Logică pentru informatică - note de curs

Universitatea Alexandru Ioan Cuza, Iași
Facultatea de Informatică
Anul universitar 2024-2025

Ștefan Ciobâcă
Andrei Arusoaie
Rodica Condurache
Cristian Masalagiu

Cuprins

Capitolul 1

Introducere

Dacă aritmetica este știința care studiază numerele și operațiile cu numere, logica este știința în care obiectul studiului este reprezentat de *propoziții* și de operații cu propoziții.

De exemplu, dacă în aritmetică observăm că suma a două numere pare este un număr par, în logică am putea observa că disjuncția a două propoziții adevărate este o propoziție adevărată.

Logica se găsește la intersecția filosofiei, matematicii și informaticii și a cunoscut cea mai mare dezvoltare începând cu anii 1950, datorită aplicațiilor numeroase în Informatică.

În acest curs, vom studia la nivel introductiv *logica propozițională* și *logica de ordinul I*.

Logica propozițională este extrem de simplă, dar conceptele pe care le studiem, metodele pe care le învățăm și problemele pe care le întâlnim în logica propozițională se pot generaliza la alte logici mai complexe. De asemenea, logica propozițională corespunde în mod fidel organizării interne la nivel abstract a calculatoarelor, în sensul în care circuitele electronice se pot modela ca formule din logica propozițională. Logica propozițională are o teorie bogată și interesantă din punct de vedere matematic (exemple: teorema de compactitate, teorema de interpolare a lui Craig). *Problema satisfiabilității* pentru logica propozițională are multe aplicații în informatică. Este deosebit de importantă atât din punct de vedere teoretic (fiind problema canonică NP-completă) cât și practic (cu aplicații în verificarea programelor, în verificarea circuitelor, în optimizare combinatorială ș.a.).

Logica de ordinul I este o extensie a logicii propoziționale și are de asemenea aplicații numeroase în informatică, dar și în matematică. De exemplu, toată matematica pe care ați învățat-o în gimnaziu/liceu se bazează pe o așa numită teorie din logica de ordinul I numită **ZFC** (teoria **Z**ermelo–**F**raenkel a mulțimilor, împreună cu axioma alegerii – Axiom of **C**hoice). În Infor-

matică, aplicații ale logicii de ordinul I apar în domeniul complexității descriptive, în baze de date relaționale, în verificarea automată a hardware-ului și software-ului, ș.a. De asemenea, multe alte logici (de exemplu, logicile de ordin superior) au aplicații în teoria limbajelor de programare, în fundamentele matematicii, teoria tipurilor etc.

Capitolul 2

Logica propozițională informală

Logica propozițională este logica propozițiilor, conectate între ele prin *conectori logici* cum ar fi *sau*, *și* și *non*. În acest capitol, vom trece prin bazele logicii propoziționale.

2.1 Propoziții

O *propoziție* este o afirmație care este sau adevărată, sau falsă. Iată câteva exemple de propoziții:

1. *Port o cămașă albastră*;
2. *Tu deții un laptop și o tabletă, dar nu un telefon inteligent*;
3. $2 + 2 = 4$ (*Doi și cu doi fac patru*);
4. $1 + 1 = 1$ (*Unu plus unu este 1*);
5. $1 + 1 \neq 1$ (*Unu cu unu nu fac 1*);
6. *Dacă* $1 + 1 = 1$*, atunci Pământul este plat*;
7. *Toate numerele naturale sunt întregi*;
8. *Toate numere raționale sunt întregi.*

Iată câteva exemple de expresii care nu sunt propoziții:

1. *Roșu și negru* (nu este o afirmație);

2. π (nu este o afirmație);

3. *Plouă?* (întrebare, nu afirmație);

4. *Pleacă!* (exclamație, nu afirmație);

5. $x > 7$ (aici avem un predicat care depinde de x; după ce fixăm o valoare pentru x, obținem o propoziție);

6. *Această afirmație este falsă.* (deși este o afirmație, nu este propoziție, deoarece nu este sau adevărată sau falsă: dacă ar fi adevărată, atunci ar fi și falsă și invers).

Câteodată nu este foarte clar dacă o afirmație este propoziție cu adevărat, sau nu este foarte clară valoarea ei de adevăr. De exemplu, suntem de acord că *zăpada este albă* în general, dar la fel de bine se poate susține și contrariul: de exemplu, există zăpadă neagră (pe stradă în țările subdezvoltate în timpul iernii), astfel încât valoarea de adevăr a afirmației *zăpada este albă* este pusă în discuție. Faptul că o afirmație este propoziție sau nu este mai mult o problemă de logică filosofică.

2.2 Propoziții atomice

Unele propoziții sunt atomice, în sensul în care nu pot fi descompuse în propoziții mai mici:

1. *Port o cămașă albastră*;

2. *Tu deții un laptop*;

3. $2 + 2 = 4$ (*Doi și cu doi fac patru*).

2.3 Conjuncții

Unele propoziții sunt compuse din altele mai mici. De exemplu, propoziția *afară plouă și eu sunt supărat* este compusă din *afară plouă* și din *sunt supărat*, legate între ele prin *și*. Dacă două propoziții, φ și respectiv ψ, sunt legate printr-un *și*, propoziția rezultată, φ *și* ψ, se numește o *conjuncție* (*conjuncția lui* φ *și* ψ).

O conjuncție este adevărată dacă ambele părți componente sunt adevărate. De exemplu, propoziția *afară plouă și eu sunt supărat* este adevărată dacă atât propoziția *afară plouă* cât și propoziția *sunt supărat* sunt adevărate. În particular, din moment ce nu sunt supărat, această conjuncție este falsă.

O conjuncție nu conține neapărat cuvântul *și* în mod explicit. De exemplu, propoziția *afară plouă, dar eu am o umbrelă* este de asemenea o conjuncție, iar părțile ei componente sunt *afară plouă* și *eu am o umbrelă*. Aceste propoziții sunt legate prin conjuncția adversativă *dar*.

Exercițiul 1. *Găsiți părțile componente ale conjuncției* mă joc acasă și învăț la școală.

Exercițiul 2. *Dați un exemplu de o conjuncție falsă și un exemplu de o conjuncție adevărată.*

2.4 Disjuncții

Disjuncțiile sunt propoziții legate între ele prin *sau*. De exemplu, *afară plouă sau sunt supărat* este o disjuncție a propozițiilor *afară plouă* și *sunt supărat*.

Exercițiul 3. *Găsiți părțile componente ale disjuncției* Voi cumpăra un laptop sau o tabletă. *Atenție! Cele două părți componente trebuie să fie propoziții (anumite cuvinte din cele două părți componente pot să fie implicite și în consecință să nu apară în text).*

O disjuncție este adevărată dacă cel puțin una din părțile sale componente este adevărată. De exemplu, disjuncția $7 > 8$ *sau* $8 > 7$ este adevărată, deoarece $8 > 7$ este adevărată.

Acest înțeles al disjuncțiilor se numește *sau inclusiv* și este standard în matematică. Câteodată cuvântul *sau* este folosit în limbaj natural în sensul de *sau exclusiv*. De exemplu, în propoziția *albul sau negrul câștigă într-un joc de go*, cuvântul *sau* are înțeles de *sau exclusiv*. Înțelesul este că sau albul câștigă, sau negrul, dar nu amândoi simultan (este exclusă opțiunea să fie ambele adevărate în același timp).

În continuare, prin disjuncție vom înțelege *sau inclusiv* (interpretarea standard în matematică).

Exercițiul 4. *Dați un exemplu de o disjuncție falsă și un exemplu de o disjuncție adevărată.*

Exercițiul 5. *Când este disjuncția* φ sau ψ *falsă (în funcție de valorile lui* φ *și* ψ*)?*

Exercițiul 6. *Putem reprezenta* sau exclusiv *utilizând* sau inclusiv*?*

2.5 Implicații

Implicațiile sunt propoziții de forma *dacă* φ *atunci* ψ. Propoziția φ se numește *antecedent* al implicației, iar propoziția ψ se numește *consecvent* (sau *concluzie*) al implicației.

Un exemplu de implicație este *dacă trec la Logică, dau o petrecere.* Antecedentul este *trec la Logică*, iar concluzia este *dau o petrecere.* Când este o implicație adevărată/falsă? O implicație este falsă dacă și numai dacă antecedentul este adevărat, dar consecventul este fals. Să presupunem că trec la Logică și că totuși nu dau o petrecere. Atunci implicația *dacă trec la Logică, dau o petrecere*, în ansamblul său, este falsă (antecedentul este adevărat, dar consecventul fals).

Înțelesul unei implicații merită o discuție mai amănunțită, deoarece poate fi contraintuitiv. Implicațiile, așa cum apar în matematică, pot fi diferite de implicațiile pe care le folosim în viața de zi cu zi. În viața de zi cu zi, când spunem *dacă trec la logică, dau o petrecere*, înțelegem că avem o legătura de cauzalitate între faptul de a trece la Logică și faptul de a da o petrecere. Alte exemple de astfel de legătură de cauzalitate: *dacă am bani, cumpăr o mașină* sau *dacă mă ajuți, te ajut.* În limbajul de zi cu zi, nu ne-am gândi niciodată să conectăm două propoziții printr-o implicație dacă cele două propoziții nu au legătură de cauzalitate între ele. De exemplu, propoziția *dacă pământul este rotund, atunci 2 + 2 = 4* nu ar avea sens (deși este adevărată).

Implicația folosită în matematică se numește *implicație materială.* Valoarea de adevăr a unei implicații depinde doar de valorile de adevăr ale părților componente (antecedentul și consecventul), nu și de legătura de cauzalitate dintre ele. Acest înțeles al implicației materiale nu corespunde tot timpul cu înțelesul din limbajul natural (e.g., limba română), dar practica arată că este singurul înțeles rezonabil în matematică (și informatică).

În particular, atât propoziția *dacă pământul este plat, atunci 2+2=5*, cât și propoziția *dacă pământul este plat, atunci* $2+2=4$ sunt adevărate, deoarece antecedentul este fals.

Exercițiul 7. *Care sunt valorile de adevăr ale propozițiilor* dacă $2+2=4$, atunci Pământul este plat *și* dacă $2+2=5$, atunci Pământul este plat*?*

Valoarea de adevăr a implicației *dacă* φ*, atunci* ψ depinde doar de valorile de adevăr ale antecedentului, φ, și consecventului, ψ, și este prezentată în următorul tabel de adevăr:

φ	ψ	dacă φ, atunci ψ
fals	*fals*	*adevărat*
fals	*adevărat*	*adevărat*
adevărat	*fals*	*fals*
adevărat	*adevărat*	*adevărat*

Următorul exemplu arată că tabelul de adevăr de mai sus este singura intrepretare rezonabilă a implicației. Sunteți cu siguranță de acord că orice număr natural este număr întreg. Altfel spus, propoziția *pentru orice număr x, dacă x este număr natural, atunci x este număr întreg* este adevărată. În

particular veți fi de acord că propoziția este adevărată în cazurile particulare $x = -10$, $x = 10$ și $x = 1.2$ (din moment ce este vorba despre *orice număr* x).

Obținem cazurile particulare *dacă* -10 *este număr natural, atunci* -10 *este număr întreg*, *dacă* 10 *este număr natural, atunci* 10 *este număr întreg* și *dacă* 1.2 *este număr natural, atunci* 1.2 *este număr întreg.*, care trebuie toate să fie adevărate. Aceste cazuri particulare exemplifică rândurile 2, 4 și 1 ale tabelului de adevăr de mai sus (de obicei, studenții nu au încredere în rândul 2). Cât despre a treia linia, o propoziție de forma *dacă* φ*, atunci* ψ, unde φ este adevărată, dar ψ este falsă, nu poate fi decât falsă. Altfel, ar trebui să acceptăm ca fiind adevărate propoziții cum ar fi *Dacă* $2 + 2 = 4$*, atunci* $2 + 2 = 5$. (antecedentul, $2 + 2 = 4$, este adevărat, consecventul, $2 + 2 = 5$, este fals).

Unele implicații pot fi relativ dificil de identificat. De exemplu, în propoziția *trec la Logică doar dacă învăț*, antecedentul este (împotriva aparențelor) *trec la Logică*, iar consecventul este *învăț*. Atenție! Propoziția de mai sus nu are același înțeles cu *dacă învăț, trec la Logică*.

> Atenție! În propozițiile de forma *trec la Logică doar dacă învăț* sau *trec la Logică numai dacă învăț*, antecedentul este *trec la Logică*, iar consecventul este *învăț*. Aceste două propoziții nu au același înțeles cu propoziția *dacă învăț, atunci trec la Logică*.

Implicațiile în limba română pot câteodată să nu folosească șablonul *dacă ... atunci ....* De exemplu, sensul cel mai rezonabil al propoziției *trec la Logică sau renunț la facultate* (aparent o disjuncție), este că *dacă nu trec la Logică, renunț la facultate* (implicație). Din fericire, cele două propoziții sunt echivalente, într-un sens pe care îl vom studia în cursurile următoare.

2.6 Negațiile

O propoziție de forma *nu este adevărat că* φ (sau pur și simplu *nu* φ) se numește *negația* lui φ. De exemplu, *nu plouă* este negația propoziției *plouă*. Valoarea de adevăr a negației unei propoziții φ este opusul valorii de adevăr a propoziției φ. În momentul în care scriu acest text, propoziția *plouă* este falsă, și deci propoziția *nu plouă* este adevărată.

Exercițiul 8. *Dați un exemplu de o propoziție falsă care folosește atât o negație, cât și o conjuncție.*

Exercițiul 9. *În Exercițiul 6 ne întrebam dacă e posibil să reprezentăm* sau exclusiv *utilizând* sau inclusiv*. Dacă utilizăm și conjuncții și negații este posibilă această reprezentare?*

2.7 Echivalențe

O propoziție de forma φ *dacă și numai dacă* ψ se numește *echivalență* sau *dublă implicație*. O astfel de propoziție, în ansamblul său, este adevărată dacă φ și ψ au aceeași valoare de adevăr (ambele false sau ambele adevărate).

De exemplu, în momentul în care scriu acest text, propoziția *plouă dacă și numai dacă ninge* este adevărată. De ce? Deoarece atât propoziția *plouă* cât și propoziția *ninge* sunt false.

Exercițiul 10. *Care este valoarea de adevăr a propoziției* Numărul 7 este impar dacă și numai dacă 7 este număr prim*?*

O echivalență este, din punct de vedere semantic, conjuncția a două implicații: φ *dacă și numai dacă* ψ transmite aceeași informație cu

$$\underbrace{\varphi \text{ dacă } \psi}_{\text{implicația inversă}} \quad \text{și} \quad \underbrace{\varphi \text{ numai dacă } \psi}_{\text{implicația directă}}.$$

Propoziția φ *dacă* ψ este aceeași cu *dacă* ψ*, atunci* φ (doar că are altă topică). Propoziția φ *numai dacă* ψ are același înțeles cu φ *doar dacă* ψ și cu *dacă* φ*, atunci* ψ, după cum am discutat în secțiunea referitoare la implicații.

2.8 Conectorii logici

Cuvintele/expresiile *și*, *sau*, *dacă-atunci*, *doar dacă*, *non*, *dacă-și-numai-dacă* (și altele similare) sunt numite *conectori logici*, deoarece pot fi folosite pentru a conecta propoziții mai mici pentru a obține propoziții mai mari.

Atenție! O propoziție este *atomică* în logica propozițională dacă nu poate fi despărțită în propoziții mai mici separate de conectorii logici discutați mai sus. De exemplu, propoziția *orice număr natural este și număr întreg* este o propoziție atomică (în logica propozițională).

Aceeași propoziție nu mai este neapărat atomică într-o altă logică mai bogată. De exemplu, în logica de ordinul I (pe care o vom studia în partea a doua a cursului), avem doi conectori suplimentari numiți *cuantificatori*. În logica de ordinul I, propoziția *orice număr natural este și număr întreg* nu este atomică.

2.9 Ambiguități în limba română

Am prezentat mai sus limbajul logicii propoziționale: propoziții atomice conectate prin *și*, *sau*, *non* etc. Până în acest moment, am folosit limba română.

Totuși, limba română (și orice alt limbaj natural) nu este potrivită pentru scopul nostru din cauza *ambiguităților*.

Iată exemple de propoziții ambigue:

1. *Ion și Maria sunt căsătoriți* (înțelesul 1: între ei; înțelesul 2: căsătoriți, dar posibil cu alte persoane);
2. *Văd negru* (înțeles 1: sunt supărat; înțeles 2: mă simt rău; înțeles 3: nu este lumină în jur etc.);
3. *Trimit mesajul lui Ion* (înțeles 1: mesajul este al lui Ion și eu îl trimit, nu se știe unde; înțeles 2: am un mesaj și îl trimit către Ion);
4. *Nu vorbesc și mănânc* (înțeles 1: neg faptul că și vorbesc și mănânc; înțeles 2: nu vorbesc, dar mănânc).

Astfel de ambiguități sunt cel puțin neplăcute dacă scopul nostru este să determinăm valoarea de adevăr a unei propoziții (dacă nici măcar nu suntem siguri ce înseamnă propoziția). Pentru peste 2000 de ani, logica a lucrat cu limbaj natural. Nevoia de a introduce un limbaj formal, simbolic, fără ambiguități, a apărut în secolele XVIII - XIX, odată cu dezvoltarea logicii matematice. În logica simbolică/formală, pe care urmează să o studiem, vom lucra cu un limbaj artificial, numit limbaj formal, care este proiectat de o asemenea manieră încât să nu conțină nicio ambiguitate.

În fapt, primul limbaj formal pe care îl vom studia va fi *limbajul formal al logicii propoziționale* (pe scurt, *logica propozițională*).

În cele de mai sus, am observat care sunt dezavantajele și eventualele capcane ale logicii informale. În domeniul programării calculatoarelor, toate cerințele sunt exprimate via limbaj natural. Specificațiile unui produs software sunt de regulă transmise sub această formă. Este foarte important pentru cei care dezvoltă propriu-zis produsul software să identifice posibile lacune și ambiguități din specificații pentru a preveni viitoare comportamente nedorite ale produsului. Unul din scopurile studierii logicii în domeniul informaticii este să dezvolte abilitatea programatorilor de a gândi sistematic și de a formula clar specificațiile unui produs.

