

### 3 skyrius Algoritmų teorija

Šiame skyriuje nagrinėsime vieną *algoritmiškai apskaičiuojamųjų funkcijų* formalizmą – rekursyvias funkcijas. Visų šiame skyriuje nagrinėjamųjų funkcijų apibrėžimų bei reikšmių aibė yra viena ir ta pati *natūraliųjų skaičių aibė*  $N = \{0, 1, 2, \dots\}$ . Rekursyviųjų funkcijų aibė sutampa su Turing mašinomis apskaičiuojamųjų funkcijų aibe. D.Hilbert suformulavo reikalavimus, kuriuos turi tenkinti algoritmiškai apskaičiuojamos funkcijos. Remdamasis jo darbais 1931 m. K.Gödel pirmasis aprašė rekursyviųjų funkcijų klasę. Vėliau, 1936 m. A.Church, pritaikęs kitas idėjas, aprašė tą pačią rekursyviųjų funkcijų klasę.

#### 3.1 Intuityvioji algoritmo samprata

XX amžiaus pradžioje atsirado poreikis tiksliai apibrėžti sąvoką *efektyvi procedūra (algoritmas)*. Pradėta manyti, kad kai kurios problemos nėra išsprendžiamos. Bet kaip tai įrodyti? Kaip išsiaiškinti, kad tam tikrai problemai spręsti nėra algoritmo? Tam nepakanka plačiai vartojamos intuityvios algoritmo sampratos:

*Seka griežtų komandų (instrukcijų), pagal kurias atliekamos operacijos, leidžiančios spręsti matematikos ar logikos uždavinius.*

Reikia turėti matematiškai tikslią algoritmo sąvoką. Ji turėtų apibendrinti intuityviai suprantamų algoritmų savybes.

Pateiksime plačiai žinomą Euklido algoritmo pavyzdį. Tarkime, yra du natūralieji skaičiai  $a_1 \geq a_2 > 0$ . Raskime didžiausiąjį bendrąjį jų daliklį. Algoritmas toks:

*Dalijame  $a_1$  iš  $a_2$ . Jei liekana  $a_3 = 0$ , tai  $a_2$  yra didžiausias bendrasis daliklis. Jei  $a_3 \neq 0$ , tai atliekame kitą veiksmą.*

*Dalijame  $a_2$  iš  $a_3$ . Jei liekana  $a_4 = 0$ , tai  $a_3$  ir yra didžiausias bendrasis daliklis. Jei  $a_4 \neq 0$ , tai atliekame kitą veiksmą.*

*Dalijame  $a_3$  iš  $a_4$  ir t.t*

Kadangi  $a_1 \geq a_2 > a_3 > \dots$ , tai po baigtinio skaičiaus žingsnių rasime didžiausią bendrąjį  $a_1$  ir  $a_2$  daliklį.

Pagrindinės savybės, kurias tenkina žinomų algoritmų pavyzdžiai yra tokios:

1. *Diskretumas*. Veiksmai išdėstyti tam tikra seka. Viena jų atlikę, pereiname prie kito. Veiksmai dar vadinami algoritmo žingsniais.

2. *Determinuotumas*. Atlikę veiksmą žinome (nurodyta) ką toliau daryti.

3. *Žingsnių elementarumas*. Algoritmo veiksmų seką galima suskaidyti į labai paprastus, elementarius, paprastai aprašomus ir lengvai įvykdomus žingsnius.

4. *Masiškumas*. Algoritmai taikomi tam tikrai aibei. Pavyzdžiui, aprašytasis Euklido algoritmas taikomas *bet kuriems* natūraliesiems skaičiams  $a_1 \geq a_2 > 0$ .

Iš kur gi kilo žodis algoritmas? Tai sulotynintas arabų (kai kurie šaltiniai nurodo, persų) matematiko Al Chorezmi (ca 783-850) vardas. Jis išgarsėjo savo knyga, kurioje aprašė veiksmus su skaičiais, perimtais iš Indijos. Naujoji pozicinė skaičiavimo sistema greitai paplito pasaulyje, o jo knyga tapo daugelio žmonių parankine knyga. Knygoje buvo daug taisyklių rinkinių, kuriuos taikant po baigtinio žingsnių skaičiaus gaunamas rezultatas.

Algoritmo sąvoka buvo tikslinama dviem būdais:

- 1) kuriama idealizuota (matematinė) skaičiavimo mašina,
- 2) aprašoma algoritmiškai apskaičiuojamų funkcijų aibė.

Po daugelio metų – 1934-1936 m. ir viena, ir kita kryptimi dirbančių mokslininkų gauta daug skirtingų algoritmo sąvokos patikslinimų. Pirmo-

sios krypties žinomiausiais darbais tapo A.Turingo ir E.Posto aprašytosios mašinos. A.Turing laikomas *informatikos mokslo tėvu*. Sukurtąją mašiną pasiūlė vadinti *elektroniniu kompiuteriu*. Antrojo pasaulinio karo metu tokia mašina jis pasinaudojo iššifruodamas vokiečių naudotą povandeniniuose laivuose kodą *Enigma*. Savo gyvenimą 1954 m. A.Turing baigė nusinuododamas kalio cianidu. Paskutiniais gyvenimo metais jis dirbo Mančesterio universitete. Intuityviai *algoritmiškai apskaičiuojama funkcija* suprantama taip: žinodamas funkcijos  $y = f(x)$  argumento reikšmę *moku apskaičiuoti* funkcijos reikšmę. Pavyzdžiui, akivaizdu, kad funkcija  $y = n^2$  algoritmiškai apskaičiuojama. O ar algoritmiškai apskaičiuojama ši funkcija, neaišku:

$$f(n) = \begin{cases} 1, & \text{jei sekoje } \pi = 3, 14... \text{ sutinkama greta stovintys lygiai} \\ & n \text{ septynetai (77...7)} \\ 0, & \text{kitu atveju} \end{cases}$$

Buvo sukurta daug metodų *algoritmiškai apskaičiuojamų funkcijų* klasei nusakyti. Žinomiausios yra K.Gödelio, A.Churcho bei S.Kleene aprašytosios funkcijų klasės. A.Church jas pavadino rekursyviosiomis funkcijomis.

**A.Churcho tezė.** *Algoritmiškai apskaičiuojamų funkcijų aibė sutampa su rekursyviųjų funkcijų aibe.*

Ši tezė buvo paskelbta 1936 metais. Teze vadinama todėl, kad tai tvirtinimas, kuriuo, A.Churcho nuomone, reikėtų tikėti, bet įrodyti negalima. Negalima įrodyti dviejų aibių lygybės, nes, viena vertu, tai matematiškai tiksli rekursyviųjų funkcijų klasė, kita vertus – intuityvi, netiksli, skirtingų žmonių skirtingai suprantama algoritmiškai apskaičiuojamų funkcijų klasė.

Kodėl tikima A.Churcho teze? Pagrindiniai argumentai yra du:

1. Visų pasiūlytų, skirtingomis idėjomis aprašytų algoritmiškai apskaičiuojamų funkcijų klasės sutampa ne tik tarpusavyje, bet ir su idealizuotų skaičiavimo mašinų apskaičiuojamomis funkcijų klasėmis,
2. Nėra žinomas joks intuityviai apskaičiuojamos funkcijos pavyzdys, kuris nebūtų rekursyvioji funkcija.

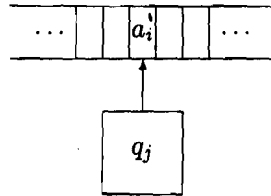
## 3.2 Turingo mašinos

**3.1 apibrėžimas.** *Turingo mašina vadiname ketvertą  $\langle Q, \Sigma, \delta, F \rangle$ , kuriame:*

- $Q = \{q_0, q_1, \dots, q_k\}$  ( $k \geq 0$ ) yra baigtinė mašinos būsenų aibė;  $q_0$  – pradinė būsena;  $F \subset Q$  – galutinių būsenų aibė,

- $\Sigma$  – baigtinė aibė, vadinama Turingo mašinos abėcėle. Tarp kiekvienos mašinos abėcėlės simbolių yra ir tuščios ląstelės simbolis  $b$ ,
- $\delta$  – perėjimų funkcija, kurios apibrėžimo aibė yra  $Q \times \Sigma$ , o reikšmės priklauso aibei  $Q \times \Sigma \times \{K, N, D\}$ .

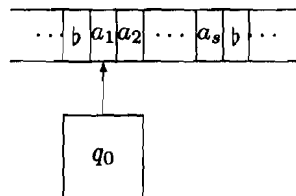
Turingo mašinos geometrinė interpretacija yra tokia. Mašiną sudaro begalinė į abi puses juosta, kuri suskirstyta į ląsteles. Mašina turi skaitymo galvutę, kuri kiekvienu laiko momentu yra ties viena kuria nors ląstele. Mašinos darbas diskretus. Pradedama kažkuriuo tai pradiniu momentu  $t_0$ . Įvykdo vieną komandą (tai vadiname mašinos žingsniu), kitu laiko momentu  $t_1$  įvykdo dar vieną komandą ir t.t. Kiekvienu laiko momentu kiekvienoje mašinos ląstelėje yra vienas kuris nors aibės  $\Sigma$  elementas. Ląstelė vadinama tuščia, jei joje įrašytas simbolis  $b$ . Geometrinė interpretacija:



Duotuoju momentu mašina yra būsenoje  $q_j$ , o skaitymo galvutė ties ląstele, kurioje įrašytas simbolis  $a_i$  (aibės  $\Sigma$  elementas).

Turingo mašinos komanda vadinsime reiškinį pavidalo  $\delta(q_i, a_j) = (q_k, a_l, X)$ , čia  $X \in \{K, D, N\}$ ,  $a_j, a_l \in \Sigma$ ,  $q_i, q_k \in Q$ . Komandos kairiaja puse vadinamas reiškinys  $\delta(q_i, a_j)$ . **Perėjimų funkcija** yra baigtinė komandų seka. Kai  $\delta$  yra vienareikšmė funkcija, t.y. kokia bebūtų pora  $(q_i, a_j)$  ( $q_i \in Q, a_j \in \Sigma$ ), tarp komandų atsirastų ne daugiau kaip viena, kurios kairiaja puse yra  $\delta(q_i, a_j)$ , **Turingo mašina vadinama determinuotąja**.

Pradiniu laiko momentu juostoje yra įrašytas kuris nors abėcėlės  $\Sigma' = \Sigma - \{b\}$  žodis. Mašina pradeda darbą būdama būsenoje  $q_0$ , skaitymo galvutė yra ties pirmąja iš kairės netuščia ląstele:



Tarkime laiko momentu  $t_i$  ( $i = 0, 1, 2, \dots$ ) skaitymo galvutė yra ties

ląstele, kurioje įrašytas aibės  $\Sigma$  elementas  $a$ , mašina yra būsenoje  $q_j$ . Tuomet ji per vieną laiko taktą (žingsnį) atlieka tokį darbą:

- ieško komandos prasidedančios  $\delta(q_j, a)$ . Jei tokios nėra, tai sakoma, kad mašina patenka į poziciją be išeities. Tuomet ji baigia darbą,
- tarkime komandų tarpe yra  $\delta(q_j, a) = (q_s, c, X)$ . Tuomet ji ją įvykdo, t.y. pereina į būseną  $q_s$ , elementą  $a$  nuvalo ir vietoje jo įrašo elementą  $c$ , bei pasislenka viena ląstele į dešinę (jei  $X = D$ ), kairę (jei  $X = K$ ) arba pasilieka ties ta pačia ląstele (jei  $X = N$ ). Jei  $q_s \in F$ , tai mašina baigia darbą ir sustoja. Juostoje esantis žodis vadinamas **mašinos darbo rezultatu**.

Tarkime pradiniais duomenimis yra žodis  $a_1 a_2 \dots a_s$ , galvutė ties pirmąja iš kairės netuščia ląstele bei mašina yra būsenoje  $q_0$ . Mašina dirba sutinkamai su perėjimų funkcija  $\delta$ . Galimi atvejai:

- mašina po baigtinio žingsnių skaičiaus patenka į poziciją be išeities arba dirba be galo ilgai. Abiem atvejais sakysime, kad mašinos darbo rezultatas neapibrėžtas (mašina neapibrėžta) su pradiniais duomenimis  $a_1 a_2 \dots a_s$ ,
- mašina po baigtinio žingsnių skaičiaus patenka į vieną iš galutinių būsenų ir juostoje yra aibės  $\Sigma'$  žodis  $b_1 b_2 \dots b_r$ . Sakysime, kad **mašina apibrėžta** ir jos darbo rezultatu su pradiniais duomenimis  $a_1 a_2 \dots a_s$  yra  $b_1 b_2 \dots b_r$ .

Tarkime duota Turingo mašina  $T = \langle Q, \Sigma, \delta, F \rangle$ . Sakysime, kad mašina  $T$  apskaičiuoja funkciją  $y = f(x)$ , jei galima rasti tokį funkcijos argumentų bei reikšmių kodavimą (pažymėkime jį *cod*) abėcėlės  $\Sigma'$  žodžiais, kad koks bebūtų  $x_0$ , jei  $f(x_0)$  apibrėžta ir  $y_0 = f(x_0)$ , tai mašina  $T$  su pradiniais duomenimis *cod*( $x_0$ ) (pradiniu laiko momentu mašina visada yra būsenoje  $q_0$ , o galvutė ties pirma netuščia ląstele) po baigtinio žingsnių skaičiaus pereina į kurią nors galutinę būseną ir mašinos darbo rezultatu yra *cod*( $y_0$ ). Atveju, kai  $f(x_0)$  neapibrėžta, mašina  $T$  taip pat neapibrėžta.

Rasti Turingo mašiną apskaičiuojančią funkciją  $f(x)$  reiškia rasti tokį ketvertą  $T = \langle Q, \Sigma, \delta, F \rangle$  ir funkcijos argumentų bei reikšmių kodavimą, kad  $T$  apskaičiuotų funkciją  $y = f(x)$ .

**Pavyzdys.** Rasti Turingo mašiną apskaičiuojančią funkciją  $y = n + 1$ , ( $n \in N$ ).

Abėcėlė  $\Sigma = \{0, 1, b\}$ . Natūraliuosius skaičius koduosime dvejetainiais skaičiais.

Perėjimų funkcija  $\delta$ :

$$\begin{aligned}\delta(q_0, 0) &= (q_0, 0, D) \\ \delta(q_0, 1) &= (q_0, 1, D) \\ \delta(q_0, b) &= (q_1, b, K)\end{aligned}$$

$$\begin{aligned}\delta(q_1, 0) &= (q_2, 1, N) \\ \delta(q_1, 1) &= (q_1, 0, K) \\ \delta(q_1, b) &= (q_2, 1, N)\end{aligned}$$

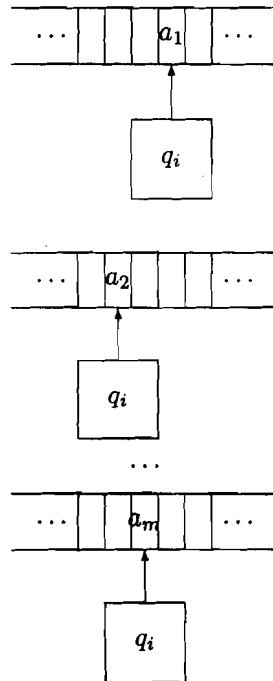
$$F = \{q_2\}, Q = \{q_0, q_1, q_2\}.$$

Panagrinėsime porą Turingo mašinų variantų.

### Daugiajuostės Turingo mašinos.

$m$ -juostė ( $m \geq 3$ ) Turingo mašina, tai ketvertas  $\langle Q, \Sigma, \delta, F \rangle$ .  $Q, \Sigma, F$  apibrėžiamos taip pat kaip ir vienajuosčių mašinų atveju.  $\delta$  – perėjimų funkcija, kurios apibrėžimo aibė  $Q \times \Sigma^m$ , o reikšmės priklauso aibei  $Q \times \Sigma^m \times \{K, D, N\}^m$ .

$m$ -juostę Turingo mašiną sudaro  $m$  begalinių į abi puses juostų, suskirstytų į ląsteles. Pirmoji juosta vadinama *įėjimo juosta*. Joje užrašomi pradiniai duomenys. Paskutinioji juosta vadinama *išėjimo juosta*. Joje užrašomas mašinos darbo rezultatas. Likusios  $m-2$  juostos vadinamos *darbinėmis*. Veiksmai su duomenimis atliekami darbinėse juostose. Mašinoje yra  $m$  skaitymo galvūčių, kiekvienai juostai po vieną. Kiekvienu laiko momentu visos galvutės būna vienoje ir toje pačioje būsenoje.



**Pavyzdys.** Duota abėcėlė  $\Sigma = \{0, 1, b\}$ . Rasime 3-juostę Turingo mašiną apskaičiuojančią funkciją  $f(x) = x^*$ , t.y. funkciją, kuri pradinius duomenis  $x$  perrašo iš kito galo. Pavyzdžiui, jei  $x = 10110$ , tai  $x^* = 01101$ .

$$\begin{aligned}\delta(q_0, 1, b, b) &= (q_0, 1, b, 1, D, N, K) \\ \delta(q_0, 0, b, b) &= (q_0, 0, b, 0, D, N, K) \\ \delta(q_0, b, b, b) &= (q_1, b, b, b, N, N, N)\end{aligned}$$

$$Q = \{q_0, q_1\}, F = \{q_1\}.$$

### Nedeterminuotosios Turingo mašinos.

Nagrinėjame vienajuostes mašinas. Aibės  $Q, \Sigma, F$  apibrėžiamos taip pat kaip ir vienajuosčių determinuotųjų mašinų atveju. Bet  $\delta$  yra daugia-reikšmė funkcija apibrėžta aibėje  $Q \times \Sigma$ . Jos reikšmių aibė kaip ir determinuotosios atveju yra  $Q \times \Sigma \times \{K, D, N\}$ . Taigi, tarp komandų gali būti ir tokios, kurių lygybių kairiosios pusės sutampa. Pavyzdžiui,

$$\begin{aligned}\delta(q_i, a) &= (q_j, b, K) \\ \delta(q_i, a) &= (q_s, c, D).\end{aligned}$$

Todėl su tais pačiais pradiniais duomenimis galima gauti skirtingus rezultatus, priklausomai nuo to, kurią komandą įvykdėme. Pavyzdžiui, galima rasti nedeterminuotąją Turingo mašiną su abėcėle  $\Sigma = \{0, 1, b\}$ , apskaičiuojančią funkciją:

$$f(x) = \begin{cases} \infty, & \text{jei } x \text{ prasideda vienetu} \\ x, & \text{jei } x \text{ prasideda nuliu} \\ x0, & \text{jei } x \text{ prasideda vienetu} \\ x1, & \text{jei } x \text{ prasideda nuliu} \end{cases}$$

Panašiai apibrėžiamos ir daugiajuostės nedeterminuotosios Turingo mašinos.

## 3.3 Baigtiniai automatai

**3.2 apibrėžimas.** Baigtiniais automatais vadinamos vienajuostės determinuotosios Turingo mašinos  $\langle Q, \Sigma, \delta, F \rangle$ , kurių komandos atrodo šitaip

$$\delta(q_i, a) = (q_j, a, D).$$

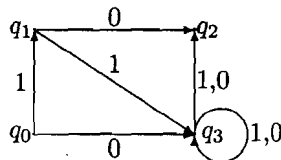
Mašina baigia darbą tada ir tikrai tada, kai sutinka pirmą tuščią ląstelę. Be to, kokie bebūtų  $q_i \in Q$  ir  $a \in \Sigma'$ , atsiras komanda, kurios kairioji pusė yra  $\delta(q_i, a)$ .

Jei Turingo mašina yra baigtinis automatas, tai pradiniai duomenys mašinos darbo metu nekinta. Peržiūrimas žodis iš kairės į dešinę. Gali

keistis tik būsenos. Sutikus pirmą tuščią ląstelę, mašina baigia darbą. Galimi du atvejai: mašina patenka į galutinę būseną arba į poziciją be išeities. Taigi, baigtinis automatas yra greitas pradinių duomenų (abėcėlės  $\Sigma$  žodžių) rūšiavimo algoritmas. Peržiūrėjęs vienus žodžius automatas patenka į galutinę būseną (tų žodžių aibė vadinama **baigtinio automato kalba**), o peržiūrėjęs likusius žodžius, automatas patenka į poziciją be išeities.

Dažniausiai baigtiniai automatai nusakomi pora  $\langle G, F \rangle$ , čia  $G$  – orientuotas grafas,  $F$  – galutinių viršūnių aibė. Grafas  $G$  nusakomas tokiu būdu. Grafo viršūnėmis yra būsenų aibė  $Q$ . Viršūnė pažymėta  $q_0$  vadinama pradine viršūne. Iš kiekvienos viršūnės išeina po vieną ir tą patį skaičių lankų, o būtent, tiek lankų, kiek yra aibėje  $\Sigma'$  raidžių. Iš viršūnės  $q_i$  lankas veda į viršūnę  $q_j$  tada ir tikrai tada, kai komandų tarpe yra  $\delta(q_i, a) = (q_j, a, D)$ . Šiuo atveju lankas žymimas raide  $a$ . Jei yra ne vienas lankas vedantis iš  $q_i$  į  $q_j$  (tarkime jie pažymėti  $b_1, \dots, b_s$ ), tai, patogumo dėlei, pavyzdžiuose brėšime tik vieną lanką ir žymėsime jį  $b_1, \dots, b_s$ . Abėcėlės  $\Sigma'$  žodis  $a_1 \dots a_v$  priklauso baigtinio automato kalbai tada ir tikrai tada, jei, pradėję kelią viršūnėje  $q_0$ , ir perėję  $v$  lankus, atitinkančius žodžio raides, patenkame į galutinę viršūnę (būseną).

Nagrinėsime aibes, kurios gali būti baigtinių automatų kalbomis. Tarkime  $\Sigma' = \{0, 1\}$ . Žodis 10 yra baigtinio automato kalba:



$$F = \{q_2\}.$$

Kai kada patogiau grafus vaizduoti ir lentelė. Pavyzdžio grafo lentelė:

	$q_0$	$q_1$	$q_2$	$q_3$
0	$q_3$	$q_2$	$q_3$	$q_3$
1	$q_1$	$q_3$	$q_3$	$q_3$

Atkreipiame dėmesį, kad nors eidami keliu 101 pereiname galutinę viršūnę, žodis 101 nepriklauso aprašytojo automato kalbai, nes kelias baigiasi viršūnėje  $q_3$ , kuri nėra galutinė.

Nepriklausomai nuo to, kokia yra abėcėlė  $\Sigma'$  ir jos žodis  $w$ , visada galima rasti baigtinį automata, kurio kalba yra žodis  $w$ . Dar daugiau, kokia bebūtų abėcėlė  $\Sigma'$  ir baigtinė jos žodžių aibė  $A$ , galima rasti baigtinį automata, kurio kalba yra aibė  $A$ . Automatas randamas panašiai kaip kad aukščiau aprašytame pavyzdyje, Taigi, **baigtinė aibė yra baigtinio au-**



tomato kalba.

Pažymėkime  $\Sigma^*$  visų galimų abėcėlės  $\Sigma'$  žodžių aibę. Tarkime  $A \subset \Sigma^*$  yra kurio nors baigtinio automato  $\Psi = \langle G, F \rangle$  kalba. Tuomet **papildinys**  $\bar{A} = \Sigma^* - A$  yra baigtinio automato  $\langle G, F' \rangle$ , čia  $F' = Q - F$  kalba.

Nagrinėjame du baigtinius automatus  $\Psi_1 = \langle G_1, F_1 \rangle$  ir  $\Psi_2 = \langle G_2, F_2 \rangle$ , kurių abėcėlėmis yra viena ir ta pati aibė. Tarkime jų kalbomis yra atitinkamai  $A_1$  ir  $A_2$ . Parodysime, kad  $A_1 \cap A_2$  bei  $A_1 \cup A_2$  yra irgi baigtinių automatų kalbos. Tuo tikslu visų pirma apibrėšime dviejų grafų  $G_1, G_2$  Dekarto sandaugą.

Grafo  $G_1 \times G_2$  viršūnėmis yra visos galimos poros  $(q_i, q_j)$ . Čia  $q_i$  yra  $G_1$  viršūnė, o  $q_j$  –  $G_2$  viršūnė. Taigi, jei pirmajame grafe yra  $m$  viršūnių, antrajame  $n$  viršūnių, tai grafe, kuris yra jų Dekarto sandauga, bus  $m \times n$  viršūnių. Iš viršūnės  $(q_i, q_j)$  eina lankas, pažymėtas kuriuo nors bendros abėcėlės simboliu  $a$ , į viršūnę  $(q_k, q_l)$  tada ir tiksliai tada, kai pirmajame grafe yra pažymėtas  $a$  ir einantis iš  $q_i$  į  $q_k$ , o antrajame – lankas pažymėtas taip pat  $a$  ir einantis iš  $q_j$  į  $q_l$ .

$$(q_i, q_j) \xrightarrow{a} (q_k, q_l)$$

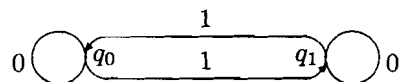
Baigtinio automato  $\langle G_1 \times G_2, F' \rangle$  ( $(q_i, q_j) \in F'$  tada ir tiksliai tada, kai  $q_i \in F_1$  ir  $q_j \in F_2$ ) kalba yra  $A_1 \cap A_2$ . Baigtinio automato  $\langle G_1 \times G_2, F'' \rangle$  ( $(q_i, q_j) \in F''$  tada ir tiksliai tada, kai  $q_i \in F_1$  arba  $q_j \in F_2$ ) kalba yra  $A_1 \cup A_2$ .

