Vilniaus universitetas

Informatikos ir informatinio mąstymo mokomoji veikla

Grafų algoritmai pagrindiniam ugdymui

Mokyklos pedagogika

Kuriame Lietuvos ateitį
2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programa

Vilniaus universitetas

Informatikos ir informatinio mąstymo mokomosios veiklos sukurtos įgyvendinant projektą „Aukštųjų mokyklų tinklo optimizavimas ir studijų kokybės gerinimas Šiaulių universitetą prijungiant prie Vilniaus universiteto“ (Nr. 09.3.1-ESFA-V-738-03-0001), finansuojamą iš Europos socialinio fondo lėšų pagal 2014–2020 metų Europos Sąjungos fondų investicijų veiksmų programos 9 prioriteto „Visuomenės švietimas ir žmogiškųjų išteklių potencialo didinimas“ įgyvendinimo priemonę Nr. 09.3.1-ESFA-V-738 „Aukštųjų mokyklų tinklo tobulinimas“.

Metodinė medžiaga „Grafų algoritmai pagrindiniam ugdymui“ skirta Mokyklos pedagogikos studijų programos moduliui „Informatikos didaktika". Tikslinė grupė – būsimi informatikos pagrindinio ar vidurinio ugdymo mokytojai. Medžiaga siejasi su Bendrosiomis informatikos ir matematikos programomis, algoritmų ir programavimo pasiekimų sritimi. Atlikdami veikloje numatytas užduotis mokiniai išsiaiškins, kas yra grafas, kokie yra populiariausi grafų algoritmai, kaip ir kur juos taikyti. Aptariami Oilerio kelio ir Hamiltono ciklų paieškos metodai, paieškos gilyn ir platyn algoritmai grafe, taip pat artimiausiojo kaimyno ir pigiausios grandies algoritmai. Pateikiamas kiekvieno algoritmo teorinis paaiškinimas ir įvairios užduotys.

Autorius: dr. Vaidas Giedrimas

Redagavo: Viktoras Dagys

Vilnius, 2022

Tikslas

Susipažinti su grafais ir algoritmais jiems apdoroti.

Ryšys su bendrosiomis programomis

Informatika: duomenų vaizdavimas.

Informacinės technologijos: piešimas kompiuteriu, duomenų apdorojimas ir pateikimas skaičiuokle.

Integracija su matematika: matematinis mąstymas, problemų sprendimas.

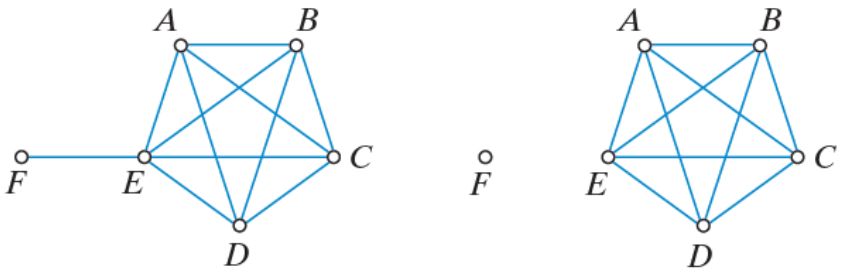
Grafų algoritmai

Grafo sąvoka

Įsivaizduokime, kad turime keletą sujungtų objektų. Šalies miestai sujungti keliais, elektronikos komponentai sujungti laidais, tarpusavyje draugauja (arba ne) socialinių tinklų vartotojai. Toks junginys vadinamas **grafu**. Jo objektai vadinami **viršūnėmis**, o jungiančios „linijos“ – **briaunomis**. Viršūnės, sujungtos briauna, vadinamos **gretimomis** arba *incidenčiomis briaunai*, nepaisant jų fizinio atstumo diagramoje. Jei yra sutarta, kad objektas gali būti sujungtas ir pats su savimi (pvz. socialinių tinklų vartotojas gali skirti *patiktuką* ir savo įrašui), tokia briauna vadinama **kilpa**. Jei du objektus jungia dvi ar daugiau briaunų, jos vadinamos **kartotinėmis**. Taip, pavyzdžiui, gali būti žemėlapyje – miestus gali jungti du ar daugiau kelių. Elektronikos, santechnikos, transporto srityse kartais judėjimas leidžiamas tik viena kryptimi. Jei briaunos turi kryptis, tada grafas vadinamas **orientuotuoju grafu**. Briaunos gali būti nevienodo *svorio*. Pvz., tarp dviejų miestų yra skirtingos būklės ar skirtingo ilgio keliai. Tokiu atveju juos atitinkančių briaunų svoriais galime laikyti užgaištą kelionei laiką arba reikiamus nuvažiuoti kilometrus.

Tiriant grafus, naudojamos tam tikros jų charakteristikos. **Keliu** grafe vadinama viršūnių ir briaunų seka, pvz., *v1*, *b1*, *v2*, *b2*, *v3*, *b3*, *v4*. Jeigu kelio pradinė ir galutinė viršūnės sutampa, jis yra vadinamas uždaru, o jei nesutampa – atviru.