2.10 Fișă de exerciții

Exercițiul 11. *Stabiliți care dintre următoarele expresii sunt propoziții:*

1. Tu ai un laptop.

2. Zăpada este albă.

3. Zăpada nu este albă.

4. Tatăl meu merge la serviciu şi eu merg la şcoală.
5. Afară plouă, dar eu am umbrelă.
6. Mâine va ploua sau nu va ploua.
7. Dacă obţin notă de trecere la logică, voi sărbători.
8. $2 + 2 = 4$. *(*Doi plus doi egal cu 4.*)*
9. Roșu și Negru.
10. π.
11. Plouă?
12. Hai la pescuit!
13. x este mai mare decât 7.
14. Această afirmație este falsă.

Exercițiul 12. *Pentru toate propozițiile identificate, stabiliți dacă sunt propoziții atomice sau compuse, iar dacă sunt compuse, stabiliți dacă sunt conjuncții, disjuncții, implicații, negații, echivalențe.*

Exercițiul 13. *Stabiliți dacă propozițiile de mai jos sunt propoziții atomice sau compuse, iar dacă sunt compuse, stabiliți tipul acestora (conjuncții, disjuncții, implicații, negații, echivalențe). Pentru fiecare propoziție compusă, discutați care sunt valorile de adevăr în funcție de valorile de adevăr ale componentelor.*

1. Am mâncat atât de mult, încât mi-a fost rău.
2. Nu am rochie, nici pantofi.
3. Ma întorc acasă sau la amicul meu.
4. Merg afară dacă nu plouă.
5. Mă duc la el doar dacă nu mă ascultă.
6. Dacă nu merg la pescuit atunci soarele nu este rotund.
7. Dacă soarele nu este rotund atunci merg la pescuit.

Exercițiul 14. *Reformulați exemplele de propoziții din Secțiunea 2.9 astfel încât ambiguitățile să fie evitate. Puteți să alegeți oricare înțeles.*

Capitolul 3

Sintaxa formală a logicii propoziționale

În continuare, vom studia limbajul formal al logicii propoziționale.

Prin *sintaxă* înțelegem, în general, un ansamblu de reguli pentru scrierea corectă. După cum am promis în capitolul anterior, în acest capitol vom dezvolta un limbaj artificial, pe care îl vom numi *logica propozițională formală* (pe scurt, îl vom numi doar *logica propozițională*). Sintaxa formală a logicii propoziționale se referă la regulile de scriere pentru acest limbaj.

3.1 Noțiunea de alfabet în informatică

În contextul informaticii, prin *alfabet* se înțelege o mulțime. Elementele unui alfabet se numesc *simboluri*. De cele mai multe ori, alfabetul este o mulțime finită (dar nu în cazul logicii propoziționale).

Care este diferența dintre o mulțime și un alfabet? A priori, niciuna. Contează intenția – ce vom face cu elementele mulțimii/alfabetului mai departe. Folosim elementele unui alfabet pentru a crea *cuvinte*. Un *cuvânt* peste un alfabet este o secvență de simboluri din alfabet.

Exemplul 15. *Fie alfabetul* $X = \{0, 1\}$. *Șirurile/secvențele de simboluri* `001010`, `101011` *și* `1` *sunt exemple de cuvinte peste alfabetul* X.

Cuvântul *vid*, format din 0 simboluri ale alfabetului este notat de obicei cu ε.

3.2 Alfabetul logicii propoziționale

Limbajul logicii propoziționale este format din *formule propoziționale*, care modelează propoziții din logica propozițională, propoziții despre care am discutat în capitolul anterior.

Formulele propoziționale sunt cuvinte peste *alfabetul logicii propoziționale* (anumite cuvinte sunt formule; nu toate).

Alfabetul logicii propoziționale este format din reuniunea următoarelor mulțimi:

1. $A = \{\mathtt{p}, \mathtt{q}, \mathtt{r}, \mathtt{p}', \mathtt{q}_1, \ldots\}$, mulțimea *variabilelor propoziționale*;
2. $\{\neg, \wedge, \vee\}$, mulțimea *conectorilor logici*;
3. $\{(,)\}$, mulțimea simbolurilor auxiliare (un simbol pentru paranteză deschisă și un simbol pentru paranteză închisă).

Astfel, mulțimea $L = A \cup \{\neg, \wedge, \vee\} \cup \{(,)\}$ este alfabetul logicii propoziționale. Iată exemple de cuvinte peste mulțimea L:

1. $))\mathtt{p} \vee \wedge$;
2. $\vee \vee \neg(\mathtt{p})$;
3. $\mathtt{p}$;
4. $\mathtt{ppp}$;
5. $\neg(\mathtt{p} \vee \mathtt{q})$.

În cele ce urmează vom vedea ca doar anumite cuvinte se numesc formule propoziționale. Unii autori folosesc termenul de *well-formed formula* sau *wff*, dar în aceste note de curs vom utiliza doar noțiunea de formulă.

De exemplu, ultimul cuvânt din lista de mai sus, $\neg(\mathtt{p} \vee \mathtt{q})$, este o formulă a logicii propoziționale, dar al doilea cuvânt, $\vee \vee \neg(\mathtt{p})$, nu este formulă. Următoarea definiție surprinde cu precizie care cuvinte sunt formule și care cuvinte nu sunt formule propoziționale.

3.3 Formule propoziționale

Definiția 16 (Mulțimea formulelor propoziționale, notată $\mathbb{LP}$)**.** *Mulțimea* $\mathbb{LP}$ *(mulțimea formulelor propoziționale) este* cea mai mică mulțime *de cuvinte peste alfabetul logicii propoziționale care satisface următoarele proprietăți:*

• *Cazul de Bază. Orice variabilă propozițională, văzută ca un cuvânt de lungime* 1*, este în mulțimea* $\mathbb{LP}$*;*

• *Cazul Inductiv 1. Dacă* $\varphi \in \mathbb{LP}$*, atunci* $\neg\varphi \in \mathbb{LP}$ *(sau: dacă cuvântul* φ *este o formulă propozițională, atunci și cuvântul care începe cu simbolul* $\neg$ *și continuă cu simbolurile din* φ *este formulă propozițională);*

• *Cazul Inductiv 2. Dacă* $\varphi_1, \varphi_2 \in \mathbb{LP}$*, atunci* $(\varphi_1 \wedge \varphi_2) \in \mathbb{LP}$ *(sau: . . . – exercițiu pentru acasă);*

• *Cazul Inductiv 3. Dacă* $\varphi_1, \varphi_2 \in \mathbb{LP}$*, atunci* $(\varphi_1 \vee \varphi_2) \in \mathbb{LP}$ *(sau: . . . – exercițiu pentru acasă).*

Iată câteva exemple de elemente ale mulțimii $\mathbb{LP}$:

$$\mathtt{p}, \quad \mathtt{q}, \quad \neg\mathtt{p}, \quad \neg\mathtt{q}, \quad \neg\mathtt{p}', \quad \neg\neg\mathtt{p}_1, \quad (\mathtt{p} \vee \mathtt{q}), \quad (\mathtt{p} \wedge \mathtt{q}),$$
$$\neg(\mathtt{p} \vee \mathtt{q}), \quad (\neg\mathtt{p} \wedge \neg\mathtt{q}), \quad \neg(\neg\neg\mathtt{p} \vee \mathtt{p}), \quad ((\mathtt{p} \vee \mathtt{q}) \wedge \mathtt{r}),$$
$$(\mathtt{p} \vee (\mathtt{q} \wedge \mathtt{r})).$$

Iată exemple de cuvinte care nu sunt în $\mathbb{LP}$:

$$\mathtt{pp}, \quad \mathtt{q}\neg\mathtt{q}, \quad \mathtt{q} \wedge \neg\mathtt{p}, \quad \mathtt{p} + \mathtt{q}.$$

Definiția mulțimii $\mathbb{LP}$ este un exemplu de *definiție inductivă*. Astfel de definiții sunt foarte importante în informatică și este obligatoriu să ajungem să le înțelegem foarte bine. Într-o definiție inductivă, există de obicei unul sau mai multe cazuri de bază, care definesc cele mai *mici* elemente ale mulțimii (în cazul nostru, variabilele propoziționale). Cazurile inductive arată cum se pot construi elemente mai *mari* pornind de la alte elemente mai *mici*, despre care știm deja că sunt în mulțime. Alt element important este restricția de minimalitate (subliniată în definiția de mai sus), care indică că **doar** elementele care pot fi construite apelând la cazurile de bază și la cazurile inductive fac parte din mulțime. Această restricție de minimalitate asigură unicitatea mulțimii (nu există o altă mulțime cu cele 4 proprietăți și care să fie minimală) și ne permite să arătăm ca anumite elemente nu fac parte din mulțime (cele care nu pot fi construite prin cazurile de bază/inductive).

De exemplu, cuvântul $\mathtt{pp}$ nu este în mulțimea $\mathbb{LP}$, deoarece nu poate fi construit prin aplicarea niciunui caz dintre cele 4 (cazul de bază sau cele trei cazuri inductive).

3.4 Cum arătăm că un cuvânt face parte din $\mathbb{LP}$

Putem arăta că un cuvânt face parte din $\mathbb{LP}$ explicând cum se pot aplica pasul de bază și pașii inductivi. Iată o demonstrație a faptului că $\neg(p \vee q) \in \mathbb{LP}$:

1. $p \in \mathbb{LP}$ (din pasul de bază, deoarece $p \in A$);
2. $q \in \mathbb{LP}$ (din pasul de bază, deoarece $q \in A$);
3. $(p \vee q) \in \mathbb{LP}$ (din pasul inductiv 3, unde $\varphi_1 = p$ și $\varphi_2 = q$);
4. $\neg(p \vee q) \in \mathbb{LP}$ (din pasul inductiv 1, unde $\varphi = (p \vee q)$).

Putem rearanja lista de mai sus într-un *arbore de construcție adnotat* echivalent pentru formula $\neg(p \vee q)$:

$$\dfrac{\dfrac{\dfrac{}{p \in \mathbb{LP}}\ \textit{pas de bază} \qquad \dfrac{}{q \in \mathbb{LP}}\ \textit{pas de bază}}{(p \vee q) \in \mathbb{LP}}\ \textit{pas inductiv 3}}{\neg(p \vee q) \in \mathbb{LP}}\ \textit{pas inductiv 1}$$

Vom vedea acest tip de notație de mai multe ori în Informatică și de aceea e important să ne familiarizăm cu ea. Fiecare linie orizontală se numește *inferență*; sub fiecare astfel de linie este concluzia inferenței și deasupra ei sunt ipotezele (0, 1 sau mai multe ipoteze). Inferențele cu 0 ipoteze se numesc axiome. Lângă fiecare linie se găsește numele regulii care a fost aplicată.

Dacă renunțăm la adnotații, obținem următorul *arbore de construcție* pentru formula $\neg(p \vee q)$:

$$\dfrac{\dfrac{\dfrac{}{p} \qquad \dfrac{}{q}}{(p \vee q)}}{\neg(p \vee q)}$$

Este ușor de văzut că un cuvânt aparține $\mathbb{LP}$ dacă și numai dacă există un arbore de construcție pentru acel cuvânt.

3.5 Conectorul principal al unei formule

O formulă care constă doar dintr-o variabilă propozițională (cum ar fi formula p sau formula q) se numește *formulă atomică*. Acest lucru explică de ce mulțimea A a variabilelor propoziționale se numește A (A, de la *atomic*).

Formulele mai complexe, cum ar fi $\neg p$ sau $(p \vee q)$, se numesc *compuse* (sau, în engleză, *moleculare*).

Orice formulă compusă are un *conector principal*, care este dat de ultima inferență din arborele său de construcție. De exemplu, conectorul principal al formulei $\neg(p \vee q)$ este $\neg$ (negația), în timp ce conectorul principal al formulei $(\neg p \vee q)$ este $\vee$ (disjuncția). Numim formulele al căror conector principal este $\wedge$ *conjuncții*. Similar, dacă conectorul principal al unei formule este $\vee$, formula este o *disjuncție*, iar dacă conectorul principal este $\neg$, atunci este o *negație*.

Orice formulă compusă are un conector principal.

3.6 Cum arătăm că un cuvânt nu face parte din $\mathbb{LP}$

Este puțin mai dificil (sau, mai bine spus, lung) să arătăm că un cuvânt nu face parte din $\mathbb{LP}$.

Dacă cuvântul nu este peste alfabetul corect, așa cum este cuvântul cu trei simboluri $p + q$ (care folosește simbolul străin $+$), atunci este evident că acesta nu face parte din $\mathbb{LP}$, deoarece $\mathbb{LP}$ conține doar cuvinte peste alfabetul logicii propoziționale.

Dacă cuvântul este peste alfabetul corect și vrem să arătăm că nu face parte din $\mathbb{LP}$, atunci trebuie să ne folosim de condiția de minimalitate. Iată cum putem arăta că $(p\neg q) \notin \mathbb{LP}$:

Să presupunem, prin reducere la absurd, că $(p\neg q) \in \mathbb{LP}$. În acest caz, din condiția de minimalitate, faptul că $(p\neg q) \in \mathbb{LP}$ trebuie să poată fi explicat prin una dintre cele patru reguli de formare (pasul de bază sau unul dintre cei trei pași inductivi).

Dar nu putem aplica cazul de bază pentru a arăta că $(p\neg q) \in \mathbb{LP}$, deoarece $(p\neg q) \notin A$.

De asemenea, nu putem aplica nici cazul inductiv 1, deoarece nu există nicio formulă φ astfel încât $(p\neg q) = \neg\varphi$ (orice formulă $\varphi \in \mathbb{LP}$ am alege, primul simbol al părții stângi ar fi $($, iar primul simbol al părții drepte ar fi $\neg$.

Nu putem aplica nici cazul inductiv 2, deoarece nu există formulele $\varphi_1, \varphi_2 \in \mathbb{LP}$ astfel încât $(p\neg q) = (\varphi_1 \wedge \varphi_2)$ (oricum am alege $\varphi_1, \varphi_2 \in \mathbb{LP}$, cuvântul din partea stângă nu conține simbolul $\wedge$, în timp ce cuvântul din dreapta îl conține).

Dintr-un motiv similar, nu putem aplica nici cazul inductiv 3.

Din moment ce niciunul dintre cele patru cazuri nu funcționează, presupunerea noastră este falsă, și deci $(p\neg q) \notin \mathbb{LP}$.

3.7 Proprietatea de citire unică

Definiția $\mathbb{LP}$ are o proprietate importantă, care se numește *proprietatea de citire unică*:

Teorema 17 (Proprietatea de citire unică a formulelor propoziționale)**.** *Pentru orice formulă $\varphi \in \mathbb{LP}$, există un unic arbore de construcție.*

Proprietatea de mai sus se mai numește și neambiguitatea gramaticii logicii propoziționale. Demonstrarea proprietății nu face parte din acest curs. Totuși, trebuie să înțelegem semnificația ei. În acest sens, vom considera o altă posibilă definiție (greșită) pentru $\mathbb{LP}$ și vom arată în ce sens această definiție alternativă este ambiguă.

Începutul unei definiții greșite pentru $\mathbb{LP}$.

Să ne imaginăm că pasul inductiv 3 ar fi:

$\vdots$

- Cazul Inductiv 3. Dacă $\varphi_1, \varphi_2 \in \mathbb{LP}$, atunci $\varphi_1 \vee \varphi_2 \in \mathbb{LP}$;

$\vdots$

Singura diferență față de definiția corectă este că nu putem paranteze în jurul disjuncțiilor. Cu această definiție alternativă, fictivă, a lui $\mathbb{LP}$, am avea că $\neg p \vee q \in \mathbb{LP}$. Totuși, acest cuvânt ar avea doi arbori de construcție diferiți:

$$\dfrac{\dfrac{\overline{p} \qquad \overline{q}}{p \vee q}}{\neg p \vee q} \qquad \dfrac{\dfrac{\overline{p}}{\neg p} \qquad \overline{q}}{\neg p \vee q}$$

O astfel de ambiguitate ar fi deosebit de neplăcută pentru scopurile noastre, deoarece nu am putea ști dacă $\neg p \vee q$ este o disjuncție (ca în arborele de construcție din dreapta) sau o negație (ca în arborele de construcție din stânga). De fapt,

> evitarea unor asemenea ambiguități sintactice a fost unul dintre motivele pentru care am părăsit sfera limbajului natural și am început să studiem logica formală.

Sfârșitul unei definiții greșite pentru $\mathbb{LP}$.

După această incursiune, ar trebui să fie clar de ce teorema de citire unică este netrivială (ca un exercițiu pentru cei cu înclinare spre matematică, încercați să o demonstrați). De asemenea, ar trebui să fie clar de ce citirea unică este o proprietate esențială a definiției formulelor: ne arată că orice formulă propozițională poate fi citită într-un singur fel; cu alte cuvinte, orice formulă propozițională nu are ambiguități sintactice.

3.8 Limbaj-obiect și meta-limbaj

O particularitate a logicii este că studiem propoziții (analizăm, de exemplu, valoarea lor de adevăr), iar pentru a efectua studiul, ne folosim de raționament care, în mod natural, sunt și ele alcătuite din propoziții. De exemplu, în cursul studiului nostru, am putea efectua următorul raționament:

> *Propoziția nu merg la școală este falsă dacă propoziția merg la școală este adevărată.*

De observat că afirmațiile care fac obiectul studiului (*nu merg la școală* și *merg la școală*) sunt propoziții, dar și întreaga afirmație *Propoziția nu merg la școală este falsă dacă propoziția merg la școală este adevărată* este la rândul ei o propoziție. Astfel, pare ca facem un raționament circular, în care studiem chiar metoda de studiu.

Această dificultate aparentă nu apare și în cazul aritmeticii, de exemplu. În aritmetică, studiem numere și operații peste numere, iar studiul îl facem folosind propoziții.

Dificultatea își are rezolvarea prin separarea strictă a *limbajului obiect* de *meta-limbaj*. Limbajul obiect este limbajul pe care îl studiem ($\mathbb{LP}$), iar meta-limbajul este limbajul pe care îl folosim pentru a efectua studiul (limba română).

Limbajul-obiect este limbajul care face obiectul studiului (în cazul nostru, logica propozițională). Meta-limbajul este limbajul pe care îl folosim pentru a comunica despre obiectul studiului (în cazul nostru, limba română).

