Vilniaus Universitetas
Informatikos institutas

Vasarė Pratuzaite ir Akvilė Ropytė
informatikos specialybė,
II kursas, 1, 3 grupės

Skirtumo paieška tarp ilgiausio ir trumpiausio kelio grafe, gauto, naudojant Floyd-Warshall algoritmą

Vilnius – 2024

Turinys

Įvadas

Saugumas ir greitis yra vienas iš svarbiausių algoritmų savybių, todėl šios ypatybės padaro algoritmą optimalų.

Prieš daugiau nei keturius dešimtmečius, kai amerikiečių kompiuterių mokslininkas Stephen‘as Warshall’as (1959) ir Robert‘as Floyd’as(1962) paskelbė savo plačiai naudojamus algoritmus, skirtus aptikti bet kokias kilpas arba apskaičiuoti visas poras trumpiausių kelių orientuotame, svoriniame grafe, įvyko reikšmingas proveržis grafų teorijoje. Jo indėlis apima Floydo-Warshall algoritmo (nepriklausomai nuo Stepheno Warshallo), kuris efektyviai randa visus trumpiausius kelius tarp grafo viršūnių, sukūrimą ir jo darbą analizuojant.

Pagrindinė algoritmo idėja yra triviali: jei egzistuoja keliai P[i,k] ir P[k,j], tai egzistuoja ir kelias P[i,j]. Tai leidžia atlikti visų porų analizę ir gauti įvairiapusę informaciją apie grafo struktūrą, sistemiškai sekant veikimo (kaimynystės) informacinę sistemą mazgas po mazgo (k) per grafą.

Šiame darbe tiriami grafai, naudojant Floyd-Warshall algoritmą, kuris skirtas apskaičiuoti trumpiausius kelius tarp visų porų viršūnių orientuotuose, svoriu paženklintuose grafuose. Grafų teorija ir algoritmai yra esminiai įrankiai kompiuterių mokslo ir matematikos srityse, turintys platų pritaikymą nuo transporto tinklų optimizavimo iki socialinių tinklų analizės.

Tyrimo tikslas yra analizuoti, pritaikius Floyd-Warshall algoritmą, gautų trumpiausių atstumų skirtumą - skirtumą tarp ilgiausio ir trumpiausio kelio. Taip pat išsiaiškinti, kokią informaciją apie grafo struktūrą galima gauti iš šios analizės. Šiame tyrime bus nagrinėjama, kaip kinta minėtas skirtumas skirtingų struktūrų grafuose.

Iškelta hipotezė - kuo daugiau kaimynių turi grafo viršūnės, tuo skirtumas tarp ilgiausio ir trumpiausio kelio bus mažesnis.

Tyrimui naudojami 2 skirtingi orientuoti svoriniai grafai, turintys po 10 viršūnių, kur kiekviena viršūnė turi 3 kaimynus viename grafe ir 4 kitame. Tyrimą sudaro šie etapai:

1. Grafų generavimas pagal nurodytas struktūrines savybes.
2. Floyd-Warshall algoritmo pritaikymas šiems grafams.
3. Gautų skirtumų tarp ilgiausio ir trumpiausio atstumo analizavimas.

1 Apibrėžimai. Tiriamoji grafų klasė

Apibrėžimas 1 (Orientuotas grafas) *Grafas $G = (V, E)$, kuriame V yra viršūnių aibė ir E yra briaunų aibė, yra orientuotas, jei kiekviena briauna turi priskirtą kryptį.*

Apibrėžimas 2 (Svorinis grafas) *Tai toks grafas $G = (V, E)$, kuriame kiekvienai briaunai $e \in E$ yra priskritas svoris - skaičius, kuris paprastai žymimas $w(e)$. Formaliu būdu, svorinį grafą galima apibrėžti kaip grafą $G = (V, E, w)$, kur $w : E \to \mathbb{R}^+$ yra funkcija, kuri kiekvienai briaunai $e \in E$ priskiria teigiamą realųjį skaičių.*

Apibrėžimas 3 (Grafo uždarinys) *Grafo G uždarinys yra grafas G^*, gautas iš grafo G, vis pridedant kraštines tarp neincidenčių viršūnių, kurių laipsnių suma yra nemažesnė nei grafo G viršūnių skaičius.*

Apibrėžimas 4 (Tranzityvumas) *Tranzityvumas – tai sąryšio R tarp dviejų aibės elementų apibūdinimas. Tarkime, turime aibę X, kurioje yra elementai $\{x, y, z\}$. Jei aibės elementų pora (x, y) tenkina sąryšį $(x\,R\,y)$ ir pora (y, z) tenkina sąryšį $(y\,R\,z)$, tai sąryšis yra tranzityvus, jei $(x\,R\,z)$ galioja ir porai (x, z) (kiekvienam x, y, z iš aibės X).*