Kiekviena viršūnė turi savo **laipsnį**, kuriuo vadinamas *incidenčių* jai (t. y. sujungtų su ja) briaunų skaičius. Orientuotuose grafuose atskirai skaičiuojamas į viršūnę *įeinančių* briaunų skaičius (vadinamas **įėjimo laipsniu**) ir išeinančių briaunų skaičius (**išėjimo laipsnis**). Grafas vadinamas **jungiu**, jei tarp bet kurių skirtingų viršūnių yra kelias. Pvz., jei tarp bet kurių pasaulio miestų egzistuotų skrydžių lėktuvais kelias (nesvarbu su kiek persėdimų), pasaulis būtų vadinamas jungiuoju. 1 pav. kairėje matome jungųjį grafą, o dešinėje – nejungųjį (nes viršūnė F yra „sala“). Salą gali sudaryti nebūtinai viena viršūnė. Jungios grafo dalys (maksimalūs jungūs *pografiai*) yra vadinamos **komponentėmis**. Briauna, kurią pašalinus iš grafo, komponenčių skaičius padidėja, yra vadinama **tiltu**.

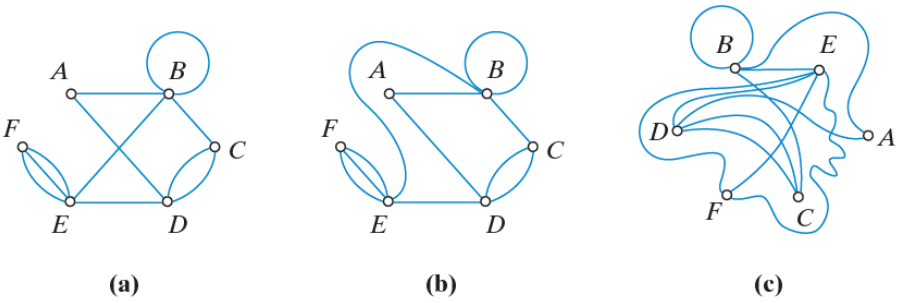


1 pav. Jungusis ir nejungusis grafai

Grafų vaizdavimas

Matematikoje grafas aprašomas kaip viršūnių aibės ir briaunų aibės pora: $G=(V, B)$.

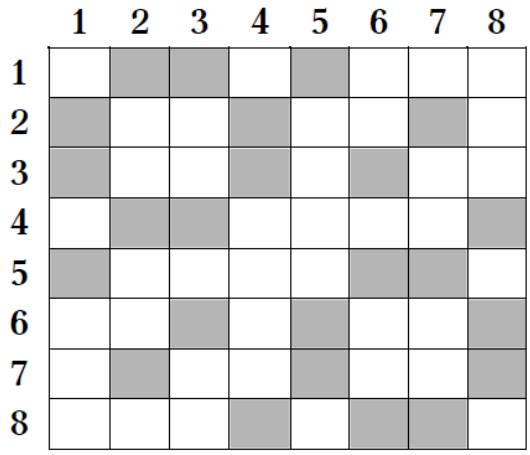
Sprendžiant praktinius uždavinius pasitelkiamas vaizdavimas masyvais ar net grafinis vaizdavimo būdas. Pastarasis ypač populiarus įvairiuose žemėlapiuose, hierarchijos ir kt. schemose. Pastebėtina, kad nors tas pats grafas gali būti pavaizduotas skirtingai, jo esmė nuo to nesikeičia. Štai 2 pav. visose diagramose pavaizduotas vienas ir tas pats grafas. Tokių skirtingai atrodančių, bet savo esme tapačių grafų pavyzdys – metro stotelių schema. Kartais ji pateikiama žemėlapyje pagal faktinę stotelių (viršūnių) buvimo vietą, o kartais (siekiant supaprastinti) – stilizuotai (panašiai kaip 2a pav.).



2 pav. Skirtinga grafo topologija

Grafai taip pat gali būti vaizduojami ir kitais būdais, pavyzdžiui, kaimynystės matrica (3 pav.), kurioje tamsus langelis rodo, kad yra lankas tarp viršūnių. Atkreipkite moksleivių dėmesį, kad tokia matrica yra simetriška vienos įstrižainės atžvilgiu ir joje saugoma informacija yra perteklinė, nebent būtų siekiama didesnio informacijos saugojimo patikimumo. „Nusitrynus“ vienam langeliui informacija apie ryšį dar nebūtų prarasta. Svoriniuose grafuose vietoj „pilko“ langelio arba „1“ langelyje būna įrašytas atstumas (kelionės trukmė, svoris, kaina...). Aptarkite su

mokiniais tuos atvejus, kai tas pačias viršūnes gali jungti du skirtingi kryptiniai lankai su skirtingais svoriais, pavyzdžiui, į kalną važiuojama lėčiau, nei į nuokalnę.



	1	2	3	4	5	6	7	8
1		■	■		■			
2	■			■			■	
3	■			■		■		
4		■	■					■
5	■					■	■	
6			■		■			■
7		■			■			■
8				■		■	■	

3 pav. Grafo vaizdavimas kaimynystės matrica

1-oji veikla

Veikla siekiama supažindinti su skirtingais grafe saugomos informacijos atvaizdavimo būdais, grafo transformacija.

Duotas grafo piešinys. Pagal jį reikia sukonstruoti panašią struktūrą iš duotų medžiagų:

- Sagų ir siūlų;
- Modelino ir makaronų;
- Gamtinių medžiagų (smilgų, akmenukų..), jei pamoka vyktų lauke.

Dabar pabandykite struktūrą transformuoti, keičiant viršūnių (sagų, akmenukų, modelino rutuliukų) padėtį, tačiau stengiantis nenutraukti esamų lankų (siūlų, makaronų, smilgų). Ar atlikus pakeitimus grafas išoriškai pasikeitė? O ar pasikeitė jo saugoma informacija (kas su kuo yra susiję)?

Alternatyvi veikla

Duotas grafo piešinys. Nupieškite kuo panašesnį Jūsų naudojama grafikos rengykle (pvz., naudojant „MS Word“ ar „MS Powerpoint Shapes“ elementus). Tada bandykite „patampyti“ viršūnes ir pakeisti jų vietą.