În acest curs, pentru a diferenția ușor între cele două, facem următoarea convenție: toate elementele din limbajul-obiect le scriem cu font sans-serif albastru (de exemplu, $(\mathtt{p} \wedge \mathtt{q})$), iar afirmațiile din meta-limbaj le scriem folosind *font negru obișnuit* (de exemplu, *orice formulă atomică este o formulă*).

> Este extrem de important să facem diferența între limbajul-obiect și meta-limbaj. În acest scop, notele de curs folosesc o convenție tipografică. Elementele meta-limbajului sunt scrise cu font obișnuit, de culoare neagră. Elementele limbajului-obiect sunt scrise astfel: $(\mathtt{p} \wedge \mathtt{q})$. Din acest motiv, dacă imprimați notele de curs, este recomandat să folosiți o imprimantă color.

3.9 Fișă de exerciții

Exercițiul 18. *Arătați că următoarele cuvinte sunt* formule propoziționale *(adică elemente ale mulțimii $\mathbb{LP}$), explicând care sunt pașii de construcție (pas de bază, respectiv unul dintre cei trei pași inductivi):*

1. $\neg \mathtt{q}$;
2. $(\mathtt{p}_1 \wedge \mathtt{q})$;
3. $\neg(\mathtt{p} \vee \mathtt{q})$;
4. $(\neg \mathtt{p} \vee \neg \mathtt{q})$;
5. $(\neg \mathtt{p} \wedge \neg \mathtt{q})$.

Exercițiul 19. *Arătați că următoarele cuvinte nu sunt elemente ale mulțimii $\mathbb{LP}$ (indicație: arătați că niciuna dintre cele 4 reguli de formare nu poate fi aplicată):*

1. $(\neg)\mathtt{q}$;
2. $\mathtt{q} \wedge \neg$;
3. $\mathtt{pq}$;
4. $\mathtt{p} \wedge \mathtt{q}$;
5. $(\mathtt{p}) \wedge (\mathtt{q})$.

Exercițiul 20. *Care dintre următoarele cuvinte sunt formule din $\mathbb{LP}$ și care nu sunt:*

1. p_1;
2. $p_1 \vee q_1$;
3. $(p_1 \vee q_1)$;
4. $(\neg p_1 \vee q_1)$;
5. $\neg(p_1 \vee q_1)$;
6. $((\neg p_1) \vee q_1)$;
7. $(\neg p)$?

Exercițiul 21. *Dați exemple de 5 formule propoziționale interesante (cu mai mulți conectori, mai multe variabile propoziționale etc.) și justificați că sunt într-adevăr formule.*

Capitolul 4

Funcții recursive peste $\mathbb{LP}$

Memento. *Dacă X este o mulțime, prin 2^X notăm* mulțimea părților lui *X, adică mulțimea tuturor submulțimilor lui X. De exemplu, dacă $X = \{1, 2, 3\}$, atunci $2^X = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$. Când X este finită, avem că $|2^X| = 2^{|X|}$, ceea ce explică notația. În liceu, este posibil să fi folosit notația $\mathcal{P}(X)$ în loc de 2^X.*

Dacă o mulțime este *definită inductiv*, putem defini *funcții recursive* care operează pe elementele mulțimii. Definiția acestor funcții urmează *structura* elementelor din mulțimea definită inductiv.

Iată un exemplu de funcție care calculează mulțimea *subformulelor unei formule*:

subf : $\mathbb{LP} \to 2^{\mathbb{LP}}$, definită prin

$$subf\Big(\varphi\Big) = \begin{cases} \{\varphi\}, & \text{dacă } \varphi \in A; \\ \{\varphi\} \cup subf\Big(\varphi'\Big), & \text{dacă } \varphi = \neg\varphi' \text{ și } \varphi' \in \mathbb{LP}; \\ \{\varphi\} \cup subf\Big(\varphi_1\Big) \cup subf\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \wedge \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}; & \\ \{\varphi\} \cup subf\Big(\varphi_1\Big) \cup subf\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \vee \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}. & \end{cases}$$

Iată cum se poate calcula $subf\Big(\varphi\Big)$ pentru $\varphi = \neg(\mathtt{p} \vee \mathtt{q})$:

$$\begin{aligned} subf\Big(\neg(\mathtt{p} \vee \mathtt{q})\Big) &= \{\neg(\mathtt{p} \vee \mathtt{q})\} \cup subf\Big((\mathtt{p} \vee \mathtt{q})\Big) \\ &= \{\neg(\mathtt{p} \vee \mathtt{q})\} \cup \{(\mathtt{p} \vee \mathtt{q})\} \cup subf\Big(\mathtt{p}\Big) \cup subf\Big(\mathtt{q}\Big) \\ &= \{\neg(\mathtt{p} \vee \mathtt{q})\} \cup \{(\mathtt{p} \vee \mathtt{q})\} \cup \{\mathtt{p}\} \cup \{\mathtt{q}\} \\ &= \{\neg(\mathtt{p} \vee \mathtt{q}), (\mathtt{p} \vee \mathtt{q}), \mathtt{p}, \mathtt{q}\}. \end{aligned}$$

Observați cele două tipuri de paranteze care apar în calculul de mai sus. Pe de o parte, avem parantezele care mărginesc argumentul funcției *subf*, iar pe de altă parte avem parantezele din alfabetul logicii propoziționale. Primul tip de paranteze sunt paranteze obișnuite din matematică, în timp ce al doilea tip sunt simboluri ale alfabetului. Pentru a le diferenția, vom adopta convenția următoare în aceste note de curs: folosim paranteze cu font obișnuit, de obicei mai mari, pentru apelul de funcție (de exemplu (sau $\Big($), și parantezele `(` și `)`, scrise cu font `sans − serif albastru`, pentru simbolurile alfabetului.

Ambele tipuri de paranteze sunt importante. De exemplu, este o greșeală să scriem $subf\Big(\mathtt{p} \vee \mathtt{q}\Big)$ în loc de $subf\Big((\mathtt{p} \vee \mathtt{q})\Big)$, din moment ce cuvântul $\mathtt{p} \vee \mathtt{q}$ nu este o formulă.

Este o greșeală să scriem $subf\Big(\mathtt{p} \vee \mathtt{q}\Big)$ în loc de $subf\Big((\mathtt{p} \vee \mathtt{q})\Big)$, din moment ce cuvântul $\mathtt{p} \vee \mathtt{q}$ nu este o formulă.

Funcția *subf* este prima dintre funcțiile pe care le vom defini recursiv, în funcție de structura formulei $\varphi \in \mathbb{LP}$. Aceste funcții se numesc *recursive structural*. Este foarte important să înțelegem cum operează aceste funcții pentru a înțelege restul cursului. În secțiunile următoare prezentăm și alte exemple de funcții recursive structural.

4.1 Arborele de sintaxă abstractă al unei formule

Următoarea funcție calculează *arborele de sintaxă abstractă* al unei formule:

$arb : \mathbb{LP} \to Trees$, definită prin:

$$arb\Big(\varphi\Big) = \begin{cases} \varphi, & \text{dacă } \varphi \in A; \\ \neg \\ arb\Big(\varphi'\Big) \end{cases}$$

, dacă $\varphi = \neg\varphi'$ și $\varphi' \in \mathbb{LP}$;

$$\wedge \quad arb\Big(\varphi_1\Big) \quad arb\Big(\varphi_2\Big)$$

, dacă $\varphi = (\varphi_1 \wedge \varphi_2)$ și $\varphi_1, \varphi_2 \in \mathbb{LP}$;

$$\vee \quad arb\Big(\varphi_1\Big) \quad arb\Big(\varphi_2\Big)$$

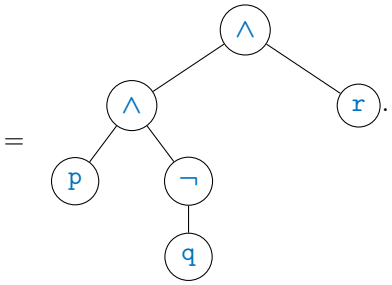
, dacă $\varphi = (\varphi_1 \vee \varphi_2)$ și $\varphi_1, \varphi_2 \in \mathbb{LP}$.

Iată un calcul al arborelui abstract de sintaxă al formulei $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$.

$$arb\Big(((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})\Big) = \quad \wedge \quad arb\Big((\mathtt{p} \wedge \neg\mathtt{q})\Big) \quad arb\Big(\mathtt{r}\Big)$$

$$= \quad \wedge \quad \mathtt{r} \quad arb\Big(\mathtt{p}\Big) \quad arb\Big(\neg\mathtt{q}\Big)$$

$$= \quad \wedge \quad \mathtt{r} \quad \mathtt{p} \quad \neg \quad arb\Big(\mathtt{q}\Big)$$



Mulțimea *Trees* este mulțimea arborilor cu rădăcină, etichetați.

Arborii de sintaxă abstractă sunt extrem de importanți deoarece, din punct de vedere conceptual, o formulă propozițională *este* arborele abstract de sintaxă. Totuși, deoarece scrierea arborilor nu este convenabilă (pentru persoane), recurgem la notația convențională, și scriem formulele în stil tradițional, de la stânga la dreapta, sub formă de cuvinte. În schimb, la implementarea pe un calculator a diferiților algoritmi care prelucrează formule logice, vom prefera reprezentarea formulelor sub forma arborelui de sintaxă abstractă în dauna reprezentării sub forma unui cuvânt (șir de caractere).

> Din punct de vedere conceptual, o formulă propozițională *este* arborele abstract de sintaxă.

4.2 Alte exemple de funcții definite recursiv

Iată și alte exemple de funcții definițe prin recursie structurală peste formule.

Prima funcție, $height : \mathbb{LP} \to \mathbb{N}$, calculează *înălțimea* arborelui abstract de sintaxă al unei formule:

$$height\Big(\varphi\Big) = \begin{cases} 1, & \text{dacă } \varphi \in A; \\ 1 + height\Big(\varphi'\Big), & \text{dacă } \varphi = \neg\varphi' \text{ și } \varphi' \in \mathbb{LP}; \\ 1 + max\Big(height\Big(\varphi_1\Big), height\Big(\varphi_2\Big)\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \wedge \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}; & \\ 1 + max\Big(height\Big(\varphi_1\Big), height\Big(\varphi_2\Big)\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \vee \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}. & \end{cases}$$

Funcția $max : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ care apare în definiția funcției *height* este funcția obișnuită care calculează maximul dintre două numere naturale.

Următoarea funcție, $size : \mathbb{LP} \to \mathbb{N}$, calculează *dimensiunea* arborelui abstract de sintaxă al unei formule (adică numărul de noduri):

$$size\Big(\varphi\Big) = \begin{cases} 1, & \text{dacă } \varphi \in A; \\ 1 + size\Big(\varphi'\Big), & \text{dacă } \varphi = \neg\varphi' \text{ și } \varphi' \in \mathbb{LP}; \\ 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \wedge \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}; & \\ 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \vee \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}. & \end{cases}$$

Ultimul exemplu, $prop : \mathbb{LP} \to 2^A$, calculează toate variabilele propoziționale care apar într-o formulă:

$$prop\Big(\varphi\Big) = \begin{cases} \{\varphi\}, & \text{dacă } \varphi \in A; \\ prop\Big(\varphi'\Big), & \text{dacă } \varphi = \neg\varphi' \text{ și } \varphi' \in \mathbb{LP}; \\ prop\Big(\varphi_1\Big) \cup prop\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \wedge \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}; & \\ prop\Big(\varphi_1\Big) \cup prop\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \vee \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}. & \end{cases}$$

Asigurați-vă că înțelegeți și că știți să definiți și să calculați cu astfel de funcții.

4.3 Demonstrații prin inducție structurală

Câteodată, vom face demonstrații prin *inducție structurală*. Sunteți deja familiarizați din liceu cu demonstrațiile prin inducție matematică.

Pentru a demonstra o teoremă de forma

$$\textit{Pentru orice } n \in \mathbb{N}\textit{, este adevărat că } P(n),$$

principiul inducției matematice postulează că este suficient să arătăm că:

- (cazul de bază) $P(0)$ *este adevărat*;
- (cazul inductiv) $P(k)$ *implică* $P(k+1)$, *pentru orice* $k \in \mathbb{N}$.

Demonstrațiile prin inducție structurală generalizează principiul de mai sus și pot fi aplicate oricărei mulțimi definite inductiv, cum ar fi $\mathbb{LP}$.

Pentru cazul $\mathbb{LP}$, principiul inducției structurale este că, pentru a demonstra o teoremă de forma

Pentru orice formulă propozițională $\varphi \in \mathbb{LP}$, *este adevărat că* $P\Big(\varphi\Big)$,

este suficient să arătăm că:

- (cazul de bază) $P\Big(\varphi'\Big)$ *are loc pentru orice formula atomică* $\varphi' \in A$ (altfel spus, proprietatea P este adevărată pentru orice formulă atomică);
- (cazul inductiv 1) $P\Big(\neg\varphi'\Big)$ *are loc dacă* $P\Big(\varphi'\Big)$ *are loc*;
- (cazul inductiv 2) $P\Big((\varphi_1 \wedge \varphi_2)\Big)$ *are loc dacă* $P\Big(\varphi_1\Big)$ *și* $P\Big(\varphi_2\Big)$ *au loc*;
- (cazul inductiv 3) $P\Big((\varphi_1 \vee \varphi_2)\Big)$ *are loc dacă* $P\Big(\varphi_1\Big)$ *și* $P\Big(\varphi_2\Big)$ au loc.

Iată un exemplu de teoremă și de demonstrația ei prin inducție structurală:

Teorema 22 (Exemplu de teoremă care poate fi demonstrată prin inducție structurală)**.** *Pentru orice formulă propozițională* φ, *avem că* $height\Big(\varphi\Big) \leq size\Big(\varphi\Big)$.

Demonstrație: Facem demonstrația prin inducție structurală, unde proprietatea P este definită astfel:

$$P(\varphi) = height\Big(\varphi\Big) \leq size\Big(\varphi\Big).$$

Este suficient să arătăm că:

1. (cazul de bază) $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$ pentru orice variabilă propozițională $\varphi' \in A$.

Dar, prin definiție, $height\Big(\varphi'\Big) = 1$ și $size\Big(\varphi'\Big) = 1$ (deoarece $\varphi' \in A$). Și $1 \leq 1$, deci am terminat de demonstrat acest caz.

2. (cazul inductiv 1) Presupunând că $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$, arătăm că $height\Big(\neg\varphi'\Big) \leq size\Big(\neg\varphi'\Big)$.

Dar, din definiție, $height\Big(\neg\varphi'\Big) = 1 + height\Big(\varphi'\Big)$ și $size\Big(\neg\varphi'\Big) = 1 + size\Big(\varphi'\Big)$. Și $1 + height\Big(\varphi'\Big) \leq 1 + size\Big(\varphi'\Big)$ (ceea ce trebuia să arătăm), din moment ce știm deja că $height\Big(\varphi'\Big) \leq size\Big(\varphi'\Big)$.

3. (cazul inductiv 2) Presupunând că $height\Big(\varphi_1\Big) \leq size\Big(\varphi_1\Big)$ și $height\Big(\varphi_2\Big) \leq size\Big(\varphi_2\Big)$, arătăm că $height\Big((\varphi_1 \wedge \varphi_2)\Big) \leq size\Big((\varphi_1 \wedge \varphi_2)\Big)$.

Dar, prin definiție, $height\Big((\varphi_1 \wedge \varphi_2)\Big) = 1+max\Big(height\Big(\varphi_1\Big), height\Big(\varphi_2\Big)\Big)$ și, din moment ce $max(a, b) \leq a + b$ (pentru orice numere naturale $a, b \in \mathbb{N}$), avem că $height\Big((\varphi_1 \wedge \varphi_2)\Big) \leq 1+height\Big(\varphi_1\Big)+height\Big(\varphi_2\Big)$. Dar $height\Big(\varphi_i\Big) \leq size\Big(\varphi_i\Big)$ $(1 \leq i \leq 2)$ din ipoteză, și deci $height\Big((\varphi_1 \wedge \varphi_2)\Big) \leq 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big)$. Dar, din definiție, $size\Big((\varphi_1 \wedge \varphi_2)\Big) = 1 + size\Big(\varphi_1\Big) + size\Big(\varphi_2\Big)$, și deci $height\Big((\varphi_1 \wedge \varphi_2)\Big) \leq size\Big((\varphi_1 \wedge \varphi_2)\Big)$, ceea ce trebuia să arătăm.

4. (cazul inductiv 3) analog cazului inductiv 2.

q.e.d.

4.4 Fișă de exerciții

Exercițiul 23. *Calculați, folosind funcția subf, mulțimea de subformule ale formulelor:*
1. $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$*;* *2.* $((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$*;* *3.* $\neg((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$*.*

Exercițiul 24. *Calculați arborii abstracți ai următoarelor formule:*

1. $((\mathtt{p} \wedge \neg\mathtt{q}) \wedge \mathtt{r})$*;*
2. $((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$*;*
3. $\neg((\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$*;*
4. $(\neg(\mathtt{p} \vee \neg\mathtt{q}) \wedge \mathtt{r})$*.*

Exercițiul 25. *Calculați, utilizând funcția height* : $\mathbb{LP} \to \mathbb{N}$*, înălțimea arborelui abstract de sintaxă pentru fiecare dintre formulele de la Exercițiul 24.*

Exercițiul 26. *Calculați, utilizând funcția size* : $\mathbb{LP} \to \mathbb{N}$*, numărul de noduri al arborelui abstract de sintaxă pentru fiecare dintre formulele de la Exercițiul 24.*

Exercițiul 27. *Calculați, utilizând funcția prop* : $\mathbb{LP} \to 2^A$*, mulțimea variabilelor propoziționale care apar în fiecare dintre formulele de la Exercițiul 24.*

Exercițiul 28. *Demonstrați utilizând principiul inducției structurale că pentru orice formulă propozițională* φ*, avem că* $height\Big(\varphi\Big) < size\Big(\varphi\Big) + 1$.

Capitolul 5

Semantica logicii propoziționale

5.1 Algebra booleană

Mulțimea $B = \{0, 1\}$ se numește mulțimea valorilor booleene (sau mulțimea valorilor de adevăr). Valorea 0 reprezintă falsul și valoarea 1 reprezintă adevărul.

Funcția $\bar{}\ : B \to B$ se numește *negație logică* și este definită astfel: $\overline{0} = 1$ și $\overline{1} = 0$.

