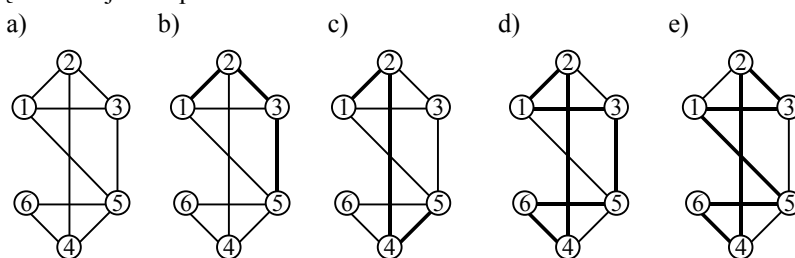


Panašų trūkumą turi ir **Ore (O.Ore) teorema, teigianti, jog jungusis grafas, turintis ne mažiau trijų viršūnių ( $p \geq 3$ ), kuriame kiekviena pora tiesiogiai nesurištų viršūnių „u“ ir „v“ tenkina sąlygą  $\deg u + \deg v \geq p$ , turi Hamiltono ciklą**. Šis požymis taip pat yra pakankamas, bet nėra būtinas Hamiltono ciklo egzistavimo požymis, nes egzistuoja daug grafų, netenkinančių Ore sąlygos, bet turinčių Hamiltono ciklą.

Truputį stipresnė yra Dirako ir Ore teoremas apibendrinanti **vengrų matematiko L. Poša (L. Posa) teorema**, kurios šiek tiek gremėzdišką formuluotę ir įrodymą skaitytojas gali rasti knygoje [98], tačiau ir ši teorema pateikia tik pakankamą, tačiau dar nebūtiną Hamiltono ciklo egzistavimo požymį. Egzistuoja teoremos, pateikiančios būtinus, bet nepakankamus tokio ciklo egzistavimo požymius, tačiau net sujungę visas tokias teoremas, pateikiančias būtinus Hamiltono ciklo egzistavimo požymius, su teoremomis, pateikiančiomis pakankamas tokių ciklų egzistavimo sąlygas, mes ne visuomet sugebėsime atsakyti į klausimą, egzistuoja duotajame grafe Hamiltono ciklas ar ne. Tokį atsakymą mes sugebėsime rasti tik tiesiogiai ištyrę duotąjį grafą, tačiau tai reikalauja nemažai laiko ir nėra lengva, ypač, kai grafas turi daug viršūnių ir briaunų, bet neturi Hamiltono ciklo.

Net ir tada, kai mes žinome, jog Hamiltono ciklas egzistuoja, surasti jį ne visada paprasta, nes, pasirinkdami grafo viršūnes ir jungdami jas į ciklą, galime susidurti su situacija, kada dėl nepakankamai toliaregiško pasirinkimo pradžioje vėliau tęsti Hamiltono ciklo sudarymą pasidaro jau nebeįmanoma. Tokią situaciją iliustruoja 143 pav.



143 pav.

Tarkime, jog duotas grafas, kuris pavaizduotas 143 „a“ pav. Kaip nesunku įsitikinti, jis turi Hamiltono ciklą.

Tarkime, jog mes pasirinkome pradinę viršūnę „1“. Taip visuomet galima padaryti, žinant, kad į Hamiltono ciklą įeis kiekviena viršūnė, taigi ir viršūnė „1“, ir visai nesvarbu, nuo kurios ciklo vietos pradėti ciklo apėjimą.

Tarkime, kad mes pradžioje pasirinkome briauną ( $1 \leftrightarrow 2$ ) ir patekome į viršūnę „2“, po to – briauną ( $2 \leftrightarrow 3$ ) ir patekome į viršūnę „3“ (žiūr. 143 „b“ pav.).

Briaunos ( $2 \leftrightarrow 3$ ) pasirinkimas mus jau atvedė į situaciją, kurioje Hamiltono ciklą sudaryti nebeįmanoma. Įsitikinsime, kad taip ir yra.

Iš viršūnės „3“ mes galime eiti tik viršūnės „5“ link, nes eidami į viršūnę „1“, uždarytume ciklą, sudarytą ne iš visų, o tik iš trijų viršūnių.

Patekę į viršūnę „5“ (žiūr. 143 „b“ pav.), mes negalime eiti viršūnės „1“ link, nes uždarytume nepilną ciklą. Būtinai eiti tik į viršūnę „4“ arba į viršūnę „6“. Kitaip būtų, mes jau negalėsime uždaryti pilno ciklo, nepatekę į viršūnę „5“ arba į viršūnę „2“, kur mes jau buvome ir kur (pagal Hamiltono ciklo apibrėžimą) neturime teisės patekti. Taigi, situacijoje, kurią vaizduoja 143 „b“ pav., Hamiltono ciklo sudaryti jau nebeįmanoma.

Į analogišką situaciją būtume patekę, jei iš viršūnės „2“ būtume ėję ne į viršūnę „3“, o į viršūnę „4“, ir po to – į viršūnę „5“ (žiūr. 143 „c“ pav.).

Jei iš viršūnės „2“, būtume nuėję viršūnės „4“ link, o po to – į viršūnę „6“, tai būtume pasiekę situaciją, kurioje sudaryti Hamiltono ciklą jau visai nesunku. Būtinai iš viršūnės „6“ patekti į viršūnę „5“, iš jos – į viršūnę „1“ ir uždaryti pilną ciklą (žiūr. 143 „d“ pav.).

Hamiltono ciklą būtume gavę ir tuo atveju, jei iš viršūnės „1“ pačioje pradžioje būtume nuėję ne į viršūnę 2, o į viršūnę „3“ ir tik po to į viršūnę „2“. Iš tos viršūnės, eidami maršrutu  $2 \rightarrow 4 \rightarrow 6 \rightarrow 5 \rightarrow 1$ , būtume uždarę Hamiltono ciklą  $\boxed{1 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 6 \rightarrow 5}$  (žiūr. 143 „e“ pav.).

Jei iš viršūnės „1“ būtume iškart nuėję viršūnės „5“ link, tai iškart būtume patekę į situaciją, kurioje sudaryti Hamiltono ciklo jau nebeįmanoma.

Tokias situacijas – „aklavietas“ – nesunku pastebėti, kai grafas yra nedidelis ir nesudėtingas. Žymiai sunkiau susigaudyti dideliuose ir sudėtinguose grafuose. Tokiais atvejais galėtų padėti Hamiltono grafo egzistavimo pakankamas ir būtinas požymis, jei jis būtų mums žinomas. Tuomet, pasirinkę kokią nors grafo viršūnę, stengtumės iš tos viršūnės išeinančią briauną pasirinkti taip, kad, tą briauną eliminavus, liktų grafas, tenkinantis Hamiltono grafo egzistavimo būtiną ir pakankamą sąlygą.

Kai grafas tenkina Dirako, Ore ar Poša požymį, tai tuo kartais irgi galima pasinaudoti, tačiau Dirako, Ore bei Poša kriterijai, nėra būtini Hamiltono ciklo egzistavimui ir per daug sugriežtina grafui keliamus reikalavimus.

Žodžiu, palieku skaitytojui pabandyti sukurti Hamiltono grafo sudarymo algoritmą, panašų į Flerio algoritmą, kuris nuosekliai (žingsnis po žingsnio) be grįžimų atgal vestų prie Hamiltono ciklo sudarymo arba parodytų, kad toks ciklas egzistuoti negali. Jei nepavyktų sudaryti algoritmo be sugrįžimų atgal, tai reiktų pabandyti sukurti bent jau tokį algoritmą, kuris leistų iš kiekvieno sugrįžimo „išspausti“ kuo daugiau naudingos informacijos algoritmo „apmokymui“, kad tolesnėje Hamiltono grafo paieškoje duotajame grafe tokių sugrįžimų skaičius būtų vis mažesnis ir mažesnis. Tokio algoritmo paieška bei jo konstravimas net ir tuo atveju, kai tokio algoritmo sukurti nepavyktų, duotų skaitytojui nemažą naudą ir prisidėtų prie jo kūrybinio mąstymo vystymosi, kas labai svarbu būsimam sistemotyrininkui.

Antrasis šio skyrelio uždavinys – **Hamiltono optimalaus ciklo paieška**, t. y. suradimas duotajame grafe tokio Hamiltono ciklo, kurio lankų svorių  $c_{ij}$  suma būtų minimali.

Atrodytų, kad antrasis šio skyrelio uždavinys yra pirmojo uždavinio tęsinys: sprendami pirmąjį uždavinį, surandame visus duotojo grafo Hamiltono ciklus, sprendami antrąjį – išrenkame iš jau gautos Hamiltono ciklų aibės tik tą (arba tik tuos) Hamiltono ciklą (ciklus), kuriam (kuriems) yra būdingas lankų svorių sumos minimumas.

Kai kuriais atvejais taip ir būna. Pavyzdžiui, taip darome, kai antrąjį Hamiltono uždavinį sprendžiame taip vadinamu „**jėgos algoritmu**“, kurį taikydami perrenkame visus duotojo grafo Hamiltono ciklų variantus, apskaičiuodami bei palygindami Hamiltono ciklų lankų svorių sumas ir išrinkdami tuos ciklus, kuriems minėtosios sumos yra mažiausios.

Tačiau „jėgos“ (arba variantų perrinkimo) algoritmo taikymas yra gana ribotas, nes didėjant duotojo grafo viršūnių skaičiui, daugumoje atvejų galimų Hamiltono ciklų variantų skaičius didėja milžiniškais tempais ir visų variantų perrinkimas tampa praktiškai nerealizuojamu dalyku.

Pavyzdžiui, pilname grafe  $K_p$ , turinčiame  $p$  viršūnių, egzistuoja  $(p-1)!$  Hamiltono ciklų. Tai nesunku įrodyti.

1. Pasirenkame bet kurią pilnojo grafo  $K_p$  viršūnę, pavyzdžiui, pirmąją, kaip konstruojamą Hamiltono ciklų apėjimo pradinį tašką, kuris gali būti bet kurioje ciklo vietoje.

2. Antrajame žingsnyje yra galimybė pasirinkti bet kurią iš  $p-1$  likusių viršūnių, t. y. galimybė pradėti sudarinėti  $p-1$  skirtingų Hamiltono ciklų. Pasirenkame vieną iš galimų  $p-1$  variantų, atsimindami, kad tai – tik vienas iš  $p-1$  galimų.

3. Trečiajame žingsnyje yra galimybė pasirinkti bet kurią iš likusių  $p-2$  viršūnių, t. y. galimybė iš kiekvieno varianto, sudaryto antrajame žingsnyje, gauti po  $p-2$  variantų trečiajame žingsnyje, taigi, iš viso jau  $(p-1)(p-2)$  variantų.

4. Ketvirtajame žingsnyje yra galimybė pasirinkti iš likusiųjų  $p-3$  viršūnių, t. y. galimybė pasirinkti vieną iš  $(p-1)(p-2)(p-3)$  variantų.

5. Penktajame žingsnyje yra galimybė pasirinkti vieną iš  $(p-1)(p-2)(p-3)(p-4)$  galimų Hamiltono ciklo pilnajame grafe  $K_p$  variantų ir t. t.

Tęsdami šią procedūrą,  $p$  – jame žingsnyje įsitikinsime, jog pilnajame grafe  $K_p$  egzistuoja  $(p-1)!$  Hamiltono ciklų. Jei pilnojo grafo tiesioginių ryšių matrica yra simetriška, t. y., jei  $\forall i \forall j (c_{ij}=c_{ji})$ , tai kiekvienam Hamiltono ciklui pilnajame grafe būtinai atsiras simetriškas ciklas, besiskiriantis nuo pirmojo tik pilnojo grafo viršūnių apėjimo kryptimi (vienas – laikrodžio rodyklės judėjimo kryptimi, kitas – prieš laikrodžio rodyklę). Todėl šiuo atveju, išrenkant optimalųjį Hamiltono ciklą, naudojantis „jėgos algoritmu“ tektų peržiūrėti ir palyginti tik  $\frac{(p-1)!}{2}$  Hamiltono ciklų. Tačiau ir tas skaičius,

didėjant  $p$ , labai greitai auga. Pavyzdžiui, kai  $p=8$ , tektų peržiūrėti jau daugiau nei 2,5 tūkstančio Hamiltono ciklų, kai  $p=10$ , daugiau nei 180 tūkstančių, o kai  $p=20$ , tai daugiau nei  $6 \cdot 10^{16}$  Hamiltono ciklų! Tokiu atveju naudojantis personaliniu kompiuteriu, galinčiu išanalizuoti 10 000 Hamiltono ciklų per sekundę, prireiktų arti 200 tūkstančių metų! Ką jau bekalbėti apie pilnąjį grafą, turintį 100 viršūnių, kuriame optimalus Hamiltono ciklo išrinkimui personaliniu kompiuteriu, naudojantis „jėgos“ algoritmu, prireiktų apie  $1,5 \cdot 10^{144}$  metų! Nedaug ką išloštume ir pasinaudodami superkompiuteriu, sugebančiu per sekundę išanalizuoti milijoną Hamiltono ciklų: optimalus ciklo išrinkimui simetriškame  $K_{100}$  prireiktų apie  $1,2 \cdot 10^{142}$  metų, kas nepalyginamai daugiau negu mūsų Visatos amžius, kuris laikomas apytikriai lygiu 20 milijardų (t. y.  $2 \cdot 10^{10}$ ) metų. O juk praktikoje sutinkami tinklai gali turėti ne tik šimtą, bet ir žymiai daugiau mazgų. Taigi, visų galimų variantų perrinkimo algoritmo taikymas praktikoje yra gana ribotas.

Perrenkamų ir lyginamų vienas su kitu variantų skaičių daugelyje atvejų pavyksta žymiai sumažinti, tuo pačiu sumažinant ir Hamiltono optimalaus ciklo radimo laiką, naudojantis **šakų ir ribų metodu**.

Šio metodo esmė yra tame, kad visų galimų Hamiltono ciklų aibė žingsnis po žingsnio dekomponuojama į poaibius – tos aibės dekompozicijos medžio **šakas** – ir kiekviename žingsnyje gauti poaibiai įvertinami, nustatant juose esantiems Hamiltono ciklams jų ilgių garantuotas negriežtas žemutines **ribas**, t. y. tokius dydžius, kurie gali būti tik mažesni už poaibyje esančių Hamiltono ciklų ilgius arba lygūs jiems. Po tokio įvertinimo tolesnei dekompozicijai pasirenkama ta šaka, kurioje ta žemutinė riba yra mažiausia ir kurioje todėl, matyt, dažniausiai galėtų būti optimalus Hamiltono ciklas.

Šakų ir ribų metodas galėtų būti gana efektyvus, jei žemutines ribas pavyktų nustatyti tiksliai (arba beveik tiksliai) ir todėl jau nebereiktų laiko nuo laiko, įvertinus eilinės dekompozicijos metu gautų šakų žemutines ribas, grįžti į dekompozicijos medyje aukščiau esančias ir anksčiau laikinai atmestas šakas, kurių žemutinės ribos, tikslinant dekompozicijos metu gaunamų šakų žemutines ribas, pasirodo, esančios mažesnės už vėliau gautų šakų ribas. Kai tokių grįžimų skaičius būna didelis, šakų ir ribų metodas tampa nebeefektyvus. Tačiau daugelyje atvejų tokių sugrįžimų iš žemiau esančių šakų į aukščiau esančias šakas nėra labai daug, ir tokiais atvejais gana greitai surandamas optimalus Hamiltono ciklas. Tik gaila, kad iki šiol dar nerastas požymis, kuris leistų, dar prieš pradedant naudotis šakų ir ribų metodu, nustatyti, ar tas metodas šiuo konkrečiu atveju bus pakankamai efektyvus, ar reikia ieškoti kokio nors kito metodo.

Naudojantis šakų ir ribų metodu, tenka spręsti dar vieną problemą: kokių požymių reikėtų naudotis, dekomponuojant Hamiltono ciklų aibę į poaibius, o šiuos – į dar smulkesnius poaibius ir t. t. Gana dažnai naudojamosi **baudos funkcijų metodu**: apskaičiuojama ta minimali bauda, kurią (ar dar didesnę) būtina tektų mokėti, jei atsisakytume įtraukti į Hamiltono ciklą tą ar kitą duotojo grafo linką, ir pradedama nuo tokio poaibio išskyrimo ir nagrinėjimo, kurį sudaro Hamiltono ciklai, turintys savo sudėtyje tą linką, už kurio neįtraukimą į Hamiltono ciklą žemutinė baudos riba būtų didžiausia. Kitaip tariant, renkamasi tie poaibiai, kurių pasirinkimas leidžia išvengti didžiausių baudų.

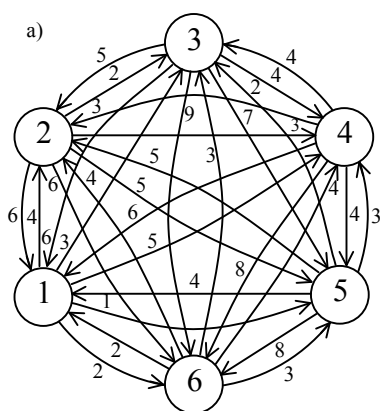
Tokia šakų ir ribų metodo sintezė su baudos funkcijų metodu realizuota **Litlo** (J. D. C. Little) **algoritme**.

Jo taikymą, ieškant Hamiltono optimalaus ciklo pilname grafe, pailiustruosiu konkrečiu pavyzdžiu.

Tarkime, jog turime nesimetrišką grafą  $K_6$ , pavaizduotą 144 „a“ pav., ir norime surasti jame optimalų Hamiltono ciklą. Kadangi grafas nesimetriškas, tai, naudojantis „jėgos“ algoritmu, tektų apskaičiuoti ir palyginti  $5! = 120$  Hamiltono ciklų ilgius. Naudojantis Litlo algoritmu darbo apimtį galime žymiai sumažinti.

Sudarysime duotojo grafo  $K_6$  matricinį modelį (žiūr. 144 „b“ pav.), įrašydami į matricos pagrindinę įstrižainę begalybės ženklus  $\infty$ , tuo apsidrausdami nuo nepilnų ciklų, vadinamų **kilpomis** (kuomet elementas „i“ ryšiu  $i \leftrightarrow i$  susijungia pats su savimi), atsiradimo.

Turėdami duotojo grafo matricinį modelį – matricą  $\|c_{ij}\|$ , pavaizduotą 144 „b“ pav., galime sudaryti **Hamiltono optimalaus ciklo paieškos duotajame grafe uždavinio analizinį matematinį modelį**. Tas modelis bus reikalingas kai kurių teoremų, kuriomis mes naudosimės sprendami šį uždavinį, įrodymui.



b)

| $i \setminus j$ | 1        | 2        | 3        | 4        | 5        | 6        | $u_i$ |
|-----------------|----------|----------|----------|----------|----------|----------|-------|
| 1               | $\infty$ | 4        | 3        | 5        | 1        | 2        | 1     |
| 2               | 6        | $\infty$ | 2        | 3        | 5        | 6        | 2     |
| 3               | 6        | 5        | $\infty$ | 4        | 7        | 9        | 4     |
| 4               | 6        | 3        | 4        | $\infty$ | 4        | 8        | 3     |
| 5               | 4        | 5        | 2        | 3        | $\infty$ | 8        | 2     |
| 6               | 2        | 4        | 3        | 4        | 3        | $\infty$ | 2     |

$\sum_{i=1}^6 u_i = 14$

c)

| $i \setminus j$ | 1        | 2        | 3        | 4        | 5        | 6        | $v_j$ |
|-----------------|----------|----------|----------|----------|----------|----------|-------|
| 1               | $\infty$ | 3        | 2        | 4        | 0        | 1        | 1     |
| 2               | 4        | $\infty$ | 0        | 1        | 3        | 4        | 2     |
| 3               | 2        | 1        | $\infty$ | 0        | 3        | 5        | 4     |
| 4               | 3        | 0        | 1        | $\infty$ | 1        | 5        | 3     |
| 5               | 2        | 3        | 0        | 1        | $\infty$ | 6        | 2     |
| 6               | 0        | 2        | 1        | 2        | 1        | $\infty$ | 2     |

$\sum_{j=1}^6 v_j = 1$

d)

| $i \setminus j$ | 1        | 2        | 3        | 4        | 5        | 6        |
|-----------------|----------|----------|----------|----------|----------|----------|
| 1               | $\infty$ | 3        | 2        | 4        | $0^1$    | $0^3$    |
| 2               | 4        | $\infty$ | $0^1$    | 1        | 3        | 3        |
| 3               | 2        | 1        | $\infty$ | $0^2$    | 3        | 4        |
| 4               | 3        | $0^2$    | 1        | $\infty$ | 1        | 4        |
| 5               | 2        | 3        | $0^1$    | 1        | $\infty$ | 5        |
| 6               | $0^3$    | 2        | 1        | 2        | 1        | $\infty$ |

e)

| $i \setminus j$ | 1        | 2        | 3        | 4        | 5        |
|-----------------|----------|----------|----------|----------|----------|
| 2               | 4        | $\infty$ | 0        | 1        | 3        |
| 3               | 2        | 1        | $\infty$ | 0        | 3        |
| 4               | 3        | 0        | 1        | $\infty$ | 1        |
| 5               | 2        | 3        | 0        | 1        | $\infty$ |
| 6               | $\infty$ | 2        | 1        | 2        | 1        |

2                      1

144 pav.

Įvesime loginius kintamuosius  $x_{ij}$ , sudarančius matricą  $\|x_{ij}\|$ , turinčią tą pačią eilę  $p$ , kaip ir duotoji matrica  $\|c_{ij}\|$ . Priimsime, jog kintamasis  $x_{ij}$  priima reikšmę, lygią 1, tuo ir tik tuo atveju, kai

lankas ( $i \rightarrow j$ ) patenka į Hamiltono ciklą. Kai lankas ( $i \rightarrow j$ ) nepatenka į Hamiltono ciklą, tai atitinkamą kintamąjį  $x_{ij}$  laikysime lygiu 0.

Kadangi į Hamiltono ciklą įeina kiekviena duotojo grafo viršūnė „i“, ir įeina tik po vieną kartą, o kartu su ta viršūne į Hamiltono ciklą įeina vienas ir tik vienas iš tos viršūnės „i“ išeinantis lankas (tarkime, jog tai – lankas ( $i \rightarrow j$ )), tai matricos  $\|x_{ij}\|$  kiekvienoje eilutėje „i“ atsiras vienas ir tik vienas langelis, kurio turinys bus lygus 1, o visi kiti tos eilutės langeliai bus lygūs 0. Todėl kintamieji  $x_{ij}$  turės

tenkinti  $p$  lygybių  $\forall i \left( \sum_{j=1}^p x_{ij} = 1 \right)$ , kur  $i=1, 2, \dots, p$ .

Kadangi į kiekvieną grafo viršūnę „j“, Hamiltono cikle būtinai įeina koks nors vienas ir tik vienas lankas ( $i \rightarrow j$ ), tai kiekviena matricos  $\|x_{ij}\|$  stulpelyje „j“ būtinai bus viena ir tik viena kintamojo  $x_{ij}$  reikšmė lygi 1, o visos kitos  $x_{ij}$  reikšmės tame stulpelyje turės būti lygios 0. Todėl kintamieji  $x_{ij}$  turės

tenkinti dar ir kitą lygybių sistemą  $\forall j \left( \sum_{i=1}^p x_{ij} = 1 \right)$ , susidedančią iš  $p$  lygybių, gaunamų keičiant „j“ nuo

1 iki  $p$ .

Kad kintamieji  $x_{ij}$  galėtų turėti tik dvi reikšmes – 0 ir 1, jie turės tenkinti dar ir  $p^2$  lygybių sistemą  $\forall i \forall j (x_{ij}^2 - x_{ij} = 0)$ .

Visas tas  $p^2 + 2p$  lygybes, vadinsime sprendžiamojo uždavinio **leistinų sprendinių srities (LSS) apribojimais**. Tuos apribojimus turės tenkinti bet kuri kintamųjų  $x_{ij}$  reikšmių aibė, aprašanti kokį nors Hamiltono ciklą.

Kad tas ciklas būtų optimalus, į tą ciklą įeinančių duotojo grafo lankų ( $i \rightarrow j$ ) svorių  $c_{ij}$  suma turi būti minimali. Tą sumą, atsižvelgdami į tai, jog kai  $x_{ij}=0$ , tai ir  $c_{ij}x_{ij}=0$ , galime išreikšti formule

$F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$  ir ieškoti tokių  $x_{ij}$  reikšmių matricos  $\|x_{ij}^*\|$ , sudarytos iš nulių ir vienetų, kad dydžiai  $x_{ij}^*$

tenkintų apribojimus  $\forall i \left( \sum_{j=1}^p x_{ij}^* = 1 \right)$ ,  $\forall j \left( \sum_{i=1}^p x_{ij}^* = 1 \right)$ ,  $\forall i \forall j \left[ (x_{ij}^*)^2 - x_{ij} = 0 \right]$  ir minimizuotą funkciją

$F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$ , vadinamą **optimizavimo uždavinio tikslo funkcija**. Kitaip tariant, ieškosime

kintamųjų  $x_{ij}$  tokių reikšmių  $x_{ij}^*$ , kurios, tenkindamos aukščiau suformuluotus apribojimus, tenkintų dar

ir sąlygą  $\forall i \forall j \left[ \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}^* \leq \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij} \right]$ .

Šio uždavinio sprendinys  $\|x_{ij}^*\|$ , suteikiantis tikslo funkcijai  $F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$  minimalią reikšmę

$F^* = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}^*$ , gali būti ne vienas, t. y. duotasis grafas gali turėti kelis optimalius Hamiltono ciklus,

sudarytus iš lankų ( $i \rightarrow j$ ), kurių  $x_{ij}=x_{ij}^*=1$ , tačiau tas optimalus sprendinys būtinai egzistuoja, jei duotasis grafas turi bent vieną Hamiltono ciklą, t. y., jei pirmasis šio skyrelio uždavinys duotajam grafui turi sprendinį.

