Logică pentru informatică - note de curs

Universitatea Alexandru Ioan Cuza, Iași
Facultatea de Informatică
Anul universitar 2022-2023

Ștefan Ciobâcă
Andrei Arusoaie
Rodica Condurache
Cristian Masalagiu

Cuprins

Capitolul 1

Motivație 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 $\mathtt{p} \in A$.

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

Nu, deoarece afirmației *Socrate este muritor* ar trebui să îi asociem o a treia variabilă propozițională $\mathtt{r}$ și nu putem trage nicio concluzie asupra lui $\tau(\mathtt{r})$ din faptul că $\tau(\mathtt{p}) = 1$ și $\tau(\mathtt{q}) = 1$. Deci, din semantica logicii propoziționale, nu putem trage concluzia că $\mathtt{r}$ este adevărată în orice atribuire în care $\mathtt{p}$ și $\mathtt{q}$ 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: $\mathtt{O}$ și $\mathtt{M}$. Predicatul $\mathtt{O}$ va fi definit astfel: $\mathtt{O}(\mathtt{x})$ va fi adevărat când $\mathtt{x}$ este om. Predicatul $\mathtt{M}(\mathtt{x})$ este adevărat când $\mathtt{x}$ este muritor. Deoarece predicatele de mai sus au fiecare câte un singur argument/parametru, ele se numesc predicate *unare*. Predicatele 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

$$(\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x}))),$$

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

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

Vom vedea că în logica de ordinul I, formula $\mathtt{M}(\mathtt{s})$ este consecință a formulelor $(\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x})))$ și respectiv $\mathtt{O}(\mathtt{s})$. 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:

$$\varphi = \Big(\forall \mathsf{x}.(\forall \mathsf{y}.(\mathsf{x} < \mathsf{y} \rightarrow \exists \mathsf{z}.(\mathsf{x} < \mathsf{z} \wedge \mathsf{z} < \mathsf{y})))\Big).$$

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 φ 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 $\mathsf{x}, \mathsf{y}, \mathsf{z}$ 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. $(\mathbb{N}, \{<, =\}, \{+, 0, 1\})$;

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

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

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

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

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

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

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

   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. $(\mathbb{R}, \{<\}, \emptyset)$.

   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 domeniul 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:

$$\varphi = \Big(\forall x.(\forall y.(x < y \to \exists z.(x < z \wedge z < y)))\Big),$$

avem că această formulă este adevărată în structura $(\mathbb{R}, \{<, =\}, \{+, -, 0, 1\})$ (între orice două numere reale distincte există cel puțin un număr real) dar este falsă în structura $(\mathbb{Z}, \{<, =\}, \{+, -, 0, 1\})$ (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 $<\subseteq \mathbb{R}^2$. În al doilea caz, domeniul variabilelor x, y, z este $\mathbb{Z}$ și simbolului $<$ îi corespunde predicatul $<\subseteq \mathbb{Z}^2$.

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

Definiția 2 (Signatură)**.** *O* signatură Σ *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 (Σ-structuri)**.** *Dacă* $\Sigma = (\mathcal{P}, \mathcal{F})$ *este o signatură, o* Σ-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* $\Sigma = (\{\mathtt{P}, \mathtt{Q}\}, \{\mathtt{f}, \mathtt{i}, \mathtt{a}, \mathtt{b}\})$ *unde* $\mathtt{P}, \mathtt{Q}$ *sunt simboluri predicative de aritate* $ar(\mathtt{P}) = ar(\mathtt{Q}) = 2$ *și* $\mathtt{f}, \mathtt{i}, \mathtt{a}, \mathtt{b}$ *sunt simboluri funcționale cu arități:* $ar(\mathtt{f}) = 2$, $ar(\mathtt{i}) = 1$ *și* $ar(\mathtt{a}) = ar(\mathtt{b}) = 0$.

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

Observație. *Dupa cum se poate observa și în Exemplul 4, pentru simboluri predicative (e.g.,* $\mathtt{P}$, $\mathtt{Q}$*) vom utiliza o culoare diferită față de culoarea simbolurilor funcționale (e.g.,* $\mathtt{f}, \mathtt{i}, \mathtt{a}, \mathtt{b}$*). 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 Σ îi putem asocia mai multe structuri, numite Σ-structuri.