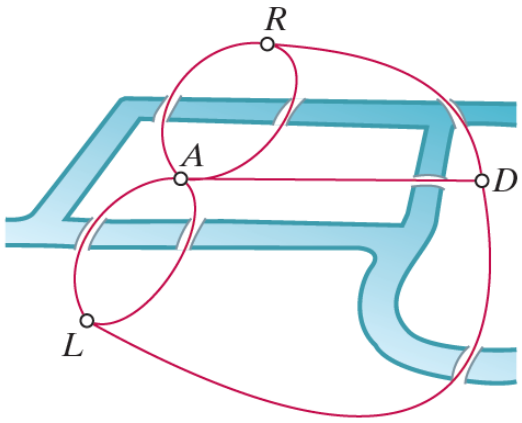
Tuomet vėl reikėtų atsakyti į pagrindinės veiklos klausimus.

1-oji užduotis

Šią užduotį reikia atlikti dirbant 4–5 asmenų grupėje. Nubraižykite jūsų bendravimo pasirinktame socialiniame tinkle intensyvumo grafą. Grafe viršūnėmis vaizduokite žmones (iš tos pačios grupės, kurioje atliekate užduotį), o briaunomis – žinutes ir (arba) komentarus vienas kito atžvilgiu. Suprantama, kad grafas bus kryptinis, o jo briaunos – skirtingų svorių. Kaip svorio (svertinį) koeficientą galite nurodyti žinučių (komentarų) skaičių. Jei bendraujate itin intensyviai, apsiribokite pastarosios dienos ar valandos faktais. Jei bendravimas yra retas, tuomet galima grafą vaizduoti su kartotinėmis briaunomis.

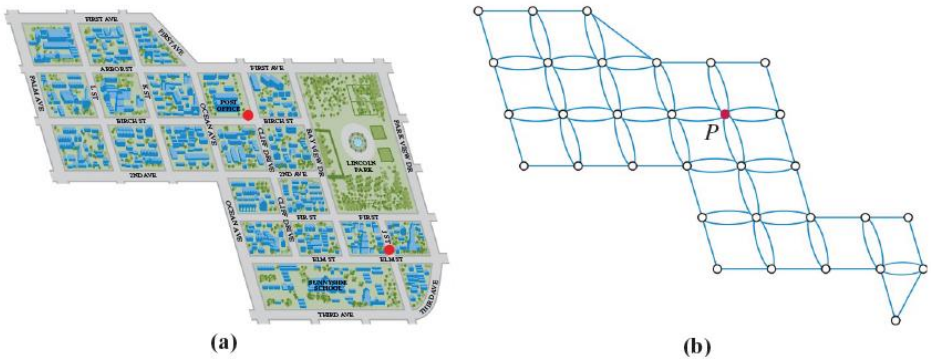
Kokie uždaviniai yra susiję su grafais?

Grafai ir jų savybės gali būti tiriami patys savaime arba pasitelkiami kaip kitų uždavinių sprendimo priemonė, jei sprendžiamas uždavinys gali būti atvaizduotas grafu. Klasikinės grafų teorijos problemos – kelio paieška arba galimų kelių savybių įvertinimas. Pirmuoju atveju ieškoma atsakymo į klausimą, ar egzistuoja tam tikras kelias grafe. Klasikinis šio tipo uždavinių pavyzdys – Septynių Karaliaučiaus tiltų problema (4 pav.).



4 pav. Karaliaučiaus tiltai, pavaizduoti grafu

Antruoju atveju dažniausiai siekiama *apeiti* visas viršūnes – tai vadinama **keliaujančio pirklio uždaviniu**. Egzistuoja ne vienas būdas (ir ne vienas kelias) tai padaryti, tačiau siekiama sugaišti mažiau laiko, sunaudoti mažiau kitų išteklių, gauti daugiausia naudos ir pan. Tokiuose uždaviniuose specialiais algoritmais vertinama briaunų svoriai ir kitos jų savybės. Pavyzdžiui, galima tirti, kuris kelias iš galimų yra pats efektyviausias patruliuojančiam policininkui konkrečiame mieste (5 pav.).

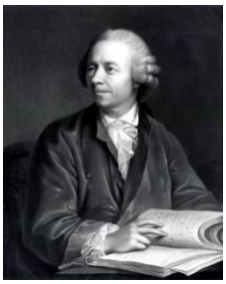


5 pav. Policijos patrulio maršruto grafas

Grafai yra naudojami ne vien transporto ar telekomunikacijų srityse. Naudojant grafą galima spręsti optimalaus tvarkaraščio sudarymo uždavinį arba tokius uždavinius kaip Keturių spalvų teorema (kuomet siekiama nuspalvinti žemėlapį taip, kad gretimos šalys nebūtų spalvinamos ta pačia spalva, tačiau iš viso panaudotų spalvų skaičius būtų minimalus). Mokiniams galima pasiūlyti padiskutuoti, kokius uždavinius dar būtų galima spręsti naudojant grafą. Diskusijos gylis priklausys nuo to, kiek detaliai bus panagrinėta, kokius realaus pasaulio objektus ir reiškinius galime atvaizduoti grafu.

Oilerio kelio paieškos algoritmas

Minėtame Septynių Karaliaučiaus tiltų uždavinyje (4 pav.) reikalaujama rasti tokį kelią per visus septynis šio miesto tiltus, kad kiekvienu tiltu būtu praeita tik po kartą. Toks kelias grafų teorijoje dabar yra vadinamas **Oilerio keliu**, šveicarų matematiko Leonardo Oilerio (*Leonhard Euler)*, daug prisidėjusio prie grafų teorijos, garbei.



