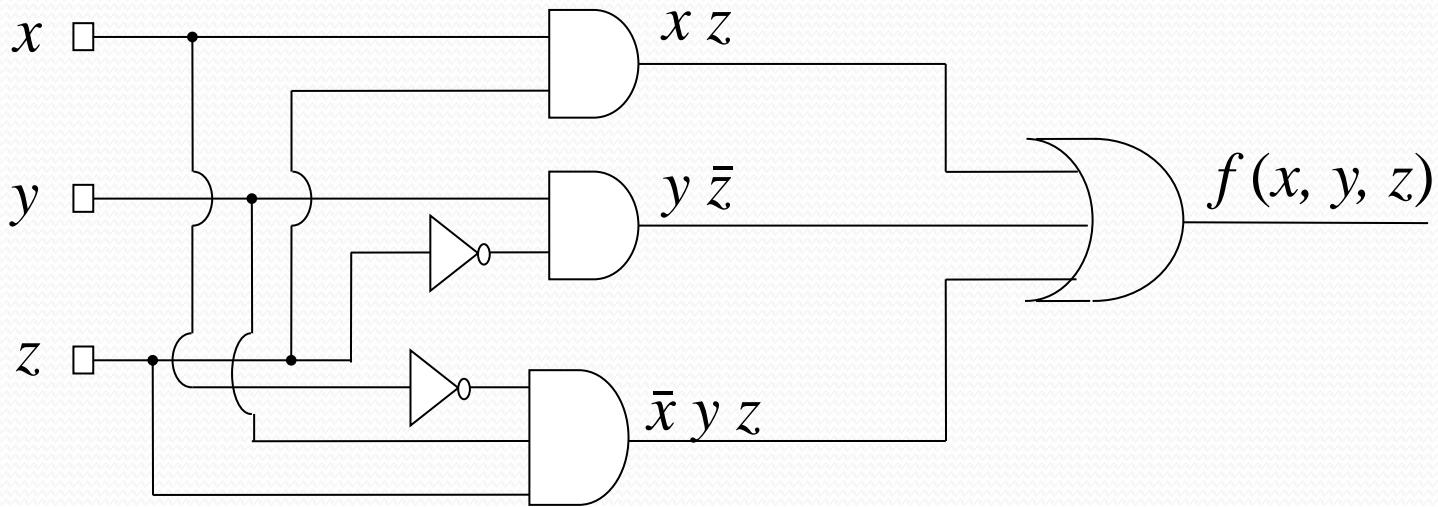
Circuite integrate

- 14-16 ”pini”
 - o parte portă de intrare
 - o parte sunt utilizate pentru conexiunea la curent
- Observație: forma disjunctivă este cel mai simplu de realizat

Exercițiu – desenați circuitul

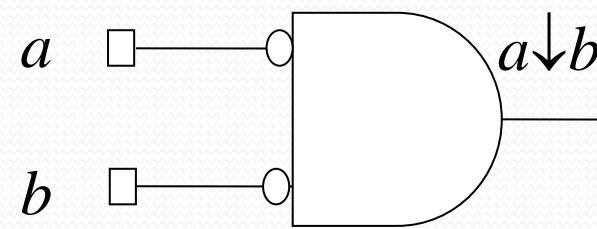
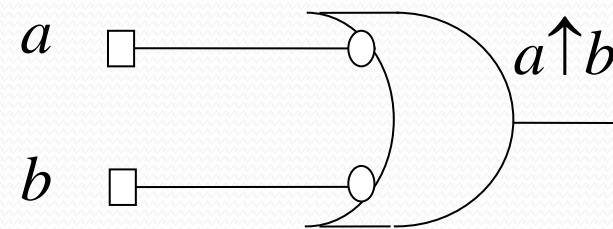
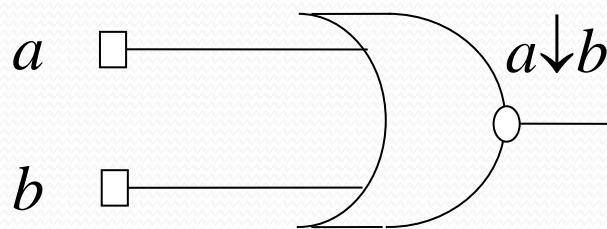
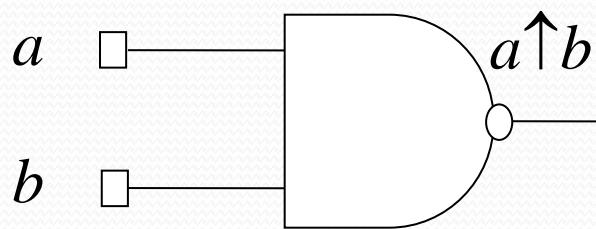
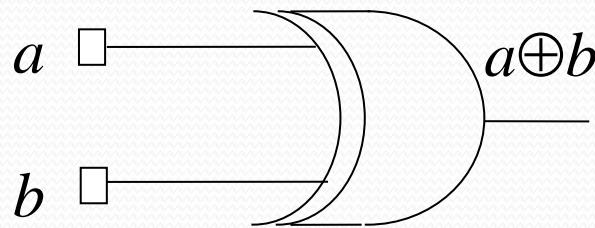


