VII. GRAFŲ TEORIJOS PAGRINDAI

7.1 Svarbiausios sąvokos

Sakykime, kad V ir E dvi netuščios aibės. Tada *aibių porą* $G := (V, E)$ *vadinsime grafu*. Aibę V vadinsime *viršūnių aibe*, o aibę $E-$ *briaunų, jungiančių (siejančių) bet kokias dvi aibės* V *viršūnes, aibė*; $E = \{e := (x, y); x, y \in V\}$. Patogu žymėti:$(x, y) = xy$. Jei poros $xy \in E$ sutvarkytos, t.y. $xy \neq yx$, tai grafas G vadinamas *digrafu* arba *orientuotu grafu*. Orientuoto grafo briaunos bus vadinamos *lankais*.

Paprastai grafo viršūnės yra vaizduojamos taškais, o briaunos lankais arba atkarpomis. Jeigu nagrinėjamas grafas yra digrafas, tai viršūnes jungianti briauna vaizduojama su rodykle. Viršūnės x, y vadinamos briaunos xy *galais* arba dažnai briaunai *incidenčiomis* viršūnėmis. Dvi viršūnės incidenčios tai pačiai briaunai vadinamos *gretimomis* viršūnėmis. Jei viena viršūnė incidenti kelioms briaunoms, tai šias briaunas vadinsime *gretimomis* briaunomis. Viršūnei v gretimų viršūnių aibę vadinsime viršūnės v *gretimų viršūniu aibe*:

$$\Gamma^+(v) := \{u \in V; (u, v) \in E\}, \;\; \Gamma(v) := \Gamma^+(v) \cup \{v\}, \;\; u \in \Gamma(v) \;\Leftrightarrow v \in \Gamma(u).$$

Jei grafo viršūnė x yra siejama su savimi, tai briauną xx vadinsime *kilpa*. Sakykime, kad M kokia nors simbolių aibė. Aibę M vadinsime *žymenų aibe*, jei egzistuoja funkcija $f : V \to M \;\; (f : E \to M)$. Šiuo atveju grafą vadinsime *pažymėtu grafu*. Jei $M \subset \mathcal{N}_n$, tai grafas vadinamas *numeruotu grafu*.

Ateityje simboliu "$G(V, E)$" žymėsime grafą, kuris neorientuotas, nepažymėtas, be kilpų ir neturi kartotinių briaunų. Tokius grafus vadinsime *paprastaisiais* grafais. Grafo G viršūnių skaičių žymėsime raide p, o briaunu skaičių- q. Taigi $p = |V|, \;\; q = |E|$. Jei $p, q \in \mathcal{N}$, tai grafai vadinami baigtiniais. Skaičius p vadinamas grafo *eile*, o $q-$ grafo *didumu*. Jei $p = 0$, tai grafas vadinamas tuščiu, o kai tarp visų viršūnių egzistuoja briaunos, tai toks grafas vadinamas pilnuoju, beje tada $q = C_p^2$. Bendrai paėmus, paprasto grafo briaunų skaičius turi savybę: $q \leq C_p^2$.

1 Teorema Tarkime, kad $n \in \mathcal{N}$. Egzistuoja

$$2^{\frac{n(n-1)}{2}}$$

$n-$ eilės numeruotų grafų.

$\ominus$

Grafų dydžiai priklauso aibei $\{0, 1, 2, \ldots, N\}$, kurios galia yra C_n^2. Nubrėžkime i dydžio grafą. Tada yra C_N^i galimybių pasirinkti i briaunų iš visos briaunų aibės. Tada visų galimų dydžių skaičių gauname tokiu būdu:

$$\sum_{i=0}^{N} C_N^i = 2^N.$$

$\oplus$

Sakysime, kad grafai $G(V_1, E_1)$ ir $G(V_2, E_2)$ yra *izomorfiški*, jeigu egzistuoja bijekcija $h : V_1 \to V_2$ tokia, kad $\forall xy \in E$, yra tenkinama sąlyga:

$$xy \in E_1 \quad \Leftrightarrow \quad h(x)h(y) \in E_2.$$

Digrafo atveju bijekcija h turi išlaikyti ir orientaciją (kryptį).

Nagrinėsime grafus izomorfizmo tikslumu, t.y. bus nagrinėjamos grafų ekvivalentumo klasės. Paprasčiau kalbant, nesvarbu kaip bus grafai vaizduojami, bet svarbu tai, kad ekvivalentumo klasei priklausantys grafai turės tą patį viršūnių ir tą patį briaunų skaičių bei atitinkamos viršūnės bus siejamos briaunomis. Skaitinės charakteristikos, vienodos izomorfiniams grafams bus vadinamos grafo *invariantais*. Taigi, p, q (viršūnių ir briaunų) skaičius yra grafo invariatai.

Grafas $G'(V', E')$ vadinamas grafo $G(V, E)$ *pografiu* (žymėsime $G' \subset G$,) jeigu $V' \subset V$ ir $E' \subset E$. Jeigu $V' = V$, tai G' vadinamas *pagrindiniu grafo G pografiu*.

Pografį G' vadinsime *tikriniu grafo G pografiu, jeigu* $V' \neq V$ *ir* $E' \neq E$

Pografį $G'(V', E')$ vadinsime *taisyklingu grafo* $G(V, E)$ *pografiu, jeigu grafui* G' *priklauso visos G briaunos:* $\forall (u, v) \in V',\ (u, v) \in E \ \Rightarrow (u, v) \in E'$.

Viršūnės v laipsniu (valentingumu)$d(v)$ vadinsime šiai viršūnei incidenčių briaunų skaičių:

$$d(v) = |\Gamma(v)|, \ \ \forall v \in V, 0 \leq d(v) \leq p - 1.$$

Kai $d(v) = 0$, viršūnė v bus vadinama *izoliuotąja*.

Simboliu $\delta(G)$ žymėsime minimalų grafo G viršūnių laipsnį, o simboliu $\Delta(G)$ žymėsime maksimalų grafo G viršūnės laipsnį:

$$\delta(G) := \delta(G(V, E)) := \min_{v \in V} d(v), \ \ \Delta(G) := \Delta(G(V, E)) := \max_{v \in V} d(v).$$

Grafą G vadinsime *reguliariu*, jeigu visos grafo viršūnės turi tą patį laipsnį, tarkime k :

$$\delta(G) = \Delta(G) = k.$$

k vadinamas grafo *reguliarumo laipsniu* ir žymėsime $r(G)$. Jei grafas nereguliarus, tai reguliarumo laipsnis yra neapibrėžtas.

Tarkime, kad grafas G yra orientuotas. Tada iš viršūnės v išeinančių lankų skaičius bus žymimas $d^+(v)$ ir vadinamas *išėjimo laipsniu*, o į viršūnę v įeinančių lankų skaičius bus žymimas $d^-(v)$ ir vadinamas *įėjimo laipsniu*.

Jeigu orientuoto grafo viršūnės skaičius $d^+(v) = 0$ tai tokia viršūnė vadinama *šaltiniu*, jei $d^-(v) = 0$, tai viršūnė vadinama *santaka*. Orientuotas grafas, kuriame yra vienas šaltinis ir viena santaka vadinamas *tinklu*.

2 Teorema *Grafo viršūnių laipsnių suma visada lygi dvigubam briaunų skaičiui:*

$$\sum_{v \in V} d(v) = 2q, \quad \sum_{v \in V} (d^+(v) + d^-(v)) = 2q.$$

$\ominus$

Skaičiuojant viršūnės laipsnius, kiekviena briauna (lankas) skaičiuojama du kartus.

$\oplus$

Grafą charakterizuosime pagal tai, kokiu būdu yra jungiamos viršūnės.

Viršūnių ir briaunų seką grafe $v_0, e_1, v_1, e_2, \ldots, e_k, v_k$, kai bet kokia viršūnės ir briaunos pora yra incidentiškos, vadinsime *keliu*. Kelias, kurio visos viršūnės skirtingos vadinamas *trasa*. Jei trasos visos briaunos skirtingos, tai trasa vadinama *grandine*. Jei trasos visos viršūnės (tuo pačiu ir briaunos) yra skirtingos, tai trasą vadinsime *paprasta grandine*. Uždarą grandinę $v_0 = v_k$, kai $k \geq 2$, vadinsime *ciklu*. Paprasta uždara grandinė yra vadinama *paprastu ciklu*. Simboliu $z(G)$ žymėsime ciklų skaičių grafe G. Jei grafe egzistuoja grandinė, sudaryta iš visų jo briaunų, tai šis grafas vadinamas *Eulerio* vardu. Jei grafe egzistuoja ciklas, apimantis visas viršūnes, tai šis grafas vadinamas Hamiltono vardu. Jei grafas orientuotas, tai kelias paprastai vadinamas *taku*, o ciklas- *kontūru*. Tarkime, kad kelias apibrėžtas tokiu būdu:

$$M: \ v_0, e_1, v_1, e_2, \ldots, e_k, v_k.$$

Tada briaunų, esančių kelyje, skaičių vadinsime kelio M ilgiu. Žymėsime: $|M| = k$.

Atstumu tarp dviejų viršūnių u, v, žymėsime $\rho(u, v)$, vadinsime trumpiausią grandinę jungiančią viršūnes u, v. Šią grandinę vadinsime *geodezine grandine*. Jei viršūnių u, v nejungia grandinė, tai $\rho(u, v) = +\infty$.

Grafo G *skersmeniu* (žymėsime D(G)) vadinsime ilgiausią geodezinę grandinę.

Sakysime, kad dvi grafo viršūnės yra *jungios*, jei egzistuoja šias viršūnes jungianti grandinė. Grafas vadinamas jungiu, jei visos šio grafo viršūnės yra jungios.

Viršūnių jungumo sąryšis yra ekvivalentumo sąryšis. Šios ekvivalentumo klasės vadinamos jungumo komponentėmis. Beje, jungumo komponentės yra grafo pografiai. Pažymėkime $k(G)-$ grafo G jungumo komponenčių skaičių. Tada, jei $k(G) = 1$, tai $G-$ susijęs grafas. Jei $k(G) > 1$, tai grafas nesusijęs. Jei grafas sudarytas tik iš izoliuotų viršūnių, tai grafą vadinsime visiškai nesusijusiu. Šiuo atveju $k(G) = p(G)$.

7.2 Grafų tipai

Grafą vadinsime *trivialiu*, jei jį sudaro viena viršūnė. Grafą, kurį sudaro k viršūnių ir kurias jungia paprastas ciklas žymėsime simboliu C_k. Pavyzdžiui kvadratas yra C_4.

Grafą, kurio bet kokia viršūnių pora yra gretimos (jungiamos briauna), vadinsime *pilnu* grafu. Pilną grafą, turintį p viršūnių žymėsime simboliu K_p. Šio grafo maksimalus briaunų skaičius yra

$$q(K_p) = \frac{p(p-1)}{2}.$$

Grafo G bet koks pilnas pografis vadinamas *klika*. Grafą $G(V, E)$ vadinsime *dvidaliu* grafu, jei egzistuoja nesikertančios aibės V_1, V_2 ir $V_1 \cup V_2 = V$ ir be to bet kokia briauna $e \in E$ yra incidentiška viršūnei iš $v_i^1 \in V_1$ ir $v_j^2 \in V_2$. Kitaip tariant, kiekviena briauna jungia skirtingų poaibių viršūnes. Aibės V_1 ir V_2 vadinamos grafo *dalimis*. Jei dvidalis grafas apima visas briaunas, jungiančias aibių V_1 ir V_2 elementus, tai šis grafas vadinamas

pilnu dvidaliu grafu. Jei $|V_1| = m$ ir $|V_2| = n$, tai pilnąjį dvidalį grafą žymėsime $K_{m,n} = G(V_1, V_2, E)$.

3 Teorema *Grafas yra dvidalus tada ir tik tada, kai grafo paprasti ciklai turi lyginį ilgį.*

$\ominus$

Būtinimas. Tarkime priešingai. T.y. egzistuoja dvidalus grafas $G(V_1, V_2; E)$ ir be to $v_1, \ldots, v_{2k+1}, v_1-$ paprastas, nelyginio ilgio ciklas. Tarkime be to, kad $v_1 \in V_1$, tada $v_2 \in V_2$, $v_3 \in V_3$, $\ldots$, $v_{2k+1} \in V_1$. Vadinasi $v_1, v_{2k+1} \in E-$ bet tai prieštarauja tam, kad grafas dvidalus. Taigi, prielaida buvo klaidinga.