Galima įrodyti teoremą, kad jei iš matricos  $\|c_{ij}\|$  bet kurios eilutės „i“ kiekvieno elemento  $c_{ij}$  atimtume (arba pridėtume) kokį nors pastovų visiems tos eilutės elementams skaičių  $u_i$ , tai ką tik suformuluoto uždavinio sprendinys  $\|x_{ij}^*\|$  nepasikeistų, t. y. tie Hamiltono ciklai, kurie buvo optimalūs grafui, turinčiam tiesioginių ryšių matricą  $\|c_{ij}\|$ , išliktų optimalūs ir grafui, turinčiam tiesioginių ryšių matricą  $\|\tilde{c}_{ij}\| = \|c_{ij} - u_i\|$ .

Galėtų pasikeisti tik optimali tikslo funkcijos  $F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$  reikšmė, tapdama

lygi  $\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}^* - \sum_{i=1}^p \sum_{j=1}^p u_i x_{ij}^* = F^* - \sum_{i=1}^p u_i \sum_{j=1}^p x_{ij}^* = F^* - \sum_{i=1}^p u_i$ , nes  $\sum_{j=1}^p x_{ij}^* = 1$ .

Taip pat galima įrodyti analogišką teoremą stulpelių „j“ atžvilgiu: jei iš matricos  $\|c_{ij}\|$  bet kurio stulpelio „j“ kiekvieno elemento  $c_{ij}$  atimtume (arba pridėtume) kokį nors pastovų visiems to stulpelio elementams skaičių  $v_j$  ir gautume matricą  $\|\tilde{c}_{ij}\| = \|c_{ij} - v_j\|$ , tai visi tie Hamiltono ciklai, kurie buvo optimalūs grafui, turinčiam tiesioginių ryšių matricą  $\|c_{ij}\|$ , išliktų optimalūs ir grafui, turinčiam tiesioginių ryšių matricą  $\|\tilde{c}_{ij}\| = \|c_{ij} - v_j\|$ . Galėtų pasikeisti tik tikslo funkcijos optimali reikšmė, tapdama

$$\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = \sum_{j=1}^p \sum_{i=1}^p (c_{ij} - v_j) x_{ij}^* = \sum_{j=1}^p \sum_{i=1}^p c_{ij} x_{ij}^* - \sum_{j=1}^p \sum_{i=1}^p v_j x_{ij}^* = F^* - \sum_{j=1}^p v_j \sum_{i=1}^p x_{ij} = F^* - \sum_{j=1}^p v_j. \quad \text{Ap-}$$

skaičiuodami funkcijos  $\tilde{F}^*$  reikšmę, mes pasinaudojome sumos asociatyvumo savybe  $\sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij} =$

$$= \sum_{j=1}^p \sum_{i=1}^p \tilde{c}_{ij} x_{ij} \text{ ir jos distributyvumo savybe } \sum_{j=1}^p \sum_{i=1}^p x_{ij} v_j = \sum_{j=1}^p v_j \sum_{i=1}^p x_{ij} \text{ bei lygybe } \sum_{j=1}^p x_{ij} = 1.$$

Abi tas teoremas apie skaičių  $u_i$  ir  $v_j$  atėmimą iš matricos eilučių bei stulpelių galime sujungti į vieną, ką mes ir padarysime, sąlyginai pavadindami tokią apibendrintą teoremą **pirmąja šio skyrelio teorema – optimalaus Hamiltono ciklo paieškos supaprastinimo teorema.**

Įrodysiu šią teoremą, t. y., įrodysiu, kad **funkcijos**  $F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$  **ir**  $\tilde{F} = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij} =$   
 $= \sum_{i=1}^p \sum_{j=1}^p (c_{ij} - u_i - v_j) x_{ij}$  **pasiekia minimumą, esant toms pačioms kintamųjų  $x_{ij}$  reikšmėms,**  
sudarantioms matricą  $\|x_{ij}^*\|$ .

Tarkime, kad  $\|x_{ij}^*\|$  minimizuoja tikslo funkciją  $F = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}$ , t. y., kad skaičius

$$F^* = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}^* \text{ yra minimali tos funkcijos reikšmė.}$$

Įrodysiu, kad skaičius  $\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = \sum_{i=1}^p \sum_{j=1}^p (c_{ij} - u_i - v_j) x_{ij}^*$  yra minimali funkcijos

$$\tilde{F} = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij} \text{ reikšmė.}$$

$$\tilde{F} = \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij} - \sum_{i=1}^p \sum_{j=1}^p u_i x_{ij} - \sum_{i=1}^p \sum_{j=1}^p v_j x_{ij} = F - \sum_{i=1}^p u_i \sum_{j=1}^p x_{ij} - \sum_{j=1}^p v_j \sum_{i=1}^p x_{ij} = F - \sum_{i=1}^p u_i - \sum_{j=1}^p v_j.$$

Kadangi dydžiai  $\sum_{i=1}^p u_i$  ir  $\sum_{j=1}^p v_j$  yra konstantos, nepriklausančios nuo  $x_{ij}$ , tai funkcijos  $\tilde{F}$  ir  $F$

pasiekia minimumą, esant toms pačioms kintamųjų  $x_{ij}$  reikšmėms:  $\min \tilde{F} = \min F - \sum_{i=1}^p u_i - \sum_{j=1}^p v_j =$

$$= \sum_{i=1}^p \sum_{j=1}^p c_{ij} x_{ij}^* - \sum_{i=1}^p u_i - \sum_{j=1}^p v_j, \text{ t. y. funkcija } \tilde{F} \text{ pasiekia minimumą, kai } \forall i \forall j (x_{ij} = x_{ij}^*), \text{ ką ir reikėjo įrodyti.}$$

Vadovaudamiesi ką tik įrodyta teorema, mes galime supaprastinti optimalaus ciklo paiešką, pereidami nuo grafo, turinčio tiesioginių ryšių matricą  $\|c_{ij}\|$ , prie grafo, turinčio paprastesnę (pavyzdžiui, turinčią mažesnes  $c_{ij}$  reikšmes, tame tarpe ir nulines) tiesioginių ryšių matricą  $\|\tilde{c}_{ij}\|$  (arba  $\tilde{\tilde{c}}_{ij}$ ). Tai ir padaryta 144 pav., pradžioje pereinant nuo matricos „b“ prie matricos „c“, o po to – nuo matricos „c“ prie matricos „d“. Tuose perėjimuose  $u_{ij} = \min_i c_{ij}$  ir  $v_j = \min_j \tilde{c}_{ij}$ . Toks konstantų  $u_i$  ir  $v_j$  pasirinkimas pagrįstas tuo, kad norima gauti matricą  $\|\tilde{c}_{ij}\| = \|c_{ij} - u_i - v_j\|$ , turinčią kuo daugiau nulių ir neturinčią neigiamų elementų. Pereinant nuo matricos „b“ prie matricos „d“, tikslo funkcija (Hamiltono ciklą

sudarančių lankų svorių suma) sumažėja dydžiu  $r_0 = \sum_{i=1}^p u_i + \sum_{j=1}^p v_j = 14 + 1 = 15$  vienetų.

Jei tokiu būdu mums pavyktų gauti tokią matricą  $\|\tilde{c}_{ij}\|$ , kuri turėtų bent po vieną nulį kiekvienoje eilutėje ir kiekviename stulpelyje, išsidėsčiusius taip, kad iš „nolinių“ lankų būtų galima sudaryti Hamiltono ciklą, tai, pasirinkę kintamiesiems  $x_{ij}$  tokias reikšmes  $x_{ij}^*$ , kad  $x_{ij}^*$  būtų lygūs 0 visuose matricos  $\|\tilde{c}_{ij}\|$  langeliuose, išskyrus po vieną kiekvienoje eilutėje ir kiekviename stulpelyje (kuriuose  $x_{ij}^*=1$  kaip tik tuose langeliuose, kuriuose  $\tilde{c}_{ij}=0$ ), gautume, jog  $\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = 0$ . Taip atsitiktų todėl, kad tuose langeliuose, kuriuose  $x_{ij}^*=1$ , turėtume  $\tilde{c}_{ij}=0$ , o visuose kituose langeliuose, kuriuose  $x_{ij}^*=0$ , sandaugos  $\tilde{c}_{ij} x_{ij}^*$  būtų lygios 0 nepriklausomai nuo  $\tilde{c}_{ij}$  reikšmės.

Kadangi, kai  $\forall i \forall j (\tilde{c}_{ij} \geq 0)$ , funkcija  $\tilde{F} = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}$ , kurios argumentai  $x_{ij}$  gali būti lygūs tik 0

arba 1, negali tapti neigiama, tai tos funkcijos reikšmė  $\tilde{F}^* = 0$  yra minimali jos reikšmė. Iš šio teiginio išplaukia, jog **jei egzistotų tokia kintamųjų  $x_{ij}$  reikšmių matrica  $\|x_{ij}^*\|$ , kuri tenkintų lygybę**

**$\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = 0$ , tai ta matrica  $\|x_{ij}^*\|$  vaizduotų optimalų Hamiltono ciklą grafe, turinčiame**

tiesioginių ryšių matricą  $\|\tilde{c}_{ij}\|$ . Šią išvadą sąlyginai pavadinsime **antrąja šio skyrelio teorema – teorema apie Hamiltono ciklo optimalumo požymį**.

Iš pirmosios ir antrosios teoremų išplaukia išvada, jog matrica  $\|x_{ij}^*\|$ , tenkinanti lygybę

$\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = 0$ , kur  $x_{ij}$  tenkina apribojimus  $\forall i \left( \sum_{j=1}^p x_{ij}^* = 1 \right)$ ,  $\forall j \left( \sum_{i=1}^p x_{ij}^* = 1 \right)$ ,  $\forall i \forall j [ (x_{ij}^*)^2 - x_{ij} = 0 ]$ ,

vaizduoja optimalų Hamiltono ciklą duotajame grafe, turinčiame tiesioginių ryšių matricą  $\|c_{ij}\|$ , ir yra šio skyrelio antrojo uždavinio sprendinys.

Naudodamiesi ką tik įrodytomis teoremomis ir iš jų išplaukiančia išvada, o taip pat jau minėtais šakų ir ribų bei baudų funkcijos metodais, ieškosime to optimalaus sprendinio  $\|x_{ij}^*\|$  (o kartu ir atitinkamos matricos  $\tilde{c}_{ij}$ ).

Nagrinėdami 144 „d“ pav. matricą, randame, kad ji dar nėra ta matrica  $\tilde{c}_{ij}$ , kuri atitiktų tokį sprendinį  $\|x_{ij}^*\|$ , kuris, tenkindamas apribojimus, tenkintų ir lygybę  $\tilde{F}^* = \sum_{i=1}^p \sum_{j=1}^p \tilde{c}_{ij} x_{ij}^* = 0$ , nes, pirmoje

eilutėje pasirinkę  $x_{15}^* = 1$  (arba  $x_{16}^* = 1$ ), mes jau negalėtume šeštame (arba septintame) stulpelyje rasti tokį  $x_{i6}^*$  (arba  $x_{i5}^*$ ), kur  $i \neq 1$ , kad sandauga  $\tilde{c}_{i6} x_{i6}^*$  (arba sandauga  $\tilde{c}_{i5} x_{i5}^*$ ) būtų lygi 0.

Tačiau 144 „d“ pav. matrica vis dėlto turi nemažai nulių ir todėl galima pasirinkti kai kurias  $x_{ij}$  reikšmes  $x_{ij}^* = 1$  taip, kad sandauga  $\tilde{c}_{ij} x_{ij}^*$  būtų lygi 0.

Kyla klausimas, nuo kurio elemento  $\tilde{c}_{ij} = 0$  reikėtų pradėti. Čia ir pasinaudosime baudų funkcijos metodu, **įvertindami žemutines ribas tų baudų, kurias neišvengiamai tektų mokėti, atsisakius pasirinkti tą ar kitą nulį matricoje  $\tilde{c}_{ij}$ , t. y. atsisakius įjungti į Hamiltono ciklą tą nulį atitinkantį lanką ( $i \rightarrow j$ ).**

Pavyzdžiui, matricos „d“ elementas  $\tilde{c}_{6,1} = 0$ . Pasirinkę  $x_{6,1} = x_{6,1}^* = 1$ , mes gautume, jog  $\tilde{c}_{6,1} \cdot x_{6,1}^* = 0$ . Tai reiškia, jog lanko ( $6 \rightarrow 1$ ) įtraukimas į Hamiltono ciklą nepadidintų tą ciklą sudarančių lankų svorių sumos. Tačiau, jei mes atsisakytume įtraukti į Hamiltono ciklą kaip tik tą lanką, priklausantį šeštajai eilutei ir pirmajam stulpeliui, tai mums būtinai reikėtų pasirinkti kokį nors kitą lanką, priklausantį šeštajai eilutei, ir kokį nors kitą lanką, priklausantį pirmajam stulpeliui.

Vietoj  $x_{61}^* = 1$  pasirinkdami, pavyzdžiui,  $x_{63}^* = 1$  ir  $x_{51}^* = 1$ , mes padidintume funkcijos  $\tilde{F}^*$  reikšmę dydžiu  $\tilde{c}_{63}x_{63}^* + \tilde{c}_{51}x_{51}^* = \tilde{c}_{63} + \tilde{c}_{51} = 1 + 2 = 3$ , t. y. turėtume, lyginant su atveju  $\tilde{c}_{61}x_{61}^* \approx \tilde{c}_{61} = 0$ , baudą  $h_{61}$ , lygią 3. Ir tai yra mažiausia iš galimų baudų, nes kitaip pasirinkdami  $x_{ij}^* = 1$  reikšmes šeštojoje eilutėje ir pirmajame stulpelyje, mes galėtume gauti arba tą patį baudos dydį arba didesnį už jį.

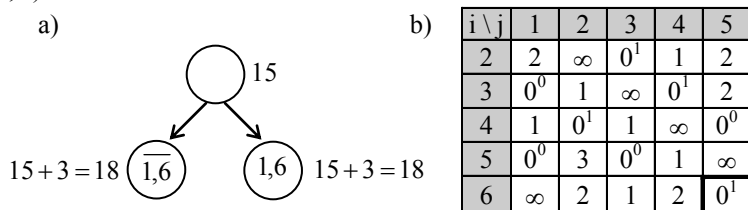
Tokiu būdu galima apskaičiuoti baudų žemutines ribas  $h_{ij}$  visiems matricos  $\|\tilde{c}_{ij}\|$  elementams, kurių reikšmės yra lygios 0. Surašysime dydžius  $h_{ij}$  matricos  $\tilde{c}_{ij}$  langeliuose kaip tuose langeliuose esančių nulių „laipsnius“ (žiūr. 144 „d“ pav.).

Kaip matome, didžiausias baudas  $h_{ij}$  iššaukia atsisakymas įtraukti į Hamiltono ciklą lankus  $(6 \rightarrow 1)$  ir  $(1 \rightarrow 6)$ , kur  $h_{61} = h_{16} = 3$ .

Natūralu, kad tikslinga įtraukti į optimalų Hamiltono ciklą kaip tik tuos lankus, kurių neištraukimas galėtų iššaukti didžiausią tikslo funkcijos  $\tilde{F}$  (o kartu – ir tikslo funkcijos  $F$ ) padidėjimą.

Kadangi yra du lankai su didžiausiomis baudomis  $h_{ij}$ , tai pradžioje pasirenkame vieną iš jų, pavyzdžiui, lanką  $(1 \rightarrow 6)$ , t. y., priimame, kad  $x_{1,6}^* = 1$  ir gauname  $\tilde{c}_{1,6}x_{1,6}^* = 0$ .

Pasirinkdami  $x_{1,6}^* = 1$ , mes šiuo lokaliu atveju pasiegsime optimaliai, tačiau **tai nereikia, jog lokaliai optimalus sprendinys būtinai bus ir globaliai optimaliu sprendiniu**. Todėl, pasirinkdami atvejį, kai į Hamiltono ciklą įtraukiame lanką  $(1 \rightarrow 6)$ , dar neatmetame ir kitos alternatyvos, kai į Hamiltono ciklą to lanko neištraukiame. Kitaip tariant, visą duotojo grafo Hamiltono ciklų aibę, turinčią  $5! = 120$  Hamiltono ciklų, suskaidome į du poaibius („šakas“) (žiūr. 145 „a“ pav.), kurių vieną pažymime  $(1,6)$ , o kitą –  $(\overline{1,6})$ .



145 pav.

Visą duotojo grafo Hamiltono ciklų aibę pažymėsime skrituliuku  $\bigcirc$  ir šalia jo parašysime pradinę  $r_0 = \sum_{i=1}^p u_i + \sum_{j=1}^p v_j$  reikšmę, lygią 15, rodančią, kokia galėtų būti minimali funkcijos  $F^*$  reikšmė

$F^*$ , jei mums pavyktų taip parinkti  $x_{ij}^*$  reikšmes, kad  $\tilde{F}$  taptų lygi 0. Kaip matėme jau anksčiau, matrica „d“ iš 144 pav. dar nėra tokia, kad tai galėtų pavykti, todėl dydis  $r_0 = 15$  yra tik žemutinė negriežta funkcijos  $F$  reikšmės riba, rodanti, kad  $F^*$  negali būti mažesnė už 15, o pati optimali funkcijos  $F$  reikšmė  $F^*$  tikrovėje gali būti ir didesnė.

Šalia skrituliuko  $(\overline{1,6})$ , vaizduojančio tą galimų Hamiltono ciklų poaibį, kuriame Hamiltono cikluose nėra lanko  $(1 \rightarrow 6)$ , užrašome tam poaibiui būdingą žemutinę negriežtą tikslo funkcijos  $F$  reikšmės  $F^*$  ribą  $15 + 3 = 18$ , kurią gauname iš ribos  $r_0 = 15$ , pridėję prie jos minimalią baudą  $h_{16} = 3$  už atsisakymą įtraukti į Hamiltono ciklą lanką  $(1 \rightarrow 6)$ .

Pasirinkus šaką  $(1,6)$ , t. y., įtraukus į Hamiltono ciklą lanką  $(1 \rightarrow 6)$ , charakterizuojamą matricoje  $\|\tilde{c}_{ij}\|$  svoriu  $\tilde{c}_{1,6} = 0$ , ir tokiu būdu išvengus baudos  $h_{ij} = 3$ , iš 144 pav. matricos „d“ galima išbraukti pirmąją eilutę ir šeštąjį stulpelį, nes toje eilutėje ir tame stulpelyje, einant šaka  $(1,6)$ , jokių  $x_{ij}$  reikšmių pasirinkinėti jau nebereikia, kadangi, pasirinkus  $x_{1,6}^* = 1$ , visi kiti  $x_{1,j}^*$ , kur  $j = 1, 2, \dots, 5$ , turi būti lygūs 0, kaip ir visi kiti  $x_{i,6}^*$ , kur  $i = 2, 3, \dots, 6$ .

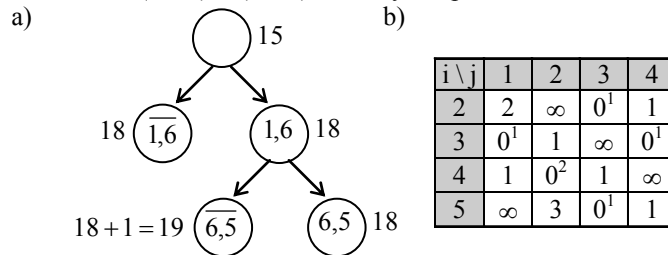
Išbraukę matricoje „d“ pirmąją eilutę ir šeštąjį stulpelį, gauname  $p-1=5$ -osios eilės matricą, pavaizduotą 144 „e“ pav., kurioje į langelį  $(6,1)$  vietoje  $\tilde{c}_{6,1} = 0$  įrašyta  $\infty$ . Taip padaryta todėl, kad

ateityje būtų išvengta nepilno ciklo, susidedančio iš lankų  $(1 \rightarrow 6)$  ir  $(6 \rightarrow 1)$ , susidarymo, pasirenkant dar ir  $x_{6,1}^* = 1$ , kai jau pasirinkta  $x_{1,6}^* = 1$ .

Kaip matome, matricoje „e“ iš pirmojo stulpelio galima atimti  $v_1 = \min_i \tilde{c}_{i1} = 2$ , o iš penktojo –  $v_5 = \min_i \tilde{c}_{i5} = 1$ , t. y., padidinti  $r$  iki  $r_0 + 3 = 18$  ir pereiti prie dar paprastesnio grafo, turinčio tą patį optimalųjį sprendinį, kaip ir duotasis, kai  $x_{1,6}^* = 1$ . Naujoji matrica, gauta iš matricos „e“, pavaizduotos 144 pav., atimant  $v_1 = 2$  ir  $v_5 = 1$ , pateikta 145 „b“ pav. Naujoji žemutinė negriežta funkcijos  $F$  reikšmės riba  $r = 15 + 3 = 18$  pavaizduota šalia šakos  $(1,6)$  (žiūr. 145 „a“ pav.).

Kaip matome iš 145 „a“ pav., po pirmojo pasirinkimo  $x_{1,6}^* = 1$ , egzistuoja dvi šakos, charakterizuojamos vienodomis negriežtomis žemutinėmis tikslo funkcijos ribomis  $r = 18$ . Kadangi tos ribos vienodos, tai, tęsdami Hamiltono ciklų aibės dekompoziciją į poaibius, galime pasirinkti bet kurią iš šių dviejų šakų. Pasirenkame šaką  $(1,6)$ , kurią gavome, pasirinkę  $x_{1,6}^* = 1$ .

Apskaičiuodami žemutines baudų ribas  $h_{ij}$  145 „b“ pav. matricos nuliams, randame, kad didžiausias  $h_{ij}$  reikšmes turi lankai  $(2 \rightarrow 3)$ ,  $(3 \rightarrow 4)$ ,  $(4 \rightarrow 2)$  ir  $(6 \rightarrow 5)$ . Pasirenkame vieną iš tų lankų, pavyzdžiui, lanką  $(6 \rightarrow 5)$ , t. y. priimame, jog  $x_{6,5}^* = 1$ , ir dekomponuojame aibę  $(1,6)$  į dvi šakas – poaibius  $(6,5)$  ir  $(\overline{6,5})$  (žiūr. 146 „a“ pav.), o iš 145 „b“ pav. matricos išbraukiame šeštąją eilutę bei penktąjį stulpelį, gaudami 146 „b“ pav. matricą, į kurios langelį  $(5,1)$  įrašome  $\infty$ , kad neleistume iš į Hamiltono ciklą patekusių lankų  $(1 \rightarrow 6)$  ir  $(6 \rightarrow 5)$  susidaryti nepilnam ciklui  $\rightarrow 1 \rightarrow 6 \rightarrow 5 \rightarrow$ .



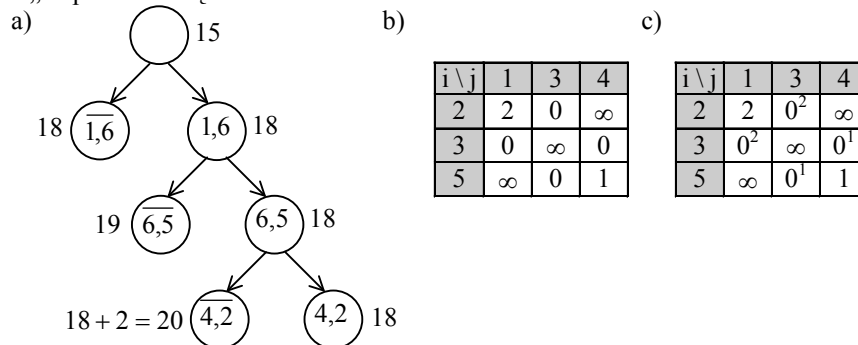
146 pav.

Kadangi kiekvienoje 146 „b“ pav. matricos eilutėje ir kiekviename stulpelyje yra po elementą, lygų nuliui, tai žemutinė tikslo funkcijos optimalios reikšmės riba  $r$  nesikeičia, todėl prie šakos  $(6,5)$  rašome tą patį įvertį 18. Šakos  $(\overline{6,5})$  įvertį gauname, pridėdami prie buvusio šakos  $(1,6)$  įverčio  $r = 18$  baudos žemutinės ribos dydį  $h_{65} = 1$ .

Nagrinėdami Hamiltono ciklų aibės dekompozicijos medį, pavaizduotą 146 „a“ pav., matome, jog jau egzistuoja trys šakos:  $(\overline{1,6})$ ,  $(\overline{6,5})$  ir  $(6,5)$ , turinčios įverčius – ribas 18, 19, 18 atitinkamai.

Pasirenkame mažiausią įvertį ir tęsiame Hamiltono ciklų aibės dekompoziciją, dekomponuodami šaką  $(6,5)$ .

