# Logică pentru informatică - Săptămâna 12
# Forme normale ale formulelor de ordinul I

În acest curs vom defini noţiunile de formule echivalente şi cea de formă normală prenex corespunzătoare unei formule din logica de ordinul I.

## 1 Formule echivalente

În diverse contexte, anumite formule pot avea același înțeles. De exemplu, formulele $\forall x.P(x,x)$ și $\forall y.P(y,y)$ au același înțeles în orice context. Un alt exemplu de formule cu acelasi inteles este $\neg\forall x.Q(x)$ şi $\exists x.\neg Q(x)$. Vom numi astfel de formule *echivalente*.

Anumite formule au același înțeles doar pentru o anumită interpretare a simbolurilor predicative și funcționale. De exemplu, dacă lucrăm într-o structură în care simbolul predicativ $P$ este interpretat printr-un predicat simetric, formulele $P(x,y)$ și respectiv $P(y,x)$ au același înțeles. Astfel de formule se numesc *echivalente în structura respectivă*.

Acest aspect este surprins de următoarele definiții.

**Definiția 1.1.** *Două formule $\varphi_1 \in$ LP1 și $\varphi_2 \in$ LP1 sunt echivalente în structura $S$ dacă, pentru orice $S$-atribuire $\alpha$,*

$$S, \alpha \models \varphi_1 \text{ ddacă } S, \alpha \models \varphi_2.$$

*Faptul că $\varphi_1$ și $\varphi_2$ sunt echivalente în structura $S$ se notează $\varphi_1 \stackrel{S}{\equiv} \varphi_2$.*

Cu alte cuvinte, două formule sunt echivalente într-o anumită structură $S$ dacă, evaluând valoarea de adevăr a formulelor în structura $S$, obținem același rezultat pentru ambele formule (ambele adevărate sau ambele false), indiferent de atribuirea $\alpha$ cu care lucrăm.

**Exemplul 1.1.** *Continuăm exemplele din cursurile anterioare. Considerăm signatura $\Sigma = (\{P\}, \{f, i, e\})$ și $\Sigma$-structura $S_1 = (\mathbb{Z}, \{=\}, \{+, -, 0\})$.*

1. *Avem că $P(x,y) \stackrel{S_1}{\equiv} P(y,x)$. De ce?*

   *Fie $\alpha$ o $S_1$-atribuire oarecare.*

*Avem* $S_1, \alpha \models P(x,y)$ *ddacă (prin definiția relației* $\models$*)* $P^{S_1}(\bar{\alpha}(x), \bar{\alpha}(y)) = 1$
*ddaca* $\bar{\alpha}(x) = \bar{\alpha}(y)$
*ddaca* $\alpha(x) = \alpha(y)$
*ddacă (prin simetria relației de egalitate)* $\alpha(y) = \alpha(y)$
*ddacă (prin definiția relației* $\models$*)* $S_1, \alpha \models P(y,x)$.

*Deci, pentru orice* $S_1$*-atribuire* $\alpha$*, avem:* $S_1, \alpha \models P(x,y)$ *ddacă* $S_1, \alpha \models P(y,x)$*, care este chiar definiția* $P(x,y) \overset{S_1}{\equiv} P(y,x)$.

2. *Avem că* $P(x_1, x_3) \overset{S_1}{\not\equiv} P(x_2, x_3)$*. De ce?*

   *Deoarece există o* $S_1$*-atribuire* $\alpha : \mathcal{X} \to \mathbb{Z}$*, definită prin* $\alpha(x_1) = 42, \alpha(x_2) = 7, \alpha(x_3) = 42$ *(pentru restul variabilelor nu este relevantă valoarea lor în atribuire) cu proprietatea că*

   $S_1, \alpha \models P(x_1, x_3)$ *(deoarece* $42 = 42$*), dar*
   $S_1, \alpha \not\models P(x_2, x_3)$ *(deoarece* $42 \neq 7$*).*

În cazul în care structura nu este fixată, avem următoarea definiție:

**Definiția 1.2.** *Două formule* $\varphi_1 \in$ `LP1` și $\varphi_2 \in$ `LP1` sunt echivalente *dacă, pentru orice structură* $S$ *și pentru orice* $S$*-atribuire* $\alpha$*,*

$$S, \alpha \models \varphi_1 \text{ ddacă } S, \alpha \models \varphi_2.$$

*Faptul că* $\varphi_1$ *și* $\varphi_2$ *sunt echivalente se notează* $\varphi_1 \equiv \varphi_2$.

