Algoritmų analizė

Laboratorinis darbas

Kinų laiškanešio uždavinys

Vykintas Kumštys, MI, MIF, VU

Uždavinio formuluotė

Duota: Neorientuotas svorinis grafas. Taip pat žinoma, kad ne daugiau kaip 2 viršūnės yra nelyginio laipsnio.
Rasti: Optimalų kinų laiškanešio maršrutą. (Oilerio ciklą)

Pagrindinių algoritmų aprašas

Dijkstra.

Algoritmui vykdyti gaunamas grafo objektas, sudarytas iš viršūnių ir kraštinių masyvų, pradžios viršūnė bei pabaigos viršūnė.

- Inicializuojame kiekvienos viršūnės atstumą lygų begalybei (atstumas nuo pradžios) išskyrus startinės, kurios atstumas bus lygus 0. Taip pat kiekvienai viršūnei masyve įrašom kaimynines viršūnes su atstumais (iš kraštinių masyvo)
- Sukame ciklą, kol nebeliks neišnagrinėtų viršūnių: imame viršūnę su mažiausiu atstumu. Jei tai nėra finišas, einame per visas jos kaimynes ir žiūrime, ar atstumas, kurį turi mūsų paimta viršūnė plius atstumas iki kaimynės, yra mažesnis nei atstumas, kurį turi kaimynė. Jei taip, priskiriame naują atstumą kaimynei. Į mapą (specialus konteineris su key, value) įrašome tos kaimynės tėvinę viršūnę (viršūnę, iš kurios atėjome).
- Prasukę pilnai ciklą gražiname mapą, kur matome kiekvienos viršūnės tėvą ir taip galime rasti trumpiausią atstumą tarp norimų viršūnių.
- Sudėtingumas O(n^2)

Fleury.

Algoritmui vykdyti gaunamos grafo viršūnės, kraštinės bei pradinis taškas.

- Algoritmas vykdomas tol, kol yra kraštinių
- Imame pradinę viršūnę, pasirenkame jos vieną iš kraštinių.
    - Jei ji nėra tiltas ir turi kaimynių, ją pridedame prie kelio masyvo, šaliname ją iš kraštinių masyvo. Nustatome pradinę viršūnė į pašalintos kraštinės galutinę viršūnę.
    - Jei ji teturi vieną kraštinę, tai ją ir renkamės (nesvarbu, kad tai tiltas)
- Sudėtingumas O(E^2)

Pilnas algoritmas:

Nuskaitome grafą iš failo (kraštines bei viršūnės įrašom į atitinkamus masyvus). Surandame viršūnes su nelyginiais laipsniais.

- Jei jų yra daugiau nei 2, algoritmo nebevykdome(neatitinka sąlygos).

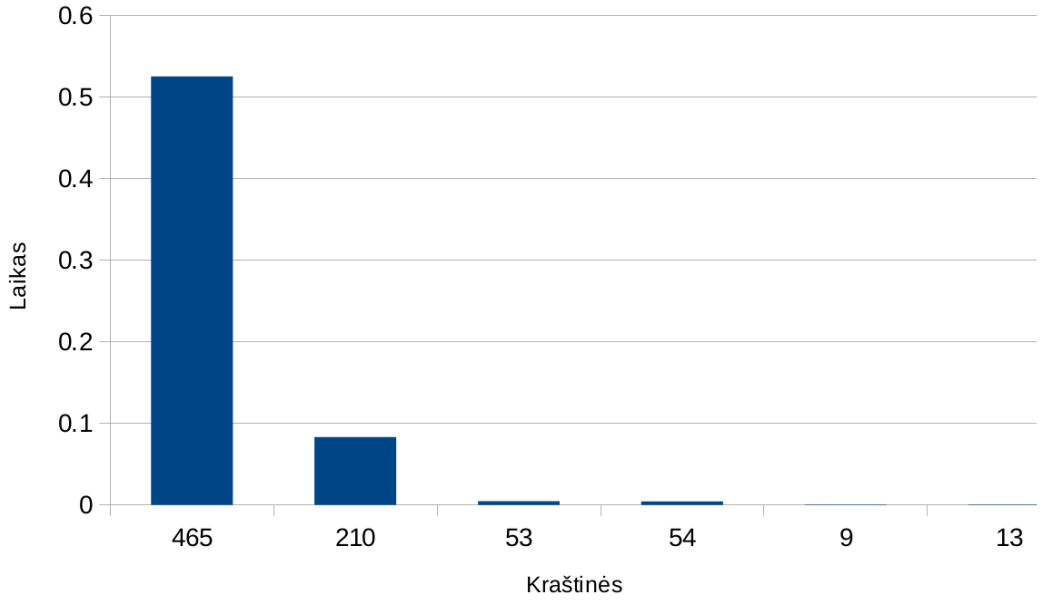
- Jei jų yra lygiai dvi, taikome Dijkstrą toms viršūnėms. Randame trumpiausią kelią tarp jų. Papildome grafą: dubliuojame kraštines, kurios įeina į rastą kelią (įrašome kraštines į kraštinių masyvą). Taip mūsų dvi viršūnės tampa lyginio laipsnio. Grafas nebeturi nelyginio laipsnio viršūnių.