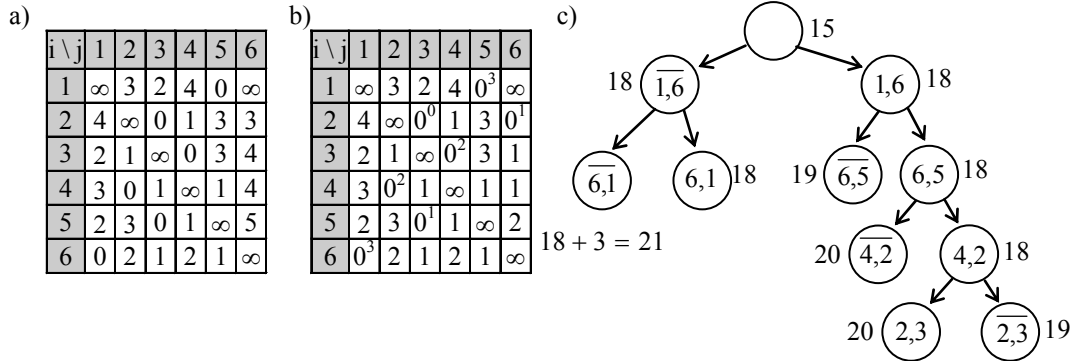
Kad nustatytume, kokį lanką įtraukti į konstruojamą Hamiltono optimalų ciklą, apskaičiuojame 146 „b“ pav. matricos nulius atitinkančias baudas. Gauname, jog didžiausią baudos žemutinę ribą, lygią 2, turi elementas  $(4 \rightarrow 2)$ . Todėl poaibį  $(6,5)$  suskaidome į du poaibius  $(4,2)$  ir  $(\overline{4,2})$  (žiūr. 147 „a“ pav.), iš 146 „b“ pav. matricos išbraukiame ketvirtąją eilutę bei antrąjį stulpelį ir, įrašę į langelį  $(2 \rightarrow 4)$   $\infty$ , gauname 147 „b“ pav. matricą.



147 pav.

Kadangi toje matricoje nėra iš vieno stulpelio ar eilutės negalima atimti jokio  $u > 0$  ar  $v > 0$ , tai žemutinę funkcijos  $F$  optimalios reikšmės  $F^*$  ribą  $r$  paliekame tą pačią, t. y. 18. Gavome 4 šakas (žiūr. 147 „a“ pav.), charakterizuojamas įverčiais 18, 19, 20, 18. Mažiausią įvertį turi šakos  $\overline{1,6}$  ir 4.2. Skaidydami šaką 4,2, gauname įverčius, didesnius už 18. Norėdami pasinaudoti šaka  $\overline{1,6}$ , turėtume grįžti į dekompozicijos pradžią, t. y. prie 144 „d“ pav. matricos, ir, įrašydami į jos  $(1 \rightarrow 6)$  langelį begalybės ženklą, pereiti prie šakos  $(1,6)$  dekompozicijos.

Gauname 148 „a“ pav. matricą.

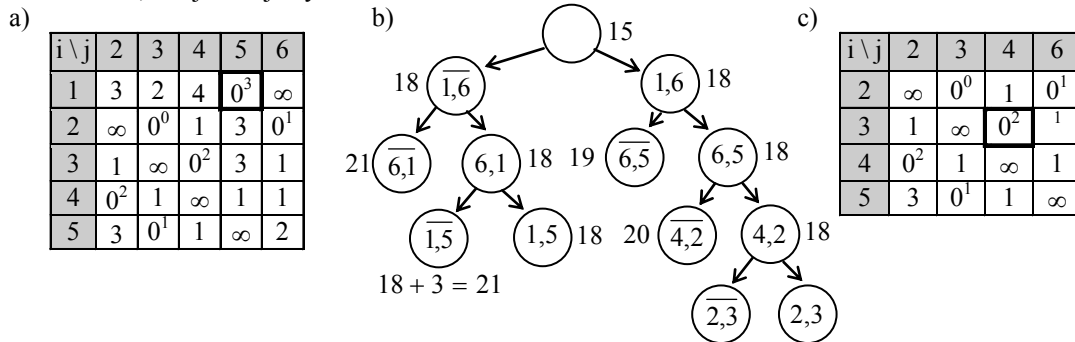


148 pav.

Iš tos matricos šeštojo stulpelio galime atimti  $v_6 = 3 = h_{1,6}$ . Gauname 148 „b“ pav. matricą, atitinkančią šaką  $(1,6)$  ir turinčią įvertį  $15 + 3 = 18$ . Apskaičiavę tos matricos nulių baudų žemutines ribas  $h_{ij}$ , gauname, kad didžiausias dydis yra, pavyzdžiui,  $h_{61} = 3$  (žiūr. 148 „b“ pav.).

Todėl poaibį  $(1,6)$  suskaidome į du poaibius  $(6,1)$  ir  $(\overline{6,1})$  (žiūr. 148 „c“ pav.) ir surandame naujų šakų įverčius.

Iš 148 „b“ pav. matricos išbraukiame pirmąjį stulpelį ir šeštąją eilutę, priimdami, kad  $x_{6,1}^* = 1$ , t. y., kad poaibio  $(1,6)$  dalį – poaibį  $(6,1)$  sudaro Hamiltono ciklai, neturintys savo sudėtyje lanko  $(1 \rightarrow 6)$  ir turintys lanką  $(6 \rightarrow 1)$ . Į naujai gautos 149 „a“ pav. matricos langelį  $(1 \rightarrow 6)$  įrašinėti begalybės ženklo jau nebereikia, nes jis ten jau yra.



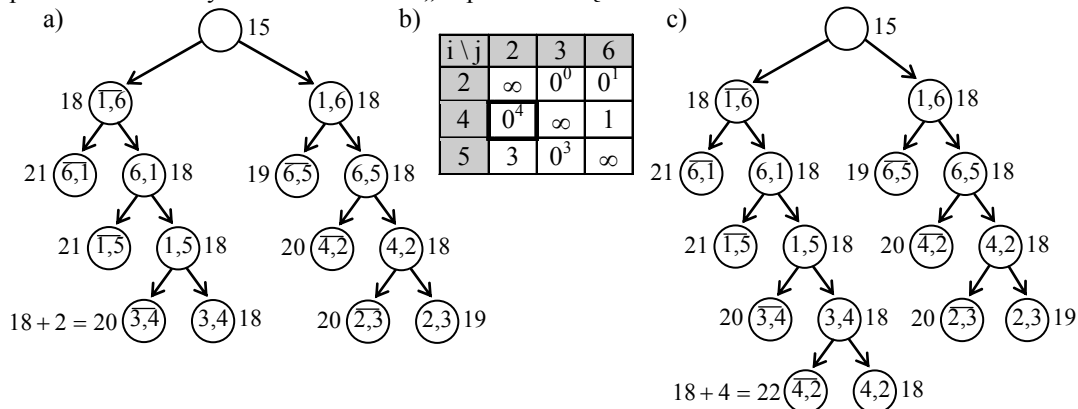
149 pav.

Kadangi iš 149 „a“ matricos stulpelių ir eilučių nieko atimti neįmanoma, tai įvertis  $r$  lieka be pasikeitimų ir todėl prie 148 „c“ pav. šakos  $(6,1)$   $r$  reikšmė išlieka lygi 18.

Įvertinę 149 „a“ pav. matricos nulių, gauname, kad didžiausias  $h_{ij}$  yra  $h_{1,5} = 3$ . Todėl poaibį  $(6,1)$  dekomponuojame į du poaibius:  $(1,5)$  ir  $(\overline{1,5})$  (žiūr. 149 „b“ pav.). Poaibio  $(1,5)$  įvertį gauname pridėję prie poaibio  $(6,1)$  įverčio 18 dydį  $h_{1,5} = 3$ , o priėmę, kad  $x_{1,5}^* = 1$ , iš 149 „a“ pav. matricos išbraukiame pirmąją eilutę ir penktąjį stulpelį. Gauname 149 „c“ pav. matricą, į kurios langelį  $5 \rightarrow 6$  įrašome  $\infty$ , kad išvengtume nepilno ciklo  $\square 6 \rightarrow 1 \rightarrow 5 \rightarrow$  sudarymo, susijungiant Hamiltono cikle jau esančios grandinės  $6 \rightarrow 1 \rightarrow 5$  galams – viršūnėms 5 ir 6.

Kadangi iš 149 „c“ pav. matricos eilučių ir stulpelių nieko negalime atimti, tai žemutinę  $F^*$  riba lieka ta pati. Todėl prie šakos  $(1,5)$  užrašome įvertį 18 ir apskaičiuojame 149 „c“ pav. matricos nulių  $h_{ij}$ . Gauname, kad didžiausias  $h_{ij}$  reikšmės turi  $h_{34}$  ir  $h_{42}$ . Pasirenkame  $h_{34} = 2$ , ir poaibį  $(1,5)$  dekomponuojame į poaibius  $(3,4)$  ir  $(\overline{3,4})$  (žiūr. 150 „a“ pav.). Šakai  $(3,4)$  apskaičiuojame įvertį

$r=18+h_{3,4}=20$  ir, pasirinkę  $x_{3,4}^*=1$ , pereiname į poaibį (3,4), išbraukdami iš 149 „c“ pav. matricos trečiąją eilutę ir ketvirtąjį stulpelį bei įrašydami į langelį (4→3) begalybės ženklą  $\infty$ , kad išvengtume nepilno ciklo susidarymo. Gauname 150 „b“ pav. matricą.

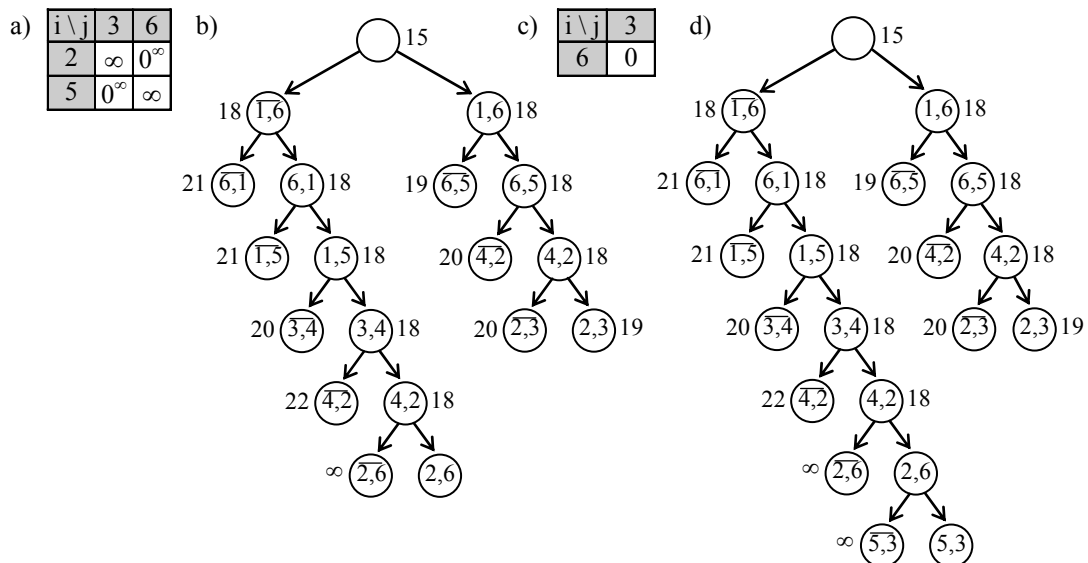


150 pav.

Kadangi 150 „b“ pav. matricoje nei iš jos eilučių nei iš stulpelių nieko atimti negalima, nepadarius neigiamais kai kurių  $\tilde{c}_{ij}$  reikšmių, tai gauname, kad poaibio (3,4) įvertis nepasikeitė ir išliko lygus 18. Apskaičiavę 150 „b“ pav. matricoje nulių  $h_{ij}$ , randame, jog maksimali  $h_{ij}$  reikšmė priklauso  $h_{4,2}=4$ . Todėl poaibį (3,4) suskaidome į poaibius (4,2) ir  $(\overline{4},2)$  (žiūr. 150 „c“ pav.) ir apskaičiuojame poaibio  $(\overline{4},2)$  įvertį. Gauname  $18+h_{4,2}=18+4=22$ . Tam, kad gautume poaibio (4,2) įvertį, iš 150 „b“ pav. matricos išbraukiame ketvirtąją eilutę ir antrąjį stulpelį. Gauname 151 „a“ pav. matricą, kurioje į langelį 2→3 įrašome  $\infty$ , kad iš Hamiltono cikle esančios grandinės  $3 \rightarrow 4 \rightarrow 2$ , kuri atsirado, priėmus  $x_{4,2}^*=1$ , nesusidarytų pilnas ciklas.

Iš 151 „a“ matricos eilučių ir stulpelių nieko atimti negalima, todėl poaibio (4,2) įvertis išlieka toks pat, t. y. 18, ką ir užrašome 150 „c“ pav. Apskaičiavę  $h_{ij}$  151 „a“ pav. matricos nuliams, randame, jog  $h_{2,6}=h_{5,3}=\infty$ . Pasirenkame, pavyzdžiui,  $x_{2,6}^*=1$  ir gauname Hamiltono ciklą dekompozicijos medį, pavaizduotą 151 „b“ pav. Iš 151 „a“ matricos išbraukiame antrąją eilutę ir šeštąjį stulpelį. Gauname 151 „c“ pav. „matricą“, susidedančią iš langelio (5→3). Į tą langelį begalybės ženklo jau nebeįrašome, nes mums jau nebereikia apsidrausti nuo pilno ciklo susidarymo. Priimame, kad  $x_{5,3}^*=1$  ir gauname pilną

optimalų Hamiltono ciklą  $\overline{6} \rightarrow 6 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow 4 \rightarrow 2 \rightarrow$ , susidedantį iš lankų  $(6 \rightarrow 1)$ ,  $(1 \rightarrow 5)$ ,  $(3 \rightarrow 4)$ ,  $(4 \rightarrow 2)$ ,  $(2 \rightarrow 6)$  ir  $(5 \rightarrow 3)$ . Jo ilgis  $c_{6,1}+c_{1,5}+c_{3,4}+c_{4,2}+c_{2,6}+c_{5,3}=18$ , kas sutampa su tikslo funkcijos  $F^*$  žemutine riba  $r=18$ . 151 „d“ pav. parodytas Hamiltono ciklą aibės dekompozicijos medis, gautas taikant Litlo algoritimą.



151 pav.

Ši medį, taikant Litlo algoritmą, galima gauti iškart, žingsnis po žingsnio, neskaidant į 144, 145, 147, 148, 149, 150 ir 151 paveikslėlius, kas buvo padaryta tik didaktiniais sumetimais, kad šios mokymo priemonės skaitytojams būtų lengviau suvokti, kaip taikomi šakų ir ribų bei baudos funkcijų metodai, sudarant Litlo algoritmą Hamiltono optimalaus ciklo paieškai.

Nors Hamiltono optimalaus ciklo paieška grafe, pavaizduotame 144 „a“ pav., užėmė gana daug laiko, tačiau vis dėlto leido pasiekti tikslą greičiau, negu apskaičiuojant visų 120 galimų Hamiltono ciklų tame grafe ilgį ir išrenkant iš jų trumpiausią.

Kaip matome iš gautų Hamiltono ciklų aibės dekompozicijos medžių, mes neiškart pataikėme į trumpiausią kelią, vedantį į optimalaus ciklo radimą, todėl mums, nuėjus tam tikrą nuotolį dekompozicijos medžiu, teko sugrįžti į medžio pradžią ir pradėti kitos jo šakos dekompoziciją. Tokių sugrįžimų, taikant Litlo algoritmą, pasitaiko daugiau ar mažiau, ir tai žymiai padidina darbo, reikalingo optimaliam Hamiltono ciklui rasti, kiekį. Kartais tokių „šokinėjimų“ nuo vienos dekompozicijos medžio šakos prie kitos būna tiek daug, kad Litlo algoritmas tampa nebeefektyvus. Todėl **negalime sakyti, kad Litlo algoritmas visais atvejais leidžia pakankamai greitai surasti optimalų Hamiltono ciklą duotajame grafe.** Iš to išplaukia, jog **bendruoju atveju Hamiltono optimalaus ciklo radimo uždavinys praktiškai dar nėra pilnai išspręstas.** Kad skaitytojas galėtų įsitikinti, jog kartais Litlo algoritmas tampa labai neefektyvus, siūlau jam pabandyti rasti optimalų Hamiltono ciklą grafe, kurio tiesioginių ryšių matrica pateikta 152 „a“ pav.

|    |       |    |    |    |    |    |    |    |   |
|----|-------|----|----|----|----|----|----|----|---|
| a) | i \ j | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8 |
| 1  | ∞     | 1  | 2  | 3  | 4  | 5  | 6  | 7  |   |
| 2  | 1     | ∞  | 13 | 12 | 11 | 10 | 9  | 8  |   |
| 3  | 2     | 13 | ∞  | 14 | 15 | 16 | 17 | 18 |   |
| 4  | 3     | 12 | 14 | ∞  | 22 | 21 | 20 | 19 |   |
| 5  | 4     | 11 | 15 | 22 | ∞  | 23 | 24 | 25 |   |
| 6  | 5     | 10 | 16 | 21 | 23 | ∞  | 27 | 26 |   |
| 7  | 6     | 9  | 17 | 20 | 24 | 27 | ∞  | 28 |   |
| 8  | 7     | 8  | 18 | 19 | 25 | 26 | 28 | ∞  |   |

|    |                |                |    |    |    |    |                 |                |   |
|----|----------------|----------------|----|----|----|----|-----------------|----------------|---|
| b) | i \ j          | 1              | 2  | 3  | 4  | 5  | 6               | 7              | 8 |
| 1  | ∞              | 1 <sup>1</sup> | 2  | 3  | 4  | 5  | 6               | 7              |   |
| 2  | 1 <sup>1</sup> | ∞              | 13 | 12 | 11 | 10 | 9               | 8 <sup>2</sup> |   |
| 3  | 2              | 13             | ∞  | 14 | 15 | 16 | 17              | 18             |   |
| 4  | 3              | 12             | 14 | ∞  | 22 | 21 | 20 <sup>5</sup> | 19             |   |
| 5  | 4              | 11             | 15 | 22 | ∞  | 23 | 24              | 25             |   |
| 6  | 5              | 10             | 16 | 21 | 23 | ∞  | 27              | 26             |   |
| 7  | 6              | 9              | 17 | 20 | 24 | 27 | ∞               | 28             |   |
| 8  | 7              | 8              | 18 | 19 | 25 | 26 | 28              | ∞              |   |

152 pav.

Kadangi, 152 „a“ pav. matrica yra simetriška, tai taikant „jėgos“ algoritmą tektų apskaičiuoti  $\frac{7!}{2} = 360$  Hamiltono ciklų ilgį ir išrinkti iš jų trumpiausią. Taikant Litlo algoritmą, šiuo atveju darbo apimtis bus ne ką mažesnė, tačiau darbas bus mažiau mechanizuotas ir pareikalaus daugiau atidumo. Skaičiuojant kompiuteriu ir taikant „jėgos“ algoritmą, bus sunaudota apie sekundę kompiuterio laiko. Apie tiek pat laiko bus sunaudota ir taikant Litlo algoritmą, tačiau programa šiuo atveju bus žymiai sudėtingesnė.

Baigdamas Litlo algoritmo ir jo taikymo apžvalgą, norėčiau atkreipti skaitytojo dėmesį, kad panaudota tame algoritme Hamiltono ciklų aibės dekompozicijos idėja gali būti pritaikyta ir sprendžiant pirmąjį šio skyrelio uždavinį, nes sudarant tokį dekompozicijos medį, galima rasti visus duotojo grafo Hamiltono ciklus.

Kadangi, esant didelėms p reikšmėms, ne tik „jėgos“, bet ir Litlo algoritmu naudotis tampa praktiškai nebeefektyvu, tai tenka ieškoti kitų Hamiltono optimalaus ciklo duotajame grafe suradimo būdų.

Tas paieškas skatina šio uždavinio aktualumas praktikoje. Jį gana dažnai vadina ne tik Hamiltono optimalaus ciklo paieškos, bet ir **keliaujančio pirklio uždaviniu (KPU)**, nes viena iš galimų šio uždavinio interpretacijų kaip tik ir sutampa su uždaviniu, išskylančiu keliaujančiam pirkliui, norinčiam aplankyti p gyvenviečių (miestelių, kaimų ir pan.), pabuvojant kiekvienoje iš jų po vieną ir tik po vieną kartą ir išleidžiant kuo mažiau pinigų kelionei (sugaištant kuo mažiau laiko, keliaujant trumpiausiu maršrutu ir pan.).

Uždavinys, išskylantis prekių išvežiotojui, išvežiojančiam prekes į prekybos taškus, taip pat yra KPU, kaip ir uždavinys, išskylantis pašto tarnautojui, pristatančiam siuntinius jų gavėjams į namus. Skirtumas tarp šių uždavinių tik tame, kad prekių išvežiotojui prekybos taškai ilgą laiką išlieka tie patys, ir jis, išsprendęs KPU vieną kartą, to sprendinio rezultatu gali naudotis ilgą laiką, o pašto tarnautojui kiekvieną kartą tenka spręsti vis naują uždavinį. Kiekvienam iš mūsų, planuojančiam aplankyti mieste keletą įstaigų bei kitų vietų, taip pat tenka spręsti KPU.

Dar vienas KPU, turintis nemažą praktinę reikšmę, tai valcavimo staklynų (bliumingų, sliabingų) darbo planavimo uždavinys. Šiuo atveju Hamiltono ciklo viršūnės – tai valcavimo staklynų darbas su tam tikro tipo valcų komplektais, valcuojant metalinius ruošinius ir gaminant tam tikro

profilio gaminius (lakštus, kampuočius, vamzdžius, metalinius strypus, turinčius kvadratinį, apvalų, lovinį, tėjinių ir t. t. profilius), o Hamiltono ciklo lankų svoriai – valcų komplektų pakeitimo trukmės, pereinant nuo vieno profilio gaminių prie kito. Planuojant valcavimo staklyną darbą, stengiamasi sudaryti tokią valcavimo darbų seką – periodiškai besikartojantį valcavimo tvarkaraštį, kad perėjimų nuo vieno darbų tipo prie kito, keičiant valcų kompleksus, trukmių suma būtų mažiausia.

Su KPU susiduriama ir elektroninių mikroschemų gamyboje, kuomet, gaminant integralinę mikroschemą, tos schemas plokštėje lazerio spinduliu gręžiama dešimtys tūkstančių labai plonų skylučių ir išskyla laiko, eikvojamo tos plokštės stumdymui ir sukiojimui po stacionaraus lazerio spinduliu, minimizavimo problema. Čia susiduriama su tokių Hamiltono optimalių ciklų paieška, kurie turi dešimtis tūkstančių viršūnių, vaizduojamų mikroschemas skylutėmis. Hamiltono ciklų lankai ir jų svoriai – tai perėjimai nuo vienos skylutės gręžimo prie kitos ir tų perėjimų trukmės, o optimalus Hamiltono ciklas – tai tos mikroschemos plokštės judėjimo po stacionaraus lazerio spinduliu maršrutas, kuris minimizuoja bendrą mikroschemos paruošimo laiką.

Taigi, praktikoje išskyla uždaviniai, kai grafo viršūnių skaičius  $p$  siekia ne tik šimtus, bet matuojamas dešimtimis tūkstančių. Kaip spręsti Hamiltono optimalių ciklų radimo uždavinius tuomet?

Tokiais atvejais tenka atsakyti tikslų optimizavimo algoritmų (tokių kaip „jėgos“ ar Litlo algoritmai) taikymo ir vietoj jų naudotis **artutiniiais optimizavimo algoritmais**, leidžiančiais per praktiškai priimtina laiką surasti sprendinius, pakankamai artimus optimaliems (atsitiktinai galinčius net sutapti su optimaliais).

Vienas paprasčiausių artutinių Hamiltono optimalaus ciklo paieškos algoritmų yra **artimiausio kaimyno algoritmas**. Jį sudaro šie žingsniai:

1) imama kokia nors nagrinėjimo grafo viršūnė „ $i^*$ “ kaip pradinis Hamiltono ciklo konstravimo taškas;

2) tarp iš tos viršūnės išeinančių lankų (t. y. tą viršūnę atitinkančioje tiesioginių ryšių matricos eilutėje) išrenkamas tas lankas, kurio svoris  $c_{i^*j}$  yra mažiausias, ir tokiu būdu patenkama į naują viršūnę  $j$ , kuri pažymima  $j^*$ ; eilutė „ $i^*$ “ ir stulpelis „ $j^*$ “ išbraukiami iš tiesioginių ryšių matricos, o į jos langelį  $(j^*, i^*)$  įrašoma  $\infty$ , kad išvengtume nepilnų ciklų susidarymo;

3) pereinama prie viršūnės „ $j^*$ “, t. y. prie eilutės, turinčios numerį „ $j^*$ “, ir tikrinama, ar dar liko neišbrauktų stulpelių; jei neišbrauktų stulpelių daugiau nei vienas, tai grįžtama į antrąjį žingsnį, laikant seniau išsirinktąjį elementą (viršūnę)  $j^*$  naujuoju  $i^*$  ir išrenkant iš likusių dar neišbrauktų langelių tokį lanką  $(i^{**}, j)$ , kurio svoris  $c_{i^{**}j}$  būtų mažiausias, išbraukiant eilutę  $i^{**}$  ir stulpelį  $j$  bei įrašant į langelį  $(j, i^*)$  begalybės ženklą, o jei liko tik vienas neišbrauktas stulpelis (jo numeris „ $j^*$ “ turi sutapti su pirmosios pasirinktos viršūnės numeriu „ $i^*$ “), tai paskutinėje neišbrauktoje eilutėje išsirenkama to neišbraukto stulpelio  $c_{ij}$ , ir algoritmas baigia darbą.

Artimiausio kaimyno algoritmo taikymą grafui, kurio tiesioginių ryšių matrica pateikta 152 „a“ pav., iliustruoja 152 „b“ pav., kuriame pirmoji pasirinktoji viršūnė turi numerį „ $i^*=1$ “.

Nagrinėdami pirmąją 152 „a“ pav. matricos eilutę, randame, jog  $\min_j c_{1j} = \min_j c_{1j} = c_{12} = 1$ .