**Exemplul 1.2.** *Continuăm exemplul anterior.*

1. *Avem că* $P(x,y) \not\equiv P(y,x)$*. De ce?*

   *Deoarece există o* $\Sigma$*-structură şi o atribuire în structura respectivă astfel încat cele două formule să ia valori de adevăr diferite.*

   *Fie structura* $S_5 = (\mathbb{Z}, \{<\}, \{+, -, 0\})$ *definită în cursul anterior şi* $S_5$*-atribuirea* $\alpha_6 : \mathcal{X} \to \mathbb{Z}$*, definită prin* $\alpha(x) = 2, \alpha(y) = 3$ *şi* $\alpha(z) = 1$ *pentru orice* $z \in \mathcal{X} \setminus \{x, y\}$.

   *Observați că singura diferență între* $S_1$ *și* $S_5$ *este faptul că simbolul predicativ* $P$ *este interpretat prin predicatul* $=$ *în* $S_1$*, în timp ce în* $S_5$ *este interpretat prin* $<$ *(relația mai mic strict peste numere întregi).*

   *Avem* $S_5, \alpha_6 \models P(x,y)$*, deoarece* $2 < 3$*, dar* $S_5, \alpha_6 \not\models P(y,x)$*, deoarece* $3 \not< 2$*. Deci formulele* $P(x,y)$ *și* $P(y,x)$ *nu sunt echivalente (chiar dacă sunt echivalente în structura* $S_1$*).*

2. *Avem că* $\forall x.P(x,z) \equiv \forall y.P(y,z)$. *De ce?*

   *Fie* $S$ *o* $\Sigma$*-structură oarecare cu domeniul* $D$ *și* $\alpha : \mathcal{X} \to D$ *o* $S$*-atribuire oarecare.*

   *Avem că*

   $$\begin{array}{ll} S,\alpha \models \forall x.P(x,z) & \textit{ddacă} \\ \textit{pentru orice } u \in D, \textit{ avem } S,\alpha[x \mapsto u] \models P(x,z) & \textit{ddacă} \\ \textit{pentru orice } u \in D, \textit{ avem } P^S\Big(\overline{\alpha[x \mapsto u]}(x), \overline{\alpha[x \mapsto u]}(z)\Big) = 1 & \textit{ddacă} \\ \textit{pentru orice } u \in D, \textit{ avem } P^S\Big(u, \alpha(z)\Big) = 1 & \textit{ddacă} \\ \textit{pentru orice } u \in D, \textit{ avem } P^S\Big(\overline{\alpha[y \mapsto u]}(y), \overline{\alpha[y \mapsto u]}(z)\Big) = 1 & \textit{ddacă} \\ \textit{pentru orice } u \in D, \textit{ avem } S,\alpha[y \mapsto u] \models P(y,z) & \textit{ddacă} \\ S,\alpha \models \forall y.P(y,z). & \end{array}$$

*Deci, pentru orice* $\Sigma$*-structură* $S$*, pentru orice* $S$*-atribuire* $\alpha$*, avem că*

$$S,\alpha \models \forall x.P(x,z) \textit{ ddacă } S,\alpha \models \forall y.P(y,z),$$

*care este chiar definiția faptului că* $\forall x.P(x,z) \equiv \forall y.P(y,z)$.

## 2 Forme normale şi Substituţii

Ca şi în cazul logicii propoziţionale, în cazul logicii de ordinul I, există o serie de forme normale corsepunzătoare formulelor.

