

# Logică pentru informatică - note de curs

Universitatea Alexandru Ioan Cuza, Iași  
Facultatea de Informatică  
Anul universitar 2025-2026

Ştefan Ciobâcă  
Andrei Arusoae  
Rodica Condurache  
Cristian Masalagiu



# Cuprins

<b>1 Motivație și introducere</b>	<b>5</b>
<b>2 Structuri și signaturi</b>	<b>7</b>
<b>3 Sintaxa logicii de ordinul I</b>	<b>11</b>
3.1 Alfabetul . . . . .	11
3.2 Termen . . . . .	12
3.3 Formule atomice . . . . .	13
3.4 Formule de ordinul I . . . . .	14
3.5 Paranteze . . . . .	16
3.6 Modelarea în LPI a afirmațiilor din limba română . . . . .	17
3.7 Modelarea în LPI a afirmațiilor despre aritmetică . . . . .	18
3.8 Variabilele unei formule . . . . .	19
3.9 Domeniul de vizibilitate al unui cuantificator - analogie cu limbajele de programare . . . . .	21
3.10 Apariții libere și legate ale variabilelor . . . . .	22
3.11 Variabile libere și variabile legate . . . . .	24
3.12 Domeniul de vizibilitate și parantetizarea formulelor . . . . .	25
3.13 Fișă de exerciții . . . . .	26
<b>4 Semantica formulelor logicii de ordinul I</b>	<b>29</b>
4.1 Atribuirri . . . . .	30
4.2 Valoarea de adevăr a unei formule de ordinul I . . . . .	32
4.3 Satisfiabilitate într-o structură fixată . . . . .	35
4.4 Validitate într-o structură fixată . . . . .	36
4.5 Satisfiabilitate . . . . .	36
4.6 Validitate . . . . .	36
4.7 Consecință semantică . . . . .	37
4.8 Multime consistentă de formule . . . . .	37
4.9 Formule echivalente . . . . .	38
4.10 Fișă de exerciții . . . . .	40

<b>5 Deductia naturală</b>	<b>43</b>
5.1 Substituții . . . . .	43
5.2 Secvențe . . . . .	46
5.3 Reguli de inferență . . . . .	47
5.4 Sistem deductiv . . . . .	48
5.5 Demonstrație formală . . . . .	49
5.6 Deductia naturală . . . . .	50
5.6.1 Regulile pentru conjuncții . . . . .	50
5.6.2 Regulile pentru implicații . . . . .	51
5.6.3 Regulile pentru disjuncții . . . . .	53
5.6.4 Regulile pentru negații . . . . .	54
5.6.5 Eliminarea cuantificatorului universal. . . . .	56
5.6.6 Introducerea cuantificatorul existențial. . . . .	57
5.6.7 Introducerea cuantificatorului universal. . . . .	58
5.6.8 Eliminarea cuantificatorului existențial . . . . .	58
5.6.9 Alte reguli . . . . .	59
5.7 Sistemul deductiv al deducției naturale . . . . .	60
5.8 Corectitudinea și completitudinea deducției naturale pentru logica de ordinul I . . . . .	61
5.9 Fișă de exerciții . . . . .	62
<b>6 Forme normale ale formulelor de ordinul I</b>	<b>63</b>
6.1 Formule echivalente . . . . .	63
6.2 Forme normale și Substituții . . . . .	65
6.3 Forma normală Prenex . . . . .	66
6.3.1 Fișă de exerciții . . . . .	69
6.4 Formule închise . . . . .	70
6.5 Forma normală Skolem . . . . .	72
6.6 Forma normală conjunctivă . . . . .	74
6.7 Forma normală Skolem clauzală . . . . .	75
6.7.1 Fișă de exerciții . . . . .	78
<b>7 Rezoluția de bază</b>	<b>79</b>
7.1 Un exemplu . . . . .	80
7.2 Fișă de exerciții . . . . .	82
<b>8 Rezoluția pentru LP1</b>	<b>83</b>
8.1 Unificare . . . . .	83
8.2 Cel mai general unificator . . . . .	84
8.3 Problemă de unificare . . . . .	85
8.4 Rezoluție de ordinul I . . . . .	89
8.5 Fișă de exerciții . . . . .	91

# Capitolul 1

## Motivătie și introducere

Logica de ordinul I, pe care o vom studia în continuare, este o extensie a logicii propoziționale, extensie care aduce un plus de expresivitate. Expresivitatea adițională este necesară pentru a putea modela anumite afirmații care nu pot fi exprimate în logica propozițională.

În logica propozițională, nu putem exprima într-un mod natural următoarea afirmație: *Orice om este muritor*.

Pentru a modela o afirmație în logica propozițională, identificăm întâi propozițiile atomice. Apoi asociem fiecărei propoziții atomice o variabilă propozițională. Propozițiile atomice sunt propozițiile care nu pot fi împărțite în alte propoziții mai mici, care să fie conectate între ele prin conectorii logici  $\neg$ ,  $\wedge$ ,  $\vee$ ,  $\rightarrow$  și respectiv  $\leftrightarrow$ .

Observăm că afirmația *Orice om este muritor* nu poate fi descompusă în afirmații indivizibile legate între ele prin conectorii logicii propoziționale, după cum este descris mai sus. Așadar, în logica propozițională, afirmația este atomică. Asociem întregii afirmații o variabilă propozițională  $p \in A$ .

Acum să modelăm afirmația *Socrate este om*. Evident, acestei a două afirmații trebuie să ii asociem o altă variabilă propozițională  $q \in A$ . Să presupunem că știm că  $p$  și  $q$  sunt adevărate. Formal, știm că lucrăm cu o atribuire  $\tau : A \rightarrow B$  astfel încât  $\tau(p) = 1$  și  $\tau(q) = 1$ . Putem trage concluzia ca afirmația *Socrate este muritor* este adevărată în atribuirea  $\tau$ ?

Nu, deoarece afirmației *Socrate este muritor* ar trebui să ii asociem o a treia variabilă propozițională  $r$  și nu putem trage nicio concluzie asupra lui  $\tau(r)$  din faptul că  $\tau(p) = 1$  și  $\tau(q) = 1$ . Deci, din semantica logicii propoziționale, nu putem trage concluzia că  $r$  este adevărată în orice atribuire în care  $p$  și  $q$  sunt adevărate, în ciuda faptului că, dacă *orice om este muritor* și *Socrate este om* atunci sigur *Socrate este muritor*. Această diferență între realitate și modelarea noastră ne indică faptul că modelarea nu este suficient

de bună.

Logica de ordinul I aduce, în plus față de logica propozițională, noțiunea de *cuantificator* (existențial sau universal) și noțiunea de *predicat*. Cuantificatorul universal este notat cu  $\forall$  (de la litera *A* întoarsă – *all* în limba engleză), iar cuantificatorul existențial este notat cu  $\exists$  (de la litera *E* întoarsă – *exists* în limba engleză).

Un predicat este o afirmație a cărei valoare de adevăr depinde de zero sau mai mulți parametri. De exemplu, pentru afirmația de mai sus, vom folosi două predicate:  $\text{O}$  și  $\text{M}$ . Predicatul  $\text{O}$  va fi definit astfel:  $\text{O}(x)$  va fi adevărat când  $x$  este om. Predicatul  $\text{M}(x)$  este adevărat când  $x$  este muritor. Deoarece predicatele de mai sus au fiecare câte un singur argument/parametru, ele se numesc predicate *unare*. Predicatelor generalizează variabilele propoziționale prin faptul că pot primi argumente. De fapt, variabilele propoziționale pot fi văzute ca predicate fără argumente.

Astfel, afirmația *orice om este muritor* va fi modelată prin formula

$$(\forall x. (\text{O}(x) \rightarrow \text{M}(x))),$$

care este citită astfel: *pentru orice  $x$ , dacă  $\text{O}$  de  $x$ , atunci  $\text{M}$  de  $x$ .* Afirmația *Socrate este om* va fi modelată prin formula  $\text{O}(s)$ , unde  $s$  este o *constantă* prin care înțelegem Socrate, la fel cum prin constanta  $0$  ne referim la numărul natural *zero*. De exemplu,  $\text{O}(s)$  este adevărat (deoarece  $s$  denotă un om), dar  $\text{O}(l)$  este fals dacă, spre exemplu,  $l$  este o constantă care *ține locul cătelului Lăbus*.

Afirmația *Socrate este muritor* va fi reprezentată prin  $\text{M}(s)$  (deoarece constanta  $s$  se referă la Socrate). Afirmația  $\text{M}(s)$  este adevărată deoarece Socrate este muritor; la fel și afirmația  $\text{M}(l)$  este adevărată.

Vom vedea că în logica de ordinul I, formula  $\text{M}(s)$  este consecință a formulelor  $(\forall x. (\text{O}(x) \rightarrow \text{M}(x)))$  și respectiv  $\text{O}(s)$ . În acest sens, logica de ordinul I este suficient de expresivă pentru a explica din punct de vedere teoretic raționamentul prin care putem deduce că *Socrate este muritor* din faptul că *Orice om este muritor* și din faptul că *Socrate este om*.

## Capitolul 2

# Structuri și signaturi

Cu siguranță ați întâlnit deja mai multe formule din logica de ordinul I, fără să știți neapărat că aveți de a face cu logica de ordinul I. Fie următoare formulă:

$$\varphi = \left( \forall x. (\forall y. (x < y \rightarrow \exists z. (x < z \wedge z < y))) \right).$$

Formula folosește un simbol  $<$  căruia îi corespunde un predicat binar  $<$  (adică o relație binară) definit astfel:  $<(x, y)$  este adevărat dacă  $x$  este mai mic strict decât  $y$ . Pentru multe predicate binare (inclusiv pentru  $<$ ), pentru a simplifica scrierea, folosim notația infixată  $(x < y)$  în loc de notația prefixată  $(<(x, y))$ .

Este formula  $\varphi$  de mai sus adevărată? Formula afirmă că între orice două valori ale variabilelor  $x, y$  există o a treia valoare, a variabilei  $z$ . Formula este adevărată dacă domeniul variabilelor  $x, y, z$  este  $\mathbb{R}$ , dar este falsă dacă domeniul este  $\mathbb{N}$  (între orice două numere reale există un al treilea, dar între două numere naturale consecutive nu există niciun alt număr natural).

În general, formulele de ordinul I se referă la o anumită *structură matematică*.

**Definiția 1** (Structură matematică). *O structură matematică este un triplet  $S = (D, Pred, Fun)$ , unde:*

- $D$  este o mulțime nevidă numită domeniu;
- fiecare  $P \in Pred$  este predicat (de o aritate oarecare) peste mulțimea  $D$ ;
- fiecare  $f \in Fun$  este funcție (de o aritate oarecare) peste mulțimea  $D$ .

Iată câteva exemple de structuri matematice:

1.  $(\mathbb{N}, \{\langle, =\}, \{+, 0, 1\})$ ;

Domeniul structurii este mulțimea numerelor naturale. Structura conține două predicate:  $\langle$  și  $=$ , ambele de aritate 2. Predicatul  $\langle$  este predicatul *mai mic* pe numere naturale, iar predicatul  $=$  este predicatul de *egalitate* a numerelor naturale.

Funcția binară  $+$ :  $\mathbb{N}^2 \rightarrow \mathbb{N}$  este funcția de adunare a numerelor naturale, iar structura conține și constantele  $0 \in \mathbb{N}$  și  $1 \in \mathbb{N}$ .

2.  $(\mathbb{R}, \{\langle, =\}, \{+, -, 0, 1\})$ ;

Această structură conține două predicate binare,  $\langle$  și  $=$ , precum și patru funcții peste  $\mathbb{R}$ : funcția binară  $+$ , funcția unară  $-$  și constantele  $0, 1 \in \mathbb{R}$ .

3.  $(\mathbb{Z}, \{\langle, =\}, \{+, -, 0, 1\})$ ;

Această structură este similară cu structura precedentă, dar domeniul este mulțimea numerelor întregi.

4.  $(B, \emptyset, \{\cdot, +, \neg\})$ ;

Această structură este o algebră booleană, unde domeniul este mulțimea valorilor de adevăr, iar funcțiile sunt cele cunoscute din prima jumătate a semestrului. Astfel de structuri, fără niciun predicat, se numesc *structuri algebrice*.

5.  $(\mathbb{R}, \{\langle\}, \emptyset)$ .

Această structură conține doar un predicat de aritate 2 (relația *mai mic* peste  $\mathbb{R}$ ) și nicio funcție. Structurile care nu conțin funcții se numesc structuri relaționale. Structurile relaționale cu domeniu finit se mai numesc baze de date relaționale și se studiază în anul 2.

Când avem o formulă de ordinul I și dorim să îi evaluăm valoarea de adevăr, trebuie să fixăm structura în care lucrăm. Revenind la formula de mai devreme:

$$\varphi = \left( \forall x. (\forall y. (x < y \rightarrow \exists z. (x < z \wedge z < y))) \right),$$

avem că această formulă este adevărată în structura  $(\mathbb{R}, \{\langle, =\}, \{+, -, 0, 1\})$  (între orice două numere reale distințe există cel puțin un număr real) dar este falsă în structura  $(\mathbb{Z}, \{\langle, =\}, \{+, -, 0, 1\})$  (deoarece nu între orice două numere întregi putem găsi un alt număr întreg – de exemplu între două numere întregi consecutive nu există niciun întreg). În primul caz, domeniul variabilelor  $x, y, z$  este  $\mathbb{R}$  și simbolului  $<$  îi corespunde predicatul  $\langle \subseteq \mathbb{R}^2$ . În al doilea caz, domeniul variabilelor  $x, y, z$  este  $\mathbb{Z}$  și simbolului  $<$  îi corespunde predicatul  $\langle \subseteq \mathbb{Z}^2$ .

Este posibil ca două structuri diferite să aibă un set de predicate și de funcții cu același nume. De exemplu, chiar structurile de mai devreme,  $(\mathbb{R}, \{<, =\}, \{+, -, 0, 1\})$  și respectiv  $(\mathbb{Z}, \{<, =\}, \{+, -, 0, 1\})$ . Deși predicatul  $< \subseteq \mathbb{R}^2$  este diferit de predicatul  $< \subseteq \mathbb{Z}^2$ , ele au același nume:  $<$ .

În general, în Matematică și în Informatică, nu facem diferența între un predicat și numele lui, respectiv între o funcție și numele funcției, dar în Logică diferența este extrem de importantă. În particular, dacă ne referim la numele unei funcții vom folosi sintagma *simbol funcțional*, iar dacă ne referim la numele unui predicat vom folosi sintagma *simbol predicativ*. De ce este importantă diferența dintre un simbol predicativ și un predicat? Deoarece vom avea (ne)voie să asociem simbolului predicativ diverse predicate, analog modului în care unei variabile într-un limbaj de programare imperativ îi putem asocia diverse valori.

Când ne intereseză doar numele funcțiilor și predicatelor (nu și funcțiile și respectiv predicatele în sine), vom utiliza signaturi:

**Definiția 2** (Signatură). O signatură  $\Sigma$  este un tuplu  $\Sigma = (\mathcal{P}, \mathcal{F})$  unde  $\mathcal{P}$  este o mulțime de simboluri predicative și  $\mathcal{F}$  este o mulțime de simboluri funcționale. Fiecare simbol  $s$  (predicativ sau funcțional) are asociat un număr natural pe care îl vom numi aritatea simbolului și îl vom nota cu  $ar(s)$ .

Unei signaturi îi putem asocia mai multe structuri:

**Definiția 3** ( $\Sigma$ -structuri). Dacă  $\Sigma = (\mathcal{P}, \mathcal{F})$  este o signatură, o  $\Sigma$ -structură este orice structură  $S = (D, Pred, Fun)$  astfel încât fiecărui simbol predicativ (sau funcțional) îi corespunde în mod unic un predicat (respectiv, o funcție).

**Exemplul 4.** Fie  $\Sigma = (\{\text{P, Q}\}, \{\text{f, i, a, b}\})$  unde  $\text{P, Q}$  sunt simboluri predicative de aritate  $ar(\text{P}) = ar(\text{Q}) = 2$  și  $\text{f, i, a, b}$  sunt simboluri funcționale cu aritățile:  $ar(\text{f}) = 2$ ,  $ar(\text{i}) = 1$  și  $ar(\text{a}) = ar(\text{b}) = 0$ .

Avem că  $(\mathbb{R}, \{<, =\}, \{+, -, 0, 1\})$  și respectiv  $(\mathbb{Z}, \{<, =\}, \{+, -, 0, 1\})$  sunt  $\Sigma$ -structuri.

**Observație.** Dupa cum se poate observa și în Exemplul 4, pentru simboluri predicative (e.g.,  $\text{P, Q}$ ) vom utiliza o culoare diferită față de culoarea simbolurilor funcționale (e.g.,  $\text{f, i, a, b}$ ). Pentru predicatele și funcțiile din structuri vom utiliza fontul obișnuit pentru formule matematice.

De reținut!

Structură = domeniu + predicate + funcții

Signatură = simboluri predicative + simboluri funcționale

Unei signaturi  $\Sigma$  îi putem asocia mai multe structuri, numite  $\Sigma$ -structuri.

**Notatie.** Multimea simbolurilor predicative dintr-o  $\Sigma$ -structură de aritate  $n$  este notată cu  $\mathcal{P}_n = \{P \mid \text{ar}(P) = n\}$ , iar multimea simbolurilor funcționale de aritate  $n$  este notată cu  $\mathcal{F}_n = \{f \mid \text{ar}(f) = n\}$ . Pentru cazul particular  $n = 0$ ,  $\mathcal{F}_0$  reprezintă multimea simbolurilor constante (simboluri funcționale de aritate 0).

# Capitolul 3

## Sintaxa logicii de ordinul I

În acest capitol vom prezenta sintaxa formulelor din logica cu predicate de ordinul I. Pentru logica de ordinul I limbajul (mulțimea de siruri de simboluri) este determinat de alegerea unei signaturi  $\Sigma$ . Practic, există mai multe limbaje de ordinul I, câte un limbaj pentru fiecare signatură  $\Sigma$ .

În continuare, vom presupune fixată o signatură  $\Sigma$  cu simboluri predicative  $\mathcal{P}$  și simboluri funcționale  $\mathcal{F}$ .

### 3.1 Alfabetul

Ca și formulele din logica propozițională, formulele din logica de ordinul I sunt siruri de simboluri peste un anumit alfabet. Spre deosebire de logica propozițională, alfabetul este mai bogat și conține următoarele simboluri:

1. *conectori logici* deja cunoscuți:  $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \perp$ , precum și doi *cuantificatori*:  $\forall, \exists$ ;
2. *variabile*: vom fixa o mulțime infinit numărabilă de variabile, notată  $\mathcal{X} = \{\textcolor{red}{x}, \textcolor{blue}{y}, \textcolor{brown}{z}, \textcolor{red}{x}', \textcolor{blue}{y}', \textcolor{brown}{x}_1, \textcolor{red}{z}'', \dots\}$  (a nu se confunda cu mulțimea variabilelor propoziționale din logica propozițională; sunt două noțiuni fundamentale diferite și din acest motiv utilizăm altă culoare pentru a le reprezenta);
3. simboluri auxiliare:  $(, )$ ,  $\ldots$ ,  $(, )$ , și  $\therefore$ ;
4. simboluri suplimentare, care sunt specifice fiecărei signaturi  $\Sigma = (\mathcal{P}, \mathcal{F})$  în parte: simbolurile funcționale din mulțimea  $\mathcal{F}$  și respectiv simbolurile predicative din mulțimea  $\mathcal{P}$ .

### 3.2 Termen

**Definiția 5.** *Mulțimea termenilor,  $\mathcal{T}$ , este cea mai mică mulțime care satisface următoarele proprietăți:*

1.  $\mathcal{F}_0 \subseteq \mathcal{T}$  (orice simbol constant este termen);
2.  $\mathcal{X} \subseteq \mathcal{T}$  (orice variabilă este termen);
3. dacă  $f \in \mathcal{F}_n$  (cu  $n > 0$ ) și  $t_1, \dots, t_n \in \mathcal{T}$ , atunci  $f(t_1, \dots, t_n) \in \mathcal{T}$  (un simbol funcțional de aritate  $n$  aplicat unui număr de exact  $n$  termeni este termen).

**Observație.** Deoarece definiția mulțimii termenilor depinde de  $\Sigma = (\mathcal{P}, \mathcal{F})$ , elementele mulțimii  $\mathcal{T}$  se mai numesc  $\Sigma$ -termeni.

Practic, termenii se construiesc aplicând simboluri funcționale peste simboluri constante și variabile.

**Exemplul 6.** Fie signatura  $\Sigma = (\{\text{P, Q}\}, \{\text{f, i, a, b}\})$  definită în Exemplul 4, unde  $ar(\text{P}) = ar(\text{Q}) = 2$ ,  $ar(\text{f}) = 2$ ,  $ar(\text{i}) = 1$ ,  $ar(\text{a}) = ar(\text{b}) = 0$ . Iată câteva exemple de termeni:  $\text{a, b, x, y, } x_1, y', i(\text{a}), i(\text{x}), i(i(\text{a})), i(i(\text{x})), f(\text{a, b}), i(f(\text{a, b})), f(f(\text{x, a}), f(\text{y, y}))$ .

**Exercițiu 7.** Identificați în lista de mai jos  $\Sigma$ -termenii:

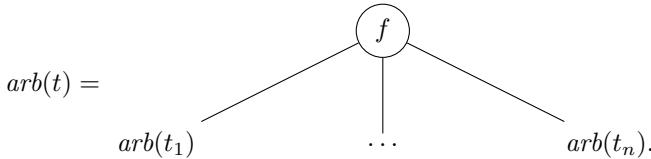
1.  $i(i(x))$ ;
2.  $i$ ;
3.  $f(x, x)$ ;
4.  $P(a, b)$ ;
5.  $i(a, a)$ ;
6.  $f(i(x), i(x))$ ;
7.  $f(i(x, x))$ ;
8.  $a(i(x))$ .

Termenii (sau, în mod echivalent, *termii*), sunt notații cu  $t, s, t_1, t_2, s_1, t'$ , etc. Deși termenii sunt scriși în mod ușual ca un sir de simboluri, ei au asociat un arbore de sintaxă abstractă definit după cum urmează:

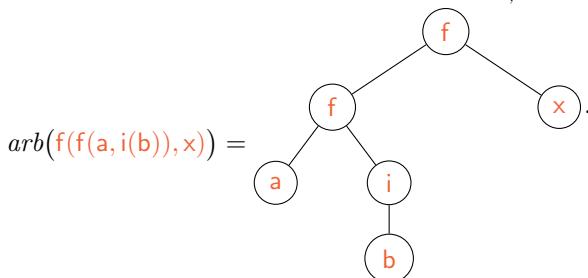
1. dacă  $t = c$  și  $c \in \mathcal{F}_0$ , atunci  $arb(t) = \bigcirc(c)$

2. dacă  $t = x$  și  $x \in \mathcal{X}$ , atunci  $\text{arb}(t) = \boxed{x}$

3. dacă  $t = f(t_1, \dots, t_n)$  și  $f \in \mathcal{F}_n$  ( $n > 0$ ),  $t_1, \dots, t_n \in \mathcal{T}$ , atunci



**Observație.** Deși formal termenii sunt definiți ca fiind siruri de simboluri peste alfabetul descris mai sus, aceștia trebuie înțeleși ca fiind arbori. De altfel, în orice software care lucrează cu termeni, această sunt memorate sub formă de arbori cu rădăcină. Iată arboarele atașat termenului  $f(f(a, i(b)), x)$ :



**Exercițiul 8.** Calculați arborii de sintaxă pentru termenii din Exemplul 6.

### 3.3 Formule atomice

**Definiția 9** (Formulă atomică). O formulă atomică este orice sir de simboluri de forma  $P(t_1, \dots, t_n)$ , unde  $P \in \mathcal{P}_n$  este un simbol predicativ de aritate  $n \geq 0$ , iar  $t_1, \dots, t_n \in \mathcal{T}$  sunt termeni. Dacă  $n = 0$ , scriem  $P$  în loc de  $P()$ .

**Exemplul 10.** Continuând Exemplul 6, folosim signatura

$$\Sigma = (\{P, Q\}, \{f, i, a, b\}),$$

unde  $\text{ar}(P) = \text{ar}(Q) = 2$ ,  $\text{ar}(f) = 2$ ,  $\text{ar}(i) = 1$ ,  $\text{ar}(a) = \text{ar}(b) = 0$ .

Iată câteva exemple de formule atomice:  $P(a, b)$ ,  $P(x, y)$ ,  $Q(i(i(x)), f(x, x))$ ,  $Q(a, b)$ ,  $P(f(f(a, i(x)), b), i(x))$ .

**Exercițiul 11.** Explicați de ce  $P(a)$ ,  $P$ ,  $i(i(x))$  nu sunt formule atomice peste signatura din Exemplul 10.

### 3.4 Formule de ordinul I

**Definiția 12** (Formule de ordinul I). *Mulțimea formulelor de ordinul I, notată  $\mathbb{LPI}$ , este cea mai mică mulțime astfel încât:*

1. (cazul de bază) orice formulă atomică este formulă (adică  $P(t_1, \dots, t_n) \in \mathbb{LPI}$  pentru orice simbol predicativ  $P \in \mathcal{P}_n$  și orice termeni  $t_1, \dots, t_n \in \mathcal{T}$ );
2. (cazurile inductive) pentru orice formule  $\varphi, \varphi_1, \varphi_2 \in \mathbb{LPI}$ , pentru orice variabilă  $x \in \mathcal{X}$ , avem că:
  - (a)  $\neg\varphi_1 \in \mathbb{LPI}$ ;
  - (b)  $(\varphi_1 \wedge \varphi_2) \in \mathbb{LPI}$ ;
  - (c)  $(\varphi_1 \vee \varphi_2) \in \mathbb{LPI}$ ;
  - (d)  $(\varphi_1 \rightarrow \varphi_2) \in \mathbb{LPI}$ ;
  - (e)  $(\varphi_1 \leftrightarrow \varphi_2) \in \mathbb{LPI}$ ;
  - (f)  $(\forall x.\varphi) \in \mathbb{LPI}$ ;
  - (g)  $(\exists x.\varphi) \in \mathbb{LPI}$ .

**Observație.** În Definiția 12, regăsim conectorii logici  $\neg, \wedge, \vee, \rightarrow$  și respectiv  $\leftrightarrow$  din logica propositională. Locul variabilelor propoziționale (deocamdată, la nivel sintactic) este luat de simbolurile predicative de aritate 0. Construcțiile  $(\forall x.\varphi)$  și  $(\exists x.\varphi)$  sunt noi.

**Exemplul 13.** Continuând Exemplul 6, folosim signatura  $\Sigma = (\{\text{P}, \text{Q}\}, \{\text{f}, \text{i}, \text{a}, \text{b}\})$ , unde  $\text{ar}(\text{P}) = \text{ar}(\text{Q}) = 2$ ,  $\text{ar}(\text{f}) = 2$ ,  $\text{ar}(\text{i}) = 1$  și  $\text{ar}(\text{a}) = \text{ar}(\text{b}) = 0$ .

Iată câteva exemple de formule din logica de ordinul I:

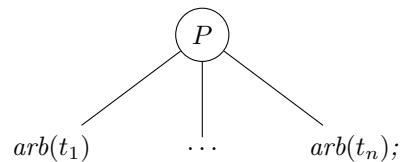
1.  $\text{P(a,b)}$ ;
2.  $\text{Q(a,b)}$ ;
3.  $\text{P(a,x)}$ ;
4.  $\neg\text{P(a,b)}$ ;
5.  $(\text{P(a,b)} \wedge \neg\text{Q(a,b)})$ ;
6.  $(\text{P(a,b)} \vee \neg\text{Q(x,y)})$ ;
7.  $(\text{P(a,b)} \rightarrow \text{P(a,b)})$ ;
8.  $((\text{P(a,b)} \rightarrow \text{P(a,b)}) \leftrightarrow (\text{P(a,b)} \rightarrow \text{P(a,b)}))$ ;

9.  $(\forall x.P(a, x))$ ;

10.  $(\exists x.\neg Q(x, y))$ .