Funcția $+ : B \times B \to B$ se numește *disjuncție logică* și este definită astfel: $0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 1$.

Funcția $\cdot : B \times B \to B$ se numește *conjuncție logică* și este definită astfel: $0 \cdot 0 = 0, 0 \cdot 1 = 0, 1 \cdot 0 = 0, 1 \cdot 1 = 1$.

Tuplul $(B, +, \cdot, \bar{})$ formează o *algebră booleană*.

5.2 Atribuiri

O *atribuire de valori de adevăr* (sau pur și simplu *atribuire* de aici înainte) este orice funcție $\tau : A \to B$. Cu alte cuvinte, o atribuire este o funcție care asociază fiecărei variabile propoziționale o valoare de adevăr.

Exemplul 29. *Fie $\tau_1 : A \to B$ o funcție definită după cum urmează:*

1. $\tau_1(\mathtt{p}) = 1$;
2. $\tau_1(\mathtt{q}) = 0$;

3. $\tau_1(\mathtt{r}) = 1$;
4. $\tau_1(a) = 0$ *pentru orice* $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.

Din moment ce este o funcție de la A *la* B, τ_1 *este o atribuire de valori de adevăr.*

Exemplul 30. *Fie* $\tau_2 : A \to B$ *o funcție definită după cum urmează:*

1. $\tau_2(\mathtt{p}) = 0$;
2. $\tau_2(\mathtt{q}) = 0$;
3. $\tau_2(\mathtt{r}) = 1$;
4. $\tau_2(a) = 1$ *pentru orice* $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.

Din moment ce este o funcție de la A *la* B, τ_2 *este de asemenea o atribuire.*

Exemplul 31. *Fie* $\tau_3 : A \to B$ *o funcție definită după cum urmează:*

1. $\tau_3(a) = 0$ *pentru orice* $a \in A$.

Din moment ce este o funcție de la A *la* B, τ_3 *este o atribuire.*

5.3 Valoarea de adevăr a unei formule într-o atribuire

Valoarea de adevăr a unei formule φ într-o atribuire τ este notată cu $\hat{\tau}\Big(\varphi\Big)$ și este definită astfel:

$$\hat{\tau}\Big(\varphi\Big) = \begin{cases} \tau(\varphi), & \text{dacă } \varphi \in A; \\ \overline{\hat{\tau}\Big(\varphi'\Big)}, & \text{dacă } \varphi = \neg\varphi' \text{ și } \varphi' \in \mathbb{LP}; \\ \hat{\tau}\Big(\varphi_1\Big) \cdot \hat{\tau}\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \wedge \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}; & \\ \hat{\tau}\Big(\varphi_1\Big) + \hat{\tau}\Big(\varphi_2\Big), & \\ \qquad \text{dacă } \varphi = (\varphi_1 \vee \varphi_2) \text{ și } \varphi_1, \varphi_2 \in \mathbb{LP}. & \end{cases}$$

De fapt, funcția $\hat{\tau} : \mathbb{LP} \to B$ se numește *extensia homomorfică* a atribuirii $\tau : A \to B$ la mulțimea de formule $\mathbb{LP}$.

Un exemplu de calcul a valorii de adevăr a formulei $(p \vee q)$ în atribuirea τ_1 este prezentat mai jos:

$\hat{\tau}_1\Big((p \vee q)\Big) = \hat{\tau}_1\Big(p\Big) + \hat{\tau}_1\Big(q\Big) = \tau_1(p) + \tau_1(q) = 1 + 0 = 1.$

Concluzionăm că valoarea de adevăr a formulei $(p \vee q)$ în τ_1 este 1.

Iată un alt exemplu, în care calculăm valoarea de adevăr a formulei $\neg(p \wedge q)$ în atribuirea τ_1:

$\hat{\tau}_1\Big(\neg(p \wedge q)\Big) = \overline{\hat{\tau}_1\Big((p \wedge q)\Big)} = \overline{\hat{\tau}_1\Big(p\Big) \cdot \hat{\tau}_1\Big(q\Big)} = \overline{\tau_1(p) \cdot \tau_1(q)} = \overline{1 \cdot 0} = \overline{0} = 1.$

Concluzionăm că valoarea de adevăr a formulei $\neg(p \wedge q)$ în τ_1 este 1.

Iată încă un exemplu, în care calculăm valorea de adevăr a formulei $\neg\neg q$ în atribuirea de valori de adevăr τ_2:

$\hat{\tau}_2\Big(\neg\neg q\Big) = \overline{\hat{\tau}_2\Big(\neg q\Big)} = \overline{\overline{\hat{\tau}_2\Big(q\Big)}} = \overline{\overline{\tau_2(q)}} = \overline{\overline{0}} = \overline{1} = 0.$

Așadar, valoarea de adevăr a formulei $\neg\neg q$ în τ_2 este 0.

Observație. • *În general, nu are sens exprimarea* valoarea de adevăr a unei formule*, deoarece o formulă poate fi adevărată într-o anumită atribuire, dar falsă în altă atribuire. Are sens să spunem* valoarea de adevăr a unei formule într-o atribuire.

• *Din același motiv, nu are sens să spunem* formula este adevărată *sau* formula este falsă*. În schimb, are sens să spunem* formula este adevărată în această atribuire *sau* formula este falsă în această atribuire*. De exemplu, formula $\neg\neg p$ este adevărată în τ_1, dar falsă în τ_2.*

• *Dacă spunem doar* valoarea de adevăr a formulei φ*, înțelegem* o funcție de la mulțimea tuturor atribuirilor la mulțimea B *și nu* o valoare de adevăr din mulțimea B*.*

> În general, nu are sens exprimarea *valoarea de adevăr a unei formule*, deoarece o formulă poate fi adevărată într-o anumită atribuire, dar falsă în altă atribuire. Are sens să spunem *valoarea de adevăr a unei formule într-o atribuire.*

Definiția 32 (Noțiunea de satisfacere)**.** *O* atribuire τ *satisface* φ *dacă* $\hat{\tau}\Big(\varphi\Big) = 1$.

În loc de τ *satisface* φ, putem folosi oricare dintre următoarele forme echivalente:

1. τ este model al formulei φ;
2. φ ține în τ;
3. τ face φ adevărată;

Exemplul 33. *Atribuirea τ_1 definită mai sus este model pentru $\neg(\mathtt{p} \wedge \mathtt{q})$. Atribuirea τ_1 nu este model al formulei $(\neg\mathtt{p} \wedge \mathtt{q})$.*

Scriem $\tau \models \varphi$ (și citim: τ este model al formulei φ; sau: τ satisface φ etc.) ddacă $\hat{\tau}\Big(\varphi\Big) = 1$. Scriem $\tau \not\models \varphi$ (și citim: τ nu este model al formulei φ; sau: τ nu satisface φ) ddacă $\hat{\tau}\Big(\varphi\Big) = 0$.

Definiția 34 (Relația de satisfacere (notată $\models$))**.** *Relația $\models$, dintre atribuiri și formule, se numește* relația de satisfacere. *Relația este definită astfel: $\tau \models \varphi$ ddacă $\hat{\tau}\Big(\varphi\Big) = 1$.*

Relația $\models$ este definită astfel: $\tau \models \varphi$ ddacă $\hat{\tau}\Big(\varphi\Big) = 1$.

5.4 Satisfiabilitate

Definiția 35. *O formulă φ este* satisfiabilă *dacă există cel puțin o atribuire τ astfel încât $\tau \models \varphi$ (i.e., dacă φ are cel puțin un model).*

Exemplul 36. *Formula $(\mathtt{p} \vee \mathtt{q})$ este satisfiabilă, din moment ce are un model (de exemplu, atribuirea τ_1, definită mai sus).*

Exemplul 37. *Formula $\neg\mathtt{p}$ este de asemenea satisfiabilă: de exemplu, atribuirea τ_3, definită mai sus, face formula adevărată.*

Exemplul 38. *Formula $(\mathtt{p} \wedge \neg\mathtt{p})$ nu este satisfiabilă, deoarece are valoarea de adevăr* fals *în orice atribuire.*

Demonstrație: *Considerăm o atribuire $\tau : A \to B$ oarecare.*
Avem că $\hat{\tau}\Big((\mathtt{p} \wedge \neg\mathtt{p})\Big) = \hat{\tau}\Big(\mathtt{p}\Big) \cdot \hat{\tau}\Big(\neg\mathtt{p}\Big) = \tau(\mathtt{p}) \cdot \overline{\hat{\tau}\Big(\mathtt{p}\Big)} = \tau(\mathtt{p}) \cdot \overline{\tau(\mathtt{p})}$.
Dar $\tau(\mathtt{p})$ poate fi 0 sau 1:

1. *în primul caz ($\tau(\mathtt{p}) = 0$), avem că $\hat{\tau}\Big((\mathtt{p} \wedge \neg\mathtt{p})\Big) = \ldots = \tau(\mathtt{p}) \cdot \overline{\tau(\mathtt{p})} = 0 \cdot \overline{0} = 0 \cdot 1 = 0$;*

2. *în al doilea caz* ($\tau(\mathtt{p}) = 1$), *avem că* $\hat{\tau}\Big((\mathtt{p} \wedge \neg\mathtt{p})\Big) = \ldots = \tau(\mathtt{p}) \cdot \overline{\tau(\mathtt{p})} = 1 \cdot \overline{1} = 1 \cdot 0 = 0$.

Observăm că în oricare dintre cele două cazuri, avem că $\hat{\tau}\Big((\mathtt{p} \wedge \neg\mathtt{p})\Big) = 0$. *Dar* τ *a fost o atribuire aleasă arbitrar, și din acest motiv* $(\mathtt{p} \wedge \neg\mathtt{p})$ *este falsă* în orice atribuire τ. *Adică formula este nesatisfiabilă.* *q.e.d.*

Definiția 39 (Contradicție)**.** *O formulă care este nesatisfiabilă se numește* contradicție.

Exemplul 40. *După cum am văzut mai sus,* $(\mathtt{p} \wedge \neg\mathtt{p})$ *este o contradicție.*

5.5 Formule valide

Definiția 41 (Formulă validă)**.** *O formulă* φ *este* validă *dacă orice atribuire* τ *are proprietatea că* $\tau \models \varphi$ *(orice atribuire este model pentru formulă).*

Definiția 42 (Tautologie)**.** *O formulă validă se mai numește și* tautologie.

Exemplul 43. *Formula* $(\mathtt{p} \vee \neg\mathtt{p})$ *este validă, deoarece este adevărată în orice atribuire: fie* τ *o atribuire oarecare; avem că* $\hat{\tau}\Big((\mathtt{p} \vee \neg\mathtt{p})\Big) = \tau(\mathtt{p}) + \overline{\tau(\mathtt{p})}$. *Deoarece* $\tau(\mathtt{p})$ *poate lua doar 2 valori posibile, adică 0 sau 1, vom avea că* $\tau(\mathtt{p}) + \overline{\tau(\mathtt{p})}$ *este sau* $0 + 1$, *sau* $1 + 0$, *deci* 1 *în ambele cazuri.*

Notație. *Faptul că formula* φ *este validă se mai notează cu* $\models \varphi$.

Exemplul 44. *Formula* $\mathtt{p}$ *nu este validă (deoarece există o atribuire, de exemplu* τ_3, *care face formula falsă).*

5.6 Formule satisfiabile, dar care nu sunt valide

Observație. *În engleză, o formulă care nu este nici contradicție și nici tautologie se numește* contingent formula. *În limba română, nu avem un cuvânt precis pentru acest concept, și vom folosi exprimarea* formulă contingentă *(sau* contingență*), supraîncărcând înțelesul cuvântului contingent din DEX.*

Definiția 45. *O formulă care nu este nici contradicție și nici tautologie se numește* contingență.

Fiecare formulă poate fi clasificată ca fiind o contradicție, o tautologie, sau o contingență.

Exemplul 46. *Iată exemple din fiecare astfel de clasă de formule:*

1. $(p \wedge \neg p)$ *este o contradicție;*
2. p *este o contingență;*
3. $(p \vee \neg p)$ *este o tautologie.*

5.7 Formule echivalente

Definiția 47. *Spunem că două formule* $\varphi_1, \varphi_2 \in \mathbb{LP}$ *sunt* echivalente, *și scriem* $\varphi_1 \equiv \varphi_2$, *dacă pentru orice atribuire* $\tau : A \to B$, $\hat{\tau}\left(\varphi_1\right) = \hat{\tau}\left(\varphi_2\right)$.

Intuitiv, formulele care sunt echivalente au același înțeles (adică exprimă același lucru).

Exemplul 48. *La începutul cursului, am văzut că formulele* p *și* $\neg\neg p$ *nu sunt egale din punct de vedere sintactic. Evident că formulele nu sunt egale, deoarece prima are* 1 *simbol, iar cealaltă* 3 *simboluri (dacă ar fi fost egale, ar fi avut același număr de simboluri). Totuși, acum suntem pregătiți să înțelegem relația dintre ele:* $p \equiv \neg\neg p$. *Cu alte cuvinte, chiar dacă nu sunt egale, ele sunt echivalente: exprimă același lucru.*
Pentru a demonstra $p \equiv \neg\neg p$, *este suficient să arătăm că formulele au aceeași valoare de adevăr în orice atribuire. Fie* τ *o atribuire oarecare. Avem că* $\hat{\tau}\left(\neg\neg p\right) = \overline{\overline{\tau(p)}} = \tau(p) = \hat{\tau}\left(p\right)$. *Pe scurt,* $\hat{\tau}\left(\neg\neg p\right) = \hat{\tau}\left(p\right)$. *Din moment ce* τ *a fost aleasă arbitrar, înseamnă că* $\hat{\tau}\left(\neg\neg p\right) = \hat{\tau}\left(p\right)$ *pentru* orice atribuire τ *și de acest motiv* p *este echivalent cu* $\neg\neg p$.

Exemplul 49. *Următoarea echivalență are loc:* $(p \vee q) \equiv \neg(\neg p \wedge \neg q)$ *(exercițiu: verificați că într-adevăr așa este).*

Iată încă două echivalențe importante, cunoscute sub denumirea de *legile lui De Morgan*:

Teorema 50. *Pentru orice formule* $\varphi_1, \varphi_2 \in \mathbb{LP}$, *avem că:*

1. $\neg(\varphi_1 \vee \varphi_2) \equiv (\neg\varphi_1 \wedge \neg\varphi_2)$;
2. $\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1 \vee \neg\varphi_2)$.

5.8 Consecință semantică

Definiția 51. *Fie* $\Gamma = \{\varphi_1, \ldots, \varphi_n, \ldots\}$ *o mulțime (posibil infinită) de formule. Spunem că* φ *este o* consecință semantică *a mulțimii* Γ, *și scriem* $\Gamma \models \varphi$, *dacă orice atribuire care este model pentru toate formulele din* Γ *este model și pentru formula* φ.

Spunem de asemenea că φ *este o consecință logică a mulțimii* Γ sau că φ *este o consecință tautologică a mulțimii* Γ în loc de φ *este o consecință semantică a mulțimii* Γ.

Exemplul 52. *Fie* $\Gamma = \{\mathtt{p}, (\neg\mathtt{p} \vee \mathtt{q})\}$. *Avem că* $\Gamma \models \mathtt{q}$.

Într-adevăr, fie τ *un model al formulelor* $\mathtt{p}$ *și* $(\neg\mathtt{p} \vee \mathtt{q})$. *Din moment ce* τ *este model pentru* $\mathtt{p}$, *avem prin definiție că* $\tau(\mathtt{p}) = 1$.

Din moment ce τ *este model al* $(\neg\mathtt{p} \vee \mathtt{q})$, *avem că* $\hat{\tau}\Big((\neg\mathtt{p} \vee \mathtt{q})\Big) = 1$. *Dar* $\hat{\tau}\Big((\neg\mathtt{p} \vee \mathtt{q})\Big) = \overline{\tau(\mathtt{p})} + \tau(\mathtt{q})$. *Dar* $\tau(\mathtt{p}) = 1$, *și deci* $\hat{\tau}\Big((\neg\mathtt{p} \vee \mathtt{q})\Big) = 0 + \tau(\mathtt{q}) = \tau(\mathtt{q})$. *Înseamnă că* $\tau(\mathtt{q}) = 1$.

Deci τ *este model pentru* $\mathtt{q}$. *Am presupus că* τ *este model pentru* $\mathtt{p}$ *și* $(\neg\mathtt{p} \vee \mathtt{q})$ *și am arătat că în mod necesar* τ *este și model pentru* $\mathtt{q}$. *Dar aceasta este fix definiția faptului că* $\{\mathtt{p}, (\neg\mathtt{p} \vee \mathtt{q})\} \models \mathtt{q}$, *ceea ce voiam să arătăm.*

Exemplul 53. *Avem că* $\{\mathtt{p}, (\mathtt{p} \vee \mathtt{r})\} \not\models \neg\mathtt{r}$, *adică* $\neg\mathtt{r}$ *nu este consecință logică a formulelor* $\mathtt{p}, (\mathtt{p} \vee \mathtt{r})$. *Pentru a arăta această* neconsecință, *este suficient să găsim un model pentru formulele* $\mathtt{p}$ *și* $(\mathtt{p} \vee \mathtt{r})$, *dar care să nu fie model al formulei* $\neg\mathtt{r}$. *Orice atribuire* τ *cu* $\tau(\mathtt{p}) = 1$ *și* $\tau(\mathtt{r}) = 1$ *(de exemplu,* τ_1*) satisface cele două proprietăți.*

Notație. *Câteodată scriem*

$$\varphi_1, \ldots, \varphi_n \models \varphi$$

în loc de

$$\{\varphi_1, \ldots, \varphi_n\} \models \varphi.$$

Observație. *Dacă* $n = 0$, $\varphi_1, \ldots, \varphi_n \models \varphi$ *se reduce la faptul că* φ *este consecință semantică a mulțimii vide. Acest lucru înseamnă că* φ *este validă, ceea ce justifică notația* $\models \varphi$ *pentru faptul că* φ *este validă.*

5.9 Mulțimi consistente de formule

O altă noțiune semantică care apare relativ des în practică și care are o legatură strânsă cu conectorul $\wedge$ este noțiunea de *mulțime (in)consistentă de formule.*

Definiția 54 (Mulțime consistentă de formule)**.** *O mulțime* $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ de formule se numește consistentă *dacă există o atribuire* τ *care să fie model pentru toate formulele* φ_i $(1 \leq i \leq n)$.

