Sprawozdanie 2

23.03.2020r.
prowadzący: Marta Emirsajłow
termin: pt 7.30

Kinga Długosz 249002

1. Cel projektu

Celem projektu była realizacja jednego z dwóch podanych algorytmów wyszukiwania najkrótszej ścieżki między dwoma wierzchołkami w grafie, jak również analiza efektywności dwóch rodzajów implementacji grafu- macierzy i listy sąsiedztwa- dla 5 różnych ilości wierzchołków.

2. Wprowadzenie

Testowany algorytm: Bellmana-Forda

Rodzaj grafu: nieskierowany

Reprezentacja grafu:

- Lista sąsiedztwa
- Macierz sąsiedztwa

Testowana liczba wierzchołków grafu:

- 10
- 50
- 100
- 500
- 1000

Testowane gęstości grafu:

- 25%
- 50%
- 75%
- 100%

3. Opis algorytmu

Algorytm Bellmana-Forda jest wolniejszy od algorytmu Dijkstry. W przeciwieństwie do niego może obsługiwać grafy z ujemnymi wagami krawędzi, warunkiem jest jednak nie występowanie ujemnych cykli.

Złożoność obliczeniowa algorytmu w zależności od rodzaju implementacji:

- Lista sąsiedztwa → ( )
- Macierz sąsiedztwa → (V^3) ,gdzie V- liczba wierzchołków, E- liczba krawędzi, gęstość grafu = $_{(\ -1)^{2*}}$

4. Przewidywane wyniki

Można zauważyć, iż złożoność algorytmu dla macierzy sąsiedztwa zależy tylko od liczby węzłów. Oznacza to, że gęstość nie powinna mieć wpływu na czas realizacji algorytmu. Dla listy sąsiedztwa znaczenie ma liczba krawędzi i gęstość grafu.

Przykładowa kalkulacja najbardziej pesymistycznego przypadku:

V=4 ,d=100% -> E=6

Dla listy O=VE=24

Dla macierzy $O=V^3=64$

W teorii:

Dla macierzy V*V*V

Dla listy V*V*(V-1)

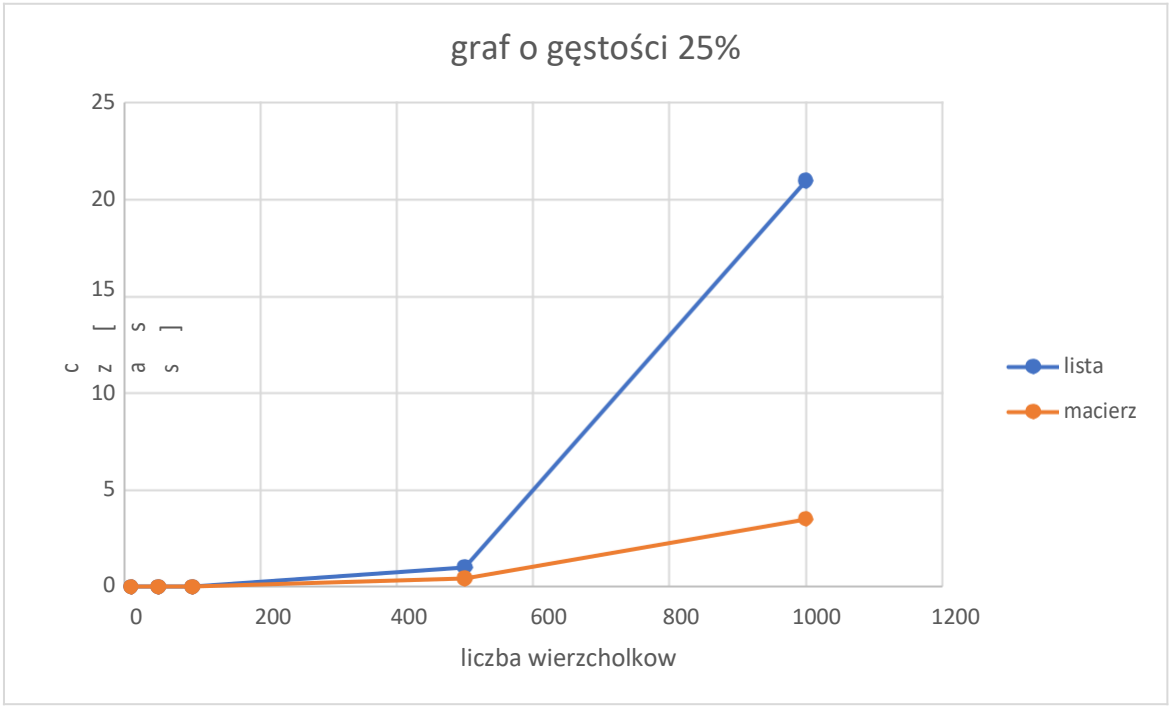
Widać więc ,że lista powinna być zawsze szybsza od macierzy.