Exercițiu – $f(x, y, z) = ?$



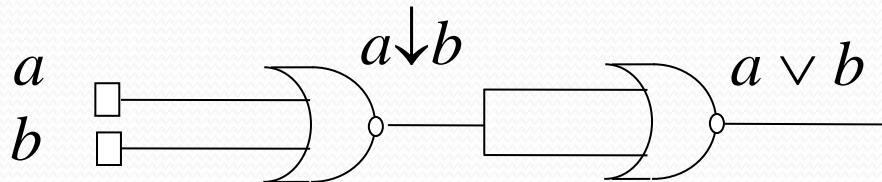
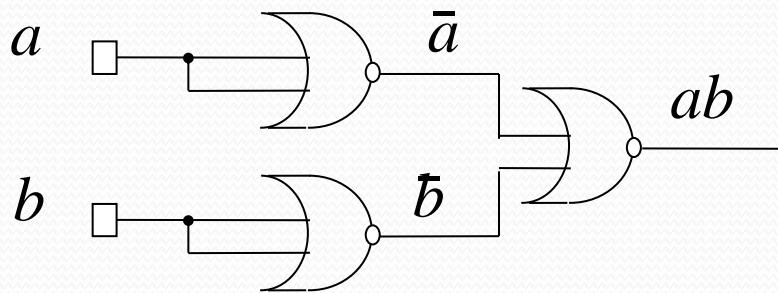
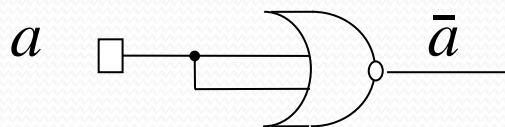
$$f(x, y, z) = x z \vee y \bar{z} \vee \bar{x} y z$$

Porti derivate



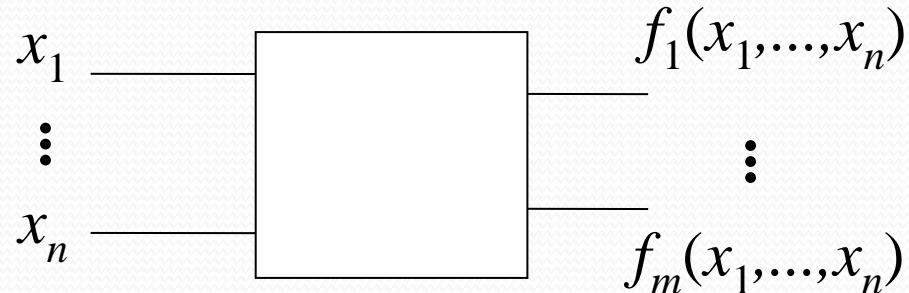
Exercițiu

- Desenați circuitele operațiilor logice „și”, „sau”, „not” folosind doar poartă „nor” / „nand”



Circuit combinational

- Un circuit logic cu m ieșiri se numește ***circuit combinational***.



Circuite logice combinaționale ∈ Hard-ul calculatorului

- decodorul
- circuitul comparator
- circuitul sumator
- detectorul de paritate
- ”shift”
- ...

Pașii principali pentru desenarea circuitelor

1. identificarea intrărilor (variabilelor) / ieșirilor (funcțiilor)
2. construirea tabelei de valori asociate
3. obținerea expresiilor funcțiilor
4. simplificarea funcțiilor
5. desenarea circuitului

Decodorul

1.

