Universitatea Tehnică a Moldovei

Facultatea - Calculatoare Informatică și Microelectronică

Disciplina: *Analiza și proiectarea algoritmilor*

**Raport**

Lucrarea de laborator Nr.4

Tema: Algoritmii Dijkstra și Floyd

A efectuat: st.gr. TI-207 Bunescu Gabriel

A verificat: lect.univ. Bîtca Ernest

Chișinău 2021

**Cuprins**

[I. Scopul lucrării 2](#_Toc88582074)

[II. Sarcina lucrării 2](#_Toc88582075)

[III. Considerații teoretice 3](#_Toc88582076)

[IV. Analiza empirică a algortimilor 4](#_Toc88582077)

[V. Concluzie: 5](#_Toc88582078)

# I. Scopul lucrării

1. Studierea programării dinamice, implementarea algoritmilor Dijkstra și Floydю

# II. Sarcina lucrării

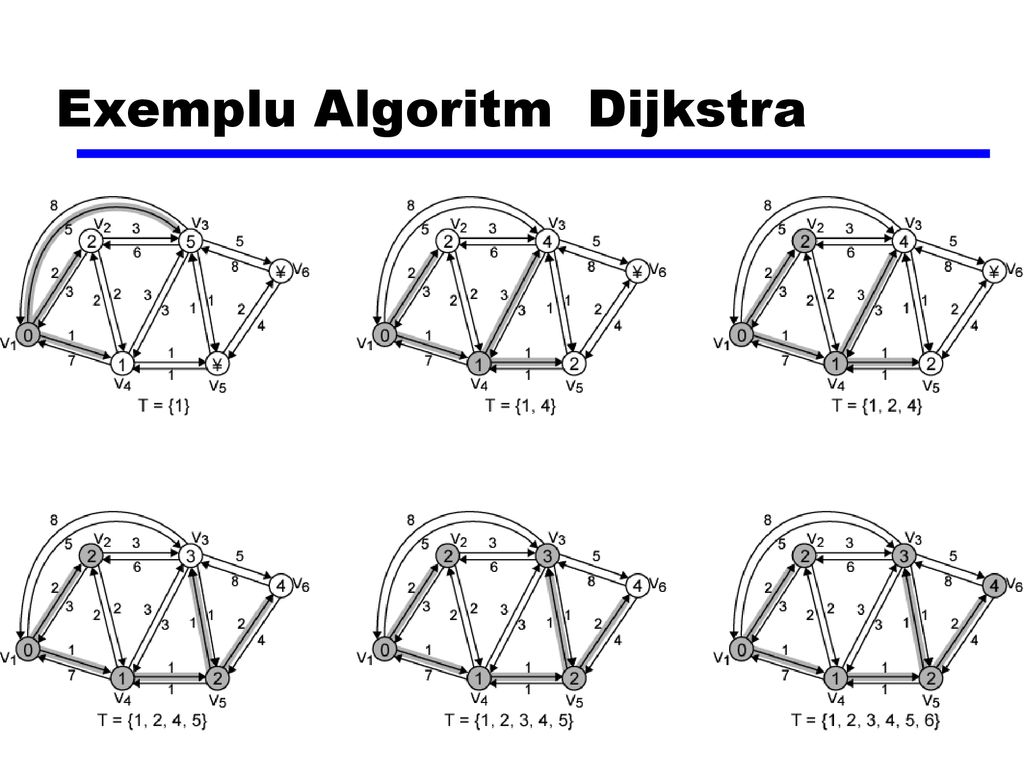
1. De implimentat algoritmii Kruskal și Prim, în baza a 3 tipuri de grafuri conexe (dens, rar și introdus de utilizator).
2. De realizat analiza empirică a algortimilor.

# III. Considerații teoretice

**Algoritmul lui Dijkstra** este o [metodă](https://ro.wikipedia.org/wiki/Metod%C4%83" \o "Metodă) de a stabili [drumul](https://ro.wikipedia.org/wiki/Drum" \o "Drum) de cost minim de la un nod de start la oricare altul dintr-un [graf](https://ro.wikipedia.org/wiki/Graf" \o "Graf). Numele este dat de [Edsger Dijkstra](https://ro.wikipedia.org/wiki/Edsger_Dijkstra" \o "Edsger Dijkstra), [savantul](https://ro.wikipedia.org/wiki/Savant) care l-a descoperit.

Algoritm

1. Se creează o listă cu distanțe, o listă cu nodul anterior, o listă cu nodurile vizitate și un nod curent.
2. Toate valorile din lista cu distanțe sunt inițializate cu o valoare infinită, cu excepția nodului de start, care este setat cu 0.
3. Toate valorile din lista cu nodurile vizitate sunt setate cu fals.
4. Toate valorile din lista cu nodurile anterioare sunt inițializate cu -1.
5. Nodul de start este setat ca nodul curent.
6. Se marchează ca vizitat nodul curent.
7. Se actualizează distanțele, pe baza nodurilor care pot fi vizitate imediat din nodul curent.
8. Se actualizează nodul curent la nodul nevizitat care poate fi vizitat prin calea cea mai scurtă de la nodul de start.
9. Se repetă (de la punctul 6) până când toate nodurile sunt vizitate.