6 pav. Leonardas Oileris (Leonhard Euler)

Šaltinis: http://s.bourdreux.free.fr/cabinet_Sigaud/chronologie/euler.jpg

Į klausimą, ar šis uždavinys turi sprendinį, ar ne, atsako pirmoji Oilerio teorema:

Jeigu grafas yra jungusis ir turi lygiai dvi nelyginio laipsnio viršūnes, tai jame yra Oilerio kelias. Jeigu grafas yra nejungus arba turi daugiau nei dvi nelyginio laipsnio viršūnes, tai toks kelias neegzistuoja.

Jei nustatoma, kad kelias egzistuoja, reikia jį rasti. Tai padeda atlikti Flero algoritmas:

1. Pirmiausia nustatoma, ar Oilerio kelias egzistuoja;
2. Pradedama nuo bet kurios viršūnės;
3. Keliaujama briauna, jei
    a) ji nėra nekeliautos dalies tiltas arba
    b) nėra kito pasirinkimo
4. Sunumeruojame briaunas perėjimo eilės tvarka;
5. Jei nebegalima toliau judėti – sprendinys gautas!

Pirmoji Oilerio teorema neapima visų galimų variantų (pvz., turime grafą su dviem viršūnėmis ir vienintele briauna), todėl ją patikslina antroji ir trečioji teoremos.

Antroji Oilerio teorema skamba taip:

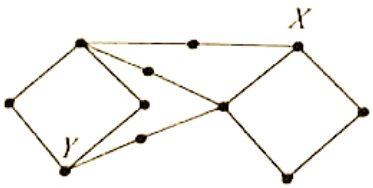
Jei jungusis grafas turi lygiai dvi nelyginio laipsnio viršūnes, tai jis turi bent vieną Oilerio kelią. Paprastai toks kelias prasideda vienoje nelyginio laipsnio viršūnėje, o baigiasi – kitoje.

Trečioji Oilerio teorema:

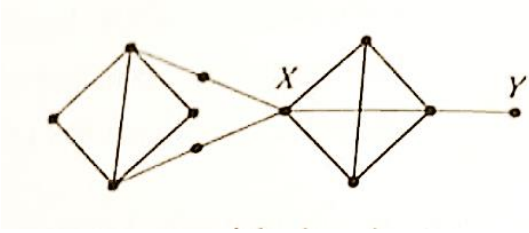
Visų grafo viršūnių laipsnių suma yra lyginis skaičius (lygus dvigubam briaunų skaičiui). Bet kurio grafo nelyginio laipsnio viršūnių skaičius yra lyginis.

2-oji užduotis

Raskite Oilerio kelią (7 pav.), kuris prasideda viršūnėje *X* ir baigiasi viršūnėje *Y*.



a)



b)

7 pav.

Kai grafas neturi lyginio laipsnio viršūnių, nesvarbu nuo kurios viršūnės pradėti. Kai turi tik dvi lyginio laipsnio viršūnes, jos ir tampa pradžios bei pabaigos taškais. O ką daryti, jei grafe yra 4, 6 ar daugiau lyginio laipsnio viršūnių? Šiuo atveju uždavinio sąlyga šiek tiek keičiasi: mes ieškome ne kelio aplankant kiekvieną briauną tik po vieną kartą, bet tokio, kad pasikartojančių briaunų būtų kuo mažiau. Kartais uždavinio sąlyga leidžia prijungti papildomas briaunas. Toks papildomų briaunų prijungimas prie esamo grafo vadinamas jo **oilerizavimu**. Svarbu pastebėti, kad papildomos briaunos jungiamos tik dubliuojant jau esamas (jei tam tikri taškai nėra sujungti, tarp jų papildomos briaunos konstruoti negalima) ir tik problemines viršūnes, kurių laipsnis yra nelyginis.

Moksleiviams oilerizavimo sąvoką galima būtų paaiškinti kurjerių bendrovės arba miesto transporto įmonės pavyzdžiais. Pavyzdžiui, Grafiškių mieste siekiama, kad autobusai nevažiuotų vieni paskui kitą, bet būtų aptarnautas tam tikras kiekis stotelių. Kaip esamą autobuso maršrutą būtų galima pakeisti, kad kiekviena gatve autobusas važiuotu kuo mažiau kartų?

Hamiltono ciklo paieškos algoritmai

Vienas iš klasikinių grafų uždavinių – **keliaujančio pirklio uždavinys** (5 pav.). Pirklys (prekybos agentas, kurjeris ir pan.) turi aplankyti visus klientus (viršūnes), kelionė turi prasidėti ir baigtis

toje pačioje viršūnėje (nes ten pirklys gyvena, ten yra kurjerio biuras ir pan.). Koks yra pigiausias pirklio kelionės maršrutas, jei viršūnes jungiantys keliai (briaunos) yra skirtingos vertės? Vertė (kaina) čia gali būti išreikšta kuro sąnaudomis, atstumu tarp viršūnių, sugaištu laiku tam atstumui įveikti ir pan.

Kuo uždavinys skiriasi nuo Oilerio ciklo (ar kelio) paieškos? Skirtumai būtų šie:

- Karaliaučiaus tiltų ir į jį panašiuose uždaviniuose siekiama po vieną kartą panaudoti grafo briaunas, o čia kalbama apie vienkartines **viršūnes**.
- Tikėtina, kad Hamiltono ciklų ar kelių yra daug, tačiau reikia **atrinkti optimalų**.