Tai reiškia, jog viršūnei  $i^*=1$  artimiausias kaimynas yra viršūnė  $j^*=2$ . Lanką  $(1 \rightarrow 2)$  152 „b“ pav. matricoje išskiriame rėmeliu ir pažymime pirmuoju numeriu, įrašydami tą numerį į langelio  $(1,2)$  viršutinį dešinįjį kampą. Kad nesusidarytų nepilnų ciklų, langelį  $(2,1)$  užstrichuojame (kas reiškia begalybės ženklo  $\infty$  įrašymą) ir taip pat pasižymime pirmuoju numeriu. 152 „b“ pav. matricoje išbraukiame eilutę, turinčią numerį  $i^*=1$ , ir stulpelį  $j^*=2$ . Kadangi matricoje dar liko daugiau negu vienas neišbrauktas stulpelis, tai kartojame antrąjį algoritmo žingsnį ir pereiname prie eilutės, turinčios numerį  $i=j^*=2$  analizės.

Randame, jog  $\min_{j \neq 2} c_{2j} = c_{28} = 8$ . Išskiriame rėmeliu langelį  $(2 \rightarrow 8)$  ir pasižymime jį antruoju

numeriu. Tuo pačiu numeriu pasižymime langelį  $(8 \rightarrow 1)$  ir užstrichuojame. Išbraukiame matricoje antrąją eilutę ir aštuntąjį stulpelį. Pereiname į aštuntąją eilutę.

Randame, jog  $\min_{j \neq 2,8} c_{8j} = c_{83} = 18$ . Išskiriame rėmeliu langelį  $(8 \rightarrow 3)$  ir pasižymime trečiuoju

numeriu. Tuo pačiu numeriu pasižymime ir užstrichuojame langelį  $(3 \rightarrow 1)$ . Išbraukiame aštuntąją eilutę ir trečiąjį stulpelį. Pereiname į trečiąją eilutę.

Randame, jog  $\min_{j \neq 2,8,3} c_{3j} = c_{3,4} = 14$ . Išskiriame rėmeliu langelį  $(3 \rightarrow 4)$  ir pažymime jį ketvirtuoju

numeriu. Tuo pačiu numeriu pažymime langelį  $(4 \rightarrow 1)$  ir užstrichuojame jį. Išbraukiame trečiąją eilutę ir ketvirtąjį stulpelį. Pereiname į ketvirtąją eilutę.

Randame, jog  $\min_j c_{4j} = c_{4,7} = 20$ . Išskiriame rėmeliu langelį (4→7) ir pažymime jį

penktuoju numeriu. Tuo pačiu numeriu pažymime langelį (7→1) ir jį užštrichuojame. Išbraukiame ketvirtąją eilutę ir septintąjį stulpelį. Pereiname į septintąją eilutę.

Randame, jog  $\min_j c_{7j} = c_{7,5} = 24$ . Išskiriame rėmeliu langelį (7→5) ir pažymime jį šeštuoju

numeriu. Tuo pačiu numeriu pažymime ir langelį (5→1) ir užštrichuojame jį. Išbraukiame septintąją eilutę ir penktąjį stulpelį. Pereiname į penktąją eilutę.

Randame, jog  $\min_j c_{5j} = c_{5,6} = 23$ . Išskiriame langelį (5→6) rėmeliu ir pažymime jį

septintuoju numeriu. Kadangi matricoje liko neišbrauktas tik vienas stulpelis, tai jau nebereikia uždrausti ciklų susidarymo, nes tai jau bus pilnas Hamiltono ciklas. Išbraukiame penktąją eilutę ir šeštąjį stulpelį. Pereiname į šeštąją eilutę.

Ten randame tik vieną neišbrauktą langelį  $c_{6,1}=5$ , atitinkantį lanką (6→1), uždarančią pilną Hamiltono ciklą. Išskiriame jį rėmeliu ir pažymime aštuntuoju numeriu. Tuo artimiausiojo kaimyno algoritmas ir baigia darbą, nes jau surastas Hamiltono ciklas  $\boxed{1 \rightarrow 2 \rightarrow 8 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 6 \rightarrow 1}$ .

Belieka apskaičiuoti jo ilgį:  $c_{12}+c_{28}+c_{83}+c_{34}+c_{47}+c_{76}+c_{61}=1+8+18+14+20+24+23+5=113$  vienetų. Surastas Hamiltono ciklas – nebūtinai optimalus (trumpiausias) Hamiltono ciklas, tačiau daugelyje atvejų jis bus ne taip jau tolimas nuo optimalaus.

Artimiausiojo kaimyno algoritmo taikymo rezultatas gali priklausyti nuo to, kuri grafo viršūnė buvo pasirinkta kaip pirmoji.

Pavyzdžiui, jei būtume pradėję algoritmo darbą nuo antrosios viršūnės, tai būtume gavę Hamiltono ciklą  $\boxed{2 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 2}$ , kurio ilgis  $c_{21}+c_{13}+c_{34}+c_{48}+c_{85}+c_{56}+c_{67}+c_{72}=1+2+14+19+25+23+27+9=120$  vienetų.

Kartais pavyksta pagerinti artimiausio kaimyno algoritmo rezultatą, naudojantis **kartotiniu artimiausio kaimyno algoritmu**, kurio esmę sudaro artimiausio kaimyno algoritmo taikymas kiekvienai grafo viršūnei, po to išrenkant iš gautųjų Hamiltono ciklų aibės, sudarytos iš p ciklų, tą (arba tuos), kuris yra trumpiausias (trumpiausi).

Pradėdami 152 „a“ matricoje nuo trečiosios viršūnės, būtume gavę Hamiltono ciklą  $\boxed{3 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 6 \rightarrow 3}$ , turintį ilgį, lygų  $2+1+8+19+20+24+23+16=113$  vienetų.

Pradėdami nuo ketvirtosios viršūnės, būtume gavę Hamiltono ciklą  $\boxed{4 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 3 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 4}$ , kurio ilgis lygus  $3+1+8+18+15+23+27+20=115$  vienetų.

Jei pradėtume nuo penktosios viršūnės, tai gautume Hamiltono ciklą  $\boxed{5 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 6 \rightarrow 5}$ , turintį tą patį ilgį, lygų  $4+1+8+18+14+20+27+23=115$  vienetų.

Pradėdami nuo šeštosios viršūnės, gauname Hamiltono ciklą  $\boxed{6 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 3 \rightarrow 4 \rightarrow 7 \rightarrow 5 \rightarrow 6}$ , kurio ilgis yra lygus  $5+1+8+18+14+20+24+23=113$  vienetų.

Kai pradėdame nuo septintosios viršūnės, tai gauname Hamiltono ciklą  $\boxed{7 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 5 \rightarrow 7}$ , turintį ilgį, lygų  $6+1+8+18+14+21+23+24=115$  vienetų.

Pradėję nuo aštuntosios viršūnės, būtume gavę Hamiltono ciklą  $\boxed{8 \rightarrow 1 \rightarrow 2 \rightarrow 7 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 5 \rightarrow 8}$ , kurio ilgis yra  $7+1+9+17+14+21+23+25=117$ .

Palyginę visus aštuonis gautus rezultatus, randame, jog kartotinis artimiausio kaimyno algoritmas davė tą patį rezultatą kaip ir paprastas artimiausio kaimyno algoritmas, kai pradėdama nuo pirmosios ar trečiosios duotojo grafo viršūnės. Tačiau, taikydami paprastąjį artimiausio kaimyno algoritmą, mes juk galėjome pradėti ir nuo antrosios viršūnės ir būtume gavę Hamiltono ciklą, kurio ilgis 120 vienetų, t. y. galėjome gauti rezultatą, kuris skiriasi nuo rezultato, gauto naudojantis kartotiniu artimiausio kaimyno algoritmu, apie pusę to, kiek pastarasis skiriasi nuo tikslaus optimalaus sprendinio.

Jei būtume pabandę taikyti kartotinį artimiausio kaimyno algoritmą grafiui, pavaizduotam 144 „a“ pav., tai būtume gavę optimalų sprendinį – Hamiltono ciklą  $\boxed{1 \rightarrow 5 \rightarrow 3 \rightarrow 4 \rightarrow 2 \rightarrow 6 \rightarrow 1}$ , turintį ilgį, lygų 18 vienetų, kai tuo tarpu paprastas artimiausio kaimyno algoritmas, pritaikytas tam grafiui, pradėdant viršūnė  $i=4$ , būtų davęs Hamiltono ciklą, turintį 24 vienetų ilgį.

Kitas Hamiltono ciklų, kurie dažnai būna artimi optimaliems, paieškos algoritmas yra **pigiausios jungties algoritmas**. Jis reikalauja mažiau darbo negu kartotinis artimiausio kaimyno algoritmas.

Naudojantis pigiausios jungties algoritmu, pradedama nuo  $\min_i \min_j c_{ij}$  duotoje matricoje  $\|c_{ij}\|$  suradimo. Kai tokių pigiausių jungčių yra keletas, tai pasirenkama viena iš jų.

Tarkime, jog pasirinkome pigiausią lanką, charakterizuojamą kaina  $c_{i^*j^*}$ . Tuomet į Hamiltono ciklą įtraukiamas lankas  $(i^* \rightarrow j^*)$ , į matricos langelį  $(j^* \rightarrow i^*)$  įrašomas ženklas  $\infty$  ir iš tiesioginių ryšių matricos išbraukiama eilutė „ $i^*$ “ ir stulpelis „ $j^*$ “.

Po to vėl ieškoma pigiausioji jungtis likusioje (po išbraukimų) matricoje. Tarkime, jog  $\min_{i \neq i^*} \min_{j \neq j^*} c_{ij} = c_{i^{**}j^{**}}$ . Tuomet į Hamiltono ciklą įjungiamas lankas  $(i^{**} \rightarrow j^{**})$  ir žiūrima, ar jis nesudaro

grandinių su jau esančiais lankais. Jei, pavyzdžiui,  $j^* = i^{**}$ , tai Hamiltono cikle egzistuoja grandinė  $(i^* \rightarrow (j^* = i^{**}) \rightarrow j^{**})$ . Tuomet, prieš išbraukdami iš matricos eilutę „ $i^{**}$ “ ir stulpelį „ $j^{**}$ “, į tos matricos langelį  $(j^{**} \rightarrow i^*)$  įrašome begalybės ženklą. Kai  $j^* = i^*$ , tai Hamiltono cikle atsiranda grandinė  $(i^{**} \rightarrow (j^* = i^*) \rightarrow j^*)$  ir į matricos langelį  $j^* \rightarrow i^{**}$  reikia įrašyti begalybės ženklą  $\infty$ . Jei  $j^* \neq i^{**}$  ir  $i^* \neq j^{**}$ , tai naujasis lankas  $(i^{**} \rightarrow j^{**})$  sudaromame Hamiltono cikle egzistuoja kol kas atskirai, nesudarydamas su jame jau esančiu lanku grandinės. Tuomet begalybės ženklas įrašomas į matricos langelį  $(j^{**} \rightarrow i^*)$  ir po to, išbraukus iš matricos eilutę „ $i^{**}$ “ ir stulpelį „ $j^{**}$ “, kartojama naujos pigiausios jungties paieška. Tas kartojimas vyksta iki tol, kol matricoje belieka viena neišbraukta jungtis, uždaranti Hamiltono ciklą. Įtraukę tą jungtį į minėtąjį ciklą, gauname artutinį šio skyrelio antrojo uždavinio (t. y. KPU) sprendinį.

Naudodamiesi pigiausios jungties algoritmu, rasime artutinį sprendinį tuo atveju, kai duotojo grafo tiesioginių ryšių matrica yra 152 „a“ pav. matrica.

Pradžioje nustatome, jog  $\min_i \min_j c_{ij} = c_{12} = c_{21} = 1$ . Pasirenkame  $(1 \rightarrow 2)$ , o į langelį  $(2 \rightarrow 1)$  įrašome  $\infty$ . Eilutę  $i=1$  ir stulpelį  $j=2$  išbraukiame.

Dabar  $\min_{i \neq 1} \min_{j \neq 2} c_{ij} = c_{31} = 2$ . Lanką  $(3 \rightarrow 1)$  įrašome į Hamiltono ciklą, kuriame jis su jau anksčiau įrašytu lanku  $(1 \rightarrow 2)$  sudaro grandinę  $(3 \rightarrow 1 \rightarrow 2)$ . Todėl į matricos langelį  $(2 \rightarrow 3)$  įrašome  $\infty$  ir iš matricos išbraukiame eilutę  $i=3$  ir stulpelį  $j=1$ .

Likusioje matricoje  $\min_{i \neq 1,3} \min_{j \neq 2,1} c_{ij} = c_{28} = 8$ . Lanką  $(2 \rightarrow 8)$  įrašome į konstruojamą Hamiltono ciklą, kur gaunama grandinė  $(3 \rightarrow 1 \rightarrow 2 \rightarrow 8)$ . Į matricos langelį  $(8 \rightarrow 3)$  įrašome  $\infty$ , o antrąją eilutę ir aštuntąjį stulpelį išbraukiame.

Dabar  $\min_{i \neq 1,3,2} \min_{j \neq 2,1,8} c_{ij} = c_{43} = 14$ . Į konstruojamą Hamiltono ciklą įjungiamo lanką  $(4 \rightarrow 3)$ , kuris su jau ten esančiais lankais sudaro grandinę  $(4 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 8)$ . Į matricos langelį  $(8 \rightarrow 4)$  įrašome  $\infty$ , o ketvirtąją eilutę ir trečiąjį stulpelį išbraukiame.

Dabar  $\min_{i \neq 1,3,2,4} \min_{j \neq 2,1,8,3} c_{ij} = c_{7,4} = 20$ . Lanką  $(7 \rightarrow 4)$  įtraukiame į konstruojamą Hamiltono ciklą, kuriame jis su ten jau esančiais lankais sudaro grandinę  $(7 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 8)$ . Į matricos langelį  $(8 \rightarrow 7)$  įrašome  $\infty$ , o septintąją eilutę ir ketvirtąjį stulpelį išbraukiame.

Likusioje trečiosios eilės matricoje  $\min_{i \neq 1,3,2,4,7} \min_{j \neq 2,1,8,3,4} c_{ij} = c_{65} = c_{56} = 23$ . Pasirenkame lanką  $(5 \rightarrow 6)$  ir įjungiamo jį į konstruojamą Hamiltono ciklą. Ten tas lankas kol kas išlieka neprijungtas prie jokios grandinės, todėl begalybės ženklą įrašome į matricos langelį  $(6 \rightarrow 5)$ . Penktąją eilutę ir šeštąjį stulpelį išbraukiame.

Likusioje antrosios eilės matricoje surandame minimalią jungtį  $c_{85}=25$  ir įjungiamo ją į konstruojamą Hamiltono ciklą, kuriame gauname grandinę  $(7 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 5 \rightarrow 6)$ . Iš matricos išbraukiame aštuntąją eilutę ir penktąjį stulpelį. Belieka tik vienas lankas  $(6 \rightarrow 7)$  su svoriu  $c_{6,7}=27$ , uždarantis pilną Hamiltono ciklą  $\boxed{7 \rightarrow 4 \rightarrow 3 \rightarrow 1 \rightarrow 2 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 7}$ , kurio ilgis yra  $c_{74}+c_{43}+c_{31}+c_{12}+c_{28}+c_{85}+c_{56}+c_{67}=20+14+2+1+8+25+23+27=120$ .

Jei būtume pritaikę pigiausios jungties algoritimą 144 „a“ pav. grafiui, tai būtume gavę Hamiltono ciklą, kurio visų lankų svorių suma būtų 2 vienetais didesnė už optimalaus Hamiltono ciklo lankų svorių sumą. Tai dar kartą patvirtina teiginį, kad kartotinis artimiausio kaimyno algoritmas dažnai leidžia gauti geresnį sprendinį negu pigiausios jungties algoritmas.

Egzistuoja visa eilė kitų artutinių Hamiltono optimalaus ciklo paieškos algoritmų.

Vienas tokių algoritmų – **artimiausiojo intarpo algoritmas**. Pagrindinė to algoritmo idėja – pradėti nuo nedidelio grafo, sudarančio duotojo grafo pografi, ciklo sudarymo, po to didinti tą ciklą,

įjungiant kiekviename žingsnyje po vieną viršūnę, tol, kol visos duotojo grafo viršūnės bus įtrauktos į Hamiltono ciklą.

Taikant šį algoritmą pradžioje pasirenkama bet kuri duotojo grafo viršūnė kaip „pradinis ciklas“, sudarytas iš vienos viršūnės ir nulio briaunų. Šis „pradinis ciklas“ koku nors būdu pažymimas.

Antrajame žingsnyje likusioje grafo dalyje surandama viršūnė, esant arčiausiai jau pasirinktos viršūnės, ir sudaromas naujas ciklas, susidedantis jau iš dviejų viršūnių. Tas ciklas pažymimas.

Tollesniuose žingsniuose likusioje grafo dalyje ieškoma viršūnių, kurių įterpimas į jau turimą ciklą minimaliai padidintų to ciklo ilgį. Šie žingsniai kartojami iki tol, kol gaunamas pilnas Hamiltono ciklas.

Pritaikius artimiausiojo intarpo algoritmą grafui, kurio tiesioginių ryšių matrica pavaizduota 152 „a“ pav., nesunku surasti Hamiltono ciklą  $\rightarrow 6 \rightarrow 2 \rightarrow 7 \rightarrow 4 \rightarrow 8 \rightarrow 1 \rightarrow 5 \rightarrow 3 \rightarrow$ , kurio ilgis yra  $10+9+20+19+7+4+15+16=100$  vienetų ir net dar trumpesnį ciklą  $\rightarrow 8 \rightarrow 2 \rightarrow 6 \rightarrow 3 \rightarrow 5 \rightarrow 1 \rightarrow 7 \rightarrow 4 \rightarrow$ , turintį jau tik  $8+10+16+15+4+6+20+19=98$  vienetų ilgį, t. y. 15 vienetų mažesnį, negu taikant kartotinį artimiausiojo kaimyno algoritmą.

Yra gana daug gana įmantrių apytiksliai optimalaus KPU sprendinio radimo algoritmų, garantuojančių, kad gautojo **apytikslio sprendinio paklaida neviršys iš anksto nustatyto paklaidos dydžio**. Šiuo metu žinomi geriausi KPU apytikslio sprendinio algoritmai leidžia rasti grafo, turinčio  $p \leq 100000$  viršūnių, apytikriai optimalų Hamiltono ciklą, kuris skiriasi nuo optimalaus ne daugiau kaip 1%, o grafe, turinčiame apie milijoną viršūnių, rasti optimalų sprendinį, kuris skirtųsi nuo optimalaus ne daugiau kaip 3,5%.

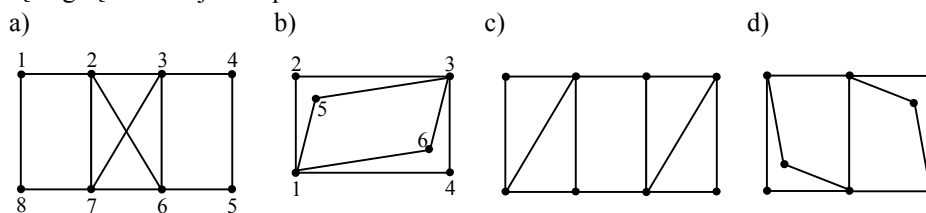
Nuolatiniai tokių algoritmų pagerinimai ir kompiuterinės technikos bei jos programinės įrangos tobulėjimas rodo, kad sprendinių tikslumas ir sprendžiamų KPU dydis ateityje dar didės. Kai kurios čia paminėtų algoritmų charakteristikos pateiktos 47-oje lentelėje, kuri paimta iš knygos [109]. Pirmajame šios lentelės stulpelyje nurodomas KPU dydis, antrajame – atsakoma į klausimą, ar, naudojantis šiuolaikine technika, jau galima surasti optimalų sprendinį ir, jei taip, tai kiek laiko sugaištų superkompiuteris, sprendžiamas šis uždavinys, trečiame, ketvirtame ir penktame stulpeliuose atsakoma į klausimą, ar galima rasti apytikrų KPU sprendinį, kurio paklaida neviršytų duotojo dydžio, ir jei taip, tai per kiek laiko. Atsakymai „Taip“ ir „Ne“ kartu informuoja ir apie tai, ar tokie uždaviniai buvo sprendžiami ir išspręsti.

Nežiūrint į tuos išpūdingus rezultatus, kurie pateikti 47-oje lentelėje, **klausimas, ar egzistuoja efektyvus KPU tikslaus sprendinio algoritmas, išlieka aktualus tiek teoriškai, tiek praktiškai.**

47. lentelė

| KPU dydis           | Tikslus optimalus sprendinys     | Artutinio algoritmo taikymas, garantuojantis, kad paklaida neviršys 3,5% | Artutinio algoritmo taikymas, garantuojantis, kad paklaida neviršys 1% | Artutinio algoritmo taikymas, garantuojantis, kad paklaida neviršys 0,75% |
|---------------------|----------------------------------|--------------------------------------------------------------------------|------------------------------------------------------------------------|---------------------------------------------------------------------------|
| $p \leq 3000$       | Taip; skaičiavimo laikas <1 sav. | Taip; skaičiavimo laikas – kelios sekundės                               | Taip; skaičiavimo laikas – kelios minutės                              | Taip; skaičiavimo laikas <1 val.                                          |
| $p \leq 1000000$    | Ne                               | Taip; skaičiavimo laikas – kelios minutės                                | Taip; skaičiavimo laikas – mažiau 2 dienų                              | Taip; skaičiavimo laikas – apie 7 mėn.                                    |
| $p \approx 1000000$ | Ne                               | Taip; skaičiavimo laikas <3 val.                                         | Ne, nes skaičiavimo laikas – šimtai metų                               | Ne, nes skaičiavimo laikas – milijonai metų                               |

Lygindami Oilerio ir Hamiltono ciklų paieškos uždavinius, matome, jog jie žymiai skiriasi savo sudėtingumu, nors jų formuluotės, atrodo, labai panašios. Pasirodo, jog **Oilerio ir Hamiltono ciklų egzistavimo sąlygos yra nepriklausomos**, t. y. galimi visi keturi tų ciklų egzistavimo ir neegzistavimo atvejai. Šį teiginį iliustruoja 153 pav.



153 pav.

- 153 „a“ pav. grafas turi ir Oilerio ir Hamiltono ciklus, kurių radimą palieku skaitytoji.  
 153 „b“ pav. grafas turi keletą Oilerio ciklų, tačiau neturi nė vieno Hamiltono ciklo.  
 153 „c“ pav. grafas neturi Oilerio ciklo, tačiau turi Hamiltono ciklą.  
 153 „d“ pav. grafas neturi nei Oilerio, nei Hamiltono ciklo.

### 1.9.2.16. Tinklinių grafikų pervedimo iš viršūninės formos į briauninę ir iš briauninės į viršūninę formą operacijos

**Tinkliniais grafikais (TG)** vadinami darbų kompleksų, kurių vykdyme dalyvauja keletas (gali būti ir labai daug) atskirų vykdytojų, grafiniai modeliai. Tokie modeliai priklauso pačių paprasčiausių vidinių modelių klasei, nes tų modelių elementų išoriniai modeliai yra labai paprasti: juos sudaro elementų charakteristikų vidutinių reikšmių rinkiniai, susidedantys iš vieno, dviejų (retkarčiais – trijų ir daugiau) pastovių skaičių – konstantų.

Nežiūrint į savo paprastumą (o gal kaip tik dėl jo), tinkliniai grafikai praktikoje taikomi labai plačiai. Visų pirma, TG – tai sudėtingų kompleksų, sudarytų iš daugelio darbų, turinčių skirtingus vykdytojus, **planavimo (tame tarpe ir – optimalaus) instrumentas**. Naudojantis TG, suderinami darbų pradžios ir baigimo terminai taip, kad darbai kuo mažiau lauktų vienas kito ir kad bendras viso darbų komplekso įvykdymo laikas būtų kuo trumpesnis. TG taikymas leidžia racionaliau paskirstyti resursus, išlyginti vykdytojų apkrovimą, padidinti darbų įvykdymo laiku tikimybes, išskirti darbus, vadinamus **kritiniais**, kuriems vadovai privalėtų skirti daugiausia dėmesio ir resursų.

Tinklinis planavimas patogus dar ir tuo, jog, naudojantis juo, žymiai sumažėja tikimybė, kad koks nors darbas, reikalingas numatytam tikslui pasiekti, bus praleistas (t. y. nebus užplanuotas, nebus numatyti jo įvykdymui reikalingi resursai ir pan.) arba bus suplanuotas tokiu laiku ir tokiomis sąlygomis, kad to darbo įvykdyti tuo metu būtų neįmanoma.

TG yra ne tik planavimo, bet ir **operatyvios darbų vykdymo proceso kontrolės**, to **proceso operatyvaus valdymo instrumentas**. TG leidžia sudėtingų sistemų ir procesų vadovams koncentruoti savo dėmesį kaip tik „siaurose“ valdymo proceso vietose, laiku pastebėti, kurie iš darbų tampa kritiniais, ir imtis reikalingų priemonių, tikslingai paskirstyti bei perskirstyti turimus resursus.

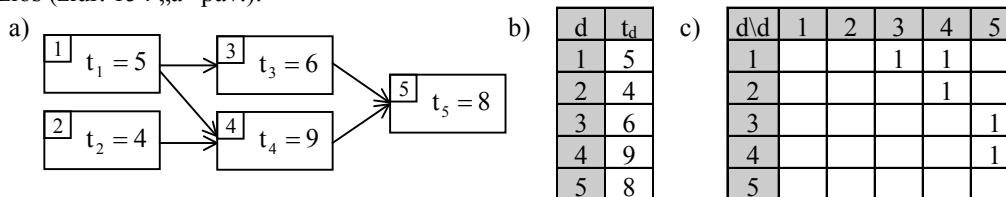
Todėl sudėtingų procesų valdymo metodologijoje susikūrė net speciali kryptis – **tinklinis planavimas ir valdymas (TPV)**. Apie visa tai bus gana plačiai kalbama antrojoje šios mokymo priemonės dalyje.