Apibrėžimas 5 (Tranzityvus grafas) *Tai toks grafas $G = (V, E)$, kurio viršūnių poroms (a, b) ir (b, c) egzistuoja viršūnių pora (a, c). T.y., jei iš a galime nueiti į b ir iš b nueiti į c, tai tada iš a galime nueiti į c.*

Apibrėžimas 6 (Tranzityvus grafo uždarinys) *Tai duomenų struktūra, kurioje pavaizduoti visi galimiausi keliai tarp kiekvienos viršūnės. Kitaip, iš grafo $G = (V, E)$ gauname tranzityvų grafo uždarinį $G^* = (V, E)$, kur yra briauna (u, v), priklausanti aibei E^*, jei grafe G yra kelias iš u į v.*

Apibrėžimas 7 (Trumpiausias kelias tarp dviejų viršūnių) *Trumpiausias kelias tarp dviejų viršūnių u ir v svoriniame orientuotame grafe $G = (V, E)$ yra tokia viršūnių seka, kad briaunų svoriai būtų mažiausi tarp visų galimų kelių iš viršūnės u į viršūnę v.*

Apibrėžimas 8 (Floyd-Warshall algoritmas) *Šis algoritmas naudojamas rasti trumpiausius kelius tarp visų porų viršūnių svoriniuose grafuose. Šis algoritmas tinka tiek orientuotiems, tiek neorientuotiems grafams ir gali tvarkyti neigiamus briaunų svorius, jei nėra neigiamų ciklų.*

2 Algoritmai

Algorithm 1 Floyd-Warshall algoritmas

Duota: Orientuotas svorinis grafas $G = (V, E)$
Rasti: Trumpiausius kelius tarp kiekvienos viršūnės
floyd_warshall (G) (iš Networkx bibliotekos)

Dabar žinomas algoritmas buvo paskelbtas Roberto Floido (angl. Robert Floyd) 1962 m. Tačiau šis algoritmas buvo labai panašus į anksčiau išleistą Stefano Varšalo (angl. Stephen Warshall) algoritmą tranzityvaus grafo uždariniui rasti. Floyd-Warshall algoritmas skirtas rasti trumpiausius kelius tarp kiekvienos viršūnės orientuotame svoriniame grafe. Algoritmas sukonstruoja naują svorinį grafą iš duoto svorinio grafo, kuriame kiekvienai briaunai pateikiamas svoris, atitinkantis minimaliam svoriui kažkokio kelio tarp tos briaunos viršūnių. Algoritmas tinka ir grafui, kurio briaunų svoriai yra neigiami, tačiau, jei grafe bus neigiamų ciklų, tai algoritmas neveiks korektiškai.

Algorithm 2 Grafo su k kaimynų generavimo algoritmas

Duota: Viršūnių skaičius n, kaimynų skaičius k, minimalus atstumas *min_distance*, maksimalus atstumas *max_distance*
Rasti: Sugeneruotą orientuotą svorinį grafą $G = (V, E)$

```
G ← tuščias grafas
for node from 1 to n do
    G.add_node(node)
end for
for node from 1 to n do
    candidate_nodes ← sąrašas visų viršūnių, išskyrus node, kurios turi mažiau nei k kaimynų
    neighbors ← atsitiktinai pasirinkti min(k, (candidate_nodes)) kaimynai iš candidate_nodes
    for neighbor in neighbors do
        weight ← atsitiktinis svoris intervale nuo min_distance iki max_distance
        G.add_edge(node, neighbor, weight=weight)
    end for
end for
return G
```

Algorithm 3 Trumpiausių kelių skirtumo tarp ilgiausio ir trumpiausio kelių algortimas

Duota: Orientuotas svorinis grafas $G = (V, E)$
Rasti: Trumpiausių kelių skirtumas tarp ilgiausio ir trumpiausio kelio

```
shortest_paths ← floyd_warshall(G)
max_shortest_path ← −∞
min_shortest_path ← +∞
for start_node in shortest_paths do
    for end_node in shortest_paths[start_node] do
        if start_node ≠ end_node then
            shortest_path ← shortest_paths[start_node][end_node]
            if shortest_path > max_shortest_path then
                max_shortest_path ← shortest_path
            end if
            if shortest_path < min_shortest_path then
                min_shortest_path ← shortest_path
            end if
        end if
    end for
end for
return max_shortest_path − min_shortest_path
```

3 Algoritmų realizacija ir tyrimų eiga

Algoritmas įgyvendintas *Python* programavimo kalba, naudojantis NetworkX, Pyplot, Random ir Numpy bibliotekomis. Grafui, sugeneruotam pagal grafo su k kaimynėmis algoritmu, buvo pritaikyta Networkx įgvendinta Floyd-Warshall algoritmo funkcija. Random buvo naudojama atsitiktiniam kelio tarp viršūnių ilgiui ir viršūnės kaimynams parinkti. Numpy naudota rasti masyvo vidurkiui, o Pyplot - atvaizduoti gautiems duomenims.