**Definiția 14** (Arborele de sintaxă abstractă asociat formulelor din  $\mathbb{LPI}$ ).  
Formulele au asociat un arbore de sintaxă abstractă definit în cele ce urmează:

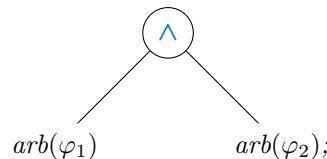
1. dacă  $\varphi = P(t_1, \dots, t_n)$ , atunci  $\text{arb}(\varphi) =$



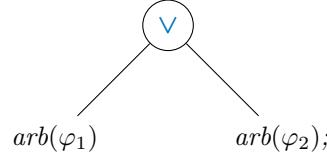
2. dacă  $\varphi = \neg\varphi_1$ , atunci  $\text{arb}(\varphi) =$



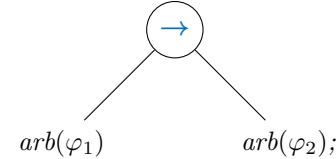
3. dacă  $\varphi = (\varphi_1 \wedge \varphi_2)$ , atunci  $\text{arb}(\varphi) =$



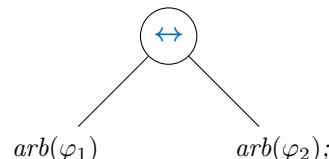
4. dacă  $\varphi = (\varphi_1 \vee \varphi_2)$ , atunci  $\text{arb}(\varphi) =$



5. dacă  $\varphi = (\varphi_1 \rightarrow \varphi_2)$ , atunci  $\text{arb}(\varphi) =$



6. dacă  $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$ , atunci  $\text{arb}(\varphi) =$





7. dacă  $\varphi = (\forall x.\varphi_1)$ , atunci  $arb(\varphi) =$

$arb(\varphi_1);$



8. dacă  $\varphi = (\exists x.\varphi_1)$ , atunci  $arb(\varphi) =$

$arb(\varphi_1).$

**Exercițiul 15.** Calculați arborele de sintaxă asociat formulelor din Exemplul 13.

### 3.5 Paranteze

Parantezele ( și ) sunt folosite pentru a marca ordinea efectuării operațiilor logice (și, sau, not, etc.) În continuare, vom renunța la parantezele care nu sunt necesare, la fel ca în cazul logicii propoziționale: dacă o formulă poate fi interpretată ca un arbore de sintaxă abstractă în două sau mai multe moduri, se vor folosi paranteze pentru a stabili arborele dorit.

De exemplu,  $\varphi_1 \vee \varphi_2 \wedge \varphi_3$  ar putea fi înțeleasă ca  $((\varphi_1 \vee \varphi_2) \wedge \varphi_3)$  sau ca  $(\varphi_1 \vee (\varphi_2 \wedge \varphi_3))$ . Pentru a evita scrierea formulelor cu prea multe paranteze, se stabilesc *priorități*. Ordinea priorității operatorilor este:

$\perp, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists,$

unde  $\neg$  este cel mai prioritar, iar  $\exists$  este cel mai puțin prioritar. Pentru a evita situațiile de ambiguitate (i.e., atunci când există mai multe moduri de a construi arborele de sintaxă), este recomandată folosirea de paranteze suplimentare.

Datorită priorității conectorilor logici, formula  $P(a, a) \vee P(b, b) \wedge P(a, b)$  va fi întotdeauna înțeleasă ca  $(P(a, a) \vee (P(b, b) \wedge P(a, b)))$  (deoarece  $\wedge$  este prioritar față de  $\vee$ ). Ca analogie, la fel se întâmplă și în cazul aritmeticiei:  $1 + 2 * 3$  va fi înțeles ca  $1 + (2 * 3)$ , deoarece  $*$  are prioritate în fața lui  $+$  ( $*$  este similar cu  $\wedge$  și  $+$  cu  $\vee$ ).

**Exercițiul 16.** Scrieți formulele din Exemplul 13 cu cât mai puține paranteze.

În Secțiunea 3.12 vom detalia modul în care cuantificatorii interacționează cu ceilalți conectori logici. Vom vedea că pentru cuantificatori avem niște reguli suplimentare în afară de priorități.

### 3.6 Modelarea în LPI a afirmațiilor din limba română

În această secțiune vom explica care este signatura folosită pentru a modela în logica de ordinul întâi afirmațiile: *Orice om este muritor, Socrate este om și respectiv Socrate este muritor.*

În primul rând, identificăm predicatele din text. Avem două predicate unare *este om* și respectiv *este muritor*. Alegem simbolul predicativ **O** pentru primul predicat și simbolul predicativ **M** pentru al doilea predicat. De asemenea, în text avem și o constantă: *Socrate*. Alegem simbolul funcțional **s** de aritate 0 pentru această constantă. Așadar, pentru a modela afirmațiile de mai sus, vom lucra cu signatura

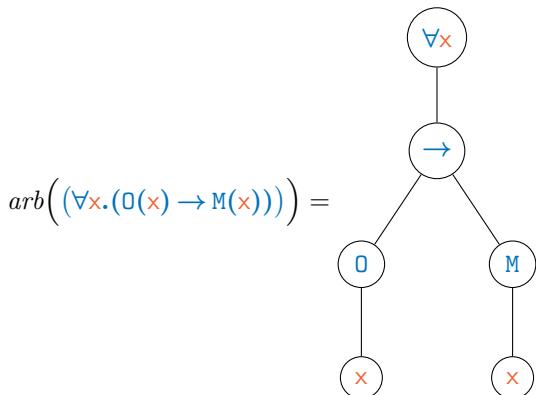
$$\Sigma = (\{O, M\}, \{s\}),$$

unde **O** și **M** sunt simboluri predicative de aritate  $ar(O) = ar(M) = 1$ , iar **s** este un simbol funcțional de aritate  $ar(s) = 0$ , adică un simbol constant.

Afirmația *orice om este muritor* va fi modelată prin formula de ordinul I

$$(\forall x.(O(x) \rightarrow M(x))),$$

al cărei arbore de sintaxă abstractă este:



Afirmația *Socrate este om* o vom modela prin formula atomică  $O(s)$ , iar afirmația *Socrate este muritor* o vom modela prin formula atomică  $M(s)$ .

Pentru signatura  $\Sigma = (\{O, M\}, \{s\})$  stabilită mai sus, există mai multe  $\Sigma$ -structuri posibile. Un exemplu este structura  $S = (D, \{O^S, M^S\}, \{s^S\})$  definită astfel:

1.  $D$  este mulțimea tuturor ființelor de pe Pământ;

2.  $\text{O}^S(x)$  este adevărat pentru orice ființă  $x$  care este și om;
3.  $\text{M}^S(x)$  este adevărat pentru orice ființă  $x$  (toate elementele domeniului sunt muritoare);
4.  $\text{s}^S$  este Socrate (Socrate, fiind o ființă, aparține multimii  $D$ ).

Anticipând puțin (vom discuta despre semantica formulelor de ordinul I în Capitolul 4), toate cele trei formule discutate în această secțiune, adică  $(\forall x.(\text{O}(x) \rightarrow \text{M}(x)))$ ,  $\text{O}(\text{s})$  și respectiv  $\text{M}(\text{s})$ , sunt adevărate în structura  $S$  definită mai sus. De fapt, calitatea raționamentului *orice om este muritor; Socrate este om; deci: Socrate este muritor* este dată de faptul că formula  $\text{M}(\text{s})$  este în mod necesar adevărată în *orice* structură în care formulele  $\text{O}(\text{s})$  și  $(\forall x.(\text{O}(x) \rightarrow \text{M}(x)))$  sunt adevărate, nu doar în structura  $S$  de mai sus.

### 3.7 Modelarea în $\mathbb{LPI}$ a afirmațiilor despre aritmetică

Fie signatura  $\Sigma = (\{\langle, =\}, \{+, -, 0, 1\})$ , unde  $\langle$  și  $=$  sunt simboluri predicative de aritate 2,  $+$  este simbol funcțional de aritate 2,  $-$  este simbol funcțional de aritate 1, iar  $0$  și  $1$  sunt simboluri constante. Iată câteva formule care fac parte din limbajul de ordinul I asociat signaturii  $\Sigma$ :

1.  $(\forall x.(\forall y.(\langle(x, y) \rightarrow \exists z.(\langle(x, z) \wedge \langle(z, y))))));$
2.  $(\forall x.(\forall y.(\exists z.(=(+(x, y), z))))));$
3.  $(\forall x.(\langle(0, x) \vee =(0, x))));$
4.  $(\forall x.(\exists y.=(x, -(y))))));$
5.  $=(+(x, y), z).$

De multe ori, în cazul simbolurilor predicative și simbolurilor funcționale binare, se folosește notația infixată (e.g.,  $x < y$  în loc de  $\langle(x, y)$ ). În acest caz, putem scrie formulele de mai sus în felul următor:

1.  $(\forall x.(\forall y.(x < y \rightarrow \exists z.(x < z \wedge z < y))))));$
2.  $(\forall x.(\forall y.(\exists z.(x + y = z))))));$
3.  $(\forall x.(0 < x \vee 0 = x));$

$$4. \left( \forall x. (\exists y. (x = -(y))) \right);$$

$$5. x + y = z.$$

Două dintre  $\Sigma$ -structurile posibile sunt  $S_1 = (\mathbb{R}, \{<, =\}, \{+, -, 0, 1\})$  și  $S_2 = (\mathbb{Z}, \{<, =\}, \{+, -, 0, 1\})$ , unde predicatele și funcțiile sunt cele cunoscute de la matematică (cu precizarea că  $-$  este funcția minus unar).

Anticipând cursul următor referitor la semantica formulelor de ordinul I, prima formulă este falsă în  $S_2$  și adevărată în  $S_1$ . A doua formulă și a patra formulă sunt adevărate atât în  $S_1$  cât și în  $S_2$ . A treia formula este falsă atât în  $S_1$  cât și în  $S_2$ . Valoarea de adevăr a celei de-a cincea formule nu depinde doar de structura în care evaluăm formula, ci și de valorile variabilelor  $x, y, z$ . Deoarece variabilele  $x, y, z$  nu apar cuantificate în formula numărul 5, acestea se numesc *libere*. Formula 5 este *satisfiabilă* atât în structura  $S_1$  cât și în structura  $S_2$ , deoarece în ambele cazuri există valori pentru variabilele  $x, y, z$  care să facă formula adevărată (e.g. valorile 1, 2, 3 pentru  $x, y$  și respectiv  $z$ ).

### 3.8 Variabilele unei formule

Cu  $vars(\varphi)$  notăm variabilele care apar în formula  $\varphi$ . De exemplu, vom avea că  $vars((\forall z. (P(x, y)))) = \{x, y, z\}$ . Definim funcția  $vars : \mathbb{LPI} \rightarrow 2^{\mathcal{X}}$  în cele ce urmează.

În primul rând, definim o funcție  $vars : \mathcal{T} \rightarrow 2^{\mathcal{X}}$  (atenție, domeniul este  $\mathcal{T}$ ) ca fiind funcția care asociază unui termen (din mulțimea  $\mathcal{T}$ ) mulțimea variabilelor care apar în acel termen. Toate definițiile care urmează vor fi definiții inductive, care oglindesc definițiile sintactice corespunzătoare. Le vom denumi simplu *definiții recursive*, fără a mai preciza explicit cazurile de bază sau pe cele inductive. Amintim că  $2^{\mathcal{X}}$  denotă mulțimea tuturor submulțimilor lui  $\mathcal{X}$ . Totodată, amintim că pentru o signatură fixată  $\Sigma$ , notăm cu  $\mathcal{P}_n$  mulțimea simbolurilor predicative de aritate  $n$  din  $\Sigma$ , iar cu  $\mathcal{F}_n$  mulțimea simbolurilor funcționale de aritate  $n$  din  $\Sigma$ .

**Definiția 17.** Funcția  $vars : \mathcal{T} \rightarrow 2^{\mathcal{X}}$  este definită recursiv după cum urmează:

1.  $vars(c) = \emptyset$ , dacă  $c \in \mathcal{F}_0$  este un simbol constant;
2.  $vars(x) = \{x\}$ , dacă  $x \in \mathcal{X}$  este o variabilă;
3.  $vars(f(t_1, \dots, t_n)) = \bigcup_{i \in \{1, \dots, n\}} vars(t_i)$ .

Putem acum defini (inductiv, prin imbricare) funcția extinsă și notată omonim,  $vars : \mathbb{LPI} \rightarrow 2^{\mathcal{X}}$ , care asociază unei formule din  $\mathbb{LPI}$  mulțimea de variabile ale formulei (adică, variabilele care apar în formulă):

**Definiția 18.** Funcția  $\text{vars} : \mathbb{LPI} \rightarrow 2^X$  este definită recursiv după cum urmează:

1.  $\text{vars}(P(t_1, \dots, t_n)) = \bigcup_{i \in \{1, \dots, n\}} \text{vars}(t_i);$
2.  $\text{vars}(\neg\varphi) = \text{vars}(\varphi);$
3.  $\text{vars}((\varphi_1 \wedge \varphi_2)) = \text{vars}(\varphi_1) \cup \text{vars}(\varphi_2);$
4.  $\text{vars}((\varphi_1 \vee \varphi_2)) = \text{vars}(\varphi_1) \cup \text{vars}(\varphi_2);$
5.  $\text{vars}((\varphi_1 \rightarrow \varphi_2)) = \text{vars}(\varphi_1) \cup \text{vars}(\varphi_2);$
6.  $\text{vars}((\varphi_1 \leftrightarrow \varphi_2)) = \text{vars}(\varphi_1) \cup \text{vars}(\varphi_2);$
7.  $\text{vars}((\forall x.\varphi)) = \text{vars}(\varphi) \cup \{x\};$
8.  $\text{vars}((\exists x.\varphi)) = \text{vars}(\varphi) \cup \{x\}.$

Să observăm că variabila  $x$  este adăugată corespunzător în mulțimea de variabile care se construiește chiar dacă ea apare doar imediat după simbolurile  $\exists$  sau  $\forall$ .

**Exemplul 19.** Fie formula  $\varphi$ :

$$\left( \left( \forall x. (P(x, y) \wedge \exists y. (P(z, f(x, y)) \wedge P(x, y))) \right) \wedge P(x, x) \right).$$

Avem  $\text{vars}(\varphi) = \{x, y, z\}$ .

**Exercițiul 20.** Fie semnatura  $\Sigma = (\{P, Q\}, \{f, i, a, b\})$ , unde  $\text{ar}(P) = \text{ar}(Q) = 2$ ,  $\text{ar}(f) = 2$ ,  $\text{ar}(i) = 1$ ,  $\text{ar}(a) = \text{ar}(b) = 0$ . Calculați  $\text{vars}(\varphi)$  pentru fiecare formulă  $\varphi$  de mai jos:

1.  $P(x, y);$
2.  $Q(a, b);$
3.  $P(a, x);$
4.  $\neg P(x, z);$
5.  $(P(x, x) \wedge \neg Q(x, z));$
6.  $(P(x, b) \vee \neg Q(z, y));$
7.  $((P(x, b) \rightarrow P(x, z)) \leftrightarrow (P(x, b) \rightarrow P(a, z)));$
8.  $(\forall x. P(a, x));$
9.  $(\exists x. \neg Q(x, y));$
10.  $\left( (\exists x. \neg Q(x, y)) \wedge (\forall y. P(y, x)) \right).$

### 3.9 Domeniul de vizibilitate al unui cuantificator - analogie cu limbajele de programare

Într-un limbaj de programare, putem declara mai multe variabile cu același nume. De exemplu, în C, putem avea următorul cod:

```
/* 1:*/ int f()
/* 2:*/
/* 3:*/ int s = 0;
/* 4:*/ for (int x = 1; x <= 10; ++x) {
/* 5:*/     for (int y = 1; y <= 10; ++y) {
/* 6:*/         s += x * y * z;
/* 7:*/         for (int x = 1; x <= 10; ++x) {
/* 8:*/             s += x * y * z;
/* 9:*/
/* 10:*/
/* 11:*/
/* 12:*/     return s;
/* 13:*/ }
```

În acest fragment de cod, sunt declarate trei variabile, două dintre variabile având același nume, și anume **x**. Domeniul de vizibilitate al variabilei **x** declarate la linia 4 este dat de liniile 4 – 11, iar domeniul de vizibilitate al variabilei **x** declarată la linia 7 este dat de liniile 7 – 9. Astfel, orice apariție a numelui **x** între liniile 7 – 9 se referă la cea de-a doua declarație a variabilei, în timp ce orice apariție a numelui **x** între liniile 4 – 11 (cu excepția liniilor 7 – 9) se referă la prima declarație a lui **x**. De exemplu, apariția lui **x** de la linia 6 se referă la variabila **x** declarată la linia 4. Apariția lui **x** de la linia 8 se referă la variabila **x** declarată la linia 7.

Liniile 4 – 11 reprezintă domeniul de vizibilitate al primei declarații a variabilei **x**, iar liniile 7 – 9 reprezintă domeniul de vizibilitate al celei de-a două declarații a variabilei **x**.

Un fenomen similar se întâmplă în formulele logicii de ordinul I. De exemplu, în formula  $(\forall x. (\forall y. (P(x, y) \wedge P(x, z) \wedge (\exists x. P(x, y))))$ , variabila **x** este cuantificată de două ori (prima dată universal, a doua oară existențial). O cuantificare a unei variabile se numește *legare* (engl. *binding*), din motive istorice. O *legare* este similară, din punctul de vedere al domeniului de vizibilitate, cu definirea unei variabile într-un limbaj de programare.

Astfel, domeniul de vizibilitate  $D_1$  al variabilei **x** cuantificate universal este  $(\forall y. (P(x, y) \wedge P(x, z) \wedge (\exists x. P(x, y))))$ , în timp ce domeniul de vizibilitate  $D_2$  al variabilei **x** cuantificate existențial este  $P(x, y)$ :

$$\left( \forall x. \underbrace{\left( \forall y. (P(x, y) \wedge P(x, z) \wedge (\exists x. \overbrace{P(x, y)}^{D_2})) \right)}_{D_1} \right)$$

Aparițiile unei variabile cuantificate care sunt prezente în domeniul de vizibilitate al acesteia se numesc *legate*, în timp ce aparițiile din afara domeniului de vizibilitate se numesc *libere*.

### 3.10 Apariții libere și legate ale variabilelor

În această secțiune vom stabili formal conceptul de apariție/variabilă legată și de apariție/variabilă liberă. Aparițiile libere/legate ale unei variabile în logica de ordinul I sunt, ca o analogie, similare cu variabilele globale/locale într-un limbaj de programare.

Mai departe vom utiliza noțiunea de *arbore de sintaxă abstractă* pentru formulele din logica de ordinul I.

**Definiția 21** (Apariție liberă). O apariție liberă a unei variabile  $x$  într-o formulă  $\varphi$  este dată de un nod în arborele formulei etichetat cu  $x$  și care are proprietatea că, mergând din nod înspre rădăcină, nu întâlnim niciun nod etichetat cu  $\forall x$  sau cu  $\exists x$ .

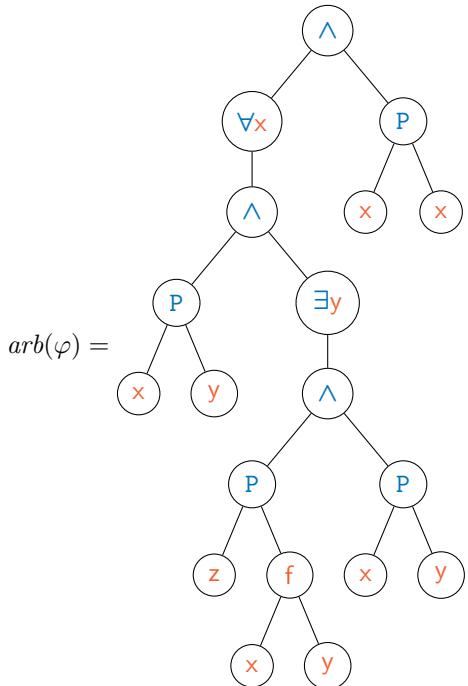
**Definiția 22** (Apariție legată). O apariție legată a unei variabile  $x$  într-o formulă  $\varphi$  este dată de un nod în arborele formulei etichetat cu  $x$  și care are proprietatea că, mergând din nod înspre rădăcină, întâlnim cără un nod etichetat cu  $\forall x$  sau cu  $\exists x$ .

Cel mai apropiat astfel de nod etichetat cu  $\forall x$  sau cu  $\exists x$  este cuantificarea care leagă apariția în cauză a variabilei  $x$ .

**Exemplul 23.** Considerăm în continuare formula

$$\varphi = \left( \left( \forall x. \left( P(x, y) \wedge \exists y. (P(z, f(x, y)) \wedge P(x, y)) \right) \right) \wedge P(x, x) \right).$$

Arborele de sintaxă abstractă al formulei  $\varphi$  este:



*În formula  $\varphi$  de mai sus, variabila  $x$  are două apariții libere. Variabila  $y$  are o apariție liberă. Variabila  $z$  are o apariție liberă. Toate aparițiile libere ale variabilelor în formula  $\varphi$  sunt marcate prin subliniere:*

$$\varphi = \left( \left( \forall \underline{x}. \left( P(\underline{x}, \underline{y}) \wedge \exists \underline{y}. \left( P(z, f(\underline{x}, \underline{y})) \wedge P(\underline{x}, \underline{y}) \right) \right) \right) \wedge P(\underline{x}, \underline{x}) \right).$$

*Toate aparițiile legate ale variabilelor în formula  $\varphi$  sunt marcate prin dublă subliniere în următoarea formulă:*

$$\varphi = \left( \left( \forall \underline{x}. \left( P(\underline{\underline{x}}, \underline{y}) \wedge \exists \underline{y}. \left( P(z, f(\underline{\underline{x}}, \underline{\underline{y}})) \wedge P(\underline{\underline{x}}, \underline{\underline{y}}) \right) \right) \right) \wedge P(\underline{\underline{x}}, \underline{\underline{x}}) \right).$$

**Observație.** *În restul capitolelor vom face o distincție clară între aparițiile (libere sau legate ale) variabilelor într-o formulă și multimiile variabilelor de acest tip (libere sau legate). Nodurile etichetate cu  $\forall x$  și respectiv  $\exists x$  nu vor fi considerate ca desemnând nici o apariție liberă, nici o apariție legată a lui  $x$ , ci ca fiind simple noduri prin care se fixează/denumește un cuantificator (sau, prin care se leagă variabila  $x$ ).*

### 3.11 Variabile libere și variabile legate

Mulțimea variabilelor unei formule  $\varphi$  care au cel puțin o apariție liberă se notează  $free(\varphi)$ .

**Definiția 24.** Funcția  $free : \mathbb{LPI} \rightarrow 2^X$  este definită recursiv în modul următor:

1.  $free(P(t_1, \dots, t_n)) = vars(t_1) \cup \dots \cup vars(t_n);$
2.  $free(\neg\varphi) = free(\varphi);$
3.  $free((\varphi_1 \wedge \varphi_2)) = free(\varphi_1) \cup free(\varphi_2);$
4.  $free((\varphi_1 \vee \varphi_2)) = free(\varphi_1) \cup free(\varphi_2);$
5.  $free((\varphi_1 \rightarrow \varphi_2)) = free(\varphi_1) \cup free(\varphi_2);$
6.  $free((\varphi_1 \leftrightarrow \varphi_2)) = free(\varphi_1) \cup free(\varphi_2);$
7.  $free((\forall x.\varphi)) = free(\varphi) \setminus \{x\};$
8.  $free((\exists x.\varphi)) = free(\varphi) \setminus \{x\}.$

**Exemplul 25.** Pentru formula

$$\varphi = \left( \left( \forall x. \left( P(x, y) \wedge \exists y. (P(z, f(x, y)) \wedge P(x, y)) \right) \right) \wedge P(x, x) \right).$$

avem că  $free(\varphi) = \{x, y, z\}$ .

**Exercițiul 26.** Calculați  $free(\varphi)$  pentru fiecare formulă  $\varphi$  din Exercițiul 20.

Cu  $bound(\varphi)$  notăm mulțimea variabilelor legate într-o formulă, cu alte cuvinte mulțimea acelor variabile  $x$  cu proprietatea că există în formulă cel puțin un nod etichetat cu  $\forall x$  sau cu  $\exists x$ .

**Definiția 27.** Funcția  $bound : \mathbb{LPI} \rightarrow 2^X$  este definită recursiv astfel:

1.  $bound(P(t_1, \dots, t_n)) = \emptyset;$
2.  $bound(\neg\varphi) = bound(\varphi);$
3.  $bound((\varphi_1 \wedge \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2);$
4.  $bound((\varphi_1 \vee \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2);$
5.  $bound((\varphi_1 \rightarrow \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2);$
6.  $bound((\varphi_1 \leftrightarrow \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2);$

7.  $\text{bound}((\forall x.\varphi)) = \text{bound}(\varphi) \cup \{x\};$
8.  $\text{bound}((\exists x.\varphi)) = \text{bound}(\varphi) \cup \{x\}.$

**Exercițiul 28.** Calculați  $\text{bound}(\varphi)$  pentru fiecare formulă  $\varphi$  din Exercițiul 20.

**Exercițiul 29.** Calculați  $\text{bound}(\varphi)$ , unde

$$\varphi = \left( \left( \forall \textcolor{red}{x}. \left( \textcolor{blue}{P}(x, y) \wedge \exists \textcolor{blue}{y}. \left( \textcolor{blue}{P}(z, f(x, y)) \wedge P(x, y) \right) \right) \wedge P(x, x) \right).$$

**Definiția 30.** Variabilele legate ale unei formule  $\varphi$  sunt elementele multimii  $\text{bound}(\varphi)$ .

**Definiția 31.** Variabilele libere ale unei formule  $\varphi$  sunt elementele multimii  $\text{free}(\varphi)$ .

**Observație.** Multimile  $\text{free}(\varphi)$  și  $\text{bound}(\varphi)$  pot avea elemente în comun.

**Observație.** O variabilă poate avea mai multe apariții într-o formulă.

După cum am precizat anterior, trebuie făcută diferența între o apariție liberă a unei variabile într-o formulă și o variabilă liberă a unei formule. Apariția liberă este indicată printr-un nod din arborele formulei, în timp ce variabila liberă este un element al multimii  $\mathcal{X}$ .

Similar, trebuie făcută diferența între o apariție legată a unei variabile într-o formulă și o variabilă legată a unei formule. Apariția legată este indicată de un nod în arborele formulei, în timp ce variabila este un element al multimii  $\mathcal{X}$ .

### 3.12 Domeniul de vizibilitate și parantetizarea formulelor

Acum că am înțeles ce este domeniul de vizibilitate a unei variabile legate (apelând și la arborele formulei), putem clarifica un aspect referitor la ordinea de prioritate a conectorilor logici (dacă privim formula doar ca text/cuvânt). Am stabilit deja că ordinea de prioritate este  $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists$ , dar cuantificatorii  $\forall$  și  $\exists$  interacționează într-un mod mai subtil cu ceilalți conectori logici. Mai precis, textual, o formulă fără paranteze se (re)parantetizează astfel încât domeniul de vizibilitate al fiecărui cuantificator să se extindă cât mai mult spre dreapta.

De exemplu, formula:

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

se parantezează în felul următor:

$$\left( \forall x. (P(x, x) \vee \neg(\exists y. (P(x, y) \wedge P(y, x)))) \right).$$

### 3.13 Fișă de exerciții

**Exercițiul 32.** Identificați o structură și o semnificație potrivite pentru afirmațiile de mai jos și apoi modelați aceste afirmații ca formule în logica de ordinul I:

Ion este student. Orice student învăță la Logică. Oricine învăță la Logică trece examenul. Orice student este om. Există un om care nu a trecut examenul. Deci nu toți oamenii sunt studenți.