Pakankamumas. Tarkime, kad grafas G yra susijęs, kadangi priešingu atveju kiekvieną komponentę galėtume nagrinėti atskirai. Naudodami žemiau pateiktą algoritmą suskaidykime aibę V į dviejų nesikertančių aibių V_1 ir V_2 sąjungą:

Įv: grafas $G(V, E)$.
Išv: grafo G dalys V_1 ir V_2
select $v \in V$ (laisvai pasirenkama viršūnė)
$V_1 := v_1$ (priskiriame viršūnę pirmam poaibiui)
$V_2 := \emptyset$
for $u \in V \setminus \{v\}$ **do**
if $d(v, u)$ lyginis, **then**
$V_1 := V_1 \cup \{u\}$
else
$V_2 := V_2 \cup \{u\}$
end if
end for

Naudodami prieštaros metodą. Tegu $u, w \in V_2$, $(u, w) \in E$, t.y. egzistuoja dvi viršūnės dalyje V_2 sujungtos briauna. Tegu v yra pradinė viršūnė, pasirinkta algoritmo pradžioje, o (v, u) ir (v, w) yra dvi geodezinės grandinės. Tada ilgiai $|(v, u)|$ ir $|(v, w)|$ yra nelyginiai skaičiai. Be to, šios grandinės turi bendrą viršūnę. Tegu v' yra viršūnė, kuri labiausiai nutolusi nuo v, jei matuosime atstumą geodezinėmis grandinėmis. Turime, kad geodezinių grandinių ilgių suma:

$$|(v', u)| + |(v', w)| = |(v, u)| + |(v, w)| - 2|(v, v')|$$

yra lyginis skaičius, o tada $v', \ldots, u, w, \ldots, v'$ yra paprastas nelyginio ilgio ciklas. Bet pastarasis tvirtinimas prieštarauja pradinėms sąlygoms. Jeigu $u, w \in V_1$, tai ilgiai $|(v, u)|$ ir $|(v, w)|$ lyginiai ir gauname, kad $v', \ldots, u, w, \ldots, v'-$ paprastas nelyginio ilgio ciklas.

$\oplus$

Jei grafas yra pilnas ir orientuotas, tai toks grafas paprastai vadinamas *turnyru.* (Šio pavadinimo genezė yra tokia: tarkime vyksta varžybos tarp žaidėjų porų, kai lygiųjų būti negali. Besivaržančių poras sujunkime lanku, rodyklę nukreipdami laimėtojo link. Tokiu būdu sudarytas grafas reprezentuoja turnyro, vieno rato, rezultatus.)

7.3 Grafų veiksmai. Grafų vaizdavimas

Pažymėkime $G_1 := G(V_1, E_1)$, $G := G(V, E)$ ir $G_2 := G(V_2, E_2)$.

1. Grafo G_1 *papildiniu* (žymėsime $\overline{G}_1$) vadinsime grafą G_2, kai

$$V_2 := V_1,\ E_2 := \overline{E}_1 := \{e \in V_1 \times V_1; e \notin E_1\}.$$

2. Apibrėžkime veiksmus, kurių dėka iš grafo bus pašalinamos viršūnės ir briaunos. Tarkime, kad $V' \subset V$ ir $xy \in E' \subset E$. Tada grafo ir viršūnių ir briaunų aibių skirtumu, atitinkamai, vadinsime tokius grafus:

$$G - V' := G[v \setminus V'], \quad \text{ir} \quad G - E' := (V, E \setminus E').$$

Grafas $G - v = G - \{v\}$ gaunamas iš grafo G pastarąjame pašalinus viršūnę v bei jai incidenčias briaunas, o grafas $G - vu = G - \{vu\}-$ išmetant tik briauną vu. Tada, kai iš grafo G atimama briauna vu, o viršūnės v ir u sutapatinamos, tai šis veiksmas vadinamas grafo *sutraukimu*.

3. Tarkime, kad $V_1 \cap V_2 \emptyset$ ir $E_1 \cap E_2 = \emptyset$. Tada grafų G_1 ir G_2 *sąjunga* vadinsime grafą $G(V, E)$, kai $V = V_1 \cup V_2$ ir $E = E_1 \cup E_2$. Šį veiksmą žymėsime $G := G_1 \cup G_2$.

4. Tarkime, kad $V_1 \cap V_2 \emptyset$ ir $E_1 \cap E_2 = \emptyset$. Tada grafų G_1 ir G_2 *suma* vadinsime grafą $G(V, E)$, kai $V = V_1 \cup V_2$ ir $E = E_1 \cup E_2 \cup \{e = (v_1, v_2); v_i \in V_i\}$. Šį veiksmą žymėsime $G := G_1 + G_2$.

Pateiksime keletą veiksmų pavyzdžių.

5. *Viršūnės v prijungimu* prie grafo G_1 (žymėsime $G_1 + v$), $v \notin V_1$, vadinsime tokią operaciją:

$$G_1 + v := G_2, \quad V_2 := V_1 \cup \{v\} \ \text{ir} \ E_2 := E_1.$$

6. *Briaunos e prijungimu* prie grafo G_1 (žymėsime $G_1 + e$), $e \notin E_1$, vadinsime tokią operaciją:

$$G_1 + e =: G_2, \quad V_2 := V_1 \ \text{ir} \ E_2 := E_1 \cup \{e\}.$$

Pateiksime keturis pagrindinius grafų vaizdavimo būdus. Vaizdavimo būdai priklauso nuo konkrečių užduočių ir dažnai įvairūs vaizdavimo būdai yra kombinuojami. Tegu $n(p, q)-$ yra atminties dydžio parametras, priklausantis nuo viršūnių bei briaunų skaičiaus.

Pateiksime keletą pavyzdžių.

1. Grafų vaizdavimas naudojant matricas. Tarkime duota matrica

$$M : \mathbf{array}[1 \ldots p, 1 \ldots p]\mathbf{of}\ 0 \ldots 1.$$

Šią matricą vadinsime sąsajų matrica, kai

$$M[i, j] = \begin{cases} 1, & v_i \text{ gretima } v_j, \\ 0, & v_i \text{ negretima } v_j, \end{cases}.$$

Kai duota sąsajų matrica $n(p, q) = O(p^2)$.

2. Grafo vaizdavimas naudojant *incidentiškumo matricą* $H : \mathbf{array}[1 \cdot p, 1 \cdot q]\mathbf{of} 0 \cdot 1$ (jei grafas orientuotas, tai $H : \mathbf{array}[1 \cdot p, 1 \cdot q]\mathbf{of} -1 \cdot 1$). Šia matrica vaizduojamas viršūnių ir briaunų incidentiškumas. Jei grafas neorientuotas tai

$$H[i,j] = \begin{cases} 1,\ v_i \text{ incidenti } e_j, \\ 0,\ v_i \text{ neincidenti } e_j, \end{cases}$$

jei orientuotas, tai

$$H[i,j] = \begin{cases} 1,\ v_i \text{ incidenti } e_j,\ e_j \text{ yra jo pab.} \\ 0,\ v_i \text{ neincidenti } e_j, \\ -1,\ v_i \text{ incidenti } e_j,\ \text{ir } e_j \text{ jo prad.} \end{cases}$$

Šiai matricai $n(p,q) = O(pq)$.

3. Sąsajų sąrašai. Reprezentuosime grafą naudodami sąrašą, kuriame bus nurodamos rodyklės pateikiamos masyve: Γ : **array**$[1{\cdot}p]$**of** $\uparrow N$ ir sąrašas gretimų viršūnių, pateikiamų sąrašu: N : **record** $v : 1 \ldots p; n :\uparrow N$**end record**.

Jei grafas neorientuotas, tai $n(p,q) = O(p+2q)$, o jei grafas orientuotas $n(p,q) = O(p+q)$.

4. Lankų masyvas. Grafas gali būti vaizduojamas masyvu

E : **array**$[1 \ldots p]$**of record**$b, e : 1 \ldots p$**end record**. Šiame sąraše pateikiamos gretimų viršūnių poros, kurios paprastai vadinamos briaunų masyvu. Šiuo atveju $n(p,q) = O(2q)$.

7.4 Klajojimas grafais

Klajojimu grafe vadinsime nuoseklų grafo viršūnių (briaunų) išvardinimą. Vienas iš svarbesnių išvardinimų yra toks, kada yra pateikiamos gretimos viršūnės. Iš įvairių klajojimo algoritmų skiriami algoritmai, kurie atlieka klajojimą į *plotį* ir į *gylį*.

Pateiksime vieną tokį algoritmą.

Įv.: Grafas $G(V,E)$ pateikiamas sarašu Γ.
Išv.: klajojimų seka.
for $v \in V$ **do**
$x[v] := 0$ (pradžioje visos viršūnės nepažymėtos)
end for
select $v \in V$ (pradinė viršūnė)
$v \to T$ (patalpiname v į duomenų struktūrą T)
$x[v] := 1$ (pažymime viršūnę v)
repeat
$u \leftarrow T$ (pasirenka viršūnę iš duomenų aibės T)
yield u (grąžina viršūnę fiksavus kad ji jau praeita)
for $w \in \Gamma(u)$ **do**
if $x[w] = 0$ **then**
$w \to T$ (patalpina w į duomenų struktūrą T)
$x[w] := 1$ pažymi viršūnę w
end if
end for
until $T = \emptyset$.

Jei T - LIFO (last in first out), tai klaidžiojimas vadinamas *paieška į gylį*, FIFI (first in first out), tai klaidžiojimas vadinamas *paieška į plotį*.

4 Teorema Jei grafas G baigtinis ir susijęs, tai klaidžiojant į plotį ir į gylį bus aplankomos visos viršūnės po vieną kartą.

$\ominus$

Vienatinumas. Remiantis aprašytu algoritmu, klaidžiojama tik viršūnėmis, kurios priklauso T. Be to į T patenka tik nepažymėtos viršūnės. Patekusios į T, viršūnės yra pažymimos. Taigi, visos viršūnės bus aplankomos tik po kartą.

Algoritmo baigtinumas. Aibėje T gali būti ne daugiau negu p viršūnių. Kiekviename žingsnyje, viena viršūnė yra pašalinama. Taigi, algoritmas tęsis ne daugiau negu p žingsnių.

Klaidžiojant aplankomos visos viršūnės. Tarkime priešingai. Tarkime, kad algoritmas baigė darbą, o viršūnė w nebuvo aplankyta. Taigi, w nepateko į T. Tada ši viršūnė nebuvo pažymėta. Vadinasi, visos viršūnės gretimos šiai viršūnei nebuvo aplankytos ir pažymėtos. Analogiškai samprotaudami gauname, kad bet kokia viršūnė, gretima nepažymėtai, taip pat nepažymėta. Taigi, jei G susijęs gauname, kad $T = \emptyset$.

$\oplus$

1. Išvada *Tegu* $(u_1, \ldots, u_p)$*- koks nors klaidžiojimas į plotį. Tada*

$$\forall i > j,\ d(u_1, u_i) \leq d(u_1, u_j).$$

Matome, kad funkcija $d(u_1, u), u \in V$ yra monotoniškai didėjanti funkcija, klaidžiojant į plotį.

2. Išvada *Tegu* $(u_1, \ldots, u_p)$*- koks nors klaidžiojimas į gylį. Tada*

$$\forall i \geq 1,\ d(u_1, u_i) \leq i \leq p.$$

Kitaip tariant, bet kokios viršūnės paieškos laikas ne mažesnis už atstumą tarp pradinės viršūnės ir ieškomosios, bet tuo pačiu metu nedidesnis negu viršūnių skaičius.

3. Išvada *Tegu* $(u_1, \ldots u_i, \ldots, u_p)$*- koks nors klaidžiojimas į plotį, o* $(v_1, \ldots, v_j, \ldots, v_p)$*- koks nors klaidžiojimas į gylį, kai* $u_i = v_j$*. Tada* $i = 2j$*.*

Kitaip tariant, klaidžiojimas į gylį yra du kartus greitesnis negu klaidžiojimas į plotį.

7.5 Grafų jungumas

5 Teorema *Grafas yra jungus tada ir tik tada, kai šio grafo negalima užrašyti dviejų grafų sąjunga.*

$\ominus$

Būtinumas. Tarkime priešingai, t.y $k(G) = 1$, bet $G = G_1 \cup G_2$. Tegu $v_1 \in V_1$ ir $v_2 \in V_2$. Tada egzistuoja grandinė $[v_1, v_2]$ ir $v_i \in V_i$. Šioje grandinėje egzistuoja briauna $e = (a, b),\ a \in V_1,\ b \in V_2$. Bet tada $e \notin E_1,\ e \notin E_2$. Tada $e \notin E$.

