Projektowanie algorytmów i metody sztucznej inteligencji
Projekt
10.05.2021

Projekt 2
Grafy

Autor
Mikołaj Zapotoczny
(252939)

Prowadzący
Mgr inż. Marta Emirsajłow

1 Wprowadzenie

Zrealizowałem zadanie na ocenę bdb. Problem najkrótszej ścieżki w grafie między dwoma wierzchołkami polega na znalezieniu w grafie ważonym najkrótszego połączenia pomiędzy tymi wierzchołkami. Szczególnymi przypadkami tego problemu są:

- znalezienie najkrótszej ścieżki od wybranego wierzchołka do wszystkich pozostałych wierzchołków.
- znalezienie najkrótszej ścieżki pomiędzy dwoma wybranymi wierzchołkami (ja wybrałem tę opcję).

Do rozwiązywania tego problemu służą (między innymi) dwa następujące algorytmy:

- Dijkstry - przy założeniu, że w grafie nie ma wag ujemnych. Pesymistyczna złożoność obliczeniowa algorytmu Dijkstry wynosi O(E+VlogV).
- Bellmana-Forda – dopuszczalne są wagi ujemne, ale niedopuszczalne jest istnienie cyklu o koszcie ujemnym. Pesymistyczna złożoność obliczeniowa algorytmu wynosi O(VE).

Ja wybrałem do realizacji algorytm Bellmanaa-Forda. Zaimplementowałem reprezentację grafu w postaci macierzy i listy. Badania wykonałem dla pięciu różnych liczb wierzchołków: 10, 50, 100, 500, 1000 oraz dla czterech różnych gęstości: 25 %, 50%, 75% i 100%. Dla każdego zestawu: reprezentacja grafu, liczba wierzchołków oraz gęstość wygenerowałem 100 losowych instancji, a na wykresach i w tabelach są wyniki uśrednione policzone potem w Excelu. Skupiłem się na grafach skierowanych.

Link do GitHuba: <KOD PROGRAMU>

2 Opis badanego algorytmu

Algorytm Bellmana-Forda – algorytm służący do wyznaczania najkrótszych ścieżek w grafie. Wyznacza najkrótsze ścieżki z jednego wierzchołka (zwanego wierzchołkiem źródłowym) do pozostałych wierzchołków. W odróżnieniu od algorytmu Dijkstry, algorytm Bellmana-Forda dopuszcza krawędzie o ujemnych wagach, nie mogą istnieć jednak ujemne cykle osiągalne z wierzchołka źródłowego. Algorytm może być również wykorzystywany do sprawdzania, czy w grafie występują ujemne cykle.

Algorytm występuje również pod nazwą algorytm Bellmana-Forda-Moore'a. Opis działania algorytmu jest poniżej.

W trakcie wykonywania algorytmu dla każdego wierzchołka zostają wyznaczone dwie wartości: koszt dotarcia do tego wierzchołka (oznaczmy go jako di) oraz poprzedni wierzchołek na ścieżce (oznaczmy go jako pi). Na początku działania algorytmu dla wierzchołka źródłowego koszt dotarcia wynosi 0 (już tam jesteśmy), a dla każdego innego wierzchołka nieskończoność (w ogóle nie wiemy, jak się tam dostać).

Następnie dla każdej krawędzi (oznaczmy, że aktualnie analizowana krawędź ma wagę k i prowadzi z wierzchołka u do wierzchołka v) wykonujemy następującą czynność:

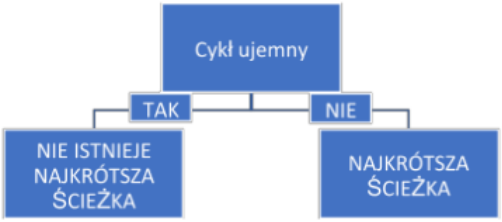
Jeżeli dv > du + k, to ustawiamy wartość dv na du + k, a wartość pv na u.

Całość (przejrzenie wszystkich krawędzi) należy powtórzyć n-1 razy, gdzie n jest liczbą wierzchołków. W każdej iteracji należy przejrzeć wszystkie krawędzie w tej samej kolejności. Jeśli w którejś iteracji nie nastąpią żadne zmiany, wykonywanie algorytmu można przerwać wcześniej.