Pentru a defini formele normale, reamintim noţiunea de *substituţie* :

- O *substituție* este o funcție $\sigma : \mathcal{X} \to \mathcal{T}$, cu proprietatea că $\sigma(x) \neq x$ pentru un număr finit de variabile $x \in \mathcal{X}$;
- *Domeniul substituției* $\sigma$ este multimea $dom(\sigma) = \{x \in \mathcal{X} \mid \sigma(x) \neq x\}$;
- *Extensia* substituției $\sigma$ la mulțimea termenilor este funcția $\sigma^\sharp : \mathcal{T} \to \mathcal{T}$, definită astfel:

  1. $\sigma^\sharp(x) = \sigma(x)$, pentru orice $x \in \mathcal{X}$;
  2. $\sigma^\sharp(c) = c$, pentru orice simbol constant $c \in \mathcal{F}_0$;
  3. $\sigma^\sharp(f(t_1, \ldots, t_n)) = f(\sigma^\sharp(t_1), \ldots, \sigma^\sharp(t_n))$, pentru orice simbol funcțional $f \in \mathcal{F}_n$ de aritate $n \in \mathbb{N}$ și orice termeni $t_1, \ldots, t_n \in \mathcal{T}$.

  Dacă $t \in \mathcal{T}$ este un termen, atunci $\sigma^\sharp(t) \in \mathcal{T}$ este termenul obținut din $t$ prin *aplicarea* substituției $\sigma$ (fiecare apariţie a unei variabile $x$ din $t$ este înlocuită cu termenul $\sigma(x)$).

- Dacă $dom(\sigma) = \{x_1, \ldots, x_n\}$, atunci substituția $\sigma$ se mai poate scrie în felul următor:

  $$\sigma = \{x_1 \mapsto \sigma(x_1), \ldots, x_n \mapsto \sigma(x_n)\}.$$

- *Restricția substituției $\sigma$ la mulțimea $V$ este o nouă substituție notată* $\sigma|_V : \mathcal{X} \to \mathcal{T}$, definită astfel:

  1. $\sigma|_V(x) = \sigma(x)$ pentru orice $x \in V$;
  2. $\sigma|_V(x) = x$ pentru orice $x \in \mathcal{X} \setminus V$.

  Practic, prin restricția unei substituții la o mulțime de variabile, se scot celelalte variabile din domeniul substituției.

- *Extensia* lui $\sigma$ la mulțimea formulelor este funcția $\sigma^\flat : \texttt{LP1} \to \texttt{LP1}$, definită astfel:

  1. $\sigma^\flat(P(t_1, \ldots, t_n)) = P(\sigma^\sharp(t_1), \ldots, \sigma^\sharp(t_n))$;
  2. $\sigma^\flat(\neg\varphi) = \neg\sigma^\flat(\varphi)$;
  3. $\sigma^\flat(\varphi_1 \wedge \varphi_2) = \sigma^\flat(\varphi_1) \wedge \sigma^\flat(\varphi_2)$;
  4. $\sigma^\flat(\varphi_1 \vee \varphi_2) = \sigma^\flat(\varphi_1) \vee \sigma^\flat(\varphi_2)$;
  5. $\sigma^\flat(\varphi_1 \to \varphi_2) = \sigma^\flat(\varphi_1) \to \sigma^\flat(\varphi_2)$;
  6. $\sigma^\flat(\varphi_1 \leftrightarrow \varphi_2) = \sigma^\flat(\varphi_1) \leftrightarrow \sigma^\flat(\varphi_2)$;
  7. $\sigma^\flat(\forall x.\varphi) = \forall x.(\rho^\flat(\varphi))$, unde $\rho = \sigma|_{dom(\sigma)\setminus\{x\}}$;
  8. $\sigma^\flat(\exists x.\varphi) = \exists x.(\rho^\flat(\varphi))$, unde $\rho = \sigma|_{dom(\sigma)\setminus\{x\}}$.

  Practic, pentru a obține formula $\sigma^\flat(\varphi)$ din formula $\varphi$, fiecare apariţie **liberă** a variabilei $x$ din formula $\varphi$ este înlocuită cu termenul $\sigma(x)$.