Pakankamumas. Tarkime, kad $k(G) > 1$. Tada egzistuoja viršūnės u, v, kurių nesieja jokia grandinė. Vadinasi u, v priklauso skirtingoms komponentėms. Pažymėkime G_1- jungumo komponentę, kuriai priklauso u, o G_2- likusios viršūnės. Tada $G = G_1 \cup G_2$.

Grafo viršūnę vadinsime *sąlyčio tašku* , jei pašalinus šią viršūnę padidėja grafo jungumo komponenčių skaičius.

1 Lema *Kiekviename netrivialiame grafe egzistuoja bent dvi viršūnės, kurios nėra sąlyčio taškai.*

6. Teorema Teisinga nelygybė:

$$p - k \leq q \leq \frac{(p-k)(p-k+1)}{2},$$

čia $p-$ viršūnių skaičius, $q-$ briaunų skaičius, $k-$ grafo jungumo komponenčių skaičius.

Šios teoremos neįrodysime.

4 Išvada*Jeigu*

$$\frac{q > (p-1)(p-2)}{2}$$

, tai grafas yra susijęs.

$\ominus$

Panagrinėkime nelygybę

$$\frac{(p-1)(p-2)}{2} < q \leq \frac{(p-k)(p-k+1)}{2}, \ \ \forall k \in \mathcal{N}.$$

Jei $k = 1$, tai

$$\frac{(p-1)(p-2)}{2} < \frac{(p-1)(p)}{2}, \ \text{ tiesa,}$$

Jei $k = 2$, tai

$$\frac{(p-1)(p-2)}{2} < \frac{(p-1)(p-2)}{2}, \ \text{ netiesa,}$$

Jei $k = 3$, tai

$$\frac{(p-1)(p-2)}{2} < \frac{(p-3)(p-2)}{2}, \ \text{ netiesa.}$$

ir t.t.

$\oplus$

Briauna, kurią pašalinus padidėja grafo jungumo komponenčių skaičius, vadinsime *tiltu*. Susijęs grafas, neturintis sąlyčio taškų, vadinamas *bloku*. Jeigu grafe $K_n, \ n \geq 2$ egzistuoja tiltas, tai egzistuoja ir sąlyčio taškas. Kiekvieno tilto galiniai taškai tuo pačiu ir sąlyčio taškai. Bet ne atvirkščiai.

7 Teorema *Tarkime, kad G susijęs grafas ir $v \in V$. Tada šie tvirtinimai ekvivalentūs:*
1) v yra sąlyčio taškas;
2) $\exists u, w \in V, u \neq w$ ir kiekvienai grafo G grandinei $[u, w]$, $v \in [u, w]$;
3) $\exists U, W, \ U \cap W = \emptyset$ ir $U \cup W = V \setminus \{v\}$ ir $\forall u \in U$ ir $\forall w \in W$ visoms grandinėms $[u, w]$, $v \in [u, w]$.

$\ominus$

$1 \Rightarrow 3$. Nagrinėkime grafą $G - v$. Šis grafas nėra susijęs, kadangi $k(G - v) > 1$, taigi šis grafas turi bent dvi jungumo komponentes,

$$V \setminus \{v\} = U \cup W,$$

čia $U-$ vieną jungumo komponentės aibė, o $W-$ likusios viršūnės.

Tarkime, kad $u \in U,\ w \in W$. Tada neegzistuoja grandinės, siejančios viršūnes u, w, priklausančias aibei $G - v$. Bet $k(G) = 1$. Tada, egzistuoja grafo G grandinė $[u, w]$ ir kiekvienai grandinei $[u, w]$, $v \in [u, w]$.

$3 \Rightarrow 2$. Akivaizdžiai.

$2 \Rightarrow 1$. Nagrinėsime aibę $G - v$. Turime, kad šiame grafe neegzistuoja grandinės siejančios viršūnes u, w. Vadinasi $k(G - v) > 1$, taigi $v-$ sąlyčio taškas.

$\oplus$

8 Teorema *Tarkime, kad G susijęs grafas ir $x \in E$. Tada šie tvirtinimai ekvivalentūs:*
1) x yra tiltas;
2) x nepriklauso jokiam paprastam ciklui;
3) $\exists u, w \in V$, ir $\forall [u, w] \in G$, $x \in [u, w]$; $v \in [u, w]$.
4) $\exists U, W, U \cap W = \emptyset$ ir $U \cup W = V$ ir $\forall u \in U$ ir $\forall w \in W$ ir visoms grandinėms $[u, w]$, $x \in [u, w] \in G$.

Įrodymas analogiškas paskutinės teroemos įrodymui.

Grafo G viršūniniu jungumu ($\chi(G)$) vadinsime mažiausią viršūnių skaičių, kurias pašalinus iš pradinio grafo gausime nejungų arba trivialų grafą. Grafą vadinsime $k-$ jungiu, jei $\chi(k) = k$.

Jei grafas nejungus, tai $\chi(G) = 0$; jei turi vieną sąlyčio tašką, tai $\chi(G) = 1$; $\chi(G) = p - 1$ jei grafas pilnas.

Grafo G briauniniu jungumu ($\lambda(G)$) vadinsime mažiausią briaunų skaičių, kurias pašalinus iš pradinio grafo gausime nejungų arba trivialų grafą.

Jei grafas nejungus, tai $\lambda(G) = 0$; jei egzistuoja tiltas, tai $\lambda(G) = 1$; $\lambda(G) = p - 1$ jei grafas pilnas.

9 Teorema *Teisingos nelygybės:*

$$\chi(G) \leq \lambda(G) \leq \delta(G).$$

$\ominus$

Parodysime, kad $\chi \leq \lambda$. Jei $\lambda = 0$, tai $\chi = 0$. Jei $\lambda = 1$, tai nagrinėjame grafe yra tiltas, taigi arba yra sąlyčio taškas, arba $G = K_2$. Bet kokiu atveju $\chi = 1$. Tarkime, kad $\lambda \geq 2$. Tada pašalinę $\lambda - 1$ briauną gausime grafą G' turintį tiltą $(u, v.)$ Kiekvienai pašalinamai $\lambda - 1$ briaunai, pašalinkime incidentinę viršūnę, jei ji nėra u arba v. Po šio veiksmo likęs grafas nesusijęs, taigi $\chi \leq \lambda - 1 < \lambda$. Jei susijęs, tai pašalinsime vieną iš tilto galų u arba v. Taigi $\chi \leq \lambda$.

Tarkime, kad $\lambda \leq \delta$. Jei $\delta = 0$, tai $\lambda = 0$. Tarkime, kad viršūnė v turi mažiausią laipsnį: $d(v) = \delta$. Pašalinkime visas briaunas, incidentines v. Turėsime $\lambda \leq \delta$.

$\oplus$

7.6 Skiriančios aibės ir nepersikertančios grandinės.

Tarkime, kad u, v yra dvi grafo G negretimos viršūnės. Dvi grandines $[u, v]$, jungiančias taškus u, v vadinsime *viršūnėmis nesikertančiomis*, jei šių grandinių bendri taškai tik u, v. Dvi grandines $[u, v]$ vadinsime *briaunomis nesikertančiomis*, jei šios grandinės neturi bendrų briaunų. Jei dvi grandinės yra *viršūnėmis nesikertančios*, tai šios grandinės ir briaunomis nesikertančios. Grafo G *viršūnėmis nesikertančių* grandinių aibę žymėsime simboliu $P(u, v)$:

$$P(u, v) = \{[u, v]; [u, v]_1 \in P \wedge [u, v]_2 \in P \Rightarrow [u, v]_1 \cup [u, v]_2 = \{u, v\}\}.$$

Grafo G viršūnių (briaunų) aibė S skiria dvi viršūnes u, v, jei šios viršūnės priklauso skirtingoms grafo $G - S$ jungumo komponentėms. *Skirianti briaunų aibė* vadinama *pjūviu*. Šią aibę žymėsime simboliu $S(u, v)$:

$$S(u, v) = \{\omega \in V; G - S = G_1 \cup G_2,\ v \in G_1, u \in G_2\}.$$

Jeigu u ir v priklauso skirtingoms jungumo komponentėms, tai $|P(u, v)| = 0$ ir $|S(u, v)| = 0$.

10 Teorema (Mengerio) *Tarkime, kad u, v yra negretimos grafo G viršūnės. Minimalus viršūnių skaičius aibėje, skiriančioje u ir v lygus didžiausiam, viršūnėmis nesikertančiomis, paprastų grandinių $[u, v]$ skaičiui:*

$$\max |P(u, v)| = \min |S(u, v)|.$$

$\oplus$

Šios teoremos įrodymo nepateiksime.

Tarkime, kad G yra susijęs grafas, o viršūnės u, v yra negretimos. Matome, kad $|P| \leq |S|$. Iš tiesų, grandinė $[u, v]$ eina per S. Jeigu $|P| > |S|$, tai S būtų viršūnė, kuriai priklausytų daugiau negu viena viršūnė iš P. Tokiu būdu, visiems S ir visiems P turime, kad $|P| \leq |S|$. Vadinasi, $\max |P(u, v)| = \min |S(u, v)|$. Teoremos esmė tame, kad bet kokiame grafe egzistuoja aibės P ir S tokios, kad teisinga lygybė: $|P| = |S|$.

Pateiksime be įrodymų keletą teoremų, kurios susiję su Mengerio teorema.

11 Teorema *Tarkime, kad u ir v dvi, laisvai pasirinktos, grafo G negretimos viršūnės. Tada maksimalus briaunomis nesikertančių grandinių $[u, v]$ skaičius lygus mažiausiam briaunų skaičiui esančiam gradinių $[u, v]$ pjūvyje.*

12 Teorema *Tam, kad grafas G būtų $k-$ susijęs būtina ir pakankama, kad bet kokios dvi negretimos viršūnės būtų sujungtos ne mažiau negu k viršūnėmis nesikertančiomis paprastomis grandinėmis.*

Kitaip tariant, koks bebūtų grafas G, bet kokios dvi negretimos viršūnės sujungtos ne mažesniu negu $\chi(G)$ viršūnėmis nesikertančiomis paprastomis grandinėmis.

Tarkime, kad $S = \{S_1, \ldots, S_n\}$- aibės E poaibių šeima. Poaibiai S_i gali kirstis. Tada aibės S *skirtingų reiškimų sistema* vadinsime aibės E, m elementų poaibį $C = \{c_1, \ldots, c_m\}$, kai $c_i \in S_i$, $i = 1, \ldots, m$. Pastebėsime, kad aibės C visi elementai yra skirtingi.

Suporavimu (*nepriklausoma briaunų aibe*) vadinsime briaunų aibę, kurioje nėra dviejų gretimų briaunų. Nepriklausomą briaunų aibę (NBA) vadinsime maksimalia, jei bet koks šios aibės viršaibis nėra NBA.

Tarkime, kad $G(V_1, V_2, E)-$ dvidalis grafas. Idealia NBA iš V_1 į V_2 vadinsime NBA, kuri apima (dengia) visas viršūnes V_1. Pastebėsime, kad ideali NBA tuo pačiu ir maksimali. Atsakysime į klausimą, kada egzistuoja ideali NBA.

Panagrinėkime tokį uždavinį. Tarkime, kad V_1 yra vaikinų aibė ir vaikinai yra pažįstami su kai kuriomis merginomis iš merginų aibės V_2. Kiek yra galimybių suporuoti vaikinus su merginomis taip, kad visose porose būtų tik pažįstami asmenys? Taigi, V_1 ir V_2 yra dvi aibės, o briaunos šiuo atveju reiškia, kad vaikinas pažįstamas su mergina. Tada idealus NBA yra minėtas visų suporavimas, kai visose porose pažįstami asmenys.

Pasirodo, kad norint rasti aibės S skirtingų reiškimų sistemą, pakanka rasti idealų NBA iš S į E.

13 Teorema*Tarkime, kad $G(V_1, V_2, E)$ yra dvidalis grafas. Idealus NBA iš V_1 į V_2 egzistuoja tada ir tik tada, kai bet koks poaibis $A \subset V_1$, turi savybę $|A| \leq |\Gamma(A)|$.*

$\ominus$

Būtinumas. Tarkime, kad egzistuoja idealus NBA iš V_1 į V_2. Tada aibei $\Gamma(A)$ priklauso $|A|$ viršūnių iš V_2, kurios suporuotos su aibės A viršūnėmis. Taigi, $|A| \leq |\Gamma(A)|$.