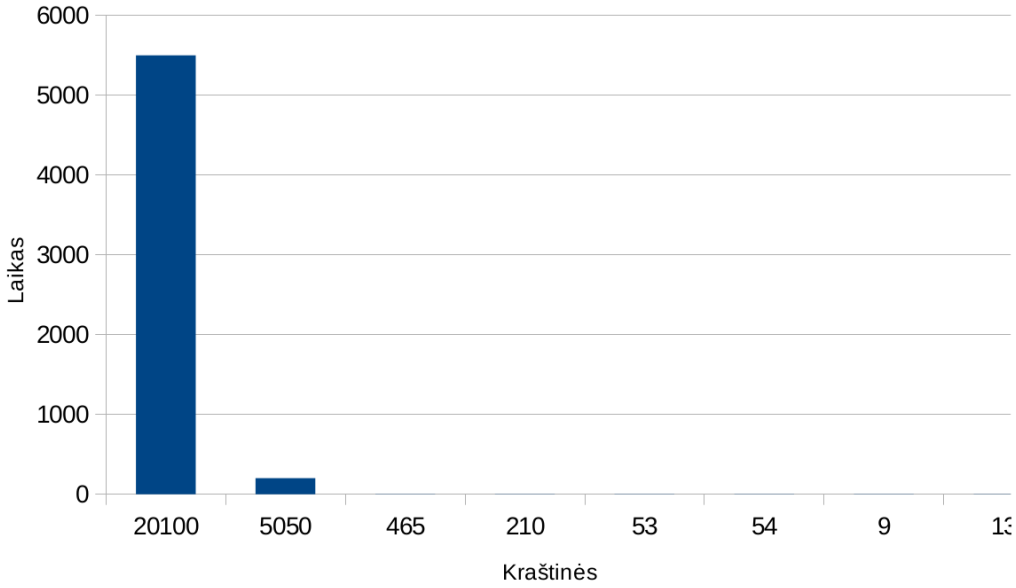
Taikome Fleury algoritmą ir randame Oilerio ciklą (norimą optimalų laiškanešio maršrutą).

Gauti rezultatai

Viršūnės	Kraštinės	Laikas(s)
201	20100	5989.254809
101	5050	199.500524
31	465	0.525356
21	210	0.083264
11	**54**	**0.00431**
11	**53**	**0.004477**
8	**13**	**0.000435**
5	**9**	**0.000403**

Lentelėje paryškinti grafai buvo su 2 nelyginio laipsnio viršūnėm (buvo taikytas Dijkstros algoritmas).





Iš grafikų matome, kad algoritmui yra labai svarbus grafo dydis. Tiek jo viršūnių, tiek kraštinių skaičius. Tiesa, būtent kraštinių skaičiaus didinimas ilgins algoritmo vykdymo laiką bene kvadratiškai. “Dėka” jų bus padidėjęs iteracijų skaičius tiek Dijkstros, tiek Fleury algoritmuose. Pastarajame visada reikės nustatinėti ar galime pašalinti kraštines, kas reiškia, kad vis turime perbėgti per visas (likusias) grafo kraštines Kaip galime matyti iš grafikų, esminis parametras šiam algoritmui vykdyti yra kraštinių skaičius.

Programos paleidimas

Programa rašyta su Python. Paleidžiama iš tekstinės eilutės python algo.py, data.txt faile nurodomas grafas tokia tvarka: iš pradžių surašomos visos viršūnės. Po to kiekvienoje eilutėje rašomos kraštinės: pradinė viršūnė, galutinė viršūnė bei svoris.

Naudota literatūra

- J.A. Bondyand U.S.R. Murty,Graph Theory with Applications, 5th edition,North-Holland,Amsterdam,1982
- http://www.it.brighton.ac.uk/staff/jt40/mm322/MM322_FleurysAlgorithm.pdf
- http://www.suffolkmaths.co.uk/pages/Maths%20Projects/Projects/Topology%20and%20Graph%20Theory/Chinese%20Postman%20Problem.pdf