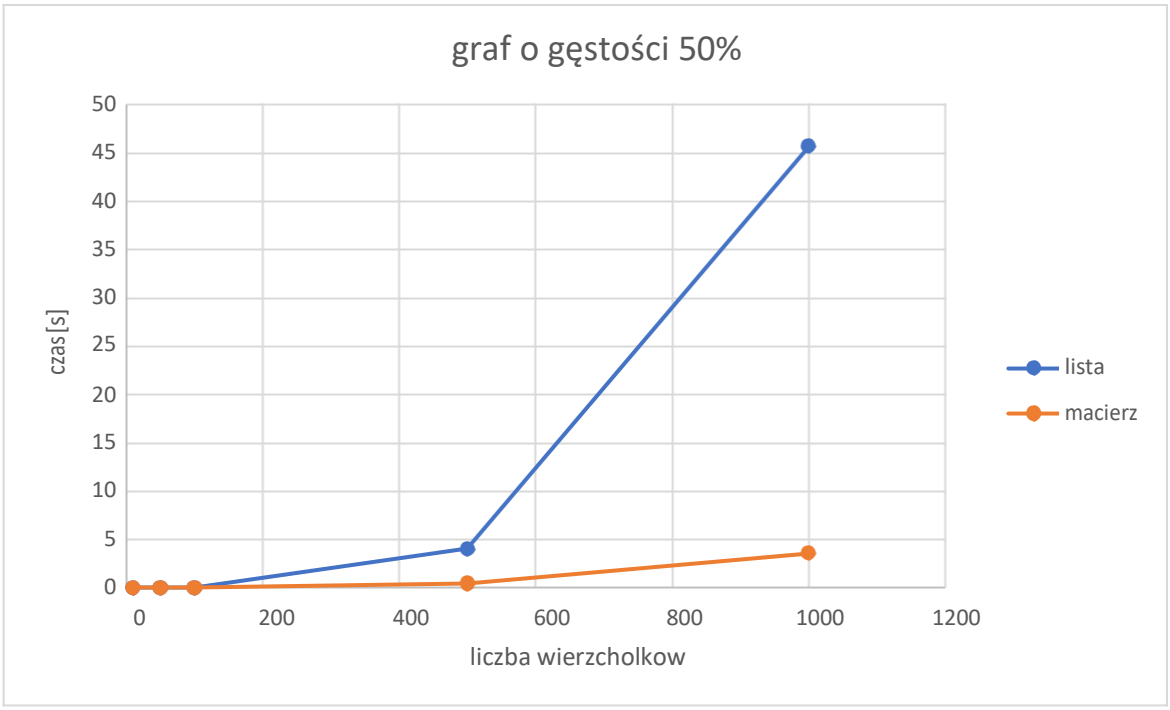
5. Budowa programu

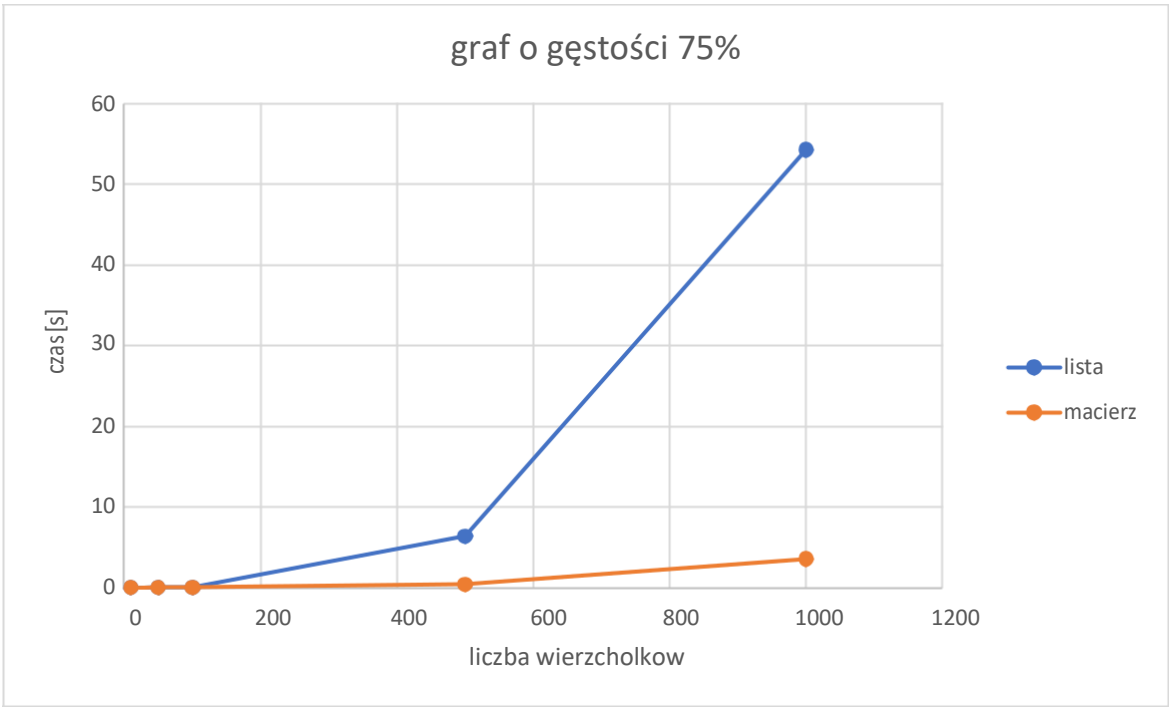
Zgodnie z zaleceniami program został napisany obiektowo. Głównymi klasami są reprezentacje grafów w formie listy i macierzy- odpowiednio GraphList i GraphMatrix. Z tego względu algorytm belllmana-forda został napisany w formie 2 funkcji, jednej dla listy i dla mierzy- odpowiednio bellman-ford_list i bellman-ford_matrix. Istnieje możliwość wygenerowania grafu o zadanej wielkości i gęstości, jak również wczytanie go z pliku .txt, którego schemat został sprecyzowany w poleceniu. Została również zaimplementowana funkcja umożliwiająca zapis wyniku działania algorytmu do pliku .txt.