Pakankamumas. Prie grafo G prijunkime dvi viršūnes u, v taip, kad u būtų gretima visoms V_1 viršūnėms, o $v-$ gretima visoms V_2 viršūnėms. Ideali NBA iš V_1 į V_2 egzistuoja tada ir tik tada, kai egzistuoja $|V_1|$ viršūnėmis nesikertančių paprastų grandinių $[u, v]$. Aišku, kad $|P(u, v)| \leq |V_1|$, kadangi V_1 skiria u ir v. Remiantis Mengerio teorema turime, kad $\max|P(u, v)| = \min|P(u, v)| = |S|$, S yra mažiausia aibė skirianti viršūnes u, v. Turime, kad $|S| \leq |V_1|$. Parodykime, kad $|S| \geq |V_1|$. Tegu $S = A \cup B$, $A \subset V_1$, $B \subset V_2$. Tada $\Gamma(V_1 \setminus A) \subset B$. Iš tiesų, jei $\Gamma(V_1 \setminus A) \not\subset, B$ tai egzistuotų kelias $[u, v_1, v_2, v]$ ir S nebūtų aibė aibė, skirianti viršūnes u, v. Taigi, $|V_1 \setminus A| \leq |\Gamma(V_1 \setminus A)| \leq |B|$. Turime, kad

$$|S| = |A| + |B| = |A| + |V_1 \setminus A| = |V_1|.$$

$\oplus$

7.7 Srautai. Maksimalaus srauto paieškos algoritmas

Šiame skyrelyje orentuoto grafo sąvoką tapatinsime su *tinklo* savoka.

Tarkime, kad $G(V, E)$ yra tinklas, o s ir t yra tinklo šaltinis ir santaka. Tinklo lankus susiesime su realiais neneigiamais skaičiais, kuriuos vadinsime lanko krūviais c, $c : E \to [0, +\infty)$. Skaičių $c(u, v)$ vadinsime lanko (u, v) *praleidžiamąja geba* (PG). Jei lanko PG lygi nuliui ($c(u, v) = 0$), tai šio lanko tinkle nėra.

Matricą C :**array** $[1 \ldots p, 1 \ldots p]$**of real** vadinsime PG matrica.

Tarkime, kad $f : E \to \mathcal{R}$ yra funkcija. Funkcijos f *divergencija* viršūnėje v vadinamas toks skaičius:

$$\mathrm{div}\,(f,u) = \sum_{\{v;(u,v)\in E\}} f(u,v) - \sum_{\{v;(v,u)\in E\}} f(u,v).$$

(Tai kas išėjo, minus tai kas atėjo.)

Funkciją $f : E \to \mathcal{R}$ vadinsime *srautu* tinkle G, jei ši funkcija tenkina tokius sąryšius:

1. $\forall (u,v) \in E,\ 0 \le f(u,v) \le c(u,v)$;
2. $\forall u \in V \setminus \{s,t\},\ \mathrm{div}\,(f,u) = 0$;

Divergenciją šaltinio taške vadinsime srauto *dydžiu*: $w(f) := \mathrm{div}\,(f,s)$.

Tegu P yra lanko (s,t) pjūvis, $P \subset E$. Kiekvienas pjūvis suskaido viršūnių aibę V į du poaibius: S,T, $S \subset V$, $T \subset V$, $S \cup T = V$, $S \cap P = \emptyset$, $s \in S$, $t \in T$ ir aibei P priklauso visi lankai jungiantys S ir T. Tada $P = P^+ \cup P^-$, čia P^+ sudaro visi lankai iš S į T, o P^- sudaro lankai iš T į S. Simboliu $F(P)$ žymėsime pjūvio P lankų srautų sumą. Aibės P visų lankų PG-ų sumą vadinsime *pjūvio PG* ir žymėsime simboliu $C(P)$. Trumpai:

$$F(P) = \sum_{e\in P} f(e), \quad C(P) = \sum_{e\in P} c(e).$$

2 Lema Teisinga lygybė:

$$w(f) = F(P^+) - F(P^-).$$

$\ominus$

Pažymėkime: $W = \sum\limits_{v\in S} \mathrm{div}(f,v)$, kai lankas $(u,v) \in E$. Jeigu $u,v \in S$, tai paskutinėje sumoje yra du lanką (u,v) atitinkantys dėmenys: $\mathrm{div}(f,u)$ ir $\mathrm{div}(f,v)$. Jų suma lygi 0. Jeigu $u \in S$, $v \in T$, tai į šią sumą patenka vienas dėmuo $\mathrm{div}(f,u)$ ir šių dėmenų suma lygi $F(P^+)$. Jei $u \in T$, $v \in S$, tai į šią sumą patenka vienas dėmuo $\mathrm{div}(f,v)$ ir šių dėmenų suma lygi $F(P^-)$. Taigi $w(f) = F(P^+) - F(P^-)$. Antra vertus $W = \sum\limits_{v\in S} \mathrm{div}(f,v) = \mathrm{div}(f,s) = w(f)$.

$\oplus$

3 Lema Teisinga lygybė:

$$\mathrm{div}(f,s) = -\mathrm{div}(f,t).$$

$\ominus$

Pažymėkime:

$$P := (S,T),\ \ S := V \setminus \{t\},\ \ T := \{t\}.$$

Tada

$$\mathrm{div}(f,s) = w(f) = F(P^+) - F(P^-) = F(P^+) = \sum_v f(v,t) = -\mathrm{div}(f,v).$$

$\oplus$

4 Lema Teisinga nelygybė:

$$w(f) \leq F(P).$$

$\ominus$

$w(f) = F(P^+) - F(P^-) \leq F(P^+) \leq F(P)$.

$\oplus$

5 Lema Teisinga nelygybė:

$$\max_f w(f) \leq \min_P C(P).$$

$\ominus$

Kadangi $w(f) \leq F(P)$, tai $\max_f w(f) \leq \min_P F(P)$. Remiantis apibrėžimu $F(P) \leq C(P)$. Vadinasi $\min_P F(P) \leq \min_P C(P)$. Naudodamiesi pastarosiomis nelygybėmis gauname, kad $\max_f w(f) \leq \min_P C(P)$.

$\oplus$

14 Teorema(Fordo Falkersono)*Maksimalus srautas tinkle lygus minimaliai pjūvio PG. T.y. egzistuoja srautas f^* toks, kad*

$$w(f^*) = \max_f w(f) = \min_P C(P).$$

$\ominus$

Parodysime, kad jei f yra maksimalus srautas, tai egzistuoja pjūvis P toks, kad $w(f) = C(P)$. Tegu G' yra grafas, gautas iš srauto G, pastarąjame pakeitus visų briaunų orientaciją. Apibrėžkime viršūnių aibę S tokiu būdu:

$$S := \{u \in V;\ \exists [s,u] \in G; \forall (u_i, u_{i+1}) \in [s,u],\ (u_i, u_{i+1} \in E) \Rightarrow f(u_i, u_{i+1}) <$$

$$C(u_i, u_{i+1}) \wedge (u_i, u_{i+1} \in E) \Rightarrow f(u_i, u_{i+1}) > 0\},$$

kitaip tariant išilgai grandinės $[s,u]$ srautas nėra maksimalus, o lankai prieš kryptį turi teigiamą srautą. Grandinė turinti minėtą savybę vadinama *augmentalia*. Taigi $S \neq \emptyset$. Tegu $T := V \setminus S$. Parodysime, kad $t \in T$. Tarkime priešingai, t.y $t \in S$. Tada, egzistuoja grandinė $[s,t] =: R$. Be to egzistuoja skaičius δ :

$$\delta := \min_{e \in R} \Delta(e), \quad \Delta(e) := \begin{cases} c(e) - f(e), & e \text{ orientuotas } R \text{ kryptimi}, \\ f(e) > 0, & e \text{ orientuotas kita negu } R \text{ kryptimi}. \end{cases}$$

Iš aibės S apibrėžimo išplaukia, kad $s \in S$. Padidinkime augmentaliosios grandinės srautą dydžiu δ :

$$f(e) := \begin{cases} f(e) + \delta, & e \text{ orientuotas } R \text{ kryptimi,} \\ f(e) - \delta, & e \text{ orientuotas kita negu } R \text{ kryptimi.} \end{cases}$$

Šis srautas tenkina sąlygą: $0 \leq f(e) \leq C(e)$, $\operatorname{div}(v) = 0$. Taigi, jei $t \in S$, tai sukonstravome srautą didesnį už maksimalų. Bet tai prieštarauja pradinei prielaidai, kad f maksimalus. Vadinasi $t \in T$ ir $T \neq \emptyset$. Taigi, S ir T skiria pjūvį P. Šiame pjūvyje visi lankai e^+ pakrauti ($f(e^+) = C(e+)$) o visi lankai e^- be krūvio ($f(e^-) = 0$), kadangi priešingu atveju aibę S būtų galima išplėsti. Tad turime, kad

$$w(f) = F(P^+) - F(P^-) = C(P^+),$$

o tai reiškia, kad P^+ yra ieškomas krūvis.

Maksimalaus srauto paieškos algoritmas

Žemiau pateiktas algoritmas apibrėžiamas naudojant lankų PG-ų matricą. Šiame algoritme yra naudojamos paskutiniosios teoremos įrodymo idėjos, kuomet sukonstruotos aibės S viršūnės yra jungiamos augmentaliomis grandinėmis su šaltiniu s. Tikriname ar $t \in S$. Jei taip, tai srautas nėra maksimalus ir jį didiname dydžiu δ. Tam, kad apibrėžtume augmentaliąsias grandines ir tuo pačiu nustatytume dydį δ algoritme naudojama tokia pagalbinė duomenų struktūra:

P :**array** $[1 \dots p]$ **of record**
s :**enum**$(-, +)$ (ženklas nustato lanko krypį)
n : $1 \dots p$ (būsimoji augmentaliosios grandinės viršūnė)
δ : **real** (dydis kuriuo bus didinams srautas)
end record

Maksimalaus srauto algoritmas

Įv: tinklas $G(V, E)$ su šaltiniu s ir santaka t, ir PG matrica C : **array** $[1 \dots p, 1 \dots p]$ **of real** .
Išv: maksimalaus srauto matrica F : **array** $[1 \dots p, 1 \dots p]$ **of real**
for $u, v \in V$ **do**
$F[u, v] := 0$
end for
M : (srauto didinimo iteracija)
for $v \in V$ **do**
$S[v] := 0; N[v] := 0; P[v] := (,,)$
end for
$S[s] := 1; P[s] := (+, s, \infty)$ (kadangi $s \in S$)
reapeat
$a := 0$ (S išplėtimo požymis)
for $v \in V$ **do**
if $S[v] = 1 \wedge N[v] = 0$ **then**
for $u \in \Gamma(v)$ **do**
if $S[v] = 0 \wedge F[v, u] < C[v, u]$ **then**

$S[u] := 1; P[u] := (+, v, \min(P[v] \cdot \delta, C[v,u] - F[v,u])); a := 1$
end if
end for
for $u \in \Gamma^{-1}(v)$ **do**
if $S[v] = 0 \wedge F[v,u] > 0$ **then**
$S[u] := 1; P[u] := (-, v, \min(P[v] \cdot \delta, F[u,v])); a := 1$
end if
end for
$N[v] := 1$
end if
end for
if $S[t]$ **then**
$x := t; \delta := P[t] \cdot \delta$
while $x \neq s$ **do**
if $P[x] \cdot s = +$ **then**
$F[P[x] \cdot n, x] := F[P[x] \cdot n, x] + \delta$
else
$F[P[x] \cdot n, x] := F[P[x] \cdot n, x] - \delta$
end if
$x := P[x] \cdot n$
end while
goto M
end if
until $a = 0$

Pradžioje imamas nulinis srautas. Pagrindiniame cikle, kurio pradžia žymima M yra didinamas srautas. Dėl šios priežasties cikle **repeat** yra plečiama viršūnių aibė S, kuri yra pasiekiama iš viršūnės s augmentaliomis grandinėmis. Ir jei į aibę S patenka viršūnė t tai srautas augmentaliosios grandinės $[s,t]$ kryptimi yra didinamas dydžiu δ ir pradedama nauja iteracija didinat srautą. Aibės S plėtimo procesas yra baigtinis, kadangi viršūnių aibė yra baigtinė, o pažymėtos viršūnės masyve N pakartotinai nėra nagrinėjamos. Jei aibės S plėtimo procesas baigiasi, ir t nepriklauso aibei S, tai remiantis Fordo-Falkersono teorema F yra maksimalus srautas.