**Pavyzdys.** Nagrinėjame du baigtinius automatus su bendra abėcėle  $\Sigma = \{0, 1\}$ .

a) Baigtinio automato  $\Psi_1 = \langle G', F' \rangle$  kalbai (žymėsime ją raide  $A$ ) priklauso visi tie  $\Sigma$  žodžiai, kuriuose vienetų skaičius dalosi iš dviejų. Pavyzdžiui, 010101100 bei 000 priklauso aibei  $A$ , o 1011011 – ne.

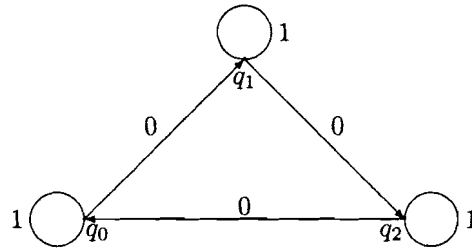
b) Baigtinio automato  $\Psi_2 = \langle G'', F'' \rangle$  kalbai (žymėsime ją raide  $B$ ) priklauso visi tie  $\Sigma$  žodžiai, kuriuose nuliukų skaičius dalosi iš trijų. Pavyzdžiui, 1000 bei 1111 priklauso aibei  $B$ , o 11011 – ne.

Baigtinis automatas  $\Psi_1$  (kilpoms krypties nenurodome, nes tai savaime aišku, jos įeina bei išeina iš tos pačios viršūnės):



$F' = \{q_0\}$ .

Baigtinis automatas  $\Psi_2$ :



$F'' = \{q_0\}$ .

Baigtinių automatų  $\Psi_1$  ir  $\Psi_2$  kalbų sankirta nusakoma automatu  $\langle G' \times G'', F_1 \rangle$ , o sąjunga – automatu  $\langle G' \times G'', F_2 \rangle$ . Čia  $F_1 = \{(q_0, q_0)\}$ ,  $F_2 = \{(q_0, q_0), (q_0, q_1), (q_0, q_2), (q_1, q_0)\}$ , o  $G' \times G''$  yra grafas:

	$(q_0, q_0)$	$(q_0, q_1)$	$(q_0, q_2)$	$(q_1, q_0)$	$(q_1, q_1)$	$(q_1, q_2)$
0	$(q_0, q_1)$	$(q_0, q_2)$	$(q_0, q_0)$	$(q_1, q_1)$	$(q_1, q_2)$	$(q_1, q_0)$
1	$(q_1, q_0)$	$(q_1, q_1)$	$(q_1, q_2)$	$(q_0, q_0)$	$(q_0, q_1)$	$(q_0, q_2)$

Taigi, jau išsiaiškinome, kad: 1) baigtinė žodžių aibė yra baigtinio automato kalba, 2) baigtinio automato kalbos papildinys irgi yra baigtinio automato kalba, 3) dviejų baigtinių automatų kalbų sankirta bei sąjunga yra irgi baigtinio automato kalba.

Dar keletas operacijų, neišvedančių iš baigtinių automatų kalbų.

4) *Dviejų baigtinių automatų kalbų konkatenacija yra irgi baigtinio automato kalba.* Dviejų tos pačios abėcėlės kalbų  $A$ ,  $B$  konkatenacija vadiname žodžių aibę  $\{uv | u \in A, v \in B\}$ .

5) *Baigtinio automato kalbos iteracija yra irgi baigtinio automato kalba.* Kalbos  $A$  iteracija vadiname žodžių aibę  $\{u_1 \dots u_k | u_i \in A; i = 1, 2, \dots, k; k \geq 2\}$ .

6) *Baigtinio automato kalbos atspindys yra irgi baigtinio automato kalba.* Kalbos  $A$  atspindžiu vadiname abėcėlės  $\Sigma$  žodžių aibę  $\{a_1 \dots a_n | a_i \in \Sigma; i = 1, 2, \dots, n; a_n a_{n-1} \dots a_1 \in A\}$ .



### 3.4 Algoritmų sudėtingumas

Nagrinėsime tik daugiajuostes Turingo mašinas  $M = \langle Q, \Sigma, \delta, F \rangle$ , tenkinančias sąlygą, kad su bet kuriais pradiniais duomenimis ( $\Sigma$  žodžiais) po baigtinio žingsnių skaičiaus mašina pereina į galutinę būseną arba patenka į poziciją be išeities (t.y. ji visada baigia darbą). Aprašysime tokių mašinų sudėtingumą. Laikas ir atmintis yra pagrindiniai sudėtingumo kriterijai.  $i(v)$  žymime žodžio  $v$  ilgį. Tarkime, kad  $t(v)$  yra žingsnių, po kurių Turingo mašina baigia darbą, kai pradiniais duomenimis yra  $\Sigma$  žodis  $v$ , skaičius.

**3.3 apibrėžimas.** *Mašinos  $M$  sudėtingumu laiko atžvilgiu vadiname funkciją*

$$T_M(n) = \max\{t(v) | i(v) = n\}.$$

Kaip matome, domimes sudėtingumu blogiausiu atveju, t.y. tarp visų pradinių žodžių ilgio  $n$  randame tą, su kuriuo mašina dirba ilgiausiai ir tų žingsnių skaičių vadiname duotosios mašinos sudėtingumu, kai pradinių duomenų ilgis yra  $n$ .

Panašiai apibrėžiamas ir sudėtingumas atminties  Tarkime  $s(n)$  yra naudojamų ląstelių skaičius visose  tose mašinos darbo metu, kai pradiniais duomenimis yra

**3.4 apibrėžimas.** *Mašinos  $M$  sudėtingumu atminties atžvilgiu vadiname funkciją*

$$S_M(n) = \max\{s(v) | i(v) = n\},.$$

**Turingo mašinos**  $M = \langle Q, \Sigma, \delta, F \rangle$  kalba vadinsime pradinių duomenų ( $\Sigma$  žodžių) aibę, su kuriais po begalinio žingsnių skaičiaus  $M$  patenka į galutinę būseną.

**3.5 apibrėžimas.** *Sakoma, kad aibė (problema)  $K$  išsprendžiama Turingo mašina  $M$ , jei jos kalba sutampa su  $K$ .*

Aibės (problemos)  $K$  išsprendžiamumo sudėtingumu vadiname Turingo mašinos, kurios kalba yra  $K$ , sudėtingumą. Jei mašina determinuotoji, tai kalbama apie problemos determinuotąjį sudėtingumą laiko ar atminties atžvilgiu, o, jei mašina nedeterminuotoji, tai apie nedeterminuotąjį sudėtingumą laiko ar atminties atžvilgiu.

Kai sakoma, kad problemos  $K$  sudėtingumas atminties atžvilgiu yra  $f(n)$ , tai turima omenyje, kad atsiras tokia Turingo mašina  $M$ , kurios kalba yra  $K$  ir, jei pradinių duomenų ilgis yra  $n$ , tai  $M$  skaičiavimo metu panaudoja ne daugiau kaip  $c \cdot f(n)$  ( $c > 0$  yra kuris nors realusis skaičius) darbinių juostų ląstelių (t.y.  $S_M(n) \leq c \cdot f(n)$ ).  $f(n)$  yra funkcija iš  $N$  į  $N$ . Naudojames tokia sudėtingumo atminties atžvilgiu samprata todėl, kad teisingas tvirtinimas:

**3.1 teorema.** *Jei problema  $K$  išsprendžiama determinuotąja (nedeterminuotąja) daugiajuoste Turingo mašina, kurios sudėtingumas atminties atžvilgiu yra  $f(n)$ , tai, koks bebūtų realusis  $c > 0$  atsiras Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas atminties atžvilgiu yra  $c \cdot f(n)$ .*

Kalbant apie nedeterminuotosios Turingo mašinos sudėtingumą atminties atžvilgiu  $f(n)$ , turima omenyje, kad kiekviename skaičiavimo medžio

kelyje naudota atmintis neviršija  $c \cdot f(n)$  darbinių juostų ląstelių.  
Esant tam tikrai sąlygai, panašus rezultatas galioja ir sudėtingumui laiko atžvilgiu:

**3.2 teorema.** *Jei problema  $K$  išsprendžiama determinuotąja (nedeterminuotąja) kuria nors daugiajuoste Turingo mašina, tenkinančia sąlygas:*

- darbinių juostų skaičius yra ne mažesnis kaip du,
- sudėtingumas laiko atžvilgiu yra  $f(n)$  ir

$$\lim_{n \rightarrow \infty} \frac{f(n)}{n} = \infty$$

*tai koks bebūtų realusis skaičius  $c > 0$ , atsirastų Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas laiko atžvilgiu yra  $c \cdot f(n)$ .*

Kai sakoma, kad problemos sudėtingumas laiko atžvilgiu yra  $f(n)$  ir  $f(n)$  tenkina aprašytąją sąlygą, tai suprantama, kad atsirastų tokia Turingo mašina  $M$ , kurios kalba yra  $K$  ir, jei pradinių duomenų ilgis lygus  $n$ , tai  $M$  baigia darbą po ne daugiau kaip  $c \cdot f(n)$  žingsnių (t.y.  $T_M(n) \leq c \cdot f(n)$ ). Kalbant apie nedeterminuotosios Turingo mašinos sudėtingumą laiko atžvilgiu  $f(n)$ , turima omenyje, kad bet kuriame skaičiavimo medžio kelyje, po ne daugiau kaip  $c \cdot f(n)$  žingsnių mašina baigia darbą.

**3.6 apibrėžimas.** *Problema  $K$  priklauso  $DTIME(f(n))$  klasei, jei egzistuoja tokia daugiajuostė determinuotoji Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas laiko atžvilgiu yra  $f(n)$ .*

**3.7 apibrėžimas.** *Problema  $K$  priklauso  $DSPACE(f(n))$  klasei, jei egzistuoja tokia daugiajuostė determinuotoji Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas atminties atžvilgiu yra  $f(n)$ .*

**3.8 apibrėžimas.** *Problema  $K$  priklauso  $NTIME(f(n))$  klasei, jei egzistuoja tokia daugiajuostė nedeterminuotoji Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas laiko atžvilgiu yra  $f(n)$ .*

**3.9 apibrėžimas.** *Problema  $K$  priklauso  $NSPACE(f(n))$  klasei, jei egzistuoja tokia daugiajuostė nedeterminuotoji Turingo mašina, kurios kalba yra  $K$  ir kurios sudėtingumas atminties atžvilgiu yra  $f(n)$ .*

Apibrėšime kai kurias sudėtingumo klases:

- $L$  yra problemų klasė, kurios išsprendžiamumo determinuotas sudėtingumas atminties atžvilgiu yra  $\log n$ , t.y.  $L = DSPACE(\log n)$ ,

- $NL$  yra problemų klasė, kurios išsprendžiamumo nedeterminuotas sudėtingumas atminties atžvilgiu yra  $\log n$ , t.y.  $NL = NSPACE(\log n)$ ,
- $P$  yra problemų klasė, kurios išsprendžiamumo determinuotas sudėtingumas laiko atžvilgiu yra polinomas  $n^k$  ( $k$  – kuris nors natūralusis skaičius), t.y.  $P = DTIME(n^k)$ ,
- $NP$  yra problemų klasė, kurios išsprendžiamumo nedeterminuotas sudėtingumas laiko atžvilgiu yra polinomas  $n^k$  ( $k \in N$ ), t.y.  $NP = NTIME(n^k)$ ,
- $PSPACE$  yra problemų klasė, kurios išsprendžiamumo determinuotas sudėtingumas atminties atžvilgiu yra polinomas  $n^k$  ( $k \in N$ ), t.y.  $PSPACE = DSPACE(n^k)$ ,
- $EXP$  yra problemų klasė, kurios išsprendžiamumo determinuotas sudėtingumas laiko atžvilgiu yra  $2^{n^k}$  ( $k \in N$ ), t.y.  $EXP = DTIME(2^{n^k})$ .

Tarp išvardintųjų klasių egzistuoja toks ryšys:

$$L \subset NL \subset P \subset NP \subset PSPACE \subset EXP.$$

Priminsime, kad vadovėlyje simbolis  $\subset$  vartojamas tokia prasme, kad  $A$  gali būti ir lygus  $B$ . Todėl, kai rašome  $P \subset NP$ , tai nereiškia, kad  $P$  nelygus  $NP$ . Tai kol kas nežinoma. Lygiai taip pat nežinoma ar lygios ir kai kurios kitos išvardintosios klasės. Yra tik įrodyta, kad  $NL \neq PSPACE$  bei  $P \neq EXP$ .

### 3.5 Primityviai rekursyvios funkcijos

Aprašysime formaliąją sistemą. Funkcijos, kurias galima gauti toje sistemoje, vadinsime rekursyviomis. Formalioji sistema sudaro bazinės funkcijos ir operatoriai, kurie taikomi turimoms funkcijoms, o rezultatas – naujosios funkcijos. Visų pirma aprašysime vieną rekursyviųjų funkcijų poaibį, vadinamąsias primitivias rekursyvias funkcijas.

**Bazinės funkcijos:** – tai konstanta 0, paskesniojo nario funkcija  $s(x) = x + 1$  ir projekcijų funkcijos  $pr_p^i(x_1, \dots, x_p) = x_i$  ( $p \geq 1; 1 \leq i \leq p$ ).

Iš bazinių funkcijų gaunamos naujos naudojantis dviem operatoriais: kompozicijos ir primityviosios rekursijos.

1. **Kompozicijos operatorius.** Tarkime, yra  $n + 1$  funkcijų:

$$f(x_1, \dots, x_n), g_1(x_1^1, \dots, x_{m_1}^1), \dots, g_n(x_1^n, \dots, x_{m_n}^n) \quad (3.1)$$

Sakome, kad funkcija  $f(g_1(x_1^1, \dots, x_{m_1}^1), \dots, g_n(x_1^n, \dots, x_{m_n}^n))$  gauta iš (3.1) funkcijų panaudojus kompozicijos operatorių.

**2. Primityviosios rekursijos operatorius.** Tarkime, yra dvi funkcijos  $g(x_1, \dots, x_{n-1}), h(x_1, \dots, x_{n+1})$ . Viena jų yra  $n-1$  argumento, o antroji  $n+1$  argumento. Apibrėžiame naują  $n$  argumentų funkciją  $f(x_1, \dots, x_n)$  pagal tokią schemą:

$$\begin{aligned} f(x_1, \dots, x_{n-1}, 0) &= g(x_1, \dots, x_{n-1}) \\ f(x_1, \dots, x_{n-1}, y+1) &= h(x_1, \dots, x_{n-1}, y, f(x_1, \dots, x_{n-1}, y)) \end{aligned}$$

Kai  $n=1$ , funkcija  $g$  yra konstanta. Sakysime, kad funkcija  $f$  gauta iš funkcijų  $g, h$ , panaudojus primityviosios rekursijos operatorių. Pabrėždami, kad paskutiniojo argumento reikšmė nelygi nuliui, rašome  $y+1$ . Kaip matome, funkcija apibrėžta rekursyviai. Norint apskaičiuoti funkcijos  $f(x_1, \dots, x_{n-1}, y+1)$  reikšmę, iš pradžių reikia rasti funkcijos reikšmę, kai paskutiniojo argumento reikšmė vienetu mažesnė. Rekursija (grįžimas) primityvi. Argumento reikšmė sumažinama vienetu.

**3.10 apibrėžimas.** *Pati mažiausia aibė, kuriai priklauso bazinės funkcijos ir kuri uždara kompozicijos bei primityviosios rekursijos atžvilgiu, vadinama primityviai rekursyvių funkcijų aibe (klase).*

Primityviai rekursinių funkcijų aibę žymėsime  $PR$ . Primityviai rekursyvios funkcijos apibrėžtos su bet kuriomis argumentų reikšmėmis. Tokias funkcijas vadinsime *visur apibrėžtomis funkcijomis*.

Aibės  $A$  charakteringoji funkcija  $\kappa$  apibrėžiama tokiu būdu:

$$\kappa_A(x) = \begin{cases} 1, & \text{jei } x \in A \\ 0, & \text{jei } x \notin A \end{cases}$$

**3.11 apibrėžimas.** *Natūraliųjų skaičių poaibis vadinamas primityviai rekursyviu, jei jo charakteringoji funkcija yra primityviai rekursyvi.*

**Pavyzdžiai:**

1. Konstanta 1 priklauso  $PR$  klasei, nes ją  $s(0)$  galima gauti iš bazinių funkcijų  $s(x)$ , 0, naudojantis kompozicijos operatoriumi.

2. Bet kuri konstanta  $n$  priklauso  $PR$  klasei, nes  $s(s(\dots s(0))\dots) \in PR$ . Čia  $n-1$  kartų taikėme kompoziciją.

3.  $x+n \in PR$ , nes  $s(s(\dots s(x))\dots) = x+n$ . Kompoziciją taikėme taip pat  $n-1$  kartų.

4.  $s(pr_3^3(x, y, z)) \in PR$ . Gauta iš bazinių funkcijų  $s(x), pr_3^3(x, y, z)$ , pritaikius kompozicijos operatorių.

5. Parodysime, kad  $x + y \in PR$ . Funkcija  $x + y$  gaunama iš  $pr_1^1$  bei  $s(pr_3^3(x, y, z))$  naudojantis primitiviosios rekursijos operatoriumi.

$$\begin{aligned} x + 0 &= pr_1^1(x) = x \\ x + (y + 1) &= (x + y) + 1 = s(pr_3^3(x, y, x + y)) = s(x + y) \end{aligned}$$

Apibrėžiame funkcijas  $sg\ x$ ,  $\bar{s}g\ x$ , bei  $x \dot{-} y$ :

$$sg\ x = \begin{cases} 1, & \text{jei } x > 0 \\ 0, & \text{jei } x = 0 \end{cases}$$

$$\bar{s}g\ x = \begin{cases} 1, & \text{jei } x = 0 \\ 0, & \text{jei } x > 0 \end{cases}$$

$$x \dot{-} y = \begin{cases} x - y, & \text{jei } x > y \\ 0, & \text{jei } x \leq y \end{cases}$$

Įrodymą, kad  $sg\ x$ ,  $\bar{s}g\ x$  bei  $x \dot{-} y$  yra primitiviai rekursyvios, paliekame pratyboms.

Funkcija  $|x - y|$  irgi primitiviai rekursyvi, nes  $|x - y| = (x \dot{-} y) + (y \dot{-} x)$ .

**3.1 lema** Jei  $g(x_1, \dots, x_n)$  primitiviai rekursyvi, tai

$$f(x_1, \dots, x_n) = \sum_{i=0}^{x_n} g(x_1, \dots, x_{n-1}, i)$$

taip pat primitiviai rekursyvi.

*Įrodymas.*

$$\begin{aligned} f(x_1, \dots, x_{n-1}, 0) &= g(x_1, \dots, x_{n-1}, 0) \\ f(x_1, \dots, x_{n-1}, y + 1) &= f(x_1, \dots, x_{n-1}, y) + g(x_1, \dots, x_{n-1}, y + 1) \end{aligned}$$

Todėl  $f(x_1, \dots, x_n)$  gaunama pritaikius primitiviosios rekursijos operatorių funkcijoms  $g(x_1, \dots, x_{n-1}, 0)$  bei  $h(x_1, \dots, x_{n+1}) = g(x_1, \dots, x_{n-1}, s(x_n)) + x_{n+1}$ , kurios yra primitiviai rekursyvios. Lema įrodyta.

Dalijame  $x$  iš  $y$ . Sveikąją dalį žymime  $[x/y]$ , o liekaną –  $rest(x, y)$ . Tarkime, kad  $[x/0] = x$ , bei  $rest(x, 0) = x$ . Remiantis 3.1 lema, nesunku įrodyti, kad  $[x/y]$  yra primitiviai rekursyvi. Tuo tikslu nagrinėjame skaičių seką:

$$1 \cdot y \dot{-} x, 2 \cdot y \dot{-} x, \dots, n \cdot y \dot{-} x, \dots, x \cdot y \dot{-} x.$$

$$\text{čia } g(x, y, k) = \bar{sg}(k \cdot y)$$

tai is L1:

$$h(x, y, k) = \sum_{i=0}^k g(x, y, i) \leftarrow$$

$$\sigma[x/y] = h(x, y, x) \leftarrow 1$$

nes, vėly  
sg(10 · y · x)  
įtrauktas

Sveikoji dalis lygi nulių sekoje skaičiui. Todėl

$$[x/y] = \sum_{i=0}^x \bar{sg}(iy \cdot x) \cdot 1.$$

$$\text{rest}(x, y) = x \cdot (y \cdot [x/y]).$$

Algoritmų teorijoje dažnai aptinkamas funkcijos apibrėžimas dalimis.

$$f(x_1, \dots, x_n) = \begin{cases} f_1(x_1, \dots, x_n), & \text{jei } \alpha_1(x_1, \dots, x_n) = 0 \\ \dots \\ f_s(x_1, \dots, x_n), & \text{jei } \alpha_s(x_1, \dots, x_n) = 0 \\ f_{s+1}(x_1, \dots, x_n), & \text{likusiais atvejais} \end{cases}$$

Be to, su bet kuriuo reikšmių  $(x_1, \dots, x_n)$  rinkiniu, tik viena iš  $\alpha_i$  gali būti lygi nuliui.

Jei funkcijos  $f_i$  ( $i = 1, \dots, s+1$ ) bei  $\alpha_i$  ( $i = 1, \dots, s$ ) primitiviai rekursyvios, tai ir  $f$  primitiviai rekursyvi, nes teisinga lygybė:

$$\begin{aligned} f(x_1, \dots, x_n) &= f_1(x_1, \dots, x_n) \cdot \bar{sg}\alpha_1(x_1, \dots, x_n) + \dots + \\ &+ f_s(x_1, \dots, x_n) \cdot \bar{sg}\alpha_s(x_1, \dots, x_n) + \\ &+ f_{s+1}(x_1, \dots, x_n) \cdot sg(\alpha_1(x_1, \dots, x_n) \cdot \dots \cdot \alpha_s(x_1, \dots, x_n)) \end{aligned}$$

Sąlygas  $\alpha_i$  galima pakeisti

$$\alpha_i = \beta_i, \alpha_i \leq \beta_i, \alpha_i < \beta_i,$$

(suprantama,  $\alpha_i, \beta_i$  primitiviai rekursyvios), nes jos redukuojamos į lygtis

$$|\alpha_i - \beta_i| = 0, \alpha_i \cdot \beta_i = 0, \bar{sg}(\beta_i \cdot \alpha_i) = 0.$$

### 3.6 Minimizavimo operatorius

Tarkime, yra  $n$  argumentų funkcija  $f$ . Apibrėžiame naują, taip pat  $n$  argumentų, funkciją  $g(x_1, \dots, x_n)$ , kurios reikšmė lygi mažiausiam  $y$ , su kuriuo  $f(x_1, \dots, x_{n-1}, y) = x_n$ . Įrodyta: jeigu  $f$  yra net primitiviai rekursyvi,  $g$  gali ir nebūti algoritmiškai apskaičiuojama funkcija. Todėl nusakydami naują funkciją  $g$ , mes privalome nurodyti ir metodą kaip ieškoti mažiausio  $y$ :

Jei  $f(x_1, \dots, x_{n-1}, 0) = x_n$ , tai funkcijos  $g$  reikšmė lygi 0, jei ne, tai tikriname ar  $f(x_1, \dots, x_{n-1}, 1) = x_n$ .

Jei  $f(x_1, \dots, x_{n-1}, 1) = x_n$ , tai funkcijos  $g$  reikšmė lygi 1, jei ne, tai tikriname ar  $f(x_1, \dots, x_{n-1}, 2) = x_n$  ir t.t.

Funkcija  $g$  gali būti ir dalinė, t.y. su kai kuriomis argumentų reikšmėmis ji gali būti ir neapibrėžta, nes, pavyzdžiui, tokio  $y$ , tenkinančio aprašytąją



lygybę, gali ir nebūti. Tačiau gali ir būti toks  $m$ , kad  $f(x_1, \dots, x_{n-1}, m) = x_n$ , bet, jei su kuriuo nors  $i < m$  funkcija  $f(x_1, \dots, x_{n-1}, i)$  neapibrėžta, tai ir  $g$  bus neapibrėžta.

Sakysime, kad  $g$  gauta pritaikius minimizacijos operatorių funkcijai  $f$ , ir žymime

$$g(x_1, \dots, x_n) = \mu_y(f(x_1, \dots, x_{n-1}, y) = x_n).$$

Naudojantis minimizavimo operatoriumi, gaunama dalinė skirtumo funkcija  $x - y = \mu_z(y + z = x)$ .

Funkcija  $f(x) = \mu_y(y - (x + 1) = 0)$  neapibrėžta su jokia  $x \in N$ , nors kiekvienam  $x$  atsiras mažiausias  $y$ . Jis lygus  $x + 1$ .

**3.12 apibrėžimas.** *Pati mažiausia aibė, kuriai priklauso bazinės funkcijos ir kuri uždara kompozicijos, primityviosios rekursijos bei minimizavimo atžvilgiu, vadinama dalinių rekursyviųjų funkcijų aibe (klase).*

Funkcija  $x - y$  yra dalinė rekursyvioji, bet ji nėra primityviai rekursyvi. Iš 3.10 ir 3.12 apibrėžimų išplaukia, kad kiekviena primityviai rekursyvi funkcija yra ir dalinė rekursyvioji.

**3.13 apibrėžimas.** *Visur apibrėžta dalinė rekursyvioji funkcija vadinama bendraja rekursyviaja funkcija.*

Dalinių rekursyviųjų funkcijų aibę žymėsime  $DR$ , o bendrųjų rekursyviųjų –  $BR$ . Iš apibrėžimų išplaukia, kad  $PR \subseteq BR \subset DR$ . Vėliau parodysime, kad egzistuoja bendrosios rekursyvosios funkcijos, kurios nėra primityviai rekursyvos. Tokiu būdu  $PR \subset BR \subset DR$ .