Observație. *Atenție! Aceeași atribuire* τ *pentru toate formulele din mulțime (nu câte o atribuire potențial diferită pentru fiecare formulă)!*

O mulțime de formule este inconsistentă (sau neconsistentă) dacă nu este consistentă.

Lema 55 (Legătura dintre mulțimile consistente și conectorul logic $\wedge$)**.** *O mulțime de formule $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ este consistentă ddacă formula*

$$(((\varphi_1 \wedge \varphi_2) \wedge \ldots) \wedge \varphi_n)$$

este satisfiabilă.

Lema 56 (Legătura dintre mulțimile inconsistente și conectorul logic $\wedge$)**.** *O mulțime de formule $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ este inconsistentă ddacă formula*

$$(((\varphi_1 \wedge \varphi_2) \wedge \ldots) \wedge \varphi_n)$$

nu este satisfiabilă.

Exercițiul 57. *Arătați că, pentru orice formulă φ, dacă $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ este o mulțime inconsistentă de formule, avem că*

$$\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \models \varphi.$$

Exercițiul 58. *Arătați că dacă*

$$\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \models (\mathtt{p} \wedge \neg \mathtt{p}),$$

atunci $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ este o mulțime inconsistentă de formule.

5.10 Aplicația 1

Ion scrie următorul cod:

```
if (((y % 4 == 0) && (y % 100 != 0)) || (y % 400 == 0))
   printf("%d is a leap year.", y);
else
   printf("%d is not a leap year.", y);
```

Ioana încearcă să simplifice codul în felul următor:

```
if (((y % 4 != 0) || (y % 100 == 0)) && (y % 400 != 0))
   printf("%d is not a leap year.", y);
else
   printf("%d is a leap year.", y);
```

Transformarea făcută de Ioana păstrează comportamentul programului? E dificil să spunem cu certitudine acest lucru dacă doar ne uităm la cod, dar putem să ne folosim de conceptele pe care le-am învățat pentru a modela problema de mai sus folosind uneltele logicii propoziționale și să determinăm dacă cele două programe au același comportament.

Înainte de toate, vom *traduce* condițiile din instrucțiunea if-else în logica propozițională. Vom identifica *propozițiile atomice* și le vom înlocui cu variabile propoziționale după cum urmează. Fie `y` un an fixat:

1. variabila propozițională $\mathtt{p}$ va ține locul propoziției atomice `(y % 4 == 0)`;
2. variabila propozițională $\mathtt{q}$ va ține locul propoziției atomice `(y % 100 == 0)`;
3. variabila propozițională $\mathtt{r}$ va ține locul propoziției atomice `(y % 400 == 0)`.

Ținând cont de traducerea de mai sus, vedem că condiția din programul lui Ion este, în limbajul logicii propoziționale, $((\mathtt{p} \wedge \neg \mathtt{q}) \vee \mathtt{r})$.

Formula Ioanei este, în limbajul logicii propoziționale, $((\neg \mathtt{p} \vee \mathtt{q}) \wedge \neg \mathtt{r})$.

Observați de asemenea că ramura `if` a primului program corespunde ramurii `else` a celui de-al doilea program și invers. Pentru ca cele două programe să aibă același comportament, este suficient ca negația formulei lui Ion să fie echivalentă cu formula Ioanei. Cu alte cuvinte, are loc echivalența:

$$\neg((\mathtt{p} \wedge \neg \mathtt{q}) \vee \mathtt{r}) \equiv ((\neg \mathtt{p} \vee \mathtt{q}) \wedge \neg \mathtt{r})?$$

Aplicând legile lui De Morgan, vedem că echivalența are într-adevăr loc și deci transformarea propusă de Ioana este corectă.

5.11 Fișă de exerciții

Exercițiul 59. *Fie $\tau : A \to B$ atribuirea definită după cum urmează: $\tau(\mathtt{p}) = 1$, $\tau(\mathtt{q}) = 0$, $\tau(\mathtt{r}) = 0$, $\tau(a) = 0$ pentru orice altă variabilă propozițională $a \in A \setminus \{\mathtt{p}, \mathtt{q}, \mathtt{r}\}$.*

Stabiliți valoarea de adevăr în atribuirea τ de mai sus a următoarelor formule:

1. $(\mathtt{p} \wedge \mathtt{q})$;
2. $(\mathtt{q} \wedge \mathtt{p})$;
3. $\neg \mathtt{q}$;

4. $(\neg q \wedge r)$;
5. $((\neg q \wedge r) \vee \neg p)$.

Exercițiul 60. *Găsiți câte o atribuire τ în care următoarele formule să fie adevărate (câte o atribuire pentru fiecare formulă):*

1. $(p \wedge q)$;
2. $(p \wedge \neg q)$;
3. $((p \wedge \neg q) \vee q)$.

Există o (singură) atribuire care să facă toate cele trei formule de mai sus adevărate?

Exercițiul 61. *Găsiți câte o atribuire τ în care următoarele formule să fie false (câte o atribuire pentru fiecare formulă):*

1. $(p \vee q)$;
2. $(q \wedge (p \vee \neg q))$;
3. $((p \wedge \neg q) \vee q)$.

Exercițiul 62. *Care dintre următoarele formule sunt satisfiabile?*

1. $(p \wedge \neg p)$;
2. $(p \vee \neg p)$;
3. $((p \vee \neg p) \wedge \neg q)$;
4. $((p \vee \neg p) \wedge (\neg p \wedge q))$;
5. $((p \vee \neg q) \wedge (\neg p \vee r))$.

Exercițiul 63. *Care dintre următoarele formule sunt valide?*

1. $(p \wedge \neg p)$;
2. $(p \vee \neg p)$;
3. p;
4. $((p \vee \neg p) \wedge \neg q)$;
5. $((p \wedge q) \vee (\neg p \wedge r))$.

Exercițiul 64. *Dați exemple de 5 contradicții interesante.*

Exercițiul 65. *Dați exemple de 5 tautologii interesante.*

Exercițiul 66. *Demonstrați că, pentru orice formule $\varphi_1, \varphi_2, \varphi_3 \in \mathbb{LP}$, au loc următoarele echivalențe:*

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

Exercițiul 67. *Putem demonstra că, pentru orice formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, $(\varphi_1 \vee \varphi_2) \equiv \varphi_1$ dacă și numai dacă $\varphi_1 \in \mathbb{LP}$ este tautologie?*

Exercițiul 68. *Putem demonstra că, pentru orice formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, $(\varphi_1 \wedge \varphi_2) \equiv \varphi_2$ dacă și numai dacă $\varphi_1 \in \mathbb{LP}$ este tautologie?*

Exercițiul 69. *Putem demonstra că, pentru orice formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, $(\varphi_1 \wedge \varphi_2) \equiv \varphi_1$ dacă și numai dacă $\varphi_1 \in \mathbb{LP}$ este contradicție?*

Exercițiul 70. *Putem demonstra că, pentru orice formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, $(\varphi_1 \vee \varphi_2) \equiv \varphi_2$ dacă și numai dacă $\varphi_1 \in \mathbb{LP}$ este contradicție?*

Exercițiul 71. *Arătați că $\neg\mathtt{p}$ este consecință semantică din $(\neg\mathtt{p} \vee \neg\mathtt{p})$.*

Exercițiul 72. *Arătați că $\mathtt{p}$ nu este consecință semantică din $(\neg\mathtt{q} \vee \mathtt{p})$.*

Exercițiul 73. *Arătați că $\mathtt{p}$ este consecință semantică din $(\neg\mathtt{q} \vee \mathtt{p})$ și $\mathtt{q}$.*

Exercițiul 74. *Arătați că $\mathtt{p}_3$ este consecință semantică din $(\neg\mathtt{p}_1 \vee (\mathtt{p}_2 \vee \mathtt{p}_3))$, $((\neg\neg\mathtt{p}_2 \vee \neg\mathtt{p}_4) \wedge (\neg\neg\mathtt{p}_4 \vee \neg\mathtt{p}_2))$ și $(\mathtt{p}_1 \wedge \neg\mathtt{p}_4)$.*

Capitolul 6

Conectori logici

Mai există doi conectori logici importanți în logica propozițională: *implicația* și *echivalența*. Conectorul *echivalență* este numit de asemenea *dublă implicație*.

Semantica unei implicații $(\varphi_1 \to \varphi_2)$ este dată de

$$\hat{\tau}\Big((\varphi_1 \to \varphi_2)\Big) = \overline{\hat{\tau}\Big(\varphi_1\Big)} + \hat{\tau}\Big(\varphi_2\Big),$$

pentru orice atribuire τ. Se poate observa ușor că

$$(\varphi_1 \to \varphi_2) \equiv (\neg\varphi_1 \vee \varphi_2).$$

Similar, semantica unei duble implicații $(\varphi_1 \leftrightarrow \varphi_2)$ este definită astfel încât

$$(\varphi_1 \leftrightarrow \varphi_2) \equiv ((\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)).$$

Exemplul 75. *Avem că* $(\mathtt{p} \to \mathtt{p})$ *este o formulă validă. De ce? Formula* $(\mathtt{p} \to \mathtt{p})$ *este echivalentă* $(\neg\mathtt{p} \vee \mathtt{p})$*, despre care putem vedea imediat că este validă.*

6.1 Mai multe logici propoziționale

Până în acest moment am studiat logica propozițională a conectorilor $\neg, \wedge, \vee$, pe care am notat-o cu $\mathbb{LP}$. De fapt, în funcție de mulțimea conectorilor logici de care avem nevoie, există mai multe logici propoziționale. Logica pe care am studiat-o până în acest moment o vom nota cu $\mathbb{LP}_{\neg,\wedge,\vee} = \mathbb{LP}$.

În funcție de conectorii logici permiși, putem obține și alte logici interesante:

1. $\mathbb{LP}_{\neg,\vee}$ este logica propozițională în care singurii conectori permiși sunt $\neg$ și $\vee$.

2. $\mathbb{LP}_{\bot,\to}$ este logica propozițională în care singurii conectori permiși sunt $\bot$ (conector de aritate 0) și $\to$. Formula $\bot$ (citire: *bottom* sau *T întors*) este falsă în orice atribuire.

3. $\mathbb{LP}_{\vee,\wedge}$ este o logică în care singurii conectori permiși sunt $\vee$ și $\wedge$.

Exercițiul 76. *Scrieți definiția formală pentru sintaxa fiecăreia dintre logicile de mai sus* [1].

Ce au în comun $\mathbb{LP}, \mathbb{LP}_{\neg,\vee}, \mathbb{LP}_{\bot,\to}$? Sunt *echiexpresive*. Adică pentru orice formulă $\varphi \in \mathbb{LP}$ există o formulă $\varphi' \in \mathbb{LP}_{\neg,\vee}$ astfel încât $\varphi \equiv \varphi'$ (și invers, pentru orice formulă $\varphi' \in \mathbb{LP}_{\neg,\vee}$ există o formulă echivalentă în $\mathbb{LP}$).

Cum putem arăta că $\mathbb{LP}$ și $\mathbb{LP}_{\neg,\vee}$ sunt la fel de expresive? Pentru una dintre direcții, este suficient să traducem toate conjuncțiile posibile din $\mathbb{LP}$ în felul următor:

$$(\varphi \wedge \varphi') \equiv \neg(\neg\varphi \vee \neg\varphi').$$

La sfârșit vom obține o formulă echivalentă care nu conține conjuncții (și deci este în $\mathbb{LP}_{\neg,\vee}$). Invers, orice formulă din $\mathbb{LP}_{\neg,\vee}$ este deja și formulă din $\mathbb{LP}$.

Cum putem arăta că $\mathbb{LP}_{\neg,\vee}$ și $\mathbb{LP}_{\bot,\to}$ sunt la fel de expresive?

Transformăm toate disjuncțiile și negațiile folosind următoarele echivalențe:

1. $(\varphi \vee \varphi') \equiv (\neg\varphi \to \varphi')$;

2. $\neg\varphi \equiv (\varphi \to \bot)$.

Operația de transformare se oprește după un număr finit de pași și rezultatul este o formulă echivalentă cu formula de la care am plecat, dar care nu conține decât conectorii $\bot$ și $\to$.

Logica $\mathbb{LP}_{\vee,\wedge}$ este strict mai puțin expresivă. De exemplu, în această logică nu există formule nesatisfiabile.

Exercițiul 77. *Explicați de ce în $\mathbb{LP}_{\vee,\wedge}$ toate formulele sunt satisfiabile.*

Observație. *Prin* logică propozițională *se înțelege orice logică care este la fel de expresivă ca $\mathbb{LP}$. De exemplu, $\mathbb{LP}_{\neg,\wedge}$, $\mathbb{LP}_{\neg,\vee}$, $\mathbb{LP}_{\neg,\to}$ sunt toate logici propoziționale, dar $\mathbb{LP}_{\neg}$ și $\mathbb{LP}_{\wedge,\vee}$ nu sunt logici propoziționale (sunt mai puțin expresive).*

[1] Pentru conectorul $\bot$ sunt două posibilități, la alegere: să considerați că este folosit într-un pas de bază sau într-un pas inductiv. Distincția dintre un caz de bază și un caz inductiv este pur didactică (face definițiile inductive mai ușor de abordat), nu fundamentală. Totuși, pentru consecvență, vom considera că $\bot$ este folosit într-un al doilea caz de bază.

6.2 Legătura dintre implicație și consecință semantică

Există o legătură importantă între *conectorul logic implicație* și noțiunea de *consecință semantică*, dată de următoarea teoremă.

Teorema 78 (Legătura dintre implicație și consecință semantică)**.** *Pentru orice două formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, avem că*

$$\varphi_1 \models \varphi_2 \text{ (formula } \varphi_2 \text{ este consecință a formulei } \varphi_1)$$

dacă și numai dacă

$$\models (\varphi_1 \to \varphi_2) \text{ (formula } (\varphi_1 \to \varphi_2) \text{ este validă).}$$

De asemenea, are loc următoarea teoremă mai generală:

Teorema 79 (Legătura generalizată dintre implicație și consecință semantică)**.** *Pentru orice formule $\varphi_1, \varphi_2, \ldots, \varphi_n, \varphi \in \mathbb{LP}$, avem că*

$$\varphi_1, \varphi_2, \ldots, \varphi_n \models \varphi$$

dacă și numai dacă

$$\models ((((\varphi_1 \wedge \varphi_2) \wedge \ldots) \wedge \varphi_n) \to \varphi).$$

O legătură similară avem între conectorul logic echivalență și noțiunea de echivalență semantică:

Teorema 80 (Legătura dintre conectorul echivalență și noțiunea semantică de echivalență)**.** *Pentru orice două formule $\varphi_1, \varphi_2 \in \mathbb{LP}$, avem că*

$$\varphi_1 \equiv \varphi_2$$

dacă și numai

$$\models (\varphi_1 \leftrightarrow \varphi_2).$$

6.3 *Traducerea* propozițiilor din limba română în $\mathbb{LP}$

Prin *traducere* se înțelege modelarea unei propoziții scrisă în limba română ca o formulă din logica propozițională. Scopul modelării poate fi: clarificarea înțelesului propoziției prin eliminarea ambiguităților sintactice, verificarea că propoziția este validă, etc.

Pentru a *traduce* propoziții din limba română în logica propozițională, trebuie să urmăm doi pași:

1. Identificarea propozițiilor atomice și asocierea cu variabile propoziționale;

2. Identificarea conectorilor logici și a ordinii acestora.

De exemplu, să presupunem că vrem să *traducem* propoziția *vreau să învăț la logică și să trec examenul dacă materia este interesantă* în logica propozițională.

Primul pas este identificarea propozițiilor atomice. În cazul nostru, găsim trei propoziții atomice:

1. *vreau să învăț la Logică*;

2. *vreau să trec examenul*; (atenție, cuvântul *vreau* nu apare explicit în propoziție)

3. *materia este interesantă*.

> Atenție! Conectorii nu fac parte din propozițiile atomice. De exemplu, a treia propoziție atomică nu este *dacă materia este interesantă*.

Asociem câte o variabilă propozițională cu fiecare propoziție atomică:

variabilă propozițională	propoziție atomică
`p`	*vreau să învăț la Logică*
`q`	*vreau să trec examenul*
`r`	*materia este interesantă*.

> Atenție! Pentru o *traducere* cât mai fidelă, dacă o propoziție apare de mai multe ori (chiar dacă nu exact cu aceleași cuvinte), trebuie să îi asociem aceeași variabilă propozițională.

Următorul pas este identificarea conectorilor logici. În exemplul de mai sus, apar doi conectori logici:

1. conectorul *și* în contextul *[...] la logică și să trec [...]*, care indică o conjuncție;

2. conectorul *dacă-atunci* (chiar dacă cuvântul *atunci* nu apare explicit) în contextul *[...] examenul dacă materia este [...]*, care indică o implicație.

Care este ordinea celor doi conectori? Cu alte cuvinte, propoziția este o conjuncție sau o implicație? Care este conectorul principal? Simțul limbii române ne spune că înțelesul este mai degrabă o implicație (de exemplu, din cauza faptului că verbul *vreau* este subînțeles în propoziția *vreau să trec examenul*). *Traducerea* este așadar

$$(\mathtt{r} \to (\mathtt{p} \wedge \mathtt{q})),$$

deoarece propoziția asociată variabilei propoziționale $\mathtt{r}$ este antecedentul implicației, chiar dacă apare sintactic la sfârșitul propoziției.

> Atenție! În ciuda denumirii, variabilele propoziționale nu sunt variabile în sens matematic. O greșeală frecventă este să scriem
>
> $$\mathtt{p} = \textit{vreau să învăț la Logică},$$
>
> ceea ce nu este corect, deoarece $\mathtt{p}$ este egal doar cu $\mathtt{p}$ și cu nimic altceva. Orice variabilă din curs apare scrisă cu font negru obișnuit. De exemplu, de multe ori folosim variabila φ pentru a ține locul unei formule.

Exercițiul 81. *Traduceți următoarea propoziție în logica propozițională:* Sau trec la Logică, sau nu trec.