**Exercițiul 33.** Fie structura  $S = (\mathbb{R}, \{Nat, Int, Prim, Par, >\}, \{+, 0, 1, 2\})$ , unde  $Nat, Int, Prim, Par$  sunt predicate unare cu următoarea semnificație:

- $Nat(u) = u$  este număr natural;
- $Int(u) = u$  este număr întreg;
- $Prim(u) = u$  este număr prim;
- $Par(u) = u$  este număr par.

Predicatul binar  $>$  este relația “mai mare” peste numere reale. Funcția  $+$  este funcția de adunare a numerelor reale. Constantele 0, 1, 2 sunt chiar numerele 0, 1, 2.

1. Propuneți o semnificație  $\Sigma$  pentru structura  $S$  de mai sus.
2. Modelați următoarele afirmații ca formule de ordinul I în semnificație asociată structurii  $S$  de mai sus:
  - (a) Orice număr natural este și număr întreg.
  - (b) Suma oricărora două numere naturale este număr natural.
  - (c) Oricum am alege un număr natural, există un număr prim care este mai mare decât numărul respectiv.
  - (d) Dacă orice număr natural este număr prim, atunci zero este număr prim.
  - (e) Oricum am alege un număr prim, există un număr prim mai mare decât el.
  - (f) Suma a două numere pare este un număr par.

- (g) Orice număr prim mai mare decât 2 este impar.
- (h) Orice număr prim poate fi scris ca suma a patru numere prime.
- (i) Suma a două numere pare este un număr impar.

**Exercițiu 34.** Dați exemplu de 5 termeni peste signaturile de la Exercițiu 33 și calculați arborele de sintaxă abstractă al acestor termeni.

**Exercițiu 35.** Dați exemplu de 5 formule peste signatura de la Exercițiu 33 și calculați arborele de sintaxă abstractă al acestora.

**Exercițiu 36.** Calculați arborele de sintaxă abstractă al următoarelor formule (indicatie: puneti paranteze în jurul formulelor, în ordinea de prioritate a conectorilor):

1.  $P(x) \vee P(y) \wedge \neg P(z);$
2.  $\neg \neg P(x) \vee P(y) \rightarrow P(x) \wedge \neg P(z);$
3.  $\forall x. \forall y. \neg \neg P(x) \vee P(y) \rightarrow P(x) \wedge \neg P(z);$
4.  $\forall x. \forall y. \neg \neg P(x) \vee P(y) \rightarrow \exists x. P(x) \wedge \neg P(x);$
5.  $\forall x'. \neg \forall x. P(x) \wedge \exists y. Q(x, y) \vee \neg Q(z, z) \rightarrow \exists z'. P(z').$

**Exercițiu 37.** Marcați aparițiile libere și respectiv aparițiile legate ale variabilelor în formulele de mai jos:

1.  $\varphi_1 = (\forall x. P(x, x) \wedge P(x, y)) \wedge P(x, z);$
2.  $\varphi_2 = (\forall x. P(f(x, x), i(x)) \wedge \exists y. (P(x, y) \wedge P(x, z))).$

**Exercițiu 38.** Identificați domeniul de vizibilitate, pentru fiecare cuantificator, în formulele  $\varphi_1$  și  $\varphi_2$  date în Exercițiu 37.

**Exercițiu 39.** Găsiți variabilele, variabilele libere și respectiv variabilele legate ale formulelor  $\varphi_1$  și  $\varphi_2$  date în Exercițiu 37.

**Exercițiu 40.** Fie  $A = \{p, q, r, \dots\}$  mulțimea variabilelor propoziționale. Vom considera signatura  $\Sigma_{\mathbb{LP}} = (A, \emptyset)$ , unde variabilele propoziționale din  $A$  sunt simboluri predicative de aritate 0.

1. Demonstrați că pentru orice formulă  $\varphi \in \mathbb{LP}$  avem  $\varphi \in \mathbb{LPI}$  (peste signatura  $\Sigma_{\mathbb{LP}}$ ).
2. Demonstrați că pentru orice formulă fără cuantificatori  $\varphi \in \mathbb{LPI}$  peste signatura  $\Sigma_{\mathbb{LP}}$ , avem că  $\varphi \in \mathbb{LP}$ .



## Capitolul 4

# Semantica formulelor logicii de ordinul I

Sintaxa logicii de ordinul I explică care sunt, din punct de vedere sintactic, formulele logicii de ordinul I. Semantica logicii de ordinul I se referă la *înțelesul* formulelor. Semantica unei formule (sau *înțelesul* formulei) va fi o valoare de adevăr. Ca și la logica propozițională, în general, valoarea de adevăr a unei formule depinde nu doar de formulă, ci și de *structura* în care evaluăm formula.

Reamintim că o *signatură*  $\Sigma = (\mathcal{P}, \mathcal{F})$  este o pereche formată dintr-o mulțime de simboluri predicative  $\mathcal{P}$  și o mulțime de simboluri funcționale  $\mathcal{F}$ . Fiecare simbol are atașat un număr natural numit aritatea simbolului.

În acest capitol vom utiliza *signatura*  $\Sigma = (\{\text{P}\}, \{\text{f}, \text{i}, \text{e}\})$ , unde  $\text{P}$  este simbol predicativ de aritate 2, iar  $\text{f}$ ,  $\text{i}$  și  $\text{e}$  sunt simboluri funcționale de aritate 2, 1 și respectiv 0. Altfel spus,  $\mathcal{P}_2 = \{\text{P}\}$ ,  $\mathcal{P}_1 = \emptyset$ ,  $\mathcal{P}_0 = \emptyset$ ,  $\mathcal{F}_2 = \{\text{f}\}$ ,  $\mathcal{F}_1 = \{\text{i}\}$  și  $\mathcal{F}_0 = \{\text{e}\}$ .

Reamintim și că, dacă  $\Sigma = (\mathcal{P}, \mathcal{F})$  este o *signatură*, prin  $\Sigma$ -*structură* înțelegem orice tuplu  $S = (D, \text{Pred}, \text{Fun})$  cu proprietatea că:

1.  $D$  este o mulțime nevidă numită domeniul structurii  $S$ ;
2. pentru fiecare simbol predicativ  $P \in \mathcal{P}$  există un predicat  $P^S \in \text{Pred}$  de aritate corespunzătoare;
3. pentru fiecare simbol funcțional  $f \in \mathcal{F}$  există o funcție  $f^S \in \text{Fun}$  de aritate corespunzătoare.

**Exemplul 41.** Mai jos avem câteva exemple de  $\Sigma$ -*structuri*:

1.  $S_1 = (\mathbb{Z}, \{=\}, \{+, -, 0\})$ ;

2.  $S_2 = (\mathbb{R}^*, \{=\}, \{\times, \cdot^{-1}, 1\});$
3.  $S_3 = (\mathbb{N}, \{=\}, \{+, s, 0\});$
4.  $S_4 = (\mathbb{N}, \{<\}, \{+, s, 0\});$
5.  $S_5 = (\mathbb{Z}, \{<\}, \{+, -, 0\}).$

Structura  $S_1$  are domeniul  $\mathbb{Z}$  (multimea numerelor întregi), predicatul asociat simbolului predicativ **P este** = (predicatul de egalitate pentru numere întregi), funcția + este funcția de adunare a numerelor întregi asociată simbolului funcțional **f**, – este funcția minus unar asociată simbolului funcțional **i**, iar simbolul constant **e** are asociată constanta 0.

Structura  $S_2$  are domeniul  $\mathbb{R}^*$  (multimea numerelor reale strict pozitive), predicatul asociat simbolului predicativ **P este** = (predicatul de egalitate pentru numere reale pozitive), funcția  $\times$  este funcția de înmulțire a numerelor reale asociată simbolului funcțional **f**,  $\cdot^{-1}$  este funcția unară asociată simbolului funcțional **i** care calculează inversul unui număr real (e.g.  $5^{-1} = \frac{1}{5}$ , iar  $\frac{1}{10}^{-1} = 10$ ), iar simbolul constant **e** are asociată constanta 1.

Structura  $S_3$  are domeniul  $\mathbb{N}$  (multimea numerelor naturale), predicatul asociat simbolului predicativ **P este** = (predicatul de egalitate pentru numere naturale), funcția + este funcția de adunare a numerelor naturale asociată simbolului funcțional **f**, s este funcția succesor (care asociază unui număr natural următorul număr natural – e.g.,  $s(7) = 8$ ) asociată simbolului funcțional **i**, iar simbolul constant **e** are asociată constanta 0.

Structura  $S_4$  are domeniul  $\mathbb{N}$  (multimea numerelor naturale), predicatul asociat simbolului predicativ **P este** < (relația mai mic peste numere naturale), funcția + este funcția de adunare a numerelor naturale asociată simbolului funcțional **f**, s este funcția succesor (care asociază unui număr natural următorul număr natural – e.g.,  $s(7) = 8$ ) asociată simbolului funcțional **i**, iar simbolul constant **e** are asociată constanta 0.

Structura  $S_5$  este similară cu  $S_1$ , doar că simbolul predicativ **P** are asociată relația mai mic în loc de egal.

Folosind notatiile de mai sus, avem că  $\text{P}^{S_4} = <$ ,  $\text{f}^{S_2} = \times$ , iar  $\text{e}^{S_1} = 0$ .

## 4.1 Atribuiriri

Asemănător cu logica propozițională, pentru a obține valoarea de adevar a unei formule într-o structură, trebuie să pornim cu fixarea unor valori concrete pentru simbolurile sintactice din alfabetul peste care este construită formula. În cazul de față, începem cu variabilele.

**Definiția 42** (Atribuire). Fie  $\Sigma$  o signatură și  $S$  o  $\Sigma$ -structură cu domeniul  $D$ . Se numește  $S$ -atribuire orice funcție

$$\alpha : \mathcal{X} \rightarrow D.$$

**Exemplul 43.** Funcția  $\alpha_1 : \mathcal{X} \rightarrow \mathbb{Z}$ , definită ca mai jos, este o  $S_1$ -atribuire:

1.  $\alpha_1(\textcolor{red}{x}_1) = 5$ ;
2.  $\alpha_1(\textcolor{red}{x}_2) = 5$ ;
3.  $\alpha_1(\textcolor{red}{x}_3) = 6$ ;
4.  $\alpha_1(x) = 0$  pentru orice  $x \in \mathcal{X} \setminus \{\textcolor{red}{x}_1, \textcolor{red}{x}_2, \textcolor{red}{x}_3\}$ .

**Exemplul 44.** Funcția  $\alpha_2 : \mathcal{X} \rightarrow \mathbb{Z}$ , definită ca mai jos, este o  $S_1$ -atribuire:

1.  $\alpha_2(\textcolor{red}{x}_1) = 6$ ;
2.  $\alpha_2(\textcolor{red}{x}_2) = 5$ ;
3.  $\alpha_2(\textcolor{red}{x}_3) = 6$ ;
4.  $\alpha_2(x) = 0$  pentru orice  $x \in \mathcal{X} \setminus \{\textcolor{red}{x}_1, \textcolor{red}{x}_2, \textcolor{red}{x}_3\}$ .

**Observație.** Atenție! A nu se confunda noțiunea de atribuire din Logica de Ordinul I cu noțiunea de atribuire de valori de adevăr din Logica Propozițională.

Acum, având la dispoziție o atribuire  $\alpha$ , putem calcula valoarea unui termen într-o asemenea atribuire. Pentru aceasta, vom folosi de fapt extensia lui  $\alpha$ , notată  $\overline{\alpha}$ ,

$$\overline{\alpha} : \mathcal{T} \rightarrow D,$$

dataă în definiția care urmează.

**Definiția 45** (Valoarea unui termen într-o atribuire). Dându-se o  $S$ -atribuire  $\alpha$  și un termen  $t \in \mathcal{T}$  peste signatura  $\Sigma$ , valoarea termenului  $t$  în atribuirea  $\alpha$  este un element al domeniului  $D$  notat cu  $\overline{\alpha}(t)$  și calculat recursiv astfel:

1.  $\overline{\alpha}(c) = c^S$  dacă  $c \in \mathcal{F}_0$  (i.e.,  $c$  este un simbol constant);
2.  $\overline{\alpha}(x) = \alpha(x)$  dacă  $x \in \mathcal{X}$  (i.e.,  $x$  este o variabilă);
3.  $\overline{\alpha}(f(t_1, \dots, t_n)) = f^S(\overline{\alpha}(t_1), \dots, \overline{\alpha}(t_n))$  dacă  $f \in \mathcal{F}_n$  este un simbol funcțional de aritate  $n$ , iar  $t_1, \dots, t_n$  sunt termeni.

**Exemplul 46.** Continuând Exemplul 43, unde  $\alpha_1$  este o  $S_1$ -atribuire, avem:

$$\begin{aligned}
 \overline{\alpha_1}(\mathbf{f}(\mathbf{i}(x_1), e)) &= \overline{\alpha_1}(\mathbf{i}(x_1)) + \overline{\alpha_1}(e) \\
 &= -(\overline{\alpha_1}(x_1)) + e^{S_1} \\
 &= -(\alpha_1(x_1)) + 0 \\
 &= -5 + 0 \\
 &= -5.
 \end{aligned}$$

Așadar, valoarea termenului  $\mathbf{f}(\mathbf{i}(x_1), e)$  în atribuirea  $\alpha_1$  este  $-5$ .

**Observație.** Intuiția este că atribuirile asociază variabilelor libere dintr-o formulă valori din domeniul structurii.

În punctul în care o variabilă este legată în formulă, valoarea variabilei este umbrită (engl. shadows) de o nouă valoare în domeniul de vizibilitate al variabilei legate. Această schimbare a valorii se face prin intermediul operației de actualizare a unei atribuiriri, definită mai jos.

**Definiția 47** (Actualizarea unei atribuiriri). Dându-se o atribuire  $\alpha$ , o variabilă  $x \in \mathcal{X}$  și un element  $u \in D$ , notăm cu  $\alpha[x \mapsto u]$  o nouă atribuire, care coincide cu  $\alpha$ , exceptând valoarea variabilei  $x$ , care devine acum  $u$ :

$$\alpha[x \mapsto u] : \mathcal{X} \rightarrow D, \text{ a.i.}$$

1.  $(\alpha[x \mapsto u])(x) = u;$
2.  $(\alpha[x \mapsto u])(y) = \alpha(y)$ , pentru orice  $y \in \mathcal{X} \setminus \{x\}$ .

**Exemplul 48.** De exemplu, atribuirea  $\alpha_1[x_1 \mapsto 6]$  este exact atribuirea  $\alpha_2$  definită în exemplele de mai sus. Valoarea termenului  $\mathbf{f}(\mathbf{i}(x_1), e)$  în atribuirea  $\alpha_1[x_1 \mapsto 6]$ , notată cu  $\alpha_1[x_1 \mapsto 6](\mathbf{f}(\mathbf{i}(x_1), e))$ , este  $-6$ .

**Exercițiul 49.** Calculați valorile de mai jos:

1.  $\overline{\alpha_1[x_1 \mapsto 10]}(\mathbf{f}(\mathbf{i}(x_1), e));$
2.  $\overline{\alpha_1[x_2 \mapsto 10]}(\mathbf{f}(\mathbf{i}(x_1), e));$
3.  $\overline{\alpha_1[x_2 \mapsto 10][x_1 \mapsto 10]}(\mathbf{f}(\mathbf{i}(x_1), e)).$

## 4.2 Valoarea de adevăr a unei formule de ordinul I

În acest moment avem ingredientele pentru a defini formal valoarea de adevăr a unei formule de ordinul I, construită peste o signură  $\Sigma$ . Această valoare se poate calcula doar într-o  $\Sigma$ -structură  $S$ , cu ajutorul unei  $S$ -atribuiriri  $\alpha$ .

Notățiile folosite sunt similare cu cele pentru logica propozițională. Astfel, notăm faptul că o formulă  $\varphi$  este adevărată într-o structură  $S$  cu o atribuire  $\alpha$  prin  $S, \alpha \models \varphi$ . Faptul că o formulă  $\varphi$  nu este adevărată într-o structură  $S$  cu o atribuire  $\alpha$  se notează cu  $S, \alpha \not\models \varphi$ .

Notăția  $S, \alpha \models \varphi$  se mai citește *S satisfacă  $\varphi$  cu atribuirea  $\alpha$* , iar  $S, \alpha \not\models \varphi$  se mai citește *S nu satisfacă  $\varphi$  cu atribuirea  $\alpha$* .

**Definiția 50.** *Faptul că o structură  $S$  satisfacă o formulă  $\varphi$  cu o anumită atribuire  $\alpha$  (echivalent,  $\varphi$  este adevărată în structura  $S$  cu atribuirea  $\alpha$ ) se definește inductiv astfel (prima linie din enumerarea care urmează desemnează cazul de bază, restul reprezentând cazurile inductive):*

1.  $S, \alpha \models P(t_1, \dots, t_n)$  dacă  $P^S(\bar{\alpha}(t_1), \dots, \bar{\alpha}(t_n))$ ;
2.  $S, \alpha \models \neg\varphi$  dacă  $S, \alpha \not\models \varphi$ ;
3.  $S, \alpha \models (\varphi_1 \wedge \varphi_2)$  dacă  $S, \alpha \models \varphi_1$  și  $S, \alpha \models \varphi_2$ ;
4.  $S, \alpha \models (\varphi_1 \vee \varphi_2)$  dacă  $S, \alpha \models \varphi_1$  sau  $S, \alpha \models \varphi_2$ ;
5.  $S, \alpha \models (\varphi_1 \rightarrow \varphi_2)$  dacă  $S, \alpha \not\models \varphi_1$  sau  $S, \alpha \models \varphi_2$ ;
6.  $S, \alpha \models (\varphi_1 \leftrightarrow \varphi_2)$  dacă (1) atât  $S, \alpha \models \varphi_1$ , cât și  $S, \alpha \models \varphi_2$ , sau (2)  $S, \alpha \not\models \varphi_1$  și  $S, \alpha \not\models \varphi_2$ ;
7.  $S, \alpha \models (\exists x.\varphi)$  dacă există  $u \in D$  astfel încât  $S, \alpha[x \mapsto u] \models \varphi$ ;
8.  $S, \alpha \models (\forall x.\varphi)$  dacă pentru orice  $u \in D$ , avem că  $S, \alpha[x \mapsto u] \models \varphi$ .

**Exemplul 51.** Vom lucra în continuare peste signatura  $\Sigma = (\{\text{P}\}, \{\text{f}, \text{i}, \text{e}\})$ ,  $\Sigma$ -structura  $S_1 = (\mathbb{Z}, \{=\}, \{+, -, 0\})$  definită la începutul capitolului și  $S_1$ -atribuirile  $\alpha_1, \alpha_2$ .

Avem că

$$\begin{array}{ll} S_1, \alpha_1 \models \text{P}(\text{x}_1, \text{x}_1) & \text{P}^{S_1}(\bar{\alpha}_1(\text{x}_1), \bar{\alpha}_1(\text{x}_1)) \\ \text{dacă} & \bar{\alpha}_1(\text{x}_1) = \bar{\alpha}_1(\text{x}_1) \\ \text{dacă} & \alpha_1(\text{x}_1) = \alpha_1(\text{x}_1) \\ \text{dacă} & 5 = 5. \end{array}$$

Din moment ce  $5 = 5$ , rezultă că  $S_1, \alpha_1 \models \text{P}(\text{x}_1, \text{x}_1)$ , adică formula  $\text{P}(\text{x}_1, \text{x}_1)$  este adevărată în structura  $S_1$  cu atribuirea  $\alpha_1$ . Altfel spus,  $S_1$  satisfacă  $\text{P}(\text{x}_1, \text{x}_1)$  cu atribuirea  $\alpha_1$ .

**Exemplul 52.** Continuând exemplul anterior, avem

$$\begin{array}{ll}
 S_1, \alpha_1 \models P(x_1, x_3) & ddacă \\
 & ddacă \\
 & ddacă \\
 & ddacă
 \end{array}
 \qquad
 \begin{array}{l}
 P^{S_1}(\overline{\alpha_1}(x_1), \overline{\alpha_1}(x_3)) \\
 \overline{\alpha_1}(x_1) = \overline{\alpha_1}(x_3) \\
 \alpha_1(x_1) = \alpha_1(x_3) \\
 5 = 6.
 \end{array}$$

Din moment ce  $5 \neq 6$ , rezultă că  $S_1, \alpha_1 \not\models P(x_1, x_3)$ , adică formula  $P(x_1, x_3)$  este falsă în structura  $S_1$  cu atribuirea  $\alpha_1$ . Altfel spus  $S_1$  nu satisface  $P(x_1, x_3)$  cu atribuirea  $\alpha_1$ .

**Exemplul 53.** Continuând exemplul anterior, avem

$$\begin{array}{ll}
 S_1, \alpha_1 \models \neg P(x_1, x_3) & ddacă \\
 & ddacă \\
 & ddacă \\
 & ddacă
 \end{array}
 \qquad
 \begin{array}{l}
 S_1, \alpha_1 \not\models P(x_1, x_3) \\
 nu P^{S_1}(\overline{\alpha_1}(x_1), \overline{\alpha_1}(x_3)) \\
 nu \overline{\alpha_1}(x_1) = \overline{\alpha_1}(x_3) \\
 \overline{\alpha_1}(x_1) \neq \overline{\alpha_1}(x_3) \\
 \alpha_1(x_1) \neq \alpha_1(x_3) \\
 5 \neq 6.
 \end{array}$$

Din moment ce  $5 \neq 6$ , rezultă că  $S_1, \alpha_1 \models \neg P(x_1, x_3)$ , adică formula  $\neg P(x_1, x_3)$  este adevărată în structura  $S_1$  cu atribuirea  $\alpha_1$ . Altfel spus,  $S_1$  satisface  $\neg P(x_1, x_3)$  cu atribuirea  $\alpha_1$ .

**Exemplul 54.** Continuând exemplul anterior, avem

$$\begin{array}{ll}
 S_1, \alpha_1 \models P(x_1, x_1) \wedge \neg P(x_1, x_3) & ddacă \\
 S_1, \alpha_1 \models P(x_1, x_1) și S_1, \alpha_1 \models \neg P(x_1, x_3) & ddacă \\
 ... și ... & ddacă
 \end{array}$$

$5 = 5$  și  $5 \neq 6$ .

Din moment ce  $5 = 5$  și  $5 \neq 6$ , rezultă că  $S_1, \alpha_1 \models P(x_1, x_1) \wedge \neg P(x_1, x_3)$ .

**Exemplul 55.** Continuând exemplul anterior, avem

$$S_1, \alpha_1 \models P(x_1, x_3) \vee P(x_1, x_1) dacă S_1, \alpha_1 \models P(x_1, x_3) sau S_1, \alpha_1 \models P(x_1, x_1).$$

Am stabilit deja că  $S_1, \alpha_1 \models P(x_1, x_3)$ , deci  $S_1, \alpha_1 \models P(x_1, x_3) \vee P(x_1, x_1)$  (chiar dacă  $S_1, \alpha_1 \not\models P(x_1, x_1)$ ).

**Exemplul 56.** Continuând exemplul anterior, avem

$S_1, \alpha_1 \models \exists x_1.P(x_1, x_3)$	ddacă
există $u \in D$ a.î. $S_1, \alpha_1[x_1 \mapsto u] \models P(x_1, x_3)$	ddacă
există $u \in D$ a.î. $P^{S_1}(\overline{\alpha_1[x_1 \mapsto u]}(x_1), \overline{\alpha_1[x_1 \mapsto u]}(x_3))$	ddacă
există $u \in D$ a.î. $\overline{\alpha_1[x_1 \mapsto u]}(x_1) = \overline{\alpha_1[x_1 \mapsto u]}(x_3)$	ddacă
există $u \in D$ a.î. $\alpha_1[x_1 \mapsto u](x_1) = \alpha_1[x_1 \mapsto u](x_3)$	ddacă
există $u \in D$ a.î. $u = \alpha_1(x_3)$	ddacă
există $u \in D$ a.î. $u = 6$ .	

Din moment ce există  $u$  (putem alege  $u = 6$ ) a.î.  $u = 6$ , avem că  $S_1, \alpha_1 \models \exists x_1.P(x_1, x_3)$ .

**Exemplul 57.** Continuând exemplul anterior, avem

$S_1, \alpha_1 \models \forall x_1 \exists x_3.P(x_1, x_3)$	ddacă
pentru orice $u \in D$ , avem că $S_1, \alpha_1[x_1 \mapsto u] \models \exists x_3.P(x_1, x_3)$	ddacă
pt. orice $u \in D$ , există $v \in D$ a.î. $S_1, \alpha_1[x_1 \mapsto u][x_3 \mapsto v] \models P(x_1, x_3)$	ddacă
...	ddacă
pentru orice $u \in D$ , avem că există $v \in D$ a.î. $u = v$ .	

Din moment ce pentru orice număr întreg  $u$ , există un număr întreg  $v$  a.î.  $u = v$ , avem că  $S_1, \alpha_1 \models \forall x_1 \exists x_3.P(x_1, x_3)$ .

**Exercițiul 58.** Arătați că  $S_1, \alpha_1 \models \forall x_1 \exists x_3.P(x_1, i(x_3))$ .

### 4.3 Satisfiabilitate într-o structură fixată

**Definiția 59** (Satisfiabilitate într-o structură fixată). O formulă  $\varphi$  este satisfiabilă într-o structură  $S$  dacă există o  $S$ -atribuire  $\alpha$  cu proprietatea că

$$S, \alpha \models \varphi.$$

**Exemplul 60.** Formula  $P(x_1, x_3)$  este satisfiabilă în structura  $S_1$ , deoarece există o atribuire, de exemplu  $\alpha_2$ , cu proprietatea că  $S_1, \alpha_2 \models P(x_1, x_3)$ .

**Exercițiul 61.** Arătați că formula  $\neg P(x_1, x_1)$  nu este satisfiabilă în structura  $S_1$  (deoarece, pentru orice atribuire  $\alpha$  aleasă, avem că  $S_1, \alpha \not\models \neg P(x_1, x_1)$ ).

## 4.4 Validitate într-o structură fixată

**Definiția 62** (Validitate într-o structură fixată). *O formulă  $\varphi$  este validă într-o structură  $S$  dacă pentru orice  $S$ -atribuire  $\alpha$ , avem că*

$$S, \alpha \models \varphi.$$

**Exemplul 63.** Formula  $P(x_1, x_3)$  nu este validă în structura  $S_1$ , deoarece există o atribuire, și anume  $\alpha_1$ , cu proprietatea că  $S_1, \alpha_1 \not\models P(x_1, x_3)$ .

**Exercițiul 64.** Arătați că formula  $P(x_1, x_1)$  este validă în structura  $S_1$  (deoarece orice atribuire  $\alpha$  am alege,  $S_1, \alpha \models P(x_1, x_1)$ ).

## 4.5 Satisfiabilitate

**Definiția 65** (Satisfiabilitate). *O formulă  $\varphi$  este satisfiabilă dacă există o structură  $S$  și o  $S$ -atribuire  $\alpha$  cu proprietatea că*

$$S, \alpha \models \varphi.$$

**Exemplul 66.** Formula  $\neg P(x_1, x_1)$  este satisfiabilă, deoarece există o structură (de exemplu  $S_5$ ) și o  $S_5$ -atribuire (de exemplu  $\alpha_1$ ) astfel încât  $S_5, \alpha_1 \models \neg P(x_1, x_1)$  (deoarece  $5 \not\prec 5$ ).

Să subliniem faptul că, deoarece  $S_5$  și  $S_1$  au același domeniu, atribuirea  $\alpha_1$  este atât o  $S_1$ -atribuire cât și o  $S_5$ -atribuire.

**Observație.** O formulă poate să nu fie satisfiabilă într-o structură fixată (de exemplu  $\neg P(x_1, x_1)$  nu este satisfiabilă în structura  $S_1$ ) și totuși să fie satisfiabilă (vezi Exemplul 66 unde aceeași formulă  $\neg P(x_1, x_1)$  este satisfiabilă).

## 4.6 Validitate

**Definiția 67** (Validitate). *O formulă  $\varphi$  este validă dacă, pentru orice structură  $S$  și pentru orice  $S$ -atribuire  $\alpha$ , avem*

$$S, \alpha \models \varphi.$$

**Exemplul 68.** Formula  $P(x_1, x_1)$  nu este validă, deoarece  $S_5, \alpha_1 \not\models P(x_1, x_1)$ . Pe de altă parte, formula  $P(x_1, x_1) \rightarrow P(x_1, x_1)$  este validă.