Relaksacja - sprawdzenie, czy przy przejściu daną krawędzią grafu (u,v) z ‘u’ do ‘v’, nie otrzymamy krótszej niż dotychczasowa ścieżki z ‘s’ do ‘v’. Jeżeli tak, to zmniejszamy oszacowanie wagi najkrótszej ścieżki d[v].

Algorytm Bellmana-Forda

Podstawowe informacje	
Ważony graf skierowany	
Problem najkrótszej ścieżki	
Złożoność czasowa (najlepszy=najgorszy)	O(|V|*|E|)
Złożoność pamięciowa	O(|V|)
Idea algorytmu polega na relaksacji	
Wagi na krawędziach mogą być ujemne	
Robimy „relaksację” dla każdej krawędzi |V| - 1 razy	



TWIERDZENIE 25.14. (Poprawność algorytmu Bellmana-Forda)
Niech $G = (V, E)$ będzie ważonym grafem skierowanym ze źródłem s i funkcją wagową $w: E \to \mathbf{R}$. Załóżmy, że algorytm Bellmana-Forda został wykonany dla grafu G. Jeśli graf G nie ma cykli o ujemnach wagach osiągalnych ze źródła s, to algorytm zwraca wartość TRUE, $d[v] = \delta(s, v)$ dla każdego wierzchołka $v \in V$ i graf poprzedników G_π jest drzewem najkrótszych ścieżek o korzeniu w s. Jeśli natomiast graf G ma cykl o ujemnej wadze osiągalny z s, to algorytm zwraca wartość FALSE.

3 Przebieg eksperymentów

Najpierw należy wygenerować dane do naszych eksperymentów. Mój program posiada taką opcję w menu początkowym, więc ją wybieram i czekam, aż dane się wygenerują do pliku tekstowego. Zajmuje to dłuższą chwilę czasu.

Gdy dane już się wygenerują to przechodzimy do wykorzystania tych danych. Wybierając opcję związaną z mierzeniem czasu uruchamia się druga część programu, która też zabiera bardzo dużo czasu (dla przypadku opisanego w sprawozdaniu wygenerowanie danych i użycie ich do obliczeń zajęło ponad 1.5h, a dane zajęły ok. 1.8 GB). Plik wynikowy wrzucam do Excela i obliczam uśrednione wyniki oraz wykreślam wykresy, które są widoczne poniżej.

Poniżej również są przedstawione fragmenty plików generowanych przez program. Jak widzimy plik dane.txt zawiera poprawną składnię przedstawioną w opisie sprawozdania, co można potwierdzić patrząc w kod. Plik pomiar.txt przedstawia czas szukania ścieżki dla listy i macierzy, nie przedstawia jednak ścieżki w grafie. Są to ogromne pliki, których otwieranie potrafi zawiesić komputer.

*dane.txt ×

Wczytywanie pliku **dane.txt** z katalogu **~/Pulpit**

```
10 11 1
0 1 86
1 2 68
2 3 97
3 4 79
4 5 14
5 6 23
6 7 7
7 8 4
8 9 43
9 0 2
9 1 2
10 11 7
0 1 37
1 2 66
2 3 97
3 4 80
4 5 87
5 6 1
6 7 32
7 8 34
8 9 51
9 0 9
9 1 5
10 11 9
0 1 97
1 2 48
2 3 73
3 4 21
4 5 50
```