Atenție la cuvintele care nu apar explicit în propozițiile atomice. Atenție la semnificația conectorului sau-sau *(indiciu: nu apare printre conectorii pe care i-am discutat, dar poate fi* emulat/simulat*).*

6.4 Aplicația 2

Următorul puzzle este preluat din cartea *Peter Smith. An Introduction to Formal Logic*: Fie majordomul, fie bucătarul a comis crima. Victima a murit otrăvită dacă bucătarul a comis crima. Majordomul a comis crima doar dacă victima a fost înjunghiată. Victima nu a murit otrăvită. Rezultă că victima a fost înjunghiată?

Asociem fiecărei propoziții atomice o variabilă propozițională, după cum urmează:

1. Pentru propoziția *majordomul a comis crima* variabila $\mathtt{p}$;
2. Pentru propoziția *bucătarul a comis crima* variabila $\mathtt{q}$;
3. Pentru propoziția *victima a murit otrăvită* variabila $\mathtt{r}_1$;

4. Pentru propoziția *victima a fost înjunghiată* variabila r_2.

Ipotezele puzzle-ului se modelează în logica propozițională ca următoarele formule:

1. $((p \vee q) \wedge \neg(p \wedge q))$ (disjuncția exclusivă dintre p și q);
2. $(q \to r_1)$;
3. $(p \to r_2)$ (atenție la sensul implicației);
4. $\neg r_1$.

Întrebarea este modelată de formula r_2.
Pentru a răspunde la puzzle cu da/nu, este suficient să verificăm dacă

$$\Big\{((p \vee q) \wedge \neg(p \wedge q)), (q \to r_1), (p \to r_2), \neg r_1\Big\} \models r_2.$$

Într-adevăr consecința semantică are loc (exercițiu).

6.5 Fișă de exerciții

Exercițiul 82. *Asociați pentru fiecare dintre afirmațiile următoare o formulă din $\mathbb{LP}$ care să modeleze înțelesul său din limba română.*

1. *Dacă afară plouă, stau în casă sau merg în mall. Nu stau în casă doar dacă nu mă plictisesc. Afară plouă și nu mă plictisesc.*
2. *Lucrez la logică doar dacă nu se poate ieși afară. Se poate ieși afară dacă nu plouă și este cald. Din moment ce nu lucrez la logică și afară este cald, înseamnă că plouă.*
3. *Lucrurile merg bine în țară dacă la conducerea țării nu sunt hoți și economia este sănătoasă. Oamenii pleacă în străinătate dacă și numai dacă lucrurile nu merg bine în țară. Economia este sănătoasă, dar oamenii pleacă în străinătate.*

Capitolul 7

Deducția Naturală

În capitolul anterior am discutat câteva noțiuni importante care țin de *semantica* logicii propoziționale:

1. valoarea de adevăr a unei formule într-o atribuire;

2. satisfiabilitate;

3. validitate;

4. echivalență;

5. consecință semantică.

Am văzut că pentru a stabili, de exemplu, că două formule sunt echivalente, este necesar un raționament nu foarte complicat, dar totuși un raționament la nivel semantic (adică raționament care folosește noțiunile semantice de valoare de adevăr, atribuire etc.). A priori, un astfel de raționament este propriu oamenilor.

Una dintre preocupările principale în logica pentru informatică este crearea de metode mecanice (adică metode pretabile implementării pe calculator) pentru raționamentele semantice aferente.

În acest capitol, prezentăm o metodă pentru mecanizarea noțiunii de consecință semantică. Prin mecanizare a noțiunii de consecință semantică înțelegem o metodă de a demonstra consecințe cu următoarele proprietăți:

1. să putem convinge pe cineva să accepte că consecința are loc, fără ca acea persoană să fie nevoită să urmeze un raționament semantic;

2. în particular, fiecare pas al demonstrației trebuie să poate fi verificat ușor (mecanic, fără a înțelege);

3. metoda trebuie să fie precisă, într-atât de precisă încât raționamentul să poată fi verificat de un calculator.

7.1 Secvențe

Definiția 83 (Secvență)**.** *O* secvență *este o pereche formată dintr-o mulțime de formule* $\{\varphi_1, \ldots, \varphi_n\}$ *și dintr-o formulă* φ*, pereche notată sub forma*

$$\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi.$$

Formulele $\varphi_1, \ldots, \varphi_n$ și φ sunt din mulțimea $\mathbb{LP}_{\neg,\wedge,\vee,\rightarrow,\bot}$.

Câteodată citim notația $\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi$ sub forma φ *este consecință sintactică din* $\{\varphi_1, \ldots, \varphi_n\}$. Formulele $\varphi_1, \ldots, \varphi_n$ se numesc antecedenții secvenței, iar φ consecventul secvenței.

De multe ori, vom nota cu $\Gamma = \{\varphi_1, \ldots, \varphi_n\}$ mulțimea de antecedenți și în acest caz vom scrie secvența astfel: $\Gamma \vdash \varphi$. De asemenea, notația uzuală în literatură permite scrierea $\varphi_1, \ldots, \varphi_n \vdash \varphi$ (adică fără acolade) în loc de $\{\varphi_1, \ldots, \varphi_n\} \vdash \varphi$, dar trebuie să ținem cont că în partea stângă a simbolului $\vdash$ avem de fapt o mulțime.

Exemplul 84. *Iată câteva exemple de secvențe:*

1. $\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{q})$*;*
2. $\{\mathtt{p}, (\mathtt{q} \wedge \neg \mathtt{r})\} \vdash (\mathtt{p} \vee \mathtt{q})$*;*
3. $\{(\mathtt{p} \wedge \mathtt{q}), (\mathtt{p} \vee \mathtt{q})\} \vdash (\mathtt{p} \wedge \mathtt{r})$.

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

7.2 Reguli de inferență

Definiția 85. *O* regulă de inferență *este un tuplu format din:*

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

O regulă de inferență se notează în felul următor:

$$\text{NUME}\ \frac{S_1 \qquad \ldots \qquad S_n}{S}\ \textit{condiție.}$$

Observație. *Este posibil ca o regulă de inferență să aibă $n = 0$ ipoteze. Astfel de reguli de inferență, cu 0 ipoteze, se numesc* axiome.

Observație. *De asemenea, este posibil să lipsească condiția de aplicare.*

Exemplul 86. *Iată câteva exemple de reguli de inferență:*

$$\wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

Toate cele trei reguli de inferență de mai sus sunt corecte*, într-un sens pe care îl vom defini mai târziu. 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:*

$$\textsc{Ipoteză}\ \frac{}{\Gamma \vdash \varphi}\ \varphi \in \Gamma.$$

Iată un exemplu de o regulă de inferență incorectă (vom vedea mai târziu în ce sens este incorectă):

$$\textsc{regulă incorectă}\ \frac{\Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi').}$$

Observație. *Pentru a fi preciși, ipotezele regulii de inferență, precum și concluzia, sunt de fapt* scheme *de secvențe și nu secvențe. Acest lucru înseamnă că o regulă de inferență are mai multe* instanțe*, obținute prin înlocuirea variabilelor matematice $\varphi, \varphi', \Gamma$ cu formule concrete. De exemplu, iată două instanțe ale regulii $\wedge i$ de mai sus:*

$$\wedge i\ \frac{\{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{p} \qquad \{\mathtt{p}, \mathtt{q}\} \vdash \mathtt{q}}{\{\mathtt{p}, \mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{q});} \qquad \wedge i\ \frac{\{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{q}) \qquad \{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash \mathtt{p}}{\{\mathtt{p}, \mathtt{q}, \mathtt{r}\} \vdash ((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{p}).}$$

În prima instanță, am înlocuit variabila matematică Γ cu mulțimea de formule $\{\mathtt{p}, \mathtt{q}\}$, variabila matematică φ cu formula $\mathtt{p}$ și variabila matematică φ' cu formula $\mathtt{q}$.

Exercițiul 87. *Stabiliți cu ce am înlocuit fiecare variabilă în cea de-a doua instanță.*

Exercițiul 88. *Explicați de ce următoarea regulă nu este instanță a regulii* $\wedge i$*:*

$$\wedge i\ \frac{\{\mathtt{p},\mathtt{q}\} \vdash \mathtt{p} \qquad \{\mathtt{p},\mathtt{q}\} \vdash \mathtt{q}}{\{\mathtt{p},\mathtt{q}\} \vdash (\mathtt{p} \wedge \mathtt{p}).}$$

7.3 Sistem deductiv

Definiția 89. *Un* sistem deductiv *este o mulțime de reguli de inferență.*

Exemplul 90. *Fie sistemul deductiv* D_1*, format din următoarele patru reguli de inferență:*

$$\textsc{Ipoteză}\ \frac{}{\Gamma \vdash \varphi,}\ \varphi \in \Gamma \qquad \wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

7.4 Demonstrație formală

Definiția 91 (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 pentru fiecare $1 \leq i \leq n$*,*

> S_i *este concluzia unei instanțe a unei reguli de inferență din sistemul deductiv care folosește ca ipoteze doar secvențe dintre* $S_1, \ldots, S_{i-1}$ *și condiția regulii este adevărată (dacă regula de inferență are condiție).*

Exemplul 92. *Iată un exemplu de demonstrație formală în sistemul* D_1 *definit mai sus:*

1. $\{\mathtt{p},\mathtt{q}\} \vdash \mathtt{p}$; *(*Ipoteză*)*

2. $\{p, q\} \vdash q$; *(Ipoteză)*

3. $\{p, q\} \vdash (p \wedge q)$; *($\wedge i$, 1, 2)*

4. $\{p, q\} \vdash (q \wedge (p \wedge q))$. *($\wedge i$, 2, 3)*

Observați că fiecare linie este adnotată cu numele regulii de inferență aplicate, precum și cu liniile la care se găsesc ipotezele necesare aplicării.

Definiția 93 (Secvență validă)**.** *O secvență $\Gamma \vdash \varphi$ este* validă într-un sistem deductiv D *dacă există o demonstrație formală $S_1, \ldots, S_n$ în D astfel încât $S_n = \Gamma \vdash \varphi$.*

Exemplul 94. *Secvența $\{p, q\} \vdash (p \wedge q)$ este validă în sistemul deductiv D_1 de mai sus, deoarece este ultima secvență din următoarea demonstrație formală:*

1. $\{p, q\} \vdash p$; *(Ipoteză)*

2. $\{p, q\} \vdash q$; *(Ipoteză)*

3. $\{p, q\} \vdash (p \wedge q)$. *($\wedge i$, 1, 2)*

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

7.5 Deducția naturală

Deducția naturală este un sistem deductiv pentru $\mathbb{LP}_{\neg,\wedge,\vee,\rightarrow,\perp}$. În acest sistem deductiv, fiecare conector logic are una sau mai multe reguli de introducere și una sau mai multe reguli de eliminare.

7.5.1 Regulile pentru conjuncții

Am văzut deja regulile de introducere și de eliminare pentru conectorul *și*:

$$\wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,} \qquad \wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi'.}$$

Acest sistem deductiv se numește deducție *naturală* deoarece regulile de inferență mimează raționamentul uman:

• Regula de introducere a conectorului *și* ne indică că putem demonstra o conjuncție $(\varphi \wedge \varphi')$ din antecedenții Γ dacă știm deja că fiecare parte a conjuncției, φ și respectiv φ', sunt consecințe ale formulelor din Γ. 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.

• Pentru conectorul *și* avem două reguli de eliminare. Prima regulă de eliminare a conectorului *și* ne indică faptul că dacă am stabilit deja că o conjuncție $(\varphi \wedge \varphi')$ este consecință a formulelor din mulțimea Γ, atunci și partea stângă a conjuncției, φ, este consecință a mulțimii Γ. A doua regulă este simetrică față de prima și ne permite să concluzionăm că a doua parte a unei conjuncții este consecința unei mulțimi de ipoteze dacă conjuncția este consecință a aceleiași mulțimi de formule.

Iată un exemplu de demonstrație formală care utilizează regulile de inferență pentru conectorul *și*.

1. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{q})$; (Ipoteză)
2. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash \mathtt{r}$; (Ipoteză)
3. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash \mathtt{p}$; $(\wedge e_1,\ 1)$
4. $\{(\mathtt{p} \wedge \mathtt{q}), \mathtt{r}\} \vdash (\mathtt{p} \wedge \mathtt{r})$. $(\wedge i,\ 3,\ 2)$

Exercițiul 95. *Arătați că următoarele secvențe sunt valide:*

1. $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r})\} \vdash (\mathtt{q} \wedge \mathtt{r})$;

2. $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r}), \mathtt{r}'\} \vdash (\mathtt{r}' \wedge \mathtt{q})$;

3. $\{((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r})\} \vdash ((\mathtt{r} \wedge \mathtt{q}) \wedge \mathtt{p})$.

7.5.2 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:

$$\to e\ \frac{\Gamma \vdash (\varphi \to \varphi') \qquad \Gamma \vdash \varphi}{\Gamma \vdash \varphi'.}$$

Regula ne indică că, presupunând că știm $(\varphi \to \varphi')$ (din Γ) și în plus știm și φ (tot din Γ), atunci știm și φ' (din Γ).

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

1. $\{(\mathtt{p} \to \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash (\mathtt{p} \wedge \mathtt{q})$; (Ipoteză)
2. $\{(\mathtt{p} \to \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash \mathtt{p}$; $(\wedge e_1,\ 1)$
3. $\{(\mathtt{p} \to \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash (\mathtt{p} \to \mathtt{r})$; (Ipoteză)
4. $\{(\mathtt{p} \to \mathtt{r}), (\mathtt{p} \wedge \mathtt{q})\} \vdash \mathtt{r}$. $(\to e,\ 3,\ 2)$

Această demonstrație formală arată că secvența $\{(p \to r), (p \wedge q)\} \vdash r$ este validă, adică faptul că formula r este o consecință a formulelor $(p \to r)$ și $(p \wedge q)$. De observat ordinea în care apar liniile 3 și 2 în explicația liniei 4 (urmează aceeași ordine cu regula de inferență).

Exercițiul 96. *Arătați că următoarele secvențe sunt valide:*

1. $\{((p \wedge q) \to r), p, q\} \vdash r$;
2. $\{(p \to r), p, q\} \vdash (q \wedge r)$.

Regula de introducere a implicației este mai subtilă. Pentru a arăta că o implicație $(\varphi \to \varphi')$ decurge din Γ, *presupunem* φ (în plus față de Γ) și arătăm φ'. Cu alte cuvinte, în ipoteza regulii, adăugăm formula φ la formulele din Γ. Regula poate fi scrisă în două moduri echivalente, care se deosebesc doar prin faptul că prima regulă folosește convenția de notație referitoare la acoladele din jurul premiselor unei secvențe, în timp ce în a doua regulă acoladele care marchează mulțimea apar explicit:

$$\to i \ \frac{\Gamma, \varphi \vdash \varphi'}{\Gamma \vdash (\varphi \to \varphi')}, \qquad \to i \ \frac{\Gamma \cup \{\varphi\} \vdash \varphi'}{\Gamma \vdash (\varphi \to \varphi')}.$$

Ce este important de observat și de înțeles la regula de introducere a implicației este că premisele secvenței se schimbă de la concluzia regulii la ipoteza regulii. Dacă în concluzie avem că formula $(\varphi \to \varphi')$ decurge din Γ, în ipoteză trebuie să arătăm că φ' decurge din premisele $\Gamma \cup \{\varphi\}$. Cu alte cuvinte, la modul intuitiv, pentru a demonstra o implicație $(\varphi \to \varphi')$, presupunem antecedentul φ și arătăm consecventul φ'.

Exemplul 97. *Să arătăm că secvența* $\{\} \vdash (p \to p)$ *este validă:*

1. $\{p\} \vdash p$; *(Ipoteză)*
2. $\{\} \vdash (p \to p)$. *$(\to i,\ 1)$*

Exemplul 98. *Să arătăm că secvența* $\{(p \to q)\} \vdash (p \to q)$ *este validă. O demonstrație simplă este:*

1. $\{(p \to q)\} \vdash (p \to q)$. *(Ipoteză)*

O altă demonstrație formală pentru aceeași secvență, demonstrație puțin mai lungă, este:

1. $\{(p \to q), p\} \vdash (p \to q)$; *(Ipoteză)*
2. $\{(p \to q), p\} \vdash p$; *(Ipoteză)*

3. $\{(p \to q), p\} \vdash q$; *($\to e$, 1, 2)*

4. $\{(p \to q)\} \vdash (p \to q)$. *($\to i$, 3)*

Exemplul 99. *Să arătăm că secvența* $\{(p \to q), (q \to r)\} \vdash (p \to r)$ *este validă:*

1. $\{(p \to q), (q \to r), p\} \vdash (p \to q)$; *(Ipoteză)*

2. $\{(p \to q), (q \to r), p\} \vdash p$; *(Ipoteză)*

3. $\{(p \to q), (q \to r), p\} \vdash q$; *($\to e$, 1, 2)*

4. $\{(p \to q), (q \to r), p\} \vdash (q \to r)$; *(Ipoteză)*

5. $\{(p \to q), (q \to r), p\} \vdash r$; *($\to e$, 4, 3)*

6. $\{(p \to q), (q \to r)\} \vdash (p \to r)$. *($\to i$, 5)*

Exercițiul 100. *Arătați că următoarele secvențe sunt valide:*

1. $\{((p \wedge q) \to r)\} \vdash (p \to (q \to r))$;

2. $\{(p \to (q \to r))\} \vdash ((p \wedge q) \to r)$.

7.5.3 Regulile pentru disjuncții

Conectorul *sau* are două reguli de introducere:

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

Prima regulă ne indică că dacă știm φ_1 (din Γ), atunci știm și $(\varphi_1 \vee \varphi_2)$ (din Γ), indiferent de φ_2. Cu alte cuvinte, intuitiv, dacă știm φ_1, știm și $(\varphi_1 \vee$ *orice altceva*$)$. A doua regulă de eliminare este simetrică, pentru partea dreapta a disjuncției.