### 3.7 Porų numeravimas

Bet kurių dviejų skaičiųjų aibių Dekarto sandauga skaiti, todėl aibė  $A = \{(x, y) : x, y \in N\}$  taip pat skaiti. Visas poras išrašysime tam tikra tvarka. Jei  $x + y < u + v$ , tai  $(x, y)$  sekoje pasitaikys anksčiau negu kad  $(u, v)$ . Jei  $x + y = u + v$  ir  $x < u$ , tai pora  $(x, y)$  taip pat bus randama anksčiau. Turime tokią porų seką:

$$(0, 0), (0, 1), (1, 0), (0, 2), (1, 1), (2, 0), (0, 3), \dots$$

Kiekvienai porai priskirsime po numerį. Numeruoti pradedame nuo nulio. Jei pora yra  $i$ -toje vietoje, tai jos numeris bus  $i - 1$ . Poros  $(x, y)$

numerį žymėsime  $\alpha_2(x, y)$ . Tada  $\alpha_2(0, 0) = 0, \alpha_2(0, 1) = 1, \alpha_2(1, 0) = 2, \dots$ . Taip numeruoti poras pasiūlė G.Cantor.

**3.2 lema.** Poros  $(x, y)$  numeris apskaičiuojamas naudojantis funkcija

$$\alpha_2(x, y) = \frac{(x + y)^2 + 3x + y}{2}.$$

*Irodymas.* Pora  $(x, y)$  yra atkarpoje

$$(0, x + y), (1, x + y - 1), \dots, (x, y), \dots, (x + y, 0)$$

Prieš atkarpą yra viena tokia pora  $(u, v)$ , kad  $u + v = 0$ , dvi poros  $(u, v)$ , kuriose  $u + v = 1$  ir t.t. Iš viso  $1 + 2 + \dots + (x + y)$  porų, t.y.

$$\frac{(x + y)(x + y + 1)}{2}$$

Pora  $(x, y)$  yra nagrinėjamosios atkarpos  $x + 1$ -oje pozicijoje. Kadangi numeravimas prasideda nuo nulio, tai

$$\alpha_2(x, y) = \frac{(x + y)(x + y + 1)}{2} + x = \frac{(x + y)^2 + 3x + y}{2}$$

Lema įrodyta.

Poros  $(x, y)$  kairiuoju nariu vadinsime  $x$ , o dešiniuoju –  $y$ . Kairiojo nario funkciją žymime  $\pi_2^1$ , o dešiniojo  $\pi_2^2$ . Jos abi yra vieno argumento funkcijos. Jei poros  $(x, y)$  numeris  $n$ , tai  $\pi_2^1(n) = x$ , o  $\pi_2^2(n) = y$ . Jos tenkina tokias savybes:

$$\begin{aligned} \alpha_2(\pi_2^1(n), \pi_2^2(n)) &= n, \\ \pi_2^1(\alpha_2(x, y)) &= x, \\ \pi_2^2(\alpha_2(x, y)) &= y. \end{aligned}$$

Toliau nesinaudosime šiuo konkrečiu Cantoro numeravimu. Svarbu tik faktas, kad toks porų numeravimas galimas, kai  $\alpha_2(x, y), \pi_2^1(n), \pi_2^2(n)$  yra primityviai rekursyvios funkcijos.

$\pi_2^1(n)$  apskaičiuojama tokiu būdu:

$$\pi_2^1(n) = x = n - \frac{1}{2} \left[ \frac{[\sqrt{8n+1}] + 1}{2} \right] \left[ \frac{[\sqrt{8n+1}] - 1}{2} \right]$$

Naudojantis porų numeravimo funkcija, aprašomas trejetų, ketvertų ir t.t. numeravimas. Pavyzdžiui,

$$\alpha_3(x_1, x_2, x_3) = \alpha_2(x_1, \alpha_2(x_2, x_3))$$

Taip numeruojant pirmieji penki sekos nariai yra trejetai:

$(0, 0, 0), (0, 0, 1), (1, 0, 0), (0, 1, 0), (1, 0, 1), \dots$

Bet kurio  $n$  dedamųjų vektoriaus numeris apibrėžiamas rekursija

$$\alpha_n(x_1, \dots, x_n) = \alpha_2(x_1, \alpha_{n-1}(x_2, \dots, x_n))$$

Jei  $\alpha_n(x_1, \dots, x_n) = x$ , tai  $\pi_n^i(x) = x_i$ . Gauname, kad

$$\begin{aligned} x_1 &= \pi_2^1(x), \\ x_2 &= \pi_2^1(\pi_2^2(x)), \\ x_3 &= \pi_2^1(\pi_2^2(\pi_2^2(x))), \\ &\dots \\ x_{n-1} &= \pi_2^1(\pi_2^2(\pi_2^2(\dots(\pi_2^2(x))\dots))), \text{ čia } \pi_2^2 \text{ įeina } n-2 \text{ kartus,} \\ x_n &= \pi_2^2(\pi_2^2(\pi_2^2(\dots(\pi_2^2(x))\dots))), \text{ čia } \pi_2^2 \text{ įeina } n-1 \text{ kartą.} \end{aligned}$$

Kadangi funkcijos  $\alpha_n, \pi_n^i$  (jos vadinamos Cantoro funkcijomis) gaunamos pritaikius kompozicijos operatorių primityviai rekursyvioms funkcijoms, tai ir jos pačios yra primityviai rekursyvios.

### 3.8 Baigtinumo problema

Nagrinėkime vienuose determinuotąsias Turingo mašinas. Be to, tarkime, kad jos tenkina sąlygas (tokias mašinas vadinsime standartinėmis):

a) kai mašina po baigtinio žingsnių skaičiaus baigia darbą, t.y. patenka į galutinę būseną, jos skaitymo galvutė turi būti ties pirmąja (iš kairės) netuščia ląstele,

b) be tuščios ląstelės simbolio  $b$ , abėcėlėje yra dar du  $(0, 1)$ . Be to, pradiniai duomenys bei rezultatai yra dvejetainiai skaičiai,

c) galutinė būseną yra tik viena.

Būsenas, kaip ir įprasta, žymime raidėmis  $q$  su indeksais, perėjimų funkciją – raide  $\delta$ , o perėjimų komandas –  $\delta(q_i, x) = (q_j, x', Y)$  (čia  $Y$  žymime vieną iš raidžių  $K, D, N$ ). Galutinę būseną žymime  $q_1$ .

Yra žinoma, kad ir kokia būtų Turingo mašina, galima rasti jai ekvivalenčią, t.y. apskaičiuojančią tą pačią funkciją, standartinę. Taigi Turingo mašinos abėcėlė yra tokia:

$$A = \{0, 1, b, q, 2, \dots, 9, \delta, =, (, ), K, D, N\} \cup \{, \}.$$

### 3.3 lema Standartinių Turingo mašinų aibė yra skaičioji.

*Įrodymas.* Remiantis 1.5 teorema, visų galimų žodžių aibė  $A^*$  yra skaiti. Tie žodžiai, kurie yra kurios nors standartinės Turing mašinos perėjimų funkcija, sudaro begalinę  $A^*$  poaibį, kuris yra skaitus, nes bet kuris skaičiosios aibės poaibis yra baigtinis arba skaitus. Lema įrodyta.

Tarkime, kad  $T_0, T_1, T_2, \dots$  – pilnas sąrašas standartinių Turingo mašinų, o  $\varphi_0, \varphi_1, \varphi_2, \dots$  – vieno argumento dalinės rekursyvos funkcijos, kurias apskaičiuoja atitinkamos Turingo mašinos, t.y.  $\varphi_i$  žymi funkciją, kurią apskaičiuoja mašina  $T_i$ .

*Baigtinumo problema:*

*Ar egzistuoja algoritmas, kuriuo naudojantis, pagal bet kurią natūraliųjų skaičių porą  $(m, n)$  galima pasakyti, ar Turingo mašina  $T_m$  su pradiniais duomenimis  $n$  (t.y. juostoje pradinio laiko momentu yra natūralųjį  $n$  atitinkantis dvejetainis skaičius) baigia darbą (t.y. po baigtinio žingsnių skaičiaus pereina į galutinę būseną), ar ne?*

Jei dalinė funkcija  $f(x)$  apibrėžta su  $x = k$ , tai žymime  $f(k) < \infty$ , jei ne, tai  $f(k) = \infty$ .

Baigtinumo problemą galima apibrėžti ir tokiu būdu:

*Ar egzistuoja algoritmas, kuriuo galima nustatyti ar  $\varphi_m(n) < \infty$ , ar  $\varphi_m(n) = \infty$  ( $m, n$  – bet kurie natūralieji skaičiai)?*

Aibė vadinama *rekursyviaja*, jei jos charakteringą funkciją yra kuri nors visur apibrėžta rekursyvioji funkcija. Kai kalbama apie aibes, kurias sudaro ne skaičiai, o kitokie elementai (formulės, funkcijos, Turingo mašinos ir kt.), tai dažniausiai užuot sakę *(ne)rekursyvi aibė*, sakome *(ne)išsprendžiama aibė (klasė, problema)*.

### 3.3 teorema. Baigtinumo problema neišsprendžiama.

*Įrodymas.* Parodysime, kad tarp visų galimų algoritmų nėra tokio, kuris išsprendžia baigtinumo problemą. Nagrinėjame funkciją

$$g(\alpha_2(x, y)) = \begin{cases} 1, & \text{jei } \varphi_x(y) < \infty \\ 0, & \text{jei } \varphi_x(y) = \infty \end{cases}$$

Tarkime, kad algoritmas, apie kurį kalbama baigtinumo problemoje, egzistuoja. Taigi atsiras tokia standartinė Turingo mašina, kad su pradiniais duomenimis  $\alpha_2(x, y)$  po baigtinio skaičiaus žingsnių mašina pereis į galutinę būseną ir juostoje bus tik viena natūščia ląstelė. Joje bus 1 arba 0 ir ties ja bus skaitymo galvutė. Tuomet  $g(\alpha_2(x, y))$  yra bendroji

rekursyvioji funkcija ir atsiras Turingo mašina, apskaičiuojanti funkciją  $\psi(x)$ :

$$\psi(x) = \begin{cases} 1, & \text{jei } g(\alpha_2(x, x)) = 0 \\ \infty, & \text{jei } g(\alpha_2(x, x)) = 1 \end{cases}$$

Ją gauname tokiu būdu. Perėjimų funkcijoje visas  $q_1$  įėtis pakeičiame  $q_k$  ( $q_k$  – kuri nors nauja būseną), bei papildome:

$$\delta(q_k, 1) = (q_k, 1, D)$$

$$\delta(q_k, b) = (q_k, b, D)$$

$$\delta(q_k, 0) = (q_1, 1, N)$$

Tarkime,  $l$  yra Turingo mašinos, apskaičiuojančios  $\psi$ , kuris nors numeris. Tuomet  $l$  bus ir  $\psi$  numeris, t.y.  $\psi = \varphi_l$ . Aiškinames, ar  $\psi(l) < \infty$ . Iš prielaidos, kad egzistuoja algoritmas, apie kurį kalbama baigtinumo problemoje, gauname prieštarą:

- 1) jei  $\psi(l) < \infty$ , tai  $g(\alpha_2(l, l)) = 0$  ir  $\varphi_l(l) = \infty$ , t.y.  $\psi(l) = \infty$ ;
- 2) jei  $\psi(l) = \infty$ , tai  $g(\alpha_2(l, l)) = 1$  ir  $\varphi_l(l) < \infty$ , t.y.  $\psi(l) < \infty$ .

Teorema įrodyta.

Bendresnę teoremą 1953 m. įrodė H.G.Rice:

*Tarkime  $X$  dalinių rekursyviųjų vieno argumento funkcijų aibė. Jei  $X$  netuščia ir nesutampa su visų dalinių rekursyviųjų vieno argumento funkcijų aibe, tai*

$$A = \{x : x \in N \text{ ir } \varphi_x \in X\}$$

*yra nerekursyvi.*

Remiantis Rice teorema, galima gauti daug nerekursyvių aibių. Pavyzdžiui:

a)  $X$  sudaro visos vieno argumento tapčiai lygios nuliui primitiviai rekursyvios funkcijos,

b)  $X$  sudaro visos bendrosios rekursyviosios vieno argumento funkcijos.

Neišsprendžiama ir tokia problema:

*Ar bet kurios dvi Turingo mašinos apskaičiuoja vieną ir tą pačią dalinę rekursyviąją funkciją?*

### 3.9 Rekursyviai skaičios aibės

Pateikiame tris skirtingus rekursyviai skaičios aibės apibrėžimus.

**3.14 apibrėžimas.** Sakome, kad aibė yra rekursyviai skaiti, jei ji sutampa su kurios nors dalinės rekursyvosios funkcijos apibrėžimo sritimi.

**3.15 apibrėžimas.** Netuščia aibė rekursyviai skaiti, jei ji sutampa su kurios nors primityviai rekursyvosios funkcijos reikšmių aibe.

**3.16 apibrėžimas.** Aibė  $A$  yra rekursyviai skaiti, jei egzistuoja tokia primityviai rekursyvi funkcija  $f(a, x)$ , kad lygtis  $f(a, x) = 0$  turi sprendinį  $x$  tada ir tik tai tada, kai  $a \in A$ .

Visi trys apibrėžimai, netuščios aibės atveju ekvivalentūs. Įrodysime tai tik keliems atvejams.

1. Tarkime aibė  $A$  rekursyviai skaiti pagal 3.15 apibrėžimą, t.y. ji netuščia ir yra tokia primityviai rekursyvi funkcija  $h(x)$ , kad  $A = \{h(0), h(1), h(2), \dots\}$ . Parodysime, kad egzistuoja tokia dalinė rekursyvi funkcija, kurios apibrėžimo sritis sutampa su  $A$  (3.14 apibrėžimas).  
Tokia funkcija yra

$$f(x) = \mu_z(h(z) = x)$$

Funkcija  $f(x)$  dalinė rekursyvi, nes gauta pritaikius minimizavimo operatorių primityviai rekursyviai funkcijai. Be to, jei  $x \in A$ , t.y.  $x = h(i)$ , tai atsiras toks  $j \leq i$ , kad  $h(j) = x$ . Vadinasi,  $f(x)$  apibrėžta. Jei  $x \notin A$ , tai su bet kuriuo  $i$  funkcija  $h(i) \neq x$  ir  $f(x)$  neapibrėžta.

2. Tarkime, aibė  $A$  rekursyviai skaiti pagal 3.15 apibrėžimą. Tada  $A$  sutampa su primityviai rekursyvosios funkcijos  $h(x)$  reikšmių aibe, t.y.  $A = \{h(0), h(1), h(2), \dots\}$ . Tuomet  $|h(x) - a| = 0$  primityviai rekursyvi (žr. skyrelį *Primityviai rekursyvosios funkcijos*) ir ji turi sprendinį tada ir tik tai tada, kai  $a \in A$ . Taigi,  $A$  rekursyviai skaiti pagal 3.16 apibrėžimą.  
 $f(a, x) = |h(x) - a|$ .

3. Tarkime, kad egzistuoja tokia primityviai rekursyvi funkcija  $f(a, x)$ , kad lygtis  $f(a, x) = 0$  turi sprendinį tada ir tik tai tada, kai  $a \in A$ , t.y.  $A$  rekursyviai skaiti pagal 3.16 apibrėžimą.  $A$  netuščia ir, tarkime, kad  $d$  yra kuris nors jos elementas. Nagrinėjame funkciją

$$h(t) = \pi_2^1(t) \bar{s} g f(\pi_2^1(t), \pi_2^2(t)) + d \cdot s g f(\pi_2^1(t), \pi_2^2(t)).$$

Ji yra primityviai rekursyvi, nes gauta pritaikius kompozicijos operatorių primityviai rekursyvioms funkcijoms. Tarkime  $a \in A$ ,  $x_0$  yra lygties

$f(a, x_0) = 0$  sprendinys ir  $t_0 = \alpha_2(a, x_0)$ . Tuomet  $\bar{sg}f(\pi_2^1(t_0), \pi_2^2(t_0)) = \bar{sg}f(a, x_0) = 1$ ,  $sgf(\pi_2^1(t_0), \pi_2^2(t_0)) = 0$  ir  $h(t_0) = \pi_2^1(t_0) = a$  t.y.  $h(t_0) \in A$ .

Tarkime, kad  $a \notin A$ . Tuomet su bet kuriuo  $x_0$  funkcija  $f(a, x_0) \neq 0$ . Kad ir koks būtų  $t_0 = \alpha_2(a, x_0)$   $\bar{sg}f(\pi_2^1(t_0), \pi_2^2(t_0)) = 0$ . Tuo tarpu  $sgf(\pi_2^1(t_0), \pi_2^2(t_0)) = 1$  su bet kuriuo  $t_0 = \alpha_2(a, x_0)$  ir  $h(t_0) = d$ , t.y.  $h(t_0) \in A$ . Taigi  $A$  rekursyviai skaiti pagal 3.15 apibrėžimą.

Norėdami atkreipti dėmesį į rekursyvių ir rekursyviai skaičių aibių skirtumą, pateiksime dar tokį apibrėžimą (palyginkite jį su 3.11 apibrėžimu).

*Tarkime, kad  $\kappa_A(x)$  dalinė rekursyvioji funkcija, tenkinanti sąlyga*

$$\kappa_A(x) = \begin{cases} 1, & \text{jei } x \in A \\ \infty, & \text{jei } x \notin A \end{cases}$$

*Tuomet  $A$  yra rekursyviai skaiti.*

Kuo skiriasi skaičiosios nuo rekursyviai skaičių aibių? Tai paaiškės vėliau. Kol kas tik pastebėsime, kad jei  $A$  yra skaiti ir abipusiškai viena-reikšmę  $N$  ir  $A$  atitiktį galime nusakyti kuria nors primityviai rekursyvia funkcija  $h(x)$  ( $A = \{h(0), h(1), \dots\}$ ), tai  $A$  taip pat ir rekursyviai skaiti (pagal 3.15 apibrėžimą). Kiekvienas natūraliųjų skaičių aibės poaibis yra baigtinis arba skaitusis. Vėliau matysime, kad egzistuoja begaliniai natūraliųjų skaičių poaibiai, kurie nėra rekursyviai skaitūs. Pateikiame keletą rekursyviai skaičių aibių pavyzdžių.

#### **Pavyzdžiai:**

1. Tuščia aibė rekursyviai skaiti, nes ji sutampa su dalinės rekursyvosios funkcijos  $\mu_z(z + (x + 1) = 0)$  (ji su jokia reikšme neapibrėžta) apibrėžimo sritimi (naudojamės 3.14 apibrėžimu).

2.  $N_- = \{1, 2, 3, \dots\}$  rekursyviai skaiti, nes sutampa su primityviai rekursyvosios funkcijos  $s(x)$  reikšmių aibe.

3. Baigtinio skaičiaus rekursyviai skaičių aibių sąjunga bei sankirta yra rekursyviai skaičios aibės.

Tarkime  $A_1, A_2, \dots, A_n$  rekursyviai skaičios aibės. Egzistuoja (pagal 3.16 apibrėžimą) tokios primityviai rekursyvosios funkcijos  $f_i(a, x)$ , kad  $f_i(a, x) = 0$  turi sprendinį tada ir tik tada, kai  $a \in A_i$ .

a) Sankirtos atveju konstruojame tokią primityviai rekursyvią funkciją

$$f(a, x) = f_1(a, \pi_n^1(x)) + \dots + f_n(a, \pi_n^n(x))$$

Su reikšmėmis  $x_1^0, \dots, x_n^0$  lygybės  $f(a, x_i^0) = 0$  ( $i = 1, \dots, n$ ) galioja tada ir tik tada, kai egzistuoja  $a \in A_1 \cap A_2 \cap \dots \cap A_n$ . Tarkime,  $x^0 =$

$\alpha_n(x_1^0, \dots, x_n^0)$ . Tuomet  $f(a, x^0) = 0$ .

b) Sąjungos atveju

$$f(a, x) = f_1(a, \pi_n^1(x)) \cdot \dots \cdot f_n(a, \pi_n^n(x))$$

**Kai kurios rekursyviai skaičių aibių savybės:**

1. Kiekviena rekursyvi aibė yra rekursyviai skaiti.

Tarkime  $\kappa_A(x)$  yra bendroji rekursyvioji charakteringoji aibės  $A$  funkcija. Tuomet  $A$  sutampa su dalinės rekursyvios funkcijos  $f(x) = \kappa_A(x) - 1$  apibrėžimo sritimi.

2. Baigtinės aibės yra rekursyvios, kartu ir rekursyviai skaičios.

Tarkime,  $A = \{a_1, \dots, a_m\}$ . Tuomet primitiviai rekursyvi  $\kappa_A(x) = \bar{s}g(|x - a_1| \cdot |x - a_2| \cdot \dots \cdot |x - a_m|)$  yra aibės  $A$  charakteringoji funkcija.

3. Jei kuri nors aibė  $A$  ir jos papildinys  $\bar{A}$  (iki natūraliųjų skaičių aibės) rekursyviai skaičios aibės, tai  $A$ , kaip ir  $\bar{A}$  yra rekursyvi.

Tarkime,  $A$  sutampa su primitiviai rekursyvios funkcijos  $f(x)$  reikšmių aibe, o  $\bar{A}$  – su  $g(x)$  reikšmių aibe (remiamės 3.15 apibrėžimu). Funkcija

$$h(x) = \mu_z(|f(z) - x| \cdot |g(z) - x| = 0)$$

apibrėžta su bet kuriuo natūraliuoju  $x$ , nes  $A \cup \bar{A} = N$  ir todėl ji yra bendroji rekursyvioji funkcija. Funkcija  $A$  ir  $\bar{A}$  charakteringosios yra šios bendrosios rekursyviosios funkcijos:

$$\kappa_A(x) = \bar{s}g(|f(h(x)) - x|,$$

$$\kappa_{\bar{A}}(x) = \bar{s}g(|g(h(x)) - x|.$$

Iš trečiosios savybės išplaukia teorema.

**3.4 teorema.** *Jei kuri nors rekursyviai skaiti aibė nėra rekursyvi, tai jos papildinys nėra nei rekursyvi, nei rekursyviai skaiti aibė.*

Ši teorema matematinėje logikoje labai svarbi. Formulėms priskiriami numeriai ir aprašytosios sąvokos bei rezultatai taikomi formulių aibėms. Vėliau matysime, kad rekursyviai skaičios, bet nerekursyvios aibės yra *tapačiai teisingų* bei *tapačiai klaidingų predikatų logikos formulių aibės*. Iš 3.4 teoremos išplaukia, kad įvykdomų predikatų logikos formulių aibė nėra nei rekursyvi, nei rekursyviai skaiti. Tas pats galioja ir formulių, kurios nėra tapačiai teisingos, aibe.

### 3.10 Ackermanno funkcijos

**3.17 apibrėžimas.** *Sakysime, kad sąryšiu  $R(x, y)$ , apibrėžtu aibėje  $A$ ,*



nusakome dalinę tvarką joje, jei sąryšis refleksyvus, tranzityvus ir antisimetrinis, t.y. kad ir kokie būtų  $x, y \in A$  ( $R(x, y) \& R(y, x)$ )  $\rightarrow x = y$ .

Sąryšius, kuriais įvedama dalinė tvarka, žymime  $\leq$ . Kai kada, patikslindami, apie kurios aibės tvarką kalbama, žymėsime  $\leq$  su indeksu, pavyzdžiui,  $\leq_A$ . Tvarka, kai bet kurie du elementai palyginami, t.y.  $R(x, y)$  apibrėžtas su bet kuriais  $x, y$  iš nagrinėjamosios aibės, vadinama tiesine.

**3.18 apibrėžimas.** Tarkime aibėse  $A, B$  įvesta tiesinė tvarka. Aibės vadinamos *potvarkomis* (žymime  $A \simeq B$ ), jei jos yra izomorfinės kaip sutvarkytos aibės, t.y. yra dinamos to paties tipo aibėmis.

Taigi, jei  $A \simeq B$ , tai egzistuoja tokia abipusiškai vienareikšmė  $A, B$  elementų atitiktis, kad nesvarbu, kas būtų  $a_1, a_2 \in A$ , jie ir juos atitinkantys  $b_1, b_2 \in B$  tenkina sąlygas:  $a_1 \leq_A a_2$  ir  $b_1 \leq_B b_2$ . Fiksuodami kurią nors netuščią aibę, galime rasti daug jai panašių aibių. Iš visų galimų aibių išskirsime kai kurias, dažniausiai matematikoje bei informatikoje naudojamą skaitines aibes ir suteiksime joms, kartu ir visoms į jas panašioms, vardus. Tie vardai vadinami *tipais* arba *ordinalais*. Pagrindinių aibių tipai:

- 1) tuščiosios – 0,
- 2) baigtinės  $N_n = \{0, 1, \dots, n-1\}$  –  $n$ ,
- 3) natūraliųjų skaičių –  $\omega$ ,
- 4) sveikųjų skaičių –  $\pi$ ,
- 5) racionaliųjų skaičių –  $\eta$ ,
- 6) realiųjų skaičių –  $\lambda$ .

Pakeitę  $\leq$  į  $\geq$ , įvedame jau kitą tvarką, vadinamąją *dualiąja tvarką*. Jei aibės  $A$  tipas  $\alpha$ , tai simboliu  $\alpha^*$  žymimas dualiosios tvarkos tipas. Apibrėšime veiksmus su tipais. Tarkime, aibės  $A$  tipas yra  $\alpha$  (tvarka  $\leq_A$ ), o  $B$  tipas –  $\beta$  (tvarka  $\leq_B$ ).

**3.19 apibrėžimas.** Tipų  $\alpha, \beta$  suma (žymime  $\alpha + \beta$ ) yra tiesinė tvarka  $\leq$  aibėje  $A \cup B$ , nusakyta tokiu būdu:

- a) jei  $x \in A, y \in B$ , tai  $x \leq y$ ,
- b) jei  $x, y \in A$  ir  $x \leq_A y$ , tai  $x \leq y$ ,
- c) jei  $x, y \in B$  ir  $x \leq_B y$ , tai  $x \leq y$ .