Notație. *Mulțimea simbolurilor predicative dintr-o Σ-structură de aritate n este notată cu $\mathcal{P}_n = \{P \mid ar(P) = n\}$, iar mulțimea simbolurilor funcționale de aritate n este notată cu $\mathcal{F}_n = \{f \mid ar(f) = n\}$. Pentru cazul particular $n = 0$, $\mathcal{F}_0$ reprezintă mulțimea 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 șiruri de simboluri) este determinat de alegerea unei signaturi Σ. Practic, există mai multe limbaje de ordinul I, câte un limbaj pentru fiecare signatură Σ.

În continuare, vom presupune fixată o signatură Σ 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 șiruri de simboluri peste un anumit alfabet. Spre deosebire de logica propozițională, alfabetul este mai bogat și conține următoarele simboluri:

1. *conectori logici* deja cunoscuți: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \bot$, precum și doi *cuantificatori*: $\forall$, $\exists$;

2. *variabile*: vom fixa o mulțime infinit numărabilă de variabile, notată $\mathcal{X} = \{\mathsf{x}, \mathsf{y}, \mathsf{z}, \mathsf{x}', \mathsf{y}', \mathsf{x}_1, \mathsf{z}'', \ldots\}$ (a nu se confunda cu mulțimea variabilelor propoziționale din logica propozițională; sunt două noțiuni fundamental diferite și din acest motiv utilizăm altă culoare pentru a le reprezenta);

3. simboluri auxiliare: $($, $)$, $.$, $,$, $($, $)$, și $,$;

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. *Mulţimea* termenilor, $\mathcal{T}$, *este cea mai mică mulţime 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* $n > 0$*) şi* $t_1, \ldots, t_n \in \mathcal{T}$*, atunci* $f(t_1, \ldots, t_n) \in \mathcal{T}$ *(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* Σ*-termeni.*

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

Exemplul 6. *Fie signatura* $\Sigma = (\{\mathtt{P}, \mathtt{Q}\}, \{\mathtt{f}, \mathtt{i}, \mathtt{a}, \mathtt{b}\})$ *definită în Exemplul 4, unde* $ar(\mathtt{P}) = ar(\mathtt{Q}) = 2$*,* $ar(\mathtt{f}) = 2$*,* $ar(\mathtt{i}) = 1$*,* $ar(\mathtt{a}) = ar(\mathtt{b}) = 0$*. Iată câteva exemple de termeni:* $\mathtt{a}$*,* $\mathtt{b}$*,* $\mathtt{x}$*,* $\mathtt{y}$*,* $\mathtt{x}_1$*,* $\mathtt{y}'$*,* $\mathtt{i(a)}$*,* $\mathtt{i(x)}$*,* $\mathtt{i(i(a))}$*,* $\mathtt{i(i(x))}$*,* $\mathtt{f(a,b)}$*,* $\mathtt{i(f(a,b))}$*,* $\mathtt{f(f(x,a),f(y,y))}$*.*

Exercițiul 7. *Identificați în lista de mai jos* Σ*-termenii:*

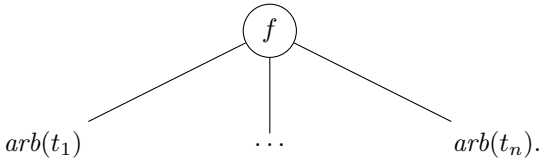
1. $\mathtt{i(i(x))}$*;*
2. $\mathtt{i}$*;*
3. $\mathtt{f(x,x)}$*;*
4. $\mathtt{P(a,b)}$*;*
5. $\mathtt{i(a,a)}$*;*
6. $\mathtt{f(i(x),i(x))}$*;*
7. $\mathtt{f(i(x,x))}$*;*
8. $\mathtt{a(i(x))}$*.*

Termenii (sau, în mod echivalent, *termii*), sunt notaţi cu t, s, t_1, t_2, s_1, t', etc. Deşi termenii sunt scrişi în mod uzual ca un şir de simboluri, ei au asociat un arbore de sintaxă abstractă definit după cum urmează:

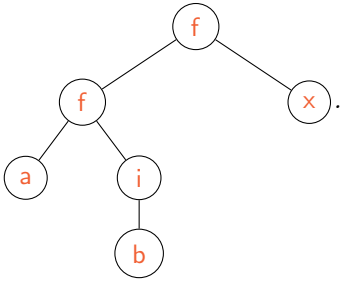
1. dacă $t = c$ și $c \in \mathcal{F}_0$, atunci $arb(t) =$ 