Exemplul 101. *Să arătăm că secvența* $\{(p \wedge q)\} \vdash (p \vee q)$ *este validă:*

1. $\{(p \wedge q)\} \vdash (p \wedge q)$; *(Ipoteză)*

2. $\{(p \wedge q)\} \vdash p$; *($\wedge e_1$, 1)*

3. $\{(p \wedge q)\} \vdash (p \vee q)$. *($\vee i_1$, 2)*

O altă demonstrație formală pentru aceeași secvență este:

1. $\{(p \wedge q)\} \vdash (p \wedge q)$; *(Ipoteză)*

2. $\{(p \wedge q)\} \vdash q$; *($\wedge e_2$, 1)*

3. $\{(p \wedge q)\} \vdash (p \vee q)$. *($\vee 2_1$, 2)*

Exercițiul 102. *Arătați că secvența* $\{(p \wedge q)\} \vdash (r \vee p)$ *este validă.*

Regula de eliminare a disjuncției este ușor mai complicată, fiind o altă regulă în care mulțimea de antecedenți ale secvențelor variază de la ipoteză la concluzie:

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

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 Γ); cu alte cuvinte, măcar una dintre formulele φ_1 și respectiv φ_2 decurge din Γ. Ipotezele 2 și 3 ne indică faptul că, indiferent care dintre formulele φ_1 și respectiv φ_2 ar avea loc, în orice caz φ' are loc. Adică dacă presupunem φ_1 (în plus față de Γ), φ' are loc, iar dacă presupunem φ_2 (în plus față de Γ), φ' tot are loc. Și atunci concluzia ne indică că φ' are loc indiferent care dintre φ_1 și respectiv φ_2 ar avea loc.

Exemplul 103. *Să arătăm că secvența* $\{(p \vee q)\} \vdash (q \vee p)$ *este validă:*

1. $\{(p \vee q), p\} \vdash p$; *(Ipoteză)*

2. $\{(p \vee q), p\} \vdash (q \vee p)$; *($\vee i_2$, 1)*

3. $\{(p \vee q), q\} \vdash q$; *(Ipoteză)*

4. $\{(p \vee q), q\} \vdash (q \vee p)$; *($\vee i_1$, 3)*

5. $\{(p \vee q)\} \vdash (p \vee q)$; *(Ipoteză)*

6. $\{(p \vee q)\} \vdash (q \vee p)$. *($\vee e$, 5, 2, 4)*

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

Exercițiul 104. *Arătați că secvența* $\{(p \vee q), (p \to r), (q \to r)\} \vdash r$ *este validă.*

Exercițiul 105. *Arătați că secvența* $\{(p \to r), (q \to r)\} \vdash ((p \vee q) \to r)$ *este validă.*

7.5.4 Regulile pentru negații

Regulile pentru introducerea și respectiv eliminarea negației vin la pachet cu o regulă pentru eliminarea conectorului $\bot$:

$$\neg i\ \frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi,} \qquad \neg e\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \neg\varphi}{\Gamma \vdash \bot,} \qquad \bot e\ \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi.}$$

Să ne readucem aminte că $\bot$ este un conector logic 0-ar (de aritate 0), adică conectează 0 formule între ele. Cu alte cuvinte, conectorul $\bot$ este de sine stătător o formulă. Semantica formulei $\bot$ este că este falsă în orice atribuire. Cu alte cuvinte, $\bot$ este o contradicție.

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

A doua regulă, pentru eliminarea negației, ne indică faptul că dacă atât o formulă φ, cât și negația sa, $\neg\varphi$, decurg din aceeași mulțime de premise Γ, atunci din Γ decurge și o contradicție, $\bot$. Vom vedea că o mulțime Γ din care decurge o contradicție este o mulțime *inconsistentă* de formule.

A treia regulă indică că dacă Γ este o mulțime inconsistentă de formule, atunci orice formulă φ decurge din Γ. Această regulă este denumită *principiul exploziei* (dacă dintr-o mulțime de premise pot deduce o contradicție, pot deduce *orice*) sau *ex falso quodlibet* (din fals, orice)[1].

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

Exemplul 106. *Să arătăm că secvența* $\{\mathtt{p}\} \vdash \neg\neg\mathtt{p}$ *este validă:*

1. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{p}$; *(*Ipoteză*)*
2. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \neg\mathtt{p}$; *(*Ipoteză*)*
3. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \bot$; *($\neg e$, 1, 2)*
4. $\{\mathtt{p}\} \vdash \neg\neg\mathtt{p}$. *($\neg i$, 3)*

Exemplul 107. *Să arătăm că secvența* $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{r}$ *este validă:*

1. $\{\mathtt{p}, \neg\mathtt{p}\} \vdash \mathtt{p}$; *(*Ipoteză*)*

[1] Există logici care resping acest principiu (de exemplu, *logica minimală*), dar aceste logici depășesc cadrul cursului.

2. $\{p, \neg p\} \vdash \neg p$; *(Ipoteză)*
3. $\{p, \neg p\} \vdash \bot$; *($\neg e$, 1, 2)*
4. $\{p, \neg p\} \vdash r$. *($\bot e$, 3)*

Exercițiul 108. *Arătați că următoarele secvențe sunt valide:*

1. $\{(p \vee q)\} \vdash \neg(\neg p \wedge \neg q)$;
2. $\{(p \wedge q)\} \vdash \neg(\neg p \vee \neg q)$;
3. $\{(\neg p \vee \neg q)\} \vdash \neg(p \wedge q)$;
4. $\{(\neg p \wedge \neg q)\} \vdash \neg(p \vee q)$;
5. $\{\neg(p \vee q)\} \vdash (\neg p \wedge \neg q)$.

7.5.5 Alte reguli

O altă regulă utilă (în special pentru demonstrarea regulilor derivate – despre care vom discuta în curând), care nu ține de un anumit conector, este regula:

$$\textsc{Extindere}\ \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi.}$$

Această regulă ne indică faptul că, daca φ este consecință a unei mulțimi de formule Γ, atunci φ este consecință și a mulțimii $\Gamma \cup \{\varphi'\}$ (indiferent de φ'). Cu alte cuvinte, putem extinde oricând mulțimea de antecedenți ai unei secvențe valide și obținem o nouă secvență validă.

Exemplul 109. *Să arătăm că secvența* $\{p, \neg q, r, (q_1 \wedge q_2)\} \vdash \neg\neg p$ *este validă:*

1. $\{p, \neg p\} \vdash p$; *(Ipoteză)*
2. $\{p, \neg p\} \vdash \neg p$; *(Ipoteză)*
3. $\{p, \neg p\} \vdash \bot$; *($\neg e$, 1, 2)*
4. $\{p\} \vdash \neg\neg p$; *($\neg i$, 3)*
5. $\{p, \neg q\} \vdash \neg\neg p$; *(Extindere, 4)*
6. $\{p, \neg q, r\} \vdash \neg\neg p$; *(Extindere, 5)*
7. $\{p, \neg q, r, (q_1 \wedge q_2)\} \vdash \neg\neg p$. *(Extindere, 6)*

Toate regulile de mai sus reprezintă deducția naturală pentru o logică care se numește *logica propozițională intuiționistă*. În acest curs, noi studiem *logica propozițională clasică*. Cele două logici au aceeași sintaxă, dar semantica este diferită. Logica intuiționistă este foarte importantă în informatică, deoarece demonstrațiile formale din această logică sunt într-o corespondență 1-la-1 cu programele de calculator. Adică oricărei demonstrații intuiționiste îi corespunde un program, și oricărui program îi corespunde o demonstrație intuiționistă. Adică, de fapt, orice demonstrație formală *este* un program. Logica intuiționistă este un caz de *logică constructivă*, adică o logică în care orice demonstrație reprezintă un algoritm de calcul.

Pentru a obține un sistem deductiv care să corespundă logicii propoziționale clasice, cea pe care o studiem, mai avem nevoie de o singură regulă:

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

Exemplul 110. *Să arătăm că secvența* $\{(\neg p \to q), \neg q\} \vdash p$ *este validă:*

1. $\{(\neg p \to q), \neg q, \neg p\} \vdash \neg p$; *(Ipoteză)*
2. $\{(\neg p \to q), \neg q, \neg p\} \vdash (\neg p \to q)$; *(Ipoteză)*
3. $\{(\neg p \to q), \neg q, \neg p\} \vdash q$; *($\to e$, 2, 1)*
4. $\{(\neg p \to q), \neg q, \neg p\} \vdash \neg q$; *(Ipoteză)*
5. $\{(\neg p \to q), \neg q, \neg p\} \vdash \bot$; *($\neg i$, 4, 3)*
6. $\{(\neg p \to q), \neg q\} \vdash \neg\neg p$; *($\neg i$, 5)*
7. $\{(\neg p \to q), \neg q\} \vdash p$. *($\neg\neg e$, 6)*

Exemplul 111. *Să arătăm că secvența* $\{\} \vdash (p \vee \neg p)$ *este validă:*

1. $\{\neg(p \vee \neg p), p\} \vdash \neg(p \vee \neg p)$; *(Ipoteză)*
2. $\{\neg(p \vee \neg p), p\} \vdash p$; *(Ipoteză)*
3. $\{\neg(p \vee \neg p), p\} \vdash (p \vee \neg p)$; *($\vee i_1$, 2)*
4. $\{\neg(p \vee \neg p), p\} \vdash \bot$; *($\neg e$, 1, 3)*
5. $\{\neg(p \vee \neg p)\} \vdash \neg p$; *($\neg i$, 4)*
6. $\{\neg(p \vee \neg p)\} \vdash (p \vee \neg p)$; *($\vee i_2$, 5)*
7. $\{\neg(p \vee \neg p)\} \vdash \neg(p \vee \neg p)$; *(Ipoteză)*

8. $\{\neg(p \vee \neg p)\} \vdash \bot$; *($\neg e$, 7, 6)*
9. $\{\} \vdash \neg\neg(p \vee \neg p)$; *($\neg i$, 8)*
10. $\{\} \vdash (p \vee \neg p)$. *($\neg\neg e$, 9)*

Exercițiul 112. *Arătați că următoarele secvențe sunt valide:*

1. $\{\neg(p \wedge q)\} \vdash (\neg p \vee \neg q)$;
2. $\{\neg(\neg p \vee \neg q)\} \vdash (p \wedge q)$;
3. $\{\neg(\neg p \wedge \neg q)\} \vdash (p \vee q)$.

7.6 Deducția naturală

Deducția naturală este sistemul deductiv alcătuit din toate regulile din secțiunile precedente. Iată aici toate regulile:

$$\wedge i\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \varphi'}{\Gamma \vdash (\varphi \wedge \varphi'),} \qquad \wedge e_1\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi,}$$

$$\wedge e_2\ \frac{\Gamma \vdash (\varphi \wedge \varphi')}{\Gamma \vdash \varphi',} \qquad \to e\ \frac{\Gamma \vdash (\varphi \to \varphi') \qquad \Gamma \vdash \varphi}{\Gamma \vdash \varphi',} \qquad \to i\ \frac{\Gamma, \varphi \vdash \varphi'}{\Gamma \vdash (\varphi \to \varphi'),}$$

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

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

$$\neg e\ \frac{\Gamma \vdash \varphi \qquad \Gamma \vdash \neg\varphi}{\Gamma \vdash \bot,} \qquad \neg i\ \frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi,} \qquad \bot e\ \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi,}$$

$$\textsc{Ipoteză}\ \frac{}{\Gamma \vdash \varphi}\ \varphi \in \Gamma, \qquad \textsc{Extindere}\ \frac{\Gamma \vdash \varphi}{\Gamma, \varphi' \vdash \varphi,} \qquad \neg\neg e\ \frac{\Gamma \vdash \neg\neg\varphi}{\Gamma \vdash \varphi.}$$

7.7 Reguli derivate

O *regulă derivată* este o regulă de inferență care se poate demonstra cu ajutorul celorlalte reguli din sistem printr-o *schemă de demonstrație formală*. O

astfel de regulă este regula de introducere a dublei negații:

$$\neg\neg i\ \frac{\Gamma \vdash \varphi}{\Gamma \vdash \neg\neg\varphi.}$$

Iată o schemă de demonstrație formală pentru regula de introducere a dublei negații. Pornim cu ipotezele regulii, iar ultima secvență din demonstrație trebuie să fie concluzia regulii.

1. $\Gamma \vdash \varphi$; (ipoteza regulii derivate)
2. $\Gamma, \neg\varphi \vdash \varphi$; (Extindere, 1)
3. $\Gamma, \neg\varphi \vdash \neg\varphi$; (Ipoteza)
4. $\Gamma, \neg\varphi \vdash \bot$; ($\neg e$, 2, 3)
5. $\Gamma \vdash \neg\neg\varphi$. ($\neg i$, 4)

După ce a fost demonstrată (ca mai sus), o regulă derivată poate fi folosită pentru a scurta alte demonstrații, similar cu modul în care scurtăm codul prin scrierea de subprograme (funcții) în limbajele de programare. Iată un exemplu de demonstrație formală care folosește regula de introducere a dublei negații:

1. $\{(\neg\neg p \to q), p\} \vdash p$; (Ipoteză)
2. $\{(\neg\neg p \to q), p\} \vdash \neg\neg p$; ($\neg\neg i$, 1)
3. $\{(\neg\neg p \to q), p\} \vdash (\neg\neg p \to q)$; (Ipoteză)
4. $\{(\neg\neg p \to q), p\} \vdash q$. ($\to e$, 3, 2)

În absența folosirii regulii derivate, ar trebui să construim o demonstrație mai lungă pentru secvența de mai sus:

1. $\{(\neg\neg p \to q), p\} \vdash p$; (Ipoteză)
2. $\{(\neg\neg p \to q), p, \neg p\} \vdash p$; (Extindere, 1)
3. $\{(\neg\neg p \to q), p, \neg p\} \vdash \neg p$; (Ipoteza)
4. $\{(\neg\neg p \to q), p, \neg p\} \vdash \bot$; ($\neg e$, 2, 3)
5. $\{(\neg\neg p \to q), p\} \vdash \neg\neg p$. ($\neg i$, 4)
6. $\{(\neg\neg p \to q), p\} \vdash (\neg\neg p \to q)$; (Ipoteză)
7. $\{(\neg\neg p \to q), p\} \vdash q$. ($\to e$, 6, 5)

Observați că liniile 1–5 din demontrația mai lungă sunt o instanță a schemei de demonstrație formală a regulii derivate.

7.8 Corectitudinea și completitudinea deducției naturale

Teorema 113 (Corectitudinea deducției naturale)**.** *Pentru orice mulțime de formule* Γ *și orice formulă* φ*, dacă secvența* $\Gamma \vdash \varphi$ *este validă, atunci* $\Gamma \models \varphi$*.*

Exercițiu: de demonstrat la seminar.

Teorema 114 (Completitudinea deducției naturale)**.** *Pentru orice mulțime de formule* Γ *și orice formulă* φ*, 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.*

Exercițiul 115. *Arătați, folosind cele două teoreme de mai sus, că o mulțime* $\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ *de formule este inconsistentă ddacă secvența* $\{\varphi_1, \varphi_2, \ldots, \varphi_n\} \vdash \bot$ *este validă.*

7.9 Fișă de exerciții

Exercițiul 116. *Demonstrați că* $(\mathtt{p} \wedge \mathtt{r})$ *este consecință sintactică din mulțimea* $\{((\mathtt{q} \wedge \mathtt{r}) \wedge \mathtt{q}), (\mathtt{p} \wedge \mathtt{p})\}$*.*

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

1. $(\mathtt{p} \wedge \mathtt{q}), \mathtt{r} \vdash (\mathtt{p} \wedge (\mathtt{r} \vee \mathtt{r}'))$*;*
2. $(\mathtt{p} \to (\mathtt{q} \to \mathtt{r})) \vdash ((\mathtt{p} \wedge \mathtt{q}) \to \mathtt{r})$*;*
3. $((\mathtt{p} \wedge \neg\mathtt{r}) \to \mathtt{q}), \neg\mathtt{q}, \mathtt{p} \vdash \mathtt{r}$*;*

Exercițiul 118. *Terminați jocul de la adresa* `https://profs.info.uaic.ro/~stefan.ciobaca/lnd.html`*. Nu trișați. Se consideră trișat: schimbarea codului JavaScript, dacă altcineva rezolvă un nivel în locul dumneavoastră, sau dacă demonstrați regulile derivate folosind chiar regulile derivate (într-un singur pas).*

Exercițiul 119. *Demonstrați că următoarele reguli sunt derivate:*

1. $\neg\neg i$*;*

2. *LEM (law of excluded middle):* $\text{LEM}\ \dfrac{}{\Gamma \vdash (\varphi \vee \neg\varphi)}$;

3. *PBC (proof by contradiction):* $\text{PBC}\ \dfrac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \varphi}$;

4. *MT (modus tollens):* $\text{MT}\ \dfrac{\Gamma \vdash (\varphi \to \varphi') \qquad \Gamma \vdash \neg\varphi'}{\Gamma \vdash \neg\varphi}$.

Exercițiul 120. *Demonstrați teorema de corectitudine (prin inducție după numărul de pași din demonstrația formală).*

Exercițiul 121. *Arătați că regula $\neg\neg e$ poate fi derivată folosind LEM (i.e., puteți folosi LEM în demonstrația formală, dar nu $\neg\neg e$).*

Notație. *Putem scrie și $\varphi \dashv \varphi'$ în loc de $\varphi' \vdash \varphi$. Scriem $\varphi \dashv\vdash \varphi'$ ca prescurtare pentru faptul că secvențele $\varphi \vdash \varphi'$ și $\varphi \dashv \varphi'$ sunt valide.*

Exercițiul 122. *Demonstrați, apelând la teoremele de corectitudine și completitudine, că $\varphi_1 \dashv\vdash \varphi_2$ dacă și numai dacă $\varphi_1 \equiv \varphi_2$.*

Capitolul 8

Forme normale

Până acum am discutat despre sintaxa și semantica logicii propoziționale, iar în ultimul capitol am abordat o primă metodă de *mecanizare* a demonstrațiilor formale, și anume *deducția naturală*.

Una dintre preocupările importante în Logică este *demonstrarea automată*, adică *căutarea automată* de demonstrații formale de către un calculator pentru o anumită conjectură/teoremă.

Deducția naturală este un sistem deductiv care are anumite calități (de exemplu, demonstrațiile formale pot fi verificate și înțelese ușor în principiu), dar performanța demonstratoarelor automate care folosesc deducția naturală lasă de dorit în practică.

Din acest motiv, vom studia un alt sistem deductiv, numit *rezoluție*, și care permite implementări eficiente în practică, deși demonstrațiile prin rezoluție nu sunt atât de elegante ca demonstrațiile prin deducție naturală.