Prisiminkime, kad jeigu uždarame kelyje apeiname visas viršūnes lygiai vieną kartą, tokį kelią vadiname **ciklu**. Jei kelias atviras (pradžios ir galo viršūnės nesutampa), tai sakoma, kad turime Hamiltono **kelią**, bet ne ciklą.

Į klausimą, ar šis uždavinys turi sprendinį, ar ne, kitaip nei Oilerio kelio atveju, paprasto atsakymo nėra. Jeigu grafas turi palyginti daug briaunų, tai tikėtina, kad jame bus Hamiltono kelias arba ciklas. Čia remiamasi Ore teorema:

Jeigu bet kurių dviejų negretimu viršūnių laipsnių suma yra didesnė arba lygi grafo eilei, tai grafas turi Hamiltono ciklą.

Praktiškai tai reiškia, kad Hamiltono ciklą turi visi grafai, kurių viršūnės gretimos bent pusei kitų viršūnių.

Klasikinis Hamiltono ciklo paieškos būdas – empirinis (t. y. bandymų ir klaidų) metodas nagrinėjant visus galimus atvejus. Šis metodas kartais dar vadinamas jėgos algoritmu (angl. *brute force*), nes nesiimama jokių pagudravimų, tikimasi tikslą pasiekti sunkiu darbu.

Kadangi uždavinio sudėtingumas auga priklausomai nuo viršūnių ir briaunų skaičiaus, galima sprendimą palengvinti pirma sudarius galimybių medį, o po to atliekant jame paiešką į gylį arba į plotį.

Paieška gilyn

Pradėjus nuo kažkurios viršūnės, panaudojant nuo jos einančius lankus aplankomos ir visos kitos briaunomis pasiekiamos viršūnės. Naudojamos dvi skirtingos viršūnių aplankymo strategijos: paieška gilyn ir platyn. Jos dažnai naudojamos kaip sudėtingesnių algoritmų dalys.

Atliekant paiešką gilyn pradedama nuo vienos viršūnės (tarkime A), tada pereinama prie jos kaimynės (tarkime B). Iš B – prie pastarosios kaimynės (tarkime C). Keliaujama tol, kol pasiekiama viršūnė, kuri nebeturi neaplankytų kaimynių (tarkime L). Tuomet grįžtame vienu žingsniu atgal ir nagrinėjame, ar viršūnė, iš kurios atėjome (tarkime K) dar turi nors vieną neaplankytą kaimynę (tarkime L2). Jei turi, – toliau tęsiama paieška gilyn nuo L2, jei ne – grįžtama dar per vieną žingsnį (mūsų atveju būtų iki viršūnės J). Gana dažnai algoritmas realizuojamas naudojant rekursiją.

Detaliau apie paiešką gilyn pasakojama „YouTube“ įraše:

https://www.youtube.com/watch?v=PMMc4VsIacU

Paieška platyn

Pradėjus nuo pasirinktos viršūnės (tarkime A), aplankomos **visos** viršūnės, kurios tik yra pasiekiamos iš jos einant viena briauna (vienu ėjimu). Tarkime B, J ir Z. Tuomet – pasiekiamos iš pastarųjų (arba dviem ėjimais iš pradinės) ir t. t. Jau aplankytoms viršūnėms žymėti sudaroma viršūnių eilė. Literatūroje kartais minima, kad ši viršūnių aplankymo strategija garantuoja, kad kiekviena viršūnė bus aplankyta trumpiausiu keliu, tačiau atkreipkite dėmesį, kad čia turima omenyje tik besvoriai lankai, o kelio ilgis tarp bet kurių viršūnių matuojamas lankais, o ne jų svorių suma.

Detaliau apie paiešką platyn pasakojama „YouTube“ įraše:

https://www.youtube.com/watch?v=xlVX7dXLS64

Pagrindinė empirinio paieškos algoritmo problema – neproporcingai didelės laiko sąnaudos optimalaus sprendimo paieškai. Pavyzdžiui, jei turime grafą su dešimčia viršūnių ir optimalaus sprendimo paieškai pasitelktume kompiuterį, galintį patikrinti milijoną Hamiltono ciklų, užtruktume kiek mažiau nei sekundę. Tačiau jei grafas turi 20 viršūnių, tas pats kompiuteris sprendimo ieškotų jau daugiau nei 1000 metų. Taip yra todėl, kad galimų Hamiltono ciklų skaičius didėja eksponentiškai pagal formulę:

$$Tinkamų\ ciklų\ skaičius = \frac{(N-1)!}{2}$$

Čia tinkama vieta padiskutuoti su moksleiviais, kuo skiriasi vieno ar kito **algoritmo efektyvumas**, kaip jis matuojamas.

Algoritmų teorijoje uždaviniai yra skiriami į tam tikras klases: NL, P, NP-*complete* (beje, keliaujančio pirklio uždavinys ir patenka į šią klasę) ir kt. Tačiau moksleiviams rekomenduojama algoritmų efektyvumą paaiškinti paprasčiau. Vertinant algoritmą reikėtų atsižvelgti į du jo parametrus: grafo, su kuriuo dirbama dydį N ir žingsnių, reikalingų algoritmui įvykdyti, skaičių V. Jei žingsnių skaičius V didėja neproporcingai daug didinant N, tuomet algoritmas laikomas neefektyviu. Jei V nuo N visai nepriklauso arba didėja nežymiai (pvz., tarp šių dydžių yra tiesinė priklausomybė), sakome, kad **algoritmas yra efektyvus**.

