KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

TAIKOMOSIOS INFORMATIKOS KATEDRA

DISKREČIOSIOS STRUKTŪROS (P170B008)

KURSINIS DARBAS

Užduoties nr. B22

Atliko:

IF-8/1 gr. studentas
Tomas Odinas

Priėmė:

Lekt. Audrius Nečiūnas

KAUNAS

2019

Turinys

1. Turinys

2. Užduotis (nr. 22)

Sudaryti algoritmą ir programą, randančią trumpiausią kelią svoriniame grafe, kurio briaunų svoriai gali būti ir neigiami.

3. Užduoties analizė

Trumpiausio kelio paieška gali būti atliekama Deikstros, A*, Floydo-Warshallo, Bellmano-Fordo algoritmais. Deikstros ir A* algoritmai netinkami grafui kuriame svoriai gali būti neigiami. Pasirinkau Bellmano-Fordo algoritmą dėl jo panašumo į Deikstros algoritmą.

Uždavinys. Duotas orgrafas $G = (V,U)$, kur V – viršūnių, o U – lankų aibės ir pradinė viršūnė. Suskaičiuoti masyvus d ir prec. Taip pat sudaryti trumpiausią maršrutą.

Metodo idėja. Metodas panašus į Deikstros algoritmą. Norint suprasti metodo esmę reikia panagrinėti kodėl Deikstros algoritmas netinkamas grafui su neigiamais svoriais. Pagal Deikstros algoritmo idėją per kiekvieną išorinio ciklo iteraciją nusistovi 1 viršūnė – tai reiškia, kad nebebus rastas trumpesnis kelias iki jos, nes papildomi žingsniai „kainuos“ daugiau. Toks „godus“ algoritmo principas neatsižvelgia į tai jog gali būti atrastas neigiamas lankas kuris sutrumpins atstumą. Bellmano-Fordo algoritmas sprendžia šią problemą atstumus iki viršūnių peržiūrėdamas n-1 kartą (n-viršūnių skaičius). Algoritmo sudėtingumas O(|V|*|U|).

4. Programos algoritmo aprašymas

Programa suskaičiuoja d ir prec masyvus iš kurių sudaromas trumpiausias kelias tarp pasirinktų viršūnių porų.

Pradiniame žingsnyje inicializuojami d ir prec masyvai. d masyvo elementai parenkami begalybės, kad algoritmo metu galėtume atstumus mažinti, o prec 0,nes pagal programos implementaciją viršūnių numeracija prasideda nuo 1.

Toliau n-1 kartą einama per visas viršūnes ir tikrinama ar galima sumažinti atstumus nuo pradinės viršūnės iki einamosios. Jei atstumą sumažinti galima, atnaujinami d ir prec masyvai.

Po n-1 iteracijų d masyve yra saugomi trumpiausi atstumai tarp viršūnių, o prec virtūnės iš kurių buvo ateita į pastarąsias.

Iš d ir prec masyvų atkuriamas trumpiausias kelias tarp pasirinktų viršūnių ir programa baigia darbą.

5. Programos tekstas

Pagrindinė funkcija:

```
function calculatePath(start, end) {
    let structure = getAdjacencyStructure();
    let nodes = structure[0];
    let edges = structure[1];

    //initialization
    let prec = Array(nodes.length).fill(0);
```

```
    let d = Array(nodes.length).fill(Infinity);
    d[start - 1] = 0;
    prec[start - 1] = start;
    //--

    //Bellman-Ford algorythm
    for (let i = 0; i < nodes.length - 1; i++) //i N-1 times
        for (let j = 0; j < nodes.length; j++) //j nodes
            for (let k = 0; k < edges[j].length; k++) //k edges
                if (d[nodes[j][k] - 1] > edges[j][k] + d[j]) {
                    d[nodes[j][k] - 1] = edges[j][k] + d[j];
                    prec[nodes[j][k] - 1] = j + 1;
                } //--

    showResult(d, prec, start, end);
}
```