Tinkliniai grafikai gali turėti **viršūninę formą** ir tuomet vadinami viršūniniais TG (VTG), o taip pat ir **briauninę formą** ir tuomet vadinami briauniniais TG (BTG).

**Viršūninį TG** sudaro dviejų tipų elementai:

- 1) **darbai** 1, 2, 3, ..., n, charakterizuojami savo trukmėmis  $t_1, t_2, t_3, \dots, t_n$  (kartais dar ir kainomis  $c_1, c_2, c_3, \dots, c_n$ , vykdytojų skaičiumi ir pan.);
- 2) **loginiai ryšiai** tarp darbų.

VTG darbai vaizduojami stačiakampiais, o loginiai ryšiai – orientuotais lankais, rodančiais loginę priklausomybę tarp darbo, iš kurio tas lankas išeina, pabaigos ir darbo, į kurį tas lankas įeina, pradžios (žiūr. 154 „a“ pav.).



154 pav.

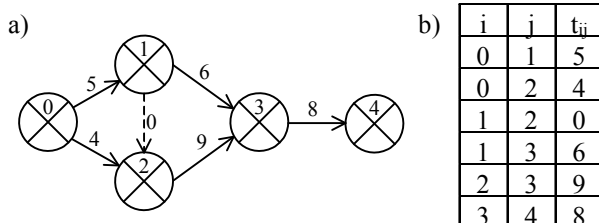
Pavyzdžiui (žiūr. 154 „a“ pav.), loginis ryšys (1→3) rodo, jog darbas  $d=3$  gali prasidėti tik darbui  $d=1$  pasibaigus, o tam, kad būtų galima pradėti darbą  $d=4$ , reikia, kad būtų baigti  $d=1$  ir  $d=2$ . Darbai  $d=1$  ir  $d=2$  gali būti pradėti nepriklausomai vienas nuo kito, o darbo  $d=5$  pradžia priklauso nuo darbų  $d=3$  ir  $d=4$ .

Darbų 1, 2, 3, ..., n sąrašas su nurodytomis tų darbų charakteristikomis  $t_1, t_2, t_3, \dots, t_n$  (kartais dar ir  $c_1, c_2, \dots, c_n$  bei kitomis) aprašo **VTG sudėtį** (žiūr. 154 „b“ pav.), o loginiai ryšiai, (pavyzdžiui, tų ryšių matrica, ar ryšių operatorius) – **VTG struktūrą** (žiūr. 154 „c“ pav.).

**Briauninis TG** taip pat turi dviejų rūšių pagrindinius elementus: 1) **darbus**, vaizduojamus orientuotais lankais, virš kurių rašomos tų darbų charakteristikos, ir 2) **įvykius**, vaizduojamus skrituliukais ir išreiškiančius momentus, kada, pasibaigus vieniems darbams, prasideda kiti darbai, t. y. darbų vykdymo kalendoriaus ypatingus taškus.

Įvykiai numeruojami, t. y. įvykiams priskiriami skirtingi numeriai, kurie kartu identifikuoja ir darbus, nurodydami, koku įvykiu tas ar kitas darbas prasideda, koku – baigiasi: kiekvienas darbas žymimas įvykių numerių pora – darbo pradžios ir darbo pabaigos įvykių numeriais. Tie numeriai leidžia nustatyti loginius ryšius tarp darbų, todėl dauguma loginių ryšių, kurie VTG buvo vaizduojami orientuotais lankais, BTG tiesiogiai nevaizduojami, o išplaukia iš įvykių numerių. Tik kai kuriais atvejais visų loginių ryšių netiesiogiai atvaizduoti nepavyksta ir tuomet tenka kai kuriuos tokius ryšius vaizduoti „fiktyviais darbais“, turinčiais trukmę, lygią 0. „Fiktyvus darbas“ jokio darbo nereiškia, o orientuotu lanku, sudarytu iš štrichų, rodo tik loginį ryšį tarp vieno darbo pabaigos ir kito pradžios.

BTG, sutampantis su VTG, pavaizduotu 154 „a“ pav., pateiktas 155 „a“ pav.



155 pav.

**BTG lentelinėje formoje** (žiūr. 155 „b“ pav.) aprašo **iškart ir BTG sudėtį ir struktūrą**, t. y. 155 „b“ pav. lentelė yra kartu ir darbų sąrašas su jų charakteristikomis (nagrinėjamu atveju – darbų trukmėmis  $t_{ij}$ ) ir ryšių tarp darbų aprašymas, nes kiekvieno darbo numeris yra to darbo pradžią ir pabaigą vaizduojančių ir su kitais darbais rišančių įvykių numerių (i,j) pora.

Tačiau, kaip jau buvo minėta, kartais atsitinka taip, kad ryšių tarp vieno darbo pabaigos ir kitų darbų pradžios vienareikšmiškam atvaizdavimui nebepakanka vien tik įvykių. Pavyzdžiui, norint parodyti, kad darbas  $d=4$  gali prasidėti tik pasibaigus kiekvienam iš darbų  $d=1$  ir  $d=2$  (žiūr. 154 „a“ pav.), briauniniame TG jau nebepakanka vien tik įvykių „1“ ir „2“.

Darbą  $d=1$  briauniniame TG vaizduoja lankas, žymimas įvykių „0“ ir „1“ pora (0, 1). To darbo pabaiga – įvykis „1“ – sutampa su darbo  $d=3$  pradžia, kurį briauniniame TG vaizduoja įvykių „1“ ir „3“ pora (1; 3) (žiūr. 155 „a“ ir „b“ pav.). Darbo  $d=2$ , kurį briauniniame TG vaizduoja lankas (0→2), (jis 155 „b“ pav. lentelėje žymimas (0; 2)), pabaiga sutampa su darbo  $d=4$ , vaizduojamo lanku (2→3), pradžia. Šie visi darbai ir ryšiai tarp jų, egzistavę viršūniniame TG, vienareikšmiškai atvaizduoti briauniniame TG.

Tačiau iškyla sunkumų, atvaizduojant darbo  $d=4$  pradžios priklausomybę nuo darbo  $d=1$  pabaigos, nes tie įvykiai 155 „a“ pav. vaizduojami skirtingais numeriais „2“ ir „1“. Sujungti du tuos įvykius į vieną negalima dėl dviejų priežasčių: 1) sujungę tuos įvykius gautume, jog ir darbo  $d=3$  pradžia priklauso nuo darbo  $d=2$  pabaigos, ko iš tikrųjų nėra (žiūr. VTG, pavaizduotą 154 „a“ pav.), 2) lenteliniame BTG modelyje (155 „b“ pav.) nebegalėtume atskirti darbo  $d=1$  nuo darbo  $d=2$ , nes abiejų tų darbų pradžią ir pabaigą įvykiai taptų vienodi: abu darbai prasidėtų įvykiu „0“ ir baigtųsi įvykiu „1“ ir abu darbai būtų žymimi vienodai, t. y. (0, 1).

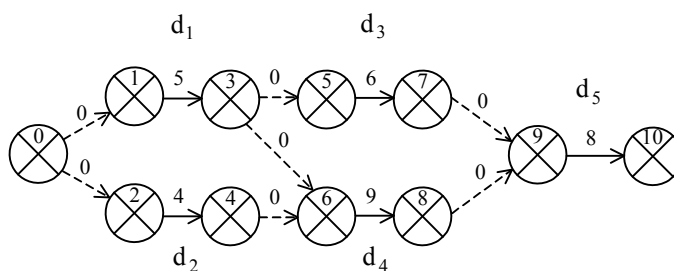
Todėl tokiais atvejais loginį ryšį tarp darbų  $d=1$  ir  $d=4$  tenka perkelti iš VTG ir į BTG, vaizduojant jį „štrichine“ rodykle (kad būtų galima atskirti nuo realių darbų), pavadinant tą rodyklę „fiktyviu darbu“, t. y. darbu, neturinčiu trukmės, o vaizduojančiu tik loginį ryšį.

Žinoma, pereinant iš VTG į BTG, buvo galima perkelti ne tik darbus  $d=1$ ,  $d=2$ ,  $d=3$ , ..., bet ir visus loginius ryšius tarp jų, vaizduojant darbus orientuotais lankais su skrituliukais tų lankų pradžioje ir pabaigoje, o ryšius – štrichiniais orientuotais lankais, jungiančiais vieno darbo pabaigas su kitų pradžiomis, kartais įvedant dar ir darbų komplekso pradžios bei pabaigos įvykius, bei tų įvykių ryšius su realiais darbais, kaip tai parodyta 156 pav.

Tačiau toks VTG pervedimas į BTG, koks parodytas 156 pav., žymiai padidintų BTG informacinį modelį, ir tokiu būdu būtų bereikalingai apkraunama kompiuterio atmintis. Todėl, pervedant VTG į BTG, stengiamasi įvedinėti kuo mažiau fiktyvių darbų, kaip tai buvo padaryta 155 pav.

Pavedimo iš VTG į BTG algoritmas, minimizuojantis įvedamų į BTG fiktyvių darbų skaičių, yra žymiai sudėtingesnis už tą, kuriuo naudodamiesi iš VTG, pavaizduoto 154 pav., gavome 156 pav. briauninį TG. Šis algoritmas bus aprašytas antrojoje mokymo priemonės dalyje, kai bus nagrinėjami tipiniai vidiniai modeliai bei tipinės jų struktūros, o dabar siūlyčiau skaitytojui pačiam pabandyti surasti tas situacijas, kada pervedant VTG į BTG neišvengiamai tenka įvedinėti fiktyvius darbus, o po to, atsižvelgiant į tokias situacijas, sukonstruoti algoritmą, pervedantį VTG į BTG taip, kad į BTG būtų įvedinėjama kiek galima mažiau fiktyvių darbų.

a)



b)

| i | j  | $t_{ij}$ |
|---|----|----------|
| 0 | 1  | 0        |
| 0 | 2  | 0        |
| 1 | 3  | 5        |
| 2 | 4  | 4        |
| 3 | 5  | 0        |
| 3 | 6  | 0        |
| 4 | 6  | 0        |
| 5 | 7  | 6        |
| 6 | 8  | 9        |
| 7 | 9  | 0        |
| 8 | 9  | 0        |
| 9 | 10 | 8        |

156 pav.

Antrojoje mokymo priemonės dalyje bus detalai aprašytas bei pademonstruotas ir atvirkštinis – pervedimo iš BTG į VTG formą – algoritmas. Šis algoritmas žymiai paprastesnis už algoritmą, pervedantį VTG į BTG, minimizuojant fiktyvių darbų skaičių. Pervedimo iš BTG į VTG algoritmas kiekvieną realų darbą ( $i_k, j_k$ ) briauniniame TG pakeičia darbu, turinčiu eilės tvarka priskiriamą numerį „k“ viršūniniame TG, o „fiktyviems darbams“ (t. y., tiems darbams, kurių  $t_{ij}=0$ ), jokio numerio nepriskiria ir į VTG darbų sąrašą tų „darbų“ neįtraukia.

Pirmajame pervedimo iš BTG į VTG etape gaunamas realių darbų sąrašas – VTG sudėties modelis, kuriame kiekvienas realus darbas turi du „numerius“: naująjį numerį „k“, kurį jis ir turės viršūniniame TG, ir senąjį – ( $i_k, j_k$ ), t. y. „numerį“, turėtąjį briauniniame TG. Atskirai sudaromas fiktyviųjų „darbų“ sąrašas, kuriame tie „darbai“ neturi naujų numerių, o tik senuosius, t. y. įvykių numerių poras ( $i, j$ ).

Antrajame pervedimo iš BTG į VTG etape sudaromas VTG struktūros modelis. Tuo tikslu sudaroma kvadratinė matrica, turinti tiek eilučių ir stulpelių, kiek VTG sudėties modelyje – realių darbų sąrašas – yra realių darbų.

Kiekvienam darbui „k“ iš to sąrašo priskiriama tos matricos k-ji eilutė ir k-asis stulpelis, o po to, pasižiūrėjus į to darbo senąjį „numerį“ ( $i_k, j_k$ ), darbų sąrašo senuosiuose numeriuose ieškoma tokių porų ( $i, j$ ), kurių pabaigos įvykio numeris „j“ sutaptų su darbo „k“ senojo „numerio“ pradinio įvykio numeriu „i“. Tarkime, kad m-jo darbo senojo „numerio“ ( $i_m, j_m$ ) pabaigos įvykio numeris  $j_m=i_k$ . Tai reiškia, jog egzistuoja loginis ryšys tarp m-jo darbo pabaigos ir k-jo darbo pradžios. Tokiu atveju loginių ryšių matricos m-sios eilutės k-jame stulpelyje įrašomas „1“, pažymintis tą loginį ryšį. Kai darbo „k“ pradžia (įvykis „i“) surišta su kelių darbų pabaiga, tai matricos k-jo stulpelio atitinkamose eilutėse analogiškai įrašomi vienetai, vaizduojantys k-jo darbo pradžios loginius ryšius su atitinkamų darbų pabaigomis.

Peržiūrėjus visus realių darbų sąrašo darbus ir įrašius į tiesioginių ryšių matricos k-jo stulpelio atitinkamas eilutes vienetus (arba neįrašius nė vieno, kas būna, kai darbas „k“ priklauso pačių pirmųjų darbų grupei), pereinama prie fiktyviųjų darbų sąrašo peržiūros. Ieškoma tokių fiktyviųjų darbų ( $i, j$ ), kurių „j“ sutaptų su „i“. Suradus tokį fiktyvų darbą, išskiriamas to „darbo“ pradinis įvykis „i“ ir peržiūrimi realių fiktyviųjų darbų sąrašai, ieškant juose tokių darbų, kurių „j“ sutaptų su išskirtuoju „i“. Jei realiųjų darbų sąrašuose surandami tokie darbai, tai tiesioginių ryšių matricos k-jo stulpelio eilutėse, turinčiose numerius tų realiųjų darbų, kurių senųjų „numerių“ „j“ sutapo su išskirtuoju „i“, įrašomi vienetai. Jei išskirtasis fiktyviojo darbo „i“ sutampa su kito fiktyvaus darbo „j“, tai tame darbe išskiriamas pradinio įvykio numeris „i“, su kuriuo vykdoma ta pati procedūra, kaip ir su seniau išskirtuoju „i“. Taip surandami visi k-jo darbo pradžios įvykio „i“ loginiai ryšiai su visų kitų darbų pabaigomis ir užpildomas tiesioginių ryšių matricos k-sis stulpelis.

Užpildžius k-jį stulpelį, pildoma k-ji tiesioginių ryšių matricos eilutė. Tuo tikslu peržiūrimas realiųjų darbų sąrašas ir ieškoma realių darbų, kurių pradinis įvykis „i“ sutaptų su k-jo darbo pabaigos įvykiu „j“. Tarkime, kad suradome tokį realų darbą ir jo numeris yra „r“. Tuomet į k-sios eilutės r-tąjį stulpelį įrašomas vienetas. Peržiūrėjus visą realių darbų sąrašą ir įrašius į k-sios eilutės kai kuriuos stulpelius vienetus (arba neįrašius nė vieno, kai k-sis darbas priklauso paskutiniųjų darbų grupei ir po jo jau jokių kitų darbų nevyksta), pereinama prie fiktyviųjų darbų sąrašo. Jei surandamas toks fiktyvusis darbas, kurio „i“ sutampa su „j“, tai išskiriamas to fiktyvaus darbo pabaigos įvykis „j“ ir realių darbų sąrašas ieškoma darbų, kurių „i“ sutaptų su išskirtuoju „j“. Suradus tokius realius darbus ir nustačius tokių darbų naujuosius numerius, į tiesioginių ryšių matricos k-sios eilutės atitinkamus stulpelius įrašomi vienetai. Tokiu būdu užpildoma matricos k-ji eilutė.

Aprašytoji procedūra su k-ju darbu kartojama su visais realių darbų sąrašo darbais ir tokiu būdu užpildoma visa tiesioginių ryšių matrica.

Kadangi kiekvienas loginis ryšys jungia du realius darbus, tai į kai kuriuos tiesioginių ryšių matricos langelius vienetukais bus įrašinėjami du kartus. Vienetuko įrašymą galima laikyti jo disjunkcija su senuoju langelio turiniu, kurios rezultatas vis vien bus „1“ ar „0“.

Užbaigus tiesioginių ryšių matricos stulpelių ir eilučių pildymą, baigiama ir BTG pervedimo į VTG procedūra.

Skaitytojui visai pagrįstai gali iškilti klausimas: „Kam reikalingos dvi TG formos: VTG ir BTG? Kodėl neužtenka vienos iš tų formų ir kodėl reikalingi pervedimo iš VTG į BTG ir iš BTG į VTG algoritmai?“

Vienareikšmiško atsakymo į šiuos klausimus šių eilučių autorius duoti nesiryžtų, tačiau kai kurias priežastis, sąlygojančias tai, kad egzistuoja dvi TG formos, nurodyti galima.

Pagrindinė priežastis, galbūt, glūdi tame, kad **sudarinėti darbų kompleksų tinklinius modelius paprasčiau yra viršūninėje formoje, vaizduojant kiekvieną atskirą darbą stačiakampiu ir jungiant tuos stačiakampius loginiais ryšiais, nusakančiais darbų vykdymo eiliškumą, o apdoroti informaciją apie TG, nustatant įvykių įvykimo ankstyvuosius ir vėlyvuosius laikus, įvykių ir darbų rezervus, išskiriant kritinius darbus ir iš jų susidedančius kritinius kelius, yra patogiau, kai TG turi briauninę formą, jungiančią savyje ir TG sudėties ir struktūros modelius.** Tačiau TG sudarymas iškart briauninėje formoje irgi nėra toks jau sunkus, kad būtų tikslinga pradžioje sudaryti VTG, o po to jį transformuoti į BTG, naudojantis jau minėtuoju gana sudėtingu algoritmu. Be to informaciniu VTG modeliu taip pat galima naudotis ir tiesiogiai, ne būtinai pervedant jį į briauninę formą.

Todėl dviejų TG formų egzistavimas, šių eilučių autoriaus nuomone, matyt, greičiau išplaukia ne tiek iš vidinių VTG ir BTG savybių, kiek iš istoriškai susiklosčiusių aplinkybių: vieniems TG teorijos kūrėjams labiau „prie širdies“ viršūninė, kitiems – briauninė TG forma.

Praktiniuose TG taikymuose dėl informacijos apdorojimo patogumo labiau paplito BTG, todėl ir šioje mokymo priemonėje (taip pat ir jos antrojoje dalyje) daugiau dėmesio skiriama operacijoms su BTG.

Apdorojant TG kompiuteriais, naudinga turėti pervedimo iš vienos formos į kitą programas, kad vartotojai galėtų naudotis TG informacijos apdorojimo programine įranga nepriklausomai nuo to, kokioje formoje (BTG ar VTG) pateikta pradinė informacija.

### **1.9.2.17. Briauninių tinklinių grafikų charakteristikų apskaičiavimo operacijos**

Sudarius kokio nors sudėtingo darbų komplekso briauninį TG, apskaičiuojamos to komplekso valdymui reikalingos BTG charakteristikos.

Vieną tokių charakteristikų grupę sudaro **įvykių charakteristikos**: įvykių „j“ įvykimo ankstyvieji laikai (įvykimo momentai)  $t_j^{(a)}$ , įvykio „j“ įvykimo leistini vėlyvieji laikai  $t_j^{(v)}$  ir įvykių rezervai  $R_j = t_j^{(v)} - t_j^{(a)}$ .

Antrąją charakteristikų grupę sudaro **darbų charakteristikos**: darbų (i,j) pilnieji (bendrieji) rezervai  $R_{ij}^{(p)}$ , darbų (i,j) laisvieji  $R_{ij}^{(l)}$  ir daliniai  $R_{ij}'$  bei  $R_{ij}''$  rezervai.

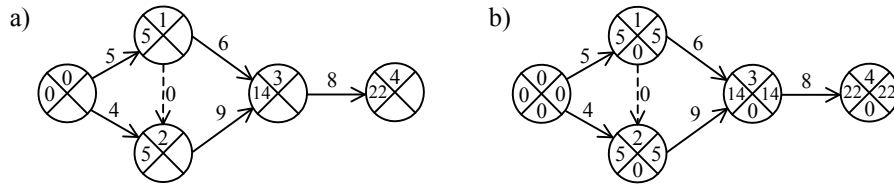
Žinant įvykių ir darbų rezervus, galima rasti **TG kritinius kelius**, jiems priklausančius **kritinius darbus** ir racionaliau paskirstyti turimus resursus.

Apskaičiuojant  $t_j^{(a)}$ , pradiniam įvykiui, kuriuo prasideda nagrinėjamojo darbų komplekso vykdymas, ankstyvasis to įvykio įvykimo laikas dažniausiai priimamas lygus 0, t. y. laikoma, kad  $t_{\text{prad}}^{(a)} = 0$ . Taip daroma tuo tikslu, kad TG charakteristikų apskaičiavimo rezultatus būtų lengviau „pririšti“ prie bet kurio pradinio momento  $t_0$ , pridedant prie apskaičiuotų visiems TG įvykiams terminų  $t_j^{(a)}$  bei  $t_i^{(v)}$  pastovų dydį  $t_0$ . Žinoma, galima ir iškart priimti  $t_{\text{prad}}^{(a)} = t_0$ , tačiau, keičiantis  $t_{\text{prad}}^{(a)} = t_0$ , TG charakteristikų perskaičiavimas būtų šiek tiek sudėtingesnis nei tuo atveju, kai  $t_{\text{prad}}^{(a)} = 0$ .

Visų kitų įvykių ankstyvieji laikai apskaičiuojami, naudojantis formule  $t_j^{(a)} = \max_i (t_i^{(a)} + t_{ij})$  ir slenkant tinkliniu grafiku nuo pradinio įvykio iki baigiamojo. Ši formulė išplaukia iš tos aplinkybės, kad naują darbą, prasidedantį įvykiu „j“, galima pradėti tik tada, kada bus baigti visi darbai, besibaigiantys

tuo įvykiu. Tie darbai gali prasidėti įvairiais įvykiais „i“ ir todėl reikia nustatyti, kuris iš visų tų darbų ( $i, j$ ), kur „j“ – fiksuotas, o „i“ – kintantis numeris, baigsis vėliausiai.

Apdorojant BTG informaciją „rankiniu būdu“, ankstyvieji įvykių įvykimo laikai įrašomi į įvykius vaizduojančių skrituliukų kairiuosius sektorius (žiūr. 157 „a“ pav., kuriame apskaičiuotos  $t_j^{(a)}$  reikšmės briauniniam TG, pateiktam 155 „a“ pav.).



157 pav.

Apdorojant BTG kompiuteriu,  $t_j^{(a)}$  apskaičiavime naudojamosi lenteliniu BTG modeliu (žiūr. 155 „b“ pav.) ir jau minėtąja formule. Kad apskaičiavimų metu būtų išvengta „šokinėjimų“ po visą darbų sąrašą, kuris gali būti ir gana ilgas, tikslinga darbus tame sąrašė iš anksto išrikiuoti taip, kad visi darbai ( $i, j$ ), turintys vienodą „j“ ir skirtingus „i“, būtų išsidėstę tame sąrašė vienas po kito vienoje grupėje ir tokių darbų jau nebebūtų kitose darbų sąrašo vietose. Tokį darbų išrikiavimą vadina „išrikiavimu nuo galo“.

„Išrikiavimo nuo galo“ algoritmas, o taip pat  $t_j^{(a)}$  apskaičiavimo kompiuteriu algoritmas aprašyti antroje šios mokymo priemonės dalyje.

Apskaičiavus TG įvykių ankstyvuosius laikus  $t_j^{(a)}$  ir suradus paskutiniojo komplekse (baigiamojo) įvykio įvykimo ankstyvąjį laiką  $t_{\text{baig}}^{(a)}$  (mūsų nagrinėjamu atveju, pavaizduotu 157 „a“ pav.,  $t_{\text{baig}}^{(a)} = t_4^{(a)} = 22$ ), galima priimti (taip dažniausiai ir daroma, nes tai supaprastina kritinio kelio radimo ir vidinių rezervų įvertinimo algoritmus), kad baigiamojo įvykio leistinas vėlyvasis įvykimo laikas  $t_{\text{baig}}^{(v)}$  yra lygus to įvykio įvykimo ankstyvajam laikui  $t_{\text{baig}}^{(a)}$ , t. y. galima priimti, kad  $t_{\text{baig}}^{(v)} = t_{\text{baig}}^{(a)}$ .

Tuomet, žinodami  $t_{\text{baig}}^{(v)}$ , galime apskaičiuoti visų kitų įvykių leistinus vėlyvuosius laikus  $t_i^{(v)}$ , slinkdami tinkliniu grafiku atgal (nuo baigiamojo įvykio link pradinio) ir naudodamiesi formule  $t_i^{(v)} = \min_j (t_j^{(v)} - t_{ij})$ . Ši formulė išplaukia iš tos aplinkybės, jog  $t_i^{(v)}$  yra toks įvykio „i“ įvykimo laiko momentas, nuo kurio pradėjus vykdyti darbą ( $i, j$ ) ir visus kitus po jo sekančius darbus, dar būtų galima suspėti užbaigti paskutinį duotojo komplekso darbą ne vėliau momento  $t_{\text{baig}}^{(v)}$ .

Apdorojant BTG informaciją „rankiniu būdu“, vėlyvieji įvykių „i“ įvykimo laikai  $t_i^{(v)}$  įrašomi į įvykius vaizduojančių skrituliukų dešiniuosius sektorius, kaip tai parodyta 157 „b“ pav.