- intrare: 4 cifre binare - x_1, x_2, x_3, x_4
- ieșire: $f_i(x_1, x_2, x_3, x_4) = 1$ pentru $x_1x_2x_3x_4 \text{ (2)} = i_{(10)}$, $i = \overline{0, 9}$

Decodorul (2)

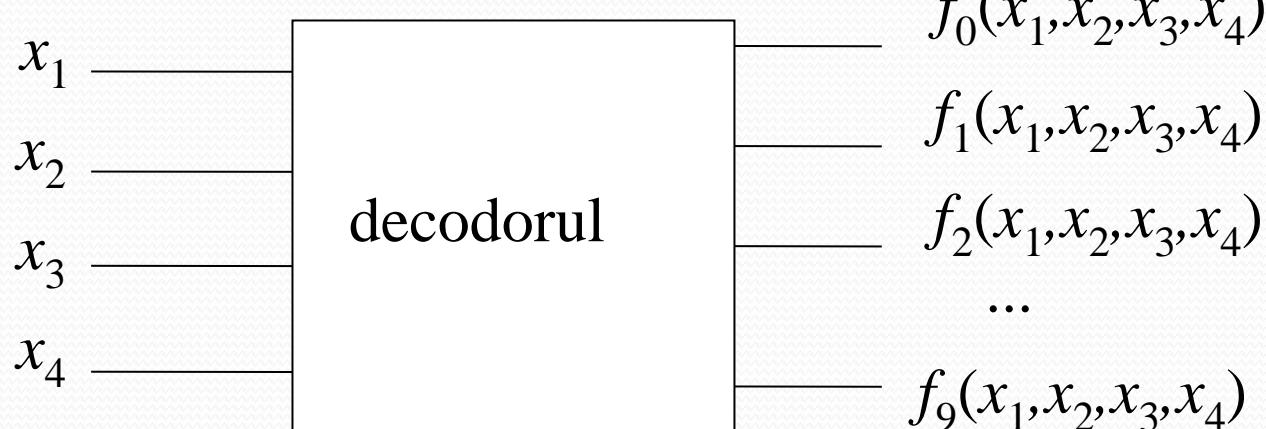
2.

3.

Circuitul decodor – forma generală

4. ...

5. ...



Circuitul comparator

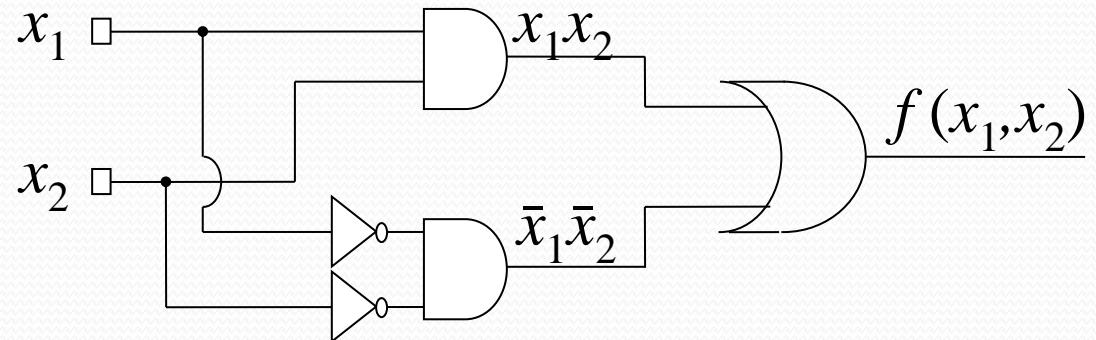
- verifică dacă două cifre binare sunt sau nu identice **1.**

2.

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

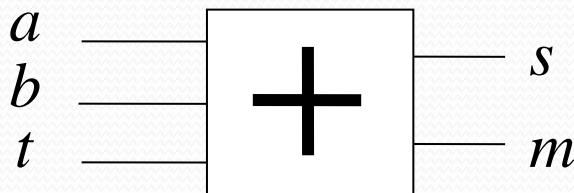
3. $f(x_1, x_2) = \bar{x}_1 \bar{x}_2 \vee x_1 x_2$ **4.**

5.



Sumatorul binar

- calculează suma a două cifre binare: a și b de pe aceeași poziție dintr-un număr binar
- 1.
- intrare: a, b , transportul t
 - ieșire: s ($= a + b$), transportul m



2.