pomiar.txt ×

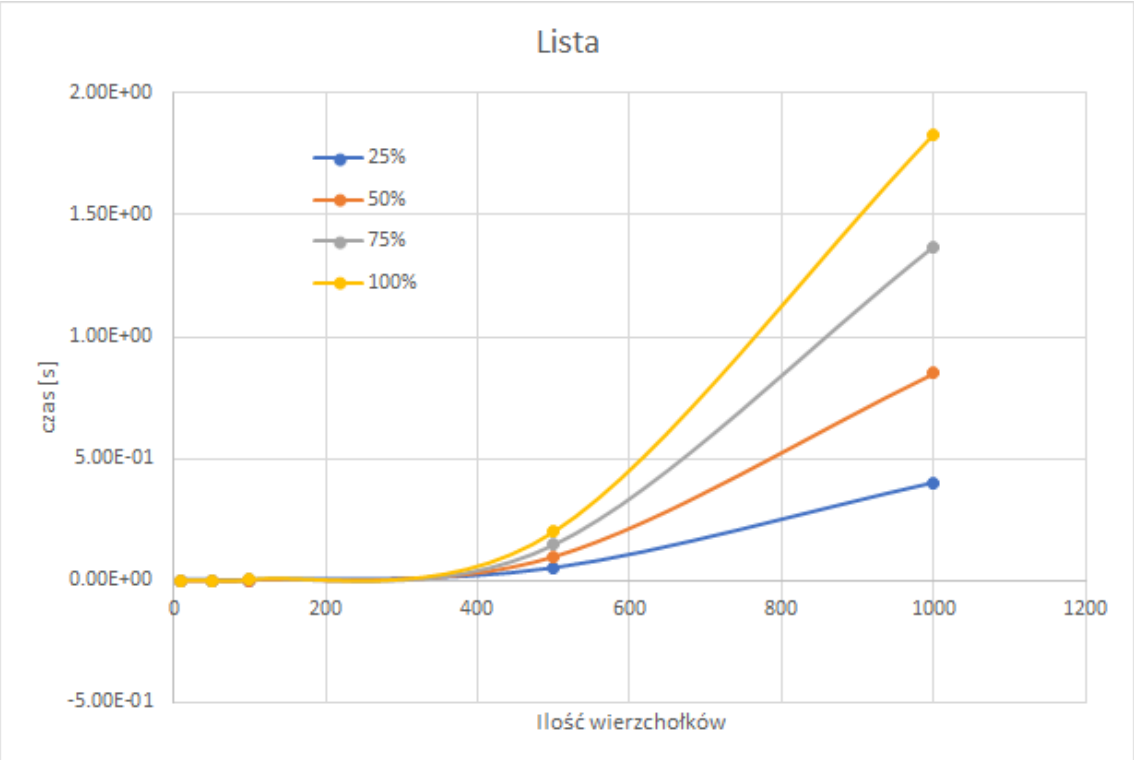
```
macierzy: 2.94888
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.82831 ,a dla
macierzy: 3.52534
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83068 ,a dla
macierzy: 2.69695
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83289 ,a dla
macierzy: 3.64519
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83323 ,a dla
macierzy: 3.04601
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83215 ,a dla
macierzy: 3.5653
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83299 ,a dla
macierzy: 3.5775
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83328 ,a dla
macierzy: 3.0882
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83354 ,a dla
macierzy: 2.82388
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83135 ,a dla
macierzy: 3.34963
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83454 ,a dla
macierzy: 3.12682
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.83217 ,a dla
macierzy: 3.29494
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.84197 ,a dla
macierzy: 3.30584
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.87469 ,a dla
macierzy: 2.7078
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.8549 ,a dla
macierzy: 3.19694
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.8488 ,a dla
macierzy: 3.66357
Czas szukania sciezki przy: 1000 i: 100% wypelnienia dla listy: 1.86165 ,a dla
macierzy: 3.09045
```