**Observație.** O formulă poate să fie validă într-o structură fixată (de exemplu  $P(x_1, x_1)$  este validă în structura  $S_1$ ) și totuși să nu fie validă (de exemplu,  $P(x_1, x_1)$  nu este validă, deoarece  $S_5, \alpha_1 \not\models P(x_1, x_1)$ ).

## 4.7 Consecință semantică

**Definiția 69.** O formulă  $\varphi$  este consecință semantică a formulelor  $\varphi_1, \dots, \varphi_n$  într-o structură fixată  $S$ , notat  $\varphi_1, \dots, \varphi_n \models_S \varphi$ , dacă, pentru orice  $S$ -atribuire  $\alpha$  pentru care  $S, \alpha \models \varphi_1, S, \alpha \models \varphi_2, \dots, S, \alpha \models \varphi_n$ , avem că  $S, \alpha \models \varphi$ .

**Exemplul 70.** Avem că  $P(x, y) \models_{S_1} P(y, x)$ , deoarece, pentru orice  $S_1$ -atribuire  $\alpha$  cu proprietatea că  $S_1, \alpha \models P(x, y)$  (adică  $\alpha(x) = \alpha(y)$ ), avem și că  $S_1, \alpha \models P(y, x)$  (adică  $\alpha(y) = \alpha(x)$ ).

Avem că  $P(x, y) \not\models_{S_5} P(y, x)$ , deoarece, pentru atribuirea  $\alpha(x) = 5, \alpha(y) = 6$ , avem că  $S_5, \alpha \models P(x, y)$  (adică  $5 < 6$ ), dar  $S_5, \alpha \not\models P(y, x)$  ( $6 \not< 5$ ).

**Definiția 71.** O formulă  $\varphi$  este consecință semantică a formulelor  $\varphi_1, \dots, \varphi_n$ , notat  $\varphi_1, \dots, \varphi_n \models \varphi$ , dacă

$$\varphi_1, \dots, \varphi_n \models_S \varphi$$

pentru orice structură  $S$ .

**Exemplul 72.** Avem că  $P(x, y) \not\models P(y, x)$ , deoarece există o structură (și anume  $S_5$ ) astfel încât  $P(x, y) \not\models_{S_5} P(y, x)$ .

**Exercițiul 73.** Arătați că:

$$\forall x. \forall y. \forall z. (P(x, y) \wedge P(y, z) \rightarrow P(x, z)), P(x_1, x_2), P(x_2, x_3) \models P(x_1, x_3).$$

Desigur că, în cele de mai sus (similar cu logica propozițională), lista  $\varphi_1, \varphi_2, \dots, \varphi_n$  denotă de fapt o mulțime având respectivele elemente.

## 4.8 Multime consistentă de formule

**Definiția 74.** O mulțime de formule  $\Gamma$  este consistentă într-o structură fixată  $S$  dacă există o  $S$ -atribuire  $\alpha$  cu proprietatea că  $S, \alpha \models \varphi$  pentru fiecare formulă  $\varphi \in \Gamma$ .

**Exemplul 75.** Avem că  $\Gamma = \{P(x_1, x_2), P(x_2, x_1)\}$  este consistentă în  $S_1$ :  $S_1$  face ambele formule adevărate cu atribuirea  $\alpha_1$ .

Aceeași mulțime de formule,  $\Gamma = \{P(x_1, x_2), P(x_2, x_1)\}$ , nu este consistentă în  $S_5$ : presupunând prin reducere la absurd că ar exista o atribuire  $\alpha$  cu care  $S_5$  să facă ambele formule din  $\Gamma$  adevărate, am avea  $\alpha(x_1) < \alpha(x_2) < \alpha(x_1)$ , ceea ce este imposibil.

**Definiția 76.** O mulțime de formule  $\Gamma$  este consistentă dacă există o structură  $S$  și o  $S$ -atribuire  $\alpha$  cu proprietatea că  $S, \alpha \models \varphi$  pentru fiecare formulă  $\varphi \in \Gamma$ .

**Exemplul 77.** Avem că  $\Gamma = \{P(x_1, x_2), P(x_2, x_1)\}$  este consistentă, deoarece este consistentă în  $S_1$ .

Avem că  $\Gamma' = \{P(x_1, x_2), \neg P(x_1, x_2)\}$  nu este consistentă.

**Exercițiul 78.** Arătați că  $\Gamma'$  este inconsistentă.

## 4.9 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 același înțeles este  $\neg(\forall x.Q(x))$  și  $(\exists x.\neg Q(x))$ . 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 predicator simetric, formulele  $P(x, y)$  și respectiv  $P(y, x)$  au același înțeles. Astfel de formule se numesc *echivalente în structura respectivă*.

Acst aspect este surprins de următoarele definiții.

**Definiția 79.** Două formule  $\varphi_1 \in \mathbb{LPI}$  și  $\varphi_2 \in \mathbb{LPI}$  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 80.** Continuăm exemplele din cursurile anterioare. Considerăm semnatura  $\Sigma = (\{P\}, \{f, i, e\})$  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  $\models$ )  $P^{S_1}(\bar{\alpha}(x), \bar{\alpha}(y))$   
 ddacă  $\bar{\alpha}(x) = \bar{\alpha}(y)$   
 ddacă  $\alpha(x) = \alpha(y)$   
 ddacă (prin simetria relației de egalitate)  $\alpha(y) = \alpha(x)$   
 ddacă  $\bar{\alpha}(y) = \bar{\alpha}(x)$   
 ddacă (prin definiția relației  $\models$ )  $S_1, \alpha \models P(y, x)$ .

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

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

Deoarece există o  $S_1$ -atribuire  $\alpha : \mathcal{X} \rightarrow \mathbb{Z}$ , definită prin  $\alpha(\textcolor{red}{x}_1) = 42, \alpha(\textcolor{red}{x}_2) = 7, \alpha(\textcolor{red}{x}_3) = 42$  (pentru restul variabilelor nu este relevantă valoarea lor în atribuire) cu proprietatea că

$$\begin{aligned} S_1, \alpha \models P(x_1, x_3) & (\text{deoarece } 42 = 42), \text{ dar} \\ S_1, \alpha \not\models P(x_2, x_3) & (\text{deoarece } 42 \neq 7). \end{aligned}$$

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

**Definiția 81.** Două formule  $\varphi_1 \in \mathbb{LPI}$  și  $\varphi_2 \in \mathbb{LPI}$  sunt echivalente dacă, pentru orice structură  $S$  și pentru orice  $S$ -atribuire  $\alpha$ ,

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

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

**Exemplul 82.** Continuăm exemplul anterior. Avem că  $P(x, y) \not\equiv P(y, x)$ . De ce?

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

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

Observați că singura diferență între  $S_1$  și  $S_5$  este faptul că simbolul predicator  $P$  este interpretat prin  $\text{predicatul } =$  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  $S_1$ ).

**Exemplul 83.** 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  $\alpha : \mathcal{X} \rightarrow D$  o  $S$ -atribuire oarecare. Avem că

$$\begin{aligned} S, \alpha \models (\forall x.P(x, z)) & \quad \text{dacă} \\ \text{pentru orice } u \in D, \text{ avem } S, \alpha[x \mapsto u] \models P(x, z) & \quad \text{dacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(\overline{\alpha[x \mapsto u]}(\textcolor{red}{x}), \overline{\alpha[x \mapsto u]}(z)\right) & \quad \text{dacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(u, \alpha(z)\right) & \quad \text{dacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(\overline{\alpha[y \mapsto u]}(\textcolor{red}{y}), \overline{\alpha[y \mapsto u]}(z)\right) & \quad \text{dacă} \\ \text{pentru orice } u \in D, \text{ avem } S, \alpha[y \mapsto u] \models P(y, z) & \quad \text{dacă} \\ S, \alpha \models (\forall y.P(y, z)). & \end{aligned}$$

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

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

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

## 4.10 Fișă de exerciții

Amintim mai jos structurile din Exemplul 41:

1.  $S_1 = (\mathbb{Z}, \{=\}, \{+, -, 0\})$ ;
2.  $S_2 = (\mathbb{R}^*, \{=\}, \{\times, \cdot^{-1}, 1\})$ ;
3.  $S_3 = (\mathbb{N}, \{=\}, \{+, s, 0\})$ ;
4.  $S_4 = (\mathbb{N}, \{<\}, \{+, s, 0\})$ .
5.  $S_5 = (\mathbb{Z}, \{<\}, \{+, -, 0\})$ .

Acstea structuri vor fi utilizate în exercițiile de mai jos.

**Exercițiu 84.** Stabiliti dacă:

1.  $S_1, \alpha_1 \models P(x_2, x_3)$ ;
2.  $S_1, \alpha_1 \models \neg P(x_2, x_3)$ ;
3.  $S_1, \alpha_1 \models \neg P(x_2, x_3) \wedge P(x_1, x_1)$ ;
4.  $S_1, \alpha_1 \models \exists x_3.P(x_2, x_3)$ ;
5.  $S_1, \alpha_1 \models \forall x_2.\exists x_3.P(x_2, x_3)$ ;
6.  $S_1, \alpha_1 \models \exists x_3.\forall x_2.P(x_2, x_3)$ ;
7.  $S_1, \alpha_2 \models \forall x_2.\exists x_3.P(x_2, i(x_3))$ ;

**Exercițiu 85.** Găsiți pentru fiecare dintre itemii de mai jos câte o  $S_2$ -atribuire  $\alpha_3$  astfel încât:

1.  $S_2, \alpha_3 \models P(x_1, x_2)$ ;
2.  $S_2, \alpha_3 \models P(f(x_1, x_2), x_3)$ ;
3.  $S_2, \alpha_3 \models P(f(x_1, x_2), i(x_3))$ ;
4.  $S_2, \alpha_3 \models P(x, e)$ ;

5.  $S_2, \alpha_3 \models \exists y.P(x, i(y));$
6.  $S_2, \alpha_3 \models \forall y.\exists x.P(x, i(y)).$

**Exercițiul 86.** Arătați că următoarele formule sunt valide în  $S_2$ :

1.  $\forall x.\exists y.P(x, i(y));$
2.  $\forall x.P(f(x, e), x);$
3.  $\forall x.P(x, i(i(x))).$

**Exercițiul 87.** Arătați că formula  $\forall x.\exists y.P(x, i(y))$  nu este validă în  $S_3$ .

**Exercițiul 88.** Găsiți o formulă care să fie satisfiabilă în  $S_1$  dar nu în  $S_3$ .

**Exercițiul 89.** Găsiți o formulă fără variabile libere care să fie satisfiabilă în  $S_5$  dar nu în  $S_4$ .

**Exercițiul 90.** Arătați că formula  $\forall x.\exists y.P(x, y)$  nu este validă.

**Exercițiul 91.** Arătați că formula  $(\forall x.P(x, x)) \rightarrow \exists x_2.P(x_1, x_2)$  este validă.

**Exercițiul 92.** Arătați că formula  $\forall x.\exists y.P(y, x)$  nu este validă.

**Exercițiul 93.** Arătați că formula  $\forall x.\neg P(x, x)$  este satisfiabilă.

**Exercițiul 94.** Arătați că formula  $\forall x.\neg P(x, x) \wedge \exists x.P(x, x)$  nu este satisfiabilă.

**Exercițiul 95.** În Exercițiul 40 (din Capitolul 3) am arătat că  $\mathbb{LPI}$  este o extensie sintactică a lui  $\mathbb{LP}$ . Pe scurt, dacă  $A = \{p, q, r, \dots\}$  este o multime de variabile propoziționale atunci construim o signațură  $\Sigma_{\mathbb{LP}} = \{A, \emptyset\}$ , unde variabilele propoziționale din  $A$  sunt simboluri predicative de aritate 0.

Semantica formulelor  $\mathbb{LPI}$  construite peste signațură  $\Sigma_{\mathbb{LP}}$  este consistentă cu semantica formulelor  $\mathbb{LP}$ . Fie  $\tau : A \rightarrow B$  o atribuire. Fie  $S = (D, \{a^S \mid a \in A\}, \emptyset)$  o  $\Sigma_{\mathbb{LP}}$ -structură, unde  $D$  este orice multime nenulă și  $a^S = \tau(a)$ , pentru orice  $a \in A$ .

Demonstrați că pentru orice  $\varphi \in \mathbb{LP}$ , avem că  $\tau \models \varphi$  dacă  $S, \alpha \models \varphi$  pentru orice  $S$ -atribuire  $\alpha : \mathcal{X} \rightarrow D$ .



# Capitolul 5

## Deductia naturală

În acest capitol, vom prezenta deductia naturală pentru logica de ordinul I. Vom defini noțiunea de *substituție*, vom reaminti câteva dintre noțiunile specifice deductiei naturale (discutate în prealabil la logica propozițională) și vom prezenta sistemul deductiv extins împreună cu proprietățile sale (corectitudine și completitudine).

**Observație.** Regulile sistemului deductiv al deductiei naturale include regulile discutate deja la logica propozițională. În acest capitol, acestea din urmă sunt reluate și exemplificate pe formule de ordinul I. Deductia naturală pentru logica de ordinul I include reguli pentru cuantificatori. Acestea sunt reguli care nu au fost studiate anterior la logica propozițională.

### 5.1 Substituții

**Definiția 96.** O substituție este o funcție  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$ , cu proprietatea că  $\sigma(x) \neq x$  pentru un număr finit de variabile  $x \in \mathcal{X}$ .

**Definiția 97.** Dacă  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$  este o substituție, atunci mulțimea  $\text{dom}(\sigma) = \{x \in \mathcal{X} \mid \sigma(x) \neq x\}$  se numește domeniul substituției  $\sigma$ .

**Observație.** Prin definiție, domeniul unei substituții este o mulțime finită.

**Definiția 98.** Dacă  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$  este o substituție, atunci extensia unică a substituției  $\sigma$  la mulțimea termenilor este funcția  $\sigma^\sharp : \mathcal{T} \rightarrow \mathcal{T}$ , definită recursiv 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.  $\sigma^\sharp(f(t_1, \dots, t_n)) = f(\sigma^\sharp(t_1), \dots, \sigma^\sharp(t_n))$ , pentru orice simbol funcțional  $f \in \mathcal{F}_n$  de aritate  $n \in \mathbb{N}^*$  și orice termeni  $t_1, \dots, t_n \in \mathcal{T}$ .

De regulă, substituțiile se notează cu  $\sigma, \tau, \sigma_0, \tau_1, \sigma'$ , etc.

**Observație.** Dacă  $t \in \mathcal{T}$  este un termen, atunci  $\sigma^\sharp(t) \in \mathcal{T}$  este termenul obținut din  $t$  prin aplicarea substituției  $\sigma$  sau termenul obținut prin aplicarea substituției  $\sigma$  asupra termenului  $t$ .

Practic, pentru a obține  $\sigma^\sharp(t)$  din  $t$ , toate aparițiile unei variabile  $x$  din  $t$  sunt înlocuite simultan cu termenul corespunzător  $\sigma(x)$ .

**Exemplul 99.** Fie substituția  $\sigma_1 : \mathcal{X} \rightarrow \mathcal{T}$  definită astfel:

1.  $\sigma_1(x_1) = x_2$ ;
2.  $\sigma_1(x_2) = f(x_3, x_4)$ ;
3.  $\sigma_1(x) = x$  pentru orice  $x \in \mathcal{X} \setminus \{x_1, x_2\}$ .

Fie termenul  $t = f(f(x_1, x_2), f(x_3, e))$ . Avem că:

$$\begin{aligned}\sigma_1^\sharp(t) &= \sigma_1^\sharp\left(f(f(x_1, x_2), f(x_3, e))\right) \\ &= f(\sigma_1^\sharp(f(x_1, x_2)), \sigma_1^\sharp(f(x_3, e))) \\ &= f(f(\sigma_1^\sharp(x_1), \sigma_1^\sharp(x_2)), f(\sigma_1^\sharp(x_3), \sigma_1^\sharp(e))) \\ &= f(f(\sigma_1(x_1), \sigma_1(x_2)), f(\sigma_1(x_3), e)) \\ &= f(f(x_2, f(x_3, x_4)), f(x_3, e)).\end{aligned}$$

Observați că prin aplicarea unei substituții asupra unui termen, se înlocuiesc (simultan) toate aparițiile variabilelor din domeniul substituției cu termenii asociați acestora.

**Notatie.** Dacă  $\text{dom}(\sigma) = \{x_1, \dots, x_n\}$ , atunci substituția  $\sigma$  se mai poate scrie în felul următor:

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

Atenție, nu este vorba de o mulțime, ci doar de o **notatie** pentru substituții.

**Exemplul 100.** Pentru substituția din Exemplul 99, avem

$$\sigma_1 = \{x_1 \mapsto x_2, x_2 \mapsto f(x_3, x_4)\}.$$

**Definiția 101.** Dacă  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$  este o substituție și  $V \subseteq \mathcal{X}$  este o submulțime de variabile, atunci restricția substituției  $\sigma$  la mulțimea  $V$  este o nouă substituție notată  $\sigma|_V : \mathcal{X} \rightarrow \mathcal{T}$ , definită astfel:

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

**Exemplul 102.** Pentru substituția din Exemplul 99, avem  $\sigma_1|_{\{\textcolor{red}{x}_1\}} = \{\textcolor{red}{x}_1 \mapsto \textcolor{blue}{x}_2\}$  și  $\sigma_1|_{\{\textcolor{red}{x}_2\}} = \{\textcolor{red}{x}_2 \mapsto \textcolor{blue}{f}(\textcolor{red}{x}_3, \textcolor{red}{x}_4)\}$ .

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

**Definiția 103.** Pentru orice substituție  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$ , extensia lui  $\sigma$  la mulțimea formulelor este funcția  $\sigma^b : \mathbb{LPI} \rightarrow \mathbb{LPI}$ , definită astfel:

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

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

**Exemplul 104.** Utilizând substituția din Exemplul 99, avem că:

$$\begin{aligned} \sigma_1^b((\forall \textcolor{red}{x}_2.P(\textcolor{red}{x}_1, \textcolor{red}{x}_2)) \wedge P(\textcolor{red}{x}_2, \textcolor{red}{x}_2)) &= \\ \sigma_1^b((\forall \textcolor{red}{x}_2.P(\textcolor{red}{x}_1, \textcolor{red}{x}_2))) \wedge \sigma_1^b(P(\textcolor{red}{x}_2, \textcolor{red}{x}_2)) &= \\ (\forall \textcolor{red}{x}_2.\sigma_1|_{\{\textcolor{red}{x}_1\}}^\sharp(P(\textcolor{red}{x}_1, \textcolor{red}{x}_2))) \wedge P(\sigma_1^\sharp(\textcolor{red}{x}_2), \sigma_1^\sharp(\textcolor{red}{x}_2)) &= \\ (\forall \textcolor{red}{x}_2.P(\sigma_1|_{\{\textcolor{red}{x}_1\}}^\sharp(\textcolor{red}{x}_1), \sigma_1|_{\{\textcolor{red}{x}_1\}}^\sharp(\textcolor{red}{x}_2))) \wedge P(\sigma_1(\textcolor{red}{x}_2), \sigma_1(\textcolor{red}{x}_2)) &= \\ (\forall \textcolor{red}{x}_2.P(\sigma_1|_{\{\textcolor{red}{x}_1\}}(\textcolor{red}{x}_1), \sigma_1|_{\{\textcolor{red}{x}_1\}}(\textcolor{red}{x}_2))) \wedge P(f(\textcolor{red}{x}_3, \textcolor{red}{x}_4), f(\textcolor{red}{x}_3, \textcolor{red}{x}_4)) &= \\ (\forall \textcolor{red}{x}_2.P(\sigma_1(\textcolor{red}{x}_1), \textcolor{red}{x}_2)) \wedge P(f(\textcolor{red}{x}_3, \textcolor{red}{x}_4), f(\textcolor{red}{x}_3, \textcolor{red}{x}_4)) &= \\ (\forall \textcolor{red}{x}_2.P(\textcolor{red}{x}_2, \textcolor{red}{x}_2)) \wedge P(f(\textcolor{red}{x}_3, \textcolor{red}{x}_4), f(\textcolor{red}{x}_3, \textcolor{red}{x}_4)). & \end{aligned}$$

**Observație.** Atenție: aparițiile legate ale variabilelor nu sunt înlocuite prin aplicarea substituției! În Exemplul 104, apariția lui  $x_2$  în  $(\forall x_2.P(x_1, x_2))$  este legată.

**Notație.** Conform Notației 5.1 pentru substituțiile care au domeniul finit mai utilizăm notația  $\{x_1 \mapsto \sigma(x_1), \dots, x_n \mapsto \sigma(x_n)\}$ . De multe ori vom utiliza substituții pentru care nu vom mai asocia un nume, deoarece ele sunt foarte simple având forma:  $\{x \mapsto t\}$ . Pentru a exprima faptul că aplicăm această substituție unei formule, conform notărilor noastre, ar trebui să scriem  $\{x \mapsto t\}(\varphi)$ . Însă în literatură sunt preferate alte notări pe care le vom utiliza și noi. O variantă este să scriem  $\varphi[t/x]$ . O altă variantă este  $\varphi[x \mapsto t]$ . În acest document vom prefera ultima notăție, adică  $\varphi[x \mapsto t]$ .

## 5.2 Secvențe

**Definiția 105** (Secvență). O secvență este o pereche de formă:

$$\{\varphi_1, \dots, \varphi_n\} \vdash \varphi,$$

unde  $\{\varphi_1, \dots, \varphi_n\} \subseteq \mathbb{LPI}$  este o mulțime de formule iar  $\varphi \in \mathbb{LPI}$  este o formulă.

Câteodată citim notația  $\{\varphi_1, \dots, \varphi_n\} \vdash \varphi$  sub forma  $\varphi$  este consecință sintactică din  $\{\varphi_1, \dots, \varphi_n\}$ . De multe ori, vom nota cu  $\Gamma = \{\varphi_1, \dots, \varphi_n\}$  mulțimea de ipoteze și în acest caz vom scrie secvența ca  $\Gamma \vdash \varphi$ .

**Observație.** Ca și în cazul logicii propoziționale, este permisă scrierea fără accolade, adică  $\varphi_1, \dots, \varphi_n \vdash \varphi$ , în loc de  $\{\varphi_1, \dots, \varphi_n\} \vdash \varphi$ . Totuși trebuie să ținem cont că în partea stângă a simbolului  $\vdash$  este tot timpul o mulțime. Această notăție fără accolade ne permite să scriem  $\varphi_1, \dots, \varphi_n, \psi \vdash \varphi$  în loc de  $\{\varphi_1, \dots, \varphi_n\} \cup \{\psi\} \vdash \varphi$ , ceea ce ușurează citirea secvențelor.

**Exemplul 106.** În multe dintre exemplele din acest material vom lucra cu o semnură  $\Sigma = (\{P, Q\}, \{a, b, f, g\})$ , unde simbolurile predicative  $P$  și  $Q$  au aritate 1, simbolurile funcționale  $f$  și  $g$  au aritate 1, iar simbolurile  $a$  și  $b$  sunt constante (de aritate 0).

**Exemplul 107.** Fie semnura  $\Sigma$  din Exemplul 106. Iată câteva exemple de secvențe:

1.  $\{P(a), Q(a)\} \vdash (P(a) \wedge Q(a));$
2.  $\{\forall x.Q(x), P(a)\} \vdash (P(a) \wedge Q(a));$
3.  $\{\exists x.Q(x)\} \vdash Q(a).$

*Mai târziu vom vedea că primele două secvențe de mai sus sunt valide, iar ultima secvență nu este validă.*

Uneori este convenabil să scriem secvențele fără accolade, ca în exemplul următor:

**Exemplul 108.** *Secvențele din Exemplul 107 pot fi scrise fără accolade astfel:*

1.  $P(a), Q(a) \vdash (P(a) \wedge Q(a));$
2.  $\forall x.Q(x), P(a) \vdash (P(a) \wedge Q(a));$
3.  $\exists x.Q(x) \vdash Q(a).$

### 5.3 Reguli de inferență

**Definiția 109.** *O regulă de inferență este un tuplu format din:*

1. *o mulțime de secvențe  $S_1, \dots, S_n$ , care se numesc ipotezele regulii;*
2. *o secvență  $S$  care se numește concluzia regulii;*
3. *o condiție de aplicare a regulii;*
4. *un nume.*

O regulă de inferență se notează în felul următor:

$$\text{NUME} \frac{S_1 \quad \dots \quad S_n}{S} \text{ condiție.}$$

**Observație.** *Regulile de inferență care au  $n = 0$  ipoteze, se numesc axioane. De asemenea, condiția de aplicare poate să lipsească.*

**Exemplul 110.** Iată câteva exemple de reguli de inferență pe care le-am întâlnit și la logica propozițională:

$$\wedge_i \frac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}, \quad \wedge_{e1} \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_1}, \quad \wedge_{e2} \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_2}.$$

*Ca și în logica propozițională, toate cele trei reguli de inferență de mai sus sunt corecte. Niciuna dintre cele trei reguli de mai sus nu are o condiție atașată. Iată și un exemplu de regulă cu  $n = 0$  ipoteze, dar cu o condiție:*

$$\text{IPOTEZĂ} \frac{}{\Gamma \vdash \varphi_1} \varphi_1 \in \Gamma.$$

*Mai jos avem un exemplu de regulă de inferență incorectă (într-un sens pe care îl vom preciza mai târziu, dar care poate fi deja intuit):*

$$\text{REGULĂ INCORECTĂ } \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}.$$

**Observație.** Ipotezele regulii de inferență, precum și concluzia, sunt de fapt scheme de secvențe și nu secvențe propriu-zise. Aceste scheme pot fi instanțiate, adică o regulă de inferență (prezentată ca mai sus) are mai multe instanțe, obținute prin înlocuirea variabilelor matematice  $\varphi, \varphi'$ ,  $\Gamma$  cu formele concrete. De exemplu, iată două instanțe ale regulii  $\wedge i$  de mai sus:

$$\begin{aligned} \wedge i \frac{\{P(a), Q(a)\} \vdash P(a) \quad \{P(a), Q(a)\} \vdash Q(a)}{\{P(a), Q(a)\} \vdash (P(a) \wedge Q(a));} \\ \wedge i \frac{\{P(a), Q(a), Q(b)\} \vdash (P(a) \wedge Q(a)) \quad \{P(a), Q(a), Q(b)\} \vdash P(a)}{\{P(a), Q(a), Q(b)\} \vdash ((P(a) \wedge Q(a)) \wedge P(a))}. \end{aligned}$$

În prima instantă, am înlocuit variabila matematică  $\Gamma$  cu multimea de formule  $\{P(a), Q(a)\}$ , variabila matematică  $\varphi$  cu formula  $P(a)$  și variabila matematică  $\varphi'$  cu formula  $Q(a)$ . Exercițiu: stabiliți singuri cu ce am înlocuit fiecare variabilă matematică în cea de-a doua instanță.

Iată un exemplu de regulă care nu este instanță a regulii  $\wedge i$  (exercițiu: explicați de ce nu):

$$? \frac{\{P(a), Q(a)\} \vdash P(a) \quad \{P(a), Q(a)\} \vdash Q(a)}{\{P(a), Q(a)\} \vdash (P(a) \wedge Q(a));}$$

## 5.4 Sistem deductiv

**Definiția 111.** Un sistem deductiv este o mulțime de reguli de inferență.

**Exemplul 112.** Fie sistemul deductiv  $D_1$ , format din următoarele patru reguli de inferență:

$$\begin{aligned} \text{IPOTEZĂ } \frac{}{\Gamma \vdash \varphi,} \varphi \in \Gamma & \quad \wedge i \frac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}, & \quad \wedge e_1 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_1}, \\ & \quad \wedge e_2 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_2}. \end{aligned}$$

## 5.5 Demonstrație formală

**Definiția 113** (Demonstrație formală). *O demonstrație formală într-un sistem deductiv este o listă de secvențe*

$$1. S_1$$

$$2. S_2$$

...

$$n. S_n,$$

cu proprietatea că fiecare secvență  $S_i$  este justificată de o regulă de inferență a sistemului deductiv din secvențele anterioare  $(S_1, \dots, S_{i-1})$ , în sensul în care  $S_i$  este concluzia unei instanțe a unei reguli de inferență din sistemul deductiv, regulă care folosește ca ipoteze doar secvențe alese dintre  $S_1, \dots, S_{i-1}$ . În plus, dacă regula de inferență are condiție, această condiție trebuie să fie adevărată. Să notăm și faptul că orice prefix al unei demonstrații este tot o demonstrație.