7.8 Trumpiausi keliai. Trumpiausių kelių paieškos algoritmai

Šiame skyrelyje nagrinėsime keletą algoritmų, kurių dėka galima rasti trumpiausią trajektoriją, jungiančią bet kokias dvi grafo viršūnes. Žinome, kad ne visuomet egzistuoja kelias jungiantis dvi viršūnes. Tad dažnai tenka iš karto spręsti dvi problemas: ar egzistuoja kelias jungiantis dvi viršūnes ir jei taip, tai rasti šį kelią, o be to ir trumpiausią.

Tarkime, kad duotas digrafas $G(V,E)$, kuriame lankai pažymėti skaičiais (svoriais arba ilgiais). Tada šį grafą galime susieti su ilgių (svorių) matrica C :

$$C[i,j] = \begin{cases} 0, & i = j \\ c_{ij}, & \text{svoris lanko tarp } i \textbf{ ir } j \\ \infty, & \text{jei kelio nera} \end{cases}.$$

Kelio ilgiu vadinsime sumą visų lankų ilgių, sudarančių nagrinėjamą kelią. Kaip jau minėjome vienas svarbiausių uždavinių- rasti trumpiausią kelią.

Šią problemą spręsti praktiškai galime naudodami *Floido* algoritmą. Naudodami šį algoritmą galime rasti trumpiausią atstumą tarp dviejų grafo viršūnių. Informaciją apie kelią bus saugoma matricoje $H[1 \cdot p, 1 \cdot p]$, kai

$$H[i,j] = \begin{cases} k, \\ 0, \end{cases}$$

čia $H[i,j]$ yra priskiriama reikšmė lygi k, jei k yra pirmoji viršūnė esanti trumpiausiame kelyje, jungiančiame viršūnes i ir j ir $H[i,j]$ yra priskiriama reikšmė 0, jei tokio kelio nėra.

Matricos H eilė yra $O(p^2)$ ir be to yra $O(p)$ viršūnių. Taigi, saugant visus kelius siejančius viršūnes gali prireikti $O(p^3)$ atminties. Tačiau naudojant žemiau pateiktą algoritmą nereikia atmintyje saugoti visų kelių.

Bet koks konkretus kelias $[u,v]$ parenkamas iš matricos naudojant algoritmą:
$w := u$; **yield** w (pirmoji viršūnė)
while $w \neq v$ **do**
$w := H[w,v]$; **yield** w (kita viršūnė)
end while
Floido algoritmas

Įv: matrica $C[1 \ldots p, 1 \ldots p]$ (lankų ilgių matrica)
Išv: matrica $T[1 \ldots p, 1 \ldots p]$ (kelių ilgių matrica) ir matrica $H[1 \ldots p, 1 \ldots p]$
for i **from** 1 **to** p **do**
for j **from** 1 **to** p **do**
$T[i,j] := C[i,j]$
if $C[i,j] = \infty$ **then**
$H[i,j] := 0$ (iš i į j kelių nėra)
else
$H[i,j] := j$ (iš i į j kelias egzistuoja)
end if
end for
end for
for i **from** 1 **to** p **do**
for j **from** 1 **to** p **do**
for k **from** 1 **to** p **do**
if $i \neq j \wedge T[j,i] \neq \infty \wedge i \neq k \wedge T[i,k] \neq \infty \wedge (T[j,k] = \infty \vee T[j,k] > T[j,i] + T[i,k])$
then
$H[j,k := H[j,i]]$ (įsimenamas naujas kelias)
T[j, k]:=T[j, i]+ T[i, k] (įsimenamas paskutinio kelio ilgis)
end if
end for
end for
for j **from** 1 **to** p **do**
if $T[j,j] < 0$ **then**

stop (nėra sprendinių: viršūnė j patenka į neigiamo ilgio ciklą)
end if
end for
end for
Jei grafe egzistuoja ciklas su neigiamu svoriu, tai sprendinys neegzistuoja.

Keletas pastabų. Pastebėsime, kad algoritmas ne visada pateikia sprendinį, kadangi ne visada egzistuoja trumpiausias atstumas. Papildomas ciklas j atžvilgiu atlieka apsaugos funkciją, t.y. jei atsiranda neigiamas ilgis - darbas nutraukiamas.

Dekstry algoritmas

Naudodami žemiau pateiktą algoritmą galime rasti trumpiausią atstumą tarp viršūnių, jei lankų ilgiai nėra neigiami.

Įv: digrafas $G[V, E]$, lankų ilgių matrica C :**array** $[1 \dots p, 1 \dots p]$ **of real** viršūnės v, t

Išv: matrica T : **array**$[1 \dots p]$ **of real** , H : **array** $[1 \dots p,]$ **of** $0 \dots p$ Jei viršūnė v yra trumpiausiame kelyje nuo s prie t, tai $T[v]-$ trumpiausio kelio nuo s iki t ilgis; $H[v]-$ viršūnė esanti prieš pat viršūnę v trumpiausiame kelyje.

for v **from** 1 **to** p **do**
$T[v] := \infty$(trumpiausias kelias nežinomas)
$X[v] := 0$ (visos viršūnės nepažymėtos)
end for
$H[s] := 0$ (s neina prieš nėvieną viršūnę)
$T[s] := 0$ (trumpiausio kelio ilgis lygus 0)
$X[s] := 1$
$v := s$ (sekanti viršūnė)
M : (atnaujinimas žymės)
for $u \in \Gamma(v)$ **do**
if $X[u] = 0 \wedge T[u] > T[v] + C[v, u]$ **then**
$T[u] = T[v] + C[v, u]$ (rastas trumpesnis kelias iš s į u per v)
$H[u] := v$ (įsimename kelią)
endif
endfor
$t := \infty$; $v := 0$
(ieškosime trumpiausio kelio)
for$u$**from** 1 **to** p **do**
if $X[u] = 0 \wedge T[u] < t$ **then**
$v := u$; $t := T[u]$ (viršūnė v yra trumpiausio kelio aibėje S pabaiga.
endif
endfor
if $v = 0$ **then**
stop (nėra trumpiausio kelio iš s į t)
endif
if $v = t$ **then**

stop (rastas trumpiausias kelias iš s į t.)
endif
$X[v] := 1$ (rastas trumpiausias kelias iš s į v)
go M

Pastebėsime, kad paskutinysis algoritmas visada duoda atsakymą, jei lankų svoriai yra neneigiami. Tam pakankama sąlyga, kad visoms viršūnėms būtų tenkinama trikampio nelygybė.

$$\forall u, v, w; \quad d(u, v) \leq d(u, w) + d(w, v).$$

Kiek trumpai apie algoritmą. Kiekvienas algoritmo žingsnis prasideda nuo žymės M. Pradedama nuo viršūnės v, kuriai žinomas trumpiausias kelias nuo s. Kitaip tariant, jei $X[v] = 1$, tai $T[v] = d(s, v)$ ir visos viršūnės esančios kelyje $[s, v]$ apibrėžiamos vektoriumi H tenkina savybę:

$$\forall u, T[u] = 1 \Rightarrow T[u] = d(s, u) \wedge T[H[u]] = 1.$$

Pastebėkime, kad pirmame žingsnyje viršūnės v vietoje yra naudojama viršūnė s, kuriai trumpiausias kelias lygus nuliui. Tegu $T[u] = d(s, u)$, visoms pažymėtoms viršūnėms u. Nagrinėkime pažymėtą viršūnę v, kuri parenkama, jei tenkinama sąlyga:

$$T[v] = \min_{X[u]=0} T[u].$$

Aišku, kad jei žinomas kelias, einantis per pažymėtas viršūnes, tai tuo pačiu žinomas ir trumpiausias kelias, kadangi priešingu atveju tai prieštarautų viršūnės v parinkimui. (Tarę, kad $T[v] > d(s, v)$, t.y. tarę, kad kelias einantis per s į v nėra trumpiausias, turime kelyje sutikti nepažymėtų viršūnių. Tarkime ši nepažymėta viršūnė yra w, $T[w] = 0$. Tada, $T[w] = d(s, w) \leq d(s, v) < T[v]$, bet tai prieštarauja v parinkimui.)

7.9 Medžiai. Medžių savybės

Grafas, kuriame nėra ciklų yra vadinamas *mišku*. Susijęs miškas vadinamas *medžiu* (*laisvuoju medžiu*). *Jei grafas G yra miškas, tai $z(G) = 0$. Tarkime, kad u, v ne gretimos grafo G viršūnės ir $x \neq (u, v) \notin E$. Jei grafas $G + x$ turi tik vieną paprastą ciklą, tai $z(G + x) = 1$ ir toks grafas vadinamas subcikliniu.*

Susijusiame grafe G yra tenkinama nelygybė: $q(G) \geq p(G) - 1$. Jei $q(G) = p(G) - 1$, tai grafas vadinamas panašiu į medį.

Nurodysime, kokias savybes turi tenkinti grafas, kad jis būtų medis.

15 Teorema *Tarkime, kad $G(V, E)$ yra grafas su p viršūnėmis, q briaunomis, k jungumo komponentėmis ir z paprastais ciklais. Be to tarkime, kad x yra briauna, jungianti bet kokią porą negretimų viršūnių. Tada tokie tvirtinimai yra ekvivalentūs:*

1. G yra medis, $k(G) = 1 \wedge z(G) = 0$;

2. Bet kokios dvi grafo viršūnės sujungtos paprasta grandine, $\forall u, v, \exists[u, v]$;

3. G yra susijęs grafas ir kiekviena briauna yra tiltas;

$$k(G) = 1 \wedge \forall e \in E, k(G - e) > 1;$$

4. G yra susijęs grafas ir panašus į medį;

$$k(G) = 1 \wedge q(G) = p(G) - 1;$$

5. G yra aciklinis grafas ir panašus į medį;

$$z(G) = 0 \wedge q(G) = p(G) - 1;$$

6. G yra aciklinis ir subciklinis grafas;

$$z(G) = 0 \wedge z(G + x) = 1;$$

7. G yra susijęs, subciklinis ir nepilnas;

$$k(G) = 1 \wedge G \neq K_p \wedge p \geq 3 \wedge z(G + x) = 1;$$

8. G yra panašus į medį ir subciklinis;

$$q(G) = p(G) - 1 \wedge G \neq K_1 \cup K_3 \wedge G \neq K_2 \cup K_3 \wedge z(G + x) = 1.$$

Šio teiginio įrodymo nepateiksime.

Lapu vadinsime viršūnę, iš kurios neišeina briauna. Kelias iš šaknies į lapą vadinamas *šaka.* Sutvarkyto medžio ilgiausia šaka vadinama medžio *aukščiu.* Mazgo *lygiu* vadinsime atstumą nuo šaknies iki lapo. Ši terminologija siejama su grafų praktiniais taikymais.

Išvada *Kiekviename netrivialiame medyje egzistuoja bent du lapai.*

$\ominus$

Tarkime, kad $G(V, E)$ yra medis. Kadangi medis susijęs grafas, tai

$$\forall u_i \in V; d(v_i) \geq 1.$$

Tarkime priešingai. T.y. $\forall v_i \in \{1, \ldots, p\}, d(v) > 1$. Tada

$$2q = \sum_{i=1}^{p} d(v_i) > 2(p-1) + 1 > 2p - 1.$$

Bet $q = p - 1$, t.y. 2q=2p-2. Gauname prieštaravimą.

$\oplus$

7.10 Orientuoti, sutvarkyti ir binariniai medžiai. Medžių vaizdavimas

Orientuotu medžiu vadinsime digrafą tenkinantį savybes:

1. Egzistuoja vienintelė viršūnė (mazgas), kurio įėjimo laipsnis yra 0. Ši viršūnė vadinama *šaknimi*;

2. Visų viršūnių įėjimo laipsniai lygų 1.

3.Ttarp šaknies ir bet kokios viršūnės egzistuoja kelias.