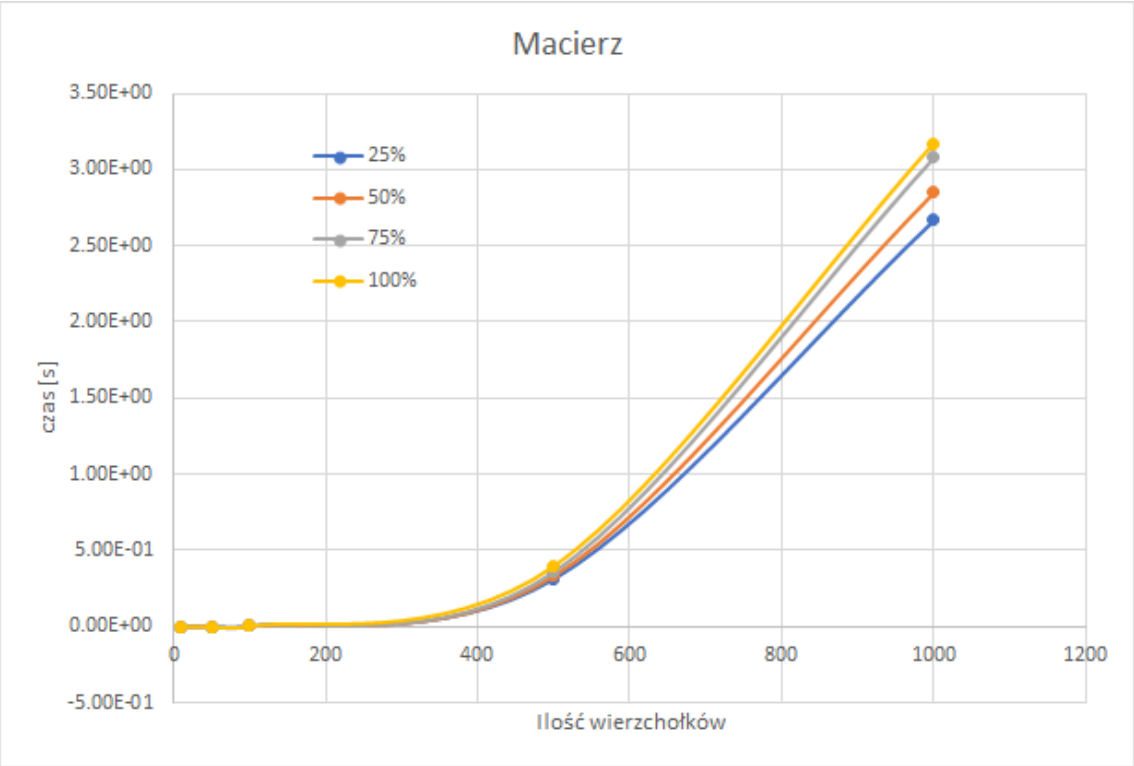
4 Wyniki

4.1 Tabelka z uśrednionymi wynikami pomiarów

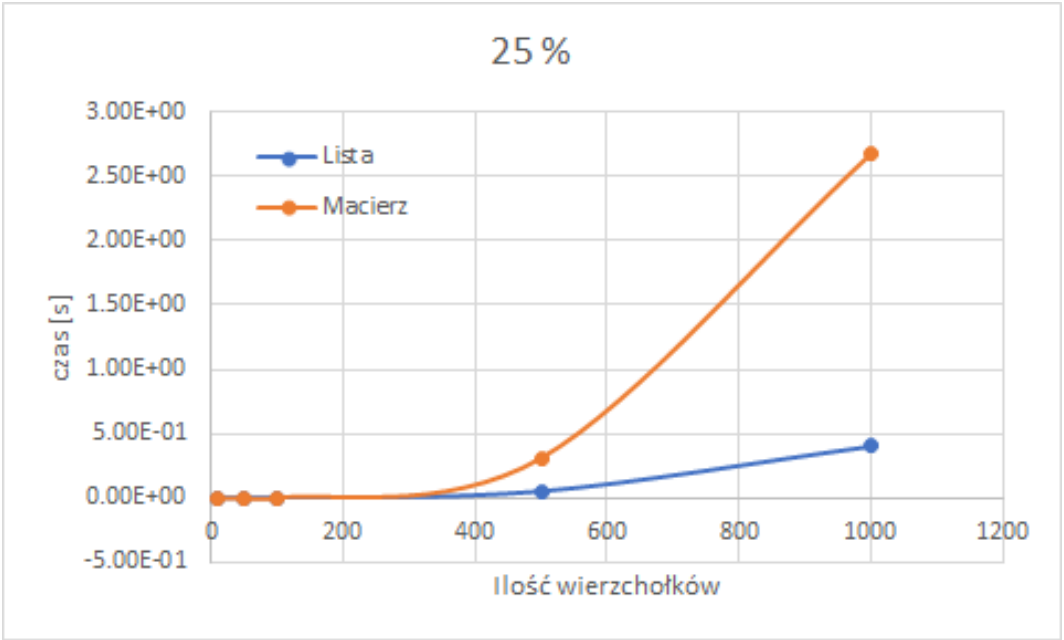
Ilość wierzchołków grafu	Gęstość [%]	czas dla listy [s]	czas dla macierzy [s]
10	25	1.17E-06	2.63E-06
50	25	5.68E-05	3.26E-04
100	25	4.61E-04	2.44E-03
500	25	4.96E-02	3.12E-01
1000	25	4.01E-01	2.67E+00
10	50	1.41E-06	2.92E-06
50	50	1.10E-04	3.50E-04
100	50	8.69E-04	2.61E-03
500	50	9.82E-02	3.35E-01
1000	50	8.49E-01	2.85E+00
10	75	1.69E-06	3.02E-06
50	75	1.72E-04	3.68E-04
100	75	1.60E-03	3.42E-03
500	75	1.46E-01	3.62E-01
1000	75	1.37E+00	3.08E+00
10	100	2.05E-06	3.25E-06
50	100	2.37E-04	3.90E-04
100	100	1.83E-03	2.95E-03
500	100	1.99E-01	3.99E-01
1000	100	1.83E+00	3.17E+00