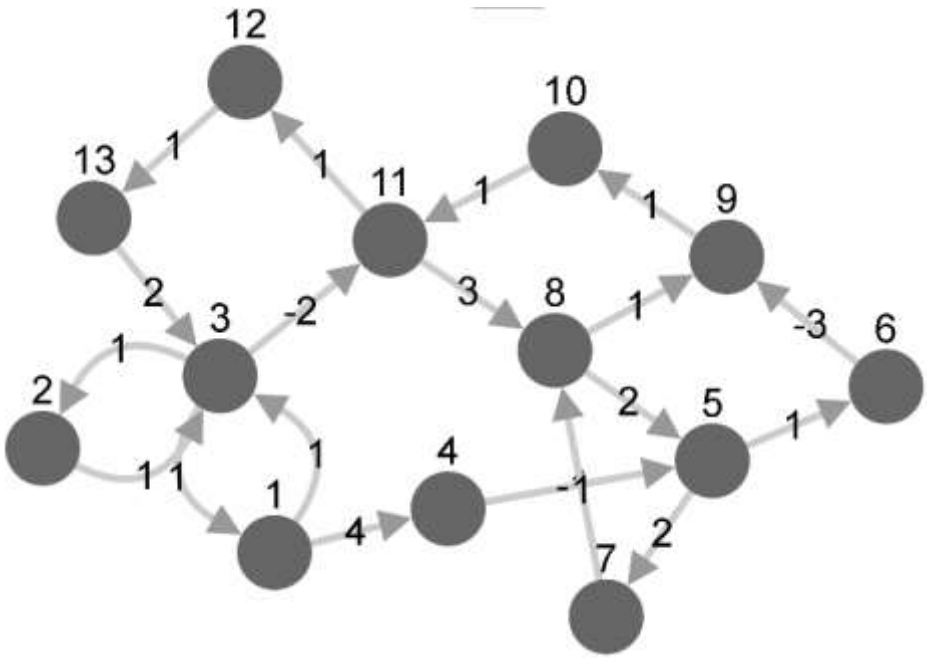
Pilnas programos kodas: https://github.com/d0ubletr0uble/d0ubletr0uble.github.io

Testavimo pavyzdžiai

Buvo panaudoti trys testavimo pavyzdžiai:

Pirmas testas

Duotas orgrafas:



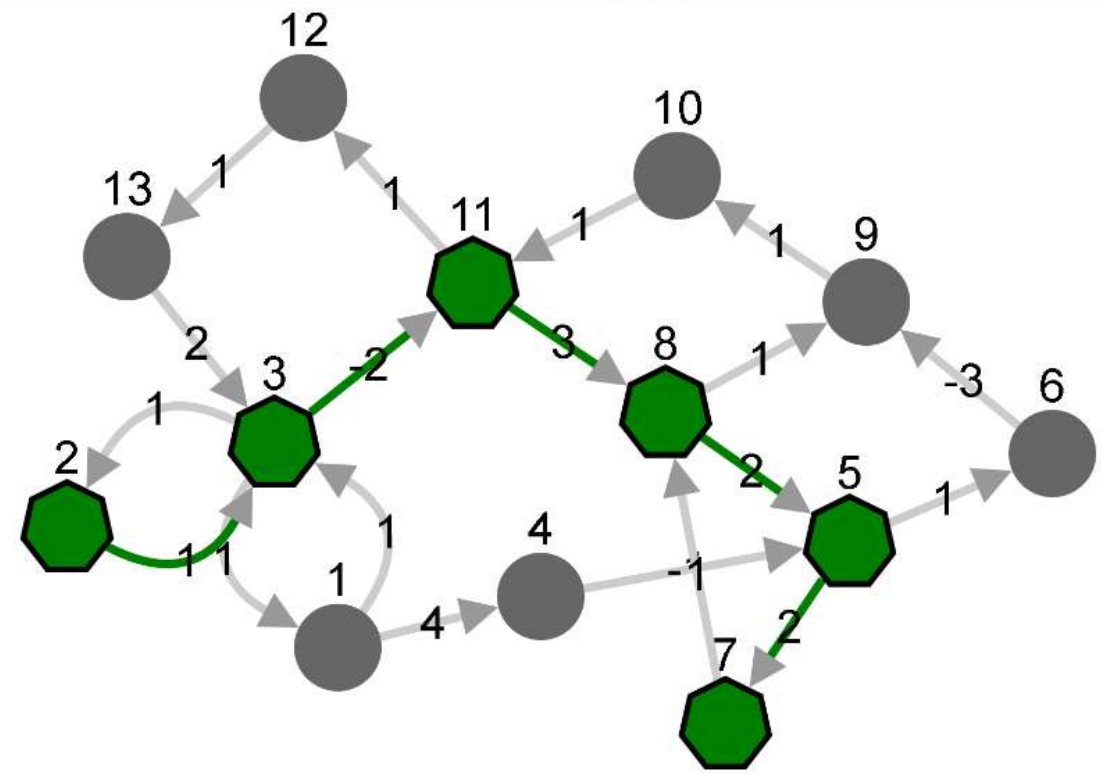
Rasti trumpiausią kelią nuo 2 iki 7 viršūnės.