**3.20 apibrėžimas.** Tipų  $\alpha, \beta$  sandauga (žymėsime  $\alpha \cdot \beta$ ) yra tiesinė tvarka  $\leq$  aibėje  $A \times B$ , nusakyta tokiu būdu:

a) jei  $y_1 \leq_B y_2$ , tai  $(x_1, y_1) \leq (x_2, y_2)$ .

b) jei  $y_1 = y_2$  ir  $x_1 \leq_A x_2$ , tai  $(x_1, y_1) \leq (x_2, y_2)$ .

Tarkime aibėje  $A = \{x_0, x_1, x_2, \dots\}$  tvarka yra  $x_0 < x_1 < x_2 < \dots$ . o aibėje  $B = \{y_0, y_1, y_2, \dots\}$  –  $y_0 < y_1 < y_2 < \dots$ . Tuomet aibėje  $A \times B$  yra tokia tvarka:

$$(x_0, y_0) < (x_1, y_0) < (x_2, y_0) < \dots < (x_0, y_1) < (x_1, y_1) < (x_2, y_1) < \dots$$

Nesunku matyti, kad a)  $\alpha + 0 = 0 + \alpha = \alpha$ , b)  $1 + \omega = \omega$ , bet  $\omega + 1 \neq \omega$ , c)  $\omega^* \neq \omega$ .  ~~$\alpha \neq \alpha^2$~~  ~~prasta žymėti  $\alpha^2$~~ .  ~~$\alpha \neq \alpha^2$~~  ~~prasta žymėti  $\alpha^2$~~ .  
Apibrėžiame funkcijas  $B_n(a, x)$ , kai  $a \geq 2$ :

$$B_0(a, x) = a + x, B_1(a, x) = a \cdot x, B_2(a, x) = a^x.$$

Tai didėjančios funkcijos.  $B_i(a, x) < B_j(a, x)$ , kai  $i < j$ , pradedant kuriuo nors  $x_0$ . Jos tenkina tokias lygybes:

$$B_1(a, 1) = a \quad B_1(a, x+1) = B_0(a, B_1(a, x))$$

$$B_2(a, 1) = a \quad B_2(a, x+1) = B_1(a, B_2(a, x))$$

Pratęskime jas (kai  $n \geq 2$ ):

$$B_{n+1}(a, 1) = a \quad B_{n+1}(a, x+1) = B_n(a, B_{n+1}(a, x))$$

Tarkime, kad  $B_{n+1}(a, 0) = 1$ , kai  $n \geq 1$ . Ackermanno funkcijos variantu, kai  $a = 2$ , vadiname  $A(n, x) = B_n(2, x)$ . Įvedame tiesinę tvarką tarp porų:

$$(0, 0) < (0, 1) < (0, 2) < \dots < (1, 0) < (1, 1) < (1, 2) < \dots < (n, 0) < (n, 1) < (n, 2) < \dots$$

Jos tipas yra  $\omega^2$ . Funkcija  $A(n, x)$  aprašoma rekursija pagal tipą  $\omega^2$ . Pastebėsime, kad reikšmei  $A(n+1, 0)$  ankstesnė yra  $A(n_1, x)$  su  $n_1 \leq n$  ir bet kuriuo  $x$ . Ackermann funkcija nusakoma tokiomis lygybėmis:

$$\begin{aligned} A(0, x) &= x + 2 \\ A(1, 0) &= 0 \\ A(y, 0) &= 1 \text{ su } y \geq 2 \\ A(y+1, x+1) &= A(y, A(y+1, x)) \text{ visiems } x, y \end{aligned}$$

Funkcija turi tokias savybes:

a)  $A(n, x) \geq 2^x$  ( $n \geq 2; x = 1, 2, \dots$ ),

$$b) A(n+1, x) \geq A(n, x) + 1.$$

$$c) A(n, x+1) > A(n, x) \quad (n, x = 1, 2, \dots),$$

$$d) A(n+1, x) \geq A(n, x+1).$$

Funkcija  $h(x) = A(x, x)$  apibrėžta su bet kuriomis  $x$  reikšmėmis, todėl ji yra bendroji rekursyvioji. Įrodysime, kad  $h(x)$  nėra primityviai rekursyvi. Naudosimes rezultatu, kad vieno argumento primityviai rekursyvių funkcijų aibė gali būti apibrėžta naudojantis tik vieno argumento primityviai rekursyviomis funkcijomis.

Įvedame naujus sudėties bei iteracijos operatorius. Juos taikysime vieno argumento primityviai rekursyvioms funkcijoms. Rezultatas – vieno argumento primityviai rekursyvi funkcija. Pritaikę sudėties operatorių funkcijoms  $f(x), g(x)$ , gauname  $f(x) + g(x)$ . Tarkime  $g(x) \in PR$ . Apibrėžiame naują funkciją  $f(x)$  tokiu būdu:  $f(0) = 0, f(x+1) = g(f(x))$ . Sakysime, kad  $f(x)$  gauta iš  $g(x)$  pritaikius iteracijos operatorių ir žymime  $I(g(x))$ .

**3.5 teorema.** *Vieno argumento primityviai rekursyvių funkcijų aibė sutampa su aibe, kuriai priklauso bazinės funkcijos  $s(x), q(x) = x \cdot \lfloor \sqrt{x} \rfloor^2$  ir kuri uždara sudėties, kompozicijos bei iteracijos atžvilgiu.*

Sakome, kad  $f(x)$  mažoruoja funkcija  $h(x)$ , jei  $f(x) < h(x)$  pradedant kuriuo nors  $x_0$ , t.y., kai  $x \geq x_0$ . Parodysime, kad kiekviena vieno argumento primityviai rekursyvi funkcija mažoruoja funkcija  $h(x)$  ir todėl  $h(x)$  nėra primityviai rekursyvi. Visų pirma įrodysime, kad ir kokia būtų vieno argumento  $f(x) \in PR$ , galima rasti tokį  $n$ , kad  $f(x)$  būtų mažoruoja funkcija  $A(n, x)$ . Remiamės 3.5 teorema, t.y. tariame, kad nagrinėjamosios funkcijos gautos iš  $s(x), q(x)$  pritaikius sudėties, kompozicijos bei iteracijos operatorius.

$$s(x) < 2^x = A(2, x) \quad (x = 2, 3, \dots)$$

$$q(x) < s(x) < 2^x = A(2, x)$$

Tarkime,  $f(x) < A(n_1, x), g(x) < A(n_2, x)$  ir  $n = n_1 + n_2$ . Tuomet  $f(x) < A(n, x)$  ir  $g(x) < A(n, x)$ .

$$f(x) + g(x) < 2 \cdot A(n, x) < 2 \cdot 2^{A(n, x)} \leq 2^{A(n+1, x)} \leq A(n, A(n+1, x)) = A(n+1, x+1) \leq A(n+2, x).$$

$$f(g(n)) < A(n, g(x)) < A(n, A(n+1, x)) = A(n+1, x+1) \leq A(n+2, x).$$

Panašiai gaunamas įvertis ir iteracijos atveju. Jei  $f(x) < A(n, x)$ , tai

$$f(n+x) < A(n, n+x) < A(n+x, n+x) = h(n+x).$$

Taigi, gavome, kad ir kokia būtų vieno argumento  $f(x) \in PR$  ji mažoruoja visur apibrėžta funkcija  $h(x)$  ir todėl  $h(x)$  nėra primitiviai rekursyvi. Tuo pačiu įrodėme, kad aibė  $PR$  yra griežtas aibės  $BR$  poaibis.

### 3.11 Universaliosios funkcijos

**3.21 apibrėžimas.** Tarkime,  $A$  kuri nors  $n$  argumentų funkcijų aibė. Funkcija  $F(x_0, x_1, \dots, x_n)$  vadinama aibės  $A$  universaliaja, jei  $A = \{F(0, x_1, \dots, x_n), F(1, x_1, \dots, x_n), \dots\}$ , t.y.  $F(i, x_1, \dots, x_n) \in A$  ( $i = 0, 1, 2, \dots$ ) ir nesvarbu kokia būtų  $f(x_1, \dots, x_n) \in A$ , atsirastų bent vienas toks natūralusis  $i$ , kad  $f(x_1, \dots, x_n) = F(i, x_1, \dots, x_n)$ .

Tarkime, kad  $A$  kuri nors visur apibrėžta  $n$  argumentų funkcijų aibė, o  $F(x_0, x_1, \dots, x_n)$  – jos universalioji. Pastebėsime, jei  $g(x_1, \dots, x_n) = F(x_1, x_1, x_2, \dots, x_n) + 1$  priklauso aibei  $A$ , tai universaliajai  $F$  atsirastų toks  $i$ , su kuriuo galioja lygybės:

$$\begin{aligned} F(i, x_1, x_2, \dots, x_n) &= F(x_1, x_1, x_2, \dots, x_n) + 1 \\ F(i, i, x_2, \dots, x_n) &= F(i, i, x_2, \dots, x_n) + 1 \end{aligned}$$

Matome: jei  $F \in PR$ , tai ir  $g \in PR$ ; jei  $F \in BR$ , tai ir  $g \in BR$ . Iš čia išplaukia du teiginiai:

a) visų  $n$  argumentų primitiviai rekursyvių funkcijų aibės universaliosios negali būti primitiviai rekursyvi funkcija,

b) visų  $n$  argumentų bendrųjų rekursyviųjų funkcijų aibės universaliosios negali būti bendroji rekursyvioji funkcija.

**3.6 teorema.** Visų vieno argumento primitiviai rekursyvių funkcijų aibei egzistuoja universalioji bendroji rekursyvioji funkcija.

*Įrodymas.* Remiantis 3.5 teorema visas vieno argumento primitiviai rekursyvias funkcijas galima gauti iš bazinių  $s(x), q(x)$  taikant sudėties, kompozicijos bei iteracijos operacijas. Funkcijoms priskirsime natūraliuosius skaičius, t.y. apibrėžiame funkcijų numeraciją. Funkcijos  $f(x)$  numerį žymėsime  $n(f(x))$  arba rašysime  $f_n(x)$ , kai jos numeris yra  $n$ . Funkcijoms  $s(x), q(x)$  priskiriame tokius numerius:  $n(s(x)) = 1, n(q(x)) = 3$ .

Tarkime  $n(f(x)) = a$ , o  $n(g(x)) = b$ . Tuomet, funkcijoms, gautoms pritaikius sudėties, kompozicijos ar iteracijos operatorius, priskiriame tok-

ius numerius:

$$\begin{aligned}n(f(x) + g(x)) &= 2 \cdot 3^a \cdot 5^b, \\n(f(g(x))) &= 4 \cdot 3^a \cdot 5^b, \\n(I(f(x))) &= 8 \cdot 3^a.\end{aligned}$$

$$\text{Pavyzdžiui, } n(I(2 \cdot s)) = n(I(s+s)) = 8 \cdot 3^{2 \cdot 3 \cdot 5}, \quad n(s+I(q)) = 2 \cdot 3 \cdot 5^{8 \cdot 3^3}.$$

Apibrėžiame dviejų argumentų funkciją  $F(n, x) = f_n(x)$ , t.y.  $F(n, x)$  lygi vieno argumento funkcijai, kurios numeris yra  $n$ .

$$F(n, x) = \begin{cases} f_a(x) + f_b(x), & \text{jei } n = 2 \cdot 3^a \cdot 5^b \\ f_a(f_b(x)), & \text{jei } n = 4 \cdot 3^a \cdot 5^b \\ f_a(f_n(x-1)), & \text{jei } n = 8 \cdot 3^a, \quad x > 0 \\ 0, & \text{jei } n = 8 \cdot 3^a, \quad x = 0 \\ q(x), & \text{jei } n = 3 \\ s(x), & \text{jei } n = 1 \end{cases}$$

Iš apibrėžimo matome, kad kiekviena funkcija turi numerį, bet ne vienintelį. Pavyzdžiui, nors  $f(x) + g(x) = g(x) + f(x)$ , bet jų numeriai bendruoju atveju, skirtingi. Ne kiekvieną natūralųjį skaičių atitinka kuri nors funkcija. Pavyzdžiui, nėra tokios funkcijos, kurios numeris lygus 7.13 ar 17. Dabar galime apibrėžti vieno argumento primityviai rekursyvių funkcijų universaliją

$$D(n, x) = \begin{cases} F(n, x), & \text{jei } n \text{ yra kurios nors funkcijos numeris} \\ 0, & \text{priešingu atveju} \end{cases}$$

$D(n, x)$  – bendroji rekursyvioji funkcija. Teorema įrodyta.

**3.7 teorema.**  $D(x_0, \alpha_n(x_1, \dots, x_n))$  yra visų  $n$  argumentu primityviai rekursyvių funkcijų aibės universalioji funkcija.

*Įrodymas.* Universaliją pažymėkime  $D^{n+1}(x_0, x_1, \dots, x_n)$ . Parodysime, kad ji lygi  $D(x_0, \alpha_n(x_1, \dots, x_n))$ . Viena vertus su kiekvienu fiksuotu  $x_0$  funkcija

$D(x_0, \alpha_n(x_1, \dots, x_n))$  primityviai rekursyvi. Antra vertus, jei  $g(x_1, \dots, x_n)$  yra kuri nors  $n$  argumentų primityviai rekursyvi funkcija, tai tokia yra ir  $f(x) = g(\pi_n^1(x), \dots, \pi_n^n(x))$ . Ji vieno argumento. Todėl atsiras toks natūralusis  $x_0$ , kad  $f(x) = D(x_0, x)$ . Skaičius  $x_0$  ir yra  $g(x_1, \dots, x_n)$  numeris, nes

$$f(\alpha_n(x_1, \dots, x_n)) = g(\pi_n^1(\alpha_n(x_1, \dots, x_n)), \dots, \pi_n^n(\alpha_n(x_1, \dots, x_n))) = g(x_1, \dots, x_n).$$

Teorema įrodyta.

Dabar aprašysime dalinių rekursyviųjų funkcijų universaliasias. Jos taip pat yra dalinės funkcijos.

**3.22 apibrėžimas.** *Dalinės rekursyvosios funkcijos  $f(x_1, \dots, x_n)$  grafiku vadiname aibę  $A = \{(x_1, \dots, x_n, y) : f(x_1, \dots, x_n) = y\}$ . Niekur neapibrėžtos funkcijos grafikas yra tuščia aibė.*

Pavyzdžiui, funkcijos  $y = x^2$  grafikas yra aibė  $\{(0, 0), (1, 1), (2, 4), (3, 9), (4, 16), \dots\}$ .

**3.8 teorema.** *Dalinių rekursyviųjų  $n$  argumentų funkcijų aibei egzistuoja universalioji funkcija.*

*Irodymas.* Bet kurios dalinės rekursyvosios funkcijos grafikas yra rekursyviai skaiti aibė, nes sutampa su  $\bar{s}g|f(x_1, \dots, x_n) - y| - 1$  apibrėžimo sritimi. Taigi, kad ir kokia būtų dalinė rekursyvi funkcija  $f(x_1, \dots, x_n)$ , remiantis 3.16 apibrėžimu galime tvirtinti, kad egzistuoja tokia primityviai rekursyvi funkcija  $g(x_1, \dots, x_n, y, z)$ , kad

$(x_1, \dots, x_n, y) \in A$  tada ir tiksliai tada, kai egzistuoja toks  $z$ , kad  $g(x_1, \dots, x_n, y, z) = 0$ .

Tarkime  $t = \alpha_2(y, z)$ . Tuomet  $(x_1, \dots, x_n, y) \in A$  tada ir tiksliai tada, kai yra toks  $t$ , kad  $g(x_1, \dots, x_n, \pi_2^1(t), \pi_2^2(t)) = 0$ . Pažymėkime  $g(x_1, \dots, x_n, \pi_2^1(t), \pi_2^2(t))$  nauja funkcija  $F(x_1, \dots, x_n, t)$ . Kad ir kokia būtų dalinė rekursyvioji funkcija  $f(x_1, \dots, x_n)$ , atsirastų tokia primityviai rekursyvi  $F(x_1, \dots, x_n, t)$ , kad

$$f(x_1, \dots, x_n) = \pi_2^1(\mu_t(F(x_1, \dots, x_n, t) = 0)) \quad (3.2)$$

Dalinių rekursyviųjų  $n$  argumentų funkcijų universalioji  $\tilde{D}^{n+1}$  gaunama tokiu būdu:

$$\tilde{D}^{n+1}(x_0, x_1, \dots, x_n) = \pi_2^1(\mu_t(D^{n+2}(x_0, x_1, \dots, x_n, t) = 0))$$

Iš tikrųjų ši funkcija su kiekvienu fiksuotu  $x_0$   $\tilde{D}^{n+1}$  yra dalinė rekursyvioji. Tačiau, jei  $f(x_1, \dots, x_n)$  yra kuri nors dalinė rekursyvioji funkcija, tai egzistuoja tokia primityviai rekursyvi  $F(x_1, \dots, x_n, t)$ , kuriai galioja lygybė (3.2). Tarkime jos numeris  $i$ . Tuomet

$$\tilde{D}^{n+1}(i, x_1, \dots, x_n) = \pi_2^1(\mu_t(D^{n+2}(i, x_1, \dots, x_n, t) = 0)).$$

Teorema įrodyta.

**3.23 apibrėžimas.** *Sakome, kad visur apibrėžta funkcija  $g(x_1, \dots, x_s)$  yra dalinės funkcijos  $f(x_1, \dots, x_s)$  pratęsimas, jei bet kuriems  $x_1^0, \dots, x_s^0$ , su kuriais  $f$  apibrėžta, galioja lygybė  $g(x_1^0, \dots, x_s^0) = f(x_1^0, \dots, x_s^0)$ .*

Ar galima pratęsti kiekvieną dalinę rekursyviąją funkciją pratęsti. t.y. ar atsiras iš bendrųjų rekursyviųjų funkcijų tokia, kuri bus jos pratęsimas? Pasirodo, kad ne visas dalines rekursyviasias funkcijas galima pratęsti.

**3.9 teorema.** *Dalinių rekursyviųjų  $s$  argumentų funkcijų universalioji  $\tilde{D}^{s+1}(x_0, x_1, \dots, x_s)$  neturi pratęsimo.*

*Įrodymas.* Nagrinėjame  $V(x) = \bar{s}g\tilde{D}^{s+1}(x, x, \dots, x)$ . Jei  $V(x)$  apibrėžta su kuriuo nors  $x_0$ , tai jos reikšmė lygi 1 arba 0. Tarkime  $V(x)$  turi pratęsimą  $W(x)$ . Į ją (vieno argumento) galima žiūrėti kaip į  $s$  argumentų funkciją

$$W(x_1) = pr_s^1(W(x_1), x_2, \dots, x_s).$$

Atsiras toks  $a$ , kad  $\tilde{D}^{s+1}(a, x_1, \dots, x_s) = W(x_1)$ . Ji visur apibrėžta. Imame  $x_1 = \dots = x_s = a$ .  $W(x)$  yra ir  $\bar{s}g\tilde{D}^{s+1}(x, x, \dots, x)$  pratęsimas. Gauname prieštarą  $W(a) = \bar{s}gW(a)$ . Taigi  $V(x)$  neturi pratęsimo. Tarkime, kad  $\tilde{D}^{s+1}(x_0, x_1, \dots, x_s)$  turi pratęsimą  $P(x_0, x_1, \dots, x_s)$ . Tuomet  $\bar{s}gP(x, x, \dots, x)$  būtų  $V(x)$  pratęsimas, o tokios tarp bendrųjų rekursyviųjų funkcijų nėra. Teorema įrodyta.

**3.10 teorema.** *Egzistuoja rekursyviai skaičiai, bet nerekursyviosios aibės.*

*Įrodymas.* Nagrinėjame universaliąją vieno argumento funkcijoms  $\tilde{D}^2(x_1, x_2)$ . Funkcija  $V(x) = \bar{s}g\tilde{D}^2(x, x)$  turi savybes:

- 1)  $V(x)$  dalinė rekursyvi,
- 2)  $V(x)$  neturi pratęsimo,
- 3)  $V(x)$  reikšmių aibė  $\{0, 1\}$ .

Lygties  $V(x) = 0$  sprendinių aibė rekursyviai skaiti, nes sutampa su dalinės rekursyvios funkcijos  $\mu_z(V(x) + z = 0)$  apibrėžimo sritimi. Jei ji būtų rekursyvi, t.y. atsirastų tokia bendroji rekursyvioji  $\kappa(x)$ , kad

$$\kappa(x) = \begin{cases} 1, & \text{jei } V(x) = 0 \\ 0, & \text{priešingu atveju,} \end{cases}$$

tai  $\bar{s}g\kappa(x)$  būtų  $V(x)$  pratęsimas. O tai prieštarauja antrai funkcijos  $V(x)$  savybei. Teorema įrodyta.

### 3.12 Kanoninis Posto skaičiavimas

Šiame skyrelyje trumpai susipažinsime su amerikiečių logiko E.L.Posto 1943 m. aprašytu *algoritmiškai apskaičiuojamųjų funkcijų* formalizmu – *kanoniniu skaičiavimu*.

**3.24 apibrėžimas.** *Kanoniniu skaičiavimu vadinsime ketvertą  $(A, P, Ak, T)$ ; čia  $A, Ak, P, T$  – baigtinės aibės.  $A$  vadinama skaičiavimo abėcėle.  $P$  yra skaičiavimo kintamųjų aibė ( $A \cap P = \emptyset$ )  $Ak$  – abėcėlės  $A$  žodžių aibė, vadinama skaičiavimo aksiomų aibe,  $T$  – taisyklių aibė pavidalo*

$$\frac{G_{1,1}p_{1,1}G_{1,2}p_{1,2}\dots G_{1,n_1}p_{1,n_1}G_{1,n_1+1}}{G_{2,1}p_{2,1}G_{2,2}p_{2,2}\dots G_{2,n_2}p_{2,n_2}G_{2,n_2+1}} \dots \frac{G_{m,1}p_{m,1}G_{m,2}p_{m,2}\dots G_{m,n_m}p_{m,n_m}G_{m,n_m+1}}{G_1p_1G_2p_2\dots G_np_nG_{n+1}}$$

$G_{i,j}$  – abėcėlės  $A$  žodžiai,  $p_{i,j}$  – kintamieji.

Žodžiai virš brūkšnio vadinami taisyklės prielaidomis, o brūkšnio apačioje – išvada. Taria, kad kintamieji, įeinantys į išvadą, sutinkami bent vienoje prielaidoje.

Taisyklę *realizuojančiu rinkiniu* vadiname reiškinių pavidalo

$$\left( \begin{array}{cccc} p^1 & p^2 & \dots & p^s \\ B_1 & B_2 & \dots & B_s \end{array} \right);$$

čia  $p^1, \dots, p^s$  – pilnas sąrašas kintamųjų, įeinančių į taisyklę, o  $B_1, \dots, B_s$  – kurie nors abėcėlės  $A$  žodžiai. Pakeitę taisyklėje visas įeitis  $p^i$  žodžiais  $A_i$  ( $i = 1, \dots, s$ ), gauname taisyklės taikymą

$$\frac{Q_1}{Q}$$

čia  $Q, Q_i$  ( $i = 1, \dots, m$ ) – abėcėlės  $A$  žodžiai. Žodžių abėcėlėje  $A$  seka vadinama išvedimu, jei kiekvienas jos narys yra aksioma arba gautas iš kairėje esančių formulių pritaikius kurią nors skaičiavimo taisyklę. Sakoma, kad žodis  $B$  išvedamas skaičiavime, jei galima rasti išvedimą, kuris baigiasi žodžiu  $B$ .

Kanoninis skaičiavimas įdomus tuo, kad savyje apima tiek Turingo mašinas, tiek ir loginius skaičiavimus. Į Turingo mašinas ir loginius skaičiavimus galime žiūrėti kaip į atskirus kanoninių skaičiavimų atvejus. Gauname ir



kitokį bendresnį formalų aparatą rekursyviai skaičioms aibėms aprašyti. Generuojami objektai nebūtinai yra skaičiai.

**Pavyzdžiai.**

$$1. A = \{|\}, P = \{p\}, Ak = \{||\}, T = \frac{p}{pp}.$$

Išvedamų skaičiavime žodžių aibė lygi  $\{||, |||, \dots, |^{2^n}\}$ .

2.  $A = \{1, 0, *\}, P = \{p, q\}, Ak = \{B\}$ ; čia  $B$  – kuris nors abėcėlės  $\{1, 0\}$  žodis.  $T$  susideda iš taisyklių:

$$\frac{p}{p*} \quad \frac{p1 * q}{p * 0q} \quad \frac{p0 * q}{p * 1q} \quad \frac{*p}{p}$$

Kai kada domina ne visi išvedami abėcėlė  $A$  žodžiai, o išvedami žodžiai abėcėlės  $A'$ , t.y. kurio nors aibės  $A$  poaibio. Tuo atveju sakoma, kad  $A'$  yra pagrindinė skaičiavimo abėcėlė. Jei antrajame pavyzdyje pagrindinė abėcėlė laikysime  $\{1, 0\}$ , tai skaičiavime išvedamas tik vienas (neskaitant aksiomos) žodis, kuris gaunamas iš  $B$ , pakeitus jame visas nuliukų įėjis vienetukais bei vienetukų įėjis nuliukais.

**3.25 apibrėžimas.** Sakome, kad du skaičiavimai yra ekvivalentūs atžvilgiu pagrindinės abėcėlės  $A$ , jei išvedamų abiejuose skaičiavimuose abėcėlės  $A$  žodžių aibės sutampa.

**3.26 apibrėžimas.** Taisyklę, kurioje bent vieno kintamojo, įeinančio į prielaidą, išvadoje nėra, vadinsime  $c$ -taisykle.

**3.4 lema.** Koks bebūtų kanoninis skaičiavimas  $\Pi = (A, P, Ak, T)$ , galima rasti jam ekvivalentų atžvilgiu pagrindinės abėcėlės  $A$ , kuriame nėra  $c$ -taisyklių.