**Exemplul 114.** Iată un exemplu de demonstrație formală în sistemul  $D_1$  introdus mai sus:

1.  $\{P(a), Q(a)\} \vdash P(a); \quad (\text{IPOTEZĂ})$
2.  $\{P(a), Q(a)\} \vdash Q(a); \quad (\text{IPOTEZĂ})$
3.  $\{P(a), Q(a)\} \vdash (P(a) \wedge Q(a)); \quad (\wedge i, 1, 2)$
4.  $\{P(a), Q(a)\} \vdash (Q(a) \wedge (P(a) \wedge Q(a))). \quad (\wedge i, 2, 3)$

Ca și în cazul logicii propozitionale, fiecare linie este adnotată cu numele regulii de inferență aplicate, plus liniile la care se găsesc ipotezele necesare aplicării (în aceeași ordine folosită pentru prezentarea sistemului deductiv).

**Observație.** Definiția demonstrației formale în logica de ordinul întâi este aceeași ca în cazul logicii propozitionale. Totuși, vom vedea mai târziu că pentru aplicarea regulilor de inferență noi, asociate cuantificatorilor, vom folosi adnotări suplimentare.

**Definiția 115** (Secvență validă). *O secvență  $\Gamma \vdash \varphi$  este validă într-un sistem deductiv  $D$  dacă există o demonstrație formală  $S_1, \dots, S_n$  în  $D$  astfel încât  $S_n = \Gamma \vdash \varphi$ .*

**Exemplul 116.** Secvența  $\{P(a), Q(a)\} \vdash (P(a) \wedge Q(a))$  este validă în sistemul deductiv  $D_1$  de mai sus, deoarece este ultima secvență din următoarea demonstrație formală:

1.  $\{\text{P(a)}, \text{Q(a)}\} \vdash \text{P(a)}$ ; (IPOTEZĂ)
2.  $\{\text{P(a)}, \text{Q(a)}\} \vdash \text{Q(a)}$ ; (IPOTEZĂ)
3.  $\{\text{P(a)}, \text{Q(a)}\} \vdash (\text{P(a}) \wedge \text{Q(a)})$ ; ( $\wedge i$ , 1, 2)

**Observație.** Atenție! Nu confundați noțiunea de secvență validă într-un sistem deductiv cu noțiunea de formulă validă.

## 5.6 Deductia naturală

Deductia naturală este un sistem deductiv pentru logica de ordinul I. De fapt, sistemul deductiv pentru logica de ordinul I include toate regulile de deducție studiate la logica propozițională. În plus, pentru logica de ordinul I mai apar reguli noi, și anume cele de introducere și eliminare a cuantificatorilor. În această secțiune vom prezenta în detaliu fiecare regulă de inferență care aparține deductiei naturale în logica de ordinul I.

### 5.6.1 Regulile pentru conjuncții

Am văzut deja regulile de introducere și de eliminare pentru conectorul “și”:

$$\wedge_i \frac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}, \quad \wedge e_1 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_1}, \quad \wedge e_2 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_2}.$$

Regulile de inferență mimează raționamentul uman, bazat în esență pe o semantică intuitivă a noțiunii de adevăr:

1. Regula de introducere a conectorului  $\wedge$  ne indică că putem demonstra o conjuncție  $(\varphi_1 \wedge \varphi_2)$  din ipotezele  $\Gamma$  dacă știm deja că fiecare parte a conjuncției,  $\varphi_1$  și respectiv  $\varphi_2$ , sunt consecințe ale ipotezelor  $\Gamma$ .

Cu alte cuvinte, pentru a arăta o conjuncție dintr-un set de ipoteze, este suficient să stabilim individual că fiecare parte a conjuncției este o consecință a ipotezelor.

2. Pentru conectorul  $\wedge$  avem două reguli de eliminare. Prima regulă de eliminare a conectorului  $\wedge$  ne precizează că dacă am stabilit deja că o conjuncție  $(\varphi_1 \wedge \varphi_2)$  este consecința unei mulțimi  $\Gamma$  de ipoteze, atunci și partea stângă a conjuncției,  $\varphi_1$ , este consecință a mulțimii  $\Gamma$ .

A doua regulă este simetrică față de prima și ne permite să concluzionăm că și partea dreaptă a unei conjuncții este consecința unei mulțimi de formule dacă și conjuncția este consecința a aceleiași mulțimi de formule.

Iată un exemplu de demonstrație formală care utilizează regulile de inferență pentru conectorul  $\wedge$ :

1.  $\{(\text{P(a)} \wedge \text{Q(a)}), \forall x.P(x)\} \vdash (\text{P(a)} \wedge \text{Q(a)});$  (IPOTEZĂ)
2.  $\{(\text{P(a)} \wedge \text{Q(a)}), \forall x.P(x)\} \vdash \forall x.P(x);$  (IPOTEZĂ)
3.  $\{(\text{P(a)} \wedge \text{Q(a)}), \forall x.P(x)\} \vdash \text{P(a)};$  ( $\wedge e_1$ , 1)
4.  $\{(\text{P(a)} \wedge \text{Q(a)}), \forall x.P(x)\} \vdash (\text{P(a)} \wedge \forall x.P(x)).$  ( $\wedge i$ , 3, 2)

### 5.6.2 Regulile pentru implicații

Regula de eliminare a implicației, numită și *modus ponens* în latină, este una dintre cele mai importante reguli de inferență pe care le aplicăm.

$$\rightarrow e \frac{\Gamma \vdash (\varphi_1 \rightarrow \varphi_2) \quad \Gamma \vdash \varphi_1}{\Gamma \vdash \varphi_2}$$

Regula ne arată că, presupunând că am demonstrat  $(\varphi_1 \rightarrow \varphi_2)$  (din  $\Gamma$ ) și în plus am demonstrat și  $\varphi_1$  (tot din  $\Gamma$ ), atunci putem demonstra  $\varphi_2$  (din  $\Gamma$ ).

Iată un exemplu de demonstrație formală care folosește regula de eliminare a implicației:

1.  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \wedge \text{Q(a)});$  (IPOTEZĂ)
2.  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\} \vdash \text{P(a)};$  ( $\wedge e_1$ , 1)
3.  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \rightarrow \forall x.P(x));$  (IPOTEZĂ)
4.  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\} \vdash \forall x.P(x).$  ( $\rightarrow e$ , 3, 1)

Această demonstrație arată că secvența  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\} \vdash \forall x.P(x)$  este validă, adică formula  $\forall x.P(x)$  este o consecință a mulțimii de formule  $\{(\text{P(a)} \rightarrow \forall x.P(x)), (\text{P(a)} \wedge \text{Q(a)})\}$ . Observați ordinea în care apar liniile 3 și 1 în explicația liniei 4: urmează aceeași ordine, fixată prin regula de inferență.

**Exercițiul 117.** Arătați că sunt valide următoarele secvențe:

1.  $\{((\text{P(a)} \wedge \text{Q(a)}) \rightarrow \forall x.P(x)), \text{P(a)}, \text{Q(a)}\} \vdash \forall x.P(x);$
2.  $\{(\text{P(a)} \rightarrow \forall x.P(x)), \text{P(a)}, \text{Q(a)}\} \vdash (\text{Q(a)} \wedge \forall x.P(x)).$

Regula de introducere a implicației este mai subtilă. Pentru a arăta că o implicație  $(\varphi_1 \rightarrow \varphi_2)$  decurge din  $\Gamma$ , presupunem  $\varphi_1$  (în plus față de  $\Gamma$ ) și arătăm  $\varphi_2$ . Regula poate fi scrisă în două moduri echivalente, care se deosebesc doar prin faptul că prima regulă folosește convenția de notație referitoare la accoladele din jurul premiselor unei secvențe, în timp ce în a doua regulă accoladele care marchează mulțimea apar explicit:

$$\rightarrow i \frac{\Gamma, \varphi_1 \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \rightarrow \varphi_2)}, \quad \rightarrow i \frac{\Gamma \cup \{\varphi_1\} \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \rightarrow \varphi_2)}.$$

Ceea ce este important de observat și de înțeles la regula de introducere a implicației este că mulțimea de premise se schimbă. Dacă în concluzie avem că formula  $(\varphi_1 \rightarrow \varphi_2)$  decurge din  $\Gamma$ , în ipoteză trebuie să arătăm că  $\varphi_2$  decurge din premisele  $\Gamma \cup \{\varphi_1\}$ . Cu alte cuvinte, la modul intuitiv, pentru a demonstra o implicație  $(\varphi_1 \rightarrow \varphi_2)$ , presupunem antecedentul  $\varphi_1$  și arătăm consecventul  $\varphi_2$ .

**Exemplul 118.** Să arătăm că secvența  $\{\} \vdash (\text{P(a)} \rightarrow \text{P(a)})$  este validă:

1.  $\{\text{P(a)}\} \vdash \text{P(a)}$ ; (IPOTEZĂ)
2.  $\{\} \vdash (\text{P(a)} \rightarrow \text{P(a)})$ . ( $\rightarrow i$ , 1)

**Exemplul 119.** Să arătăm că secvența  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)})\} \vdash (\text{P(a)} \rightarrow \text{P(b)})$  este validă:

1.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)}), \text{P(a)}\} \vdash (\text{P(a)} \rightarrow \text{Q(a)})$ ; (IPOTEZĂ)
2.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)}), \text{P(a)}\} \vdash \text{P(a)}$ ; (IPOTEZĂ)
3.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)}), \text{P(a)}\} \vdash \text{Q(a)}$ ; ( $\rightarrow e$ , 1, 2)
4.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)}), \text{P(a)}\} \vdash (\text{Q(a)} \rightarrow \text{P(b)})$ ; (IPOTEZĂ)
5.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)}), \text{P(a)}\} \vdash \text{P(b)}$ ; ( $\rightarrow e$ , 4, 3)
6.  $\{(\text{P(a)} \rightarrow \text{Q(a)}), (\text{Q(a)} \rightarrow \text{P(b)})\} \vdash (\text{P(a)} \rightarrow \text{P(b)})$ . ( $\rightarrow i$ , 5)

**Exercițiul 120.** Arătați că următoarele secvențe sunt valide:

1.  $\{((\text{P(a)} \wedge \text{Q(a)}) \rightarrow \text{P(b)}), \text{P(a)}, \text{Q(a)}\} \vdash \text{P(b)}$ ;
2.  $\{((\text{P(a)} \wedge \text{Q(a)}) \rightarrow \text{P(b)})\} \vdash (\text{P(a)} \rightarrow (\text{Q(a)} \rightarrow \text{P(b)}))$ ;
3.  $\{(\text{P(a)} \rightarrow (\text{Q(a)} \rightarrow \text{P(b)}))\} \vdash ((\text{P(a)} \wedge \text{Q(a)}) \rightarrow \text{P(b)})$ .

### 5.6.3 Regulile pentru disjuncții

Conectorul  $\vee$  are două reguli de introducere:

$$\vee i_1 \frac{\Gamma \vdash \varphi_1}{\Gamma \vdash (\varphi_1 \vee \varphi_2)}, \quad \vee i_2 \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \vee \varphi_2)}.$$

Prima regulă ne arată că dacă știm  $\varphi_1$  (din  $\Gamma$ ), atunci știm și  $(\varphi_1 \vee \varphi_2)$  (din  $\Gamma$ ), indiferent de  $\varphi_2$ . A doua regulă de eliminare este simetrică, pentru partea dreaptă a disjuncției.

**Exemplul 121.** Să arătăm că secvența  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \vee \text{Q(a)})$  este validă:

1.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \wedge \text{Q(a)});$  (IPOTEZĂ)
2.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash \text{P(a)};$  ( $\wedge e_1$ , 1)
3.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \vee \text{Q(a)}).$  ( $\vee i_1$ , 2)

O altă demonstrație formală pentru aceeași secvență este:

1.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \wedge \text{Q(a)});$  (IPOTEZĂ)
2.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash \text{Q(a)};$  ( $\wedge e_2$ , 1)
3.  $\{(\text{P(a)} \wedge \text{Q(a)})\} \vdash (\text{P(a)} \vee \text{Q(a)}).$  ( $\vee i_2$ , 2)

Regula de eliminare a disjuncției este ușor mai complicată, fiind o altă regulă în care multimea de premise variază de la ipoteză la concluzie:

$$\vee e \frac{\Gamma \vdash (\varphi_1 \vee \varphi_2) \quad \Gamma, \varphi_1 \vdash \varphi' \quad \Gamma, \varphi_2 \vdash \varphi'}{\Gamma \vdash \varphi'}$$

Prima ipoteză a regulii,  $\Gamma \vdash (\varphi_1 \vee \varphi_2)$ , este ușor de înțeles: pentru a “elimina” o disjuncție, trebuie să avem o disjuncție printre ipoteze (disjuncție pe care să o “eliminăm”). Ultimele două ipoteze ale regulii de eliminare a disjuncției trebuie înțelese intuitiv după cum urmează. Din prima ipoteză știm  $(\varphi_1 \vee \varphi_2)$  (din  $\Gamma$ ); cu alte cuvinte, căcăd una dintre formulele  $\varphi_1$  și respectiv  $\varphi_2$  decurge din  $\Gamma$ . Ipotezele 2 și 3 ne indică faptul că, indiferent care dintre formulele  $\varphi_1$  și respectiv  $\varphi_2$  ar avea loc, în orice caz  $\varphi'$  are loc. Adică dacă presupunem  $\varphi_1$  (în plus față de  $\Gamma$ ),  $\varphi'$  are loc, iar dacă presupunem  $\varphi_2$  (în plus față de  $\Gamma$ ),  $\varphi'$  tot are loc. și atunci concluzia ne indică că  $\varphi'$  are loc indiferent care dintre  $\varphi_1$  și respectiv  $\varphi_2$  ar avea loc.

**Exemplul 122.** Să arătăm că secvența  $\{(\text{P}(a) \vee \text{Q}(a))\} \vdash (\text{Q}(a) \vee \text{P}(a))$  este validă:

1.  $\{(\text{P}(a) \vee \text{Q}(a)), \text{P}(a)\} \vdash \text{P}(a);$  (IPOTEZĂ)
2.  $\{(\text{P}(a) \vee \text{Q}(a)), \text{P}(a)\} \vdash (\text{Q}(a) \vee \text{P}(a));$  ( $\vee i_2$ , 1)
3.  $\{(\text{P}(a) \vee \text{Q}(a)), \text{Q}(a)\} \vdash \text{Q}(a);$  (IPOTEZĂ)
4.  $\{(\text{P}(a) \vee \text{Q}(a)), \text{Q}(a)\} \vdash (\text{Q}(a) \vee \text{P}(a));$  ( $\vee i_1$ , 1)
5.  $\{(\text{P}(a) \vee \text{Q}(a))\} \vdash (\text{P}(a) \vee \text{Q}(a));$  (IPOTEZĂ)
6.  $\{(\text{P}(a) \vee \text{Q}(a))\} \vdash (\text{Q}(a) \vee \text{P}(a)).$  ( $\vee e$ , 5, 2, 4)

Observați cu atenție modul în care mulțimea de premise variază de la o secvență la alta pe parcursul demonstrației formale, respectând regulile de inferență.

**Exercițiul 123.** Găsiți o demonstrație formală pentru secvența

$$\{(\text{P}(a) \vee \text{Q}(a)), (\text{P}(a) \rightarrow \text{P}(b)), (\text{Q}(a) \rightarrow \text{P}(b))\} \vdash \text{P}(b).$$

#### 5.6.4 Regulile pentru negații

Regulile pentru introducerea și respectiv eliminarea negației sunt prezentate împreună cu o regulă pentru eliminarea lui  $\perp$ :

$$\neg i \frac{\Gamma, \varphi \vdash \perp}{\Gamma \vdash \neg \varphi} \quad \neg e \frac{\Gamma \vdash \varphi \quad \Gamma \vdash \neg \varphi}{\Gamma \vdash \perp} \quad \perp e \frac{\Gamma \vdash \perp}{\Gamma \vdash \varphi}$$

Să ne readucem aminte că  $\perp$  este un conector logic de aritate 0. Cu alte cuvinte, conectorul  $\perp$  este de sine stătător o formulă. Semantica formulei  $\perp$  este că este falsă în orice atribuire. Cu alte cuvinte,  $\perp$  este o contradicție.

Prima regulă dintre cele de mai sus, cea de introducere a negației, este ușor de explicitat intuitiv: cum putem arăta că o formulă de forma  $\neg \varphi$  decurge din premisele  $\Gamma$ ? Presupunem, în plus față de premisele  $\Gamma$ , că avem  $\varphi$  și arătăm că din  $\Gamma$  și  $\varphi$  decurge o contradicție ( $\Gamma, \varphi \vdash \perp$ ). În acest fel, arătăm că  $\neg \varphi$  decurge din  $\Gamma$ .

A doua regulă, pentru eliminarea negației, ne indică faptul că dacă atât o formulă  $\varphi$ , cât și negația sa,  $\neg \varphi$ , decurg din aceeași mulțime de premise  $\Gamma$ , atunci din  $\Gamma$  decurge și o contradicție,  $\perp$ . O mulțime  $\Gamma$  din care decurge o contradicție se numește și mulțime *inconsistență* de formule.

A treia regulă indică că, dacă  $\Gamma$  este o mulțime inconsistentă de formule, atunci orice formulă  $\varphi$  decurge din  $\Gamma$ .

Nu există nicio regulă pentru introducerea conectorului  $\perp$  (sau, regula de eliminare a negației se poate considera ca fiind și regula de introducere a lui  $\perp$ ).

**Exemplul 124.** Să arătăm că secvența  $\{P(a)\} \vdash \neg\neg P(a)$  este validă:

1.  $\{P(a), \neg P(a)\} \vdash P(a);$  (IPOTEZĂ)
2.  $\{P(a), \neg P(a)\} \vdash \neg P(a);$  (IPOTEZĂ)
3.  $\{P(a), \neg P(a)\} \vdash \perp;$  ( $\neg e$ , 1, 2)
4.  $\{P(a)\} \vdash \neg\neg P(a).$  ( $\neg i$ , 3)

**Exemplul 125.** Să arătăm că secvența  $\{P(a), \neg P(a)\} \vdash P(b)$  este validă:

1.  $\{P(a), \neg P(a)\} \vdash P(a);$  (IPOTEZĂ)
2.  $\{P(a), \neg P(a)\} \vdash \neg P(a);$  (IPOTEZĂ)
3.  $\{P(a), \neg P(a)\} \vdash \perp;$  ( $\neg e$ , 1, 2)
4.  $\{P(a), \neg P(a)\} \vdash P(b).$  ( $\perp e$ , 3)

### Eliminarea dublei negații

La logica propozițională am întâlnit și următoarea regulă pentru eliminarea dublei negații:

$$\neg\neg e \frac{\Gamma \vdash \neg\neg \varphi}{\Gamma \vdash \varphi}$$

**Exemplul 126.** Să arătăm că secvența  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a)\} \vdash P(a)$  este validă:

1.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a), \neg P(a)\} \vdash \neg P(a);$  (IPOTEZĂ)
2.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a), \neg P(a)\} \vdash (\neg P(a) \rightarrow Q(a));$  (IPOTEZĂ)
3.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a), \neg P(a)\} \vdash Q(a);$  ( $\rightarrow e$ , 2, 1)
4.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a), \neg P(a)\} \vdash \neg Q(a);$  (IPOTEZĂ)
5.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a), \neg P(a)\} \vdash \perp;$  ( $\neg i$ , 4, 3)
6.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a)\} \vdash \neg\neg P(a);$  ( $\neg i$ , 5)
7.  $\{(\neg P(a) \rightarrow Q(a)), \neg Q(a)\} \vdash P(a).$  ( $\neg\neg e$ , 6)

**Exemplul 127.** Să arătăm că secvența  $\{\} \vdash (\text{P(a)} \vee \neg\text{P(a)})$  este validă:

1.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)}), \text{P(a)}\} \vdash \neg(\text{P(a)} \vee \neg\text{P(a)});$  (IPOTEZĂ)
2.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)}), \text{P(a)}\} \vdash \text{P(a)};$  (IPOTEZĂ)
3.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)}), \text{P(a)}\} \vdash (\text{P(a)} \vee \neg\text{P(a)});$  ( $\vee i_1, 2$ )
4.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)}), \text{P(a)}\} \vdash \perp;$  ( $\neg e, 1, 3$ )
5.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)})\} \vdash \neg\text{P(a)};$  ( $\neg i, 4$ )
6.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)})\} \vdash (\text{P(a)} \vee \neg\text{P(a)});$  ( $\vee i_2, 5$ )
7.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)})\} \vdash \neg(\text{P(a)} \vee \neg\text{P(a)});$  (IPOTEZĂ)
8.  $\{\neg(\text{P(a)} \vee \neg\text{P(a)})\} \vdash \perp;$  ( $\neg e, 7, 6$ )
9.  $\{\} \vdash \neg\neg(\text{P(a)} \vee \neg\text{P(a)});$  ( $\neg i, 8$ )
10.  $\{\} \vdash (\text{P(a)} \vee \neg\text{P(a)}).$  ( $\neg\neg e, 9$ )

### 5.6.5 Eliminarea cuantificatorului universal.

Regula pentru eliminarea cuantificatorului universal este:

$$\forall e \frac{\Gamma \vdash (\forall x.\varphi)}{\Gamma \vdash \varphi[x \mapsto t]} \quad vars(t) \cap bound(\varphi) = \emptyset$$

Regula de eliminare a cuantificatorului universal este foarte simplă: practic, dacă știm că  $(\forall x.\varphi)$  este o consecință sintactică din  $\Gamma$  atunci putem instanția variabila legată  $x$  cu orice termen  $t$  cu condiția ca  $t$  să nu conțină variabile legate în  $\varphi$ .

**Exercițiul 128.** Întrebare: regula de eliminare de mai sus mai are sens dacă  $x$  nu apare în  $\varphi$ ? De exemplu, din  $\Gamma \vdash (\forall x.\text{P(a)})$  putem deduce că  $\Gamma \vdash \text{P(a)}[x \mapsto b]$ ?

**Exemplul 129.** Să ne amintim un exemplu discutat anterior în care aveam două afirmații: Orice om este muritor și Socrate este om. Putem trage concluzia că Socrate este muritor? Pentru a răspunde la întrebare, am putea încerca să demonstrăm secvența

$$\{\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x)), \text{Om}(s)\} \vdash \text{Muritor}(s),$$

unde  $\text{Om}$  și  $\text{Muritor}$  sunt predicate de aritate 1 iar  $s$  este o constantă (simbol funcțional de aritate 0) asociată numelui Socrate. Iată demonstrația formală a secvenței:

1.  $\{\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x)), \text{Om}(s)\} \vdash \forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x))$  (IPOTEZĂ)
2.  $\{\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x)), \text{Om}(s)\} \vdash (\text{Om}(s) \rightarrow \text{Muritor}(s))$  ( $\forall e, 1, s$ )
3.  $\{\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x)), \text{Om}(s)\} \vdash \text{Om}(s)$  (IPOTEZĂ)
4.  $\{\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x)), \text{Om}(s)\} \vdash \text{Muritor}(s)$  ( $\rightarrow e, 2, 3$ )

*La pasul 2 al demonstrației am utilizat regula  $\forall e$ , care instanțiază în formula  $\forall x.(\text{Om}(x) \rightarrow \text{Muritor}(x))$  variabila legată  $x$  cu  $s$ :  $(\text{Om}(s) \rightarrow \text{Muritor}(s))$ . În limbaj natural, am dedus prin rationament sintactic că dacă Socrate este om și Orice om este muritor atunci Socrate este muritor.*

### 5.6.6 Introducerea cuantificatorul existențial.

Există o oarecare dualitate a regulilor pentru introducerea și eliminarea cuantificatorilor. Astfel, regula de introducere a cuantificatorului existențial de mai jos poate să se vadă ca duala regulii de eliminare a cuantificatorului universal:

$$\exists i \frac{\Gamma \vdash \varphi[x \mapsto t]}{\Gamma \vdash (\exists x.\varphi)} \quad \text{vars}(t) \cap \text{bound}(\varphi) = \emptyset$$

Regula ne indică faptul că putem deduce  $(\exists x.\varphi)$  atunci când  $\varphi[x \mapsto t]$  este consecință semantică din  $\Gamma$ . Informal, dacă există un  $x$  concret – și anume  $t$  – astfel încât  $\varphi[x \mapsto t]$  este adeverată (cu condiția ca  $t$  să nu conțină variabile legate în  $\varphi$ ) vom trage concluzia că  $(\exists x.\varphi)$  este adeverată. Aici,  $t$  joacă rolul unui *mărtor* pentru care formula din concluzie este adeverată.

**Exemplul 130.** Să arătăm că secvența  $\{P(a)\} \vdash \exists x.P(x)$  este validă:

1.  $\{P(a)\} \vdash P(a)$  (IPOTEZĂ)
2.  $\{P(a)\} \vdash \exists x.P(x)$  ( $\exists i, 1$ )

*Observați că în acest caz, metavariabila  $\varphi$  este  $P(x)$  din regula  $\exists_e$ , iar  $\varphi[x \mapsto a]$  este  $P(x)[x \mapsto a]$ , adică  $P(a)$ .*

**Exemplul 131.** Să arătăm că secvența  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash \exists x.Q(x)$  este validă:

1.  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash \forall x.(P(x) \rightarrow Q(x))$  (IPOTEZĂ)
2.  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash P(a)$  (IPOTEZĂ)
3.  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash (P(a) \rightarrow Q(a))$  ( $\forall e, 1, a$ )
4.  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash Q(a)$  ( $\rightarrow e, 3, 2$ )
5.  $\{\forall x.(P(x) \rightarrow Q(x)), P(a)\} \vdash \exists x.Q(x)$  ( $\exists i, 4$ )

### 5.6.7 Introducerea cuantificatorului universal.

Regula de introducere a cuantificatorului universal este:

$$\forall i \frac{\Gamma \vdash \varphi[x \mapsto x_0]}{\Gamma \vdash (\forall x.\varphi)} \quad x_0 \notin vars(\Gamma, \varphi)$$

Regula de mai sus ne spune că vom putea deriva concluzia  $\Gamma \vdash (\forall x.\varphi)$  dacă vom arăta că  $\varphi[x \mapsto x_0]$  este consecință sintactică din  $\Gamma$ , unde  $x_0$  este o variabilă nouă: ea nu mai apare în alte formule și asupra ei nu avem nici o constrângere.

**Exemplul 132.** Să arătăm că secvența  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash \forall x.Q(x)$  este validă:

1.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash \forall x.(P(x) \rightarrow Q(x))$  (IPOTEZĂ)
2.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash \forall x.P(x)$  (IPOTEZĂ)
3.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash (P(x_0) \rightarrow Q(x_0))$  ( $\forall e, 1, x_0$ )
4.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash P(x_0)$  ( $\forall e, 2, x_0$ )
5.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash Q(x_0)$  ( $\rightarrow e, 3, 4$ )
6.  $\{\forall x.(P(x) \rightarrow Q(x)), \forall x.P(x)\} \vdash \forall x.Q(x)$  ( $\forall i, 5$ )

Observați că pentru secvențele 3, 4 și 5 utilizăm variabila nouă  $x_0$ . La nivel intuitiv,  $Q(x_0)$  va avea loc pentru orice  $x_0$ .