# 3 Forma normală prenex

**Definiția 3.1.** *O formulă* $\varphi$ este în formă normală prenex *dacă*

$$\varphi = Q_1x_1.Q_2x_2.\ldots.Q_nx_n.\varphi',$$

*unde:*

1. $Q_i \in \{\forall, \exists\}$ *(pentru orice* $1 \leq i \leq n$*);*
2. $\varphi'$ *nu conține cuantificatori.*

Practic, o formulă este în formă normală prenex (FNP), dacă toți cuantificatorii sunt "în fața formulei".

**Exemplul 3.1.**
- *formula* $\forall x.\exists y.(P(x, y) \wedge \neg P(z, y))$ *este în formă normală prenex;*
- *formula* $\forall x.((\exists y.P(x, y)) \wedge \neg P(z, y))$ *nu este în formă normală prenex.*

Orice formulă poate fi adusă în formă normală prenex, fapt surpins de următoarea teoremă:

**Teorema 3.1.** *Pentru orice formulă $\varphi \in$ `LP1`, există $\varphi' \in$ `LP1` astfel încât:*

1. *$\varphi'$ este în formă normală prenex;*
2. *$\varphi \equiv \varphi'$.*

*Formula $\varphi'$ este o* formă normală prenex a formulei $\varphi$.

În continuare, demonstrăm teorema de mai sus prin intermediul unui algoritm care calculează $\varphi'$ pornind de la $\varphi$. Pentru a prezenta algoritmul, avem nevoie de următoarea lemă:

**Lema 3.1** (Lema redenumirii)**.** *Fie $\varphi \in$ `LP1` o formulă și $x, y \in \mathcal{X}$ două variabile cu proprietatea că $y \notin vars(\varphi)$.*

*Atunci au loc echivalențele:*

$$\forall x.\varphi \equiv \forall y.(\sigma^\flat(\varphi)) \text{ și } \exists x.\varphi \equiv \exists y.(\sigma^\flat(\varphi)),$$

*unde $\sigma = \{x \mapsto y\}$.*

Cu alte cuvinte, comform lemei redenumirii, în formula $\forall x.\varphi$ putem înlocui cuantificatorul $\forall x$ (respectiv $\exists x$) cu un cuantificator $\forall y$ (respectiv $\exists y$) la alegere cu condiţia că $y$ să nu fie variabilă a formulei $\varphi$. De asemenea, apariţiile libere ale variabilei $x$ în $\varphi$ trebuie înlocuite cu $y$ prin aplicarea substituţiei $\sigma = \{x \mapsto y\}$ asupra formulei $\varphi$.

**Exemplul 3.2.** *Fie formula $\forall x.P(x, y)$. Deoarece $z \notin free(P(x, y))$, avem prin lema redenumirii că $\forall x.P(x, y) \equiv \forall z.P(z, y)$.*

*Atenție, $\forall x.P(x, y) \not\equiv \forall y.P(y, y)$ (echivalența nu poate fi explicată prin lema redenumirii deoarece $y \in free(P(x, y))$ și nici nu are loc).*

Suntem acum pregătiți să prezentăm, schițat, demonstrația Teoremei 3.1.

*Proof.* Se aplică următoarele echivalențe, de la stânga la dreapta:

1. $(\forall x.\varphi_1) \wedge \varphi_2 \equiv \forall x.(\varphi_1 \wedge \varphi_2)$, dacă $x \notin free(\varphi_2)$;
2. $(\forall x.\varphi_1) \vee \varphi_2 \equiv \forall x.(\varphi_1 \vee \varphi_2)$, dacă $x \notin free(\varphi_2)$;
3. $(\exists x.\varphi_1) \wedge \varphi_2 \equiv \exists x.(\varphi_1 \wedge \varphi_2)$, dacă $x \notin free(\varphi_2)$;
4. $(\exists x.\varphi_1) \vee \varphi_2 \equiv \exists x.(\varphi_1 \vee \varphi_2)$, dacă $x \notin free(\varphi_2)$;
5. $\neg\forall x.\varphi \equiv \exists x.\neg\varphi$;
6. $\neg\exists x.\varphi \equiv \forall x.\neg\varphi$.

