Data:	08.05.2019
Student:	Maciej Popieluch 241565
Kurs:	PAMSI
Prowadzący:	dr inż. Łukasz Jeleń
Termin zajęć:	Środa 7:30

Wprowadzenie

W projekcie zbadano efektywność działania algorytmu Dijkstry. Badania wykonano dla dwóch reprezentacji grafu (macierz sąsiedztwa, lista sąsiedztwa), pięciu liczb wierzchołków (10, 50, 100, 500, 1000) i czterech gęstości (25%, 50%, 75%, 100%).

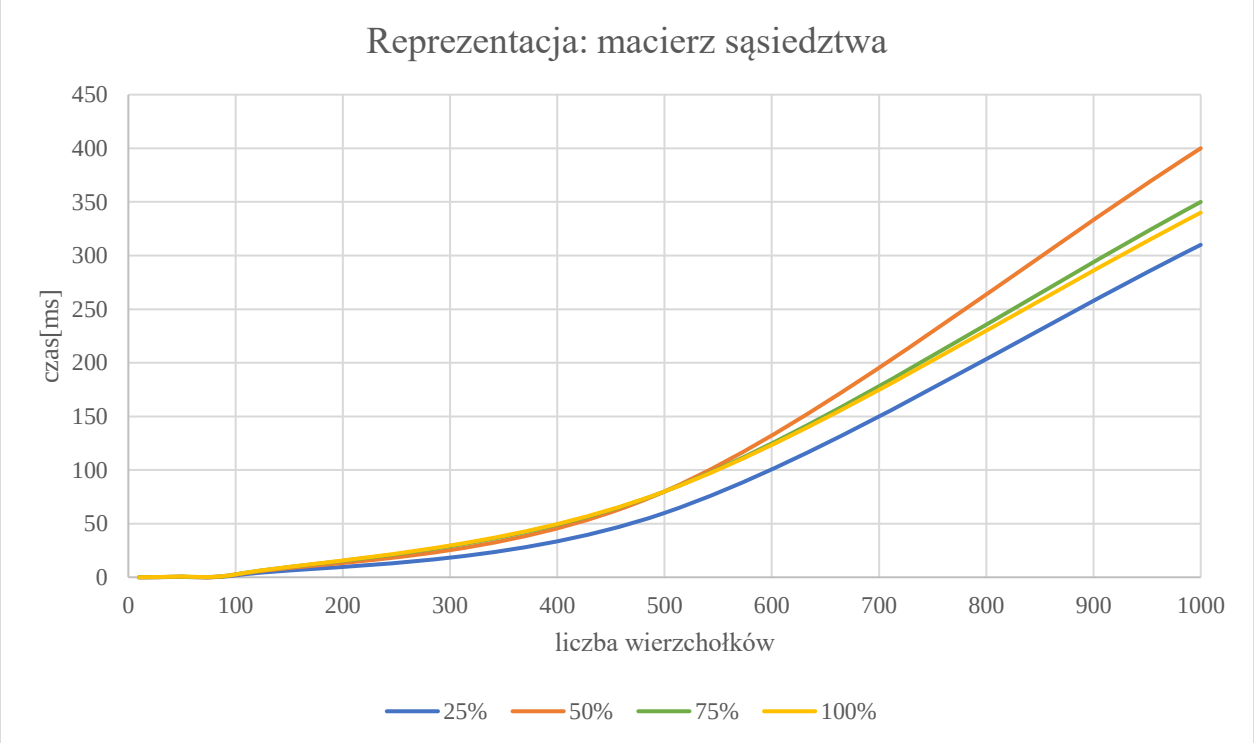
W algorytmie Dijkstry zaimplementowano kolejkę priorytetową typu *min* w formie kopca, której elementy przechowuje lista dwukierunkowa.

Dane do grafu – połączenia wierzchołków i wagi połączeń są wczytywane z pliku tekstowego, wyniki – odległości każdego wierzchołka do wierzchołka startowego i najkrótsza ścieżka, również są zapisywane do pliku tekstowego.