Apdorojant BTG informaciją kompiuteriu, dydžių  $t_i^{(v)}$  apskaičiavimas vykdomas, naudojantis lenteliniu BTG modeliu ir jau minėtąja formule  $t_i^{(v)} = \min_j (t_j^{(v)} - t_{ij})$ . Kad apskaičiavimų metu būtų išvengta „šokinėjimo“ po visą darbų sąrašą, tikslinga visus darbus išrikiuoti sąrašė taip, kad visi darbai, turintys vienodus „i“ ir skirtingus „j“, būtų išsidėstę tiesiogiai vienas po kito. Tokį darbų sąrašo išrikiavimą vadiname „išrikiavimu nuo pradžios“.

„Išrikiavimo nuo pradžios“ algoritmas, o taip pat ir  $t_i^{(v)}$  apskaičiavimo kompiuteriu algoritmas aprašyti antroje šios mokymo priemonės dalyje.

Suradus  $t_i^{(a)}$  ir  $t_i^{(v)}$ , apskaičiuojami įvykių rezervai  $R_i = t_i^{(v)} - t_i^{(a)}$ , kurie įrašomi į žemutinį skrituliukų vaizduojančių įvykius, sektorių (žiūr. 157 „b“ pav.).

Užbaigus įvykių charakteristikų apskaičiavimą, pereinama prie darbų charakteristikų. Darbų ( $i, j$ ) pilnieji rezervai  $R_{i,j}^{(p)}$  apskaičiuojami, naudojantis formule  $R_{i,j}^{(p)} = t_j^{(v)} - t_i^{(a)} - t_{ij}$ . Šie rezervai vadinami **pilnaisiais**, todėl, kad, juos apskaičiuojant, laikoma, jog darbą ( $i, j$ ) leista baigti vėliausiu leistinu laiku  $t_j^{(v)}$ , o pradėti – anksčiausiu galimu laiku  $t_i^{(a)}$ , ir, jei tokiu atveju skirtumas  $t_j^{(v)} - t_i^{(a)}$  yra didesnis už  $t_{ij}$ , tai likęs rezervinis laikas ir sudaro patį didžiausią (pilnąjį) rezervą, kokį tas darbas gali turėti tame TG. Šis rezervas dažnai vadinamas **bendruoju**, nes jis priklauso ne tik darbui ( $i, j$ ), bet ir visai darbų grandinei, jungiančiai pradinį įvykį su baigiamuoju, į kurios sudėtį įeina tas darbas ( $i, j$ ).

Tokią grandinę vadiname **TG keliu**, ir  $R_{i,j}^{(p)}$  yra ne tik darbo  $(i,j)$ , bet ir kelio, į kurį įeina darbas  $(i,j)$  charakteristika.

Galima įrodyti (palieku tai skaitytojiui), kad jei darbo  $(i,j)$  bendrasis (pilnasis) rezervas  $R_{i,j}^{(p)}=0$ , tai tinkliniame grafike būtina egzistuoja kelias, kurį sudarančių visų darbų pilnieji rezervai yra lygūs 0. Tokį kelią vadina **TG kritiniu keliu**, ir jis tinkliniame grafe gali būti ne vienas. Taip pat galima įrodyti, kad bent vienas kritinis kelias kiekviename TG būtina egzistuoja.

Nesunku įrodyti, kad **kritinis kelias – tai ilgiausias kelias tinkliniame grafike**. Šią kritinio kelio savybę galima laikyti jo apibrėžimu ir toks apibrėžimas yra netgi bendresnis už apibrėžimą, kad TG kritiniu keliu vadinasi toks jo kelias, kuriame kiekvieno darbo pilnasis rezervas yra lygus 0. Taip yra todėl, kad **apibrėžimas, jog TG kritinis kelias yra ilgiausias TG kelias, tinka visais atvejais, o antrasis apibrėžimas – tik tada, kada, apskaičiuojant  $t_i^{(v)}$ , buvo naudojamas lygybė  $t_{baig}^{(v)} = t_{baig}^{(a)}$** .

Taip pat galima įrodyti, jog tuo atveju, kai tenkinama lygybė  $t_{baig}^{(v)} = t_{baig}^{(a)}$ , tai visų įvykių „i“, priklausančių kritiniam keliui, rezervai  $R_i=0$ . Šis požymis palengvina kritinių kelių tinkliniame grafe radimą, tačiau jis dar neleidžia visais atvejais vienareikšmiškai nuspręsti, ar darbas, jungiantis du įvykius, kurių rezervai lygūs 0, priklauso ar nepriklauso kritiniams keliui.

Pavyzdžiui, darbas  $(0,1)$ , priklausantis BTG, pavaizduotame 157 „b“ pav., priklauso kritiniam keliui, nes jo pilnasis rezervas  $R_{0,1}^{(p)} = t_1^{(v)} - t_0^{(a)} - t_{0,1} = 5 - 0 - 5 = 0$ . O tuo tarpu darbas  $(1,3)$ , nors ir jungia įvykius „1“ ir „3“, turinčius nulinius rezervus  $R_1=5-5=0$ ,  $R_3=14-14=0$ , nepriklauso kritiniam keliui, nes  $R_{1,3}^{(p)} = t_3^{(v)} - t_1^{(a)} - t_{1,3} = 14 - 5 - 6 = 3 > 0$ . Analogiškai randame, kad  $R_{0,2}^{(p)} = 1$ ,  $R_{1,2}^{(p)} = 0$ ,  $R_{2,3}^{(p)} = 0$ ,  $R_{3,4}^{(p)} = 0$ , t. y. kad 157 pav. TG turi tik vieną kritinį kelią, kurį sudaro darbai  $(0,1)$ ,  $(1,2)$ ,  $(2,3)$  ir  $(3,4)$ . To kritinio kelio ilgis  $\lambda$  lygus  $5+0+9+8=22$ , kai tuo tarpu kitų dviejų kelių ilgiai yra  $5+6+8=21$  ir  $4+9+8=21$ , t. y. trumpesnis už  $\lambda$ .

Kita svarbi darbų  $(i,j)$  charakteristika yra jų **laisvieji rezervai**  $R_{ij}^{(l)} = t_i^{(a)} - t_j^{(v)} - t_{ij}$ . Kaip matyti iš šios formulės, darbo  $(i,j)$  laisvuju rezervu vadinamas toks laiko rezervas, kuris išlieka net tada, kada darbas pradamas vėliausiu leistinu laiku ir turi būti baigtas anksčiausiu įmanomu laiku. Jeigu toks laiko rezervas vis dėlto kartais būna didesnis už 0, tai reiškia, jog tokį rezervą turinčio darbo rezervus galima panaudoti kituose darbuose, pavyzdžiui, permetant dalį darbo jėgos, vykdančios darbą, turintį teigiamą laisvąjį rezervą, darbo, priklausančio kritiniam keliui, vykdymui. Net ir tuo atveju, kai kokio nors darbo laisvasis rezervas yra neigiamas, ne būtina tam darbui reikalingi papildomi resursai, nes gali būti, kad tuo pačiu metu to darbo pilnasis rezervas yra teigiamas dydis. Nesunku įrodyti, kad darbų, priklausančių kritiniam keliui, laisvieji rezervai visada lygūs 0, o taip pat tai, kad jei kurio nors darbo laisvasis rezervas yra neigiamas, tai jo pilnasis rezervas – teigiamas dydis.

Apskaičiavę darbų, priklausančių TG, pavaizduotame 157 pav., laisvuosius rezervus, gauname  $R_{0,1}^{(l)} = 0$ ,  $R_{0,2}^{(l)} = 1$ ,  $R_{1,2}^{(l)} = 0$ ,  $R_{1,3}^{(l)} = 3$ ,  $R_{2,3}^{(l)} = 0$ ,  $R_{3,4}^{(l)} = 0$ .

Kartais apskaičiuojami ir TG darbų **daliniai rezervai**  $R'_{i,j} = t_j^{(v)} - t_i^{(v)} - t_{ij}$  ir  $R''_{i,j} = t_j^{(a)} - t_i^{(a)} - t_{ij}$ . Kaip matyti iš šių rezervų apskaičiavimo formulių, daliniai rezervai (kaip ir  $R_{i,j}^{(p)}$ ) negali tapti neigiamais dydžiais, tačiau negali viršyti  $R_{i,j}^{(p)}$ .

Nesunku įrodyti, jog kritinio kelio darbams  $(i,j)$  galioja lygybės  $R_{i,j}^{(p)} = R_{i,j}^{(l)} = R'_{i,j} = R''_{i,j} = 0$ . Palieku šio teiginio bei eilės kitų šiame skyrelyje suformuluotų teiginių įrodymą skaitytojiui, o taip pat ir formulių, išreiškiančių priklausomybes tarp darbų ir įvykių rezervų (pavyzdžiui,  $R_{i,j}^{(p)} - R_{i,j}^{(l)} = R_j + R_i$ ,  $R_{i,j}^{(p)} + R_{i,j}^{(l)} = R'_{i,j} + R''_{i,j}$  ir t. t.), išvedimą.

Pilnesnę informaciją apie TG charakteristikas ir jų apskaičiavimo metodus skaitytojas gali rasti antroje šios mokymo priemonės dalyje.

### 1.9.2.18. TG optimizavimo operacijos

**TG optimizavimu** vadinamas toks iš išorės gaunamų ir viduje turimų resursų (rezervų) perskirstymas, kurio dėka gaunamas naujas TG, turintis geresnes charakteristikas.

Optimizuojant TG, gali keistis šios jo charakteristikos:

1) gali būti trumpinamas TG kritinio kelio ilgis  $\lambda$ , t. y. gali būti trumpinama viso darbų komplekso įvykdymo trukmė;

2) gali būti mažinama išlaidų, reikalingų darbų komplekso įvykdymui, suma, nepabloginant kitų TG charakteristikų;

3) gali būti didinamos tikimybės  $p_{ij}$ , kad numatyti darbai  $t_{ij}$  bus įvykdyti ne vėliau TG nustatytų terminų, t. y. gali būti didinama tikimybė, kad visas darbų kompleksas bus įvykdytas ne vėliau nustatyto (direktyvinio) termino.

Egzistuoja gana daug TG optimizavimo sistemų. Vienos jų leidžia pagerinti kurią nors vieną iš čia išvardintų charakteristikų ( $\lambda$ ,  $C$ ,  $p_{ij}$ ,  $P$ ), išlaikant kitas charakteristikas tam tikrose ribose, tačiau egzistuoja ir tokios sistemos, kuriomis naudojantis galima palaipsniui gerinti ne tik kurią nors vieną, bet ir kelias TG charakteristikas.

Su sistemų ir procesų optimizavimo metodais skaitytojas bus supažindintas šios mokymo priemonės 1.15. skyriuje, o antrojoje jos dalyje bus nagrinėjamos ir specialiai TG optimizavimui skirtos sistemos CPM ir PERT.

**PERT metodas** taikomas informacijos apdorojimui tokiuose TG, kurių darbų trukmės  $t_{ij}$  yra atsitiktiniai dydžiai. Šis metodas, ekspertams nustačius  $(i,j)$  trukmių  $t_{ij}$  optimistines  $t_{i,j}^{(opt)} = a_{ij}$ , pesimistines  $t_{i,j}^{(pes)} = b_{ij}$  ir labiausiai tikėtinas reikšmes  $t_{i,j}^{(tik)} = m_{ij}$ , leidžia apskaičiuoti vidutines darbų įvykdymo trukmes  $\overline{t_{ij}}$ , jų dispersijas, darbų įvykdymo nustatytais terminais tikimybes ir po to, perskirstant turimus resursus, didinti tas tikimybes bei trumpinti viso darbų komplekso kritinio kelio ilgį  $\lambda$ .

**CPM („kritinio kelio metodas“)** taikomas determinuotiems TG, kai žinomos darbų  $(i,j)$  įvykdymo kainų  $c_{ij}$  priklausomybės nuo darbų  $t_{ij}$  trukmių, o taip pat žinomos tų trukmių ribinės reikšmės  $a_{ij}$  ir  $b_{ij}$ , kur  $a_{ij} \leq t_{ij} \leq b_{ij}$ .

Naudojantis CPM ir žinant duotąją išlaidų sumą  $C$  (arba papildomą išlaidų sumą  $\Delta C$ ), galima apskaičiuoti minimalų kritinio kelio ilgį  $\lambda$ , kurį galima pasiekti už tą sumą  $C$  (arba maksimalų to kelio sutrumpinimą  $\Delta \lambda$ , kainuojantį lygiai  $\Delta C$ ), o taip pat išspręsti ir atvirkštinį uždavinį: žinant norimą kritinio kelio ilgį  $\lambda$ , apskaičiuoti minimalias išlaidas  $C$ , reikalingas viso komplekso darbų įvykdymui per duotąjį laiką (arba minimalią papildomų išlaidų sumą  $\Delta C$ , reikalingą kritinio kelio sutrumpinimui duotuoju dydžiu  $\Delta \lambda$ ).

Optimizuojant TG, gali būti perskirstomos ne tik darbų vykdymui skirtos lėšos, bet ir darbo jėga bei darbų vykdymui išskirtos techninės priemonės.

Konkretūs tokio optimizavimo pavyzdžiai (pavyzdžiui, kepyklos cecho rekonstrukcijos TG), variacinio skaičiavimo metodų taikymai ir pan., bus išnagrinėti antrojoje šios mokymo priemonės dalyje, kurią studijuodami studentai ne tik susipažins su CPM ir PERT metodais, bet ir privalės sukurti taikomųjų programų paketus, sprendžiančius visus 1.9.2.16., 1.9.2.17. ir 1.9.2.18. skyreliuose išvardintus uždavinius.

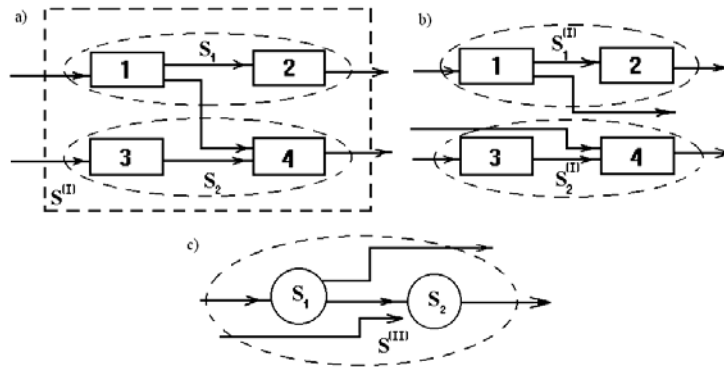
### 1.9.2.19. Struktūros dekompozicijos operacija

Šios operacijos reikiamybė sąlygojama tuo, jog sistemos, turinčios daug elementų, gali tapti **neapžvelgiamomis**, ir mes galime susidurti su **pirmuoju informaciniu barjeru** (žiūr. 1.6.4.13. skyrelį). Išeitį iš tokios padėties gali duoti **sistemos dekompozicija į mažesnius posistemius**, išskirtų posistemių išorinių modelių sudarymas ir po to visos sistemos vidinio modelio sudarymas posistemių, traktuojamų kaip sistemos elementai, lygyje. Tokiu būdu, vietoje **vienaešeloninės** sunkiai apžvelgiamos sudėtingos sistemos struktūros  $S$  modelio, gauname **dviešeloninį tos struktūros modelį**, kuris pritaikytas tam, kad **struktūrą apžvelgtume dalimis ir dviem pakopomis**. Jei sistemos struktūra tokia sudėtinga, jog ir išskirti posistemiai – sudėtingos, sunkiai apžvelgiamos sistemos (o daryti daug smulkių posistemių netikslinga, nes tuomet sistemos modelis posistemių lygyje taps neapžvelgiamu), tai išskirtieji pirmojo lygio posistemiai dekomponuojami į smulkesnius antrojo lygio posistemius, šie savo ruožtu gali būti dekomponuoti į trečiojo lygio posistemius ir t. t., **kol bus pasiekti tokie posistemiai, kuriuos bus galima aprašyti jų išoriniais modeliais ir traktuoti kaip elementus**. Tokiu būdu mes gausime **daugiaėšeloninį sudėtingos sistemos struktūros modelį**, apžvelgiamą dalimis ir pakopomis. Kad sugebėtume sudaryti daugiaėšeloninį sistemos struktūros modelį, pakanka išmokyti sudaryti dviešeloninį modelį, nes **daugiaėšeloninio modelio sudarymo procesas dekomponuojasi į dviešeloninių modelių sudarymo procesus**.

**Sistemos vienaėšeloninės struktūros dekompozicija ir tos struktūros dviešeloninio modelio sudarymas gali būti vykdomi grafinėje, matricinėje, operatorinėje ir analizinėje formose (stratose [18]).** Pailiustruosiu tokią dekompoziciją konkrečiais pavyzdžiais.

Tarkime, jog turime sistemos struktūrą  $S$ , susidedančią iš 4 elementų ir pavaizduotą 158 „a“ pav. Dekomponuosime šią sistemą į du posistemius  $S_1$  ir  $S_2$  taip, kad į posistemį  $S_1$  patektų pirmasis ir

antrasis elementai, o į posistemį  $S_2$  – trečiasis ir ketvirtasis. (žiūr. 158 „b“ pav.). Gausime dviešeloninę sistemą  $S^{(II)}$ , susidedančią iš „elementų“ – posistemių  $S_1^{(I)}$  ir  $S_2^{(I)}$ , kurie patys yra sistemos, susidedančios iš elementų 1 ir 2 bei 3 ir 4 (žiūr. 158 „c“ ir 158 „b“ pav.)



158 pav.

Pavaizduotą 158 pav. **grafinę sistemos S dekompoziciją** bei dviejų pirmojo lygio ir vienos antrojo lygio struktūrų  $S_1^{(I)}$ ,  $S_2^{(I)}$ ,  $S^{(II)}$  sudarymą galima išreikšti ir **matricinėje formoje** (žiūr. 159 pav., kur numeriais „0“, „(01)“ ir „(02)“ pažymėta sistemos S ir posistemių  $S_1$  ir  $S_2$  išorinė aplinka).

|                  |   |   |   |   |   |   |
|------------------|---|---|---|---|---|---|
| $S^{(I)} \equiv$ |   | 0 | 1 | 2 | 3 | 4 |
|                  | 0 | 0 | 1 | 0 | 1 | 0 |
|                  | 1 | 0 | 0 | 1 | 0 | 1 |
|                  | 2 | 1 | 0 | 0 | 0 | 0 |
|                  | 3 | 0 | 0 | 0 | 0 | 1 |
|                  | 4 | 1 | 0 | 0 | 0 | 0 |

|                    |      |   |   |      |
|--------------------|------|---|---|------|
| $S_1^{(I)} \equiv$ |      | 1 | 2 | (01) |
|                    | 1    | 0 | 1 | 1    |
|                    | 2    | 0 | 0 | 1    |
|                    | (01) | 1 | 0 | 0    |

|                    |      |   |   |      |
|--------------------|------|---|---|------|
| $S_2^{(I)} \equiv$ |      | 3 | 4 | (02) |
|                    | 3    | 0 | 1 | 0    |
|                    | 4    | 0 | 0 | 1    |
|                    | (02) | 1 | 1 | 0    |

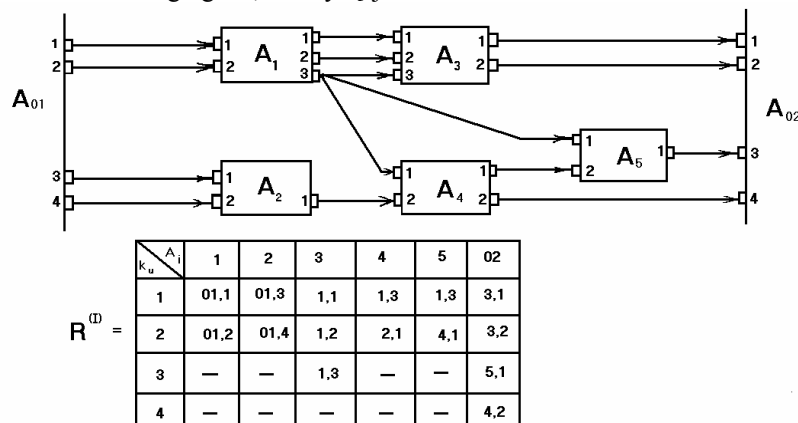
|                   |       |       |       |   |
|-------------------|-------|-------|-------|---|
| $S^{(II)} \equiv$ |       | $S_1$ | $S_2$ | 0 |
|                   | $S_1$ | 0     | 1     | 1 |
|                   | $S_2$ | 0     | 0     | 1 |
|                   | 0     | 1     | 1     | 0 |

159 pav.

Kaip jau buvo minėta 1.7.7. skyrelyje, didelių sistemų struktūrų aprašymas matricinėje formoje dažnai nėra efektyvus, nes matricioje daugumoje atvejų daug vietos užima tušti („nuliniai“) langeliai. Efektyvesnė aprašymo priemonė – agregatų jungimo operatoriai (ryšių operatoriai), taip pat aprašyti 1.7.7. skyrelyje.

Ivykdysime sistemos struktūros dekompoziciją **operatorinėje formoje**, iliustruodami tą dekompoziciją grafinėje formoje.

Tarkime, jog mums reikia dekomponuoti struktūrą  $S^{(I)}$ , pavaizduotą 160 pav. grafinėje ir operatorinėje formose, kur  $A_{01}$  ir  $A_{02}$  – agregatai, imituojuojantys sistemos išorinę aplinką, o  $A_1, A_2, A_3, A_4$  ir  $A_5$  – vidiniai sistemos  $S^{(I)}$  agregatai, turintys įėjimo kontaktus k.

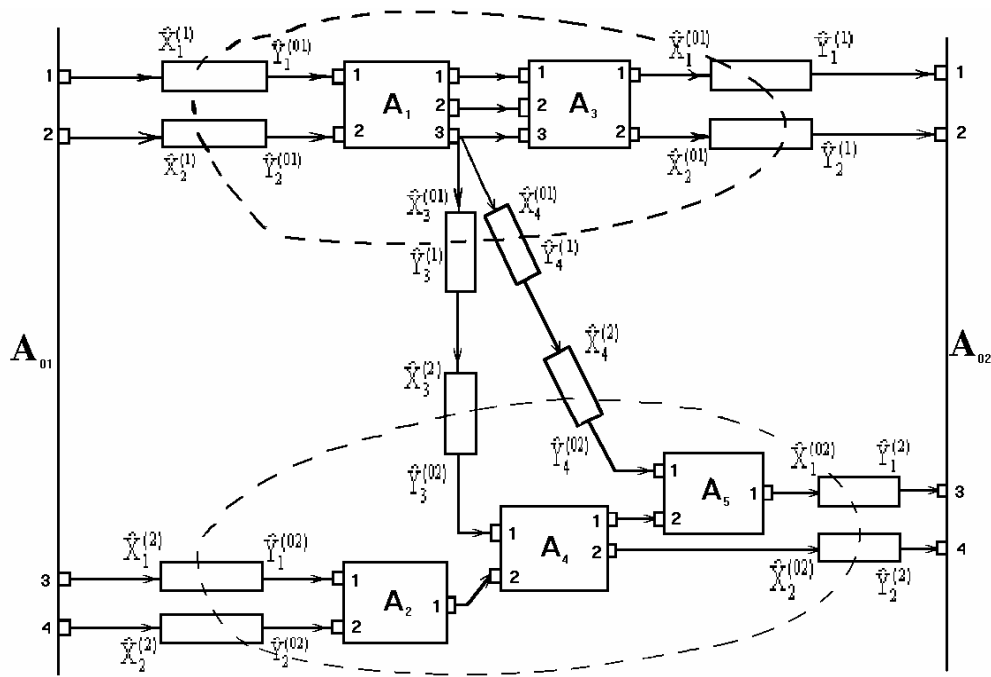


160 pav.

Tarkime, kad struktūrą  $S^{(I)}$  reikia dekomponuoti į du posistemius. Į pirmąjį posistemį, kurio struktūrą pažymėsime identifikatoriumi  $S_1^{(I)}$ , įjungsime agregatus  $A_1$  ir  $A_3$ , į antrąjį –  $S_2^{(I)}$  – agregatus  $A_2, A_4$  ir  $A_5$ .

Gautus posistemius  $S_1^{(I)}$  ir  $S_2^{(I)}$  traktuosime dvejopai: 1) kaip **naujas savarankiškas sistemas**, susidedančias iš agregatų  $A_i$ ; 2) kaip sistemos  $S$  (ir sistemos  $S^{(II)}$ , kas yra tas pats) **antrojo lygio elementus**, sudarančius sistemą  $S^{(II)}$ .

Sudarysime pirmojo posistemio agregatų jungimo operatorių  $R_1^{(I)}$ , žiūrėdami į jį kaip į sistemą, susidedančią iš agregatų  $A_1$  ir  $A_3$ . Visa, kas neįeina į šį posistemį, vadinsim jo aplinka  $A_{(01)}$ . Įvesime šio posistemio sąveikos su jo aplinka  $A_{(01)}$  **fiktyvius dvigubus kontaktus** (žiūr. 161 pav.) – „muitinių postus“.



161 pav.

Šiuos kontaktus vadinsime fiktyviais todėl, kad jų nėra bei nebuvo dekomponuojamoje pradinėje sistemoje  $S$  ir jie sukuriama tik dekompozicijos metu. Juos taip pat vadinsime dvigubais, nes kiekvienas iš jų yra kartu ir įėjimo ir išėjimo kontaktas. Fiktyvius dvigubus kontaktus žymėsime raidėmis  $X$  ir  $Y$  su „stogeliais“.