3-oji užduotis

Apskaičiuokite, kiek žingsnių atlieka jūsų pasirinktas algoritmas. Kaip žingsnių skaičius priklauso nuo pradinių parametrų (pvz., viršūnių skaičiaus)? Pabandykite išvesti (užrašyti) tos priklausomybės formulę. Kaip manote, ar algoritmas yra efektyvus, ar ne? Atsakymą pagrįskite.

4-oji užduotis

Palyginkite dviejų algoritmų, kurių V ir N priklausomybės jau žinomos (yra duotos formulės) efektyvumą. Kuris iš jų efektyvesnis? Ar bent vieną iš jų galima laikyti efektyviu?

5-oji užduotis

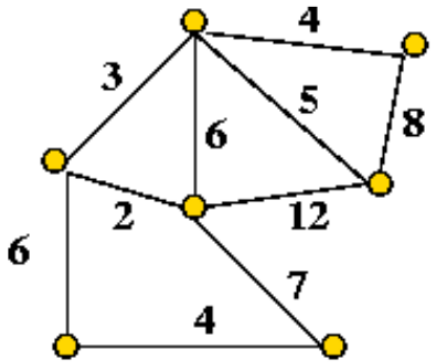
Jei nustatėte, pagal kokią funkciją tarpusavyje priklauso V ir N, užrašykite šią funkciją naudodami „MS Word“ lygtis.

Kaip jau minėta, optimalaus ciklo paieška užtrunka pernelyg daug laiko, todėl praktikoje naudojami įvairūs ***apytikriai*** algoritmai.

Štai keletas apytikrių algoritmų:

- Artimiausio kaimyno algoritmas;

- Pigiausios grandies algoritmas;
- Klarko-Raito algoritmas.



8 pav. Grafas keliaujančio pirklio uždaviniui spręsti

https://www2.seas.gwu.edu/~simhaweb/champalg/tsp/figures/tsp2.gif

Artimiausio kaimyno algoritmas

Laikomas pačiu paprasčiausiu algoritmu keliaujančio prikilo uždaviniui spręsti:

1. Imamos dvi aibės: U ir V. Aibėje U yra tik pirmoji viršūnė, o visos kitos – aibėje V.
2. Imama paskutinė į aibę U įdėta viršūnė *u* ir ieškoma trumpiausios briaunos nuo jos iki bet kuriuos viršūnės *v*, esančios aibėje V. Radus tokią ji iš aibės V perkeliama į aibę U.
3. Procesas kartojamos tol, kol aibėje V nebelieka viršūnių. Atrinktos briaunos ir sudaro, kaip manoma, trumpiausią kelią.

Aibės sąvokos čia padėtų kuriant kompiuterinę programą uždavinio sprendimui, o patį kelio paieškos algoritmą moksleiviams reikėtų aiškinti supaprastintai:

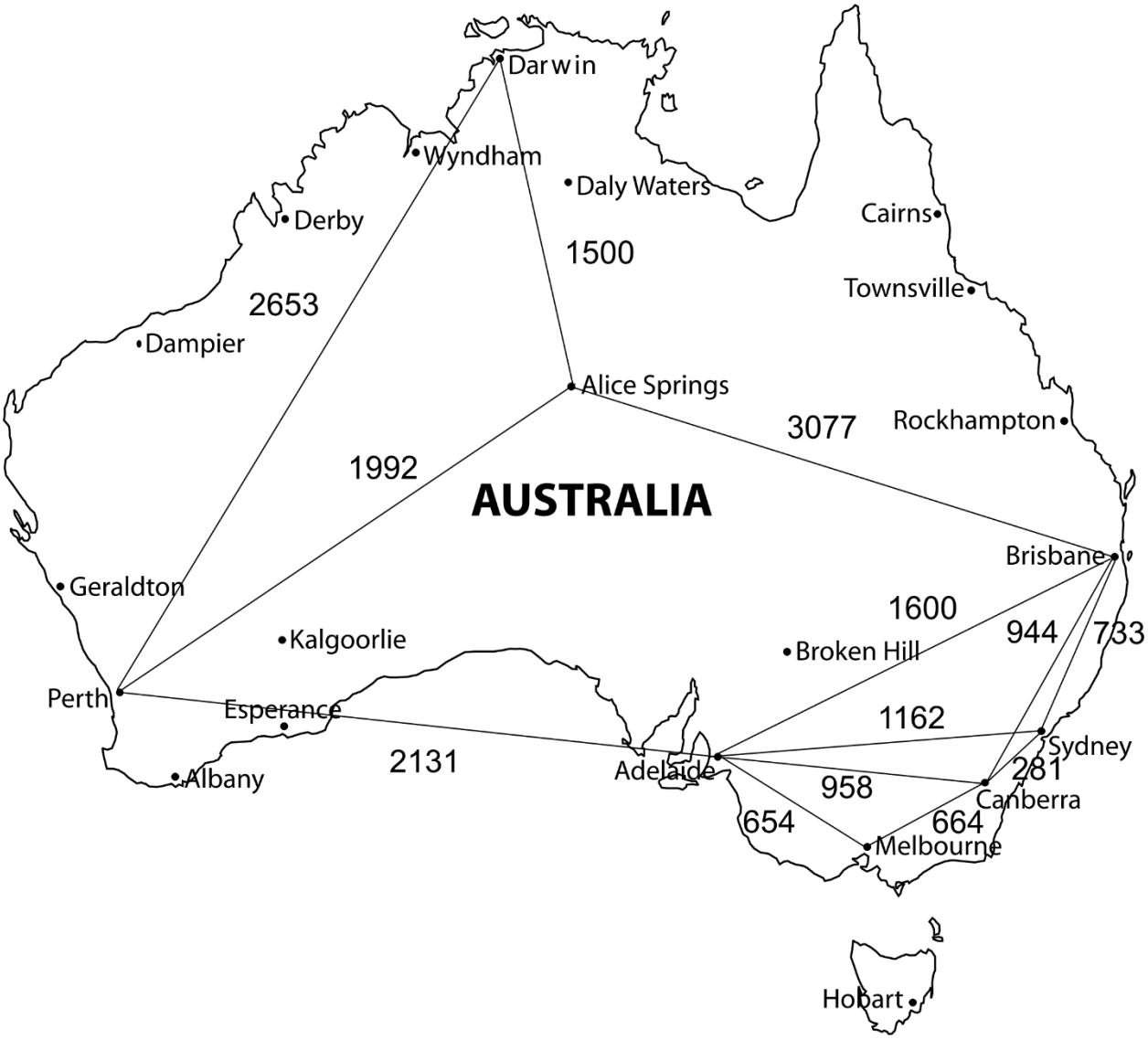
1. Įsivaizduokite, kad esate priklys ir pradedate kelionę iš namų taško.
2. Kitu miestu rinkitės tą, iki kurio yra trumpiausias atstumas (mažiausia kaina ar pan.) Jei tokių yra keli, galima rinktis bet kurį iš jų.
3. 2-ą žingsnį kartokite tol, kol aplankysite visus miestus. Iš paskutinio aplankyto miesto grįžkite namo.