În cazul în care una dintre primele patru echivalențe nu poate fi aplicată din cauza restricției $x \notin free(\varphi_2)$, trebuie să aplicăm mai întai lema redenumirii pentru a redenumi convenabil variabila legată $x$.

De asemenea, vom folosi, când este necesar, comutativitatea conectorilor $\wedge$ și $\vee$, adică:

1. $\varphi_1 \wedge \varphi_2 \equiv \varphi_2 \wedge \varphi_1$;

2. $\varphi_1 \vee \varphi_2 \equiv \varphi_2 \vee \varphi_1$.

Efectul primelor 6 echivalențe de mai sus este de muta cuantificatorii, în arborele formulei, deasupra conectorilor $\wedge, \vee, \neg$, asigurând astfel terminarea algoritmului și faptul că într-un final toți cuantificatorii vor fi cât mai apropiați de rădăcina arborelui.

Pentru a trata conectorii $\rightarrow, \leftrightarrow$, putem folosi "traducerea" lor cu ajutorul conectorilor $\wedge, \vee, \neg$:

1. $\varphi_1 \rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$;

2. $\varphi_1 \leftrightarrow \varphi_2 \equiv (\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)$.

$\square$

În continuare, dăm un exemplu de calcul al unei forme normale prenex pentru o formulă, folosind algoritmul de mai sus.

**Exemplul 3.3.** *Fie formula* $\varphi = \Big(\forall x.\neg\big(P(x,x) \wedge \neg\exists y.P(x,y)\big)\Big) \wedge P(x,x)$.

*Nu putem aplica prima echivalență pentru a aduce cuantificatorul* $\forall x$ *în fața formulei deoarece* $x \in \mathit{free}(P(x,x))$. *Așadar trebuie să aplicăm întâi lema redenumirii (L.R.):*

$$\begin{array}{ccl}
\varphi & = & \Big(\forall x.\neg\big(P(x,x) \wedge \neg\exists y.P(x,y)\big)\Big) \wedge P(x,x) \\
 & \overset{L.R.}{\equiv} & \Big(\forall z.\neg\big(P(z,z) \wedge \neg\exists y.P(z,y)\big)\Big) \wedge P(x,x) \\
 & \overset{1}{\equiv} & \forall z.\Big(\neg\big(P(z,z) \wedge \neg\exists y.P(z,y)\big) \wedge P(x,x)\Big) \\
 & \overset{6}{\equiv} & \forall z.\Big(\neg\big(P(z,z) \wedge \forall y.\neg P(z,y)\big) \wedge P(x,x)\Big) \\
 & \overset{commutativity\ \wedge}{\equiv} & \forall z.\Big(\neg\big((\forall y.\neg P(z,y)) \wedge P(z,z)\big) \wedge P(x,x)\Big) \\
 & \overset{1}{\equiv} & \forall z.\Big(\neg\big(\forall y.(\neg P(z,y) \wedge P(z,z))\big) \wedge P(x,x)\Big) \\
 & \overset{commutativity\ \wedge}{\equiv} & \forall z.\Big(\neg\big(\forall y.(P(z,z) \wedge \neg P(z,y))\big) \wedge P(x,x)\Big) \\
 & \overset{5}{\equiv} & \forall z.\Big(\big(\exists y.\neg(P(z,z) \wedge \neg P(z,y))\big) \wedge P(x,x)\Big) \\
 & \overset{3}{\equiv} & \forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x)).
\end{array}$$

*Așadar, am găsit că formula* $\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$ *este o formă normală prenex a formulei* $\Big(\forall x.\neg\big(P(x,x) \wedge \neg\exists y.P(x,y)\big)\Big) \wedge P(x,x)$.