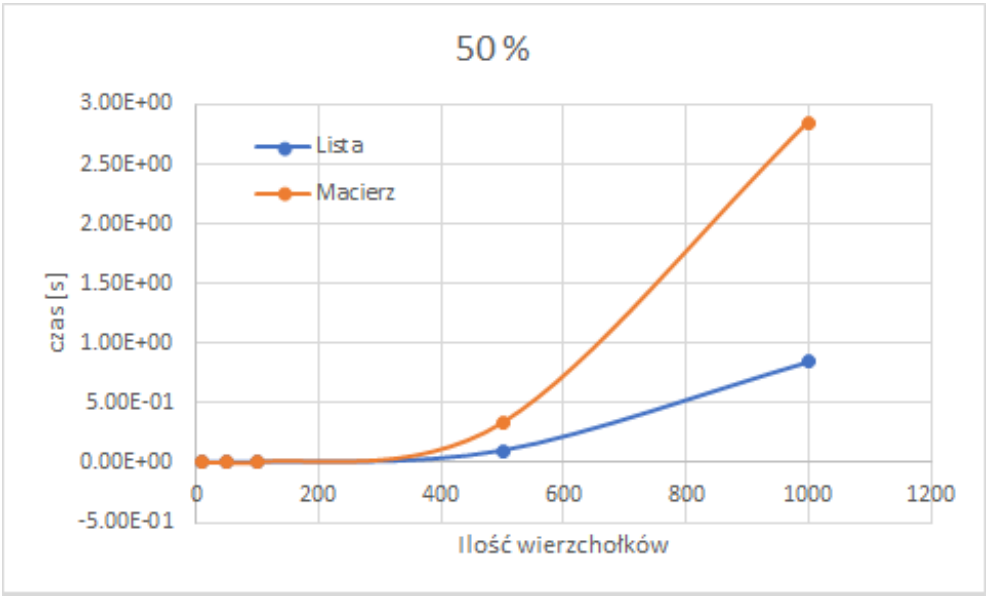
4.2 Wykresy dla każdej reprezentacji grafu osobno