Įrodymas. Tarkime  $A = \{a_1, a_2, \dots, a_n\}$  ir skaičiavimo  $\Pi$  taisyklėje  $G_1, \dots, G_m / G$  kintamųjų  $p_1, \dots, p_s$ , įeinančių į prielaidas, išvadoje  $G$  nėra. Naujasis skaičiavimas, kuriame eliminuota nagrinėjamojo  $c$ -taisyklė, ir kuris ekvivalentus skaičiavimui  $\Pi$  atžvilgiu pagrindinės abėcėlės  $A$ , gaunamas tokiu būdu. Abėcėlė papildoma nauju simboliu, pavyzdžiui  $*$ . O taisyklė keičiama tokiomis:

$$\frac{\begin{matrix} G_1 \\ \vdots \\ G_m \end{matrix}}{p_1 \dots p_s * G} \quad \frac{pa_i * q}{p * q} \quad (i = 1, \dots, n) \quad \frac{*q}{q}.$$

Taigi, generuodami tam tikrus "tarpinius" žodžius, kuriuose yra  $*$ , gauname skaičiavimą, kuriame eliminuota  $c$ -taisyklė ir jis ekvivalentus skaičiavimui  $\Pi$  atžvilgiu pagrindinės abėcėlės  $A$ . Lema įrodyta.

**3.27 apibrėžimas.** *Kanoninis skaičiavimas  $\Pi = (A, P, Ak, T)$  vadinamas normaliuoju, jei aibėje  $Ak$  tėra vienas elementas, o visos taisyklės yra pavidalo*

$$\frac{Gq}{qG'};$$

čia  $G, G'$  – abėcėlės  $A$  žodžiai.

Parodysime, kaip galima modeliuoti Turingo mašinos darbą esant fiksuotiems pradiniam duomenims. Tuo tikslu nagrinėsime determinuotą standartinę Turingo mašiną su vienkuse viena juosta. Tarkime, Turingo mašinos abėcėlė  $A \cup \{b\}$ ,  $A = \{a_1, \dots, a_m\}$ . Būsenų aibė  $Q = \{q_0, q_1, \dots, q_s\}$  čia  $q_0$  – pradinė būsena. Pradiniai duomenys  $e_1 e_2 \dots e_v$  – yra abėcėlės  $A$  žodžiai. Jie užrašomi pirmose (iš kairės į dešinę)  $v$  ląstelėse. Po baigtinio skaičiaus žingsnių Turingo mašina pereina į galutinę būseną (pažymėsime ją  $q_s$ ) arba dirba be galo ilgai. Jei ji baigia darbą, tai pereina į galutinę būseną  $q_s$  ir skaitymo galvutė yra ties pirmąja ląstele. Tuščios ląstelės gali būti tik galutinio rezultato dešinėje, t.y. tuščios ląstelės prirašomos tik iš dešinės. Skaičiavimo eigoje jų negali būti tarp abėcėlės  $A$  žodžių. Nors iš pirmo žvilgsnio ir atrodo, kad Turingo mašina turi tenkinti daug apribojimų, bet yra žinoma, kad ir kokia būtų Turingo mašina, galima rasti jai ekvivalenčią, tenkinančią išvardytus apribojimus.

Pastarosios darbą modeliuojantis normalusis kanoninis skaičiavimas gaunamas tokiu būdu. Abėcėlė  $B = \{a_1, \dots, a_m, q_0, q_1, \dots, q_s, b, *\}$ , pagrindinė abėcėlė  $A \subset B$ .  $P = \{p\}$ ,  $Ak = \{*q_0 e_1 e_2 \dots e_v\}$ . Taisyklės:

$$\frac{xp}{px} \quad (x \in B) \quad \frac{*q_s p}{p**} \quad \frac{b**p}{p**} \quad \frac{x**p}{p} \quad (x \in A)$$

Kiekvieną mašinos komandą atitiks po taisyklę. Komandoms pavidalo  $\delta(q_i, a_j) = (q_u, a_v, D)$ ,  $\delta(q_i, b) = (q_u, y, D)$  ( $y \in A \cup \{b\}$ ),  $\delta(q_i, a_j) = (q_u, a_v, K)$ ,  $\delta(q_i, y) = (q_u, z, N)$  ( $y, z \in A \cup \{b\}$ ) priskiriame taisykles:

$$\frac{q_i a_j p}{p a_v q_u} \quad \frac{q_i * p}{p y q_u *} \quad \frac{x q_i a_j p}{p q_j x a_v} \quad (x \in A) \quad \frac{q_i y p}{p q_u z}$$

Kita teorema priklauso logikui E.L.Postui.

**3.11 teorema.** *Kad ir koks būtų kanoninis skaičiavimas su pagrindine abėcėle  $A$ , galima rasti jam ekvivalentų normalųjį atžvilgiu  $A$ .*

### 3.13 Lambda skaičiavimas

Nagrinėsime dar vieną algoritmiškai apskaičiuojamųjų funkcijų formalizmą –  $\lambda$ -skaičiavimą. Jį 1930 metais aprašė amerikiečių logikas A.Church. Programavimo kalbos LISP pagrindimas remiasi  $\lambda$ -skaičiavimo savybėmis.

Tą patį užrašą  $x + y$  galim laikyti: a) skaičiumi  $x + y$ , b) vieno argumento funkcija  $f(x) = x + y$ , c) vieno argumento funkcija  $g(y) = x + y$ , d) dviejų argumentų funkcija  $h(x, y) = x + y$ .  $\lambda$ -skaičiavime tuos atvejus galima atskirti sintaksiškai. Jei  $E$  žymi skaičių (t.y. atvejas  $a$ ), tai vieno argumento funkcija  $f(x)$  bus žymima  $\lambda x.E$ . vieno argumento funkcija  $g(y) = \lambda y.E$ , o dviejų argumentų funkcija  $\lambda x.\lambda y.E$ .

$\lambda$ -skaičiavimo reiškinius bus kai kurios baigtinės aibės  $\{\lambda, (.), ., a, b, c, \dots, x, y, z, a', b', c', \dots, z', a'', b'', \dots\}$  elementų sekos.  $a, b, c, \dots, a', b', c', \dots, a'', b'', c'', \dots$  vadinami kintamaisiais.

#### 3.28 apibrėžimas. ( $\lambda$ -skaičiavimo termo).

1. *Kintamasis yra terminas.*
2. *Jei  $E_1, E_2$  – terminai, tai  $(E_1 E_2)$  irgi terminas (aplikacija).*
3. *Jei  $x$  – kintamasis,  $E$  – terminas, tai  $\lambda x.E$  irgi terminas (abstrakcija).*

Taigi,  $\lambda$ -skaičiavimo terminai konstruojami naudojantis tik dvejomis operacijomis – aplikacija bei abstrakcija.

Termų pavyzdžiai:

$$\begin{aligned} & ((uv)z)t, \\ & x(\lambda y.((x\lambda x.z))y), \\ & \lambda x.\lambda y.\lambda z.(uv). \end{aligned}$$

Išorinius skliaustus dažniausiai praleisime.

Kintamųjų įėtis (išskyrus tas, kurios yra tiesiogiai simbolio  $\lambda$  dešinėje pusėje) skirstysime į laisvasias ir suvaržytąsias.

#### 3.29 apibrėžimas.

- *Jei terminas  $E = x$ , tai kintamojo  $x$  įėtis terme  $E$  yra laisva,*
- *Jei  $E = (E_1 E_2)$ , tai visos laisvosios  $x$  įėitys termuose  $E_1, E_2$  yra laisvosiomis ir terme  $E$ ,*
- *Jei  $E = \lambda y.E'$  ir  $x \neq y$ , tai visos laisvosios  $x$  įėitys terme  $E'$  yra laisvosiomis ir terme  $E$ . o jei  $x = y$ , tai terme  $E$  nėra laisvųjų  $x$  įeičių.*

Tarkime terme  $E$  sutinkamas terminas pavidalo  $\lambda x.E'$ . Tuomet nagrinėjamosios  $\lambda x$  įėities veikimo sritimi vadinamas terminas  $E'$ . Kintamąjį  $x$  terme  $E$  vadinsime laisvu, jei yra bent viena  $x$  laisva įėtis terme  $E$ . Terminas vadinamas uždaru, jei jame nėra laisvų kintamųjų. Jei kintamojo  $x$  įėtis

terme  $E$  nėra laisva, tai ji vadinama suvaržyta.

#### Pavyzdžiai.

1. Terme  $x((\lambda x.y)x)$  pirmoji bei trečioji  $x$  įeitys, bei  $y$  įeitis yra laisvosios.
2. Termas  $\lambda x.\lambda y((xy)y)$  yra uždaras,
3. Terme  $(\lambda x.(xy))(\lambda y.(xy))$  antroji  $x$  įeitis suvaržytoji, o trečioji laisva: pirmoji  $y$  įeitis laisva, o trečioji suvaržyta.

Du termai vadinami  $\alpha$ -ekvivalentūs:

- a) jei  $E_1$  gautas iš  $E_2$  pervardijus visas kurio nors kintamojo laisvasias įeitis nauju, neįeinančiu į  $E_2$  kintamuoju,
- b) jei  $E_1$  gautas iš  $E_2$  pakeitus kurio nors termo atrodančio šitaip  $\lambda x.E'_2$  įeiti termu  $\lambda z.E''_2$ .  $E''_2$  gautas iš  $E'_2$ , pervardijus visas laisvasias  $x$  įeitis termu  $E'_2$  kintamuoju  $z$ . Čia  $z$  – naujas, neįeinantis į  $E_2$  kintamasis.

Jei  $E_1$   $\alpha$ -ekvivalentus  $E_2$ , o  $E_2$   $\alpha$ -ekvivalentus  $E_3$ , tai ir  $E_1$   $\alpha$ -ekvivalentus  $E_3$ .  $\lambda$ -skaičiavime nagrinėjami termai su tikslumu iki  $\alpha$ -ekvivalentumo. Laikysime, kad termuose bet kuris kintamasis yra arba laisvasis, arba suvaržytasis.

**Pavyzdys.** Termai  $(\lambda x.x)(\lambda x.x)$  ir  $(\lambda x.x)(\lambda y.y)$  bei  $((xy)(\lambda x.(xx)))(\lambda y.y)$  ir  $((xv)(\lambda z.(zz)))(\lambda y.y)$  yra  $\alpha$ -ekvivalentūs.

Terme  $E$  pakeitę kintamąjį  $x$  termu  $X$ , gauname naują termą  $E'$ . Žymėsime  $E' = E[X/x]$ .

**3.30 apibrėžimas.** Termas atrodantis šitaip  $(\lambda x.E)Y$  vadinamas **redekso**, o  $E[Y/x]$  jo santrauka.

Termo  $\beta$ -redukcija vyksta tokiu būdu: ieškomas pirmas iš kairės redexas ir jis pakeičiamas jo santrauka. Tai vadinama redukcijos žingsniu ir žymima simboliu  $\triangleright$ . Peržiūrime gautąjį termą vėl iš kairės į dešinę ir, jei randame redexą, keičiame jį jo santrauka. Termas vadinamas **normaliniu**, jei neįmanoma jo daugiau redukuoti, t.y. jame nebėra redexų. Normalinė forma – tai termas, į kurį redukovome pradinį.

#### Pavyzdžiai.

1.  $((\lambda x.\lambda z.(zx))u)v \triangleright (\lambda z.(zu))v \triangleright vu$ ,
2.  $(\lambda n.\lambda f.\lambda x.((nf)(fx)))(\lambda u.\lambda v.v) \triangleright \lambda f.\lambda x.(((\lambda u.\lambda v.v)f)(fx)) \triangleright \lambda f.\lambda x.((\lambda v.v)(fx)) \triangleright \lambda f.\lambda x.(fx)$ .

Termas vadinamas nenormalizuojamu, jei jo neįmanoma redukuoti į normalinį termą. Pavyzdžiui, termas  $(\lambda x.(xx))(\lambda x.(xx))$  yra nenormalizuojamas:

$$(\lambda x.(xx))(\lambda x.(xx)) \triangleright (\lambda x.(xx))(\lambda x.(xx)) \triangleright \dots$$

Loginės konstantos nusakomos termiais:

$$0 = \lambda x. \lambda y. y$$

$$1 = \lambda x. \lambda y. x$$

Dar du  $\beta$ -redukcijos pavyzdžiai:

$$(0u)v = ((\lambda x. \lambda y. y)u)v \triangleright (\lambda y. y)v \triangleright v$$

$$(1u)v = ((\lambda x. \lambda y. x)u)v \triangleright (\lambda y. u)v \triangleright u$$

$E_1^k E_2$  žymėsime termą  $E_1(\dots(E_1(E_1 E_2))\dots)$ .  $E_1$  kartojamas  $k$  kartų. Jei  $k = 0$ , tai  $E_1^k E_2 = E_2$ .

**3.31 apibrėžimas.** Termas  $\underline{k} = \lambda f. \lambda x. (f^k x)$  vadinamas  $\lambda$ -skaičiavimo natūraliuoju skaičiumi (Church skaičiumi).

**3.32 apibrėžimas.**  $\lambda$ -skaičiavimo termas  $E$  definiuoja dalinę funkciją  $f(x_1, \dots, x_n)$ , jei kokie bebūtų natūralieji  $k_1, \dots, k_n, k$ , iš to, kad  $f(k_1, \dots, k_n) = k$  išplaukia, kad  $(\dots((E \underline{k}_1) \underline{k}_2) \dots) \underline{k}_n$  redukuojamas į normalinį termą  $\underline{k}$ . o jei  $f(k_1, \dots, k_n)$  neapibrėžta, tai  $(\dots((E \underline{k}_1) \underline{k}_2) \dots) \underline{k}_n$  nenormalizuojamas.

Definuojamų funkcijų pavyzdžiai.

1. **Konstanta** 0 definiuojama termu  $\lambda x. \underline{0}$ .

$$(\lambda x. (\lambda f. \lambda y. y)) \underline{k} \triangleright \lambda f. \lambda y. y = \underline{0}.$$

2. **Projekcijos funkcijos**  $pr_p^i(x_1, \dots, x_p) = x_i$  ( $1 \leq i \leq p$ ) definiuojamos termiais  $\lambda x_1. \lambda x_2. \dots \lambda x_p. x_i$ .

$$(\dots((\lambda x_1. \lambda x_2. \dots \lambda x_p. x_i) \underline{k}_1) \underline{k}_2) \dots) \underline{k}_n \triangleright (\dots((\lambda x_2. \dots \lambda x_p. x_i) \underline{k}_2) \dots) \underline{k}_n \triangleright \dots \triangleright \underline{k}_i.$$

3. **Paskesniojo nario funkcija**  $s(x)$  definiuojama termu  $\lambda n. \lambda f. \lambda x. ((nf)(fx))$ .

Irodyta, kad kiekvieną dalinę rekursyviąją funkciją galima definiuoti  $\lambda$ -skaičiavimo termu.

Kai kurios paprastos funkcijos definiuojamos ganėtinai sudėtingais termiais. Pavyzdžiui, pirmesnio nario funkcija

$$prm(n) = \begin{cases} 0, & \text{jei } n = 0 \\ k, & \text{jei } n = k + 1 \end{cases}$$

definiuojama termu

$$\lambda n. \lambda h. \lambda a. ((n(\lambda c. \lambda f. (((f(hc)1))) (c1)) (\lambda g. ((ga)a))) \underline{0})$$

Loginės operacijos  $\neg$ ,  $\&$ ,  $\vee$  definiuojamos termiais:

$$\neg = \lambda x.((x)1).$$

$$\& = \lambda x.\lambda y.((xy)x).$$

$$\vee = \lambda x.\lambda y.((xx)y).$$

1940 m. A.Church aprašė pirmąją tipizuoto  $\lambda$ -skaičiavimo versiją. Kai kuriems  $\lambda$ -skaičiavimo terminams priskirsime tipus ir nagrinėsime tik  $\lambda$ -skaičiavimo terminus su tipais. Tipizuotame skaičiavime definiuojamos tik visur apibrėžtosios funkcijos.

Programavime, kaip ir atitinkamuose skaičiavimų modeliuose, korektiškos programos tenkina tam tikras sąlygas. Tame tarpe, programa turi baigti darbą, kai pradiniais yra reikalaujamo tipo duomenys. Ta prasme tipizuotas skaičiavimas yra žingsnis į priekį kuriant modelius, kuriuose tegalima parašyti korektiškas programas.

Tipų abėcėlė susideda iš bazinių tipų (žymėkime juos  $A, B, C, \dots$ ) ir konstruktoriaus  $\rightarrow$ .

### 3.33 apibrėžimas (tipo).

- Bazinis tipas yra tipas,
- Jei  $\gamma, \delta$  yra tipai, tai  $(\gamma \rightarrow \delta)$  yra irgi tipas.

Kai termui  $E$  priskirtas tipas  $\delta$ , žymime  $E : \delta$ . Visų pirma priskiriami baziniai tipai kintamiesiems. Termų, kuriems jau priskirti tipai, aibę vadinsime kontekstu (žymime  $\Gamma$ ). Jei iš konteksto  $\Gamma$  išplaukia, kad terminas  $E$  yra  $A$  tipo, tai rašome  $\Gamma \vdash E : A$ . Tarkime, kad kintamiesiems jau priskirti tipai. Tuomet likusiems terminams priskiriami tipai tokiu būdu:

- jei  $\Gamma \vdash X : A$  ir  $\Gamma \vdash Y : (A \rightarrow B)$ , tai termui  $(YX)$  priskiriamas tipas  $B$ .
- jei  $\Gamma \vdash x : A$  ir  $\Gamma \vdash Y : B$ , tai termui  $\lambda x.Y$  priskiriamas tipas  $(A \rightarrow B)$ .

**Pavyzdys.** Tarkime kintamasis  $x$  yra  $A$ , o  $f - (A \rightarrow A)$  tipo. Tuomet:

- a)  $(A \rightarrow A)$  yra termo  $\lambda x.x$  tipas,
- b)  $\lambda x.(xx)$  neturi tipo,
- c)  $((A \rightarrow A) \rightarrow (A \rightarrow A))$  yra termo  $\lambda f.\lambda x.(fx)$  tipas.
- d)  $((A \rightarrow A) \rightarrow (A \rightarrow A))$  yra ir termų  $\lambda f.\lambda x.(f^n x)$  tipas.

Yra tamprus ryšys tarp teiginių logikos formulių išvedimų natūraliosios dedukcijos sistemoje ir tipizuotų termų. Curry ir Howard įrodė, kad jei termo  $E$  tipas lygus  $A$ , tai  $E$  konstrukcija (seka termų nurodančių kaip jis gautas) nusako formulės  $A$  išvedimą natūraliosios dedukcijos sistemoje.

### 3.14 Pratimai

1. Raskite determinuotąją viena juostę Turing mašiną su abėcėle  $\Sigma = \{0, 1, \circ\}$ , apskaičiuojančią funkciją (pradiniais duomenimis yra abėcėlės

$\{0, 1\}$  žodžiai):

a)  $f(x) \equiv 0$ ,

b)  $y = f(x)$ :  $y$  gaunamas iš  $x$ , pakeitus jame vienu metu visus vienetukus nuliais, o nulius vienetais.

c)

$$f(x) = \begin{cases} \infty, & \text{jei žodyje } x \text{ nėra nulių} \\ 1, & \text{priešingu atveju} \end{cases}$$

d)  $f(x) = x - 1$ .

e)

$$f(x) = \begin{cases} 1, & \text{jei žodyje } x \text{ yra lyginis skaičius vienetų} \\ 0, & \text{priešingu atveju} \end{cases}$$

f)

$$f(x) = \begin{cases} x, & \text{jei žodyje } x \text{ pirmosios dvi raidės sutampa} \\ 101, & \text{priešingu atveju} \end{cases}$$

2. Abėcėlė  $\Sigma = \{0, 1, *, \gamma\}$ . Pradiniais duomenimis yra žodžiai pavidalo  $x * y$ , ( $x, y$  abėcėlės  $\{0, 1\}$  žodžiai). Žodžio  $z$  ilgis žymimas  $il(z)$ . Rasti determinuotąją vienajuostę Turingo mašiną, apskaičiuojančią funkciją:

$$f(x * y) = \begin{cases} 1, & \text{jei } il(x) \geq il(y) \\ 0, & \text{priešingu atveju} \end{cases}$$

3. Raskite baigtinį automata su abėcėle  $\Sigma = \{0, 1\}$ , kurio kalba yra:

a) žodžiai pavidalo  $0^m 10^n$  ( $m, n \geq 0$ ).

b) visi žodžiai, kuriuose ne mažiau kaip trys vienetukai.

4. Įrodykite, kad funkcijos yra primitiviai rekursyvios:

a)  $x \cdot y$

b)  $x^y$

c)  $x - 1$

d)  $n!$ , kai  $0! = 1$

5. Funkcija  $g(x_1, \dots, x_n)$  yra primitiviai rekursyvi. Įrodykite, kad funkcija  $f$  irgi primitiviai rekursyvi, kai:

a)

$$f(x_1, \dots, x_n) = \prod_{i=0}^{x_n} g(x_1, \dots, x_{n-1}, i).$$

b)

$$f(x_1, \dots, x_{n-1}, y, z) = \begin{cases} \sum_{i=y}^z g(x_1, \dots, x_{n-1}, i), & \text{jei } y \leq z \\ 0, & \text{jei } y > z \end{cases}$$

6. Žinoma, kad  $n$  argumentų funkcijos  $f, k, h$  primitiviai rekursyvios. Įrodykite, kad primitiviai rekursyvi yra:

$$f(x_1, \dots, x_n) = \sum_{i=h(x_1, \dots, x_n)}^{k(x_1, \dots, x_n)} g(x_1, \dots, x_{n-1}, i)$$

7. Įrodykite, kad primitiviai rekursyvi yra funkcija

$$\text{div}(x, y) = \begin{cases} 1, & \text{jei } x \text{ dalijasi iš } y \\ 0, & \text{priešingu atveju} \end{cases}$$

8. Įrodykite, kad funkcija  $nd \ x$ , kurios reikšmė lygi  $x$  daliklių (įskaitant ir vienetą) skaičiui, primitiviai rekursyvi.

9. Parašykite pirmuosius tris ketvertus pagal Cantoro numeraciją.

10. Aibė  $A$  yra rekursyvi. Įrodykite, kad jos papildinys taip pat rekursyvi aibė.

11. Aibės  $A_1, \dots, A_n$  yra rekursyvios. Įrodykite, kad jų sąjunga bei sankirta taip pat rekursyvios aibės.

12. Įrodykite, kad jei  $f(x) \in PR$ , tai lygties  $f(x) = 0$  sprendinių aibė yra rekursyvi.

13. Ar  $\pi^* = \pi$  ?

14. Kam lygus  $\omega^* + \omega$  tipas ?

15. Nustatyti kokio tipo yra aibė  
 $(0, 0) < (0, 1) < (0, 2) < \dots < (1, 0) < (1, 1) < (1, 2) < \dots$



16. Kokią funkciją apibrėžia  $B_3(a, n)$  ?

17. Įrodykite, kad iš bazinių funkcijų  $s(x), q(x)$ , naudojantis sudėties, kompozicijos bei iteracijos operatoriais, galima gauti funkciją:

a)  $pr_1^1(x)$ , b)  $f(x) \equiv 0$ , c)  $sg\ x$ .

18. Kam lygi funkcija:

a)  $I(x + 2 \cdot \sqrt{x} + 1)$ , b)  $I(\bar{s}gx)$ ?

19. Raskite funkcijų, aprašytų 17-toje užduotyje, numerius.

20. Redukuokite termus:

a)  $((\lambda a. \lambda v. \lambda f. \lambda x. (a(f((vf)x))))1)1$ ,

b)  $(\lambda x. ((x0)1))1$ .

c)  $((\lambda x. \lambda y. ((xy)x))0)1$ .

d)  $((\lambda x. \lambda y. ((xx)y))1)1$ .

## 4 skyrius

### Teiginių skaičiavimai

Skaiciavimu nusakome įrodomų jame formulų aibę. Dažniausiai tai tapačiai teisingų ar tapačiai klaidingų formulų aibės. Skaiciavimu nusakoma aibė rekursyviai skaičioji. Skaiciavimas – tai metodas, kuriuo įrodome, kad aibė rekursyviai skaičioji. Jei ji nėra išsprendžiama, tai jos papildinys nėra rekursyviai skaitusis ir neegzistuoja skaičiavimo, kuriame išvedamų formulų aibė būtų lygi papildiniui.

#### 4.1 Hilberto tipo skaičiavimas

Nagrinėsime pataisytą ir papildytą aksiomomis konjunkcijai bei disjunkcijai 1879 metais G.Frege aprašytą skaičiavimą. Vėliau buvo sukurta skaičiavimų ir su kitokiomis aksiomomis tai pačiai išvedamų formulų aibei. Skaiciavimus nagrinėjo bei gavo kai kuriuos svarbius rezultatus vokiečių matematikas D.Hilbert. Įprasta tokius skaičiavimus vadinti **Hilberto tipo skaičiavimais**.

Skaiciavimas (visur žemiau vadinsime jį *teiginių skaičiavimu*) nusakomas aksiomomis ir taisykle.

*Aksiomos* ( $A, B, C$  – bet kurios formulės):

- 1.1  $A \rightarrow (B \rightarrow A)$
- 1.2  $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
- 2.1  $(A \& B) \rightarrow A$
- 2.2  $(A \& B) \rightarrow B$
- 2.3  $(A \rightarrow B) \rightarrow ((A \rightarrow C) \rightarrow (A \rightarrow (B \& C)))$
- 3.1  $A \rightarrow (A \vee B)$
- 3.2  $B \rightarrow (A \vee B)$
- 3.3  $(A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C))$

$$4.1 \quad (A \rightarrow B) \rightarrow (\neg B \rightarrow \neg A)$$

$$4.2 \quad A \rightarrow \neg\neg A$$

$$4.3 \quad \neg\neg A \rightarrow A$$

Šios 1.1-4.3 aksiomos vadinamos aksiomų schemomis. Skaiciavime yra be galo daug aksiomų. Jos gaunamos iš aksiomų schemų.  $A, B, C$  keičiame bet kokiais formulėmis.