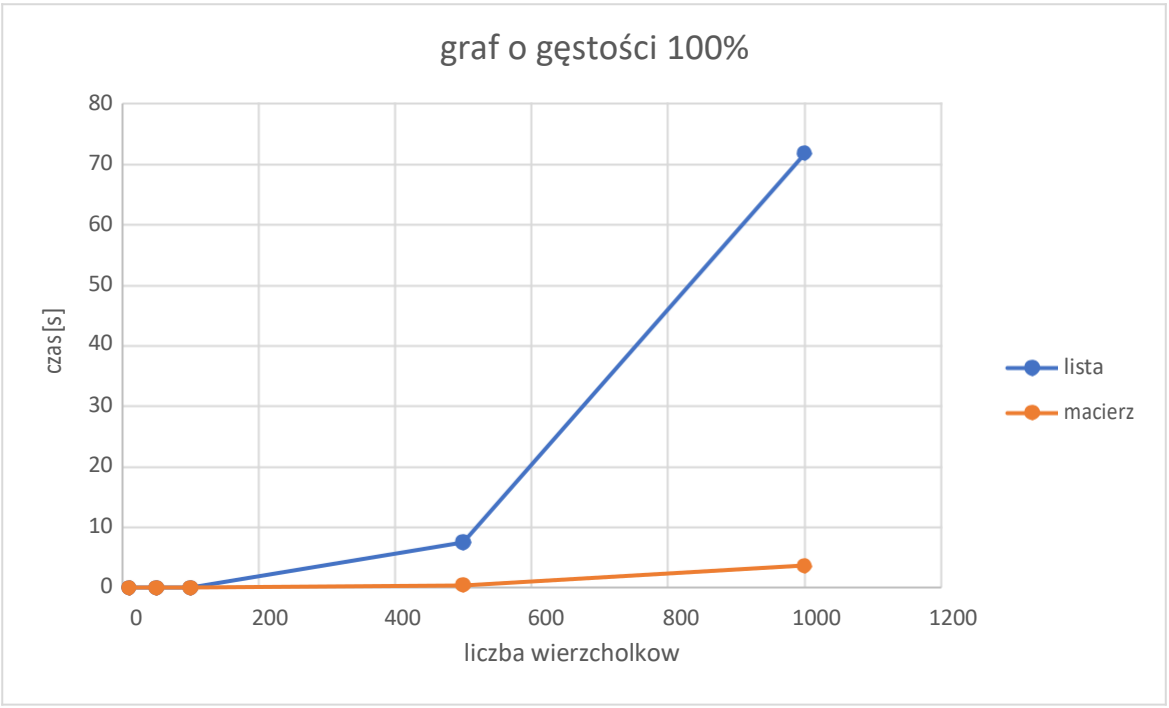
6. Wyniki pomiarów

- czas algorytmu od liczby wierzchołków dla konkretnej gęstości dla różnych reprezentacji grafu