Rezultatai:

result: 2,3,11,8,5,7

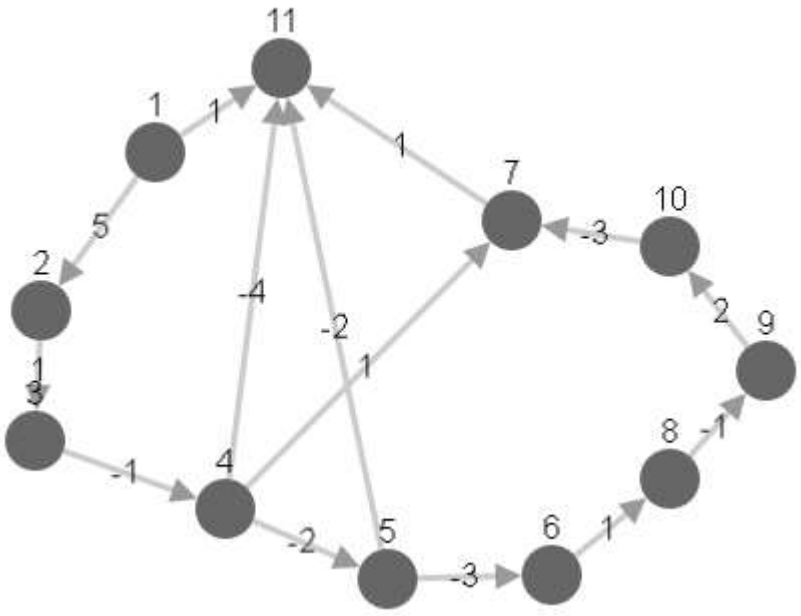
d: 2,0,1,6,4,5,6,2,2,3,-1,0,1

prec: 3,2,2,1,8,5,5,11,6,9,3,11,12



Antras testas

Duotas orgrafas:

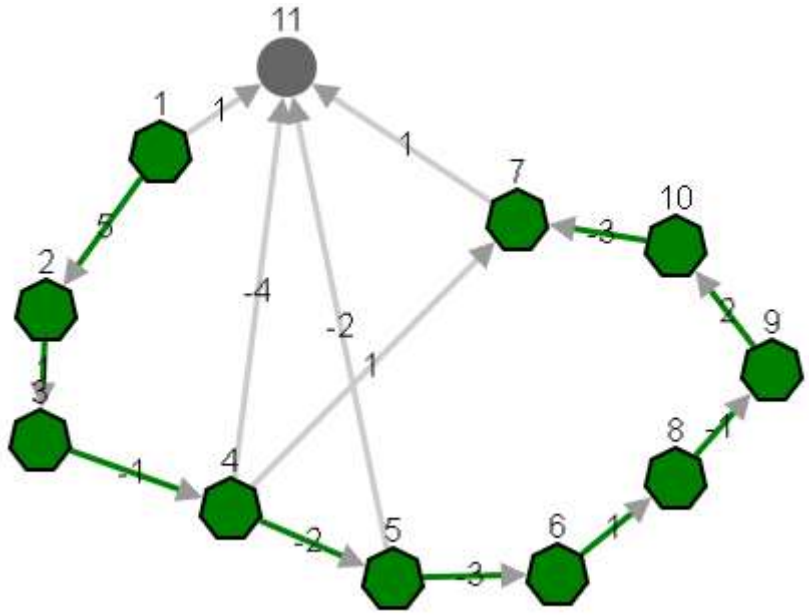


Rasti trumpiausią kelią nuo 1 iki 7 viršūnės.