*Când facem calculele, de obicei nu marcăm explicit aplicarea unei comutativități și scriem mai pe scurt, în felul următor:*

$$
\begin{aligned}
\varphi &= \Big(\forall x.\neg\big(P(x,x) \wedge \neg\exists y.P(x,y)\big)\Big) \wedge P(x,x) \\
&\stackrel{L.R.}{\equiv} \Big(\forall z.\neg\big(P(z,z) \wedge \neg\exists y.P(z,y)\big)\Big) \wedge P(x,x) \\
&\stackrel{1}{\equiv} \forall z.\Big(\neg\big(P(z,z) \wedge \neg\exists y.P(z,y)\big) \wedge P(x,x)\Big) \\
&\stackrel{6}{\equiv} \forall z.\Big(\neg\big(P(z,z) \wedge \forall y.\neg P(z,y)\big) \wedge P(x,x)\Big) \\
&\stackrel{1}{\equiv} \forall z.\Big(\neg\big(\forall y.(P(z,z) \wedge \neg P(z,y))\big) \wedge P(x,x)\Big) \\
&\stackrel{5}{\equiv} \forall z.\Big(\big(\exists y.\neg(P(z,z) \wedge \neg P(z,y))\big) \wedge P(x,x)\Big) \\
&\stackrel{3}{\equiv} \forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x)).
\end{aligned}
$$

# 4 Formule închise

**Definiția 4.1.** *O formulă $\varphi \in$ `LP1` este* închisă *dacă free($\varphi$) $= \emptyset$.*

Cu alte cuvinte, dacă o formulă nu are variabile libere, aceasta se numește închisă. Formulele închise se mai numesc și *propoziții* (engl. sentences).

**Definiția 4.2.** *O formulă care nu este închisă se numește* deschisă.

**Exemplul 4.1.** *Formula $\forall x.P(x,x) \wedge \exists y.P(y,x)$ este o formulă inchisă deoarece free($\forall x.P(x,x) \wedge \exists y.P(y,x)$) $= \emptyset$.*

*Formula $\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$ nu este închisă (este deschisă), deoarece free($\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$) $= \{x\}$.*

**Definiția 4.3.** *Fie $\varphi \in$ `LP1` o formulă și free($\varphi$) $= \{x_1, \ldots, x_n\}$ mulțimea variabilelor libere ale acesteia.*

*Formula*

$$\exists x_1.\exists x_2.\ldots.\exists x_n.\varphi$$

*se numește închiderea existențială a formulei $\varphi$.*

**Observația 4.1.** *Închiderea existențială a unei formule este o formulă închisă.*

**Exemplul 4.2.** *Închiderea existențială a formulei $\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$ este $\exists x.\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$.*

**Definiția 4.4.** *Două formule $\varphi_1 \in$ `LP1` și $\varphi_2 \in$ `LP1` sunt* echisatisfiabile, *dacă:*

1. *sau atât $\varphi_1$ cât și $\varphi_2$ sunt satisfiabile;*

2. *sau nici $\varphi_1$ și nici $\varphi_2$ nu sunt satisfiabile.*

Cu alte cuvinte, singurele cazuri când două formule nu sunt echisatisfiabile sunt când una dintre formule este satisfiabilă, iar cealaltă nu este satisfiabilă.

**Teorema 4.1.** *Orice formulă este echisatisfiabilă cu închiderea ei existențială.*

**Definiția 4.5.** *Fie* $\varphi \in$ `LP1` *o formulă și free*$(\varphi) = \{x_1, \ldots, x_n\}$ *mulțimea variabilelor libere ale acesteia.*

*Formula*

$$\forall x_1.\forall x_2.\ldots.\forall x_n.\varphi$$

*se numește* închiderea universală *a formulei* $\varphi$.

**Observația 4.2.** *Închiderea universală a unei formule este o formulă închisă.*

**Exemplul 4.3.** *Închiderea universală a formulei* $\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$ *este* $\forall x.\forall z.\exists y.(\neg(P(z,z) \wedge \neg P(z,y)) \wedge P(x,x))$.

**Teorema 4.2.** *O formulă este validă dacă și numai dacă închiderea ei universală este validă.*