Wyniki

Macierz sąsiedztwa					
	10	50	100	500	1000
25	0,01	0,38	1,8	60	310
50	0,02	0,55	2,4	80	400
75	0,02	0,62	2,7	80	350
100	0,03	0,71	2,7	80	340

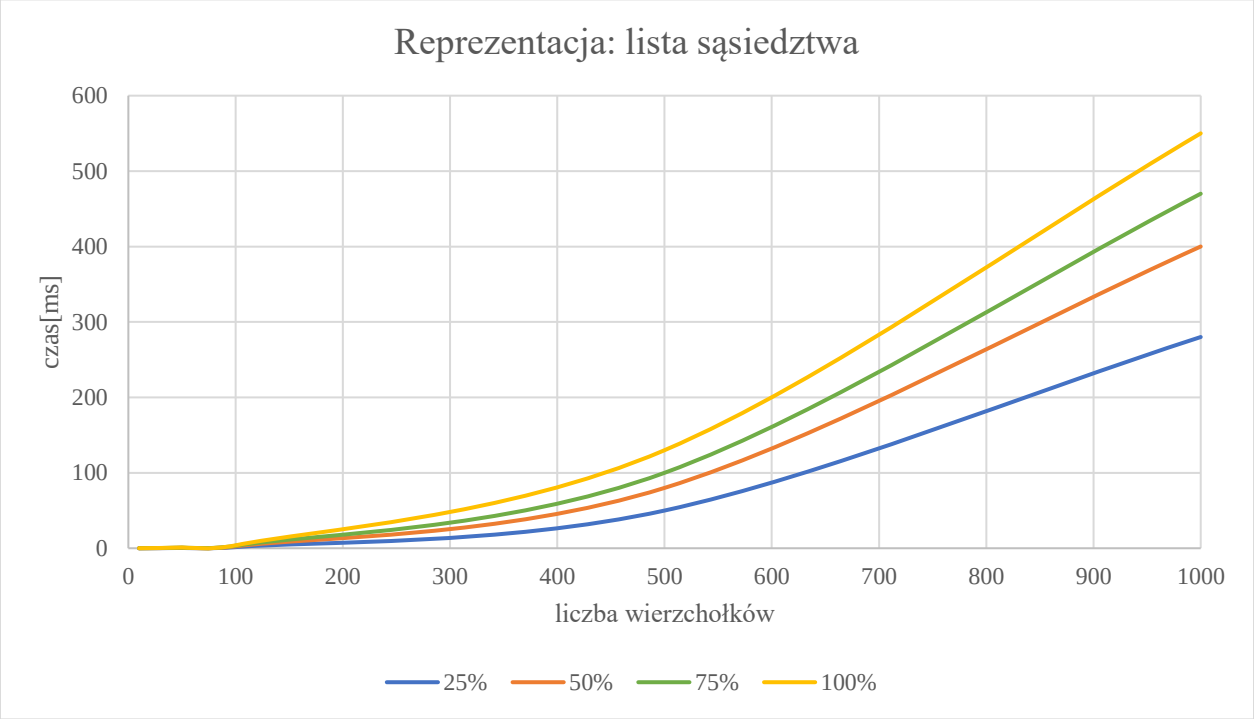
Tabela 1. Czas wykonywania instancji w milisekundach, w zależności od ilości wierzchołków grafu. Uzyskane czasy to uśrednione wyniki po wykonanych 100 losowych instancjach.