**Exercițiul 133.** Arătați că următoarele secvențe sunt valide:

1.  $\{\forall x.(P(x) \wedge Q(x))\} \vdash \forall x.P(x);$
2.  $\{\forall x.Q(x), P(a)\} \vdash P(a) \wedge Q(a);$
3.  $\{\forall x.P(x), \forall x.Q(x)\} \vdash \forall x.(P(x) \wedge Q(x)).$

### 5.6.8 Eliminarea cuantificatorului existențial

Regula pentru eliminarea cuantificatorului existențial este următoarea:

$$\exists e \frac{\Gamma \vdash (\exists x.\varphi) \quad \Gamma \cup \{\varphi[x \mapsto x_0]\} \vdash \psi}{\Gamma \vdash \psi} \quad x_0 \notin vars(\Gamma, \varphi, \psi)$$

Prima ipoteză a regulii este  $\Gamma \vdash (\exists x.\varphi)$ , care, la nivel intuitiv, ne asigură că există cel puțin un termen (pot fi mai mulți) care îl poate înlocui  $x$  astfel încât  $\varphi$  este consecință sintactică din  $\Gamma$ . Nu știm însă care sunt acești termeni (în cazul în care sunt mai mulți). Știm doar că dacă unul există și îi vom nota generic cu  $x_0$ . Pentru a demonstra concluzia, adică  $\psi$  este consecință sintactică din  $\Gamma$ , va trebui să facem o analiză de cazuri după toți  $x_0$ . Practic, acest lucru este sumarizat de cea de-a doua ipoteză a regulii unde trebuie arătat că  $\psi$  este consecință sintactică din  $\Gamma \cup \{\varphi[x \mapsto x_0]\}$ .

**Exemplul 134.** Să arătăm că secvența  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x)\} \vdash \exists x.Q(x)$  este validă:

1.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x)\} \vdash \exists x.P(x)$  (IPOTEZĂ)
2.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x), P(x_0)\} \vdash P(x_0)$  (IPOTEZĂ)
3.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x), P(x_0)\} \vdash \forall x.(P(x) \rightarrow Q(x))$  (IPOTEZĂ)
4.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x), P(x_0)\} \vdash (P(x_0) \rightarrow Q(x_0))$  ( $\forall e, 3, x_0$ )
5.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x), P(x_0)\} \vdash Q(x_0)$  ( $\rightarrow e, 4, 2$ )
6.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x), P(x_0)\} \vdash \exists x.Q(x)$  ( $\exists i, 5$ )
7.  $\{\forall x.(P(x) \rightarrow Q(x)), \exists x.P(x)\} \vdash \exists x.Q(x)$  ( $\exists e, 1, 6$ )

Observați că pentru a demonstra secvența 7, am utilizat secvența 1 și secvența 6. Aceasta din urmă a fost demonstrată de pașii 2, 3, 4 și 5, unde am utilizat ca ipoteză și formula  $P(x_0) (= P(x)[x \mapsto x_0])$ .

### 5.6.9 Alte reguli

O altă regulă utilă, care nu ține de un anumit conector, este regula de extindere, care a fost prezentă și în capitolul dedicat logicii propoziționale:

$$\text{EXTINDERE } \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi}$$

Această regulă ne indică faptul că, dacă  $\varphi$  este consecință a unei multimi de formule  $\Gamma$ , atunci  $\varphi$  este consecință și a multimii  $\Gamma \cup \{\varphi'\}$  (indiferent de  $\varphi'$ ). Cu alte cuvinte, putem extinde oricând multimea de premise ale unei secvențe valide și obținem o nouă secvență validă.

**Exemplul 135.** Arătăm că secvența  $\{P(a), \neg Q(a), P(f(a)), (P(b) \wedge Q(b))\} \vdash \neg \neg P(a)$  este validă:

1.  $\{P(a), \neg P(a)\} \vdash P(a);$  (IPOTEZĂ)

2.  $\{P(a), \neg P(a)\} \vdash \neg P(a);$  (IPOTEZĂ)

3.  $\{P(a), \neg P(a)\} \vdash \perp;$  ( $\neg e$ , 1, 2)

4.  $\{P(a)\} \vdash \neg \neg P(a);$  ( $\neg i$ , 3)

5.  $\{P(a), \neg Q(a)\} \vdash \neg \neg P(a);$  (EXTINDERE, 4)

6.  $\{P(a), \neg Q(a), P(f(a))\} \vdash \neg \neg P(a);$  (EXTINDERE, 5)

7.  $\{P(a), \neg Q(a), P(f(a)), (P(b) \wedge Q(b))\} \vdash \neg \neg P(a).$  (EXTINDERE, 6)

## 5.7 Sistemul deductiv al deducției naturale

Deducrea naturală pentru logica de ordinul I este sistemul deductiv alcătuit din toate regulile din secțiunile precedente. Iată aici sumarizate toate regulile:

$$\begin{array}{c}
 \wedge i \frac{\Gamma \vdash \varphi_1 \quad \Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}, \quad \wedge e_1 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_1}, \\[10pt]
 \wedge e_2 \frac{\Gamma \vdash (\varphi_1 \wedge \varphi_2)}{\Gamma \vdash \varphi_2}, \quad \rightarrow e \frac{\Gamma \vdash (\varphi_1 \rightarrow \varphi_2) \quad \Gamma \vdash \varphi_1}{\Gamma \vdash \varphi_2}, \\[10pt]
 \rightarrow i \frac{\Gamma, \varphi_1 \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \rightarrow \varphi_2)}, \quad \vee i_1 \frac{\Gamma \vdash \varphi_1}{\Gamma \vdash (\varphi_1 \vee \varphi_2)}, \quad \vee i_2 \frac{\Gamma \vdash \varphi_2}{\Gamma \vdash (\varphi_1 \vee \varphi_2)}, \\[10pt]
 \vee e \frac{\Gamma \vdash (\varphi_1 \vee \varphi_2) \quad \Gamma, \varphi_1 \vdash \varphi' \quad \Gamma, \varphi_2 \vdash \varphi'}{\Gamma \vdash \varphi'}, \\[10pt]
 \neg e \frac{\Gamma \vdash \varphi \quad \Gamma \vdash \neg \varphi}{\Gamma \vdash \perp}, \quad \neg i \frac{\Gamma, \varphi \vdash \perp}{\Gamma \vdash \neg \varphi}, \quad \perp e \frac{\Gamma \vdash \perp}{\Gamma \vdash \varphi}, \\[10pt]
 \text{IPOTEZĂ } \frac{}{\Gamma \vdash \varphi} \varphi \in \Gamma, \quad \text{EXTINDERE } \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi}, \quad \neg \neg e \frac{\Gamma \vdash \neg \neg \varphi}{\Gamma \vdash \varphi}. \\[10pt]
 \forall e \frac{\Gamma \vdash (\forall x. \varphi)}{\Gamma \vdash \varphi[x \mapsto t]} \text{ vars}(t) \cap \text{bound}(\varphi) = \emptyset \\[10pt]
 \exists i \frac{\Gamma \vdash \varphi[x \mapsto t]}{\Gamma \vdash (\exists x. \varphi)} \text{ vars}(t) \cap \text{bound}(\varphi) = \emptyset \\[10pt]
 \forall i \frac{\Gamma \vdash \varphi[x \mapsto x_0]}{\Gamma \vdash (\forall x. \varphi)} x_0 \notin \text{vars}(\Gamma, \varphi) \\[10pt]
 \exists e \frac{\Gamma \vdash (\exists x. \varphi) \quad \Gamma \cup \{\varphi[x \mapsto x_0]\} \vdash \psi}{\Gamma \vdash \psi} x_0 \notin \text{vars}(\Gamma, \varphi, \psi)
 \end{array}$$

Desigur că putem folosi în demonstrații și regulile derivează (pe care le-am prezentat în cazul logicii propoziționale).

## 5.8 Corectitudinea și completitudinea deducției naturale pentru logica de ordinul I

**Teorema 136** (Corectitudinea deducției naturale). *Pentru orice multime de formule  $\Gamma$  și orice formulă  $\varphi$ , dacă secvența  $\Gamma \vdash \varphi$  este validă, atunci  $\Gamma \models \varphi$ .*

**Exercițiul 137.** Demonstrați Teorema 136.

**Teorema 138** (Completitudinea deducției naturale). *Pentru orice multime de formule  $\Gamma$  și orice formulă  $\varphi$ , dacă  $\Gamma \models \varphi$  atunci secvența  $\Gamma \vdash \varphi$  este validă.*

Demonstrația teoremei de completitudine depășește nivelul cursului.

**Observație.** De remarcat că, folosind teoremele de corectitudine și respectiv de completitudine, relația  $\vdash$  coincide cu relația  $\models$ , deși au definiții cu totul diferite.

## 5.9 Fișă de exerciții

**Exercițiul 139.** Arătați că următoarele secvențe sunt valide:

1.  $\{((P(a) \wedge Q(a)) \wedge \forall x.P(x))\} \vdash (Q(a) \wedge \forall x.P(x));$
2.  $\{((P(a) \wedge Q(a)) \wedge \forall x.P(x)), \forall x.Q(x)\} \vdash (\forall x.Q(x) \wedge Q(a));$
3.  $\{((P(a) \wedge Q(a)) \wedge \forall x.P(x))\} \vdash (\forall x.P(x) \wedge (Q(a) \wedge P(a)));$
4.  $\{((P(a) \wedge Q(a)) \rightarrow \forall x.P(x)), P(a), Q(a)\} \vdash \forall x.P(x);$
5.  $\{(P(a) \rightarrow \forall x.P(x)), P(a), Q(a)\} \vdash (Q(a) \wedge \forall x.P(x));$
6.  $\{(P(a) \rightarrow P(b)), (Q(a) \rightarrow P(b))\} \vdash ((P(a) \vee Q(a)) \rightarrow P(b));$
7.  $\{\neg(P(a) \wedge Q(a))\} \vdash (\neg P(a) \vee \neg Q(a));$
8.  $\{\neg(\neg P(a) \vee \neg Q(a))\} \vdash (P(a) \wedge Q(a));$
9.  $\{\neg(\neg P(a) \wedge \neg Q(a))\} \vdash (P(a) \vee Q(a));$

**Exercițiul 140.** Stabiliți care dintre secvențele de mai jos sunt valide:

1.  $\{\forall x.(P(x) \wedge Q(x))\} \vdash \forall x.P(x);$
2.  $\{\forall x.Q(x), P(a)\} \vdash (P(a) \wedge Q(a));$
3.  $\{\forall x.P(x), \forall x.Q(x)\} \vdash \forall x.(P(x) \wedge Q(x));$
4.  $\{\exists x.\exists y.P(x, y)\} \vdash \exists y.\exists x.P(x, y);$
5.  $\{\exists x.\forall y.P(x, y)\} \vdash \forall y.\exists x.P(x, y);$  Dar invers:  $\{\forall y.\exists x.P(x, y)\} \vdash \exists x.\forall y.P(x, y)$ ?
6.  $\{\neg(\exists x.P(x))\} \vdash \forall x.\neg P(x);$
7.  $\{\forall x.\neg P(x)\} \vdash \neg(\exists x.P(x));$

# Capitolul 6

## Forme normale ale formulelor de ordinul I

În acest capitol vom defini noțiunile de formule echivalente și cea de formă normală Prenex (FNP), formă normală Skolem (FNS) și formă normală Skolem Clauzală (FNSC) corespunzătoare unei formule din logica de ordinul I.

### 6.1 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 același înțeles este  $\neg(\forall x.Q(x))$  și  $(\exists x.\neg Q(x))$ . 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 141.** Două formule  $\varphi_1 \in \mathbb{L}^{\mathbb{P}I}$  și  $\varphi_2 \in \mathbb{L}^{\mathbb{P}I}$  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 142.** Continuăm exemplele din cursurile anterioare. Considerăm *signatura*  $\Sigma = (\{\text{P}\}, \{\text{f}, \text{i}, \text{e}\})$  și  $\Sigma$ -*structura*  $S_1 = (\mathbb{Z}, \{=\}, \{+, -, 0\})$ .

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

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

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

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

2. Avem că  $\text{P}(\text{x}_1, \text{x}_3) \stackrel{S_1}{\not\equiv} \text{P}(\text{x}_2, \text{x}_3)$ . De ce?

Deoarece există o  $S_1$ -atribuire  $\alpha : \mathcal{X} \rightarrow \mathbb{Z}$ , definită prin  $\alpha(\text{x}_1) = 42$ ,  $\alpha(\text{x}_2) = 7$ ,  $\alpha(\text{x}_3) = 42$  (pentru restul variabilelor nu este relevantă valoarea lor în atribuire) cu proprietatea că

$S_1, \alpha \models \text{P}(\text{x}_1, \text{x}_3)$  (deoarece  $42 = 42$ ), dar  
 $S_1, \alpha \not\models \text{P}(\text{x}_2, \text{x}_3)$  (deoarece  $42 \neq 7$ ).

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

**Definiția 143.** Două formule  $\varphi_1 \in \mathbb{LPI}$  și  $\varphi_2 \in \mathbb{LPI}$  sunt echivalente dacă, pentru orice structură  $S$  și pentru orice  $S$ -atribuire  $\alpha$ ,

$S, \alpha \models \varphi_1$  ddacă  $S, \alpha \models \varphi_2$ .

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

**Exemplul 144.** Continuăm exemplul anterior. Avem că  $\text{P}(\text{x}, \text{y}) \not\equiv \text{P}(\text{y}, \text{x})$ . De ce?

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

Fie structura  $S_5 = (\mathbb{Z}, \{\langle\}, \{+, -, 0\})$  definită în cursul anterior și  $S_5$ -atribuirea  $\alpha_6 : \mathcal{X} \rightarrow \mathbb{Z}$ , definită prin  $\alpha_6(\textcolor{red}{x}) = 2, \alpha_6(\textcolor{blue}{y}) = 3$  și  $\alpha_6(z) = 1$  pentru orice  $z \in \mathcal{X} \setminus \{\textcolor{red}{x}, \textcolor{blue}{y}\}$ .

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

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

**Exemplul 145.** Avem că  $(\forall \textcolor{red}{x}. \textcolor{blue}{P}(\textcolor{red}{x}, z)) \equiv (\forall \textcolor{blue}{y}. \textcolor{blue}{P}(\textcolor{blue}{y}, z))$ . De ce?

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

$$\begin{aligned} S, \alpha \models (\forall \textcolor{red}{x}. \textcolor{blue}{P}(\textcolor{red}{x}, z)) && \text{ddacă} \\ \text{pentru orice } u \in D, \text{ avem } S, \alpha[\textcolor{red}{x} \mapsto u] \models \textcolor{blue}{P}(\textcolor{red}{x}, z) && \text{ddacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(\overline{\alpha[\textcolor{red}{x} \mapsto u]}(\textcolor{red}{x}), \overline{\alpha[\textcolor{red}{x} \mapsto u]}(z)\right) && \text{ddacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(u, \alpha(z)\right) && \text{ddacă} \\ \text{pentru orice } u \in D, \text{ avem } P^S\left(\overline{\alpha[\textcolor{blue}{y} \mapsto u]}(\textcolor{blue}{y}), \overline{\alpha[\textcolor{blue}{y} \mapsto u]}(z)\right) && \text{ddacă} \\ \text{pentru orice } u \in D, \text{ avem } S, \alpha[\textcolor{blue}{y} \mapsto u] \models \textcolor{blue}{P}(\textcolor{blue}{y}, z) && \text{ddacă} \\ S, \alpha \models (\forall \textcolor{blue}{y}. \textcolor{blue}{P}(\textcolor{blue}{y}, z)). && \end{aligned}$$

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

$$S, \alpha \models (\forall \textcolor{red}{x}. \textcolor{blue}{P}(\textcolor{red}{x}, z)) \text{ ddacă } S, \alpha \models (\forall \textcolor{blue}{y}. \textcolor{blue}{P}(\textcolor{blue}{y}, z)),$$

care este chiar definiția faptului că  $(\forall \textcolor{red}{x}. \textcolor{blue}{P}(\textcolor{red}{x}, z)) \equiv (\forall \textcolor{blue}{y}. \textcolor{blue}{P}(\textcolor{blue}{y}, z))$ .

## 6.2 Forme normale și Substituții

Pentru a defini forme normale pentru formulele din logica de ordinul I, reamintim noțiunea de *substituție* :

- O *substituție* este o funcție  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$ , cu proprietatea că  $\sigma(x) \neq x$  pentru un număr finit de variabile  $x \in \mathcal{X}$ ;
- *Domeniul substituției*  $\sigma$  este multimea  $\text{dom}(\sigma) = \{x \in \mathcal{X} \mid \sigma(x) \neq x\}$ ;
- *Extensia* substituției  $\sigma$  la multimea termenilor este funcția  $\sigma^\sharp : \mathcal{T} \rightarrow \mathcal{T}$ , 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.  $\sigma^\sharp(f(t_1, \dots, t_n)) = f(\sigma^\sharp(t_1), \dots, \sigma^\sharp(t_n))$ , pentru orice simbol funcțional  $f \in \mathcal{F}_n$  de aritate  $n \in \mathbb{N}$  și orice termeni  $t_1, \dots, t_n \in \mathcal{T}$ .

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 apariție a unei variabile  $x$  din  $t$  este înlocuită cu termenul  $\sigma(x)$ ).

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

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

- *Restricția substituției  $\sigma$  la multimea  $V$  este o nouă substituție notată  $\sigma|_V : \mathcal{X} \rightarrow \mathcal{T}$ , 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 multime de variabile, se scot celelalte variabile din domeniul substituției.

- *Extensia lui  $\sigma$  la multimea formulelor este funcția  $\sigma^\flat : \mathbb{LPI} \rightarrow \mathbb{LPI}$ , definită astfel:*

1.  $\sigma^\flat(P(t_1, \dots, t_n)) = P(\sigma^\sharp(t_1), \dots, \sigma^\sharp(t_n))$ ;
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.  $\sigma^\flat((\varphi_1 \rightarrow \varphi_2)) = (\sigma^\flat(\varphi_1) \rightarrow \sigma^\flat(\varphi_2))$ ;
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  $\rho = \sigma|_{\text{dom}(\sigma) \setminus \{\textcolor{red}{x}\}}$ ;
8.  $\sigma^\flat((\exists x.\varphi)) = (\exists x.(\rho^\flat(\varphi)))$ , unde  $\rho = \sigma|_{\text{dom}(\sigma) \setminus \{\textcolor{red}{x}\}}$ .

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

### 6.3 Forma normală Prenex

**Definiția 146.** O formulă  $\varphi$  este în formă normală prenex dacă

$$\varphi = Q_1 x_1. Q_2 x_2. \dots. Q_n x_n. \varphi',$$

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 147.** 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 surprins de următoarea teoremă:

**Teorema 148.** Pentru orice formulă  $\varphi \in \mathbb{LPI}$ , există  $\varphi' \in \mathbb{LPI}$  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 următoarele rezultate:

**Teorema 149** (Teorema de înlocuire). Fie  $\varphi, \varphi' \in \mathbb{LPI}$  astfel încât  $\varphi \equiv \varphi'$  și  $\varphi_1 \in PL_1$  ce conține  $\varphi$  ca subformulă.

Fie formula  $\varphi_2$  obținută din  $\varphi_1$  prin înlocuirea unei apariții a lui  $\varphi$  cu  $\varphi'$ .

Atunci  $\varphi_1 \equiv \varphi_2$ .

**Lema 150** (Lema redenumirii). Fie  $\varphi \in \mathbb{LPI}$  o formulă și  $x, y \in \mathcal{X}$  două variabile cu proprietatea că  $y \notin \text{free}(\varphi)$ .

Atunci au loc echivalențele:

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

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

Cu alte cuvinte, conform lemei redenumirii, în formula  $\forall x.\varphi$  putem înlocui cuantificatorul  $\forall m$  (respectiv  $\exists m$ ) cu un cuantificator  $\forall y$  (respectiv  $\exists y$ ) la alegere cu condiția ca  $y$  să nu fie variabilă a formulei  $\varphi$  și să obținem o formulă echivalentă. De asemenea, aparițiile libere ale variabilei  $x$  în  $\varphi$  trebuie înlocuite cu  $y$  prin aplicarea substituției  $\sigma = \{x \mapsto y\}$  asupra formulei  $\varphi$ .

**Exemplul 151.** Fie formula  $(\forall x.P(x, y))$ . Deoarece  $z \notin \text{free}(P(x, y))$ , 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  $y \in \text{free}((\forall x.P(x, y)))$  și nici nu are loc).

Suntem acum pregătiți să prezintăm, schițat, demonstrația Teoremei 148.

**Demonstrație:** Se aplică Teorema 149 (de inlocuire) în care sunt folosite Lema 150 (redenumirii) și următoarele echivalențe, de la stânga la dreapta:

1.  $((\forall x.\varphi_1) \wedge \varphi_2) \equiv \forall x.(\varphi_1 \wedge \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
2.  $((\forall x.\varphi_1) \vee \varphi_2) \equiv \forall x.(\varphi_1 \vee \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
3.  $((\exists x.\varphi_1) \wedge \varphi_2) \equiv \exists x.(\varphi_1 \wedge \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
4.  $((\exists x.\varphi_1) \vee \varphi_2) \equiv \exists x.(\varphi_1 \vee \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
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  $x \notin \text{free}(\varphi_2)$ , 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:

7.  $(\varphi_1 \wedge \varphi_2) \equiv (\varphi_2 \wedge \varphi_1)$ ;
8.  $(\varphi_1 \vee \varphi_2) \equiv (\varphi_2 \vee \varphi_1)$ .

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 apropiati de rădăcina arborelui.

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

9.  $(\varphi_1 \rightarrow \varphi_2) \equiv \neg\varphi_1 \vee \varphi_2$ ;
10.  $(\varphi_1 \leftrightarrow \varphi_2) \equiv ((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1))$ .

q.e.d.

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

**Exemplul 152.** Fie formula  $\varphi = \left( \left( \forall x. \neg(P(x, x) \wedge \neg \exists y. P(x, y)) \right) \wedge P(x, x) \right)$ .

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

$$\begin{aligned}
 \varphi &= \left( \forall x. \neg(P(x, x) \wedge \neg \exists y. P(x, y)) \right) \wedge P(x, x) \\
 &\stackrel{L.R.}{\equiv} \left( \forall z. \neg(P(z, z) \wedge \neg \exists y. P(z, y)) \right) \wedge P(x, x) \\
 &\stackrel{1}{\equiv} \forall z. \left( \neg(P(z, z) \wedge \neg \exists y. P(z, y)) \wedge P(x, x) \right) \\
 &\stackrel{6}{\equiv} \forall z. \left( \neg(P(z, z) \wedge \forall y. \neg P(z, y)) \wedge P(x, x) \right) \\
 &\stackrel{7(\text{comutativitate } \wedge)}{\equiv} \forall z. \left( \neg((\forall y. \neg P(z, y)) \wedge P(z, z)) \wedge P(x, x) \right) \\
 &\stackrel{1}{\equiv} \forall z. \left( \neg(\forall y. (\neg P(z, y) \wedge P(z, z))) \wedge P(x, x) \right) \\
 &\stackrel{7(\text{comutativitate } \wedge)}{\equiv} \forall z. \left( \neg(\forall y. (P(z, z) \wedge \neg P(z, y))) \wedge P(x, x) \right) \\
 &\stackrel{5}{\equiv} \forall z. \left( (\exists y. \neg(P(z, z) \wedge \neg P(z, y))) \wedge P(x, x) \right) \\
 &\stackrel{3}{\equiv} \forall z. \exists y. (\neg(P(z, z) \wedge \neg P(z, y)) \wedge P(x, x)).
 \end{aligned}$$

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  $(\forall x. \neg(P(x, x) \wedge \neg \exists y. P(x, y))) \wedge P(x, x)$ .

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

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

### 6.3.1 Fișă de exerciții

Fie  $\Sigma = (\{P, Q\}, \{f, i, e\})$  unde  $P \in \mathcal{P}_2$ ,  $Q \in \mathcal{P}_1$ ,  $f \in \mathcal{F}_2$ ,  $i \in \mathcal{F}_1$  și  $e \in \mathcal{F}_0$ .

**Exercițiul 153.** Arătați că următoarele echivalențe au loc:

1.  $P(e, x) \stackrel{S}{\equiv} P(e, f(x, x))$  unde  $S$  este  $\Sigma$ -structura  $S = (\mathbb{N}, \{<, Par\}, \{+, s, 0\})$ .
2.  $\neg \forall x. \varphi \equiv \exists x. \neg \varphi$ ;
3.  $\neg \exists x. \varphi \equiv \forall x. \neg \varphi$

4.  $((\forall x.\varphi_1) \wedge \varphi_2) \equiv \forall x.(\varphi_1 \wedge \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
5.  $((\forall x.\varphi_1) \vee \varphi_2) \equiv \forall x.(\varphi_1 \vee \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
6.  $((\exists x.\varphi_1) \wedge \varphi_2) \equiv \exists x.(\varphi_1 \wedge \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
7.  $((\exists x.\varphi_1) \vee \varphi_2) \equiv \exists x.(\varphi_1 \vee \varphi_2)$ , dacă  $x \notin \text{free}(\varphi_2)$ ;
8.  $\forall x.(P(x, x) \wedge Q(x)) \equiv (\forall x.P(x, x)) \wedge (\forall x.Q(x))$
9.  $\forall x.(P(x, x) \vee Q(x)) \not\equiv (\forall x.P(x, x)) \vee (\forall x.Q(x))$
10.  $\exists x.(P(x, x) \wedge Q(x)) \not\equiv (\exists x.P(x, x)) \wedge (\exists x.Q(x))$
11.  $\exists x.(P(x, x) \vee Q(x)) \equiv (\exists x.P(x, x)) \vee (\exists x.Q(x))$

**Exercițiul 154.** Fie substituția  $\sigma : \mathcal{X} \rightarrow \mathcal{T}$  astfel încât  $\sigma(x) = i(y)$ ,  $\sigma(y) = f(x, z)$  și  $\sigma(z) = z$  pentru  $z \in \mathcal{X} \setminus \{x, y\}$ . Aplicați substituția  $\sigma$  pentru următoarele formule:

1.  $\varphi = (\forall x.P(x, y) \rightarrow P(i(y), x))$
2.  $\varphi = P(x, y) \wedge \exists y.Q(y \rightarrow \forall x.P(x, y))$

**Exercițiul 155.** Calculați câte o FNP pentru fiecare dintre formulele:

1.  $\forall x.(P(x, y) \wedge \exists x.P(x, x))$
2.  $(\exists z.P(x, y)) \vee P(z, z)$ ;
3.  $(\exists z.P(x, z)) \wedge (\forall x.P(x, z))$ ;
4.  $(\exists z.P(x, z)) \rightarrow P(x, x)$ .
5.  $\neg(\exists x.P(x, y) \vee \forall z.P(z, z)) \wedge \exists y.P(x, y)$
6.  $\forall x.\exists y.P(x, y) \rightarrow \neg\exists x.\neg\exists y.P(x, y)$

## 6.4 Formule închise

**Definiția 156.** O formulă  $\varphi \in \mathbb{LPI}$  este închisă dacă  $\text{free}(\varphi) = \emptyset$ .

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

**Definiția 157.** O formulă care nu este închisă se numește deschisă.

**Exemplul 158.** Formula  $(\forall x.P(x, x) \wedge \exists y.P(y, x))$  este o formulă închisă deoarece  $\text{free}((\forall x.P(x, x) \wedge \exists y.P(y, x))) = \emptyset$ .

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

**Definiția 159.** Fie  $\varphi \in \mathbb{LPI}$  o formulă și  $\text{free}(\varphi) = \{x_1, \dots, x_n\}$  mulțimea variabilelor libere ale acesteia.

Formula

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

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

**Observație.** Închiderea existentială a unei formule este o formulă închisă.

**Exemplul 160.** Închiderea existentială 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 161.** Două formule  $\varphi_1 \in \mathbb{LPI}$  și  $\varphi_2 \in \mathbb{LPI}$  sunt echisatisfiabile, dacă:

1. atât  $\varphi_1$  cât și  $\varphi_2$  sunt satisfiabile; sau
2. 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 162.** Orice formulă este echisatisfiabilă cu închiderea ei existentială.