Pavyzdžiui, kontaktai  $\hat{X}_1^{(1)}$  ir  $\hat{X}_2^{(2)}$  yra pirmojo posistemio, kaip sistemos  $S^{(II)}$  antrojo lygio elemento, įėjimo kontaktai. Tuo pačiu metu šie kontaktai, pažymėti kaip  $\hat{Y}_1^{(01)}$  ir  $\hat{Y}_2^{(01)}$  yra pirmojo posistemio aplinkos  $A_{(01)}$  išėjimo kontaktai, nuo kurių iš aplinkos  $A_{(01)}$  poveikiai patenka į pirmąjį posistemį, traktuojamą kaip savarankiška sistema, susidedanti iš agregatų  $A_1$  ir  $A_3$ .

Analogiškai gauname, kad kontaktai  $\hat{Y}_1^{(1)}$ ,  $\hat{Y}_2^{(1)}$ ,  $\hat{Y}_3^{(1)}$ ,  $\hat{Y}_4^{(1)}$  yra pirmojo posistemio – sistemos  $S^{(II)}$  antrojo lygio elemento – išėjimo kontaktai. Tuo pačiu metu šie kontaktai, kaip kontaktai  $\hat{X}_1^{(01)}$ ,  $\hat{X}_2^{(01)}$ ,  $\hat{X}_3^{(01)}$ ,  $\hat{X}_4^{(01)}$ , yra pirmojo posistemio aplinkos  $A_{(01)}$  įėjimo kontaktai.

Pirmojo posistemio agregatų jungimo operatorius  $R_1^{(I)}$  turės tris stulpelius ir keturias eilutes (žiūr. 48 lentelę). Analogiškai gausime ir operatorių  $R_2^{(I)}$  (žiūr. 49 lentelę).

48. lentelė

|   | 1      | 3   | (01) |
|---|--------|-----|------|
| 1 | (01),1 | 1,1 | 3,1  |
| 2 | (01),2 | 1,2 | 3,2  |
| 3 | –      | 1,3 | 1,3  |
| 4 | –      | –   | 1,3  |

$R_1^{(I)} =$

49. lentelė

|   | 2      | 4      | 5      | (02) |
|---|--------|--------|--------|------|
| 1 | (02),1 | (02),3 | (02),4 | 5,1  |
| 2 | (02),2 | 2,1    | 4,1    | 4,2  |

$R_2^{(I)} =$

Žiūrėdami į dekompozicijos metu išskirtus posistemius kaip į sistemos  $S^{(II)}$  antrojo lygio elementus  $S_1^{(I)}$  ir  $S_2^{(I)}$ , sudarysime jų jungimo į sistemą  $S^{(III)}$  operatorių, t. y. operatorių  $R^{(III)}$  (žiūr. 50 lentelę):

50. lentelė

|   |       |               |               |
|---|-------|---------------|---------------|
|   | $S_1$ | $S_2$         | 02            |
| 1 | 01,1  | 01,3          | $S_1^{(I)},1$ |
| 2 | 01,2  | 01,4          | $S_1^{(I)},2$ |
| 3 | –     | $S_1^{(I)},3$ | $S_2^{(I)},1$ |
| 4 | –     | $S_1^{(I)},4$ | $S_2^{(I)},2$ |

Dekomponuoti sistemą į posistemius galima ir neturint struktūros S grafo. Tereikia tik turėti operatorių  $R^{(I)}$  ir posistemių sudėtį.

Tarkime, turime tokį  $R^{(I)}$  (žiūr. 51 lentelę).

51. lentelė

| $k_u \backslash A_j$ | 1    | 2   | 3   | 4   | 5   | 6    | 7   | 8   | 02  |
|----------------------|------|-----|-----|-----|-----|------|-----|-----|-----|
| 1                    | 01,1 | 1,1 | 2,1 | 3,1 | 4,1 | 01,3 | 6,2 | 1,2 | 5,1 |
| 2                    | 01,2 | 1,2 | 2,2 | 3,2 | 4,2 | 01,4 | 6,3 | 7,1 | 5,2 |
| 3                    | 01,3 | 6,1 | 2,3 | 8,1 | 4,3 | –    | –   | 7,2 | 4,3 |
| 4                    | –    | –   | –   | 8,2 | –   | –    | –   | 6,3 | 8,2 |

Tarkime, kad į pirmąją sistemos posistemį jungiami agregatai  $A_1$  ir  $A_2$ , į antrąją –  $A_3, A_4, A_5$  į trečiąją –  $A_6, A_7$  ir  $A_8$ .

Tuomet gausime tokius jungimo operatorius  $R_1^{(I)}, R_2^{(I)}, R_3^{(I)}, R^{II}$  (žiūr. 52÷55 lenteles).

52. lentelė  $R_1^{(I)}$

| $k_u \backslash A_j$ | 1      | 2      | (01) |
|----------------------|--------|--------|------|
| 1                    | (01),1 | 1,1    | 2,1  |
| 2                    | (01),2 | 1,2    | 2,2  |
| 3                    | (01),3 | (01),4 | 2,3  |
| 4                    | –      | –      | 1,2  |

53. lentelė  $R_2^{(I)}$

| $k_u \backslash A_j$ | 3      | 4      | 5   | (02) |
|----------------------|--------|--------|-----|------|
| 1                    | (02),1 | 3,1    | 4,1 | 5,1  |
| 2                    | (02),2 | 3,2    | 4,2 | 5,2  |
| 3                    | (02),3 | (02),4 | 4,3 | 4,3  |
| 4                    | –      | (02),5 | –   | –    |

54. lentelė  $R_3^{(I)}$

| $k_u \backslash A_j$ | 6      | 7   | 8      | (03) |
|----------------------|--------|-----|--------|------|
| 1                    | (03),1 | 6,2 | (03),3 | 6,2  |
| 2                    | (03),2 | 6,3 | 7,1    | 8,1  |
| 3                    | –      | –   | 7,2    | 8,2  |
| 4                    | –      | –   | 6,3    | 8,3  |

55. lentelė  $R^{II}$

| $k_u \backslash S_j^{(I)}$ | $S_1^{(I)}$   | $S_2^{(I)}$   | $S_3^{(I)}$   | 02            |
|----------------------------|---------------|---------------|---------------|---------------|
| 1                          | 01,1          | $S_1^{(I)},1$ | 01,3          | $S_2^{(I)},1$ |
| 2                          | 01,2          | $S_1^{(I)},2$ | 01,4          | $S_2^{(I)},2$ |
| 3                          | 01,3          | $S_1^{(I)},3$ | $S_1^{(I)},4$ | $S_2^{(I)},3$ |
| 4                          | $S_3^{(I)},1$ | $S_3^{(I)},2$ | –             | $S_3^{(I)},4$ |
| 5                          | –             | $S_3^{(I)},3$ | –             | –             |

Aprašysiu dekompozicijos algoritmą operatorinėje formoje.

**Operatorių  $R_1^{(I)}, R_2^{(I)}, R_3^{(I)}$  ir  $R^{II}$  gavimo algoritmas iš operatoriaus  $R^{(I)}$**

1) Kadangi į pirmąją posistemį patenka agregatai  $A_1$  ir  $A_2$ , tai žiūrime į operatoriaus  $R^{(I)}$  stulpelius 1 ir 2 ir sudarome operatoriaus  $R_1^{(I)}$  stulpelius 1 ir 2. Šiuos stulpelius užpildome taip: jei

operatoriaus  $R^{(I)}$  stulpelių 1 ir 2 langeliuose užrašyti vidiniai posistemo  $S_1^{(I)}$  adresai, tai tuos adresus perkeliame į atitinkamus operatoriaus  $R_1^{(I)}$  stulpelių 1 ir 2 langelius, jei operatoriaus  $R^{(I)}$  stulpelių langeliuose užrašyti išoriniai posistemio  $S_1$  adresai, tai tuos adresus eilės tvarka pakeičiame posistemo išorinės aplinkos  $A_{(01)}$  kontaktų adresais: (01),1; (01),2 ir t. t.

2) Operatoriaus  $R_1^{(I)}$  stulpelį (01), atitinkantį posistemo  $S_1^{(I)}$  išorinės aplinkos  $A_{(01)}$  įėjimą užpildome taip: likusiuose operatoriaus  $R^{(I)}$  stulpeliuose (t. y. stulpeliuose 3, 4, 5, 6, 7, 8, 02) ieškome agregatų 1 ir 2 išėjimo kontaktų adresų ir juos eilės tvarka surašome į operatoriaus  $R_1^{(I)}$  stulpelį (01).

3) Kartojame algoritmo etapus 1 ir 2, sudarydami operatorių  $R_2^{(I)}$ , t. y. išskiriame operatorių  $R^{(I)}$  stulpelius 3, 4, 5 ir sudarome operatoriaus  $R_2^{(I)}$  stulpelius 3, 4, 5, (02) ir t. t.

4) Analogiškai sudarome operatorių  $R_3^{(I)}$ .

5) Sudarome operatoriaus  $R^{(II)}$  stulpelius, atitinkančius sistemos posistemius  $S_1^{(I)}$ ,  $S_2^{(I)}$ ,  $S_3^{(I)}$  ir jos išorinės aplinkos 02 įėjimą.

Stulpelį  $S_1^{(I)}$  užpildome taip: ieškome operatorių  $R_1^{(I)}$  langelių, atitinkančių posistemo  $S_1^{(I)}$  įėjimus iš jo aplinkos, (01) ir žiūrime, ką jie atitinka operatorių  $R^{(I)}$ . Jei jie atitinka sistemos  $S$  aplinkos 01 išėjimus, tai perrašome jų adresus į stulpelį  $S_1^{(I)}$ , o jei jie atitinka kokių nors sistemos  $S$  agregatų išėjimus, tai žiūrime, kokiam posistemio tas agregatas priklauso, ir to posistemo operatorių surandame, kokį to posistemo išėjimą į jo aplinką tas agregato išėjimas atitinka. To išėjimo adresą ir užrašome stulpelyje  $S_1^{(I)}$ .

Analogiškai užpildome stulpelius  $S_2^{(I)}$  ir  $S_3^{(I)}$ .

Operatoriaus  $R^{(II)}$  stulpelį 02 užpildome, žiūrėdami į operatoriaus  $R^{(I)}$  stulpelį 02 ir tik pakeisdami agregatų išėjimo kontaktų adresus į atitinkamų posistemių atitinkamų išėjimo kontaktų adresus.

Rekomenduojame skaitytojui, naudojantis šiuo algoritmu, savarankiškai gauti 52÷55 lenteles.

Išnagrinėsime **analizinės matricinės dekompozicijos pavyzdį**, paimtą iš knygos [112].

Tarkime, kad loginės schemos  $S$  išėjimo funkcija yra  $y = f(x_4, x_3, x_2, x_1) = \overline{x}_4 \overline{x}_2 \overline{x}_1 \vee \overline{x}_4 x_3 \overline{x}_2 \vee \overline{x}_4 \overline{x}_3 x_2 x_1 \vee x_4 \overline{x}_3 \overline{x}_2 x_1 \vee x_4 x_2 \overline{x}_1 \vee x_4 x_3 x_2$ .

Kadangi funkcija turi 4 argumentus, tai jos maksimalus galimas sudėtingumas lygus 16 (tai galima gauti ir vertinant sudėtingumą maksimaliu argumentų galimų reikšmių kombinacijų skaičiumi  $2^m = 2^4 = 16$ , ir maksimaliu galimų vienpusių ryšių tarp elementų skaičiumi  $\approx m^2 = 4^2 = 16$ ).

Tarkime, kad toks sudėtingumas mums atrodo didelis (žinoma, tikrovėje toks sudėtingumas dar nėra didelis, tačiau tam, kad paaiškintume analizinės matricinės dekompozicijos metodą, nebūtina nagrinėti sudėtingą sistemą, kas užimtų daug vietos ir laiko; todėl ir pavartotas posakis „Tarkime, kad toks sudėtingumas mums atrodo didelis“). Pabandysime sumažinti sistemos modelio sudėtingumą, dekomponuodami jį.

Suskaidysime funkcijos argumentų aibę į poaibius. Pavyzdžiui, sudarysime du poaibius  $(x_4, x_2)$  ir  $(x_3, x_1)$ . Ir pirmame ir antrame poaibiuose galimos 4 skirtingos argumentų reikšmių kombinacijos: (0, 0); (0, 1); (1, 0); (1, 1). Sunumeruosime jas: 0, 1, 2, 3. Sudarysime funkcijos  $f(x_4, x_3, x_2, x_1)$  reikšmių priklausomybės nuo šių numerių matricą (žiūr. 56. lentelę), įrašydami į matricos langelius funkcijos  $f(x_4, x_3, x_2, x_1)$  reikšmes, atitinkančias tas jos argumentų reikšmes, kurias nurodo tų reikšmių kombinacijos numeriai.

**56. lentelė**

|                                                 |       |                                                 |   |   |   |
|-------------------------------------------------|-------|-------------------------------------------------|---|---|---|
|                                                 |       | $x_3 \ x_1$ reikšmių kombinacijų numeriai $n_1$ |   |   |   |
| $x_4 \ x_2$ reikšmių kombinacijų numeriai $n_2$ | $n_1$ | 0                                               | 1 | 2 | 3 |
|                                                 | 0     | 1                                               | 0 | 1 | 1 |
|                                                 | 1     | 0                                               | 1 | 0 | 0 |
|                                                 | 2     | 0                                               | 1 | 0 | 0 |
|                                                 | 3     | 1                                               | 0 | 1 | 1 |

Kiekvieną šios matricos stulpelį atitinka pastovios argumentų  $x_3$  ir  $x_1$  reikšmės, o kiekvieną jos eilutę – pastovios argumentų  $x_4$  ir  $x_2$  reikšmės.

Kadangi matricoje randame tik dviejų skirtingų tipų stulpelius, tai juos galime aprašyti viena argumentų  $x_3$  ir  $x_1$  logine funkcija  $\varphi_1(x_3, x_1) = \varphi_1(n_1)$ , kurios reikšmė visam stulpeliui bus pastovi, nes stulpelyje nesikeičia argumentų  $x_3$  ir  $x_1$  reikšmės.

Tarkime, kad  $\varphi_1(0) = \varphi_1(2) = \varphi_1(3) = 0$ , o  $\varphi_1(1) = 1$ , t. y.  $\varphi_1(\bar{x}_3, \bar{x}_1) = \varphi_1(x_3, \bar{x}_1) = \varphi_1(x_3, x_1) = 0$ , o  $\varphi_1(\bar{x}_3, x_1) = 1$ .

Iš čia gauname, kad funkcijos  $\varphi_1(x_3, x_1)$  TDNF yra  $\varphi_1(x_3, x_1) = \bar{x}_3 x_1$ .

Analogiškai randame, kad matricos eilutes galima aprašyti viena funkcija  $\varphi_2(x_4, x_2) = \varphi_2(n_2)$ . Pasirinkę  $\varphi_2$  taip, kad  $\varphi_2(0) = \varphi_2(3) = 0$ , o  $\varphi_2(1) = \varphi_2(2) = 1$ , gausime, jog funkcijos  $\varphi_2(x_4, x_2)$  TDNF yra  $\varphi_2(x_4, x_2) = \bar{x}_4 x_2 \vee x_4 \bar{x}_2$ .

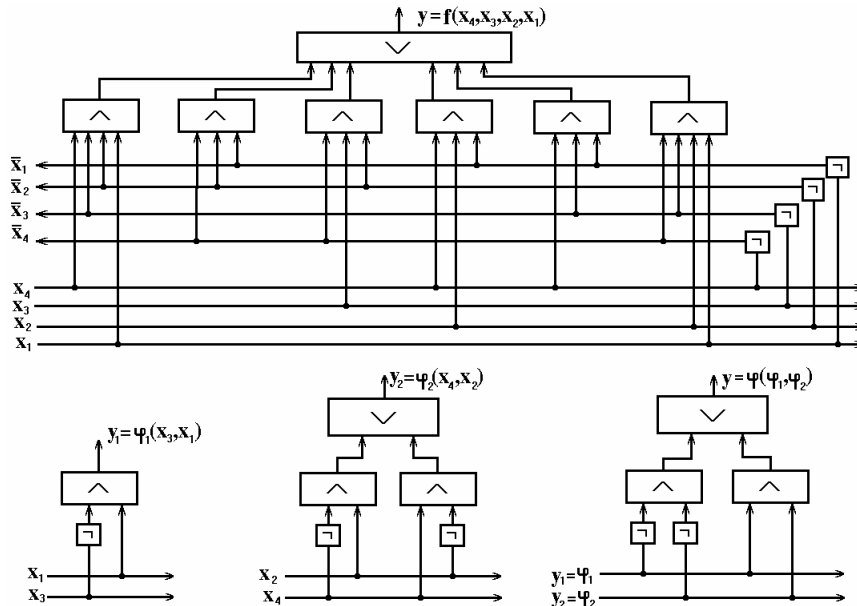
Jei nevienodų eilučių ar stulpelių rūšių būtų daugiau negu po 2, tai jų aprašymui reiktų daugiau loginių funkcijų  $\varphi_i$ . Pavyzdžiui, jei stulpelių rūšių skaičius būtų lygus  $k$ , tai stulpelių aprašymui reiktų  $r = \log_2 k$  funkcijų, kai  $\log_2 k$  – sveikas skaičius, ir  $r = E(\log_2 k) + 1$  funkcijų, kai  $\log_2 k$  – skaičius, turintis kokią nors sveikąją dalį  $E(\log_2 k)$  ir nelygią nuliui trupmeninę dalį  $(\log_2 k - E(\log_2 k))$ .

Sudarę posistemių funkcijas  $\varphi_1(x_3, x_1)$  ir  $\varphi_2(x_4, x_2)$ , sudarysime funkcijos  $y = f(x_4, x_3, x_2, x_1)$  struktūrą  $S^{(III)}$  naujų argumentų  $\varphi_1$  ir  $\varphi_2$  – lygyje, t. y. rasime funkciją  $\varphi(\varphi_1, \varphi_2) = f(x_4, x_3, x_2, x_1)$ .

Nagrinėdami funkcijos  $f(x_4, x_3, x_2, x_1)$  reikšmių matricą, pavaizduotą 56-ojoje lentelėje, ir prisimindami pasirinktas funkcijų  $\varphi_1(x_3, x_1)$  ir  $\varphi_2(x_4, x_2)$  reikšmes tos matricos stulpeliams ir eilutėms, gauname, jog  $\varphi(\varphi_1, \varphi_2) = 1$ , kai  $\varphi_1 = \varphi_2$ , ir  $\varphi(\varphi_1, \varphi_2) = 0$ , kai  $\varphi_1 \neq \varphi_2$ . Iš čia išvedame, kad funkcijos  $\varphi(\varphi_1, \varphi_2)$  TDNF yra  $\varphi(\varphi_1, \varphi_2) = \bar{\varphi}_1 \bar{\varphi}_2 \vee \varphi_1 \varphi_2$ . Tikrai, kai  $\varphi_1 = \varphi_2 = 0$ , tai gauname 56-osios lentelės nulinio, antrojo ir trečiojo stulpelių nulines ir trečiosios eilučių elementus, lygius 1. Tokias pačias elementų reikšmes gauname ir kai  $\varphi_1 = \varphi_2 = 1$ , t. y. pirmojo stulpelio pirmojoje ir antrojoje eilutėse. Visi kiti 56-osios lentelės elementai lygūs 0.

Palyginę šį dekompozicijos metodą su agregatų jungimo operatoriaus  $R^{(I)}$  dekompozicija, matome, kad **funkcija  $f(x_4, x_3, x_2, x_1)$  atitinka visos sistemos pirmojo lygio operatorių  $R^{(I)}$ , funkcijos  $\varphi_1(x_3, x_1)$  ir  $\varphi_2(x_4, x_2)$  – sistemos posistemių pirmojo lygio operatorius  $R_1^{(I)}$  ir  $R_2^{(I)}$ , funkcija  $\varphi(\varphi_1, \varphi_2)$  – visos sistemos antrojo lygio operatorių  $R^{(III)}$ .**

Funkcija  $y = f(x_4, x_3, x_2, x_1)$  ir jos dekompozicijos rezultatai grafiškai pavaizduoti 162 pav.



162 pav.

Analizuodami ir lygindami 162 pav. schemas, matome, kad dekompozicijos rezultatai paprastesni ir lengviau apžvelgiami negu nedekomponuota schema. Tai patvirtina ir sudėtingumo įverčiai: vietoj vienos 4 argumentų funkcijos gavome 3 dviejų argumentų funkcijas, kurių bendras maksimalus sudėtingumas  $3 \cdot 2^2 = 12 < 16$ .

**Dekompozicijos operacija duoda vienareikšmį rezultatą tik tuomet, kai nurodoma posistemių sudėtis.** Pavyzdžiui, vienareikšmė yra struktūros  $S$ , pavaizduotos 160 ir 161 pav., dekompozicija, kurią formaliai galima užrašyti taip:

$$D[S^{(I)}(A_1, A_2, A_3, A_4, A_5)] \cong S^{(II)}(S_1^{(I)}(A_1, A_3), S_2^{(I)}(A_2, A_4, A_5)),$$

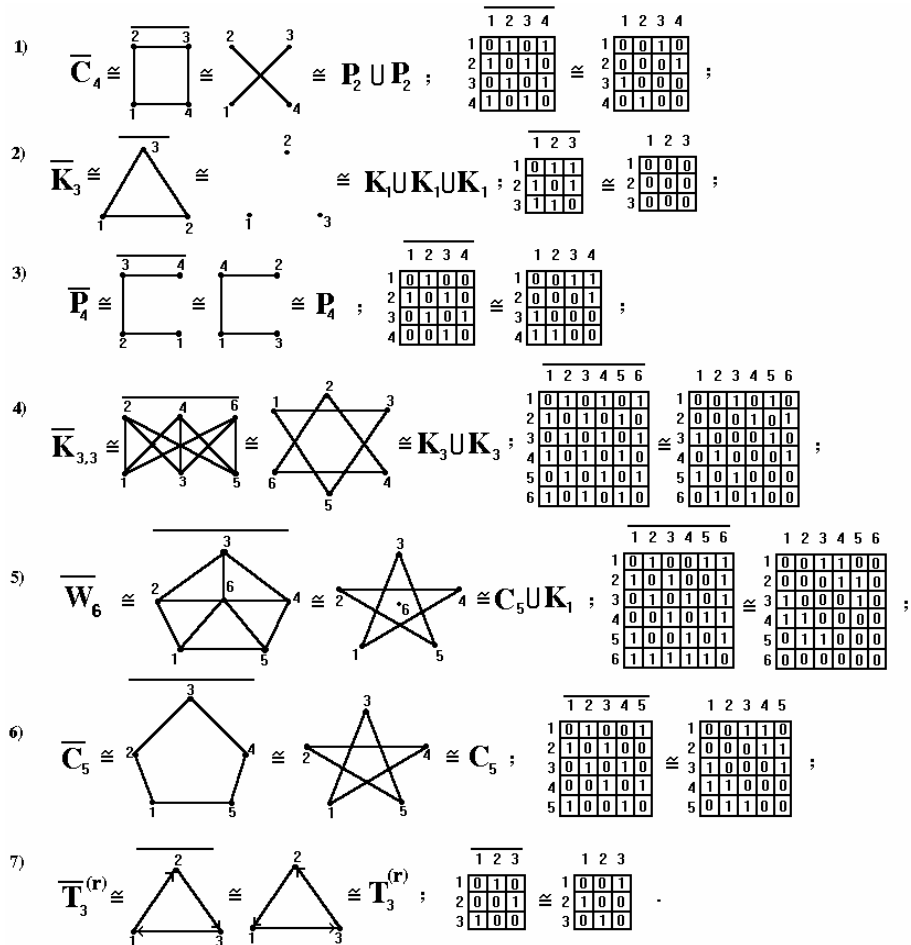
kur  $D$  – dekompozicijos operacijos ženklas,  $S^{(I)}$  ir  $S^{(II)}$  – sistemos  $S$  vienaeseloninė ir jos antrojo ešelono struktūra,  $S_1^{(I)}$  ir  $S_2^{(I)}$  – sistemos  $S$  posistemių vienaeseloninės struktūros.

Jei, prieš dekomponuojant sistemą  $S$ , jos būsimų posistemių skaičius ir jų sudėtis nėra užduoti, tai galima gauti įvairius dekompozicijos  $D[S]$  rezultatus. Tokios dekompozicijos uždavinį galima padaryti labiau apibrėžtą, pareikalavus, kad dekompozicija būtų koku tai atžvilgiu optimali: maksimaliai sumažintų sudėtingumą, nutrauktų minimalų tarpelementinių ryšių skaičių, skaidant sistemą į posistemių, t. y. gautų operatorių  $R^{(II)}$  su minimaliu tarposistemių ryšių skaičiumi ir pan. Tokią dekompozicijos operaciją žymėsime  $D_k^{opt}[S]$ , kur  $k$  – dekompozicijos efektyvumo rodiklis, kurio reikšmę dekompozicijos metu mes stengiamės optimizuoti. Nagrinėjant sudėtingas sistemas, kuriant jų vidinius modelius, projektuojant tokių sistemų valdymo sistemas, dekompozicijos ir tos dekompozicijos optimizavimo uždavinius tenka spręsti gana dažnai. Su tokiais uždaviniais susipažinsime antrojoje šios mokymo priemonės dalyje, kai būsimės susipažinę su **optimizaciniais modeliais** (žiūr. 1.15. skyrių).