t	a	b	s	m
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

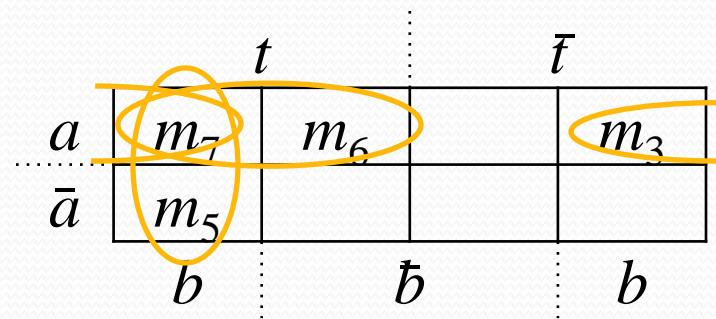
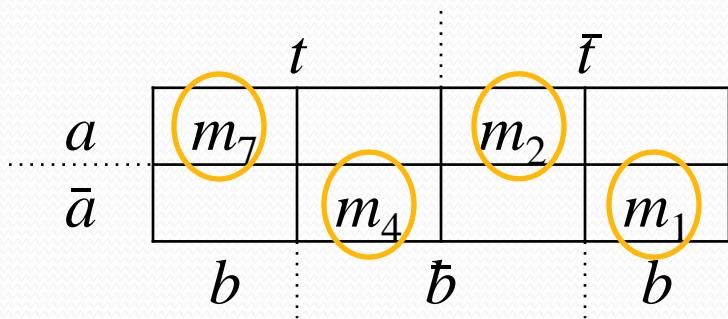
3. $s(a,b,t) = \bar{t} \bar{a}b \vee \bar{t} a\bar{b} \vee t\bar{a} \bar{b} \vee tab$

$$m(a,b,t) = \bar{t} ab \vee t\bar{a} b \vee ta\bar{b} \vee tab$$

4. Simplificarea

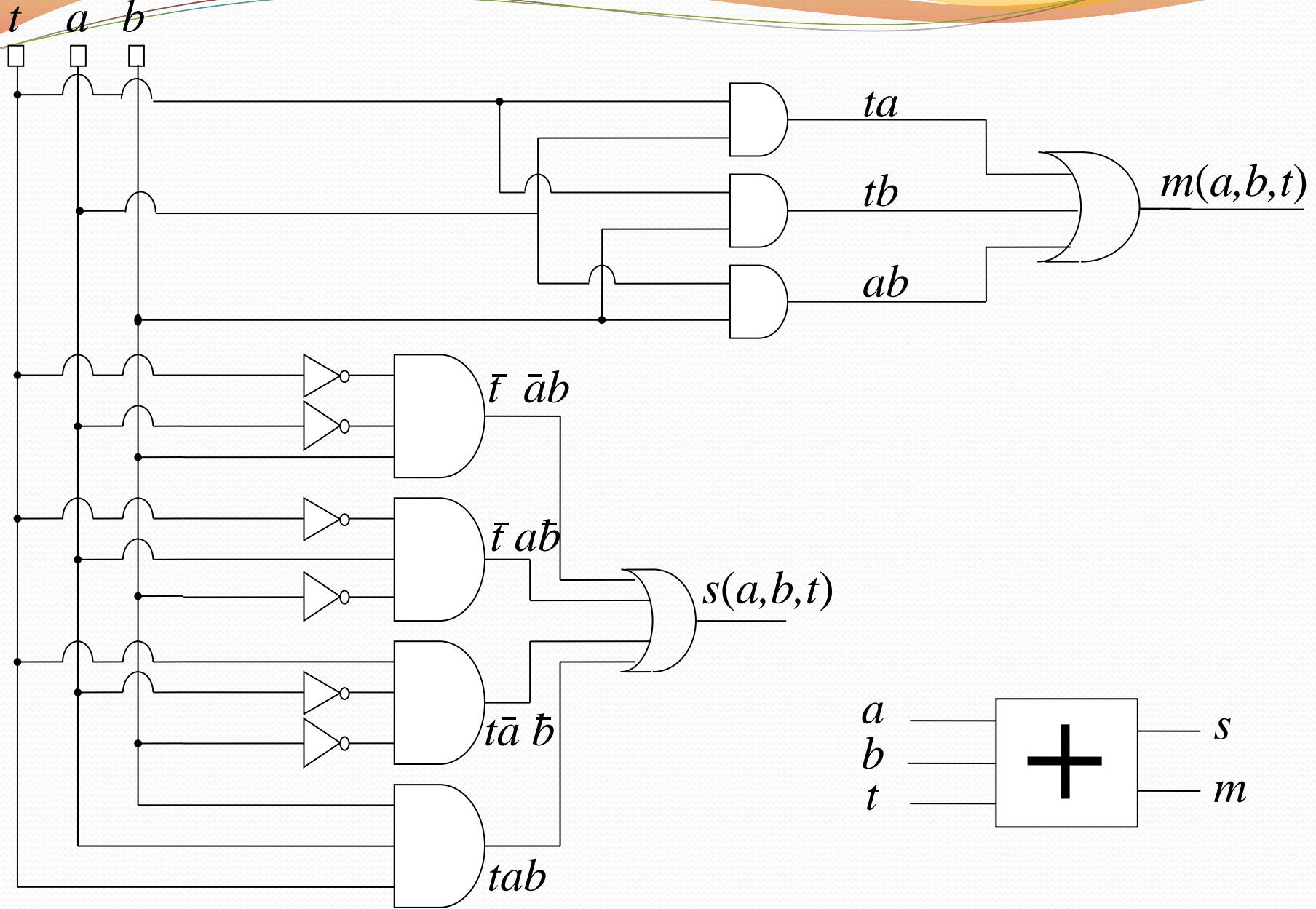
$$m(a,b,t) = \bar{t}ab \vee t\bar{a}b \vee tab \bar{b} \vee tab$$