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

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

$arb(t) =$ 

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

$arb($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 șir de simboluri de forma* $P(t_1, \ldots, t_n)$*, unde* $P \in \mathcal{P}_n$ *este un simbol predicativ de aritate* $n \geq 0$*, iar* $t_1, \ldots, t_n \in \mathcal{T}$ *sunt termeni. Dacă* $n = 0$*, scriem* P *în loc de* $P()$*.*

Exemplul 10. *Continuând Exemplul 6, folosim signatura*

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

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

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)**.** *Mulţimea formulelor de ordinul I, notată* $\mathbb{LPI}$*, este cea mai mică mulţime astfel incât:*

1. *(cazul de bază) orice formulă atomică este formulă (adică* $P(t_1, \ldots, t_n) \in \mathbb{LPI}$ *pentru orice simbol predicativ* $P \in \mathcal{P}_n$ *și orice termeni* $t_1, \ldots, t_n \in \mathcal{T}$*;*
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) $(\varphi_1 \to \varphi_2) \in \mathbb{LPI}$*;*
   (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* $\neg, \wedge, \vee, \to$ *și respectiv* $\leftrightarrow$ *din logica propozițională. 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* $\Sigma = (\{\mathtt{P}, \mathtt{Q}\}, \{\mathtt{f}, \mathtt{i}, \mathtt{a}, \mathtt{b}\})$*, unde* $ar(\mathtt{P}) = ar(\mathtt{Q}) = 2$*,* $ar(\mathtt{f}) = 2$*,* $ar(\mathtt{i}) = 1$ *și* $ar(\mathtt{a}) = ar(\mathtt{b}) = 0$*.*

Iată câteva exemple de formule din logica de ordinul I:

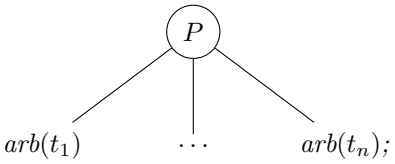
1. $\mathtt{P(a,b)}$*;*
2. $\mathtt{Q(a,b)}$*;*
3. $\mathtt{P(a,x)}$*;*
4. $\neg\mathtt{P(a,b)}$*;*
5. $(\mathtt{P(a,b)} \wedge \neg\mathtt{Q(a,b)})$*;*
6. $(\mathtt{P(a,b)} \vee \neg\mathtt{Q(x,y)})$*;*
7. $(\mathtt{P(a,b)} \to \mathtt{P(a,b)})$*;*
8. $((\mathtt{P(a,b)} \to \mathtt{P(a,b)}) \leftrightarrow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

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

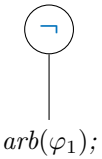
10. $(\exists x.\neg Q(x, y))$.

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

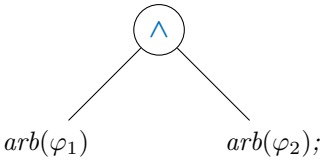
1. *dacă* $\varphi = P(t_1, \ldots, t_n)$, *atunci* $arb(\varphi) =$



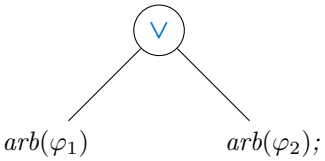
2. *dacă* $\varphi = \neg\varphi_1$, *atunci* $arb(\varphi) =$



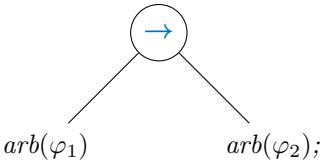
3. *dacă* $\varphi = (\varphi_1 \wedge \varphi_2)$, *atunci* $arb(\varphi) =$



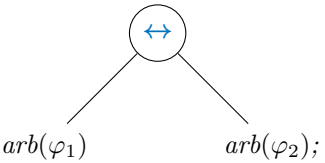
4. *dacă* $\varphi = (\varphi_1 \vee \varphi_2)$, *atunci* $arb(\varphi) =$

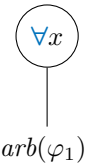


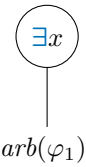
5. *dacă* $\varphi = (\varphi_1 \rightarrow \varphi_2)$, *atunci* $arb(\varphi) =$



6. *dacă* $\varphi = (\varphi_1 \leftrightarrow \varphi_2)$, *atunci* $arb(\varphi) =$