result: 1,2,3,4,5,6,8,9,10,7

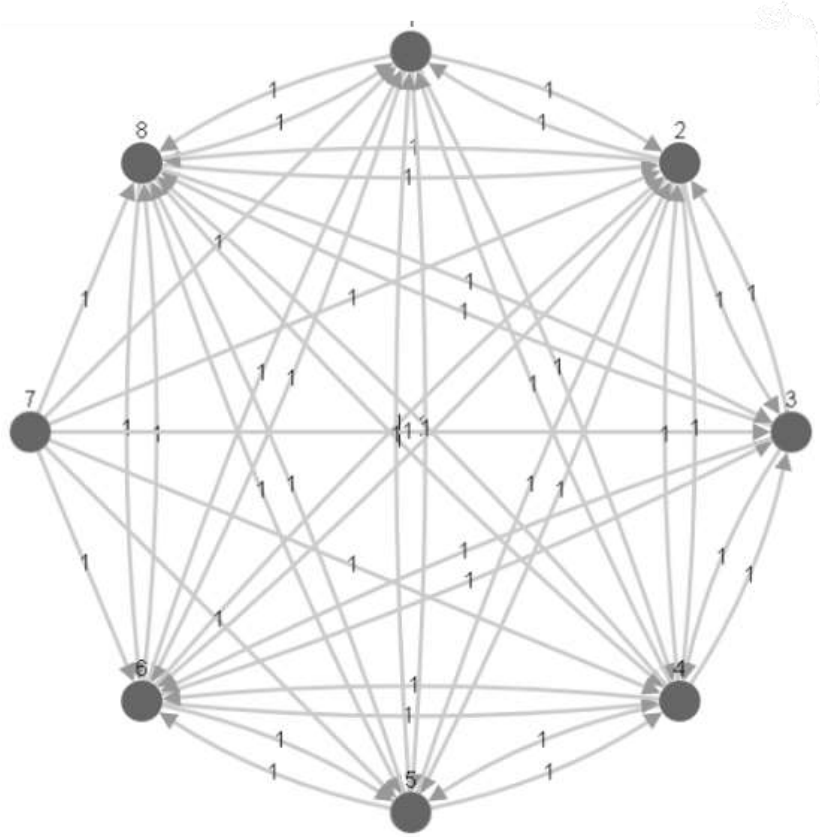
d: 0,5,6,5,3,0,-1,1,0,2,0

prec: 1,1,2,3,4,5,10,6,8,9,7



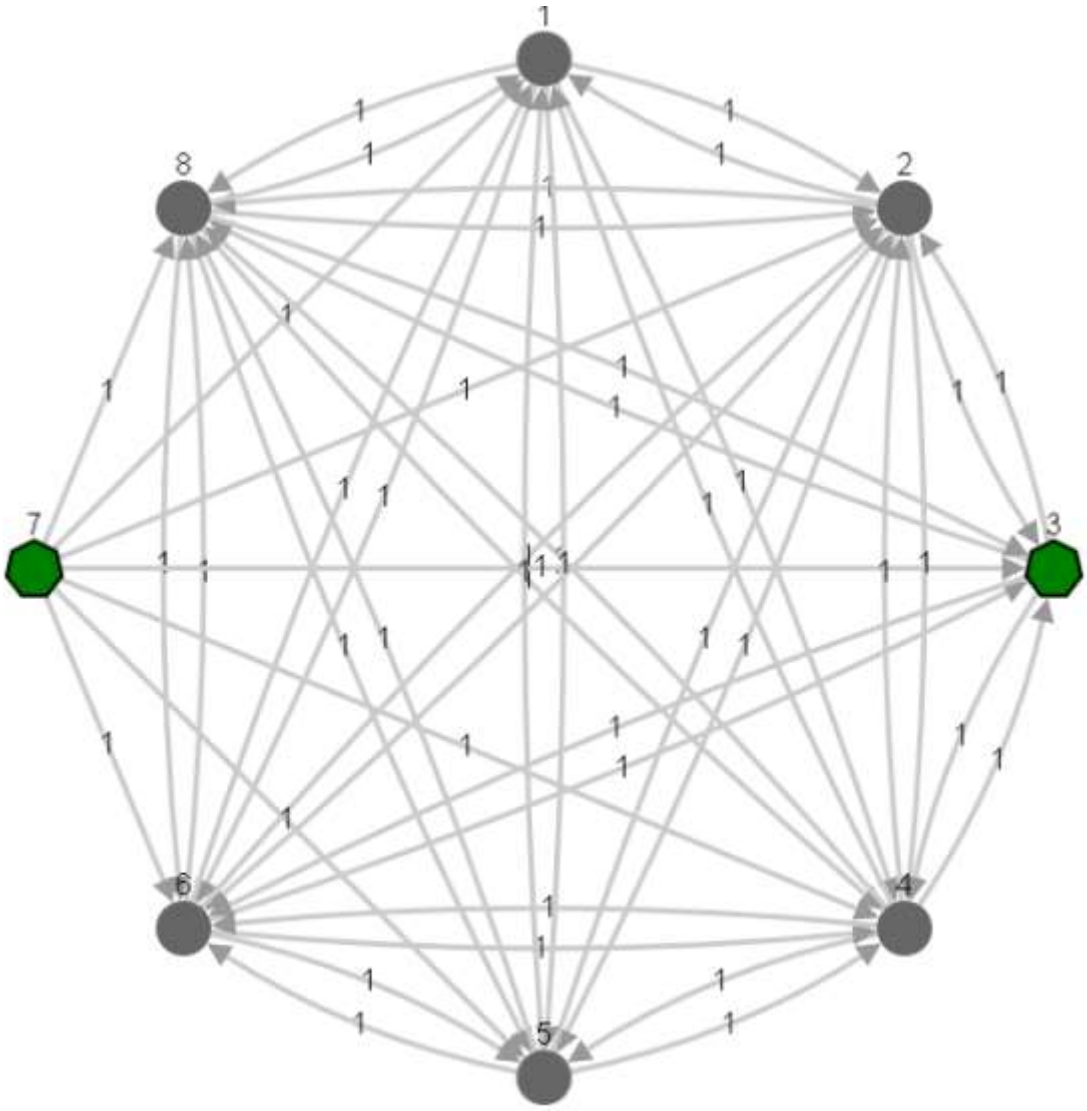
Trečias testas

Duotas orgrafas:



Rasti trumpiausią kelią tarp 3 ir 7

Result: Path doesn't exist :(



6. Išvados

Programa veikia teisingai. Išskirtinis atvejis, kai susidaro neigiamų lankų ciklas – tada vis galima trumpinti nueitą kelią ir programa niekad nesustoja. Bellmano-Fordo algoritmas tokios problemos nesprendžia. Algoritmo sudėtingumas O(|V|*|U|) (čia V-viršūnių sk., U-lankų sk.).

Programa parašyta su JavaScript, todėl ja galima naudotis ir mobiliuosiuose įrenginiuose.

7. Šaltinių sąrašas

1. Bang-Jensen, Jørgen; Gutin, Gregory (2000). "Section 2.3.4: The Bellman-Ford-Moore algorithm". Digraphs: Theory, Algorithms and Applications (First ed.). ISBN 978-1-84800-997-4.

2. Franz M, Lopes CT, Huck G, Dong Y, Sumer O, Bader GD Bioinformatics (2016) 32 (2): 309-311 first published online September 28, 2015 doi:10.1093/bioinformatics/btv557 https://academic.oup.com/bioinformatics/article/32/2/309/1744007