**Definiția 163.** Fie  $\varphi \in \mathbb{LPI}$  o formulă și  $\text{free}(\varphi) = \{x_1, \dots, x_n\}$  mulțimea variabilelor libere ale acesteia.

Formula

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

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

**Observație.** Închiderea universală a unei formule este o formulă închisă.

**Exemplul 164.** 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 165.** O formulă este validă dacă și numai dacă închiderea ei universală este validă.

## 6.5 Forma normală Skolem

**Definiția 166 (FNS).** O formulă  $\varphi$  este în formă normală Skolem (prescurtat, FNS) dacă

$$\varphi = \forall x_1 \dots \forall x_n. \varphi',$$

unde:

1.  $\varphi'$  nu conține cuantificatori și
2.  $\text{free}(\varphi') \subseteq \{x_1, \dots, x_n\}$ .

Cu alte cuvinte, o formulă este în formă normală Skolem dacă conține doar cuantificatori universali aflați la "începutul" formulei, iar toate aparițiile variabilelor din formulă sunt apariții legate (avem cuantificatori pentru toate variabilele ce apar în formulă).

**Observație.** O formulă aflată în FNS este obligatoriu închisă, deoarece toate variabilele libere ale formulei  $\varphi'$  sunt cuantificate universal în  $\varphi$  (datorită condiției 2), și deci nu mai pot exista variabile libere în  $\varphi$ .

**Exemplul 167.** În continuare, vom lucra peste signatura  $\Sigma = (\{\mathbf{P}, \mathbf{Q}, \mathbf{R}\}, \{\mathbf{f}, \mathbf{i}, \mathbf{e}\})$ , unde  $\mathbf{P}, \mathbf{Q}, \mathbf{R}$  sunt simboluri predicative de aritate 2, 1 și respectiv 3,  $\mathbf{f}$  și  $\mathbf{i}$  sunt simboluri funcționale de aritate 2 și respectiv 1, iar  $\mathbf{e}$  este simbol funcțional de aritate 0 (constantă).

Exemple de formule în FNS:

$$\forall x.P(x, i(e)) \quad \forall x.\forall y. (P(f(x, e), y) \wedge \neg(R(x, i(f(y, y)), e) \vee Q(y)))$$

Exemple de formule care nu sunt în FNS:

$$\exists x.P(x, x) \quad \forall x. (Q(e) \wedge \neg(Q(x) \vee Q(y))) \quad Q(e) \wedge \forall x.Q(x)$$

În cazul formulelor care nu sunt în FNS, motivele pentru care acestea nu sunt în FNS sunt urmatoarele: prima formulă conține cuantificator existențial; în cea de-a doua formulă nu avem cuantificator pentru variabila  $y$ ; iar în cea de-a treia formulă, cuantificatorul  $\forall x$  nu se află la începutul formulei.

**Teorema 168** (Teorema de aducere în FNS). Pentru orice formulă  $\varphi \in \mathbb{LPI}$ , există o formulă  $\varphi' \in \mathbb{LPI}$  astfel încât:

1.  $\varphi'$  este în formă normală Skolem;
2.  $\varphi$  și  $\varphi'$  sunt echivalentibile.

**Demonstrație:** [Schită de demonstrație]

1. Calculăm o formulă  $\varphi_1$ , aflată în FNP și echivalentă cu formula  $\varphi$  (folosind Teorema de aducere în FNP din cursul precedent);
2. Calculăm o formulă  $\varphi_2$ , închiderea existențială a lui  $\varphi_1$  ( $\varphi_2$  va fi echisatisfabilă cu  $\varphi_1$  și deci cu  $\varphi$ );
3. Aplicăm în mod repetat lema de Skolemizare pe formula  $\varphi_2$ , lemă prezentată mai jos.

Rezultatul va fi o formulă aflată în FNS, echisatisfabilă cu formula de la care am plecat.

q.e.d.

**Lema 169** (Skolem). *Fie  $\varphi = \forall x_1. \forall x_2. \dots. \forall x_k. \exists x. \varphi'$ , unde  $k \geq 0$ ,  $\varphi' \in \mathbb{LPI}$  ( $\varphi'$  poate conține alți cuantificatori). Cu alte cuvinte, există  $k$  cuantificatori universalî înapoi de primul cuantificator existențial.*

*Fie  $f \in \mathcal{F}_k$  un simbol funcțional de aritate  $k$  care nu apare în  $\varphi$  (un simbol funcțional proaspăt – engl. fresh).*

*Avem că  $\varphi$  este echisatisfabilă cu*

$$\forall x_1. \dots. \forall x_k. (\sigma^b(\varphi')),$$

*unde  $\sigma = \{x \mapsto f(x_1, \dots, x_k)\}$ .*

**Demonstrație:** [Schită de demonstrație]

Considerând că formula  $\varphi$  este peste signatura  $\Sigma = (\mathcal{P}, \mathcal{F})$ , observăm că formula nou obținută este o formulă peste signatura  $\Sigma' = (\mathcal{P}, \mathcal{F} \cup \{f\})$  ce conține, pe lângă simbolurile predicative și cele funcționale din signatura  $\Sigma$ , și simbolul funcțional  $f$  de aritate  $k$ .

*Implicația directă:*

Presupunem că există o  $\Sigma$ -structură  $S$  și o atribuire  $\alpha$  astfel încât  $S, \alpha \models \varphi$  și găsim o  $\Sigma'$ -structură  $S'$  și o atribuire  $\alpha'$  astfel încât

$$S', \alpha' \models \forall x_1. \forall x_2. \dots. \forall x_k. (\sigma^b(\varphi'))$$

Atenție! Structura  $S'$  este peste signatura  $\Sigma'$  care este mai bogată decât signatura structurii  $S$  (apare simbolul funcțional  $f$  de aritate  $k$ , care este nou). Rezultatele calculate de funcția  $f$  sunt în concordanță cu valorile alese pentru variabila  $x$  în funcție de valorile variabilelor  $x_1 \dots x_k$  (știm că există o astfel de valoare aleasă astfel încât  $\varphi$  să fie satisfăcută de  $S$  și  $\alpha$ ). În schimb, atribuirea  $\alpha'$  poate coincide cu atribuirea  $\alpha$ .

*Implicația inversă:*

Presupunem că există o  $\Sigma'$ -structură  $S'$  și o atribuire  $\alpha'$  astfel încât  $S', \alpha' \models \forall x_1. \forall x_2. \dots. \forall x_k. (\sigma^b(\varphi')).$

Găsim o  $\Sigma$ -structură  $S$  și atribuirea  $\sigma$  astfel încât  $S, \alpha \models \varphi$ . ( $S$  este obținută din  $S'$  prin eliminarea funcției  $f$ ). Din nou, atribuirea  $\sigma$  poate fi identică cu  $\sigma'$ .

q.e.d.

**Exercițiul 170.** Completați demonstrația de mai sus.

**Exemplul 171.** Calculăm o formă normală Skolem pentru formula

$$\varphi = \forall x. \exists y. \forall z. \exists z'. (P(x, y) \leftrightarrow P(z, z')).$$

Prin Lema 169, avem că  $\varphi$  este echisatisfiabilă cu

$$\varphi_1 = \forall x. \forall z. \exists z'. (P(x, g(x)) \leftrightarrow P(z, z')),$$

unde  $g$  este un simbol funcțional nou, de aritate 1.

Aplicând din nou Lema 169, avem că formula  $\varphi_1$  este echisatisfiabilă cu

$$\varphi_2 = \forall x. \forall z. (P(x, g(x)) \leftrightarrow P(z, h(x, z))),$$

unde  $h$  este un simbol funcțional nou, de aritate 2.

În concluzie,  $\varphi_2$  este în FNS și este echisatisfiabilă cu  $\varphi$ , deci este o formă normală Skolem a formulei  $\varphi$ .

**Observație.** Signatura formei normale Skolem a unei formule este mai bogată decât signature formulei de la care am plecat, din cauza adăugării simbolurilor Skolem.

## 6.6 Forma normală conjunctivă

**Definiția 172** (Literal). O formulă  $\varphi \in \mathbb{LPI}$  se numește literal dacă există un simbol predicativ  $P \in \mathcal{P}_n$  de aritate  $n \geq 0$  și  $n$  termeni  $t_1, \dots, t_n \in \mathcal{T}$  astfel încât

$$\varphi = P(t_1, \dots, t_n) \text{ sau } \varphi = \neg P(t_1, \dots, t_n).$$

Cu alte cuvinte, un literal este o formulă atomică, sau negația unei formule atomice.

**Exemplul 173.** Exemple de literali:

$$P(x, x) \quad \neg P(x, i(y)) \quad \neg R(a, b, c) \quad \neg P(x, x) \quad Q(i(x)) \quad R(a, f(x), b)$$

Exemple de formule care nu sunt literali:

$$P(x, y) \wedge P(x, y) \quad \neg \neg P(x, x) \quad \forall x. P(x, x)$$

**Definiția 174** (Clauză). O formulă  $\varphi \in \mathbb{LPI}$  se numește clauză dacă există  $n$  literali  $\varphi_1, \dots, \varphi_n \in \mathbb{FOL}$  astfel încât

$$\varphi = \varphi_1 \vee \varphi_2 \vee \dots \vee \varphi_n.$$

**Exemplul 175.** Exemple de clauze:

$$P(x, x) \vee R(x, e, y) \vee \neg P(x, f(e, y)) \quad P(x, x) \quad \square \quad P(x, x)$$

$$P(x, x) \vee R(x, f(e, e), y) \vee \neg P(x, f(e, y)) \vee \neg P(e, i(x))$$

**Observație.** Un caz particular este reprezentat de clauza vidă, notată  $\square$ , care este disjuncția a 0 literali. Clauza vidă este o formulă nesatisfiabilă.

Un alt caz particular este reprezentat de literali. Orice literal este și clauză, fiind considerat disjuncția unui singur literal.

**Definiția 176** (FNC). O formulă  $\varphi$  este în formă normală clauzală (sau, echivalent, în formă normală conjunctivă) dacă există  $n \geq 1$  clauze  $\varphi_1, \dots, \varphi_n$  astfel încât

$$\varphi = \varphi_1 \wedge \varphi_2 \wedge \dots \wedge \varphi_n.$$

**Exemplul 177.** Următoarele formule sunt în FNC:

$$(P(x, x) \vee Q(x)) \wedge (\neg P(x, y) \vee R(x, y, e))$$

$$(P(x, y) \vee Q(i(x)) \vee \neg Q(e)) \wedge (\neg P(x, x)) \wedge (\neg Q(f(z, z)) \vee R(x, z))$$

## 6.7 Forma normală Skolem clauzală

**Definiția 178.** O formulă  $\varphi$  este în formă normală Skolem clauzală (prescurtat FNSC) dacă

1.  $\varphi$  este în formă normală Skolem și

2.  $\varphi'$  este în formă normală clauzală, unde:  $\varphi = \forall x_1 \dots \forall x_n \varphi'$ , iar  $\varphi'$  nu are cuantificatori (cu alte cuvinte,  $\varphi'$  este subformula obținută din  $\varphi$  prin stergerea cuantificatorilor).

**Exemplul 179.** Exemple de formule în FNSC:

$$\forall x. \forall y. ((P(x, x) \vee \neg Q(i(x)) \wedge (P(e, y) \vee \neg Q(e)))$$

$$\forall x. \forall y. (Q(e) \wedge (\neg R(x, e, y) \vee Q(i(y))))$$

$$\forall x. \forall y. \forall z. (P(x, y) \wedge (Q(x) \vee R(x, y, z)) \wedge \neg Q(x)).$$

*Exemple de formule care nu sunt în FNSC:*

$$\exists x.Q(x)$$

$$\forall x.\left(Q(e) \wedge (\neg R(x, y, z) \vee Q(y))\right)$$

$$\forall x.\forall y.\left(Q(e) \wedge \neg(Q(x) \vee Q(y))\right)$$

*In cazul ultimelor formule, motivele pentru care acestea nu sunt in FNSC sunt următoarele:*

- $\exists x.Q(x)$  conține cuantificator existential (neacceptat în FNS)
- $\forall x.\left(Q(e) \wedge (\neg R(x, y, z) \vee Q(y))\right)$  nu este in FNS deoarece conține variabile libere (y și z)
- $\forall x.\forall y.\left(Q(e) \wedge \neg(Q(x) \vee Q(y))\right)$  este în FNS, dar formula  $\varphi' = \left(Q(e) \wedge \neg(Q(x) \vee Q(y))\right)$  nu este în formă normală conjunctivă

**Teorema 180.** Pentru orice formulă  $\varphi \in \mathbb{LPI}$  aflată în FNS există o formulă  $\varphi' \in \mathbb{LPI}$  astfel încât:

1.  $\varphi'$  este în FNSC și
2.  $\varphi \equiv \varphi'$ .

**Demonstrație:** [Schită de demonstrație]

Se aplică de la stânga la dreapta următoarele echivalențe:

1.  $\varphi_1 \leftrightarrow \varphi_2 \equiv ((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1))$
2.  $(\varphi_1 \rightarrow \varphi_2) \equiv (\neg \varphi_1 \vee \varphi_2)$
3.  $\neg \neg \varphi \equiv \varphi;$
4.  $\neg(\varphi_1 \vee \varphi_2) \equiv \neg \varphi_1 \wedge \neg \varphi_2;$
5.  $\neg(\varphi_1 \wedge \varphi_2) \equiv \neg \varphi_1 \vee \neg \varphi_2;$
6.  $\varphi_1 \vee (\varphi_2 \wedge \varphi_3) \equiv (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3).$

De asemenea, se permite folosirea liberă a asociativității și comutativității conectorilor  $\vee$  și  $\wedge$ .

Primele două echivalențe elimină orice folosire a cuantificatorilor  $\leftrightarrow$  și  $\rightarrow$ .

Următoarele trei echivalențe se asigură că negațiile pot fi aplicate doar formulelor atomice.

Ultima echivalentă se asigură că  $\vee$ -ul nu apare peste  $\wedge$ .

În final, arborele sintactic asociat formulei rezultate va avea spre rădăcină, după cuantificatorii universali,  $\wedge$ , urmat de un nivel de  $\vee$ , urmat de  $\neg$ , urmat de formule atomice, ceea ce înseamnă că formula rezultată este în FNSC.

q.e.d.

**Observație.** Un rezultat al teoremelor 168 și 180 este faptul că pentru orice formulă  $\varphi \in \mathbb{LPI}$  există o formulă  $\varphi' \in \mathbb{LPI}$  astfel încât  $\varphi'$  este în FNSC iar formulele  $\varphi$  și  $\varphi'$  sunt echisatisfiabile.

Pentru a găsi această formulă, trebuie să urmăm pașii următori:

1. Calculăm formula  $\varphi_1$  în FNP echivalentă cu  $\varphi$ ;
2. Calculăm  $\varphi_2$  închiderea existențială pentru  $\varphi_1$ ;
3. Aplicăm lema de Skolemizare pentru a elimina cuantificatorii existentiali și obținem formula  $\varphi_3$ ;
4. Aplicăm echivalențele din teorema 180 pentru a pune formula  $\varphi_3$  în FNSC.

**Exemplul 181.** Calculăm o FNSC pentru formula  $\varphi = (\forall x.P(x, z)) \wedge (\exists y.\neg P(y, z))$ .

$$\begin{aligned}\varphi &= (\forall x.P(x, z)) \wedge (\exists y.\neg P(y, z)) \\ &\equiv \forall x.(P(x, z) \wedge (\exists y.\neg P(y, z))) \\ &\equiv \forall x.\exists y.(P(x, z) \wedge \neg P(y, z))\end{aligned}$$

Așadar, o formă normală prenex (FNP) a formulei  $\varphi$  este

$$\varphi_1 = \forall x.\exists y.(P(x, z) \wedge \neg P(y, z))$$

Continuăm prin calcularea închiderii existențiale pentru  $\varphi_1$ . Aceasta este

$$\varphi_2 = \exists z.\forall x.\exists y.(P(x, z) \wedge \neg P(y, z)).$$

Conform Teoremei 162,  $\varphi_2$  este echisatisfiabil cu  $\varphi_1$ .

În continuare aplicăm lema de scolemizare (Lema 169) pentru eliminarea cuantificatorilor existentiali din  $\varphi_2$ . Astfel,  $\varphi_2$  este echisatisfiabil cu

$$\varphi_3 = \forall x.\exists y.(P(x, c) \wedge \neg P(y, c))$$

unde  $c$  este un simbol constant nou (deoarece în  $\varphi_2$ , cuantificatorul  $\exists z$  nu are alti cuantificatori universalii în față). Aplicând din nou lema de scolemizare, obținem că  $\varphi_3$  este equisatisfiabil cu

$$\varphi_4 = \forall x.(P(x, c) \wedge \neg P(g(x), c))$$

unde  $g$  este un simbol funcțional nou de aritate 1 (deoarece în  $\varphi_3$ , cuantificatorul  $\exists y$  este precedat doar de cuantificatorul universal  $\forall x$ ). Formula  $\varphi_4$  obținută este o FNSC pentru  $\varphi$  deoarece este în FNS și formula  $\varphi' = (P(x, c) \wedge \neg P(g(x), c))$  este în FNC.

### 6.7.1 Fișă de exerciții

**Exercițiul 182.** Găsiți câte o FNSC pentru următoarele formule:

1.  $\neg((\forall x.Q(x)) \rightarrow (\exists x.Q(x)))$ ;

2.  $(\exists z.P(x,z)) \wedge (\forall x.P(x,z))$ ;

# Capitolul 7

## Rezoluția de bază

Pentru a testa satisfiabilitatea unei formule aflate în FNSC, putem folosi sistemul deductiv descris în secțiunea aceasta.

**Definiția 183.** Un termen  $t \in \mathcal{T}$  se numește termen de bază dacă  $\text{vars}(t) = \emptyset$ .

În engleză, termen de bază = ground term.

**Definiția 184.** O substituție  $\sigma = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$  se numește substituție de bază dacă  $t_1, \dots, t_n$  sunt termeni de bază.

În engleză, substituție de bază = ground substitution.

**Definiția 185.** Rezoluția de bază este un sistem deductiv<sup>1</sup> pentru clauze, cu următoarea regulă de inferență, numită rezoluție de bază:

$$\frac{C_1 \vee P(t_1, \dots, t_n) \quad C_2 \vee \neg P(t'_1, \dots, t'_n)}{\sigma_1^b(C_1) \vee \sigma_2^b(C_2)} \quad \begin{array}{l} \sigma_1 \text{ E SUBST. DE BAZĂ} \\ \sigma_2 \text{ E SUBST. DE BAZĂ} \\ \sigma_1^\sharp(t_i) = \sigma_2^\sharp(t'_i) \text{ PENTRU ORICE } 1 \leq i \leq n \end{array}$$

**Teorema 186** (Teorema rezoluției de bază). O formulă  $\varphi$  aflată în FNSC este nesatisfiabilă dacă și numai dacă se poate obține  $\square$  prin rezoluție de bază pornind de la clauzele din  $\varphi$ .

**Exemplul 187.** Continuam exemplul 181 pentru a arata ca formula

$$\varphi = (\forall x. P(x, z)) \wedge (\exists y. \neg P(y, z))$$

nu este satisfiabilă. În Exemplul 181 am calculat o FNSC

$$\varphi_4 = \forall x. (P(x, c) \wedge \neg P(g(x), c))$$

<sup>1</sup>De fapt, sistemul deductiv mai conține o regulă, numită factorizare, pe care o vom vedea mai târziu

pentru formula  $\varphi$ . Avem că  $\varphi_4$  este equisatisfiabilă cu  $\varphi$ .

Vom arata folosind rezolutia de baza că formula  $\varphi_4$  este nesatisfiabilă.

$$1. \quad P(x, c)$$

$$2. \quad \neg P(g(x), c)$$

$$3. \quad \square (1, 2, \sigma_1 = \{x \mapsto g(e)\}, \sigma_2 = \{x \mapsto e\}, \sigma_1^b(P(x, c)) = \sigma_2^b(P(g(x), c))).$$

Deoarece am ajuns la clauza vidă, conform Teoremei 186, avem că  $\varphi_4$  este nesatisfiabilă. Dar  $\varphi$  este equisatisfiabilă cu  $\varphi_4$ . Așadar și  $\varphi$  este nesatisfiabilă.

## 7.1 Un exemplu

Ne interesează să stabilim validitatea formulei

$$\varphi = (\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \rightarrow (\forall x. (Q(x) \rightarrow Q(i(i(x)))))$$

Este ușor de văzut că o formulă este validă dacă și numai dacă negatia ei este nesatisfiabilă. Astfel încât, pentru a stabili că  $\varphi$  este validă, este suficient să aratăm că

$$\neg\varphi = \neg((\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \rightarrow (\forall x. (Q(x) \rightarrow Q(i(i(x)))))$$

este nesatisfiabilă.

Vom calcula o FNSC a formulei  $\neg\varphi$ , FNSC despre care știm că este echisatisfiabilă cu formula  $\neg\varphi$ . Primul pas este să găsim o FNP:

$$\begin{aligned} \neg\varphi &= \neg((\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \rightarrow (\forall x. (Q(x) \rightarrow Q(i(i(x))))) \\ &\equiv \neg(\neg(\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \vee (\forall x. (Q(x) \rightarrow Q(i(i(x))))) \\ &\equiv (\neg\neg\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \wedge (\neg\forall x. (Q(x) \rightarrow Q(i(i(x))))) \\ &\equiv (\forall x. (Q(x) \leftrightarrow \neg Q(i(x)))) \wedge (\exists x. \neg(Q(x) \rightarrow Q(i(i(x))))) \\ &\equiv \forall x. ((Q(x) \leftrightarrow \neg Q(i(x))) \wedge \exists x. \neg(Q(x) \rightarrow Q(i(i(x))))) \\ &\equiv \forall x. ((Q(x) \leftrightarrow \neg Q(i(x))) \wedge \exists x'. \neg(Q(x') \rightarrow Q(i(i(x'))))) \\ &\equiv \forall x. \exists x'. ((Q(x) \leftrightarrow \neg Q(i(x))) \wedge \neg(Q(x') \rightarrow Q(i(i(x'))))) \\ &\equiv \forall x. \exists x'. ((Q(x) \rightarrow \neg Q(i(x))) \wedge (\neg Q(i(x)) \rightarrow Q(x)) \wedge \neg(Q(x') \rightarrow Q(i(i(x'))))) \\ &\equiv \forall x. \exists x'. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(x') \vee Q(i(i(x'))))) \end{aligned}$$

Așadar, o formă normală prenex a formulei  $\neg\varphi$  este

$$\varphi_1 = \forall x. \exists x'. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(x') \vee Q(i(i(x')))))$$

Continuăm prin găsirea unei FNS a formulei  $\varphi_1$ , prin aplicarea lemei de skolemizare, și folosind un simbol Skolem proaspăt  $g$ , de aritate 1:

$$\varphi_1 = \forall x. \exists x'. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(g(x)) \vee Q(i(i(g(x))))))$$

echisatisfiabilă cu

$$\forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(g(x)) \vee Q(i(i(g(x))))))$$

Așadar, o FNS a formulei  $\neg\varphi$  este formula

$$\varphi_2 = \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(g(x)) \vee Q(i(i(g(x))))))$$

Continuăm, pentru a găsi o FNSC a formulei  $\varphi_2$ :

$$\begin{aligned} \varphi_2 &= \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge \neg(\neg Q(g(x)) \vee Q(i(i(g(x)))))) \\ &\equiv \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (\neg\neg Q(i(x)) \vee Q(x)) \wedge (\neg\neg Q(g(x)) \wedge \neg Q(i(i(g(x)))))) \\ &\equiv \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (Q(i(x)) \vee Q(x)) \wedge (Q(g(x)) \wedge \neg Q(i(i(g(x)))))) \\ &\equiv \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (Q(i(x)) \vee Q(x)) \wedge (Q(g(x)) \wedge (\neg Q(i(i(g(x))))))) \end{aligned}$$

Am obținut astăzi formula

$$\varphi_3 = \forall x. ((\neg Q(x) \vee \neg Q(i(x))) \wedge (Q(i(x)) \vee Q(x)) \wedge (Q(g(x)) \wedge (\neg Q(i(i(g(x))))))$$

care este o FNSC pentru formula  $\neg\varphi$ .

Vom arăta, folosind rezoluția de bază, că formula  $\varphi_3$  este nesatisfiabilă folosind rezoluția de bază:

1.  $\neg Q(x) \vee \neg Q(i(x))$
2.  $Q(i(x)) \vee Q(x)$
3.  $Q(g(x))$
4.  $\neg Q(i(i(g(x))))$
5.  $\neg Q(i(g(e)))$  (3,1, $\sigma_1 = \{x \mapsto e\}, \sigma_2 = \{x \mapsto g(e)\}, \sigma_1^b(Q(g(x))) = \sigma_2^b(Q(x))$ );

6.  $\mathbb{Q}(\text{i}(\text{i}(g(e))))$  ( $2, 5, \sigma_1 = \{\text{x} \mapsto \text{i}(g(\text{e}))\}, \sigma_2 = \{\}, \sigma_1^\flat(\mathbb{Q}(\text{x})) = \sigma_2^\flat(\mathbb{Q}(\text{i}(g(\text{e}))))$ );
7.  $\square (6, 4, \sigma_1 = \{\}, \sigma_2 = \{\text{x} \mapsto \text{e}\}, \sigma_1^\flat(\mathbb{Q}(\text{i}(\text{i}(g(\text{e})))))) = \sigma_2^\flat(\mathbb{Q}(\text{i}(\text{i}(g(\text{x})))))$ .

Deoarece am ajuns la clauza vidă, concluzionăm prin Teorema 186 că formula  $\forall \text{x}. ((\neg \mathbb{Q}(\text{x}) \vee \neg \mathbb{Q}(\text{i}(\text{x}))) \wedge (\mathbb{Q}(\text{i}(\text{x})) \vee \mathbb{Q}(\text{x})) \wedge (\mathbb{Q}(g(\text{x}))) \wedge (\neg \mathbb{Q}(\text{i}(\text{i}(g(\text{x}))))))$  este nesatisfiabilă.

Dar formula de mai sus este o FNSC a formulei  $\neg \varphi$ , deci cele două sunt echisatisfiabile. Deci  $\neg \varphi$  este de asemenea nesatisfiabilă. Deci formula cu care am pornit,

$$\varphi = (\forall \text{x}. (\mathbb{Q}(\text{x}) \leftrightarrow \neg \mathbb{Q}(\text{i}(\text{x})))) \rightarrow (\forall \text{x}. (\mathbb{Q}(\text{x}) \rightarrow \mathbb{Q}(\text{i}(\text{i}(\text{x}))))))$$

este validă.

## 7.2 Fișă de exerciții

**Exercițiul 188.** Arătați că următoarele formule aflate în FNSC sunt nesatisfiabile, folosind rezoluția de bază:

1.  $\forall \text{x}. \forall \text{y}. \forall \text{z}. ((\neg P(\text{x}, \text{z}) \vee R(\text{x}, \text{x}, \text{z})) \wedge (\neg R(\text{e}, \text{x}, \text{e})) \wedge (P(\text{e}, \text{y})))$ ;
2.  $\forall \text{x}. \forall \text{y}. ((\neg P(\text{x}, \text{y}) \vee Q(\text{x}) \vee Q(\text{y})) \wedge (\neg Q(\text{i}(\text{i}(\text{e})))) \wedge (P(\text{i}(\text{x}), \text{i}(\text{x}))))$ .

**Exercițiul 189.** Stabiliți prin rezoluție de bază că următoarele formule sunt valide:

1.  $(\forall \text{x}. Q(\text{x})) \rightarrow (\exists \text{x}. Q(\text{x}))$ ;
2.  $((\forall \text{x}. \forall \text{y}. \forall \text{z}. (P(\text{x}, \text{y}) \wedge P(\text{y}, \text{z}) \rightarrow P(\text{x}, \text{z}))) \wedge P(\text{x}, \text{y}) \wedge P(\text{y}, \text{x})) \rightarrow P(\text{x}, \text{x})$ ;
3.  $(\neg \forall \text{x}. Q(\text{x})) \leftrightarrow (\exists \text{x}. \neg Q(\text{x}))$ ;
4.  $(\neg \exists \text{x}. Q(\text{x})) \leftrightarrow (\forall \text{x}. \neg Q(\text{x}))$ ;
5.  $(\exists \text{y}. \forall \text{x}. P(\text{x}, \text{y})) \rightarrow (\forall \text{x}. \exists \text{y}. P(\text{x}, \text{y}))$ ;
6.  $(\forall \text{x}. (P(\text{x}, \text{x}) \leftrightarrow Q(\text{x}))) \rightarrow (P(\text{e}, \text{e}) \rightarrow Q(\text{e}))$ .