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

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

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 înţeleasă 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 *priorităţi*. Ordinea priorității operatorilor este:

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

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 $\mathtt{P(a,a)} \vee \mathtt{P(b,b)} \wedge \mathtt{P(a,b)}$ va fi întotdeauna înţeleasă ca $(\mathtt{P(a,a)} \vee (\mathtt{P(b,b)} \wedge \mathtt{P(a,c)}))$ (deoarece $\wedge$ este prioritar faţă de $\vee$). Ca analogie, la fel se întâmplă şi în cazul aritmeticii: $1 + 2 * 3$ va fi înteles ca $1 + (2 \times 3)$, deoarece $\times$ are prioritate în faţa lui $+$ ($\times$ este similar cu $\wedge$ şi $+$ 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 $\mathbb{LPI}$ 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 $\mathtt{O}$ pentru primul predicat și simbolul predicativ $\mathtt{M}$ pentru al doilea predicat. De asemenea, în text avem și o constantă: Socrate. Alegem simbolul funcțional $\mathtt{s}$ de aritate 0 pentru această constantă. Așadar, pentru a modela afirmațiile de mai sus, vom lucra cu signatura

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

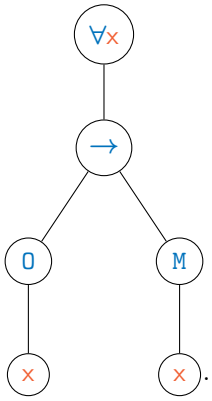
unde $\mathtt{O}$ și $\mathtt{M}$ sunt simboluri predicative de aritate $ar(\mathtt{O}) = ar(\mathtt{M}) = 1$, iar $\mathtt{s}$ este un simbol funcțional de aritate $ar(\mathtt{s}) = 0$, adică un simbol constant.

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

$$(\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x}))),$$

al cărei arbore de sintaxă abstractă este:

$arb\Big((\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x})))\Big) =$



.

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

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

1. D este mulțimea tuturor ființelor de pe Pământ;

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

Anticipând puțin (vom discuta despre semantica formulelor de ordinul I în Capitolul **??**), toate cele trei formule discutate în această secțiune, adică $(\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x})))$, $\mathtt{O}(\mathtt{s})$ și respectiv $\mathtt{M}(\mathtt{s})$, 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 $\mathtt{M}(\mathtt{s})$ este în mod necesar adevărată în *orice* structură în care formulele $\mathtt{O}(\mathtt{s})$ și $(\forall \mathtt{x}.(\mathtt{O}(\mathtt{x}) \to \mathtt{M}(\mathtt{x})))$ sunt adevărate, nu doar în structura S de mai sus.

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

Fie signatura $\Sigma = (\{<, =\}, \{+, -, 0, 1\})$, unde $<$ 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 Σ:

1. $\Big(\forall \mathtt{x}.\big(\forall \mathtt{y}.\big(<(\mathtt{x}, \mathtt{y}) \to \exists \mathtt{z}.(<(\mathtt{x}, \mathtt{z}) \wedge <(\mathtt{z}, \mathtt{y}))\big)\big)\Big)$;
2. $\Big(\forall \mathtt{x}.\big(\forall \mathtt{y}.\big(\exists \mathtt{z}.(=(+(\mathtt{x}, \mathtt{y}), \mathtt{z}))\big)\big)\Big)$;
3. $\Big(\forall \mathtt{x}.(<(0, \mathtt{x}) \vee =(0, \mathtt{x}))\Big)$;
4. $\Big(\forall \mathtt{x}.\big(\exists \mathtt{y}.(=(\mathtt{x}, -(\mathtt{y})))\big)\Big)$;
5. $=(+(\mathtt{x}, \mathtt{y}), \mathtt{z})$.

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

1. $\Big(\forall \mathtt{x}.\big(\forall \mathtt{y}.\big(\mathtt{x} < \mathtt{y} \to \exists \mathtt{z}.(\mathtt{x} < \mathtt{z} \wedge \mathtt{z} < \mathtt{y})\big)\big)\Big)$;
2. $\Big(\forall \mathtt{x}.\big(\forall \mathtt{y}.\big(\exists \mathtt{z}.(\mathtt{x} + \mathtt{y} = \mathtt{z})\big)\big)\Big)$;
3. $\Big(\forall \mathtt{x}.(0 < \mathtt{x} \vee 0 = \mathtt{x})\Big)$;

4. $\Big(\forall x.\big(\exists y.(x = -(y))\big)\Big)$;
5. $x + y = z$.