Kartais jau pačiame grafe matyti, kad yra optimalesnis uždavinio sprendimas, jei pradžios (o kartu ir pabaigos) tašku pasirinksime ne namus, o kitą tašką. Nuo jo pradedant taip pat galima

konstruoti ciklą, tik po to jį reikia perrašyti taip, tarsi jis prasidėtų namų viršūnėje. Tai atlikti nebus sunku – ciklas bet kokiu atveju eis ir pro namų viršūnę, nes aplanko jas visas. Modifikuotas algoritmas, kai pradžios taškas pasirenkamas kitas nei namų ar net atsitiktinis, vadinamas *Kartotiniu artimiausio kaimyno algoritmu*.

2-a veikla

Atidžiai išnagrinėkite artimiausio kaimyno algoritmą ir atlikite jį su šiuo grafu:



Pigiausios grandies algoritmas

Pristatant šį algoritmą reikia atkreipti dėmesį, kad kartotiniame artimiausio kaimyno algoritme nebuvo reikalaujama maršrutą konstruoti nuo namų viršūnės. O jeigu taip, galbūt galima tą maršrutą surankioti atskirai po grandį? Čia susipažįstame su būtent tai atliekančiu *Pigiausios grandies algoritmu*:

1. Raskite visame grafe patį trumpiausią atstumą (mažiausią kainą ar pan.) ir šią briauną pasirinkite (kokiu nors būdu pažymėkite).
2. Imkite antrą pagal trumpumą briauną, patikrinkite, ar niekas netrukdo, ir kartokite šį žingsnį su kitomis briaunomis, kol susidarys Hamiltono ciklas.

Pamąstykite ir aptarkite

1. Kas gali trukdyti? Kokias atvejais negalima rinktis briaunos?

- Jei pasirinkta briauna uždarytų mažesnį ciklą (dar liktų neapeitų viršūnių);
- Jei tai būtų jau trečioji pažymėta briauna, išeinanti iš viršūnės. Siekiame, kad prie kiekvienos viršūnės galiausiai būtų pažymėtos lygiai dvi briaunos.

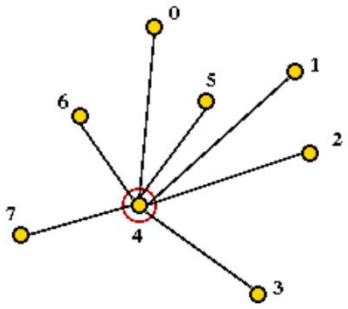
2. Ką daryti, jei 1-u ar 2-u žingsniu randame dvi vienodo ilgio briaunas? Pasirenkame bet kurią iš jų!

Klarko-Raito algoritmas

Abu minėti keliaujančio pirklio uždavinio sprendimo algoritmai priskiriami godžiųjų algoritmų grupei (angl. *greedy algorithms*). Tokiuose algoritmuose vadovaujamasi principu „Čiupk, kas šiuo metu yra geriausio!". Kartais tai veikia, o kartais ne. Praktikoje yra algoritmų, kuriuose remiamasi bent jau apytiksliais paskaičiavimais, objektyviu situacijos vertinimu. Kaip pavyzdį imkime mažiau literatūroje minima *Klarko-Raito algoritmą*.

Šį algoritmą G. Clarke'as ir J. W. Wrightas pasiūlė dar 1964 m. Jis sudarytas iš tokių žingsnių:

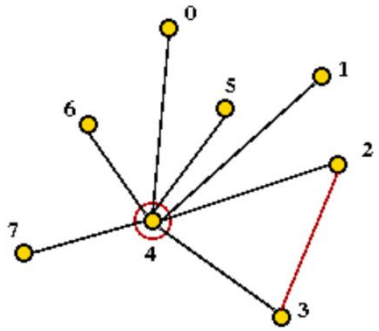
- Viena viršūnė paskelbiama centrine ir visos kitos (vadinamos paprastomis arba dukterinėmis) sujungiamos su ja briaunomis (9 pav.);



9 pav. Klarko-Raito algoritmo 1-asis žingsnis

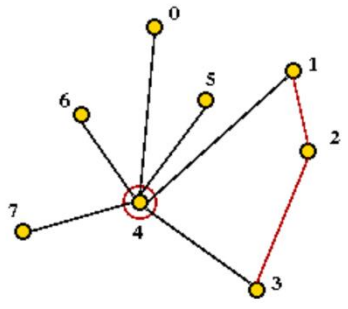
Šaltinis: https://www2.seas.gwu.edu/~simhaweb/champalg/tsp/figures/cw4.png