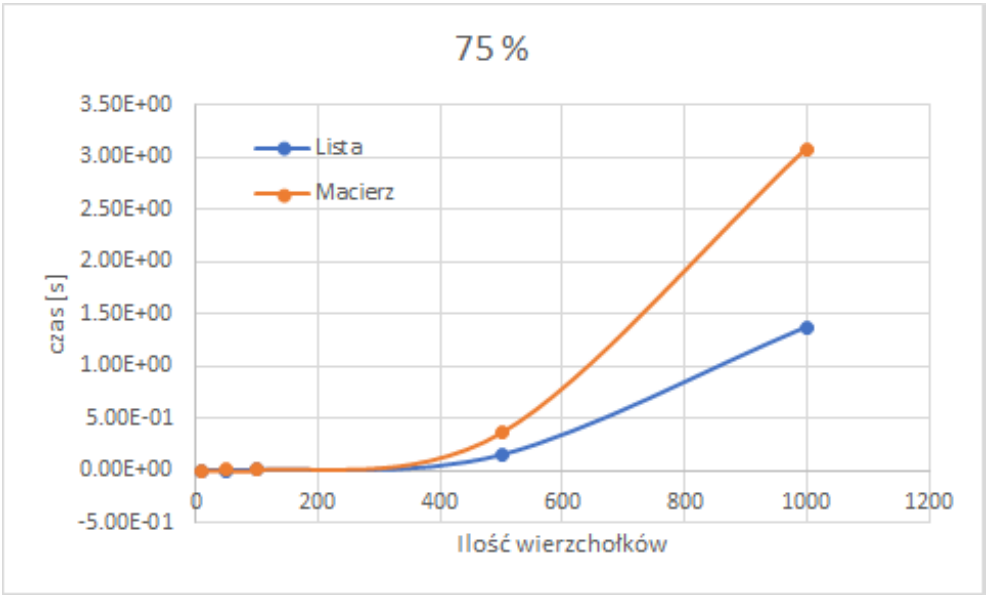


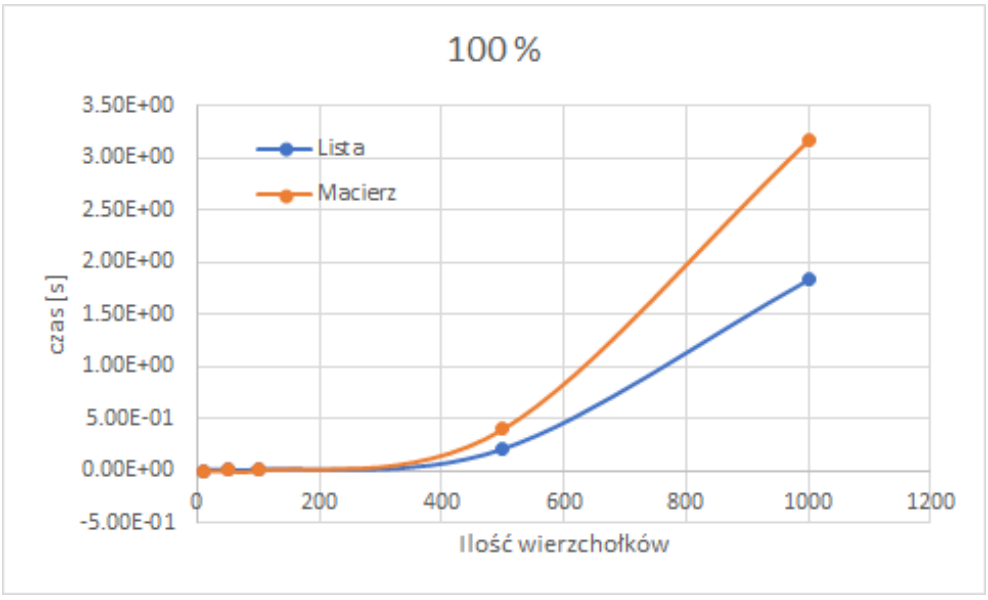


4.3 Wykresy dla każdej gęstości osobno




50 %
3.00E+00
2.50E+00
2.00E+00
1.50E+00
1.00E+00
5.00E-01
0.00E+00
-5.00E-01
czas [s]
Lista
Macierz
0
200
400
600
800
1000
1200
Ilość wierzchołków


75 %
3.50E+00
3.00E+00
2.50E+00
2.00E+00
1.50E+00
1.00E+00
5.00E-01
0.00E+00
-5.00E-01
czas [s]
Lista
Macierz
0
200
400
600
800
1000
1200
Ilość wierzchołków


100 %
3.50E+00
3.00E+00
2.50E+00
2.00E+00
1.50E+00
1.00E+00
5.00E-01
0.00E+00
-5.00E-01
czas [s]
Lista
Macierz
0
200
400
600
800
1000
1200
Ilość wierzchołków

5 Wnioski

- Próby otwarcia plików tekstowych ważących ok. 1.5 GB potrafią zawiesić komputer.
- Widzimy, że im większa gęstość tym czas trwania się wydłuża, co nie jest zaskoczeniem.
- Widzimy, że czas trwania algorytmu na macierzach jest dłuższy niż na listach.
- Widzimy też, że dla macierzy nie ma tak dużej różnicy w czasie gdy grafy różnią się gęstością. Dla list jest ona trochę większa.
- Widzimy też że wraz z wzrostem procentowym gęstości macierzy i listy zmniejsza się rozbieżność w czasie trwania realizacji algorytmu.
- Widzimy, że dla większej liczby wierzchołków algorytmy trwają o wile dłużej niż dla wartości mniejszych.

6 Źródła informacji

- Materiały udostępnione przez prowadzącą.
- https://docplayer.pl/83602648-Podstawy-programowania-2-grafy-i-ich-reprezentacje-arkadiusz-chrobot-9-czerwca-2016.html
- http://www.algorytm.org/klasyczne/grafy-i-ich-reprezentacje.html
- https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm
- http://algorytmy.ency.pl/artykul/algorytm_bellmana_forda