16 Teorema *Jei medis sutvarkytas, tai:*

1. $q = p - 1$;

2. jei sutvarkytame medyje briaunas paliksime neorientuotas, tai gausime laisvąjį medį;

3. sutvarkytame medyje nėra ciklų;

4. egzistuoja vienintelis kelias jungiantis šaknį su bet kokia viršūne;

5. pografis, kurį sudaro viršūnės, pasiekiamos keliais iš viršūnės v *yra sutvarkytas medis su šaknimi* v;

6. Jeigu laisvame medyje, bet kokią viršūnę fiksuoti ir laikyti ją šaknimi, tai gausime sutvarkytą medį.

$\ominus$

1. Kiekviena briauna įeina į kokią nors viršūnę. Bet tada

$$\forall v \in V \setminus \{u\}, d^+(v) = 1.$$

Bet tada $q = p - 1$.

2. Tarkime, kad G sutvarkytas medis ir G' gautas iš G panaikinus pastarąjame briaunų orientaciją, $u-$ šaknis.

Tada $\forall v_1, v_2 \in V, \exists [v_1, u] \in G' \wedge \exists [u, v_2] \in G'$. Turime $\forall v_1, v_2 \, \exists [v_1, v_2]$, taigi grafas G' yra susijęs. Bet remiantis paskutiniąja teorema- šis grafas yra medis.

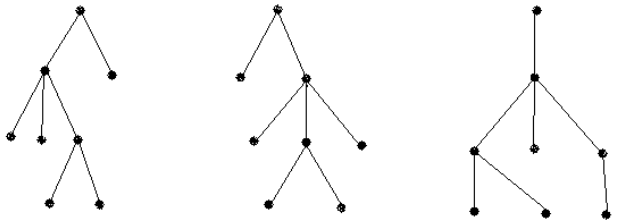
3. Šis atvejis išplaukia iš 2.

4. Tarkime priešingai. T.y. grafe G tarp viršūnių u, v egzistuoja du keliai. Bet tada grafe G' egzistuoja ciklas. Prieštaravimas.

5. Tarkime, kad G_v- taisyklingas pografis, apibrėžtas viršūnėmis, pasiekiamomis keliais iš v. Tada $d^+_{G_v}(v) = 0$, priešingu atveju mazgas v būtų pasiekiamas iš mazgo $v' \in G_v$. Bet tada grafe G_v, o tuo pačiu ir G egzistuotų kontūras. Bet tai prieštarauja 3. Turime, kad $\forall v' \in G_v \setminus \{v\}, \; d^+(v') = 1$, kadangi $G_v \subset G$. Visos G_v viršūnės yra pasiekiamos iš v. Bet tai reiškia, kad G_v yra medis.

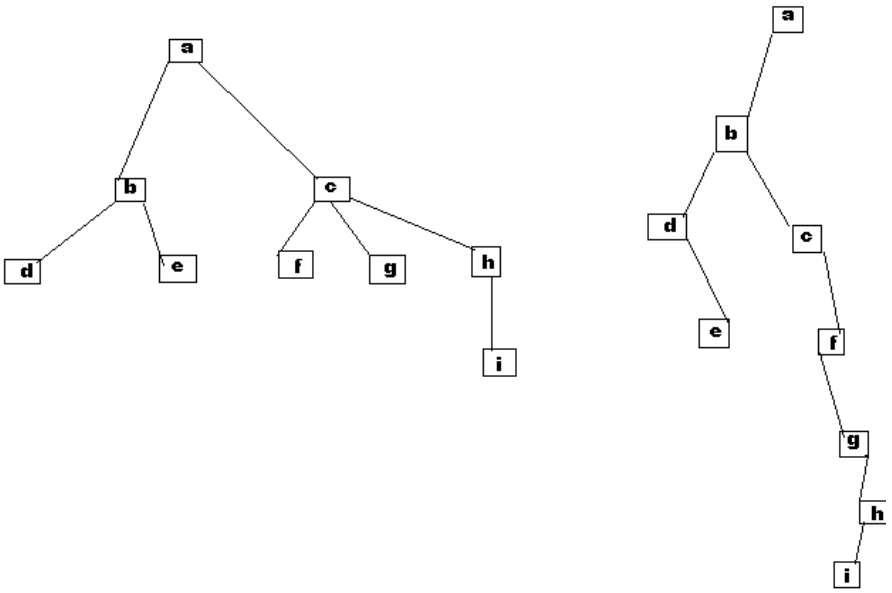
6. Tarkime, kad viršūnė u yra šaknis ir lankai orientuoti nuo šaknies. Tada $d^+(u) = 0$; ir $\forall v \in V \setminus \{u\}, \; d^+(v) = 1$. Taigi grafas yra orientuotas medis.

$\oplus$



Binariniu medžiu vadinsime aibę viršūnių, kurios arba tuščios arba kurios yra šaknys su dvi susikertančiais binariniais medžiais- kairiojo ir dešiniojo. Binarinis medis nėra sutvarkytas.

Kiekvieną medį galima orientuoti parinkus vieną viršūnę šaknimi. T.y. kiekvieną sutvarkytą medį galime sutvarkyti įvairiai. Be to, bet kokį sutvarkytą medį galima išreikšti binariniu medžiu, nurodant dešinįjį ir kairįjį ryšius. Beje, žemiau pateiktame pav. kairėje pusėje pateikiamas sutvarkytas medis, o dešinėje jam atitinkantis binarinis medis.



Pangrinėkime binarinių medžių reprezentavimo problemą. Tarkime, kad $n(p)$ atminties apimtis, reikalinga binariniam medžiui vizualizuoti.

1. Kiekvienas mazgas aprašomas dydžiu $N(l, r)$ apimančiu dešinį ir kairį mazgus

ir parametru i reikalingu saugoti informacijai apie mazgą. Medis charakterizuojamas šaknimi. Charakteristika N apibrėžiama tokiu būdu: $N =$**records** $i : info; l, r, :\uparrow N$ **end record**. Šiai reprezentacijai reikalinga atpitis su apimtimi $n(p) = 3p$.

2. Visos viršūnės patalpinamos masyve ir visos pomedžio, kurio šaknis v, viršūnės saugomos iš karto po jos. Kartu su kiekviena viršūne saugomas viršūnės indeksas. Medis T apibrėžiamas tokiu būdu: T : **array** $[1 \dots p]$**ofrecord** $i : info; k : 1 \dots p$ **endrecord**. Šiai reprezentacijai reikialinga atmintis lygi $n(p) = 2p$.

3. Vietoje sąryšių fiksuosime viršūnių specialius žymenis, pavyzdžiui, tegu 0 reiškia lapą, 1 reiškia kad viršūnė susieta kairiuoju ryšiu, bet nėra dešiniojo ryšio, 2 žymi, kad yra dešinysis ryšys, bet nėra kairiojo, o 3 yra abu ryšiai t.y. kairysis ir dešinysis. Medį apibrėžkime tokiu būdu: T : **array** $[1 \dots p]$**of record** $i : info; d : 0 \dots 3$ **end record**. Šiai reprezentacijai reikalinga $n(p) = 2p$ atmintis. Jei gu papildomai viršūnės laipsnis žinomas iš informacijos, kuri saugoma viršūnėje, tai galima laipsnio taipogi nesaugoti atmintyje. Šiuo atveju medžių reiškimas gana kompaktiškas- reiškimui reikalinga atmintis yra $n(p) = p$.

Paprastai algoritmai, kuriuose nagrinėjami medžiai, negali apsieiti be klaidžiojimo medžiuose. Visada medžiuose galime klaidžioti rekursyviai, tačiau toks paieškos būdas yra neefektyvus. Žemiau pateikiame dažniausiai naudojamus klaidžiojimus medžiuose:

1. kairysis klaidžiojimas-
a) patenkama į šaknį,
b) aplankomas kairysis pomedis,
c) aplankomas dešinysis pomedis;
2. simetrinis (atvirkštinis) klaidžiojimas -
a) aplankomas kairysis pomedis,
b) patenkama į šaknį,
c) aplankomas dešinysis pomedis;
3. dešinysis klaidžiojimas-
a) aplankomas kairysis pomedis,
b) aplankomas dešinysis pomedis,
c) patenkama į šaknį.

Pateiksime simetrinio klaidžiojimo, binariniame medyje, algoritmą.

Įv: binarinis medis, $r-$ šaknies nuoroda.
Iš: simetrinį klaidžiojimą atitinkanti viršūnių seka.
$T =: \emptyset$; $p := r$ (pradžioje stekas tuščias, o p parodo medžio šaknį)
M : (viršūnės, kurią nurodo p analizė)
$p =$ **nil then**
$T = \emptyset$ **then**
stop(klaidžiojimas baigtas)
end if
$p \leftarrow T$ (kairysis pomedis aplankytas)
yield p (eilinė viršūnė)
$p := p.r$ (pradedama klaidžioti dešiniąjame pomedyje)
else
$p \rightarrow T$ (prisimenama viršūnė)
$p := p.l$ (pradedamas klaidžiojimas kairiąjame pomedyje)

e ; **if**
go $\rightarrow M$

7.11 Rūšiavimo medžiai

Duomenų saugojimo bei jų paieškos mechanizmas paprastai vadinamas *asociatyviąja atmintimi (AA)*. Naudojant AA duomenys paprastai yra skirstomi į dalis (įrašus), kurie gali būti bet kokios prigimties ir bet kokio ilgio. Kiekvienas įrašas susiejamas su *raktu*. Raktas, tai kažkoks elementas iš visiškai sutvarkytos aibės. Jis paprastai būna paprastas, kompaktiškas ir patogus naudoti. Įrašas yra pasiekiamas naudojantis raktu.

Pateiksime keletą AA pavyzdžių.

1. Žodynas arba enciklopedija paprastai yra skirstomi į įrašus (stripsnius), o raktai, tai straipsnių pavadinimai.

2. Adresų knyga: raktas yra abonento vardas, o įrašas- informacija (tel numeris, adresas).

3. Bankų sąskaita: raktas yra sąskaitos numeris, o įrašas- finansinė informacija.

Dirbant su AA turi būti galimybė atlikti tokius veiksmus:

1) prijungti papidomą įrašą (raktą);

2) rasti įrašą (raktą);

3) pašalinti įrašą (raktą)

Šių operacijų efektyvumas priklauso nuo duomenų, naudojamų AA struktūros.

Aptarsime AA realizavimo metodiką. Tiksliau kalbant, kaip yra panaudojamos duomenų struktūros AA išreikšti.

Yra naudojami:

1. Nesutvarkyti masyvai;

2. sutvarkyti masyvai;

3. *rūšiavimo medžiai*- binariniai medžiai, kurių kiekviena viršūnė turi individualų raktą, be to raktų reikšmės kairiąjame pomedyje yra mažesnės už viršūnių raktų reikšmes dešiniąjame pomedyje.

4. Šef-lentelės.

Naudojant nesutvarkytus masyvus, algoritmai kurių pagalba realizuojama asociatyvi atmintis yra tokie:

1. operacija "prijungti" (įrašą, raktą) realizuojama atliekant įrašą masyvo pabaigoje;

2. operacija "paieška "(rakto, įrašo) realizuojama tikrinant visus įrašus (raktus).

3. operacija "pašalinti "(raktą, įrašą) realizuojama peržiūrint visus įrašus, o po to, pašalinus įrašą visus likusius perstumiame per vieną poziciją į priekį.

Panagrinėsime algoritmus, kuriuose realizuojama rūšiavimo medžio operacija.

Naudojant sutvarkytus masyvus, kuriais reiškiama AA įrašo paieškos operacija turint šio įrašo raktą vidutiniškai trunka $O(\log_2 n)$, n įrašų kiekis.

Binarinis algoritmas

Įv: Sutvarkytas masyvas A : **array** $[1 \dots n]$**ofrecord** $k : key; i :\ info$ **endrecord**; raktas $a : key$.

Išv: irašo indeksas kai žinomas raktas a masyve A arba išvedamas 0, jei įrašo su tokiu raktu nėra.

$b := 1$ (masyvo dalies pradinis indeksas, kai atliekama paieška)

$e := n$ (masyvo dalies pabaigos indeksas, kai atliekama paieška))
while $b \leq e$ **do**
$c := \frac{(b+e)}{2}$ (tikrinamo elemento indeksas)
if $A[c].k < a$ **then**
$e := c - 1$ (tęsiame paiešką pirmoje pusėje)
else if $A[c].k < a$ **then**
$b := c + 1$ (tęsiame paiešką antroje pusėje)
else
return c (rado ieškomą raktą)
end while
return 0 (ieškomo rakto nėra masyve)

Pateiksime algoritmą, kuriuo realizuojama viršūnės su nurodytu raktu, paieška rūšiavimo medyje.