- Kiekvienai dukterinių viršūnių porai skaičiuojami kelionės iš vienos į kitą per centrinę viršūnę svoriai (kaina). Tada šis skaičius lyginamas su tiesioginės briaunos tarp dukterinių viršūnių svoriu (kaina). Jei tiesioginė briauna yra mažesnė – ji ir pasirenkama (10 pav.). Pradedant nuo antros (sutrumpinančios) briaunos iš grafo šalinama briauna, jungianti centrinę ir nagrinėjamąją dukterinę viršūnę (kadangi ši briauna nebebus naudojama, jau rastas trumpesnis kelias), žr. 11 pav.



10 pav. Klarko-Raito algoritmo 1-asis žingsnis

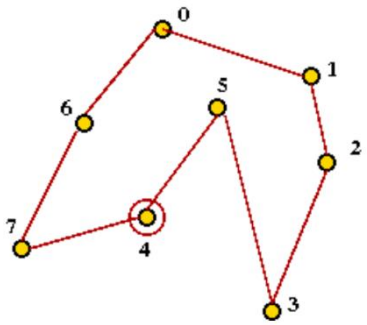
Šaltinis: https://www2.seas.gwu.edu/~simhaweb/champalg/tsp/figures/cw5.png



11 pav. Klarko-Raito algoritmo 2-asis žingsnis

Šaltinis: https://www2.seas.gwu.edu/~simhaweb/champalg/tsp/figures/cw6.png

- Procesas kartojamas tol, kol sujungiamos visos viršūnės, centrinė viršūnė tampa paprastąja (12 pav.).



12 pav. Klarko-Raito algoritmo paskutinis žingsnis

Šaltinis: https://www2.seas.gwu.edu/~simhaweb/champalg/tsp/figures/cw8.png

Algoritmas ne tik padeda surasti Hamiltono ciklą, bet gerai parodo alternatyvą – juk galėjome iki kiekvieno miesto keliauti tiesiog iš centrinės viršūnės ir grįžti. Labai greitai moksleiviai pamatys, kad toks pasirinkimas daugeliu atvejų yra neefektyvus ir taip bus galima pagrįsti, kodėl mes formuluojame keliaujančio pirklio uždavinį.

3-oji veikla

Imkime kokią nors konkrečią ar hipotetinę krovinių pervežimo bendrovę, nuspręskime, kuriuose Europos miestuose ji vykdo veiklą (kuriuos turi aplankyti). Tarkime, kad bendrovės centrinė bazė yra Vilniuje, jos vilkikas po vieną kartą turi aplankyti kitus miestus ir galiausiai vėl grįžti į Vilnių. Atstumus tarp miestų galime nustatyti naudodami šį įrankį:

https://www.distancecalculator.net

Pabandykime išspręsti šį **keliaujančio pirklio** uždavinį dviem atvejais:

- Be apribojimų;
- Su apribojimais, kylančiais iš Europos sąjungos mobilumo paketo reglamento (EB) Nr. 561/2006 (https://ltsa.lrv.lt/lt/veiklos-sritys/keliu-transportas-5/mobilumo-paketas) dalies, reikalaujančiais po tam tikro laiko grįžti į savo gyvenamąjį miestą.

Kuriuo atveju bendra kelionės trukmė yra trumpesnė? Kaip keičiasi nagrinėtieji algoritmai antruoju atveju? Ar pigiausios grandies algoritmas antruoju atveju gali būti naudojamas?

6-oji užduotis

Šią užduotį reikia atlikti dirbant 4–5 asmenų grupėje. Nubraižykite grafą, kurio viršūnės esate jūs, o kiekviena briauna – galimas kelias vienam pas kitą. Įvertinkite briaunų svorį, pvz., kilometrais. Suraskite optimalią namų seką, ką po ko reikia aplankyti, jei norima aplankyti visus draugus nukeliaujant mažiausią galimą atstumą.

Grafų uždavinių sprendimas kompiuteriu

Kaip minėta, keliaujančio pirklio ar kitą uždavinį patogu spręsti kompiuteriu (ypač jei viršūnių skaičius yra didesnis). Paprastai tam kuriama programa ta programavimo kalba, kurios mokosi moksleiviai. Tačiau yra dar ir kita galimybė – panaudoti „MS Excel“ skaičiuoklės funkciją *Solver*. Jos naudojimo pavyzdį, skirtą maršrutui po Malaizijos salas, galite rasti čia: https://www.theoptimizationexpert.com/case-studies/tsp/

Daugiau grafų algoritmų ir jų programavimo pavyzdžių galima rasti ir „YouTube“ įrašuose:

https://www.youtube.com/watch?v=tWVWeAqZ0WU&ab_channel=freeCodeCamp.org

Šaltiniai

Matuliauskas, K. Grafų teorija. 2010. https://kestutis.matuliauskas.lt/K.Matuliauskas_Grafu-teorija_v1.2.pdf

Tannenbaumas, P., Arnoldas, R. Kelionės į šiuolaikinę matematiką. Vilnius: TEV, 1995, 512 p.

Tannenbaum, P. Excursions in Modern Mathematics. Pearson, 2018, 596p.

Petrauskas, L., Skūpienė, J. Informatikos olimpiados: algoritmai ir taikymo pavyzdžiai. Vilnius, 2006. https://siom.lmio.lt/m/t/knyga.pdf