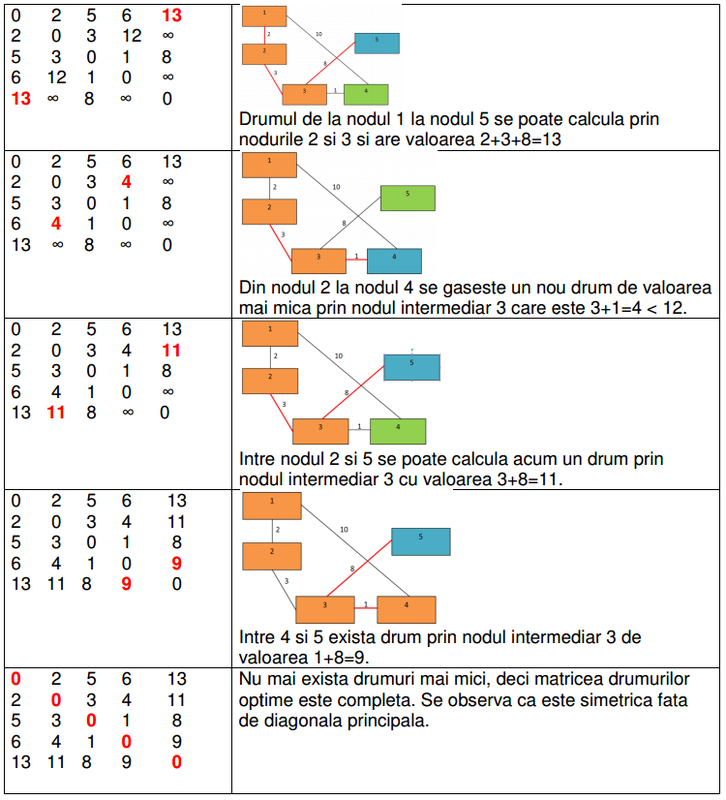


Algoritmul Roy-Floyd este folosit in diverse domeni des intalnite, de la

controlul avioanelor de pe un anumit aeroport pana la jocuri informatice,

rolul principal fiind acela de gasire a drumului de cost minim intre un obiect

principal si o tinta anume.



Dijkstra și Floyd

# IV. Analiza empirică a algortimilor

Tabelul 1. Caz favorabil

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Vârfuri | | 10 | 50 | 100 | 250 | 500 | 750 | 1000 |
| Floyd | Iteratii | 1036 | 126176 | 1004851 | 15655876 | 125124251 | 422015231 | 1000498501 |
| Timp | 0.000122 | 0.000870 | 0.007545 | 0.143780 | 0.7143 | 3.016323 | 5.804519 |
| Dijkstra | Iteratii | 1080 | 127400 | 1009800 | 156787000 | 125249000 | 1000998000 | 1000998000 |
| Timp | 0.00206 | 0.002025 | 0.010310 | 0.208867 | 1.957478 | 9.397624 | 11.136161 |

Fig1.1.Cazul favorabil,Iterații

Fig1.2.Cazul favorabil,Timpul

Tabelul 2. Caz mediu

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Vârfuri | | 10 | 50 | 100 | 250 | 500 | 750 | 1000 |
| Floyd | Iteratii | 1130 | 135470 | 1061769 | 16199641 | 128030545 | 429558611 | 1014805179 |
| Timp | 0.000109 | 0.00101 | 0.11328 | 0.13747 | 0.790954 | 3.36433 | 8.220425 |
| Dijkstra | Iteratii | 1150 | 138700 | 1074300 | 16248750 | 129617000 | 438321750 | 1043641000 |
| Timp | 0.002088 | 0.001448 | 0.011797 | 0.185329 | 1.21134 | 4.229998 | 10.184295 |

Fig2.1.Cazul mediu,Iterații

Fig2.2.Cazul mediu,Timpul

Tabelul 3.Caz defavorabil

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Vârfuri | | 10 | 50 | 100 | 250 | 500 | 750 | 1000 |
| Floyd | Iteratii | 1084 | 135156 | 1058732 | 16172400 | 127911506 | 429130764 | 1013838398 |
| Timp | 0.00016 | 0.001239 | 0.010198 | 0.197156 | 1.304249 | 4.756564 | 11.145589 |
| Dijkstra | Iteratii | 1260 | 139400 | 1070300 | 16403500 | 131849500 | 447167250 | 1076346000 |
| Timp | 0.000348 | 0.001349 | 0.012779 | 0.211794 | 1.645894 | 6.03904 | 15.445723 |

Fig3.1.Cazul defavorabil,Iterații

Fig3.2.Cazul defavorabil,Timpul

# V. Concluzie:

În această lucrare de laborator s-au studiat algoritmii după tehnica greedy, Prim și Kruskal, conform timpului de execuție și a numărului de iterații. Algoritmii Greedy aleg soluția optimă pentru fiecare etapă în speranța de a găsi un optim global, iar algoritmii cercetați aleg muchiile fie cu ponderea cea mai mică după o sortare fără a forma un ciclu, fie începând cu un vârf se alege muchia cu cea mai mică pondere la acel moment.

În cazul meu eu ma verificat trei cazurii de complexitare: favorabil, mediu și defavorabil, pentru diferite valori ale vîrfurilor de la 10 pînă la 1000.

Problemele care cercetează arborii minimi de acoperire au întrebuițări în viața cotidiană în telecomunicații, rețele de transport, electrice. Algoritmi care se bazează pe MSP sunt aplicați pentru recunoașterea scrisului de mână cu expresii matematice, proiectarea circuitelor și măsurarea omogenității materialelor bidimensionale.

**VI Codul sursă:**