$$s(a,b,t) = \bar{t}\bar{a}b \vee \bar{t}a\bar{b} \vee t\bar{a}\bar{b} \vee tab$$



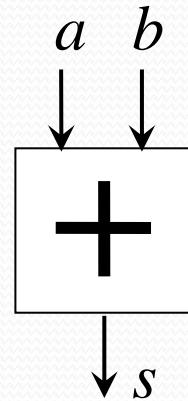
$$m(a,b,t) = ta \vee tb \vee ab$$

5. Circuitele

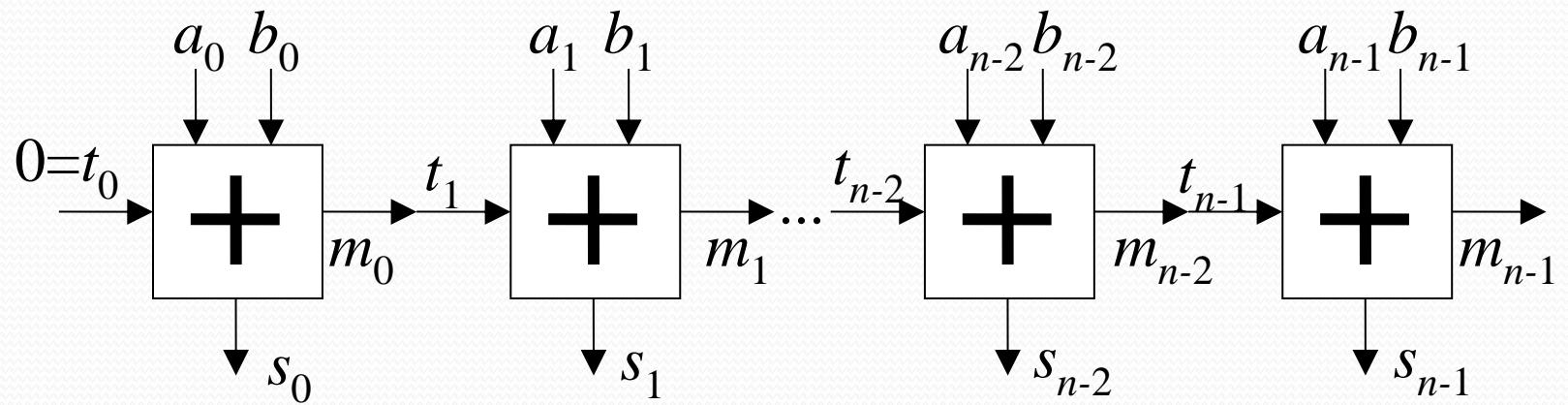


Sumatorul binar cu n poziții

- $a = a_{n-1} \dots a_0 {}_{(2)}$ și $b = b_{n-1} \dots b_0 {}_{(2)}$
- $s = s_{n-1} \dots s_0 {}_{(2)}$

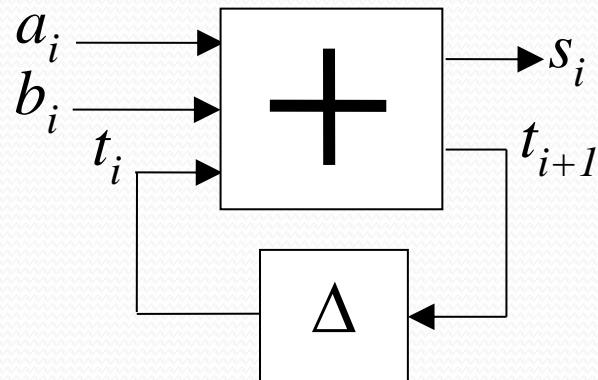


Componere de sumatoare simple



Circuit cu întârziere

- cifra de transport obținută la un pas se folosește în pasul următor



Indicații "anti - încâlcire"