Algoritmas:

1. Sugeneruojama 10000 grafų ir jiems pritaikomas Floyd-Warshall algoritmas.
2. Ieškomas skirtumas tarp ilgiausio ir trumpiausio kelio.
3. Šis procesas kartojamas 200 kartų.

Pagrindiniai tyrimo objektai:

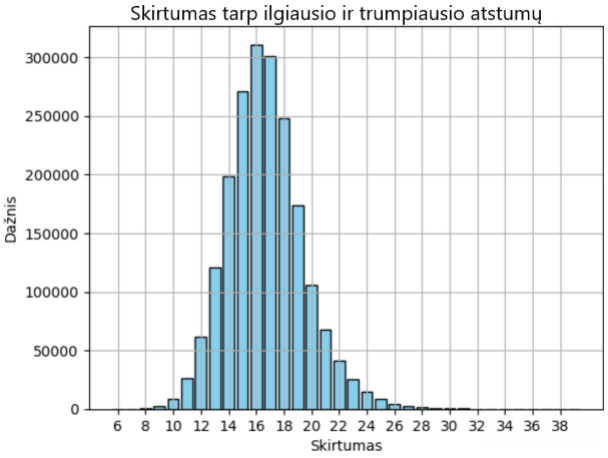
1. Ilgiausio ir trumpiausio kelio skirtumas trumpiausių kelių grafe.
2. Didžiausi skirtumai.
3. Skirtumo vidurkis.

Tyrimai vykdyti pagal skirtingą grafo viršūnių kaimynių skaičių: k = 3 ir k = 4.

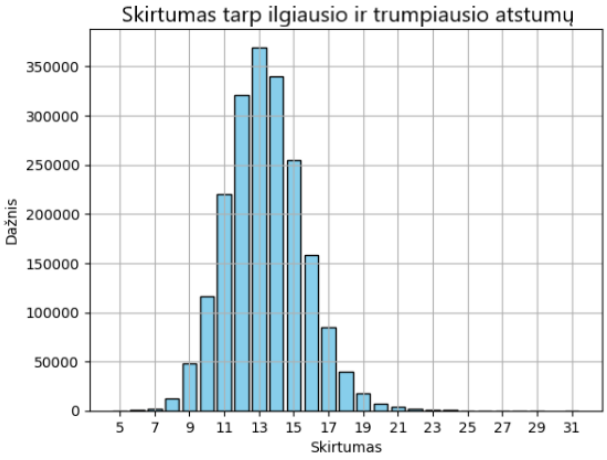
4 Tyrimų rezultatai ir jų aptarimas

Paveikslėliuose 1a ir 1b pavaizduoti skirtumai trumpiausių atstumų grafuose, kai kiekviena viršūnė turi atitinkamai 3 ir 4 kaimynus. Pirmuoju atveju dažniausiai pasikartojantys skirtumai yra 16 ir 17, na o kai kiekvienos viršūnės kaimynių skaičius yra 4, tai dažniausias skirtumas yra 13. Šiek tiek mažiau pasikartoja ir 12 bei 14.

Taigi, dažniausiai pasitaikantis skirtumas, kai viršūnės turi po 4 kaimynus, yra 3 vienetais mažesnis, nei grafe, kurio viršūnės turi po 3 kaimynus.