Viršūnės paieška rūšiavimo medyje
Įv: rūšiavimo medis T, viršūnės raktas a.
Išv: rastos viršūnės rodiklis (nuoroda) p arba **nil**, jei nėra rakto a.
$p := T$ (nuoroda į tikrinamą viršūnę)
while $p \neq$ **nil do**
if$a < p.i$ **then**
$p := p.l$ (tęsiama paieška iš kairės)
else if $a > p.i$ **then**
$p := p.r$ (tęsiama paieška iš dešinės)
else
return p (viršūnė rasta)
end if
end while

Aptarsime algoritmą, kurio dėka atliekami įrašymai į rūšiavimo medį. Jeigu viršūnė su nurodytu raktu jau yra, tai įrašas neakliekamas.

Pradžioje aprašykime viršūnės kūrimo funkciją "NewNode."
Įv: raktas a.
Išv: nuoroda p į sukurtą viršūnę.
$new(p); p.i = a; p.l :=$ **nil**$; p.r :=$ **nil**
return p

Įv: rūšiavimo medis T, su nuoroda į viršūnę; raktas a.
Išv: modifikuotas rušiavimo medis.
if $T =$ **nil then**
$T := NewNode(a)$ (pirma viršūnė medyje)
return$T$
end if
$p := T$ (nuoroda į dabartinę viršūnę)
M : (analizė dabartinės viršūnės)
if $a < p.i$ **then**

if $p.l$ =**nil then**
$q := NewNode(a)$ (sukuriame naują viršūnę)
$p.l := q$ (prijungiame ją prie p iš kairės)
return$T$
else
$p := p.l$(ieškome vietos įrašams iš kairės)
goto M
end if
end if
if $a > p.i$ **then**
if $p.l =$ **nil then**
$q := NewNode(a)$ (sukuriame naują viršūnę)
$p.r := q$ (prijungiame ją prie p iš dešinės)
return$T$
else
$p := p.r$(ieškome vietos įrašams iš kairės)
goto M
end if
end if
return T

Paskutinysis įrašymo algoritmas analogiškas paieškos algoritmui: medyje yra ieškoma tokia viršūnė, kuri turi laisvą ryšį ir prie pastarojo jungiame naują viršūnę taip, kad būtų išlaikyta rūšiavimo medžio struktūra. T.y. jei naujas raktas mažesnis negu nagrinėjamas, tai naują viršūnę galime prijungti iš kairės arba reikia rasti kitą tinkamą vietą iš kairės. Analogiškai daroma iš dešinės, jei raktas didesnis.

Nagrinėsime viršūnės pašalinimo (ištrynimo) algoritmą. Prieš tai aprašykime dvi papildomas procedūras.

Procedūra Find

Įv: rūšiavimo medis T, su nuoroda į viršūnę; raktas a.

Išv: $p-$ nuododa į rastą viršūnę arba **nil**- jei medyje nėra tokio rakto; q nuoroda į viršūnės "tėvą" p, $s-$ būdas viršūnei q prijungti prie viršūnės p($s = -1$, jei p iš kairės nuo q; $s = +1$, jei p iš dešinės nuo q, $s = 0$, jei p šaknis.)

$p := T; q :=$ **nil**; $s := 0$
while $p \neq$**nil do**
if$p.i = a$**then**
returnp, q, s
end if
$q := p$ (išsaugoma reikšmė p)
if $a < p.i$ **then**
$p := p.l; s := -1$(paieška iš kairės)
else
$p := p.r; s := +1$(paieška iš dešinės)
end if

end while

Procedūra Delete

Įv: $p1-$ pašalinamos viršūnės nuoroda, $p2-$ viršūnės prie kurios jungiama nuoroda; $p3-$ jungiamos viršūnės nuoroda; $s-$ jungimo būdas

Išv: pertvarkytas medis

if $s = -1$**then**

$p2.l := p3$ (prijungiame iš kairės)

end if

if $s = +1$**then**

$p2.r := p3$ (prijungiame iš kairės)

end if

dispose(p1) (pašalina mazgą)

Aptarsime viršūnių pašalinimo (trinimo), iš rūšiavimo medžio, algoritmą.

Įv: rūšiavimo medis T, nuoroda į šaknį, raktas a.

Išv: modifikuotas rūšiavimo medis.

Find(T, a, p, q, s) (paieška šalinamos viršūnės)

if $p =$ **nil then**

return$T$ (nėra tokios viršūnės, veiksmas neatliekamas)

end if

if $p.r =$ **nil then**

$Delete(p, q, p.l, s)$ (žr pav. a)

else

$u := p.r$

if $u.l =$**nil then**

$u.l = p.l$

Delete(p, q, u, s) (žr pav. b)

else

$w := u; v := u.l$

while $v.l \neq$bf nil do

$w := u; v := u.l$

end while

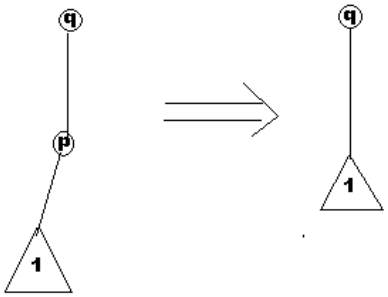
$p.i := v.i$ (prijungiame ją prie p iš kairės)

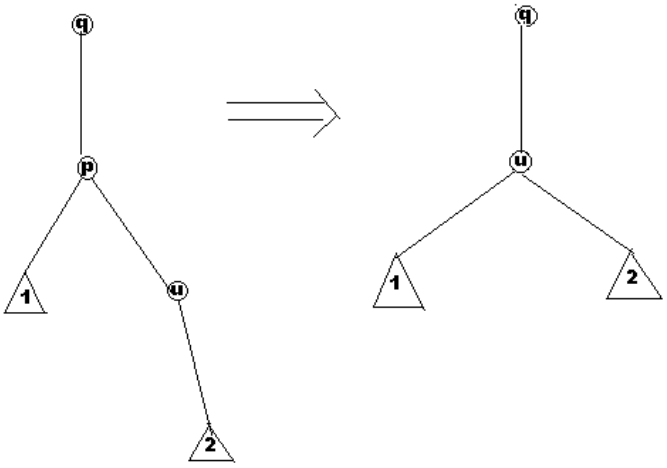
$Delete(v, w, v.r, -1)$(žr pav. c)

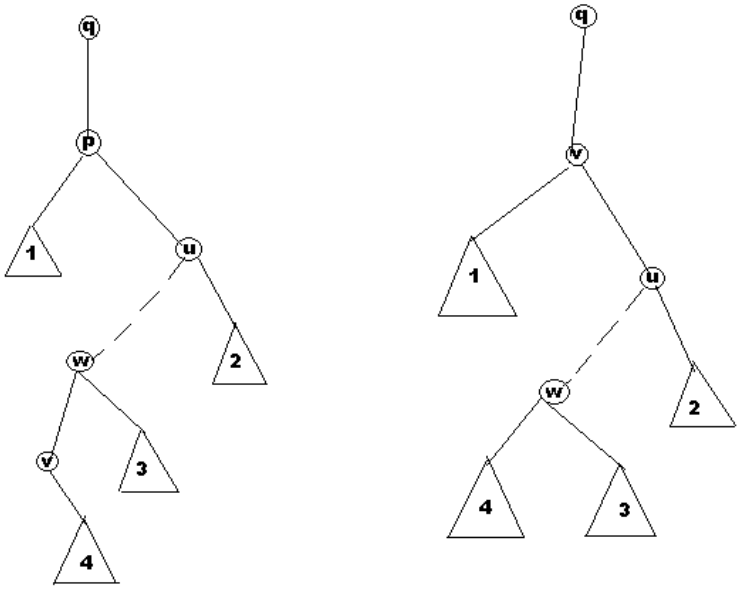
end if

end if

return$T$


q
p
1
q
1


q
p
1
u
2
q
u
1
2



Viršūnę pašalinant yra pertvarkomas visas rūšiavimo medis. Aptarkime visus tris galimus atvejus.

1. Dešinioji šalinamos viršūnės p šaka yra tuščia (911a.) Šiuo atveju kairysis viršūnės p pomedis 1 prijungiamas prie gimininges viršūnės q iš tos pačios pusės, iš kurios buvo pajungta viršūnė p.

2. dešinioji šalinamos viršūnės šaka yra netuščia ir jungiasi su viršūne u, kurios kairioji pusė yra tuščia (911b). Šiuo atveju viršūnės p kairysis pomedis 1 prijungiamas prie viršūnės u iš kairės, o pati viršūnė u pajungiama prie gimininges viršūnės q iš tos pusės iš kurios buvo pajungta viršūnė p.

3. Dešinioji viršūnės p šaka netuščia ir jungiasi su viršūne u, kurios kairioji šaka taip pat netuščia. Kadangi medis baigtinis, tai tai iš viršūnės u galima nusileisti iki viršūnės v, kurios kairoji šaka tuščia (žr. pav.). Šiuo atveju atliekamos dvi medžio transformacijos. Iš pradžių, informacija viršūnėje p keičiama viršūnės v informacija. Kadangi viršūnė v yra dešiniąjame viršūnės p pomedyje ir viršūnės u kairiąjame pomedyje, be to $p.i < v.i < u.i$. Taigi, rūšiavimo medžio savybės yra tenkinamos. Pastebėsime, kad dešinysis v viršūnės 4 pomedis prijungiamas iš kairės prie viršūnės w, o pati viršūnė v yra pašalinama. Kadangi 4pomedis priklausė viršūnės w kairiąjam pomedžiui, tai rūšiavimo medžio savybės yra tenkinamos.

Pažymėkime trumpiniais operacija:

$J-$ "prijungimo" operacija

$P-$ "paieškos" operacija

$S-$ pašalinimo operacija

NSM-nesutvarkytas masyvas; SM- sutvarkytas masyvas; RM- rūšiavimo medis.

operacija	NSM	SM	RM
J	$O(1)$	$O(n)$	$O(\log_2 n) \ldots O(n)$
P	$O(n)$	$O(\log_2 n)$	$O(\log_2 n) \ldots O(n)$
S	$O(n)$	$O(n)$	$O(\log_2 n) \ldots O(n)$

Operacijų efektyvumas yra apribotas medžio aukščiu.

7.12 Išlyginti ir subalansuoti medžiai

Sutvarkytas medis vadinamas *išlygintu* medžiu jei visos viršūnės, kurių laipsnis mažesnis negu 2 išsidėstę tame pačiame arba dviejuose paskutiniuose lygiuose. Tarkime, kad p yra viršūnių skaičius. Tada išlygintam medžiui egzistuoja aukščio h minimali reikšmė, sąlygota viršūnių skaičiaus.

16 Teorema *Jei medis išlygintas, tai*

$$\log_2(p+1) - 1 \leq h < \log_2(p+1).$$

$\ominus$

Tarkime, kad medžio lygis yra i. Tada šiame lygis maksimalus viršūnių skaičius yra 2^i. Vadinasi

$$\sum_{i=0}^{h-1} 2^i < p \leq \sum_{i=0}^{h} 2^i.$$

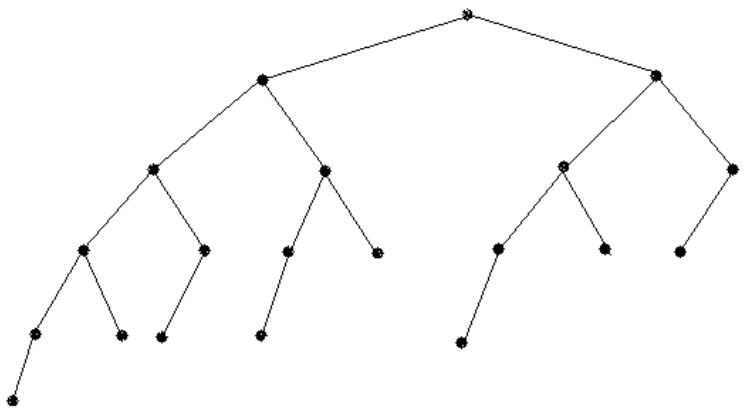
Iš paskutiniųjų nelygybių išplaukia, kad $2^h - 1 < p \leq 2^{h+1} - 1$ arba $2^h < p + 1 \leq 2^{h+1}$. Logaritmuodami gauname

$$h < \log_2(p+1) \wedge h + 1 \geq \log_2(p+1).$$

Taigi, $\log_2(p+1) \leq h < \log_2(p+1)$.