Două dintre Σ-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 φ. De exemplu, vom avea că $vars\big((\forall z.(P(x, y)))\big) = \{x, y, z\}$. Definim funcția $vars : \mathbb{LPI} \to 2^{\mathcal{X}}$ în cele ce urmează.

În primul rând, definim o funcție $vars : \mathcal{T} \to 2^{\mathcal{X}}$ (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 Σ, notăm cu $\mathcal{P}_n$ mulțimea simbolurilor predicative de aritate n din Σ, iar cu $\mathcal{F}_n$ mulțimea simbolurilor funcționale de aritate n din Σ.

Definiția 17. *Funcția $vars : \mathcal{T} \to 2^{\mathcal{X}}$ 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. *$vars(f(t_1, \ldots, t_n)) = \bigcup_{i \in \{1,\ldots,n\}} vars(t_i)$.*

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

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

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

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* φ:

$$\Big(\big(\forall \mathtt{x}.(\mathtt{P}(\mathtt{x},\mathtt{y}) \wedge \exists \mathtt{y}.(\mathtt{P}(\mathtt{z},\mathtt{f}(\mathtt{x},\mathtt{y})) \wedge \mathtt{P}(\mathtt{x},\mathtt{y})))\big) \wedge \mathtt{P}(\mathtt{x},\mathtt{x})\Big).$$

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

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

1. $\mathtt{P}(\mathtt{x},\mathtt{y})$;
2. $\mathtt{Q}(\mathtt{a},\mathtt{b})$;
3. $\mathtt{P}(\mathtt{a},\mathtt{x})$;
4. $\neg\mathtt{P}(\mathtt{x},\mathtt{z})$;
5. $(\mathtt{P}(\mathtt{x},\mathtt{x}) \wedge \neg\mathtt{Q}(\mathtt{x},\mathtt{z}))$;
6. $(\mathtt{P}(\mathtt{x},\mathtt{b}) \vee \neg\mathtt{Q}(\mathtt{z},\mathtt{y}))$;
7. $((\mathtt{P}(\mathtt{x},\mathtt{b}) \to \mathtt{P}(\mathtt{x},\mathtt{z})) \leftrightarrow (\mathtt{P}(\mathtt{x},\mathtt{b}) \to \mathtt{P}(\mathtt{a},\mathtt{z})))$;
8. $(\forall \mathtt{x}.\mathtt{P}(\mathtt{a},\mathtt{x}))$;
9. $(\exists \mathtt{x}.\neg\mathtt{Q}(\mathtt{x},\mathtt{y}))$;
10. $\Big((\exists \mathtt{x}.\neg\mathtt{Q}(\mathtt{x},\mathtt{y})) \wedge (\forall \mathtt{y}.\mathtt{P}(\mathtt{y},\mathtt{x}))\Big)$.

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 variabile `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 $\Big(\forall x.\big(\forall y.(P(x,y) \wedge P(x,z) \wedge (\exists x.P(x,y)))\big)\Big)$, 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 $\big(\forall y.(P(x,y) \wedge P(x,z) \wedge (\exists x.P(x,y)))\big)$, în timp ce domeniul de vizibilitate D_2 al variabilei x cuantificate existențial este $P(x,y)$:

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

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 (Apariţie liberă)**.** *O* apariţie liberă a unei variabile x într-o formulă φ *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 (Apariţie legată)**.** *O* apariţie legată a unei variabile x într-o formulă φ *este dată de un nod în arborele formulei etichetat cu* x *și care are proprietatea că, mergând din nod înspre rădăcină, întâlnim măcar un nod etichetat cu* $\forall x$ *sau cu* $\exists x$*.*

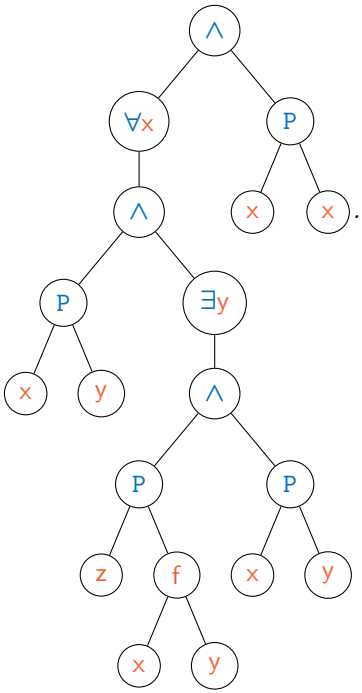
Cel mai apropiat astfel de nod etichetat cu $\forall x$ *sau cu* $\exists x$ *este cuantificarea care* leagă *aparița în cauză a variabilei* x*.*