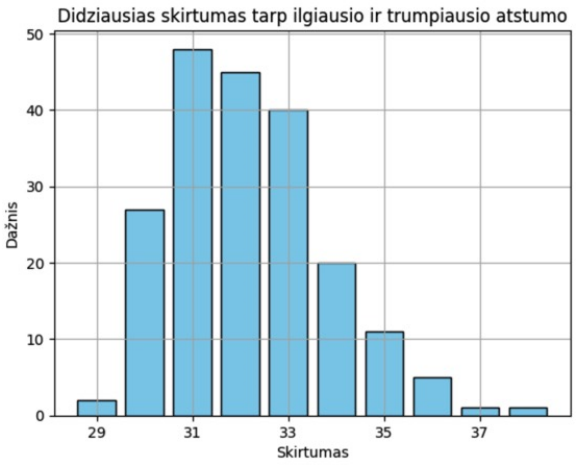
(a) viršūnės kaimynių skaičius = 3



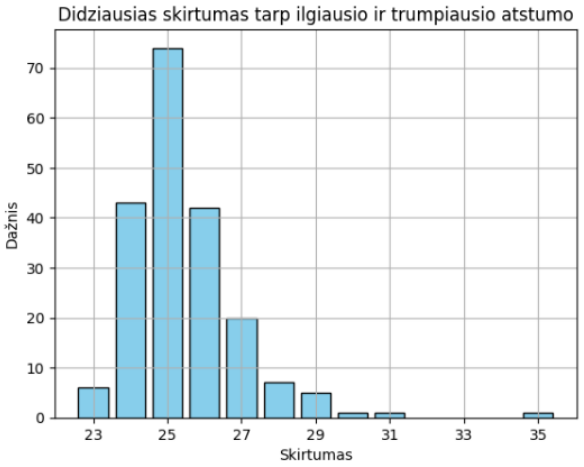
(b) viršūnės kaimynių skaičius = 4

1 pav.

Sekančiuose grafikuose matomi didžiausi skirtumai tarp ilgiausių ir trumpiausių kelių grafuose. Kiekviena viršūnė atitinkamai grafuose turi po 3 ir 4 kaimynes. Paveikslėlyje 2a, kai viršūnių skaičius yra trys, nesunku pastebėti, kad labiausiai pasikartojantys didžiausi skirtumai yra 31, 32, bei 33, o gretimoje diagramoje 2b gerokai mažiau – tik 24, 25 bei 26. Taigi, kai grafo viršūnės turi vienetu daugiau kaimynų, tai dažniausiai pasikartojantis didžiausias skirtumas sumažėja per 6 vienetus.



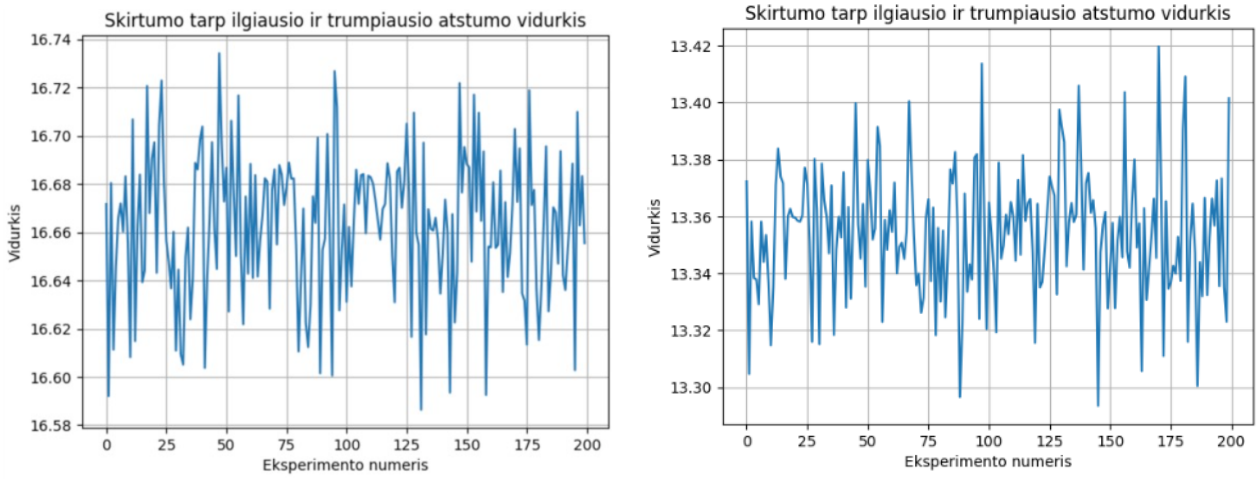
(a) viršūnės kaimynių skaičius = 3



(b) viršūnės kaimynių skaičius = 4

2 pav.

Toliau buvo nagrinėjamas gautų skirtumų vidurkis. Iš gautų rezultatų galima pastebėti, kad jis kinta nežymiai. 3a paveikslėlyje vidurkis svyruoja nuo apytiksliai 16,59 iki 16,74. Apytikslis diapazonas yra 0,15. Na, o 3b grafike vidurkis svyruoja nuo šiek tiek mažiau nei 13,30 iki 13,42. Palyginus, svyravimo diapazonas, kai grafo viršūnės turi po 4 kaimynus, yra mažesnis.