Spre deosebire de deducția naturală, care lucrează cu formule oarecare, rezoluția este limitată la formule care au o anumită *formă*, numită *formă normală clauzală*. Scopul acestui capitol este studiul acestei forme normale.

8.1 Notații

În practică, când scriem formule din $\mathbb{LP}$, nu scriem parantezele decât dacă este strict necesar.

La fel cum dacă scriem $5 \times -3 + 2$ înțelegem $(5 \times (-3)) + 2$, la fel în loc de $\mathtt{p} \wedge \neg \mathtt{q} \vee \mathtt{r}$ vom înțelege $((\mathtt{p} \wedge \neg \mathtt{q}) \vee \mathtt{r})$. În acest sens, negația $\neg$ este operatorul cu prioritatea cea mai mare, urmat de $\wedge$ și apoi urmat de $\vee$. Putem ține minte această convenție asociind:

1. $\neg$ cu minusul unar;

2. $\wedge$ cu înmulțirea și

3. $\vee$ cu adunarea.

Mai mult, operatorii $\wedge$ și $\vee$ sunt asociativi și comutativi. Din acest motiv, permitem scrierea $\mathtt{p} \wedge \mathtt{q} \wedge \mathtt{r}$ în loc de $(\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r})$ sau $(\mathtt{p} \wedge (\mathtt{q} \wedge \mathtt{r}))$.

În continuare, vom permite scrierea de formule fără paranteze, respectând următoarea ordine de prioritate a conectorilor logici:

$$\bot, \neg, \wedge, \vee, \rightarrow, \leftrightarrow.$$

De exemplu, prin șirul de simboluri

$$\neg\neg \mathtt{p} \vee \bot \wedge \mathtt{p} \rightarrow \neg \mathtt{p} \wedge \mathtt{q} \leftrightarrow \mathtt{q}$$

înțelegem formula

$$(((\neg\neg \mathtt{p} \vee (\bot \wedge \mathtt{p})) \rightarrow (\neg \mathtt{p} \wedge \mathtt{q})) \leftrightarrow \mathtt{q}).$$

De asemenea, vom nota $((\varphi_1 \vee \varphi_2) \vee \varphi_3)$ cu $\varphi_1 \vee \varphi_2 \vee \varphi_3$ (adică într-o secvență de $\vee$-uri se face asocierea la stânga).

De asemenea, vom nota $((\varphi_1 \wedge \varphi_2) \wedge \varphi_3)$ cu $\varphi_1 \wedge \varphi_2 \wedge \varphi_3$ (adică într-o secvență de $\wedge$-uri se face asocierea la stânga).

Exemplul 123. *Scrierea*

$$\mathtt{p} \wedge \mathtt{q} \wedge \mathtt{r} \vee \neg \mathtt{p} \wedge \neg \mathtt{q} \wedge \neg \mathtt{r}$$

reprezintă formula

$$(((\mathtt{p} \wedge \mathtt{q}) \wedge \mathtt{r}) \vee ((\neg \mathtt{p} \wedge \neg \mathtt{q}) \wedge \neg \mathtt{r})).$$

Exemplul 124. *Scrierea*

$$\mathtt{p}_1 \wedge \mathtt{p}_2 \wedge \mathtt{p}_3 \wedge \mathtt{p}_4$$

reprezintă formula

$$(((\mathtt{p}_1 \wedge \mathtt{p}_2) \wedge \mathtt{p}_3) \wedge \mathtt{p}_4).$$

Exemplul 125. *Scrierea*

$$\mathtt{p}_1 \vee \mathtt{p}_2 \vee \mathtt{p}_3 \vee \mathtt{p}_4$$

reprezintă formula

$$(((\mathtt{p}_1 \vee \mathtt{p}_2) \vee \mathtt{p}_3) \vee \mathtt{p}_4).$$

8.2 Teorema de Înlocuire

Prezentăm o teoremă pe care am utilizat-o implicit până acum:

Teorema 126 (Teorema de Înlocuire)**.** *Fie φ, φ' două formule astfel încât $\varphi \equiv \varphi'$. Fie φ_1 o formulă care conține φ ca subformulă. Fie φ_2 formula obținută din φ_1 prin înlocuirea unei apariții a formulei φ cu φ'.*

Atunci $\varphi_1 \equiv \varphi_2$.

Cu alte cuvinte, relația $\equiv$ este o *congruență*.

Exemplul 127. *Fie $\varphi = \mathtt{p}$ și $\varphi' = \neg\neg\mathtt{p}$.*

Fie $\varphi_1 = (\mathtt{p} \vee \mathtt{q})$ și $\varphi_2 = (\neg\neg\mathtt{p} \vee \mathtt{q})$.

Prin teorema de înlocuire, obținem $\varphi_1 \equiv \varphi_2$. Cu alte cuvinte, prin înlocuirea unei subformule cu una echivalentă, noua formulă este echivalentă cu formula inițială.

8.3 Literal

Definiția 128 (Literal)**.** *O formulă φ se numește* literal *dacă există o variabilă propozițională $a \in A$ astfel încât*

$$\varphi = a \qquad \text{sau} \qquad \varphi = \neg a.$$

Exemplul 129. *De exemplu, formulele $\mathtt{p}, \mathtt{q}, \neg\mathtt{p}, \neg\mathtt{p}', \mathtt{q}_1$ sunt literali, dar formulele $(\mathtt{p} \vee \mathtt{q}), \neg\neg\mathtt{p}, \neg\neg\neg\mathtt{q}_1, (\neg\mathtt{p} \wedge \mathtt{q})$ nu sunt literali.*

8.4 Clauză

Definiția 130. *O formulă φ se numește* clauză *dacă există $n \geq 1$ literali $\varphi_1, \ldots, \varphi_n$ astfel încât*

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

Cu alte cuvinte, o clauză este o disjuncție de literali.

Exemplul 131. *Următoarele formule sunt clauze:*

1. $\mathtt{p} \vee \mathtt{q} \vee \mathtt{r}$;
2. $\mathtt{p} \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;
3. $\neg\mathtt{p} \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;
4. $\neg\mathtt{p}_1 \vee \mathtt{p}_1 \vee \mathtt{p}_2 \vee \neg\mathtt{q} \vee \neg\mathtt{r}$;

5. $(\neg p_1 \vee p_1)$;
6. $\neg p_1$ $(n = 1)$.
7. p $(n = 1)$.

Următoarele formule nu sunt clauze:

1. $p \wedge r$; *(conjuncție în loc de disjuncție)*
2. $p \vee \neg\neg q \vee \neg r$; *(nu e disjuncție de* literali*)*
3. $\neg\neg p \vee p \wedge \neg p_1$. *(apare $\neg\neg$, deci nu avem literali; apare $\wedge$)*

Observație. *Definiția 130 menționează explicit că numărul de literali n este ≥ 1. Totuși, pentru cazul particular $n = 0$ literali, obținem o clauză specială numită* clauza vidă*, care are o notație dedicată: $\square$. Considerăm că $\square$ reprezintă o formulă nesatisfiabilă (din punct de vedere semantic, clauza vidă este echivalentă cu $\bot$). Această clauză vidă va fi utilizată mai târziu într-un capitol în care discutăm despre rezoluție în logica propozițională.*

8.5 Forma Normală Conjunctivă

Definiția 132 (FNC)**.** *O formulă φ* este în FNC *dacă există $n \geq 1$ clauze $\varphi_1, \ldots, \varphi_n$ astfel încât*

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

Cu alte cuvinte, o formulă în FNC este o conjuncție de disjuncții de literali. Sau: o formulă în FNC este o conjuncție de clauze. FNC se mai numește și *formă normală clauzală.*

Exemplul 133. *Următoarele formule sunt în FNC:*

1. $(\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r)$;
2. $\neg p \wedge (r \vee \neg p \vee r') \wedge \neg r$;
3. $\neg p \wedge r \wedge \neg r$;
4. $\neg p$;
5. p.

Următoarele formule nu sunt în FNC:

1. $\neg(\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r)$; *(prima clauză are conectorul $\neg$ în față)*
2. $\neg p \wedge \neg(r \vee \neg p \vee r')$; *(a doua clauză are $\neg$ în față)*
3. $\neg p \vee (r \wedge \neg p \wedge r')$; *(conectorul principal e disjuncția, în loc de conjuncție)*
4. $\neg\neg p$; *(apare $\neg\neg$, deci nu avem literali)*
5. $(p \vee (q \wedge r))$. *(disjuncție de conjuncții, nu conjuncție de disjuncții de literali)*

8.6 Aducerea unei formule în FNC

Teorema 134 (Teorema de aducere a unei formule în FNC)**.** *Pentru orice formulă φ, există o formulă φ', aflată în FNC, astfel încât $\varphi \equiv \varphi'$.*

Demonstrație: [Schiță de demonstrație]
Prin aplicarea repetată a teoremei de înlocuire, folosind 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. $(\varphi_1 \vee (\varphi_2 \wedge \varphi_3)) \equiv ((\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3))$;
4. $((\varphi_2 \wedge \varphi_3) \vee \varphi_1) \equiv ((\varphi_2 \vee \varphi_1) \wedge (\varphi_3 \vee \varphi_1))$;
5. $(\varphi_1 \vee (\varphi_2 \vee \varphi_3)) \equiv ((\varphi_1 \vee \varphi_2) \vee \varphi_3)$;
6. $(\varphi_1 \wedge (\varphi_2 \wedge \varphi_3)) \equiv ((\varphi_1 \wedge \varphi_2) \wedge \varphi_3)$;
7. $\neg(\varphi_1 \vee \varphi_2) \equiv (\neg\varphi_1 \wedge \neg\varphi_2)$;
8. $\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1 \vee \neg\varphi_2)$;
9. $\neg\neg\varphi \equiv \varphi$.

Primele două echivalențe asigură faptul că din formulă dispar toate implicațiile și dublele implicații.

Echivalențele 3 și 4 asigură că în arborele formulei toate disjuncțiile *coboară* sub conjuncții.

Echivalențele 5 și 6 asigură asociativitatea lanțurilor de disjuncții și respectiv de conjuncții (astfel încât să punem scrie $\varphi_1 \vee \varphi_2 \vee \varphi_3 \vee \ldots \vee \varphi_n$ în loc de $((((\varphi_1 \vee \varphi_2) \vee \varphi_3) \vee \ldots) \vee \varphi_n)$.

Echivalențele 7 și 8 asigură faptul că negațiile ajung *sub* conjuncții și disjuncții în arborele formulei.

Ultima echivalență asigură că nu există două negații *una sub alta* în arborele de sintaxă abstractă.

Aplicarea echivalențelor de mai sus se oprește (nu se pot aplica la infinit – de ce?).

Rezultatul va fi o formulă în care conjuncțiile sunt *deasupra* disjuncțiilor, care la rândul lor sunt *deasupra* eventualelor negații în arborele abstract de sintaxă, adică o formulă în FNC. q.e.d.

Exemplul 135. *Să aducem formula* $((\neg p \to \neg q) \leftrightarrow (q \to p))$ *în FNC:*

$$
\begin{aligned}
& ((\neg p \to \neg q) \leftrightarrow (q \to p)) \\
\overset{1}{\equiv}\ & ((\neg p \to \neg q) \to (q \to p)) \wedge ((q \to p) \to (\neg p \to \neg q)) \\
\overset{2}{\equiv}\ & ((\neg\neg p \vee \neg q) \to (\neg q \vee p)) \wedge ((\neg q \vee p) \to (\neg\neg p \vee \neg q)) \\
\overset{2}{\equiv}\ & (\neg(\neg\neg p \vee \neg q) \vee (\neg q \vee p)) \wedge (\neg(\neg q \vee p) \vee (\neg\neg p \vee \neg q)) \\
\overset{7}{\equiv}\ & ((\neg\neg\neg p \wedge \neg\neg q) \vee (\neg q \vee p)) \wedge ((\neg\neg q \wedge \neg p) \vee (\neg\neg p \vee \neg q)) \\
\overset{9}{\equiv}\ & ((\neg p \wedge q) \vee (\neg q \vee p)) \wedge ((q \wedge \neg p) \vee (p \vee \neg q)) \\
\overset{4}{\equiv}\ & ((\neg p \vee (\neg q \vee p)) \wedge (q \vee (\neg q \vee p))) \wedge ((q \vee (p \vee \neg q)) \wedge (\neg p \vee (p \vee \neg q))) \\
\overset{5}{\equiv}\ & (((\neg p \vee \neg q) \vee p) \wedge ((q \vee \neg q) \vee p)) \wedge (((q \vee p) \vee \neg q) \wedge ((\neg p \vee p) \vee \neg q)) \\
=\ & ((\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p)) \wedge ((q \vee p \vee \neg q) \wedge (\neg p \vee p \vee \neg q)) \\
\overset{6}{\equiv}\ & ((((\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p)) \wedge (q \vee p \vee \neg q)) \wedge (\neg p \vee p \vee \neg q)) \\
=\ & (\neg p \vee \neg q \vee p) \wedge (q \vee \neg q \vee p) \wedge (q \vee p \vee \neg q) \wedge (\neg p \vee p \vee \neg q).
\end{aligned}
$$

8.7 Forma Normală Disjunctivă

Definiția 136 (FND)**.** *O formulă este în FND dacă este o disjuncție de conjuncții de literali.*

Exemplul 137. *Următoarele formule sunt în FND:*

1. $(\neg p \wedge q) \vee (r \wedge \neg p \wedge r') \vee (\neg p \wedge \neg r)$;
2. $\neg p \vee (r \wedge \neg p \wedge r') \vee \neg r$;
3. $\neg p \vee r \vee \neg r$;
4. $\neg p$;
5. p.

Următoarele formule nu sunt în FND:

1. $\neg(\neg p \wedge q) \vee (r \wedge \neg p \wedge r') \vee (\neg p \wedge \neg r)$;
2. $(\neg p \vee \neg(r \wedge \neg p \wedge r'))$;
3. $\neg\neg p \wedge (r \vee \neg p \vee r')$;
4. $\neg\neg p$;
5. $(p \wedge (q \vee r))$.

Exercițiul 138. *Enunțați și schițați demonstrația teoremei de aducere în FND.*

8.8 Legătura dintre FNC și FND

Definiția 139. *Complementul unei formule $\varphi \in \mathbb{LP}_{\neg,\wedge,\vee}$ se notează φ^c și este definit astfel:*

1. $a^c = \neg a$, *pentru orice* $a \in A$;
2. $(\neg\varphi)^c = \varphi$, *pentru orice* $\varphi \in \mathbb{LP}$;
3. $(\varphi_1 \wedge \varphi_2)^c = ({\varphi_1}^c \vee {\varphi_2}^c)$, *pentru orice* $\varphi_1, \varphi_2 \in \mathbb{LP}$;
4. $(\varphi_1 \vee \varphi_2)^c = ({\varphi_1}^c \wedge {\varphi_2}^c)$, *pentru orice* $\varphi_1, \varphi_2 \in \mathbb{LP}$.

Exemplul 140. *Iată câteva exemple de calcul pentru funcția* complement*:*

1. $(\neg p)^c = p$ *(atenție, complementul lui* $\neg p$ *nu este* $\neg\neg p$*);*
2. $(\neg(\neg p \wedge q)\vee(r \wedge \neg p \wedge r')\vee(\neg p \wedge \neg r))^c =$
   $(\neg p \wedge q)\wedge(\neg r \vee p \vee \neg r')\wedge(p \vee r)$;
3. $((\neg p \vee q) \wedge (r \vee \neg p \vee r') \wedge (\neg p \vee \neg r))^c =$
   $(p \wedge \neg q) \vee (\neg r \wedge p \wedge \neg r') \vee (p \wedge r)$.

Teorema 141 (Complementul este echivalent cu negația)**.** *Pentru orice formulă $\varphi \in \mathbb{LP}_{\neg,\wedge,\vee}$, avem că*

$$\varphi^c \equiv \neg\varphi.$$

Exercițiul 142. *Demonstrați teorema de mai sus prin inducție structurală.*

Exemplul 143. $(p \wedge (q \vee \neg r))^c = (\neg p \vee (\neg q \wedge r)) \equiv \neg(p \wedge (q \vee \neg r))$.

Teorema 144 (Legătura dintre FNC și FND)**.** *Dacă φ_1 este o formulă în FNC și φ_2 o formulă în FND, atunci*

$${\varphi_1}^c \text{ este în FND}$$

și

$${\varphi_2}^c \text{ este în FNC.}$$

8.9 Fișă de exerciții

Exercițiul 145. *Exprimați cu cât mai puține paranteze următoarele formule:*
1. $((p \wedge \neg q) \wedge r)$*;* *2.* $((p \vee \neg q) \wedge r)$*;* *3.* $\neg((p \vee \neg q) \wedge r)$*.*

Exercițiul 146. *Scrieți arborele abstract pentru următoarele formule, având grijă să le parantezați conform ordinii de prioritate a operatorilor:*
1. $p \wedge q \vee r$*;* *2.* $\neg p \wedge \neg q \vee \neg r$*;* *3.* $p \wedge \neg q \vee r$*;* *4.* $\neg p \wedge \neg(q \vee r)$*;* *5.* $p \vee \neg p \wedge \neg(q \vee r)$*.*

Exercițiul 147. *Aduceți următoarele formule în FNC:*

1. $((p \wedge q) \vee r)$*;*
2. $((p \vee q) \wedge r)$*;*
3. $\neg((p \vee q) \wedge r)$*;*
4. $\neg((p \wedge q) \vee r)$*;*
5. $((p \wedge q) \vee (\neg p \wedge \neg q))$*;*
6. $((p \wedge (q \wedge r)) \vee \neg p)$*;*
7. $\neg(\neg(p \wedge q) \vee (p \vee q))$*;*
8. $(\neg(p \wedge q) \rightarrow (\neg p \wedge \neg q))$*;*
9. $(p \leftrightarrow (q \rightarrow (\neg p \wedge \neg q)))$*;*
10. $((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p))$*;*
11. $(p_1 \wedge q_1) \vee (p_2 \wedge q_2) \vee \ldots \vee (p_n \wedge q_n)$ *(rezolvați întâi pentru* $n = 2$ *și* $n = 3$*, apoi pentru un* n *oarecare);*

Exercițiul 148. *Calculați o FNC pentru formula* $p \vee q \rightarrow p \wedge \neg r$ *(atenție la ordinea operatorilor).*

Exercițiul 149. *Calculați complementul formulelor aflate în FNC calculate mai sus.*