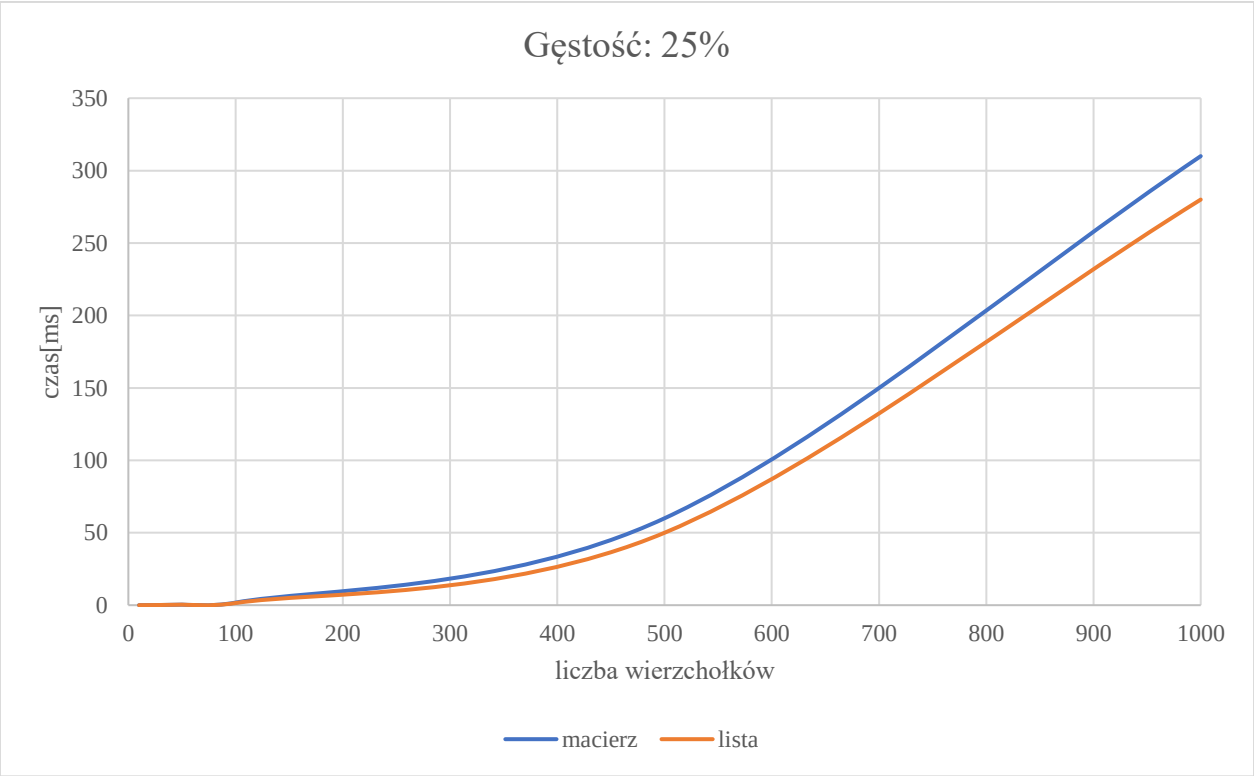
Wykres 1. Zależność czasu wyszukiwania najkrótszych ścieżek od ilości wierzchołków grafu dla różnych gęstości grafów.

Lista sąsiedztwa					
	10	50	100	500	1000
25	0,01	0,37	1,5	50	280
50	0,02	0,60	2,5	80	400
75	0,03	0,80	3,3	100	470
100	0,05	0,99	3,9	130	550

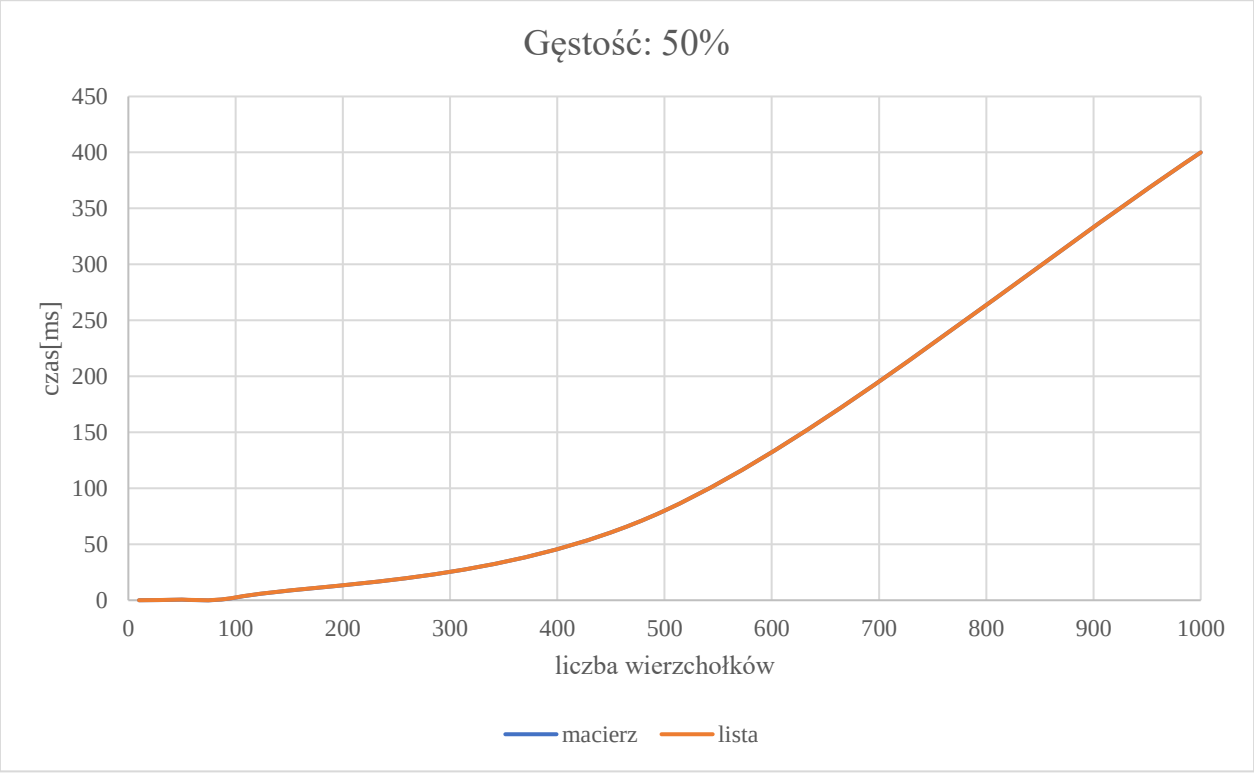
Tabela 2. Czas wykonywania instancji w milisekundach, w zależności od ilości wierzchołków grafu. Uzyskane czasy to uśrednione wyniki po wykonanych 100 losowych instancjach.