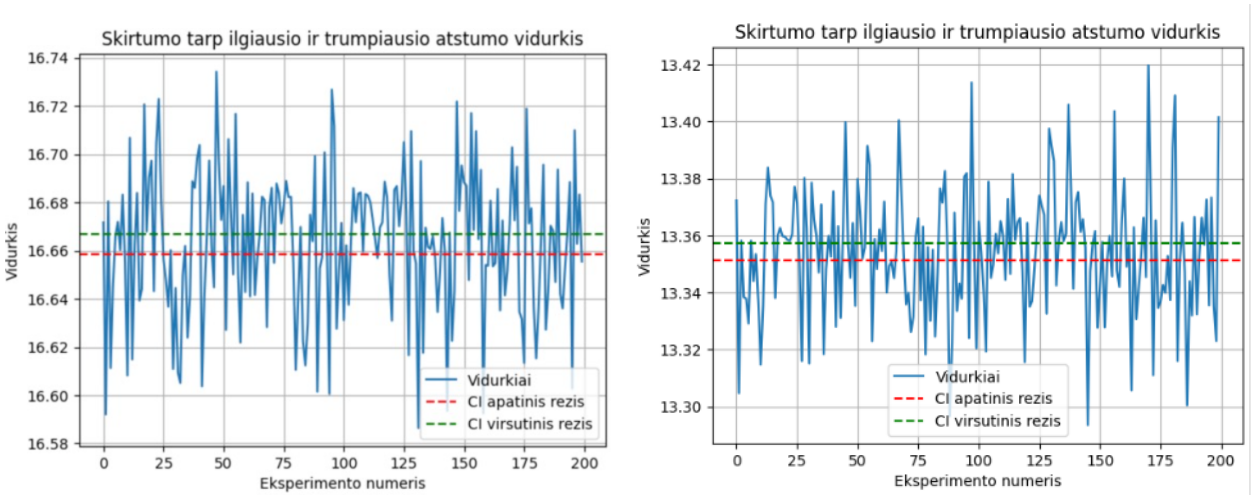


(a) viršūnės kaimynių skaičius = 3

(b) viršūnės kaimynių skaičius = 4

3 pav.

Taip pat dar paskaičiavome gautų skirtumų vidurkių (paveikslėliuose 3a ir 3b) pasikliovimo intervalus. Naudojome bootstrap funkciją. Kai viršūnių kaimynių skaičius yra 3, tai pasikliovimo apatinis rėžis yra 16,6586, o viršutinis 16,6670. Na, o kai kaimynių skaičius yra 4, tai paskliovimo intervalas yra [13.35123, 13.35732].



(a) viršūnės kaimynių skaičius = 3

(b) viršūnės kaimynių skaičius = 4

4 pav.

Išvados

Atlikus gautų rezultatų analizę, galima padaryti daugiausia trivalias išvadas:

- Skaičiuodami skirtumą tarp ilgiausio ir trumpiausio atstumo, dažniausiai pasitaikantis skirtumas, kai viršūnės turi po 4 kaimynus, yra 3 vienetais mažesnis nei grafuose, kur viršūnės turi po 3 kaimynus.
- Analizuodami didžiausius skirtumus tarp ilgiausio ir trumpiausio atstumo pastebime, kad padidėjus viršūnių kaimynų skaičiui vienetu, dažniausiai pasikartojantis didžiausias skirtumas sumažėja 6 vienetais.
- Pastebimas skirtumas tarp vidurkio svyravimo diapazonų yra nedidelis, o tyrimo rezultatai patvirtiną spėjimą, kad didelį kaimynių kiekį turinti viršūnė įtakos mažesnio skirtumo rezultatą tarp ilgiausio ir trumpiausio kelio.

Literatūra

[1] Wikipedia *Floyd-Warshall algorithm* https://en.wikipedia.org/wiki/Floyd-Warshall_algorithm

[2] Programiz.com *Floyd-Warshall algorithm* https://www.programiz.com/dsa/floyd-warshall-algorithm

[3] ICASI 2018 *Floyd-Warshall algorithm in shortest path problem* https://books.google.lt/books?hl=lt&lr=&id=MDBdEAAAQBAJ&oi=fnd&pg=PA47&dq=Search+for+connected+components+in+a+direct+graph+floyd+warshall&ots=I6YerbMVn3&sig=ZqSXvl_7DE2dbCMQBMw1aN4VnL8&redir_esc=y#v=onepage&q&f=false

Priedai. Kodas

Tyrimo kodas Collab aplinkoje:
`https://colab.research.google.com/drive/1nkvBS-fKhPbs9gpjYyCdsVGuO-59Rf4L?usp=sharing`