Vienintelė teiginių skaičiavimo taisyklė yra *modus ponens* (MP):

$$\frac{A, \quad A \rightarrow B}{B};$$

čia  $A$  ir  $B$  – bet kurios formulės.

**4.1 apibrėžimas.** Įrodymu teiginių skaičiavime vadiname baigtinę formulių seką, kurioje kiekviena formulė yra arba aksioma, arba gauta iš prieš ją esančių formulių pagal *modus ponens* taisyklę.

**4.2 apibrėžimas.** Sakome, kad formulė  $A$  įrodoma teiginių skaičiavime (žymėsime  $\vdash A$ ), jei galime rasti įrodymą, kurio paskutinis narys yra  $A$ .

**Pavyzdys.** Kad ir kokia būtų formulė  $A$ , teiginių skaičiavime įrodoma formulė  $A \rightarrow A$ . Jos įrodymas yra seka. Aksiomą

$$(A \rightarrow ((A \rightarrow A) \rightarrow A)) \rightarrow ((A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A)) \quad (4.1)$$

gauname iš 1.2 aksiomų schemos, vietoje  $A$  įrašę  $A$ , vietoje  $B$  –  $(A \rightarrow A) \rightarrow$ , vietoje  $C$  –  $A$ .

$$A \rightarrow ((A \rightarrow A) \rightarrow A) \quad (4.2)$$

gauname iš 1.1 aksiomų schemos vietoje  $A$  įrašę  $A$ , vietoj  $B$  –  $(A \rightarrow A)$ .

$$(A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A) \quad (4.3)$$

išplaukia iš (4.1) ir (4.2) formulių pagal MP taisyklę.

$$A \rightarrow (A \rightarrow A) \quad (4.4)$$

gauname iš 1.1 aksiomų schemos vietoje  $A$  įrašę  $A$ , vietoj  $B$  –  $A$ .

$$A \rightarrow A$$

išplaukia iš (4.3) ir (4.4) formulių pagal MP taisyklę.

**4.1 teorema.** *Jei formulė įrodoma teiginių skaičiavime, tai ji tapčiai teisinga.*

*Įrodymas.* Formulės

$$1.1 \ p \rightarrow (q \rightarrow p)$$

$$1.2 \ (p \rightarrow (q \rightarrow r)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow r))$$

...

$$4.3 \ \neg\neg p \rightarrow p$$

yra tapčiai teisingos formulės (tuo galime įsitikinti sudarę teisingumo lenteles). Bet kuri aksioma gaunama iš minėtų tapčiai teisingų formulių pakeitus  $p, q, r$  konkrečiomis formulėmis ir todėl yra tapčiai teisinga (žr. 2 skyrių). Jei formulės  $A$  ir  $A \rightarrow B$  tapčiai teisingos, tai ir  $B$  tapčiai teisinga. Iš tikrųjų, jei su kuria nors interpretacija,  $B$  klaidinga, tai su ta pačia interpretacija turėtų ir  $A$  būti klaidinga, nes pagal prielaidą,  $A \rightarrow B$  yra tapčiai teisinga, o tai prieštarauja formulės  $A$  tapčiam teisingumui. Todėl, jei kuri nors seka yra įrodymas, tai kiekvienas sekos narys (iš jų ir paskutinis) yra tapčiai teisinga formulė. Teorema įrodyta.

Dėl paprastumo aksiomų schemas vadiname *aksiomomis*.

**4.3 apibrėžimas.** *Sakome, kad teiginių skaičiavimo aksioma yra nepriklausoma, jei išbraukę ją iš sąrašo, gauname skaičiavimą, kuriame ji neįrodoma.*

**4.2 teorema.** *Visos teiginių skaičiavimo aksiomos yra nepriklausomos.*

Teoremos įrodymą galima rasti vadovėlyje: S.Norgėla, *Matematinė logika*, TEV, 2004.

## 4.2 Dedukcijos teorema

Raide  $\Gamma$  žymėsime baigtinę formulių seką, kuri gali būti ir tuščia.

**4.4 apibrėžimas.** *Formulės  $B$  išvedimu iš prielaidų  $\Gamma$  vadiname baigtinę formulių seką  $B_1, B_2, \dots, B_m$ , kurioje  $B_i$  ( $1 \leq i \leq m$ ) yra arba aksioma, arba viena iš prielaidų, arba gauta iš prieš ją esančių formulių  $B_l, B_k$  ( $l, k < i$ ) pagal MP taisyklę, ir  $B_m = B$  (žymima  $\Gamma \vdash B$ .)*

Ženklas  $\vdash B$  reiškia, kad  $B$  išvedama iš tuščio prielaidų sąrašo, t.y. įrodoma duotame skaičiavime. Sąvoka *įrodymas* vartojama tada, kai prielaidų sąrašas tuščias. Kadangi sąvoka *išvedimas* bendresnė, prielaidų sąrašas juk gali būti ir tuščias, todėl sąvoką *įrodymas* vartojame tik tada, kai norime *pabrėžti*, jog prielaidų sąrašas tuščias.

Pateikiame kai kurias išvedimų iš prielaidų savybes:

1.  $\Gamma, B \vdash B$ .
2. Jeigu  $\Gamma \vdash B$ , tai  $\Gamma, A \vdash B$ .
3. Jeigu  $\Gamma, A, C \vdash B$ , tai  $\Gamma, C, A \vdash B$ .
4. Jeigu  $\Gamma, A, A \vdash B$ , tai  $\Gamma, A \vdash B$ .
5. Jeigu  $\Gamma, A \vdash B$  ir  $\Gamma \vdash A$ , tai  $\Gamma \vdash B$ . Atskiru atveju, jei  $\Gamma, A \vdash B$  ir  $\vdash A$ , tai  $\Gamma \vdash B$ .

Tarkime, kad

$$B_1, B_2, \dots, B_{m-1}, B \quad (4.5)$$

yra  $B$  išvedimas iš prielaidų  $\Gamma, A$  ir seka

$$A_1, A_2, \dots, A_{k-1}, A \quad (4.6)$$

yra  $A$  išvedimas iš prielaidų  $\Gamma$  (atskiru atveju  $A$  įrodoma teiginių skaičiavime). Tada formulės  $B$  išvedimas iš prielaidų  $\Gamma$  gaunamas (4.5) sekoje formules  $A$  pakeitus (4.6) seka.

6. Jeigu  $\Gamma \vdash A_1, \Gamma \vdash A_2, \dots, \Gamma \vdash A_n$  ir  $A_1, \dots, A_n \vdash B$ , tai  $\Gamma \vdash B$ .

*Įrodymas.* Jeigu  $A_1, \dots, A_n \vdash B$ , tai pagal 2 savybę gauname  $\Gamma, A_1, \dots, A_n \vdash B$ . Pasinaudoje  $\Gamma \vdash A_n$  ir 5 savybe, gauname  $\Gamma, A_1, \dots, A_{n-1} \vdash B$ . Panašiai eliminuojame ir kitas  $A_i$ . Lieka  $\Gamma \vdash B$ . Savybė įrodyta.

7. Jeigu  $\Gamma \vdash A \rightarrow B$ , tai  $\Gamma, A \vdash B$ .

*Įrodymas.* Tarkime, kad  $B_1, \dots, B_{m-1}, A \rightarrow B$  yra formulės  $A \rightarrow B$  išvedimas iš prielaidų  $\Gamma$ . Prijungę prie sekos  $A$  (kaip prielaidą) bei  $B$  (pagal MP taisyklę iš  $A \rightarrow B$  ir  $A$ ), gauname išvedimą:

$$B_1, B_2, \dots, B_{m-1}, A \rightarrow B, A, B$$

Savybė įrodyta.

**4.3 teorema** (dedukcijos).  $\Gamma, A \vdash B$  tada ir tikrai tada, kai  $\Gamma \vdash A \rightarrow B$ .

*Įrodymas.* Jei  $\Gamma \vdash A \rightarrow B$ , tai  $\Gamma, A \vdash B$  (7 savybė). Reikia įrodyti: jei  $\Gamma, A \vdash B$ , tai  $\Gamma \vdash A \rightarrow B$ .  
Tarkime, kad

$$B_1, B_2, \dots, B_{m-1}, B_m. \quad (4.7)$$

(čia  $B_m = B$ ) yra formulės  $B$  išvedimas iš prielaidų  $\Gamma, A$ .  
Juo remdamiesi pasistengsime gauti formulės  $A \rightarrow B$  išvedimą iš prielaidų  $\Gamma$ . Šios sekos narių  $B_i$  ( $1 \leq i \leq m$ ) pakeitę į  $A \rightarrow B_i$ , sudarome seką

$$A \rightarrow B_1, \dots, A \rightarrow B_i, \dots, A \rightarrow B_m. \quad (4.8)$$

Be abejo, taip gauta seka (4.8) nebūtinai yra išvedimas, t.y. kiekvienas sekos narys nebūtinai yra aksioma arba prielaida iš  $\Gamma$ , arba gauta iš kairėje stovinčių formulių pagal MP taisyklę. Kiekvieną (4.8) sekos narį pakeiskime tokia formulių seka, kad po pakeitimų gauta seka taptų formulės  $A \rightarrow B$  išvedimu iš prielaidų  $\Gamma$ .

Nagrinėkime formulę  $A \rightarrow B_i$ . Galimi tokie atvejai:

- 1)  $B_i$  yra aksioma.
- 2)  $B_i$  yra prielaida iš sąrašo  $\Gamma$ .
- 3)  $B_i = A$ .
- 4)  $B_i$  gaunama pagal MP taisyklę iš formulių  $B_j, B_k$  ( $j, k < i$ ).

Kiekvieną šių atvejų panagrinėsime atskirai.

1)  $A \rightarrow B_i$  pakeiskime  $A \rightarrow B_i$  įrodymu:  $B_i$  (aksioma),  $B_i \rightarrow (A \rightarrow B_i)$  (1.1 aksioma),  $A \rightarrow B_i$  (pagal MP taisyklę).

2)  $A \rightarrow B_i$  keiskime analogiška seka (tik šiuo atveju  $B_i$  – prielaida).

3)  $A \rightarrow A$  keiskime jos įrodymu, kuris anksčiau buvo pateiktas pavyzdyje.

4) Šis atvejis galimas tik tada, kai  $i \geq 3$ . Kadangi  $B_i$  gauta pagal MP taisyklę iš  $B_j$  ir  $B_k$ , tai  $B_k = B_j \rightarrow B_i$  (arba  $B_j = B_k \rightarrow B_i$ : šiuo atveju įrodoma panašiai).  $A \rightarrow B_i$  keiskime tokia seka:

$(A \rightarrow (B_j \rightarrow B_i)) \rightarrow ((A \rightarrow B_j) \rightarrow (A \rightarrow B_i))$  (1.2 aksioma)

$(A \rightarrow B_j) \rightarrow (A \rightarrow B_i)$  (pagal MP taisyklę iš prieš stovinčios formulės ir formulės  $A \rightarrow B_k$ , kuri sekoje (4.8) yra kairiau formulės  $A \rightarrow B_i$ , nes  $k < i$ ; primename, kad  $A \rightarrow B_k = A \rightarrow (B_j \rightarrow B_i)$ ).

$A \rightarrow B_i$  (pagal MP taisyklę iš prieš stovinčios formulės ir  $A \rightarrow B_j$ ).

Atlikę tokius keitimus, vietoje kiekvienos formulės  $A \rightarrow B_i$  ( $i = 1, \dots, m$ ) gauname formulės  $A \rightarrow B_m = A \rightarrow B$  išvedimą. Pastebėsime, kad darydami keitimus (4.8) sekoje, naudojomes 1.1 ir 1.2 aksiomomis. Teorema įrodyta.

**Išvada.** Jei  $A_1, \dots, A_n \vdash B$ , tai  $\vdash A_1 \rightarrow (A_2 \rightarrow \dots \rightarrow (A_n \rightarrow B) \dots)$ .

Ją įrodyti galima taikant dedukcijos teoremą  $n$  kartų.

**Pavyzdys.** Naudodamiesi dedukcijos teorema, įrodykite, kad formulė  $(A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C))$  išvedama teiginių skaičiavime.

Įrodymas. Formulė išvedama teiginių skaičiavime tada ir tikrai tada, kai  $A \rightarrow B \vdash (B \rightarrow C) \rightarrow (A \rightarrow C)$ . Dar du kartus pasinaudoje dedukcijos teorema, gauname, kad pradinė formulė išvedama teiginių skaičiavime tada ir tikrai tada, kai  $C$  išvedama iš prielaidų  $A \rightarrow B, B \rightarrow C, A$ . Pastarosios išvedimą nesunku rasti:

$A$  (prielaida)

$A \rightarrow B$  (prielaida)

$B$  (pagal taisyklę MP)

$B \rightarrow C$  (prielaida)

$C$  (pagal MP taisyklę).

Išvedimas nesinaudojant dedukcijos teorema ilgesnis ir sudėtingesnis. Tiesa, neradome išvedimo pradinės formulės. Dedukcijos teorema vienos formulės išvedimą suvedame į kitos formulės su kitokiomis prielaidomis išvedimą. Mes tik įrodėme, kad pradinė formulė išvedama teiginių skaičiavime, kai egzistuoja  $C$  išvedimas ir jį (pradinės formulės išvedimą), naudojantis dedukcijos teoremos įrodymo eiga bei gautuoju  $C$  išvedimu, galim rasti.

Formali teorija vadinama absoliučiai neprieštaringa, jeigu ne visos teorijos formulės jame yra įrodomos.

Formalių teorijų, tarp kurių taisyklių yra ir *modus ponens* ir kuriose įrodoma formulė  $\neg A \rightarrow (A \rightarrow B)$  (t.y. loginėms formaliosioms teorijoms), neprieštaringumas apibrėžiamas taip:

**4.5 apibrėžimas.** Teorija vadinama **neprieštaringa**, jei neegzistuoja joje tokios formulės, kad ji bei jos neigimas, t.y. jos abi, būtų

*įrodomos teorijoje.*

Parodysime, kad teiginių skaičiavime įrodoma formulė  $\neg A \rightarrow (A \rightarrow B)$ . Ši formulė bus įrodoma tada ir tikrai tada, kai  $\neg A, A \vdash B$  (tai išplaukia iš dedukcijos teoremos). Pateiksime pastarosios išvedimą.

1.  $(\neg B \rightarrow A) \rightarrow (\neg A \rightarrow \neg\neg B)$  (4.1 aksioma,  $A$  pakeitėme  $\neg B$ .  $B$  pakeitėme  $A$ ).
2.  $A \rightarrow (\neg B \rightarrow A)$  (1.1 aksioma,  $B$  pakeitėme  $\neg B$ ).
3.  $A$  (prielaida),
4.  $\neg B \rightarrow A$  (pagal MP taisyklę iš 2, 3 formulių),
5.  $\neg A \rightarrow \neg\neg B$  (pagal MP taisyklę iš 1, 4 formulių),
6.  $\neg A$  (prielaida),
7.  $\neg\neg B$  (pagal MP taisyklę iš 5, 6 formulių),
8.  $\neg\neg B \rightarrow B$  (4.3 aksioma,  $A$  pakeitėme  $B$ ),
9.  $B$  (pagal MP taisyklę iš 7, 8 formulių).

Prieštaringa teorija bloga tuo, kad joje įrodoma bet kuri jos formulė. ir kartu toji teorija tampa nereikalinga.

Taigi, jei teiginių skaičiavime atsirastų tokia formulė, kuri ir kurios neigimas įrodomi, tai teiginių skaičiavime įrodoma ir bet kuri jo formulė.

**4.4 teorema.** *Teiginių skaičiavimas yra neprieštaringas.*

*Įrodymas.* Iš 4.1 teoremos išplaukia, kad jei kuri nors formulė įrodoma teiginių skaičiavime, tai ji tapati teisinga. Kadangi bet kurios tapati teisingos formulės neigimas yra tapati klaidinga formulė, tai neatsiras tokios formulės, kad ji bei jos neigimas būtų įrodomi teiginių skaičiavime. Teorema įrodyta.

Teiginių skaičiavimas yra pilnas tapati teisingų formulių atžvigiui. t.y. bet kuri formulė  $F$  įrodoma teiginių skaičiavime tada ir tikrai tada, kai  $F$  tapati teisinga. Pilnumo įrodymą galima rasti vadovėlyje: S.Norgėla. *Matematinė logika*. TEV, 2004.

**Kiti Hilberto tipo teiginių skaičiavimai.**

Yra sukurta daug pilnų ir neprieštaringų Hilberto tipo teiginių skaičiavimų.



Visų juose išvedamų formulių aibė yra ta pati – tapčiai teisingų formulių aibė. Pateiksime du skaičiavimus. Abu teturi tik po vieną *modus ponens* taisyklę. Nuo jau nagrinėtojo šie skaičiavimai skiriasi tik aksiomų schemomis. Antrojo formulėse yra tik neigimo ir implikacijos loginės operacijos.

Pirmojo skaičiavimo *aksiomos*:

$$1.1 \ A \rightarrow (B \rightarrow A),$$

$$1.2 \ (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)).$$

$$2.1 \ (A \& B) \rightarrow A,$$

$$2.2 \ (A \& B) \rightarrow B,$$

$$2.3 \ A \rightarrow (B \rightarrow (A \& B)).$$

$$3.1 \ A \rightarrow (A \vee B),$$

$$3.2 \ B \rightarrow (A \vee B).$$

$$3.3 \ (A \rightarrow C) \rightarrow ((B \rightarrow C) \rightarrow ((A \vee B) \rightarrow C)),$$

$$4.1 \ (A \rightarrow B) \rightarrow ((A \rightarrow \neg B) \rightarrow \neg A),$$

$$4.2 \ \neg\neg A \rightarrow A.$$

Antrojo skaičiavimo *aksiomos*:

$$1. \ A \rightarrow (B \rightarrow A).$$

$$2. \ (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)),$$

$$3. \ (\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B).$$

### 4.3 Gentzeno skaičiavimas

Hilberto tipo teiginių skaičiavimuose sunku rasti formulių įrodymus. Ne daug padeda ir išvedimų iš prielaidų savybės (pavyzdžiui, dedukcijos teorema). Net gana paprastų (pavyzdžiui,  $A \vee \neg A$ ; žr. 4.2 skyrelį) formulių įrodymai gan ilgi. O kaip įsitikinti, kad teiginių skaičiavime kuri nors formulė neišrodoma? Tik praėjus daugiau kaip penkiems dešimtmečiams po G.Frege darbų apie teiginių skaičiavimus, vokiečių logikas G.Gentzen 1930 m. aprašė kitokio tipo loginį skaičiavimą, kuris padarė perversmą

logikoje. Išvedimo paieška, bent jau teiginių logikos atveju, tapo *mechaninė* (paaikšinsime tai vėliau). Yra daug G.Gentzeno skaičiavimo variantų. Be nagrinėjamojo, kurį vadiname **sekvenciniu skaičiavimu G**, šiame skyrelyje aprašytas skaičiavimas  $G'$  ir dar vienas 6 skyriuje.

**4.6 apibrėžimas.** *Sekvencija vadiname reiškinių  $A_1, \dots, A_n \vdash B_1, \dots, B_m$ : čia  $A_i$  ( $i = 1, \dots, n$ ), bei  $B_i$  ( $i = 1, \dots, m$ ) yra formulės ir  $n + m \neq 0$ .*

Raidėmis  $\Gamma, \Gamma', \Gamma'', \Delta, \Delta', \Delta''$  žymime baigtines formulių sekas. Jos gali būti ir tuščios. Sekvencijoje  $\Gamma \vdash \Delta$  seka  $\Gamma$  vadinama **antecedentu**, o  $\Delta$  – **sukcedentu**.

*Aksiomos:*  $\Gamma', A, \Gamma'' \vdash \Delta', A, \Delta''$

*Taisyklės:*

$$(\rightarrow \vdash) \frac{\Gamma', \Gamma'' \vdash \Delta', A, \Delta'' \quad \Gamma', B, \Gamma'' \vdash \Delta', \Delta''}{\Gamma', A \rightarrow B, \Gamma'' \vdash \Delta', \Delta''}$$

$$(\vdash \rightarrow) \frac{\Gamma', A, \Gamma'' \vdash \Delta', B, \Delta''}{\Gamma', \Gamma'' \vdash \Delta', A \rightarrow B, \Delta''}$$

$$(\& \vdash) \frac{\Gamma', A, B, \Gamma'' \vdash \Delta}{\Gamma', A \& B, \Gamma'' \vdash \Delta} \quad (\vdash \&) \frac{\Gamma \vdash \Delta', A, \Delta'' \quad \Gamma \vdash \Delta', B, \Delta''}{\Gamma \vdash \Delta', A \& B, \Delta''}$$

$$(\vee \vdash) \frac{\Gamma', A, \Gamma'' \vdash \Delta \quad \Gamma', B, \Gamma'' \vdash \Delta}{\Gamma', A \vee B, \Gamma'' \vdash \Delta} \quad (\vdash \vee) \frac{\Gamma \vdash \Delta', A, B, \Delta''}{\Gamma \vdash \Delta', A \vee B, \Delta''}$$

$$(\neg \vdash) \frac{\Gamma', \Gamma'' \vdash \Delta', A, \Delta''}{\Gamma', \neg A, \Gamma'' \vdash \Delta', \Delta''} \quad (\vdash \neg) \frac{\Gamma', A, \Gamma'' \vdash \Delta', \Delta''}{\Gamma', \Gamma'' \vdash \Delta', \neg A, \Delta''}$$

Pakeitę kurioje nors taisyklėje  $\Gamma, \Gamma', \Gamma'', \Delta, \Delta', \Delta'', A, B$  konkrečiomis formulėmis, gauname **taisyklės taikymą**.

Sekvencijos, esančios taisyklėje virš brūkšnio arba jos taikyme virš brūkšnio, vadinamos **prielaidomis** (jų gali būti ne daugiau kaip dvi), o žemiau brūkšnio – **išvada** (ji visada viena).

Pavyzdžiui, taisyklės  $(\vee \vdash)$  išvada yra  $\Gamma', A \vee B, \Gamma'' \vdash \Delta$ , o prielaidomis yra  $\Gamma', A, \Gamma'' \vdash \Delta$ ,  $\Gamma', B, \Gamma'' \vdash \Delta$ .

Jei  $\Gamma \vdash \Delta$  yra kurios nors taisyklės  $\alpha$  taikymo išvada, o  $\Gamma' \vdash \Delta'$ ,  $\Gamma'' \vdash \Delta''$  (jų gali būti ir viena) prielaidos, tai sakoma, kad  $\Gamma \vdash \Delta$  yra

gauta iš  $\Gamma' \vdash \Delta'$ ,  $\Gamma'' \vdash \Delta''$ , pritaikius taisyklę  $\alpha$ .

**4.7 apibrėžimas.** *Sekvencijos išvedimu sekvenciniame skaičiavime  $G$  vadiname medį, kurio visose galinėse viršūnėse (lapuose) yra aksiomos, likusiose viršūnėse – formulės, gautos pagal kurią nors sekvencinio skaičiavimo taisyklę iš tiesiogiai virš jų medyje esančių formulių, ir šaknyje esanti sekvencija lygi pradinei.*

**Pavyzdžiai:** 1.  $\vdash A \vee \neg A$ .

$$\frac{\frac{A \vdash A}{\vdash A, \neg A}}{\vdash A \vee \neg A}$$

2.  $(A \& B) \rightarrow C, A \& \neg C \vdash \neg B$ .

$$\frac{\frac{\frac{A, B \vdash C, A \quad A, B \vdash C, B}{A, B \vdash C, A \& B} \quad C, A, B \vdash C}{(A \& B) \rightarrow C, A, B \vdash C}}{\frac{(A \& B) \rightarrow C, A, \neg C, B \vdash}{(A \& B) \rightarrow C, A \& \neg C, B \vdash}}{\frac{(A \& B) \rightarrow C, A \& \neg C \vdash \neg B}$$

Primename, kad išvedimuose brūkšniai atitinka grafo lankus (kryptis – iš apačios į viršų), o sekvencijos – viršūnes. Ilgiausiame kelyje nuo šaknies iki viršūnės aptinkamų sekvencijų skaičius, vadinamas išvedimo aukščiu.

Taisyklių (taisyklių taikymų)  $(\rightarrow \vdash), (\vdash \rightarrow)$  *centrine formule* vadinama  $(A \rightarrow B)$ , taisyklių  $(\& \vdash), (\vdash \&)$  –  $A \& B$ , taisyklių  $(\vee \vdash), (\vdash \vee)$  –  $A \vee B$ , bei taisyklių  $(\neg \vdash), (\vdash \neg)$  –  $\neg A$ .

G. Gentzen įrodė, kad formulė  $A$  *tapačiai teisinga tada ir tik tai tada, kai sekvencija  $\vdash A$  išvedama skaičiavime  $G$ .*

Iš čia išplaukia, kad formulė  $A$  *tapačiai klaidinga tada ir tik tai tada, kai  $A \vdash$  išvedama skaičiavime  $G$ .*

**4.8 apibrėžimas.** *Sakome, kad taisyklė  $\alpha$  apverčiama, jei jos prielaidos išvedamos sekvenciniame skaičiavime tada ir tik tai tada, kai išvedama išvada.*

**4.5 teorema.** *Visos sekvencinio skaičiavimo  $G$  taisyklės apverčiamos.*

*Irodymas.* Nagrinėjame taisyklę ( $\vdash \&$ ) ir sekvenciją

$$\Gamma \vdash \Delta', A \& B, \Delta'' \quad (4.9)$$

Irodysime, kad  $\Gamma \vdash \Delta', A \& B, \Delta''$  išvedama sekvenciniame skaičiavime  $G$  tada ir tikrai tada, kai išvedamos abi sekvencijos  $\Gamma \vdash \Delta', A, \Delta''$  ir  $\Gamma \vdash \Delta', B, \Delta''$ . Jei jos išvedamos, tai ir (4.9) išvedama. Išvedimo medis yra

$$\frac{\frac{D_1}{\Gamma \vdash \Delta', A, \Delta''} \quad \frac{D_2}{\Gamma \vdash \Delta', B, \Delta''}}{\Gamma \vdash \Delta', A \& B, \Delta''};$$

čia

$$\frac{D_1}{\Gamma \vdash \Delta', A, \Delta''}, \quad \frac{D_2}{\Gamma \vdash \Delta', B, \Delta''}$$

yra sekvencijų  $\Gamma \vdash \Delta', A, \Delta''$  ir  $\Gamma \vdash \Delta', B, \Delta''$  išvedimų medžiai.