Exemplul 23. *Considerăm în continuare formula*

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

Arborele de sintaxă abstractă al formulei φ *este:*

$arb(\varphi) =$



În formula φ 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 φ sunt marcate prin subliniere:

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

Toate aparițiile legate ale variabilelor în formula φ sunt marcate prin dublă subliniere în următoarea formulă:

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

Observație. *În restul capitolelor vom face o distincție clară între aparițiile (libere sau legate ale) variabilelor într-o formulă și mulțimile 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

Mulţimea variabilelor unei formule φ care au cel puţin o apariţie liberă se notează $free(\varphi)$.

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

1. $free(P(t_1, \ldots, t_n)) = vars(t_1) \cup \ldots \cup vars(t_n)$;
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. $free((\varphi_1 \to \varphi_2)) = free(\varphi_1) \cup free(\varphi_2)$;
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*

$$\varphi = \Big(\big(\forall \mathsf{x}.\big(\mathsf{P}(\mathsf{x},\mathsf{y}) \wedge \exists \mathsf{y}.(\mathsf{P}(\mathsf{z},\mathsf{f}(\mathsf{x},\mathsf{y})) \wedge \mathsf{P}(\mathsf{x},\mathsf{y}))\big)\big) \wedge \mathsf{P}(\mathsf{x},\mathsf{x})\Big).$$

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

Exercițiul 26. *Calculați* $free(\varphi)$ *pentru fiecare formulă* φ *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* $bound : \mathbb{LPI} \to 2^{\mathcal{X}}$ *este definită recursiv astfel:*

1. $bound(P(t_1, \ldots, t_n)) = \emptyset$;
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. $bound((\varphi_1 \to \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2)$;
6. $bound((\varphi_1 \leftrightarrow \varphi_2)) = bound(\varphi_1) \cup bound(\varphi_2)$;

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

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

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

$$\varphi = \Big(\big(\forall \mathsf{x}.\big(\mathsf{P}(\mathsf{x},\mathsf{y}) \wedge \exists \mathsf{y}.\big(\mathsf{P}(\mathsf{z},\mathsf{f}(\mathsf{x},\mathsf{y})) \wedge \mathsf{P}(\mathsf{x},\mathsf{y})\big)\big)\Big) \wedge \mathsf{P}(\mathsf{x},\mathsf{x})\Big).$$

Definiția 30. *Variabilele legate ale unei formule φ sunt elementele mulțimii $bound(\varphi)$.*

Definiția 31. *Variabilele libere ale unei formule φ sunt elementele mulțimii $free(\varphi)$.*

Observație. *Mulțimile $free(\varphi)$ şi $bound(\varphi)$ 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 un nod din arborele formulei, în timp ce variabila liberă este un element al mulțimii $\mathcal{X}$.*

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 mulțimii $\mathcal{X}$.*

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:

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

se parantezează în felul următor:

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

3.13 Fișă de exerciții

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

Ion este student. Orice student învață la Logică. Oricine învață 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 signatură* Σ *pentru structura* S *de mai sus.*
2. *Modelați următoarele afirmații ca formule de ordinul I în signatura asociată structurii* S *de mai sus:*
   (a) *Orice număr natural este și număr întreg.*
   (b) *Suma oricăror 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.*

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

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

Exercițiul 36. *Calculați arborele e sintaxă abstractă al următoarelor formule (indicație: puneți paranteze în jurul formulelor, în ordinea de prioritate a conectorilor):*

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

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

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

Exercițiul 38. *Identificați domeniul de vizibilitate, pentru fiecare cuantificator, în formulele φ_1 și φ_2 date în Exercițiul 37.*

Exercițiul 39. *Găsiți variabilele, variabilele libere și respectiv variabilele legate ale formulelor φ_1 și φ_2 date în Exercițiul 37.*

Exercițiul 40. *Fie $A = \{\mathtt{p},\mathtt{q},\mathtt{r},\ldots\}$ 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 $\varphi \in \mathbb{LP}$.*