$\oplus$

Binarinį medį vadinsime *subalansuotu medžiu* jei bet kokio, to paties lygio kairiojo ir dešiniojo pomedžių viršūnių aukščiai skiriasi ne daugiau negu vienetu. Žemiau pateikiamas grafinis subalansuoto medžio pavyzdys.



17 Teorema *Jei medis subalansuotas, tai* $h < 2\log_2 p$.

$\ominus$

Panagrinėkime medį, kurio bet kokio kairiojo pomedžio aukštis vienetu didesnis negu dešiniojo aukštis. Jei fiksuosime viršūnių skaičių, tai tarp visų subalansuotų medžių, turinčių tą patį viršūnių skaičių, tokie medžiai turės didžiausią aukštį. Pažymėkime simboliu P_h- viršūnių skaičių tokiame medyje, kai aukštis h. Tada $P_h = P_{h-1} + P_{h-2} + 1$, be to $P_0 = 1, \;\; P_1 = 2, \;\; P_2 = 4$. Parodykime, kad $P_h \geq \sqrt{2}^h$. Naudosime indukcijos metodą. Turime, kad $P_0 = 1 \geq \sqrt{2}^0$. Taigi, aibė tų h kuriems nagrinėjama nelygybė teisinga yra netuščia. Pažymėkime šių h aibę T. Tegu $h \in T$. Tada $P_h \geq \sqrt{2}^h$. Parodykime, kad ir $h + 1 \in T$.

$$P_{h+1} = P_h + P_{h-1} + 1 \geq (\sqrt{2})^h + (\sqrt{2})^{h-1} + 1 = (\sqrt{2})^h(1 + \frac{1}{\sqrt{2}} + \frac{1}{(\sqrt{2})^h}) >$$

$$(\sqrt{2})^h(1 + \frac{1}{\sqrt{2}}) > (\sqrt{2})^h\sqrt{2} = (\sqrt{2})^{h+1}.$$

Taigi, visiems $h \in \mathcal{N}_0$, $P_h \geq \sqrt{2}^h$.

Turime, kad $\sqrt{2}^h \leq P_h$, vadinasi $h/2 \leq \log_2 P_h$ arba $h \leq 2\log_2 p$, kadangi $P_h = p$, čia p yra viršūnių skaičius.

$\oplus$

Pastebėsime, kad subalansuotuose medžiuose paieška atliekama lėčiau negu išlygintuose medžiuose. Tačiau praktikoje dažnai rūšiavimo medžius geriau reikšti subalansuotais medžiais, kadangi yra žinoma daug algoritmų, kuriuos naudojant viršūnių trinimui

arba įrašymui yra išlaikomas medžio subalansuotumas ir kas svarbu transformuojant medį transformuoti reikia ne visas viršūnes.

7.13 Ciklai

Tarkime, kad grafas yra jungus. Tada grafo *pjūviu* vadinsime briaunų aibę, kurias pašalinus iš grafo gaunamas nejungus grafas. Pjūvis bus vadinamas *paprastu*, jei neegzistuoja šios aibės poaibio, kuris yra pjūvis. Kuo daugiau grafe ciklų, tuo sunkiau grafą išskaidyti į dvi nesusijusias dalis. Tuo tarpu medžio bet kokia briauna yra pjūvis.

Jungų grafą vadinsime *Eilerio* grafu, jei egzistuoja šiame grafe ciklas, kuriam priklauso visos grafo briaunos tik po vieną kartą (kelis kartu briauna nėra aplankoma apeinant ciklą). Patį ciklą vadinsime *Eilerio ciklu*. Aišku, kad Eilerio ciklui priklauso ne tik visos briaunos bet ir visos viršūnės. Jei grafe egzistuoja grandinė, kuriai priklauso visos skirtingos grafo viršūnės, tai tokia grandinė vadinama *Eilerio* grandine.

18 Teorema *Jei grafas G yra netrivialus ir jungus, tai žemiau pateikti tvirtinimai yra ekvivalentūs:*

1) G yra Eilerio grafas;

2) kiekviena grafo G viršūnė turi lyginį laipsnį;

3) ciklo G briaunų aibę galima išskaidyti į paprastus ciklus.

$\ominus$

$1 \Rightarrow 2$ Tarkime, kad Z yra Eilerio ciklas. Judėdami šiuo ciklu skaičiuosime viršūnių laipsnius. Nesunku suprasti, kad aplankę viršūnę, jos laipsnį padidiname 2. Tad kiek bebūtų briaunų aibėje Z aplankius jas visas ir sudėję visų viršūnių laipsnius gausime, kad šių laipsnių suma- lyginis skaičius.

$2 \Rightarrow 3$ G yra netrivialus ir jungus, taigi $\forall v, d(v) > 0$. Viršūnių laipsniai yra lyginiai, vadinasi $\forall v, d(v) \geq 2$. Vadinasi:

$$2q = \sum d(v) \geq 2p \Rightarrow q \geq p \Rightarrow q > p - 1.$$

Matome, kad grafas G yra ne medis, vadinasi egzistuoja bent vienas paprastas ciklas Z_1. Tada aibė $G - Z_1$ yra pografis, kuriame vėl visos viršūnės turi lyginį laipsnį. Pašalinkime izoliuotas viršūnes. Tokiu būdu aibė $G - Z_1$ tenkina sąlygą 2. Vadinasi egzistuoja paprastas ciklas $Z_2 \subset G - Z_1$. Toliau išskiriame ciklą Z_i, tol, kol grafas bus netuščias. Turime: $E = \cup Z_i$ ir $\cap Z_i = \emptyset$.

$3 \Rightarrow 1$ Imkime kokį nors skaidinį Z_i iš aukščiau sudaryto skaidinio. Jei $Z_i = E$, tai teorema įrodyta. Priešingu atveju egzistuoja ciklas Z_j toks, kad

$$\exists v_i (v_i \in Z_i \wedge v_i \in Z_j),$$

nes grafas susijęs. Kelias $Z_i \cup Z_j$yra ciklas ir jam priklauso visos briaunos po vieną kartą. Jei $Z_1 \cup Z_2 = E$, tai teorema įrodyta. Jei ne, tai egzistuoja ciklas Z_k toks, kad $\exists v_2, (v_j \in Z_i \cup Z_j \wedge v_2 \in Z_3)$. Toliau mes didinsime Eilerio ciklą, kol neapims viso skaidinio.

$\oplus$

Pateiksime algoritmą, kurio dėka galima nustatyti Eilerio ciklą, jei žinoma, kad grafas - Eilerio.

Įv. Eilerio grafas $G(V,E)$, apibrėžtas sąrašu viršūnių $\Gamma(v)$, nurodant $\forall v$ su šia viršūne visas gretimas viršūnes.

Išv. Eilerio ciklo viršūnių seka.

$S := \emptyset$ (viršūnių saugojimo aibė (stekas))

select $v \in V$ (laisvai pasirenkame viršūnę)

$v \to S$ (priskiriame viršūnę aibei S)

while $S \neq \emptyset$ **do**

$v \leftarrow S;\ v \to S$

if $\Gamma(v) = \emptyset$**then**

$v \leftarrow S$ **yield** v

else

select $u \in \Gamma(v)$ (pasirenkame pirmą viršūnę iš gretimų viršūnių aibės)

$u \to S$(padedame viršūnę į aibę)

$\Gamma(v) := \Gamma(v) \setminus \{u\}; \Gamma(u) := \Gamma \setminus \{v\}$ (pašaliname briauną (u,v))

end if

end while

Pastaba. Pradedame nuo bet kokios viršūnės konstruojame kelią, pašalindami briaunas ir įsimindami viršūnes aibėje S iki tol, kol eilinės viršūnės gretimų viršūnių aibė bus tuščia, o tai reikš, kad kelio tęsti negalime. Pastebėsime, kad taip elgdamiesi mes būtinai pasieksime viršūnę nuo kurios pradėjome, kadangi priešingu atveju gautume, kad viršūnė v turi nelyginį laipsnį. Tad iš grafo buvo šalinamos briaunos, o viršūnės buvo saugomos aibėje S.

Be įrodymo pateiksime įdomų rezultatą.

19 Teorema Tarkime, kad $G(p)$ yra visų grafų, turinčių p viršūnių, aibė, o $E(p) \subset G(p)$ yra Eilerio grafai. Tada

$$\lim_{p\to\infty} \frac{|E(p)|}{|G(p)|} = 0.$$

Grafą vadinsime *Hamiltono* grafu, jei egzistuoja grandinė, kuriai priklauso visos grafo viršūnes po vieną kartą. Hamiltono grandinei priklauso nebūtinai visos briaunos. Jei grandinė yra ciklas, tai šis ciklas vadinamas *Hamiltono* ciklu.

20 Teorema *Tarkime, kad* $\forall v \in V, d(v) \geq p/2$. *Tada grafas* $G(V,E)$ *yra Hamiltono grafas.*

$\ominus$

Tarkime priešingai. T.y. ši savybė tenkinama, bet grafas nėra Hamiltono grafas. Prijunkime prie grafo G minimalų viršūnių skaičių $u_1, \ldots u_n$ tokių, kad grafas $G' := G + u_1 + \ldots + u_n$ butų Hamiltono grafas. Tegu $v, u_1, w, \ldots, v$ yra Hamiltono ciklas grafe G' $u_1 \notin G, v \in G$. Tokia pora u, v būtinai egzistuoja, kadangi priešingu atveju G būtų Hamiltono grafas. Tada $w \in G, w \notin \{u_1, \ldots, u_n\}$ kadangi priešingu atveju u_1 būtų nereikalinga. Be to viršūnė v gretima viršūnei w, kitu atveju u_1 būtų nereikalinga.

Jeigu cikle $v, u_1, w, \ldots, v', u', \ldots v$ egzistuoja viršūnė w' gretima viršūnei w, tai viršūnė v' negretima su viršūne v, kadangi priešingu atveju galima būtų sudaryti Hamiltono ciklą

$v, u_1, w, \ldots, v', u', \ldots v$ be viršūnės u_1, paėmus viršūnes $w, \ldots, v'$ atvirkščia tvarka. Tad gauname, kad grafo G' viršūnių, negretimų viršūnei v skaičius ne mažesnis negu skaičius viršūnių, gretimų viršūnei w. Bet kiekvienai grafo viršūnei $w \in G$ turime, kad $d(w) \geq p/2 + n$ ir $d(v) \geq p/2 + n$.

Bendras viršūnių, gretimų ir negretimų viršūnių skaičius, viršūnei v yra lygus $n+p-1$. Taigi:

$$n + p - 1 = \overline{d}(v) + d(V) \geq d(w) + d(v) \geq \frac{p}{2} + n + \frac{p}{2} + n = 2n + p.$$

Kadangi nelygybė $0 \geq n + 1$ yra klaidinga, tai prielaida taip pat klaidinga.

$\oplus$

21 Teorema Tarkime, kad $G(p)$ yra visų grafų, turinčių p viršūnių, aibė, o $H(p) \subset G(p)$ yra Hamiltono grafai. Tada

$$\lim_{p \to \infty} \frac{|H(p)|}{|G(p)|} = 1.$$

Taigi, asimptotiškai beveik visi grafai yra Hamiltono grafai.

Uždaviniai

1. Naudodami grafų teorijos sąvokas aprašykite ekvivalentumo sąryšį kokioje nors baigtinėje aibėje.
2. Įrodykite, kad jei grafas netrivialus, tai egzistuoja to paties laipsnio viršūnės.
3. Įrodykite, kad jei f yra netiesinės tvarkos sąryšis, tai grafas neturi kontūrų.
4. Įrodykite, kad jei $\delta(G) > \frac{(p-1)}{2}$ tai grafas susijęs.
5. Nupieškite visų medžių, turinčių 7 viršūnes, diagramas.

Literatūra

1. M. Bloznelis, Kombinatorikos paskaitų ciklas (metodinė priemonė), VU, 1996
2. Bulota K, Survila P. Algebra ir skaičių teorija. I dalis . Vilnius: Mokslas, 1976.
3. D. Jungnickel, Graphs, networks and algorithms, Springer, 2004
4. E. Manstavičius, Kombinatorikos ir grafų teorijos pradmenys (paskaitų konspektas) http://www.mif.vu.lt/katedra/ttsk/bylos/man/man.html
5. F. A. Novikov, Diskrečioji matematika programuotojams, Piter, 2001 (rusų kalba)