### 1.9.2.20. Struktūros papildymo („neigimo“) operacija

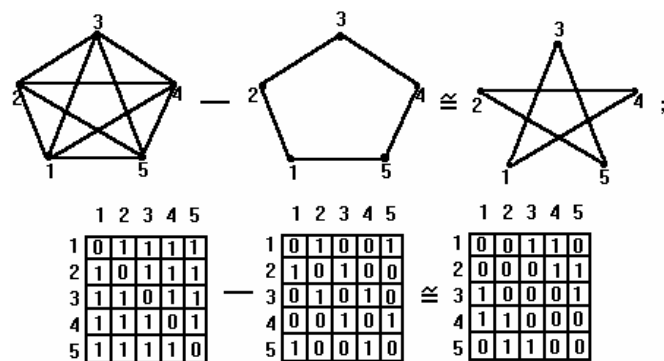
Ši operacija yra monarinė ir žymima ženklu  $\bar{S}$ . Jei struktūra  $S$  yra paprastas grafas arba orgrafas, tai  $\bar{S}$  yra struktūros  $S$  papildymas iki pilno grafo, turinčio tiek pat viršūnių, kaip ir grafas  $S$ . Vykdamas operaciją  $\bar{S}$ , matricoje  $S$  visų langelių turinys, išskyrus langelius, priklausančius pagrindinei teigiamai įstrižainei, keičiamas į jų loginius neiginius. Šios operacijos ir jos rezultatų pavyzdžiai parodyti 163 pav.

Jei struktūros  $S$  grafas turi  $p$  viršūnių ir  $q$  briaunų (arba orgrafas  $\ell$  lankų), tai struktūros  $\bar{S} \cong S_1$  grafas turės  $p_1 = p$  viršūnių ir  $q_1 = \frac{p(p-1)}{2} - q$  briaunų (arba orgrafas  $\ell_1 = \frac{p(p-1)}{2} - \ell$  lankų).



163 pav.

Operacija  $\bar{S}$  gimininga struktūrų atimties operacijai  $S_i - S_j$ . Jei iš struktūros  $S$  elementų sudarytume pilną grafą  $K_p$ , kur  $p$  – struktūros  $S$  elementų skaičius, ir rastume skirtumą  $K_p - S$ , tai nesunkiai pastebėtume, kad  $K_p - S = \bar{S}$ . Šį dėsningumą iliustruoja 164 pav.

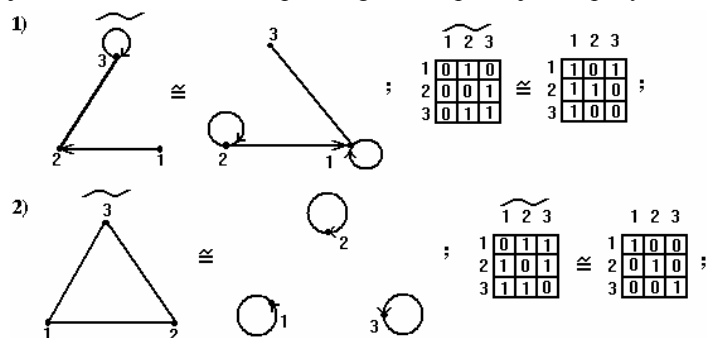


164 pav.

Galima įrodyti, kad lygybė  $K_p - S = \bar{S}$  galioja bet kokiems  $p$  ir bet kokioms struktūroms  $S$ , vaizduojamoms paprastais grafais ar orgrafais.

Kaip jau buvo minėta, operacija  $\bar{S}$  taikoma tik paprastiems grafams ir orgrafams, tačiau netaikoma pseudografams ir multigrafams. Tai reiškia, kad operacija  $\bar{S}$  nepriklauso bendrųjų operacijų klasei, t. y. ši operacija yra speciali.

**Pseudografams galima įvesti analogišką operaciją  $\tilde{S}$ , kuri keičia visų struktūros  $S$  matricos langelių turinį į jų loginį neiginį.** Operaciją  $\tilde{S}$  galima taikyti ir paprastiems grafams ir orgrafams. Tuomet jos rezultatas bus būtinai pseudografas. Operacijos  $\tilde{S}$  pavyzdžiai parodyti 165 pav.

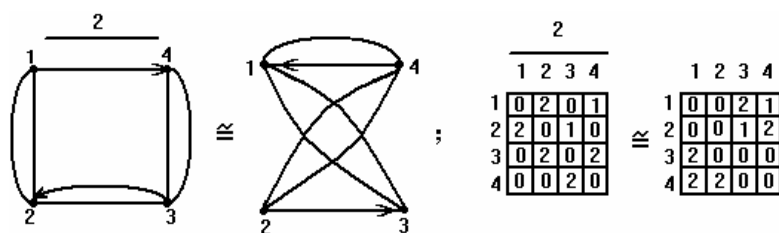


165 pav.

Operacijos  $\bar{S}$  ir  $\tilde{S}$  turi vieną bendrą savybę, būdingą loginiam neigimui:  $\bar{\bar{S}} \cong S$  ir  $\tilde{\tilde{S}} \cong S$ . Nors operacijos  $\tilde{S}$  taikymo sritis ir platesnė, tačiau ir ši operacija dar nėra bendroji, nes ji nėra apibrėžta, kai  $S$  – multigrafas.

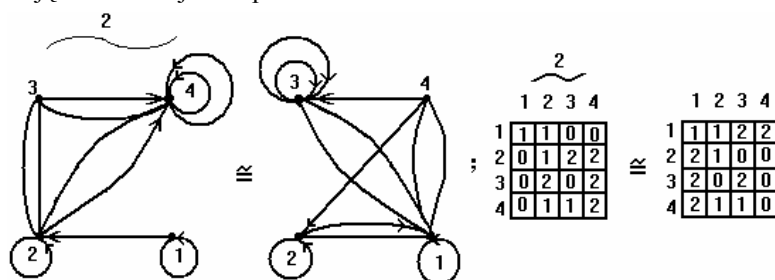
**Multigrafams galima įvesti operacijas  $\overset{m}{S}$  ir  $\overset{m}{\tilde{S}}$ , kur papildymas  $\overset{m}{m}$  suprantamas, kaip papildymas iki pilno  $m$ -multigrafo (tačiau ne pseudografo), o papildymas  $\overset{m}{m}$  suprantamas, kaip papildymas iki  $m$ -multipseudografo.**

Operacijos  $\overset{m}{S}$  pavyzdys, kai  $m=2$ , parodytas 166 pav.



166 pav.

O operaciją  $\overset{m}{\sim} S$  vaizduoja 167 pav.



167 pav.

Vykdamt operaciją  $\overset{m}{\sim} S$ , gauname matricą, kuri papildo visus matricos  $S$  langelius (išskyrus langelius, priklausančius pagrindinei teigiamajai įstrižainei) iki skaičiaus  $m$ . Matrica  $\overset{m}{\sim} S$  – visų matricos  $S$  langelių turinio papildymas iki skaičiaus  $m$ . Jei operaciją  $\overset{m}{\sim} S$  dar negalima pavadinti bendrąja, nes jos negalima taikyti pseudografams, **tai operacija  $\overset{m}{\sim} S$ , jei  $m$  pakankamai didelis skaičius, priklauso struktūrų algebros bendrųjų monarinių operacijų klasei**, nes ši operacija gali būti taikoma ir paprastiems grafams, ir orgrafams, ir pseudografams, ir multigrafams.

Operacijos  $\overset{m}{\sim}$  ir  $\overset{m}{\sim}$  taip pat turi loginio neigimo savybę:  $\overset{m}{\sim} S \equiv S$ ,  $\overset{m}{\sim} S \equiv S$ .

Jei dvejetainėje logikoje nulį (0) pakeistume neigiamu vienetu (–1), tai loginio neigimo operaciją galėtume interpretuoti, kaip skaičiaus ženklo keitimo operaciją, t. y. galėtume teigti, jog  $\bar{r} = -r$ .

Tokiu būdu gautume **aritetinę Bulio algebros interpretaciją**, kurioje  $(a \wedge b) = \min(a, b)$ ,  $(a \vee b) = \max(a, b)$ ,  $\bar{a} = -a$ . Šioje interpretacijoje galiotų visi Bulio algebros dėsniai.

Naudojantis tokia loginio neigimo operacijos traktuote, galima būtų ją taikyti visiškai vienodai ir paprastiems grafams, ir orgrafams, ir pseudografams, ir net multigrafams. Jei struktūroje  $S$  koks nors ryšys  $r_{s,u,v} = a$ , tai struktūroje  $\bar{S}$  jį atitinkantis ryšys  $r_{\bar{s},u,v}$  taptų lygus (–a). Tokioje traktuotėje dėsnis  $\bar{\bar{S}} \equiv S$  liktų galiojančiu.

### 1.9.2.21. Struktūrų jungimo operacijos

Iki šiol nagrinėjome algebrines operacijas vienoje struktūroje: operacijas su struktūros elementais bei ryšiais tarp jų, struktūros charakteristikų apskaičiavimo bei optimizavimo operacijas, o taip pat monarines struktūrų algebros operacijas, kuriose buvo operuojama su pačia struktūra kaip operacijos operandu.

Dabar pereisiu prie struktūrų algebros binarinių operacijų, kurių operandai yra dvi duotosios struktūros, o operacijos rezultatas – nauja struktūra, gauta iš operacijos operandų.

Kadangi tokių operacijų operandais gali būti ir elementariosios struktūros, susidedančios iš vieno elemento, tai šiai operacijų grupei galėtų būti priskirtos ir operacijos, aprašytos ankstesniuose šio skyriaus skyreliuose. Mes tokiomis operacijomis galėsime naudotis, nagrinėdami sudėtingesnes operacijas, jų pakartotinai neaprašinėdami.

Šiai operacijų grupei priklauso ir Šimbelo matricų daugybos operacija, iš kurios, kaip tos operacijos dalinis atvejis, išplaukia tiesioginių ryšių matricos kėlimo Šimbelo laipsniu operacija, naudojama ieškant tinkle trumpiausių maršrutų, jungiančių du duotojo tinklo mazgus. Tos operacijos čia irgi nenagrinėsime, nes ji jau aprašyta 1.9.2.11. skyrelyje.

Kai kurios binarinės operacijos yra asociatyvios, todėl iš tokių binarinių operacijų gali būti sukonstruotos  $n$ -arinės operacijos su struktūromis. Kai tokios binarinės operacijos yra dar ir komutatyvios, tuomet ir atitinkamos  $n$ -arinės operacijos irgi yra komutatyvios ir, jas vykdant, tampa nebesvarbi operandų tvarka.

Kai kurios binarinės operacijos su struktūromis turi neutralius elementus ir, naudojantis tokiomis operacijomis, iš struktūrų galima sudaryti algebrines struktūras – monoidus. Kai tokios algebrinės operacijos su struktūromis turi dar ir atvirkštinės operacijas, tai, naudojantis tokiomis

struktūrų algebros operacijomis, gali būti sudarytos algebrinės struktūros, vadinamos grupėmis. Apie grupių svarbą jau buvo rašyta 1.9.1.5. skyrelyje, todėl čia nebesikartosiu.

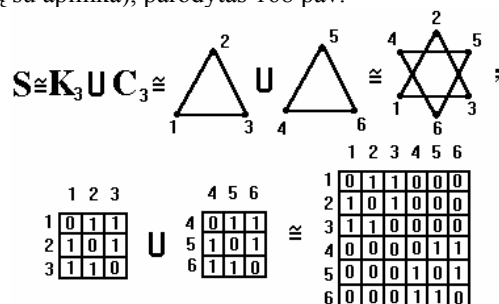
Struktūrų algebros binarinių operacijų aprašymą pradėsiu nuo paprastų ir tik vėliau pereisiu prie generuojančių operacijų, o paprastų binarinių operacijų aprašymą – nuo dažniausiai sutinkamų ir sistemotyrai labai svarbių operacijų vadinamų **struktūrų jungimo operacijomis**, apžvalgos.

Šiai operacijų šeimai priklauso **struktūrų sąsajos, struktūrų kompozicijos, struktūrų grafinės sumos** operacijos ir įvairios **tarpinės struktūrų jungimo operacijos modifikacijos**. Prijungę prie šių operacijų dar ir generuojančias struktūrų jungimo(s) operacijas gauname operacijų aibę, leidžiančią aprašyti įvairias chemines reakcijas.

### 1.9.2.21.1. Struktūrų sąsajos operacija

Struktūrų jungimas yra procesas, atvirkštinis struktūros dekompozicijai, ir turi eilę modifikacijų, kurios skiriasi viena nuo kitos atsirandančių jungimo metu **naujų ryšių išsidėstymu ir skaičiumi**. Pati paprasčiausia modifikacija žymima ženklu „ $\cup$ “. Ji vadinama **struktūrų sąsajos operacija**.<sup>\*</sup> Ši operacija **sutampa su struktūrų paprastos disjunkcijos operacija**, t. y.  $S_1 \cup S_2 \cong S_1 \vee S_2$ . Vykdamas struktūrų sąsajos operaciją, dvi sistemos (ir jų struktūros) jungiamos į sistemą  $S$  taip, kad jokių naujų ryšių neatsiranda. **Tai reiškia, jog ši operacija vaizduoja tokį sistemų jungimo procesą, kuris yra tiesinis, dėl ko jo emerđentiškumas lygus 0**. Tokį sistemų sujungimą vadinsime **pasyviuoju sujungimu**.

Operacijos  $S_1 \cup S_2$  ir jos rezultatų pavyzdys, kai jungiamos sistemos ir jų struktūros  $S_1$  ir  $S_2$  yra uždaros (t. y. neturi ryšių su aplinka), parodytas 168 pav.



168 pav.

Jei uždaros struktūros  $S_i$  ir  $S_j$  neturi bendrų elementų, tai struktūros  $S \cong S_i \cup S_j$  matrica gaunama taip, kaip parodyta 169 pav., kur  $\mathbb{0}$  – „nulinė matrica“, t. y. matrica, kurios visuose langeliuose įrašyti nuliai.

$$S_i \cup S_j \cong \begin{bmatrix} + & + \\ + & S_i \\ + & + \end{bmatrix} \cup \begin{bmatrix} + & + \\ + & S_j \\ + & + \end{bmatrix} \cong \begin{bmatrix} + & + \\ + & S_i & \mathbb{0} \\ \mathbb{0} & + & S_j \\ + & + & + \end{bmatrix}$$

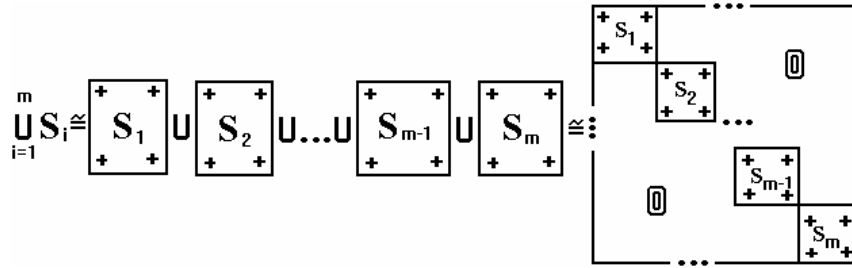
169 pav.

Struktūros  $S \cong S_i \cup S_j$  viršūnių ir lankų skaičiai  $p$  ir  $\ell$ , kai uždaros struktūros  $S_i$  ir  $S_j$  neturi bendrų elementų, lygūs:  $p = p_i + p_j$ ;  $\ell = \ell_i + \ell_j$ .

Operacija  $\cup$  asociatyvi ir komutatyvi, todėl **šią operaciją galima vartoti ne tik kaip binarinę, bet ir kaip n-arinę**.

Jei uždarų struktūrų  $S_1, S_2, \dots, S_m$  poros neturi bendrų elementų, tai operacijos  $\bigcup_{i=1}^m S_i$  rezultato matrica gaunama analogiškai (žiūr. 170 pav.).

<sup>\*</sup> Gana dažnai sąsajos operaciją „ $\cup$ “ vadina **sąjungos operacija**. Interpretuodami struktūrų jungimą plačiau (ne tik kaip sąsają „ $\cup$ “, bet ir kaip grafinę sumą „ $+$ “ bei įvairias kitas struktūrų jungimo modifikacijas), mes operaciją „ $\cup$ “ vadinsime „sąsaja“.

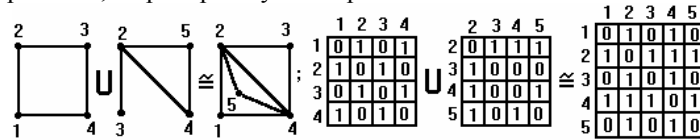


170 pav.

Struktūros  $\bigcup_{i=1}^m S_i$  viršūnių ir lankų skaičiai:  $p = \sum_{i=1}^m p_i$ ;  $\ell = \sum_{i=1}^m \ell_i$ .

Kaip matome, struktūrų  $S_1, S_2, S_3, \dots$  aibė, paimta kartu su struktūrų sąsajos operacija  $\bigcup$ , sudaro **algebrinę struktūrą**, vadinamą **pusgrapiu**. Kadangi struktūrų sąsajos operacijos atžvilgiu egzistuoja „neutralusis elementas“ – „struktūra“, vaizduojama „nulinė matrica“ (t. y. matrica, kurios visų langelių turinys yra lygus 0), tai minėtąjį pusgrupį galima laikyti **monoidu**.

Jei uždaros struktūros  $S_i$  ir  $S_j$  turi bendrų elementų, tai vykdant operaciją  $\bigcup$ , bendri elementai ir bendri ryšiai sutapatunami, kaip tai parodyta 171 pav.



171 pav.

Tai reiškia, kad vykdant operaciją  $S_i \cup S_j$ , struktūros  $S_i$  ir  $S_j$  ir jų matricos „užsideda“ viena ant kitos ir su matricų elementais  $r_{i,u,v}$  ir  $r_{j,u,v}$  vykdoma loginės sudėties (disjunkcijos  $\vee$ ) operacija:  $r_{u,v} = r_{i,u,v} \vee r_{j,u,v}$ .

Šis tvirtinimas ir formulė galioja ir atvejui, kai struktūros  $S_i$  ir  $S_j$  neturi bendrų elementų ir bendrų ryšių.

Jeį pažymėtume dviejų struktūrų  $S_i$  ir  $S_j$  bendrų elementų (grafo viršūnių) skaičių ženklu  $[p_i \cap p_j]$ , o bendrų ryšių skaičių – ženklu  $[\ell_i \cap \ell_j]$  (t. y. dviejų matricų bendrų vienetukų skaičių), tai struktūros  $S \cong S_i \cup S_j$  elementų skaičių  $p$  galima būtų išreikšti formule  $p = p_i + p_j - [p_i \cap p_j]$ , o jo vienpusių ryšių skaičių  $\ell$  – formule  $\ell = \ell_i + \ell_j - [\ell_i \cap \ell_j]$ , kas asocijuojasi su tikimybinės logikos formule  $A \vee B = A + B - A \wedge B$  bei aibių teorijos formule  $|A \cup B| = |A| + |B| - |A \cap B|$ .

Struktūros  $S_i \cup S_j$  elementų skaičiaus  $p$  ir joje egzistuojančių vienpusių ryšių skaičiaus  $\ell$  apskaičiavimo formules, naudojantis įvestais žymėjimais  $[p_1 \cap p_2 \cap p_3 \cap \dots]$ , galima apibendrinti ir operacijai  $S \cong \bigcup_{i=1}^m S_i$ :

$$r_{u,v} = \bigvee_{i=1}^m r_{i,u,v} = r_{1,u,v} \vee r_{2,u,v} \vee \dots \vee r_{m,u,v};$$

$$p = \sum_{i=1}^m p_i - \sum_{i=1}^{m-1} \sum_{j=i+1}^m [p_i \cap p_j] + \sum_{i=1}^{m-2} \sum_{j=i+1}^{m-1} \sum_{k=j+1}^m [p_i \cap p_j \cap p_k] -$$

$$- \sum_{i=1}^{m-3} \sum_{j=i+1}^{m-2} \sum_{k=j+1}^{m-1} \sum_{\ell=k+1}^m [p_i \cap p_j \cap p_k \cap p_\ell] + \dots + (-1)^{m+1} \left[ \bigcap_{i=1}^m p_i \right]$$

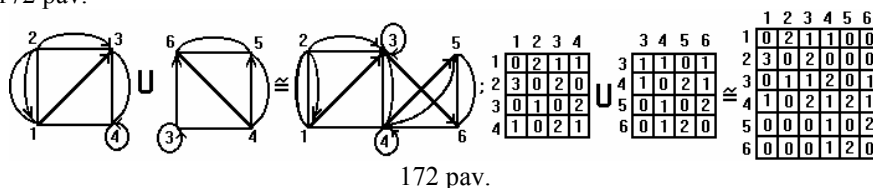
$$\ell = \sum_{i=1}^m \ell_i - \sum_{i=1}^{m-1} \sum_{j=i+1}^m [\ell_i \cap \ell_j] + \dots + (-1)^{m+1} \left[ \bigcap_{i=1}^m \ell_i \right].$$

Rekomenduočiau skaitytojui pabandyti išvesti ar bent jau patikrinti šias formules. Jos tinka ir tam atvejui, kai struktūros  $S_1, S_2, \dots, S_m$  neturi bendrų elementų ir ryšių. Tuomet  $p = \sum_{i=1}^m p_i$  ir  $\ell = \sum_{i=1}^m \ell_i$ .

Kai struktūros  $S_1, S_2, \dots, S_i, \dots, S_m$  yra tik skirtingi paprasti elementai, tai visi  $p_i=1$  ir visi  $\ell_i=0$ . Tuomet gauname, jog  $m$  elementų sąsajos viršūnių skaičius  $p=m$ , o lankų skaičius  $\ell=0$ , ką ir turėjome gauti, atsižvelgdami į struktūrų sąsajos operacijos apibrėžimą.

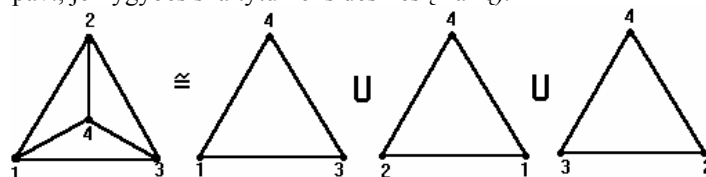
Kai struktūros  $S_1, S_2, \dots, S_i, \dots, S_m$  yra izomorfiškos ir nė viena jų pora neturi bendrų elementų, tai  $\bigcup_{i=1}^m S_i$  galima užrašyti kaip  $mS$ , kur  $S$  yra struktūra, izomorfiška bet kuriai iš  $S_i$ , t. y.  $S \cong S_1 \cong S_2 \cong \dots \cong S_m$ , o  $mS$  reiškia struktūrą  $S$ , pakartotą sąsajos operacijos operandu  $m$  kartų, t. y.  $mS \cong S_1 \cup S_2 \cup \dots \cup S_m$ . Kitaip tariant,  $mS$  – tai sutrumpintas aibės, susidedančios iš  $m$  izomorfiškų struktūrų žymėjimas.

Struktūrų sąsajos operaciją galima taikyti ne tik paprastiems grafams ir orgrafams (loginėms matricoms), bet ir multigrafams (aritmetinėms matricoms) bei pseudografams. Tik **multigrafų atveju disjunkcijos operaciją  $\vee$  reikia apibendrinti:  $a \vee b = \max(a, b)$** . Operacijos  $\cup$  taikymą multigrafams vaizduoja 172 pav.



172 pav.

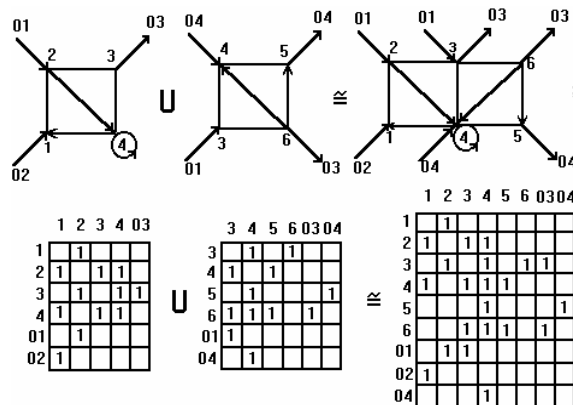
Tokią **apibendrintą disjunkcijos  $\vee$  interpretaciją galima taikyti tiek loginėms, tiek skaitmeninėms, tiek turinčioms bendrų elementų, tiek neturinčioms jų matricoms, o taip pat negriežtai apibrėžtoms aibėms** (1.16 skyrelį) todėl, įvedus minėtą disjunkcijos apibendrinimą, **struktūrų algebros operaciją  $\cup$  galima priskirti prie bendrųjų operacijų klasės**. Šią operaciją galima būtų taikyti ne tik struktūrų jungimui, jungiant kelias sistemas į vieną, kai tarp jungiamų sistemų neatsiranda naujų ryšių, bet ir dekomponuojant sudėtingą sistemą į paprastesnes (žiūr. 173 pav., o taip pat 170, 171 ir 172 pav., jei lygybes skaitytume iš dešinės į kairę).



173 pav.

Tą patį struktūrų sąsajos  $S \cong S_i \cup S_j$  operacijos įvykdymo algoritimą galima taikyti ir atviroms struktūroms, jei tos struktūros per savo išorines aplinkas neturi tarpusavio tiesioginių ryšių (žiūr. 174 pav.).

Žinoma, struktūrų sąsajos operaciją  $S \cong S_i \cup S_j$  (o taip pat apibendrintą  $n$ -arinę struktūrų sąsajos operaciją  $\bigcup_{i=1}^m S_i$ ) galima taikyti ir atviroms struktūroms, turinčioms bendras išorines aplinkas ir kai kuriuos bendrus ryšius su išorine aplinka. Tuomet bendros išorinės aplinkos ir bendri ryšiai sutapatinami, uždedant bendrąsias išorines ir vidines susiejamų struktūrų dalis vieną ant kitos, kaip tai buvo padaryta 171, 172 ir 173 pav., o taip pat 174 pav.



174 pav.