# Capitolul 8

## Rezoluția pentru LP1

Rezoluția pentru LP1 este o metodă de demonstrare a nesatisfiabilității unei formule de ordinul I aflată în FNSC. Avantajul rezoluției este că este făcută mecanic, printr-o metodă inventată de Robinson în anii 1960 numită unificare; aşadar este mai ușor pretabilă spre a fi mecanizată (implementată într-un program pe calculator).

### 8.1 Unificare

**Definiția 190** (Unificator). O substituție  $\sigma$  este unificator al termenilor  $t_1$  și  $t_2$  dacă  $\sigma^\sharp(t_1) = \sigma^\sharp(t_2)$ .

**Exemplul 191.** Fie termenii  $t_1 = f(x, h(y))$  și  $t_2 = f(h(z), z')$ . Un unificator pentru  $t_1$  și  $t_2$  este:

$$\sigma = \{z \mapsto a, x \mapsto h(a), z' \mapsto h(y)\} \quad (\sigma^\sharp(t_1) = f(h(a), h(y)) = \sigma^\sharp(t_2)).$$

Un alt unificator al celor doi termeni este:

$$\sigma_1 = \{x \mapsto h(z), z' \mapsto h(y)\} \quad (\sigma_1^\sharp(t_1) = f(h(z), h(y)) = \sigma_1^\sharp(t_2)).$$

**Definiția 192** (Termeni unificabili). Doi termeni sunt unificabili dacă au cel puțin un unificator.

**Exemplul 193.** Termenii  $t_1 = f(x, y)$  și  $t_2 = h(z)$  nu au unificator, deci nu sunt unificabili. De ce?

Pentru orice substituție  $\sigma$  avem  $\sigma^\sharp(t_1) = \sigma^\sharp(f(x, y)) = f(\sigma^\sharp(x), \sigma^\sharp(y)) \neq h(\sigma^\sharp(z)) = \sigma^\sharp(h(z)) = \sigma^\sharp(t_2)$  (intuitiv, orice substituție am aplica peste cei doi

termeni, în primul caz rezultatul are ca funcție principală  $\mathbf{f}$ , iar în al doilea caz avem  $\mathbf{h}$ ).

Termenii  $t_1 = \mathbf{x}$  și  $t_2 = \mathbf{h}(\mathbf{x})$  nu au unificator. De ce?

Să presupunem că ar exista un unificator al lor,  $\sigma$ . Din moment ce  $\sigma^\sharp(t_1) = \sigma^\sharp(t_2)$ , înseamnă că arborii abstracti corespunzatori termenilor  $\sigma^\sharp(t_1)$  și  $\sigma^\sharp(t_2)$  au același număr de noduri. Notăm cu  $\text{noduri}(t)$  numărul de noduri ale arborelui corespunzător unui termen  $t$ .

Avem că  $\text{noduri}(\sigma^\sharp(t_2)) = \text{noduri}(\sigma^\sharp(\mathbf{h}(\mathbf{x}))) = 1 + \text{noduri}(\sigma^\sharp(\mathbf{x})) = 1 + \text{noduri}(\sigma^\sharp(t_1)) > \text{noduri}(\sigma^\sharp(t_1))$ , ceea ce reprezintă o contradicție, deoarece trebuie să avem  $\text{noduri}(\sigma^\sharp(t_2)) = \text{noduri}(\sigma^\sharp(t_1))$ . Prin urmare, presupusul unificator  $\sigma$  nu există.

**Definiția 194** (Compunerea a două substituții). Fie  $\sigma_1, \sigma_2$  două substituții. Substituția  $\sigma_2 \circ \sigma_1 : \mathcal{X} \rightarrow \mathcal{T}$ , denumită compunerea substituțiilor  $\sigma_1$  și  $\sigma_2$ , este definită astfel:

- $(\sigma_2 \circ \sigma_1)(x) = \sigma_2^\sharp(\sigma_1(x))$ , pentru orice  $x \in \mathcal{X}$ .

**Exercițiul 195.** Verificați că funcția  $\sigma_2 \circ \sigma_1$  este într-adevăr o substituție (adică că mulțimea acelor variabile  $x$  cu proprietatea că  $(\sigma_2 \circ \sigma_1)(x) \neq x$  este finită).

**Exemplul 196.** Continuând exemplul anterior, fie substituțiile  $\sigma = \{\mathbf{z} \mapsto \mathbf{a}, \mathbf{x} \mapsto \mathbf{h}(\mathbf{a}), \mathbf{z}' \mapsto \mathbf{h}(\mathbf{y})\}$ ,  $\sigma_1 = \{\mathbf{x} \mapsto \mathbf{h}(\mathbf{z}), \mathbf{z}' \mapsto \mathbf{h}(\mathbf{y})\}$  și respectiv  $\sigma_2 = \{\mathbf{z} \mapsto \mathbf{a}\}$ . Avem că  $\sigma = \sigma_2 \circ \sigma_1$  deoarece:

$$\begin{aligned} \sigma_2 \circ \sigma_1(\mathbf{x}) &= \sigma_2^\sharp(\sigma_1(\mathbf{x})) = \sigma_2^\sharp(\mathbf{h}(\mathbf{z})) = \mathbf{h}(\mathbf{a}) = \sigma(\mathbf{x}) \\ \sigma_2 \circ \sigma_1(\mathbf{z}) &= \sigma_2^\sharp(\sigma_1(\mathbf{z})) = \sigma_2^\sharp(\mathbf{z}) = \mathbf{a} = \sigma(\mathbf{z}) \\ \sigma_2 \circ \sigma_1(\mathbf{z}') &= \sigma_2^\sharp(\sigma_1(\mathbf{z}')) = \sigma_2^\sharp(\mathbf{h}(\mathbf{y})) = \mathbf{h}(\mathbf{y}) = \sigma(\mathbf{z}') \\ \sigma_2 \circ \sigma_1(y') &= \sigma_2^\sharp(\sigma_1(y')) = \sigma_2^\sharp(y') = y' = \sigma(y'), \text{ for any } y' \in \mathcal{X} \setminus \{x, z, z'\} \end{aligned}$$

**Definiția 197** (Substituție mai generală). O substituție  $\sigma_1$  este mai generală decât o substituție  $\sigma$  dacă  $\sigma$  se poate obține prin compunerea substituției  $\sigma_1$  cu o altă substituție  $\sigma_2$ :  $\sigma = \sigma_2 \circ \sigma_1$ .

**Exemplul 198.** De exemplu,  $\sigma_1 = \{\mathbf{x} \mapsto \mathbf{h}(\mathbf{z}), \mathbf{z}' \mapsto \mathbf{h}(\mathbf{y})\}$  este mai generală decât  $\sigma = \{\mathbf{z} \mapsto \mathbf{a}, \mathbf{x} \mapsto \mathbf{h}(\mathbf{a}), \mathbf{z}' \mapsto \mathbf{h}(\mathbf{y})\}$ , deoarece  $\sigma = \sigma_2 \circ \sigma_1$ , unde  $\sigma_2$  este definită în exemplul de mai sus.

## 8.2 Cel mai general unificator

**Definiția 199** (Cel mai general unificator). O substituție  $\sigma$  este cel mai general unificator al termenilor  $t_1$  și  $t_2$  dacă:

1.  $\sigma$  este unificator al termenilor  $t_1, t_2$  și
2.  $\sigma$  este o substituție mai generală decât orice unificator al  $t_1, t_2$ .

**Exemplul 200.** Fie  $t_1 = f(x, a)$  și  $t_2 = f(y, a)$ . Unificatorul  $\{y \mapsto x\}$  este mai general decât  $\{x \mapsto a, y \mapsto a\}$  și orice alt unificator al celor doi termeni.

**Exemplul 201.** Substituția  $\sigma_1 = \{x \mapsto h(z), z' \mapsto h(y)\}$  este cel mai general unificator al termenilor  $t_1 = f(x, h(y))$  și  $t_2 = f(h(z), z')$ .

**Teorema 202** (Teorema existenței celui mai general unificator). Orice doi termeni unificabili au un cel mai general unificator.

**Observație.** În general, cel mai general unificator nu este unic.

**Exemplul 203.** Un unificator pentru termenii  $h(x)$  și  $h(y)$  este substituția  $\{x \mapsto a, y \mapsto a\}$  (dar nu este cel mai general unificator).

Un cel mai general unificator este  $\{x \mapsto y\}$ . Un alt cel mai general unificator este  $\{y \mapsto x\}$ .

În continuare, vom prezenta un algoritm pentru calculul unui cel mai general unificator.

În acest scop, avem nevoie de generalizarea noțiunii de unificare pentru mai multe perechi de termeni.

### 8.3 Problemă de unificare

**Definiția 204** (Problemă de unificare). O problemă de unificare  $\mathbb{P}$  este:

- sau o mulțime

$$\mathbb{P} = \{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\}$$

formată din  $n$  perechi de termeni

- sau simbolul special

$$\mathbb{P} = \perp \text{ (citat bottom).}$$

**Definiția 205** (Soluție a unei probleme de unificare). O substituție  $\sigma$  este o soluție a unei probleme de unificare  $\mathbb{P}$  dacă:

1. problema este de forma  $\mathbb{P} = \{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\}$  și
2.  $\sigma$  este unificator pentru  $t_i$  și  $t'_i$ , pentru orice  $i \in \{1, \dots, n\}$ .

**Definiția 206** (Mulțimea soluțiilor unei probleme de unificare). Cu  $\text{unif}(\mathbb{P})$  notăm mulțimea soluțiilor unei probleme de unificare  $\mathbb{P}$ :

$$\text{unif}(\mathbb{P}) = \{\sigma \mid \sigma \text{ este soluție a problemei } \mathbb{P}\}.$$

**Observație.** Prin definiția noțiunii de soluție a unei probleme de unificare, dacă  $\mathbb{P} = \perp$ , atunci  $\text{unif}(\mathbb{P}) = \emptyset$ .

**Exemplul 207.** Fie  $\mathbb{P} = \{\mathbf{f(x, a)} \doteq \mathbf{f(y, a)}\}$ . Avem că  $\text{unif}(\mathbb{P}) = \{\{\mathbf{x} \mapsto \mathbf{z}, \mathbf{y} \mapsto \mathbf{z}\}, \{\mathbf{x} \mapsto \mathbf{y}\}, \dots\}$ .

**Definiția 208** (Cea mai generală soluție). Substituția  $\sigma$  este cea mai generală soluție pentru o problemă de unificare  $\mathbb{P} = \{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\}$  dacă:

1.  $\sigma$  este soluție pentru  $\mathbb{P}$ :  $\sigma^\sharp(t_i) = \sigma^\sharp(t'_i)$ , pentru orice  $1 \leq i \leq n$ ;

2.  $\sigma$  este mai generală decât orice altă soluție pentru  $P$ .

**Observație.** Observați că în cazul în care  $\text{unif}(\mathbb{P}) \neq \emptyset$  (problema de unificare are soluții), atunci există cel puțin o cea mai generală soluție pentru  $\mathbb{P}$ .

**Notatie.** Cu  $\text{mgu}(\mathbb{P})$  notăm o cea mai generală soluție a problemei de unificare  $P$  (dacă problema  $\mathbb{P}$  are soluție).

Cu  $\text{mgu}(t_1, t_2)$  notăm un cel mai general unificator al termenilor  $t_1, t_2$  (dacă termenii sunt unificabili).

**Observație.**  $\text{mgu}(t_1, t_2) = \text{mgu}(\{t_1 \doteq t_2\})$ .

**Definiția 209** (Formă rezolvată). O problemă de unificare  $\mathbb{P}$  este în formă rezolvată dacă  $\mathbb{P} = \perp$  sau  $\mathbb{P} = \{x_1 \doteq t'_1, \dots, x_n \doteq t'_n\}$  și  $x_i \notin \text{vars}(t_j)$  pentru orice  $i, j \in \{1, \dots, n\}$ .

**Exemplul 210.** Următoarele probleme de unificare sunt în formă rezolvată:

- $\mathbb{P}_1 = \{\mathbf{x}_3 \doteq \mathbf{f(g(a, a), a)}, \mathbf{x}_2 \doteq \mathbf{a}, \mathbf{x}_1 \doteq \mathbf{a}\}$

- $\mathbb{P}_2 = \perp$

Următoarele probleme de unificare NU sunt în formă rezolvată:

- $\mathbb{P}_3 = \{\mathbf{f(x, a)} \doteq \mathbf{f(y, a)}\}$  (deoarecec în "perechea"  $\mathbf{f(x, a)} \doteq \mathbf{f(y, a)}$  termenul din stânga nu este o variabilă)

- $\mathbb{P}_4 = \{\mathbf{x}_1 \doteq \mathbf{f(x}_2, \mathbf{x}_3), \mathbf{x}_2 \doteq \mathbf{x}_3\}$  (deoarece, chiar dacă avem doar variabile în partea stângă, există variabilă  $x_2$  în partea stângă  $a \doteq$  și apare și în partea dreaptă  $a \doteq$ )

De ce este utilă forma rezolvată a problemelor de unificare?

**Lema 211.** Dacă  $\mathbb{P} = \{x_1 \doteq t'_1, \dots, x_n \doteq t'_n\}$  este în formă rezolvată, atunci  $\{x_1 \mapsto t'_1, \dots, x_n \mapsto t'_n\}$  este cea mai generală soluție a problemei  $\mathbb{P}$ .

Următoarele reguli pot fi folosite pentru aducerea unei probleme de unificare în formă rezolvată:

STERGERE

$$\mathbb{P} \cup \{t \doteq t\} \Rightarrow \mathbb{P}$$

DESCOMPUNERE

$$\mathbb{P} \cup \{f(t_1, \dots, t_n) \doteq f(t'_1, \dots, t'_n)\} \Rightarrow \\ \mathbb{P} \cup \{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\}$$

ORIENTARE

$$\mathbb{P} \cup \{f(t_1, \dots, t_n) \doteq x\} \Rightarrow \mathbb{P} \cup \{x \doteq f(t_1, \dots, t_n)\}$$

ELIMINARE

$$\mathbb{P} \cup \{x \doteq t\} \Rightarrow \sigma^\sharp(\mathbb{P}) \cup \{x \doteq t\}$$

dacă  $x \notin vars(t), x \in vars(\mathbb{P})$  (unde  $\sigma = \{x \mapsto t\}$ )

CONFLICT

$$\mathbb{P} \cup \{f(t_1, \dots, t_n) \doteq g(t'_1, \dots, t'_m)\} \Rightarrow \perp$$

dacă  $f \neq g$

OCCURS CHECK

$$\mathbb{P} \cup \{x \doteq f(t_1, \dots, t_n)\} \Rightarrow \perp$$

dacă  $x \in vars(f(t_1, \dots, t_n))$

Transformările de mai sus au următoarele proprietăți:

**Lema 212** (Progres). *Dacă  $\mathbb{P}$  nu este în formă rezolvată, atunci există  $\mathbb{P}'$  astfel încât  $\mathbb{P} \Rightarrow \mathbb{P}'$ .*

**Lema 213** (Păstrarea soluțiilor). *Dacă  $\mathbb{P} \Rightarrow \mathbb{P}'$ , atunci  $unif(\mathbb{P}) = unif(\mathbb{P}')$ .*

**Lema 214** (Terminare). *Nu există o secvență infinită  $\mathbb{P} \Rightarrow \mathbb{P}_1 \Rightarrow \mathbb{P}_2 \Rightarrow \dots \Rightarrow \mathbb{P}_i \Rightarrow \dots$*

**Corolarul 215.** *Regulile precedente constituie un algoritm de calcul al unei cele mai generale soluții pentru o problemă de unificare, dacă aceasta există.*

**Exemplul 216.**

$$\begin{aligned}
 \mathbb{P} = \{f(g(x_1, a), x_2) \doteq x_3, f(x_2, x_2) \doteq f(a, x_1)\} &\xrightarrow{\text{DESCOMPUNERE}} \\
 \{f(g(x_1, a), x_2) \doteq x_3, x_2 \doteq a, x_2 \doteq x_1\} &\xrightarrow{\text{ELIMINARE}} \\
 \{f(g(x_1, a), a) \doteq x_3, x_2 \doteq a, a \doteq x_1\} &\xrightarrow{\text{ORIENTARE}} \\
 \{f(g(x_1, a), a) \doteq x_3, x_2 \doteq a, x_1 \doteq a\} &\xrightarrow{\text{ELIMINARE}} \\
 \{f(g(a, a), a) \doteq x_3, x_2 \doteq a, x_1 \doteq a\} &\xrightarrow{\text{ORIENTARE}} \\
 \{x_3 \doteq f(g(a, a), a), x_2 \doteq a, x_1 \doteq a\}.
 \end{aligned}$$

*Concluzie:*  $\{x_3 \mapsto f(g(a, a), a), x_2 \mapsto a, x_1 \mapsto a\}$  este cea mai generală soluție a problemei de unificare inițiale.

**Exemplul 217.**

$$\mathbb{P} = \{f(g(x_1, a), x_2) \doteq x_3, f'(x_2) \doteq f'(x_3)\} \xrightarrow{\text{DESCOMPUNERE}}$$

$$\{f(g(x_1, a), x_2) \doteq x_3, x_2 \doteq x_3\} \xrightarrow{\text{ORIENTARE}}$$

$$\{x_3 \doteq f(g(x_1, a), x_2), x_2 \doteq x_3\} \xrightarrow{\text{ELIMINARE}}$$

*Explicati de ce nu se mai poate aplica orientare*

$$\{x_3 \doteq f(g(x_1, a), x_3), x_2 \doteq x_3\} \xrightarrow{\text{OCCURS CHECK}}$$

$\perp$ .

*Concluzie:*  $\text{unif}(\mathbb{P}) = \emptyset$ .

**Exemplul 218.**

$$\begin{aligned}
 \mathbb{P} = \{f(g(x_1, a), x_2) \doteq x_3, f(g(x_4, x_5)) \doteq f(x_3)\} &\xrightarrow{\text{DESCOMPUNERE}} \\
 \{f(g(x_1, a), x_2) \doteq x_3, g(x_4, x_5) \doteq x_3\} &\xrightarrow{\text{ORIENTARE}} \\
 \{f(g(x_1, a), x_2) \doteq x_3, x_3 \doteq g(x_4, x_5)\} &\xrightarrow{\text{ELIMINARE}} \\
 \{f(g(x_1, a), x_2) \doteq g(x_4, x_5), x_3 \doteq g(x_4, x_5)\} &\xrightarrow{\text{CONFLICT}} \\
 &\perp.
 \end{aligned}$$

Concluzie:  $\text{unif}(\mathbb{P}) = \emptyset$ .

## 8.4 Rezoluție de ordinul I

Rezoluția pentru logica de ordinul I este un sistem deductiv pentru clauze alcătuit din următoarele două reguli de inferență:

$$\text{REZOLUȚIE BINARĂ } \frac{P(t_1, \dots, t_n) \vee C_1 \quad \neg P(t'_1, \dots, t'_n) \vee C_2}{\sigma^b(C_1 \vee C_2)} \quad V_1 \cap V_2 = \emptyset \quad \sigma = \text{mgu}(\{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\})$$

unde  $V_1 = \text{vars}(P(t_1, \dots, t_n) \vee C_1)$  și  $V_2 = \text{vars}(\neg P(t'_1, \dots, t'_n) \vee C_2)$ .

$$\text{FACTORIZARE POZITIVĂ } \frac{P(t_1, \dots, t_n) \vee P(t'_1, \dots, t'_n) \vee C}{\sigma^b(P(t_1, \dots, t_n) \vee C)} \quad \sigma = \text{mgu}(\{t_1 \doteq t'_1, \dots, t_n \doteq t'_n\})$$

**Observație.** • În cazul în care clauzele care reprezintă ipotezele regulii REZOLUȚIE BINARĂ au variabile în comun ( $V_1 \cap V_2 \neq \emptyset$ ), variabilele uneia dintre clauze trebuie redenumite înainte de a aplica regula (vezi exemplul de mai jos);

- În cazul în care problema de unificare care apare în regula de rezoluție nu are soluție, regula nu poate fi aplicată.
- Regula de factorizare pozitivă are o singură ipoteză.
- În cazul în care problema de unificare care apare în regula de factorizare nu are soluție, regula nu poate fi aplicată.

- Regula de factorizare pozitivă este necesară pentru completitudine (vezi exerciții seminar).

**Teorema 219** (Teorema rezoluției). O formulă  $\varphi = \forall x_1 \dots \forall x_n. (C_1 \wedge C_2 \wedge \dots \wedge C_m)$ , aflată în FNSC, este nesatisfiabilă dacă și numai dacă  $\square$  se poate obține din clauzele  $C_1, \dots, C_m$ , aplicând regulile REZOLUȚIE BINARĂ și FACTORIZARE POZITIVĂ.

**Exemplul 220.** Să demonstrăm că  $\forall x. (P(x) \wedge (\neg P(h(x)) \vee Q(f(x))) \wedge (\neg Q(f(g(a)))))$  este nesatisfiabilă, prin rezoluție de ordinul I:

$$1. P(x)$$

$$2. \neg P(h(x)) \vee Q(f(x))$$

$$3. \neg Q(f(g(a)))$$

$$4. Q(f(x)) \quad \text{rezoluție binară între 1 și 2:}$$

$$\frac{P(x') \quad \neg P(h(x)) \vee Q(f(x))}{\sigma^b(Q(f(x)))} \sigma = \{x' \mapsto h(x)\} = mgu(\{x' \doteq h(x)\})$$

$$5. \square \quad \text{rezoluție între 4 și 3:}$$

$$\frac{Q(f(x)) \quad \neg Q(f(g(a)))}{\sigma^b(\square)} \sigma = \{x \mapsto g(a)\} = mgu(\{f(x) \doteq f(g(a))\})$$

**Exemplul 221.** Ne propunem să demonstrăm că  $\varphi = (\forall x. (P(x) \rightarrow Q(x))) \wedge P(e) \rightarrow Q(e)$  este validă folosind rezoluția pentru LPII.

Formula  $\varphi$  este validă dacă și numai dacă  $\neg\varphi$  este nesatisfiabilă. Pentru aceasta, trebuie să punem formula  $\neg\varphi$  în FNSC. Primul pas este de a pune formula  $\neg\varphi$  în FNP:

$$\begin{aligned} \neg\varphi &= \neg((\forall x. (P(x) \rightarrow Q(x)) \wedge P(e)) \rightarrow Q(e)) \\ &\equiv \neg(\neg(\forall x. (P(x) \rightarrow Q(x)) \wedge P(e)) \vee Q(e)) \\ &\equiv \neg(\exists x. \neg((P(x) \rightarrow Q(x)) \wedge P(e))) \vee Q(e) \\ &\equiv \neg(\exists x. \neg((P(x) \rightarrow Q(x)) \wedge P(e))) \vee Q(e) \\ &\equiv \forall x. \neg(\neg((P(x) \rightarrow Q(x)) \wedge P(e)) \vee Q(e)) \text{ în FNP} \end{aligned}$$

Așadar formula  $\varphi_1 = \forall x. \neg(\neg((P(x) \rightarrow Q(x)) \wedge P(e)) \vee Q(e))$  este o FNP pentru  $\varphi$  și  $\varphi \equiv \varphi_1$ . Formula  $\varphi_1$  este de asemenea închisă și nu are cuantificatori existențiali și deci nu necesită calcularea închiderii existențiale sau aplicării Skolemizării. Așadar, este în FNS.

Acum trebuie să punem formula  $\varphi_1$  în FNSC:

$$\begin{aligned}\varphi_1 &= \forall x. \neg \left( \neg((P(x) \rightarrow Q(x)) \wedge P(e)) \vee Q(e) \right) \\ &\equiv \forall x. \neg \neg ((P(x) \rightarrow Q(x)) \wedge P(e)) \wedge \neg Q(e) \\ &\equiv \forall x. ((P(x) \rightarrow Q(x)) \wedge P(e)) \wedge \neg Q(e) \\ &\equiv \forall x. (\neg P(x) \vee Q(x)) \wedge P(e) \wedge \neg Q(e) \text{ in CSNF}\end{aligned}$$

Formula  $\varphi_2 = \forall x. (\neg P(x) \vee Q(x)) \wedge P(e) \wedge \neg Q(e)$  este o FNSC pentru  $\varphi_1$  și  $\varphi_2 \equiv \varphi_1$ .

Acum putem aplica rezoluția folosind clauzele din  $\varphi_2$  pentru a demonstra că aceasta nu este satisfiabilă.

1.  $\neg P(x) \vee Q(x)$

2.  $P(e)$

3.  $\neg Q(e)$

4.  $\neg P(e)$  Rezoluție binară între 1 și 3:

$$\frac{\neg P(x) \vee Q(x) \quad \neg Q(e)}{\sigma^b(\neg P(e))} \sigma = \{x \mapsto e\} = mgu(\{x \doteq e\})$$

5.  $\square$  Rezoluție binară între 2 și 4:

$$\frac{P(e) \quad \neg P(e)}{\sigma^b(\square)} \sigma = \emptyset = mgu(\{e \doteq e\})$$

Din aceasta rezultă că  $\varphi_2$  nu este satisfiabilă.

Dar,  $\varphi_2 \equiv \varphi_1$  și  $\neg\varphi \equiv \varphi_1$  și deci  $\neg\varphi \equiv \varphi_2$ . Asta înseamnă că  $\neg\varphi$  nu este satisfiabilă, și deci  $\varphi$  este validă.

## 8.5 Fișă de exerciții

**Exercițiul 222.** Rezolvați următoarele probleme de unificare:

1.  $\{f(x, y) \doteq f(y, x), g(x) \doteq g(z)\};$

2.  $\{f(x, y) \doteq f(y, x), g(x) \doteq a\};$

3.  $\{f(f(x, y), z) \doteq f(y, x), g(z) \doteq g(a)\};$

4.  $\{f(g(x), y) \doteq f(y, z), z \doteq h(a)\};$

5.  $\{x_1 \doteq f(x_2, x_2), x_2 \doteq f(x_3, x_3), x_3 \doteq f(x_4, x_4)\}.$

**Exercițiul 223.** Arătați că următoarele formule aflate în FNSC sunt nesatisfiabile, folosind rezoluția de ordinul I:

1.  $\forall x. \forall y. \forall z. ((\neg P(x, z) \vee R(x, x, z)) \wedge (\neg R(e, x, e)) \wedge (P(e, y)))$ ;
2.  $\forall x. \forall y. ((\neg P(x, y) \vee Q(x) \vee Q(y)) \wedge (\neg Q(i(i(e)))) \wedge (P(i(x), i(x))))$ .

**Exercițiul 224.** Stabiliti prin rezoluție de ordinul I că următoarele formule sunt valide:

1.  $(\forall x. (P(x) \rightarrow Q(x))) \wedge P(e) \rightarrow Q(e)$
2.  $((\forall x. \forall y. \forall z. (P(x, y) \wedge P(y, z) \rightarrow P(x, z))) \wedge P(x, y) \wedge P(y, x)) \rightarrow P(x, x)$ ;
3.  $(\forall x. Q(x)) \rightarrow (\exists x. Q(x))$ ;
4.  $(\neg \forall x. Q(x)) \leftrightarrow (\exists x. \neg Q(x))$ ;
5.  $(\neg \exists x. Q(x)) \leftrightarrow (\forall x. \neg Q(x))$ ;
6.  $(\exists y. \forall x. P(x, y)) \rightarrow (\forall x. \exists y. P(x, y))$ ;
7.  $(\forall x. (P(x, x) \leftrightarrow Q(x))) \rightarrow (P(e, e) \rightarrow Q(e))$ .