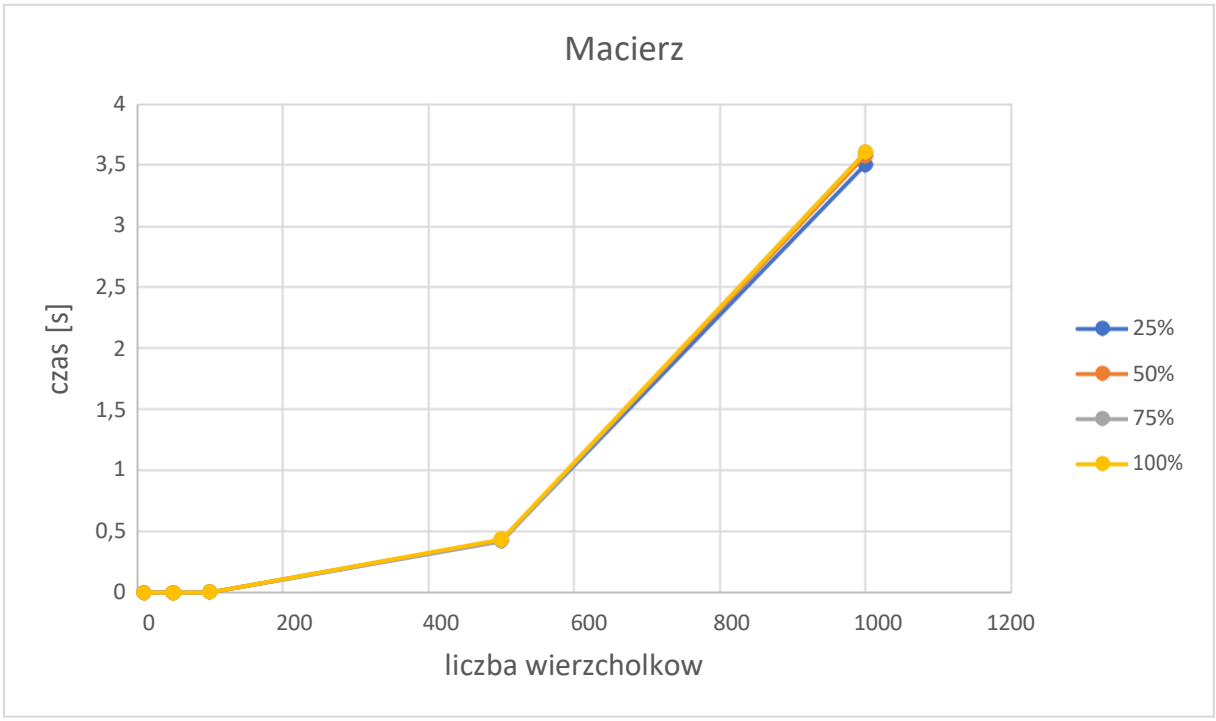





graf o gęstości 75%
60
50
40
30
20
10
0
czas [s]
0
200
400
600
800
1000
1200
liczba wierzcholkow
lista
macierz


graf o gęstości 100%
80
70
60
50
40
30
20
10
0
czas [s]
0
200
400
600
800
1000
1200
liczba wierzcholkow
lista
macierz

- czas algorytmu od liczby wierzchołków dla różnych gęstości tej samej reprezentacji grafu



7. Tabele z średnimi czasami na podstawie których powstały powyższe wykresy

LISTA					
		liczba wierzchołków			
gęstość	10	50	100	500	1000
25%	0,00000464	0,0002197	0,002081	1,00658	21,003
50%	0,000006997	0,0004822	0,005947	4,01169	45,7336
75%	0,000007232	0,0007326	0,011701	6,36476	54,3071
100%	0,000008411	0,0009573	0,012878	7,52743	71,8095

MACIERZ					
	liczba wierzchołków				
gęstość	10	50	100	500	1000
25%	0,000004617	0,0004451	0,003588	0,426132	3,50215
50%	0,000004824	0,000444	0,003537	0,425833	3,57526
75%	0,000005017	0,0004486	0,003568	0,423239	3,6051
100%	0,000005321	0,0004486	0,003629	0,435827	3,6071

8. Wnioski

Tak jak się spodziewano czas działania algorytmu Bellmana-Forda dla implementacji grafu w formie macierzy sąsiedztwa nie zależy od gęstości. Natomiast dla implementacji w formie listy sąsiedztwa czas rośnie wraz ze wzrostem gęstości grafu. To co jest sprzeczne z teorią to porównanie czasu działania algorytmu w zależności od rodzaju implementacji. Mianowicie spodziewano się krótszego czasu dla listy, a wyszło na odwrót- implementacja grafu w formie macierzy sąsiedztwa działa szybciej. Różnica wynika z ubogo zaimplementowanego algorytmu Bellmana-Forda dla reprezentacji grafu jako listy sąsiedztwa.