Tarkime, (4.9) išvedama. Irodysime, kad sekvencijos  $\Gamma \vdash \Delta', A, \Delta''$  ir  $\Gamma \vdash \Delta', B, \Delta''$  išvedamos sekvenciniame skaičiavime  $G$ . Irodysime indukcija pagal (4.9) sekvencijos švedimo aukštį  $l$ .

*Indukcinė prielaida.* Tarkime  $l = 0$ , t.y. (4.9) yra aksioma. Tuomet sekvencijos (4.9) antecedente ir sukcedente yra viena ir ta pati formulė  $F$ . Jei  $F \neq A \& B$ , tai  $\Gamma \vdash \Delta', A, \Delta''$  bei  $\Gamma \vdash \Delta', B, \Delta''$  yra tap pat aksiomos ir kartu išvedamos skaičiavime. Jei  $F = A \& B$ , t.y.  $\Gamma = \Gamma', A \& B, \Gamma''$ , tai jų išvedimai tokie:

$$\frac{\Gamma', A, B, \Gamma'' \vdash \Delta', A, \Delta''}{\Gamma', A \& B, \Gamma'' \vdash \Delta', A, \Delta''}, \quad \frac{\Gamma', A, B, \Gamma'' \vdash \Delta', B, \Delta''}{\Gamma', A \& B, \Gamma'' \vdash \Delta', B, \Delta''}.$$

Tarkime tvirtinimas teisingas, kai  $l < m$ . Parodysime, kad jis teisingas ir kai  $l = m$ .

*1 atvejis.* Pirmasis, iš apačios į viršų, (4.9) išvedime taisyklės taikymas yra ( $\vdash \&$ ) su centrine formule  $A \& B$ , t.y. išvedimo medis yra pavidalo ( $D_1, D_2$  – išvedimų medžiai):

$$\frac{\frac{D_1}{\Gamma \vdash \Delta', A, \Delta''} \quad \frac{D_2}{\Gamma \vdash \Delta', B, \Delta''}}{\Gamma \vdash \Delta', A \& B, \Delta''}.$$

Tuomet nagrinėjamųjų sekvencijų išvedimų medžiais bus

$$\frac{D_1}{\Gamma \vdash \Delta', A, \Delta''}, \quad \frac{D_2}{\Gamma \vdash \Delta', B, \Delta''}.$$

2 atvejis. Pirmojo taisyklės taikymo centrine formule nėra  $A \& B$  ir sekvencijos (4.9) išvedimo medžio aukštis lygus  $m$ . Tarkime išvedimo medis yra pavidalo

$$\frac{\frac{D_1}{\Gamma' \vdash \Delta'_1, A \& B, \Delta''_1} \quad \frac{D_2}{\Gamma'' \vdash \Delta'_2, A \& B, \Delta''_2}}{\Gamma \vdash \Delta', A \& B, \Delta''}$$

Panašiai būtų įrodoma, jei (4.9) sekvencijos išvedime prielaidos būtų ne dvi, o viena sekvencija. Abiejų išvedimų medžių

$$\frac{D_1}{\Gamma' \vdash \Delta'_1, A \& B, \Delta''_1}, \quad \frac{D_2}{\Gamma'' \vdash \Delta'_2, A \& B, \Delta''_2}$$

aukščiau neviršija  $m - 1$ . Todėl jiems galioja indukcinė prielaida:

a)  $\Gamma' \vdash \Delta'_1, A \& B, \Delta''_1$  išvedama tada ir tik tada, kai išvedamos abi sekvencijos  $\Gamma' \vdash \Delta'_1, A, \Delta''_1$  ir  $\Gamma' \vdash \Delta'_1, B, \Delta''_1$ . Taigi, atsiras pastarųjų dviejų išvedimų medžiai:

$$\frac{D'_1}{\Gamma' \vdash \Delta'_1, A, \Delta''_1}, \quad \frac{D'_2}{\Gamma' \vdash \Delta'_1, B, \Delta''_1}$$

b)  $\Gamma'' \vdash \Delta'_2, A \& B, \Delta''_2$  išvedama tada ir tik tada, kai išvedamos abi sekvencijos  $\Gamma'' \vdash \Delta'_2, A, \Delta''_2$  ir  $\Gamma'' \vdash \Delta'_2, B, \Delta''_2$ . Taigi, atsiras pastarųjų dviejų išvedimų medžiai:

$$\frac{D'_3}{\Gamma'' \vdash \Delta'_2, A, \Delta''_2}, \quad \frac{D'_4}{\Gamma'' \vdash \Delta'_2, B, \Delta''_2}$$

Tuomet  $\Gamma \vdash \Delta', A, \Delta''$  bei  $\Gamma \vdash \Delta', B, \Delta''$  išvedimų medžiai yra:

$$\frac{\frac{D'_1}{\Gamma' \vdash \Delta'_1, A, \Delta''_1} \quad \frac{D'_3}{\Gamma'' \vdash \Delta'_2, A, \Delta''_2}}{\Gamma \vdash \Delta', A, \Delta''}, \quad \frac{\frac{D'_2}{\Gamma' \vdash \Delta'_1, B, \Delta''_1} \quad \frac{D'_4}{\Gamma'' \vdash \Delta'_2, B, \Delta''_2}}{\Gamma \vdash \Delta', B, \Delta''}$$

Panašiai įrodomas ir likusiųjų taisyklių apverčiamumas. Teorema įrodyta.

Praktiškai išvedimo medis konstruojamas iš apačios į viršų, o taisyklės – apverčiamos, todėl patogiu naudotis kitu išvedimo apibrėžimu.

**4.9 apibrėžimas.** *Sekvencijos išvedimu vadinsime medžio pavidalo orientuotą grafą, kurio visos viršūnės pažymėtos sekvencijomis (šaknis – pradine sekvencija) ir virš kiekvienos viršūnės visos tiesiogiai esančios sekvencijos gautos iš nagrinėjamąją viršūnę atitinkančios sekvencijos, pritaikius kurią nors sekvencinio skaičiavimo taisyklę. Visoms medžio galines*

viršūnes, t.y. lapus atitinkančios sekvencijos yra aksiomos.

Išvedimas sekvenciniame skaičiavime  $G$  yra *mechaninis* ta prasme, kad, jei galima rinktis, kurią taisyklę taikyti, tai galima taikyti bet kurią iš jų (išplaukia iš 4.5 teoremos). Sekvencija bus išvedama tada ir tik tada, kai išvedamos visos gautosios. Sekvencinio skaičiavimo taisyklėms būdinga tai, kad pritaikius kurią nors jų gaunamos sekvencijos yra paprastesnės, t.y. loginių operacijų skaičius vienetu mažesnis. Todėl, jei pradinėje sekvencijoje yra  $n$  operacijų įeičių, tai pritaikę ne daugiau kaip  $n$  kartų skaičiavimo taisykles, arba visose medžio viršūnėse bus aksiomos, arba vienoje jų bus sekvencija, kuri nėra aksioma ir kurioje nėra loginių operacijų įeičių. Taigi, turime *mechaninę procedūrą*, kuria patikrinama, ar sekvencija išvedama. O tai reiškia, kad išvedamų sekvencijų aibė yra rekursyvi. Rekursyvi aibė yra ir jos papildinys.

**4.10 apibrėžimas.** Antisekvencija vadiname reiškinių  $A_1, \dots, A_n \dashv B_1, \dots, B_m$ ; čia  $A_i$  ( $i = 1, \dots, n$ ),  $B_i$  ( $i = 1, \dots, m$ ) yra formulės ir  $n + m \neq 0$ .

Panagrinėkime skaičiavimą  $\bar{G}$ .

*Aksiomos:*  $\Gamma \dashv \Delta$ . Sekos  $\Gamma, \Delta$  yra tik iš loginių kintamųjų. Be to, nėra to paties loginio kintamojo, įeinančio ir į antecedentą, ir į sukcedentą.

*Taisyklės:*

$$(\rightarrow \dashv_1) \frac{\Gamma', \Gamma'' \dashv \Delta', A, \Delta''}{\Gamma', A \rightarrow B, \Gamma'' \dashv \Delta', \Delta''}, \quad (\rightarrow \dashv_2) \frac{\Gamma', B, \Gamma'' \dashv \Delta', \Delta''}{\Gamma', A \rightarrow B, \Gamma'' \dashv \Delta', \Delta''}.$$

$$(\dashv \rightarrow) \frac{\Gamma', A, \Gamma'' \dashv \Delta', B, \Delta''}{\Gamma', \Gamma'' \dashv \Delta', A \rightarrow B, \Delta''},$$

$$(\& \dashv) \frac{\Gamma', A, B, \Gamma'' \dashv \Delta}{\Gamma', A \& B, \Gamma'' \dashv \Delta}, \quad (\dashv \&) \frac{\Gamma \dashv \Delta', A_i, \Delta''}{\Gamma \dashv \Delta', A_1 \& A_2, \Delta''} \quad (i \in \{1, 2\}).$$

$$(\vee \dashv) \frac{\Gamma', A_i, \Gamma'' \dashv \Delta}{\Gamma', A_1 \vee A_2, \Gamma'' \dashv \Delta} \quad (i \in \{1, 2\}), \quad (\dashv \vee) \frac{\Gamma \dashv \Delta', A, B, \Delta''}{\Gamma \dashv \Delta', A \vee B, \Delta''}.$$

$$(\neg \dashv) \frac{\Gamma', \Gamma'' \dashv \Delta', A, \Delta''}{\Gamma', \neg A, \Gamma'' \dashv \Delta', \Delta''}, \quad (\dashv \neg) \frac{\Gamma', A, \Gamma'' \dashv \Delta', \Delta''}{\Gamma', \Gamma'' \dashv \Delta', \neg A, \Delta''}.$$

**Pavyzdys.**  $p \vee q \dashv p \& q$ .

$$\frac{\frac{\frac{\text{aksioma}}{q \rightarrow p}}{p \vee q \rightarrow p}}{p \vee q \rightarrow p \& q}$$

Skaiciavimą, kuris skiriasi nuo  $G$  tik tuo, kad aksiomomis yra sekvencijos pavidalo  $\Gamma', p, \Gamma'' \vdash \Delta', p, \Delta''$ , t.y. aksiomos antecedente ir sukcedente turi būti vieno ir to paties loginio kintamojo įeitys, pažymėkime  $G'$ .

**4.6 teorema.** *Sekvencija išvedama skaičiavime  $G$  tada ir tikrai tada, kai ji išvedama skaičiavime  $G'$ .*

*Irodymas.* Jei kuri nors sekvencija išvedama skaičiavime  $G'$ , tai kartu ji išvedama ir skaičiavime  $G$ . Parodysime, kad jei kuri nors sekvencija išvedama skaičiavime  $G$ , tai galima rasti ir jos išvedimą skaičiavime  $G'$ . Pakanka parodyti, kad kiekviena sekvencija pavidalo  $\Gamma', A, \Gamma'' \vdash \Delta', A, \Delta''$  išvedama skaičiavime  $G'$ . Taikysime indukciją pagal loginių operacijų įeičių formulėje  $A$  skaičių (žymėsime  $l(A)$ ).

Tarkime, kad  $l(A) = 0$ . Tuomet  $A$  yra loginis kintamasis ir sekvencija išvedama skaičiavime  $G'$ , nes tai yra to skaičiavimo aksioma. Tarkime, teorema teisinga su  $l(A) < m$ . Parodysime, kad ji teisinga, kai  $l(A) = m$ .  $A$  gali būti vieno iš pavidalų: a)  $B \rightarrow C$ , b)  $B \& C$ , c)  $B \vee C$ , d)  $\neg B$ . Sekvencija yra pavidalo  $\Gamma', B \rightarrow C, \Gamma'' \vdash \Delta', B \rightarrow C, \Delta''$ . Nagrinėjame medį

$$\frac{\frac{\Gamma', B, \Gamma'' \vdash \Delta', B, C, \Delta'' \quad \Gamma', C, B, \Gamma'' \vdash \Delta', C, \Delta''}{\Gamma', B \rightarrow C, B, \Gamma'' \vdash \Delta', C, \Delta''}}{\Gamma', B \rightarrow C, \Gamma'' \vdash \Delta', B \rightarrow C, \Delta''}$$

Kadangi  $l(B) < m$  ir  $l(C) < m$ , tai galioja indukcijos prielaida ir sekvencijas  $\Gamma', B, \Gamma'' \vdash \Delta', B, C, \Delta''$  bei  $\Gamma', C, B, \Gamma'' \vdash \Delta', C, \Delta''$ , jei jos dar nėra  $G'$  aksiomos, galima pratęsti iki aksiomų. Panašiai įrodomi ir likusieji atvejai. Teorema įrodyta.

#### 4.4 Natūralioji dedukcija

Lenkas S.Jaskowski ir vokiečių G.Gentzen 1934 m. nepriklausomai vienas nuo kito aprašė taip vadinamąsias **natūraliosios dedukcijos** sistemas. Skaiciavimai vadinami **natūraliosiomis** sistemomis, nes perėjimai nuo prielaidų prie išvadų geriausiai (iš visų žinomų skaičiavimų) modeliuoja tiek šnekamosios kalbos, tiek ir mokslininkų vartojamus išvedimuose (įrodymuose)

samprotavimus.

Nagrinėjamas skaičiavimas apibendrina natūraliųjų skaičiavimų variantus ir skiriasi nuo pradinių tokio tipo skaičiavimų. Kaip ir sekvenciniame skaičiavime, sekvencija  $\vdash F$  išvedama tada ir tikrai tada, kai  $F$  tapaciai teisinga. Atkreipiame dėmesį, kad sekvencijų sukcedentuose yra ne daugiau kaip viena formulė.

• *Aksioma:*

$$A \vdash A$$

•  $\rightarrow$  įvedimas

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B},$$

•  $\rightarrow$  eliminacija

$$\frac{\Gamma \vdash A \quad \Delta \vdash A \rightarrow B}{\Gamma, \Delta \vdash B},$$

•  $\&$  įvedimas

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \& B}.$$

•  $\&_1$  eliminacija

$$\frac{\Gamma \vdash A \& B}{\Gamma \vdash A},$$

•  $\&_2$  eliminacija

$$\frac{\Gamma \vdash A \& B}{\Gamma \vdash B},$$

•  $\vee_1$  įvedimas

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B},$$

•  $\vee_2$  įvedimas

$$\frac{\Gamma \vdash B}{\Gamma \vdash A \vee B},$$

•  $\vee$  eliminacija

$$\frac{\Gamma \vdash A \vee B \quad \Delta, A \vdash C \quad \Delta', B \vdash C}{\Gamma, \Delta, \Delta' \vdash C}.$$

•  $\neg$  įvedimas

$$\frac{\Gamma, A \vdash}{\Gamma \vdash \neg A},$$

•  $\neg_1$  eliminacija

$$\frac{\Gamma \vdash A \quad \Delta \vdash \neg A}{\Gamma, \Delta \vdash}.$$



- $\neg_2$  eliminacija

$$\frac{\Gamma, \neg A \vdash}{\Gamma \vdash A}.$$

*Struktūrinės taisyklės:*

- silpninimas

$$\frac{\Gamma \vdash}{\Gamma \vdash A}, \quad \frac{\Gamma \vdash A}{\Gamma, B \vdash A},$$

- perstatymas

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C},$$

- kartojimas

$$\frac{\Gamma, A, A, \Delta \vdash C}{\Gamma, A, \Delta \vdash C}.$$

Čia  $\Gamma, \Delta, \Delta'$  yra baigtinės formulių sekos (gali būti ir tuščios),  $A, B, C$  - formulės. Taisyklių prielaidose esančių sekvenčių tvarka nesvarbi.

Dėl patogumo išvedimo medyje aptiktą sekvenciją  $\Gamma, A, \Delta \vdash A$  taip pat laikysime aksioma, nes, naudojantis tik struktūrinėmis taisyklėmis, nesunku ją pratęsti iki norimos sekvencijos  $A \vdash A$ . Be to, naudojantis tik struktūrinėmis taisyklėmis, iš  $\Gamma, \Delta \vdash A$  galima gauti  $\Delta, \Gamma \vdash A$ , todėl taikant taisykles galime nekreipti dėmesio į išvados antecedente esančių formulių tvarką.

Pateikiame išvedimų natūraliosios dedukcijos sistemoje porą pavyzdžių:

$$\frac{\frac{\frac{A \vdash A}{A \vdash A \vee B} \quad \neg(A \vee B) \vdash \neg(A \vee B)}{\neg(A \vee B), A \vdash} \quad \frac{\frac{B \vdash B}{B \vdash A \vee B} \quad \neg(A \vee B) \vdash \neg(A \vee B)}{\neg(A \vee B), B \vdash}}{\neg(A \vee B) \vdash \neg A \quad \neg(A \vee B) \vdash \neg B} \\ \hline \neg(A \vee B) \vdash \neg A \& \neg B \\ \hline \vdash \neg(A \vee B) \rightarrow (\neg A \& \neg B)$$

$$\frac{A \rightarrow B, B \rightarrow C, A \vdash B \rightarrow C \quad \frac{A \rightarrow B, A \vdash A \rightarrow B \quad B \rightarrow C, A \vdash A}{A \rightarrow B, B \rightarrow C, A \vdash B}}{A \rightarrow B, B \rightarrow C, A \vdash C}$$

Aprašysime dar vieną natūraliosios dedukcijos sistemos variantą. Jį sudaro dviejų grupių taisyklių.

Pirmoji grupė taisyklių.

1. *Modus ponens* (MP): iš  $A$  ir  $A \rightarrow B$  išvedama  $B$ .

$$\frac{A \quad A \rightarrow B}{B}$$

2. *Modus tollens* (MT): iš  $A \rightarrow B$  ir  $\neg B$  išvedama  $\neg A$ .

$$\frac{A \rightarrow B \quad \neg B}{\neg A}$$

3. *Hipotetinis silogizmas* (HS): iš  $A \rightarrow B$  ir  $B \rightarrow C$  išvedama  $A \rightarrow C$ .

$$\frac{A \rightarrow B \quad B \rightarrow C}{A \rightarrow C}$$

4. *Disjunktivus silogizmas* (DS): iš  $A \vee B$  ir  $\neg A$  išvedama  $B$ .

$$\frac{A \vee B \quad \neg A}{B}$$

5. *Prijungimas* (Pri): iš  $A$  išvedama  $A \vee B$ .

$$\frac{A}{A \vee B}$$

6. *Skaidymas* (Sk): iš  $A \& B$  išvedama  $A$ .

$$\frac{A \& B}{A}$$

7. *Apjungimas* (Ap): iš  $A$  ir  $B$  išvedama  $A \& B$ .

$$\frac{A \quad B}{A \& B}$$

8. *Konstruktinio dilema* (KD): iš  $(A \rightarrow B) \& (C \rightarrow D)$  ir  $A \vee C$  išvedama  $B \vee D$ .

$$\frac{(A \rightarrow B) \& (C \rightarrow D) \quad A \vee C}{B \vee D}$$

Antroji grupė taisyklių. Kiekvienoje taisyklėje nurodysime po porą ekvivalenčių formulių. Išvedime leidžiama iš vienos jų gauti (išvesti) antrąją.

9. *Dvigubas neigimas* (DN):  $A$  ir  $\neg \neg A$ .

10. *Transpozicija* (Tr):  $A \rightarrow B$  ir  $\neg B \rightarrow \neg A$ .

- 11a. *Komutatyvumas* (Kom):  $A \vee B$  ir  $B \vee A$ .

- 11b. *Komutatyvumas* (Kom):  $A \& B$  ir  $B \& A$ .
- 12a. *Asociatyvumas* (As):  $A \vee (B \vee C)$  ir  $(A \vee B) \vee C$ .
- 12b. *Asociatyvumas* (As):  $A \& (B \& C)$  ir  $(A \& B) \& C$ .
- 13a. *Distributyvumas* (Dist):  $A \& (B \vee C)$  ir  $(A \& B) \vee (A \& C)$ .
- 13b. *Distributyvumas* (Dist):  $A \vee (B \& C)$  ir  $(A \vee B) \& (A \vee C)$ .
- 14a. *De Morgano dėsnis* (DeM):  $\neg(A \& B)$  ir  $\neg A \vee \neg B$ .
- 14b. *De Morgano dėsnis* (DeM):  $\neg(A \vee B)$  ir  $\neg A \& \neg B$ .
15. *Sąlygos dėsnis* (Sl):  $A \rightarrow B$  ir  $\neg A \vee B$ .
16. *Dvigubos sąlygos dėsnis* (Dsl):  $A \leftrightarrow B$  ir  $(A \rightarrow B) \& (B \rightarrow A)$ .
17. *Sukeitimas* (Suk):  $(A \& B) \rightarrow C$  ir  $A \rightarrow (B \rightarrow C)$ .
18. *Absorbcijos* (Abs):  $A \rightarrow B$  ir  $A \rightarrow (A \& B)$ .
- 19a. *Prastinimas* (Pr):  $A$  ir  $A \vee A$ .
- 19b. *Prastinimas* (Pr):  $A$  ir  $A \& A$ .

Formulės  $F$  išvedimu iš formulių  $F_1, \dots, F_n$  vadinsime baigtinę formulių seką  $G_1, \dots, G_s$ , kurioje  $G_i$  ( $i = 1, \dots, s$ ) yra arba prielaida (duotoji formulė). t.y.  $G_i \in \{F_1, \dots, F_n\}$ , arba gauta iš kairėje jos esančių formulių pagal kurią nors taisyklę.

**Pavyzdys.** Parodysime, kad iš  $\neg p, q \vee r, p \leftrightarrow q, r \rightarrow u$  išvedama  $u$ .

1. $p \leftrightarrow q$	prielaida
2. $\neg p$	prielaida
3. $q \vee r$	prielaida
4. $r \rightarrow u$	prielaida
5. $(p \rightarrow q) \& (q \rightarrow p)$	iš (1) pagal Dsl taisyklę
6. $q \rightarrow p$	iš (5) pagal Sk taisyklę
7. $\neg q$	iš (1), (6) pagal MT taisyklę
8. $r$	iš (3), (7) pagal DS taisyklę
9. $u$	iš (4), (8) pagal MP taisyklę

## 4.5 Disjunktų dedukcinė sistema

Priminsime, kad *disjunktų* vadiname literų disjunkciją, t.y formulę pavidalo  $l_1 \vee \dots \vee l_s$ ; čia  $l_i$  ( $i = 1, \dots, s$ ) yra literos. Disjunktus žymėsime  $C, C_0, C_1, \dots$ . Tuščias disjunktas žymimas simboliu  $\square$ . Šio skyrelio nagrinėjimo objektas – aibės, kurių elementais yra disjunktai bei iš jų išvedami disjunktai. *Deductio* (lotyniškai) – išvedimas.

Yra tik viena išvedimo taisyklė – **atkirtos taisyklė**. Ji taikoma dviems disjunktams, rezultatas – vienas disjunktas. Taisyklė yra labai paprasta:

$$\frac{C_1 \vee p \vee C_2, \quad C_3 \vee \neg p \vee C_4}{C_1 \vee C_2 \vee C_3 \vee C_4}$$

Paiškinsime ją. Taisyklę (sutrumpintai žymėsime ją AT) galima taikyti tik tuo atveju, kai viename disjunktų yra kurio nors loginio kintamojo (taisyklėje jis pažymėtas raide  $p$ ) įeitis, o antrajame – jo neigimas. Kadangi  $A \vee B \equiv B \vee A$  ir literų tvarka disjunktuose nesvarbi, tai atkirtos taisyklę galima nusakyti ir taip:

$$\frac{p \vee C_1, \quad \neg p \vee C_2}{C_1 \vee C_2}.$$

Kai kada atkertamą literą patiksliname ir sakome, kad *taikome atkirtos taisyklę atžvilgiu kintamojo  $p$* . Be to, tarsime, kad bet kurio loginio kintamojo įeitis bet kuriame disjunkte yra tik viena. Pavyzdžiui,  $p \vee p \vee \neg q \vee \neg q$  laikysime lygiu  $p \vee \neg q$  ir nagrinėsime pastarąjį.

**4.11 apibrėžimas.** Sakysime, kad disjunktas  $C$  išvedamas iš disjunktų aibės  $S$  (žymėsime  $S \vdash C$ ), jei yra tokia baigtinė disjunktų seka  $C_1, \dots, C_u$ , kurioje kiekvienas  $C_i$  ( $i = 1, \dots, u$ ) arba priklauso aibei  $S$ , arba yra gautas iš kairėje jo stovinčių disjunktų pagal atkirtos taisyklę. Be to,  $C_u = C$ .

Nagrinėjamoji išvedimo sistema dažniausiai vadinama *teiginių logikos metodu*.

**Pavyzdžiai.** Skliaustuose nurodome, kaip gautas disjunktas, t.y. ar jis priklauso pradinei aibei  $S$ , ar gautas pagal atkirtos taisyklę.

1.  $S = \{\neg p \vee q, \neg q \vee r, p \vee q \vee r, \neg r\}$ . Parodysime, kad  $S \vdash q$ .

$\neg p \vee q(S), \neg q \vee r(S), \neg p \vee r(AT), \neg r(S), \neg p(AT), p \vee q \vee r(S), q \vee r(AT), q(AT)$ .

2.  $S = \{\neg p \vee q \vee r, \neg p \vee \neg r, \neg q, p\}$ . Parodysime, kad  $S \vdash \square$ .

$\neg p \vee q \vee r(S), p(S), q \vee r(AT), \neg q(S), r(AT), \neg p \vee \neg r(S), \neg p(AT), \square(AT)$ .

Disjunktų išvedimai aprašomi ir kitokiais būdais. Paaiškinsime tai remdamiesi antruoju pavyzdžiu.

a) Išvedimas kaip taisyklių taikymų seka:

$$\frac{\neg p \vee q \vee r, p}{q \vee r}, \quad \frac{q \vee r, \neg q}{r}, \quad \frac{\neg p \vee \neg r, r}{\neg p}, \quad \frac{p, \neg p}{\square}.$$

b) Išvedimas kaip orientuotas grafas.

$$\frac{\frac{\frac{\neg p \vee q \vee r \quad p}{q \vee r} \quad \neg q}{r} \quad \neg p \vee \neg r}{\neg p} \quad p$$

□

**4.12 apibrėžimas.** Formulų aibė vadinama prieštarąja, jei nesvarbu, kokia būtų interpretacija, atsiras aibėje bent viena klaidinga formulė.

Pagal apibrėžimą, aibė  $S = \{C_1, \dots, C_s\}$  prieštarąja tada ir tik tada, kai formulė  $C_1 \& C_2 \& \dots \& C_s$  yra tapati klaidinga. Atkreipiame dėmesį, kad tuščias disjunktas neįvykdomas.

**4.7 teorema.** Jei  $S \vdash C$  ir  $C$  nėra įvykdomas, tai aibė  $S$  prieštarąja.

*Irodymas.* Tarkime  $C_1, C_2, \dots, C_s = C$  yra disjunktų  $C$  išvedimas iš aibės  $S$  ir aibė  $S$  įvykdoma. Atsiras interpretacija  $\nu$ , su kuria visi aibės  $S$  disjunktai teisingi.

Išvedimo ilgiu vadinsime formulų, esančių išvedimo sekoje, skaičių. Taikydami indukciją pagal išvedimo ilgį  $s$  parodysime, kad su ta pačia interpretacija  $\nu$  ir  $C$  yra teisingas.

Jei  $s = 1$ , tai  $C_1 \in S$  ir todėl  $\nu(C_1) = t$ . Tarkime, kad visi disjunktai  $C_i$  ( $i < m$ ) tenkina sąlygą  $\nu(C_i) = t$ . Parodysime, kad ir  $\nu(C_m) = t$ .  $C_m$  yra arba aibės  $S$  elementas (tuomet yra duota, kad  $\nu(C_m) = t$ ), arba gautas iš kairėje jo esančių disjunktų (pažymėkime juos  $C_j, C_k$ ) pagal atkirtos taisyklę.

Tarkime  $C_j = p \vee C'_j$ ,  $C_k = \neg p \vee C'_k$ , ir  $C_m = C'_j \vee C'_k$ . Pagal indukcijos prielaidą abu  $C_j, C_k$  teisingi su interpretacija  $\nu$ . Galimi atvejai:

a)  $\nu(p) = t$ . b)  $\nu(p) = k$ . Atveju (a)  $\nu(C'_k) = t$  ir todėl  $\nu(C_m) = t$ , o atveju (b) –  $\nu(C'_j) = t$  ir todėl  $\nu(C_m) = t$ , t.y.  $C_m$  įvykdomas su ta pačia interpretacija.

Taigi gavome: jei  $S$  įvykdoma, tai ir  $C$  įvykdomas. Jei  $S \vdash C$  ir  $C$  nėra įvykdomas, tai aibė  $S$  prieštaringa. Teorema įrodyta.

**Išvada.** *Jei iš disjunktų aibės  $S$  išvedamas tuščias disjunktas, tai aibė  $S$  prieštaringa.*

Tarkime aibės  $S = \{C_1, \dots, C_m, C_{m+1}, \dots, C_s\}$  disjunktai tenkina savybes:

a) kuris nors loginis kintamasis  $p$  įeina į  $C_i$  ( $i = m+1, \dots, s$ ) ir neįeina į  $C_j$  ( $j = 1, \dots, m$ ),

b) litera  $\neg p$  neįeina į jokią aibės  $S$  disjunktą.

Tuomet  $S$  prieštaringa tada ir tikrai tada, kai prieštaringa aibė  $S' = \{C_1, \dots, C_m\}$ . Iš tikrųjų, jei atsirastų interpretacija  $\nu$ , su kuria  $\nu(C_i) = t$  ( $i = 1, \dots, m$ ), tai pratęsus ją ( $p = t$ ), gausime, kad  $S$  įvykdoma. Jei nėra interpretacijos, su kuria  $S'$  įvykdoma, t.y., jei  $S'$  prieštaringa, tai tokia bus ir  $S$ .

Gavome tam tikrą tuščio disjunkto išvedimo paieškos taktiką: *išbraukti visus tuos disjunktus, kuriuose yra loginis kintamasis, tenkinantis sąlygas (a), (b)*. Jei gauta aibė tuščia, tai ji įvykdoma (su  $p = t$ ).

Panašiai galim elgtis ir tuo atveju, kai aibė  $S = \{C_1, \dots, C_m, C_{m+1}, \dots, C_s\}$  tenkina sąlygas:

a) atsiras toks loginis kintamasis  $p$ , kad prieš kiekvieną jo įeitį yra neigimas (yra tik  $\neg p$  įeitis),

b) litera  $\neg p$  įeina į visas  $C_i$  ( $i = m+1, \dots, s$ ) ir neįeina į  $C_j$  ( $j = 1, \dots, m$ ).

**4.8 teorema.** *Jei disjunktų aibė prieštaringa, tai iš  $S$  išvedamas tuščias disjunktas.*

*Įrodymas.* Taikome indukciją pagal skirtingų loginių kintamųjų, aptinkamų aibėje  $S$  skaičių (žymime  $l$ ). Pavyzdžiui, jei  $S = \{\neg p \vee q, \neg p \vee \neg q, p\}$ , tai  $l = 2$ ; jei  $S = \{p, \neg p, p \vee q, p \vee q \vee r\}$ , tai  $l = 3$ .

Indukcijos bazė ( $l = 1$ ). Tuomet  $S$  yra vieno iš pavidalų: a)  $\{p\}$ .

b)  $\{\neg p\}$ , c)  $\{p, \neg p\}$ . Tik atveju (c) aibė prieštaringa ir tik šiuo atveju išvedamas tuščias disjunktas.

Nesunku matyti, kad jei aibėje  $S$  yra disjunktas pavidalo  $p \vee \neg p \vee C$ , tai išbraukę jį iš  $S$ , gausime aibę, kuri prieštaringa tada ir tik tai tada, kai prieštaringa  $S$ . Tariaime, kad tokių disjunktų nagrinėjamoje aibėje nėra.

Tarkime: jei aibėje  $S$  yra  $l < m$  skirtingų loginių kintamųjų ir ji prieštaringa, tai iš jos išvedamas tuščias disjunktas. Parodysime, kad jei aibėje  $S$  yra  $l = m$  skirtingų loginių kintamųjų ir ji prieštaringa, tai iš jos išvedamas tuščias disjunktas.

Tarkime  $p$  yra kuris nors loginis kintamasis tenkinantis sąlygas: a) yra aibėje  $S$  disjunktas, kuriame yra  $\neg p$  įeitis, bei atsiras kitas disjunktas, kuriame yra  $p$  įeitis ir nėra  $\neg p$  įeities. Jei tokio loginio kintamojo neatiras, tai aibė būtų įvykdoma.

Pažymėkime  $S_p$  aibę visų tų disjunktų, kuriuose yra  $p$  įeitis (kartu jai priklauso tuo pačiu ir visi tie disjunktai, kuriuose yra  $\neg p$  įeitis). Suskaidome  $S_p$  į du poaibius. Aibei  $S_p^-$  priklauso visi tie disjunktai, kuriuose pasitaiko  $\neg p$  įeitis, likusieji disjunktai priklauso aibei  $S_p^+$ .  
 $S_{p,-} = S_p^- \cup S_p^+$  ir  $S_p^- \cap S_p^+ = \emptyset$ .

Taikome atžvilgiu  $p$  atkirtos taisyklę, imdami vieną disjunktą iš  $S_p^-$ , o kitą iš  $S_p^+$ . Visų gautų tokiu būdu disjunktų aibę pažymėkime  $at(S_p)$ . Aibės  $at(S_p)$  disjunktuose nėra  $p$  (kartu ir  $\neg p$ ) įeičių. Parodysime, kad aibė  $S$  įvykdoma tada ir tik tai tada, kai įvykdoma

$$(S - S_p) \cup at(S_p). \quad (4.10)$$

1. Tarkime  $S$  įvykdoma. Tuomet visi disjunktai iš  $at(S_p)$  taip pat įvykdomi, nes gauti iš įvykdomų disjunktų, pritaikius atkirtos taisyklę (žiūrėk 4.7 teoremos įrodymą).  $S - S_p$  įvykdoma, kadangi yra įvykdomos aibės poaibis. Be to, abi aibės įvykdomos su viena ir ta pačia interpretacija. Taigi (4.10) įvykdoma.

2. Tarkime aibė (4.10) įvykdoma. Vadinasi, yra interpretacija  $\nu$ , su kuria visi disjunktai iš (4.10) teisingi. Parodysime, kad  $\nu$  galima pratęsti taip, t.y. priskirti kintamajam  $p$  tokią reikšmę, kad būtų įvykdoma  $S_p$ . Kartu su ta pačia interpretacija bus įvykdoma ir  $S$ .

Tegul  $S_p^+ = \{C'_1 \vee p, C'_2 \vee p, \dots, C'_v \vee p\}$ ,

o  $S_p^- = \{C''_1 \vee \neg p, C''_2 \vee \neg p, \dots, C''_r \vee \neg p\}$ .

Tuomet

$$at(S_p) = \left\{ \begin{array}{cccc} C'_1 \vee C''_1, & C'_1 \vee C''_2, & \dots & C'_1 \vee C''_r \\ \dots & \dots & \dots & \dots \\ C'_v \vee C''_1, & C'_v \vee C''_2, & \dots & C'_v \vee C''_r \end{array} \right\}.$$

a) Tarkime, egzistuoja toks  $i$  ( $1 \leq i \leq v$ ), kad  $\nu(C'_i) = k$ . Tuomet  $\nu(C''_j) = t$  ( $j = 1, \dots, r$ ) ir kintamajam  $p$  galime priskirti reikšmę  $t$ .

b) Sakysime, kad su visais  $i$  ( $1 \leq i \leq v$ )  $\nu(C'_i) = t$ . tuomet kintamajam  $p$  priskiriame reikšmę  $k$ .

Gavome, kad aibė  $S$  įvykdoma tada ir tikrai tada, kai įvykdoma (4.10), t.y. T.y. aibė  $S$  prieštaringa tada ir tikrai tada, kai prieštaringa (4.10). Aibė (4.10) gauta iš  $S$  taikant atkirtos taisyklę ir jos disjunktuose aptinkamas ne daugiau kaip  $m - 1$  loginis kintamasis. Jai galioja indukcijos prielaida. Ji prieštaringa tada ir tikrai tada, kai iš jos išvedamas tuščias disjunktas. Teorema įrodyta.

Atkreipiame dėmesį, kad aibės  $\{p \vee C_1, \neg p \vee C_2\}$  ir  $\{C_1 \vee C_2\}$  yra vienu metu abi prieštaringos, arba ne, bet  $(p \vee C_1) \& (\neg p \vee C_2)$  ir  $C_1 \vee C_2$  nėra ekvivalenčios formulės. Pavyzdžiui,  $S' = \{p \vee q, \neg p \vee r\}$ ,  $S'' = \{q \vee r\}$ .  $(p \vee q) \& (\neg p \vee r)$  nėra ekvivalenti formulei  $q \vee r$ , nes su  $p = q = k$ ,  $r = t$ . jų reikšmės skiriasi.

Paaiškinsim, kaip aprašytasis metodas taikomas loginėms išvadoms nustatyti. Klausima, ar formulė  $F$  yra formulių aibės  $\{F_1, \dots, F_n\}$  loginė išvada. Tai, kas duota, įprasta rašyti virš brūkšnio, o išvadą (tikslą, tai, ką reikia įrodyti) – žemiau brūkšnio.

$$\begin{array}{c} F_1 \\ \vdots \\ F_n \\ \hline F \end{array}$$

$F$  yra loginė išvada tada ir tikrai tada, kai

$$(F_1 \& \dots \& F_n) \rightarrow F \quad (4.11)$$

yra tapačiai teisinga formulė. Norime patikrinti, ar (4.11) tapačiai teisinga formulė. Tuo tikslu taikysime *paneigimo metodą*, t.y. tikrinsime, ar (4.11) neigimas yra tapačiai klaidinga formulė.

$$\begin{aligned} \neg((F_1 \& \dots \& F_n) \rightarrow F) &\equiv \\ \neg(\neg(F_1 \& \dots \& F_n) \vee F) &\equiv \\ F_1 \& \dots \& F_n \& \neg F. \end{aligned}$$

Gavome, kad  $F$  yra  $\{F_1, \dots, F_n\}$  loginė išvada tada ir tikrai tada, kai  $\{F_1, \dots, F_n, \neg F\}$  yra prieštaringa aibė, t.y. prie prielaidų aibės reikia prijungti tikslą su neigimu. Transformuojame  $F_1, \dots, F_n, \neg F$  į normaliąsias



Iš  $S$  išvedamas tuščias disjunktas:

$$\frac{j \vee \neg a, \quad a}{j}, \quad \frac{j, \quad \neg j \vee p}{p}, \quad \frac{p, \quad \neg p}{\square}.$$

Taigi, laikantis paskaitos lankymo susitarimo, Petras privalo būti paskaitoje.

QED

QED (quod erat demonstrandum) (lot.) – ką ir reikėjo įrodyti.

Tam tikras privalumas tuščio disjunkto išvedimo paieškai gaunamas, kai nagrinėjamųjų disjunktų aibė susideda tikrai iš *Horno disjunktų*.

**4.13 apibrėžimas.** Disjunktas  $l_1 \vee \dots \vee l_s$  vadinamas *Horno*, jei jame yra ne daugiau kaip viena neigimo įeitis

Pavyzdžiui,  $\neg p \vee q \vee r$ ,  $p \vee q \vee r$ ,  $p \vee \neg q \vee s \vee r$  yra Horno disjunktai, bet  $\neg p \vee \neg q \vee r$  nėra Horno disjunktas.



Skaitmeninė logika

4.13. Horno disjunktai

Šiame skyrelyje pristatysime sekvencinę logiką, kuri yra pagrįsta ryšį. Naudosimės amerikiečių logiko G.Mintso idėjomis, aprašytais 1988 metais.

Nagrinėjame disjunktų aibę  $S = \{C'_1, \dots, C'_s\}$ . Užduotis – patikrinti ar  $S$  prieštaringa. Tikriname dviem skirtingais būdais: a) ar  $C'_1, \dots, C'_s \vdash$  išvedama sekvenciniame skaičiavime, b) ar iš  $S$  išvedamas tuščias disjunktas. Patikslinsime sekvenčinį skaičiavimą bei rezoliucijų metodą, kuriais naudosimės šiame skyrelyje, ir parodysime, kaip pagal išvedimą sekvenciniame skaičiavime randamas pradinę sekvenciją atitinkančio disjunkto išvedimas.



Tarkime,  $S = \{C'_1, \dots, C'_s\}$  yra pradinė disjunktų aibė ir  $p_1, \dots, p_n$  pilnas sąrašas loginių konstantų, t.y.  $\{0, 1\}$ .

**Sekvencinis skaičiavimas.**

*Aksiomos:*  $\Gamma', l, \Gamma'', \neg l, \Gamma'' \vdash$ .

*Išvedimo taisyklė:*

$$(\vee) \frac{\Gamma', l_1, \Gamma'' \vdash \quad \Gamma', l_2, \Gamma'' \vdash \quad \dots \quad \Gamma', l_n, \Gamma'' \vdash}{\Gamma', l_1 \vee \dots \vee l_n, \Gamma'' \vdash}; \quad (4.12)$$

čia  $l_i$  – literos. Be to, tariama, kad  $\neg\neg p$  lygus  $p$  ( $p$  – loginis kintamasis).



Rezoliucijų skaičiavimai

**Aksiomos.**  $C_1, \dots, C_n \vdash p_1 \vee \dots \vee p_n$ , kur  $C_1, \dots, C_n$  – pradinis disjunktas, kuriame gali būti tik literos iš sąrašo  $p_1, \neg p_1, \dots, p_n, \neg p_n$ .

**Išvedimo taisyklė:**

$$\frac{l_1 \vee \dots \vee l_n, \quad \neg l_1 \vee C_1, \dots, \neg l_n \vee C_n}{C_1 \vee \dots \vee C_n};$$

čia  $l_1 \vee \dots \vee l_n$  priklauso pradinei disjunktų aibei  $S$ , t.y. aksioma.

**Pastaba.** Jei būtų įprasta atkirtos taisyklė

$$\frac{p \vee C', \quad \neg p \vee C''}{C' \vee C''},$$

tai reikalavimu, kad viena prielaidų ( $p \vee C', \neg p \vee C''$ ) priklausytų pradinei disjunktų aibei, apibrėžtumėm nepilną skaičiavimą.

Pavyzdžiui, iš  $S = \{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q\}$  nebūtų išvedamas tuščias disjunktas, nors ji ir yra prieštaranga.

Tarkime, turime  $\Gamma \vdash$  išvedimą sekvenciniame skaičiavime. Parodysime kaip jį galime transformuoti į išvedimą rezoliucijų skaičiavime.  $\Gamma$  yra sąrašas formulių, tarp kurių gali būti ir literų. Visų jų aibę pažymėkime raide  $P$ . Tada  $P' = \{p_{i_1}, \dots, p_{i_u}\}$  – pilnas sąrašas skirtingų loginių kintamųjų iš  $P$ , o  $P'' = \{\neg p_{j_1}, \dots, \neg p_{j_v}\}$  – likusios literos.  $P = P' \cup P''$  ir  $P' \cap P'' = \emptyset$ . Sekvencijai  $\Gamma \vdash$  priskiriame disjunktą  $\neg p_{i_1} \vee \dots \vee \neg p_{i_u} \vee p_{j_1} \vee \dots \vee p_{j_v}$  (žymime jį  $\neg P' \vee \neg P''$ ). Jį vadiname *sekvenciją atitinkančiu disjunktą*. Taisyklę (4.12) transformuojame į rezoliucijos taisyklę

$$\frac{l_1 \vee \dots \vee l_n, \quad \neg l_1 \vee \neg P' \vee \neg P'', \dots, \neg l_n \vee \neg P' \vee \neg P''}{\neg P' \vee \neg P''};$$

čia  $\neg P' \vee \neg P''$  yra sekvenciją  $\Gamma', \Gamma'' \vdash$  iš (4.12) atitinkantis disjunktas.

**Pavyzdys.**  $p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q \vdash$ .

Nagrinėsime tik vieną išvedimo šaką. Panašiai nagrinėjama ir antroji (pažymėta išvedime skaičiumi 2).

$$\frac{\frac{\frac{p, q, p, \neg p \vdash \quad p, q, p, \neg q \vdash}{p, q, p, \neg p \vee \neg q \vdash} \quad p, q, \neg q, \neg p \vee \neg q \vdash}{p, q, p \vee \neg q, \neg p \vee \neg q \vdash} \quad 2}{\frac{p, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q \vdash}{p \vee q, \neg p \vee q, p \vee \neg q, \neg p \vee \neg q \vdash}}.$$

Pagal išvedimo medį konstruojame rezoliucijų metodu pradinę sekven-  
ciją atitinkancio disjunkto išvedimą:

$$\begin{array}{c}
 \frac{p \vee q, \quad \frac{\neg p \vee q, \quad \frac{\text{aksioma}}{p \vee \neg p}}{\neg p \vee \neg q}, \quad \frac{p \vee \neg q, \quad \frac{\text{aksioma}}{\neg p \vee \neg p \vee \neg q}, \quad \frac{\text{aksioma}}{q \vee \neg p \vee \neg q}}{\neg q \vee \neg p}}{\neg p} \quad \frac{2}{\neg q} \\
 \hline
 \square
 \end{array}$$

#### 4.7 Pratimai

1. Raskite sekvenčių išvedimus natūraliosios dedukcijos sistemoje:

a)  $\vdash (\neg A \vee \neg B) \rightarrow \neg(A \& B).$

b)  $\vdash (\neg A \& \neg B) \rightarrow \neg(A \vee B),$

c)  $\vdash \neg(A \& B) \rightarrow (\neg A \vee \neg B).$

e)  $\vdash (A \vee B) \rightarrow (B \vee A),$

f)  $B \& (C \rightarrow D), (A \rightarrow B) \rightarrow (B \rightarrow C) \vdash D.$

g)  $(A \rightarrow B) \rightarrow C, \neg(C \vee D), B \vdash.$

h)  $(A \& B) \rightarrow C, A \& \neg C \vdash \neg B,$

i)  $A \rightarrow (B \rightarrow C), A \rightarrow B, A \vdash C.$

j)  $A \rightarrow B, B \rightarrow C, A \vdash C.$

k)  $\vdash (\neg A \rightarrow B) \rightarrow (\neg B \rightarrow A).$

2. Raskite formulinių išvedimus teiginių skaičiavime:

c)  $\neg\neg\neg A \rightarrow A.$

konjunkcines formas. pakeičiame konjunkcijos operacijas kableliais ir gauname disjunktų aibę  $S$ , kuri prieštaringa tada ir tikrai tada, kai  $F$  yra  $\{F_1, \dots, F_n\}$  loginė išvada. Savo ruožtu,  $S$  prieštaringa tada ir tikrai tada, kai iš  $S$  išvedamas tuščias disjunktas.

Taigi, norime nustatyti ar  $F$  yra  $\{F_1, \dots, F_n\}$  loginė išvada. Šią problemą redukuojame į tuščio disjunkto išvedimo iš tam tikros disjunktų aibės uždavinį. Tokį uždavinio sprendimą vadinsime *loginės išvados nustatymu naudojantis rezoliucijų metodu*.

### Pavyzdžiai.

1. Sekmadiniais nedirbama. Šiandien darbo diena. Vadinasi, šiandien nėra sekmadienis.

Pažymėkime:  $s$  – šiandien sekmadienis,  $n$  – šiandien nedarbo diena. Klausima, ar samprotavimas:

$$\frac{s \rightarrow n \quad \neg n}{\neg s}$$

teisingas (pagrįstas), t.y. ar  $\neg s$  yra  $\{s \rightarrow n, \neg n\}$  loginė išvada. Transformuojame pastarąją aibę į disjunktų aibę  $S = \{\neg s \vee n, \neg n, s\}$ . Tikriname, ar  $S \vdash \square$ .

$$\frac{\neg s \vee n, \quad s}{n} \quad \frac{n, \quad \neg n}{\square}$$

Taigi, samprotavimas teisingas.

2. Algis, Jonas ir Petras susitarė dėl paskaitos lankymo tvarkos: a) jei į paskaitą neateina Jonas, tai neateina ir Algis, b) jei į paskaitą ateina Jonas, tai turi ateiti ir Algis su Petru. Klausima, ar šiomis sąlygomis privalo paskaitoje dalyvauti Petras, kai žinoma, kad joje yra Algis?

Pažymėkime:  $a$  – paskaitoje yra Algis,  $j$  – paskaitoje yra Jonas,  $p$  – paskaitoje yra Petras.

Tuomet užduotis užrašoma taip:

$$\frac{\neg j \rightarrow \neg a \quad j \rightarrow (a \& p)}{a \rightarrow p}$$

Tikriname, ar aibė  $\{\neg j \rightarrow \neg a, j \rightarrow (a \& p), \neg(a \rightarrow p)\}$  prieštaringa? Transformuojame į disjunktų aibę  $S = \{j \vee \neg a, \neg j \vee a, \neg j \vee p, a, \neg p\}$ .

3. Taikydami dedukcijos teorema raskite formulių išvedimus teiginių skaičiavime:

a)  $(A \rightarrow B) \rightarrow ((C \vee A) \rightarrow (C \vee B)).$

b)  $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \& B) \rightarrow C).$

c)  $(A \rightarrow B) \rightarrow ((C \& A) \rightarrow (D \vee B)).$

d)  $(A \rightarrow B) \rightarrow ((A \& C) \rightarrow (B \& C)).$

4. Raskite 1 uždavinio sekvencijų išvedimus sekvenciniame skaičiavime  $G$ .

5. Raskite antisekvencijų išvedimus skaičiavime  $\bar{G}$ :

a)  $p \rightarrow q, \neg q \vee r \dashv r \rightarrow p,$

b)  $(p \& q) \rightarrow r, (p \vee r) \rightarrow q \dashv (q \vee r) \rightarrow p,$

c)  $p \vee q, \neg p \vee r, r \vee \neg q \dashv p \& \neg r.$

6. Ar išvedamas tuščias disjunktas iš aibės:

a)  $\{p \vee q, \neg p \vee \neg q\},$

b)  $\{r \vee \neg s \vee \neg u, \neg p \vee q \vee \neg u, p \vee \neg u \vee \neg s, \neg q \vee \neg r \vee p \vee \neg s, s, \neg s \vee u, \neg s \vee q, \neg p\}?$

7. Rezoliucijų metodu patikrinkite, ar formulė tapati klaidinga:

a)  $(\neg p \vee q) \& \neg((q \rightarrow r) \rightarrow (\neg p \vee r)).$

b)  $((p \rightarrow q) \rightarrow r) \& (\neg(r \vee s) \& q).$