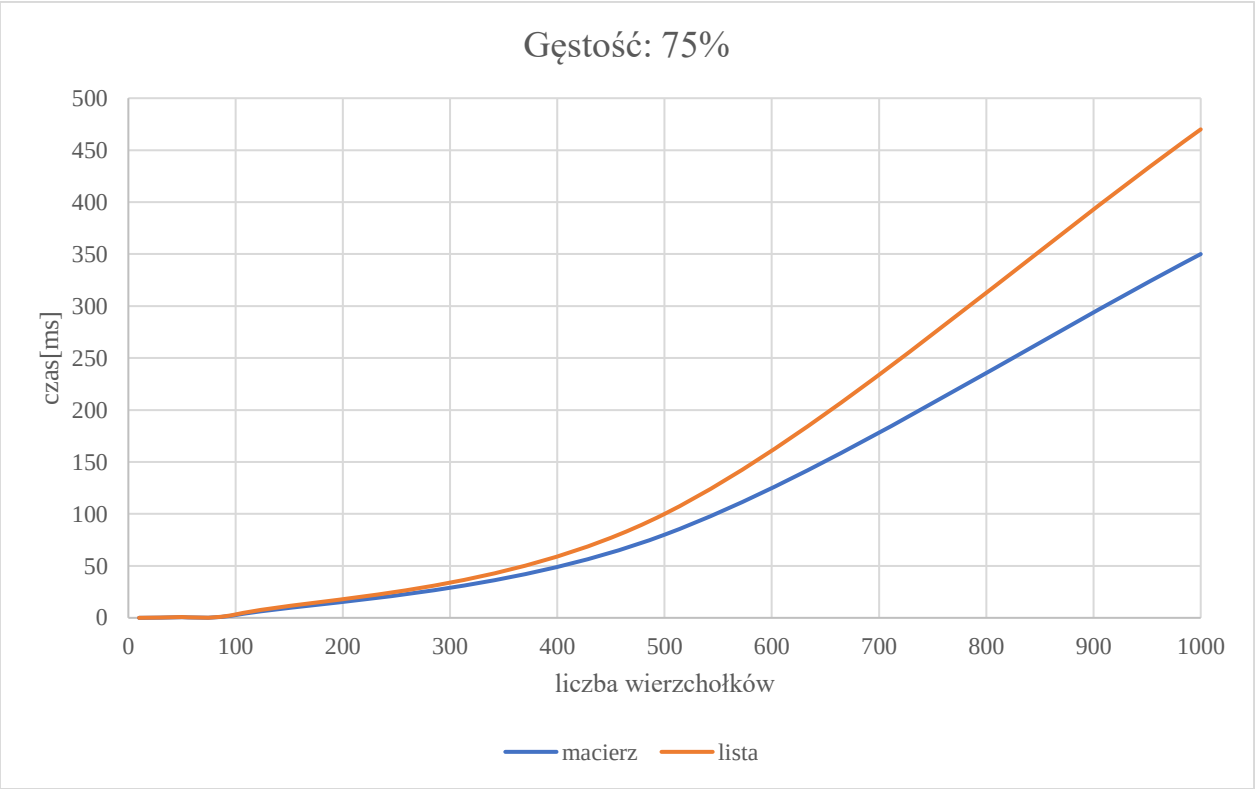
Wykres 2. Zależność czasu wyszukiwania najkrótszych ścieżek od ilości wierzchołków grafu dla różnych gęstości grafów.



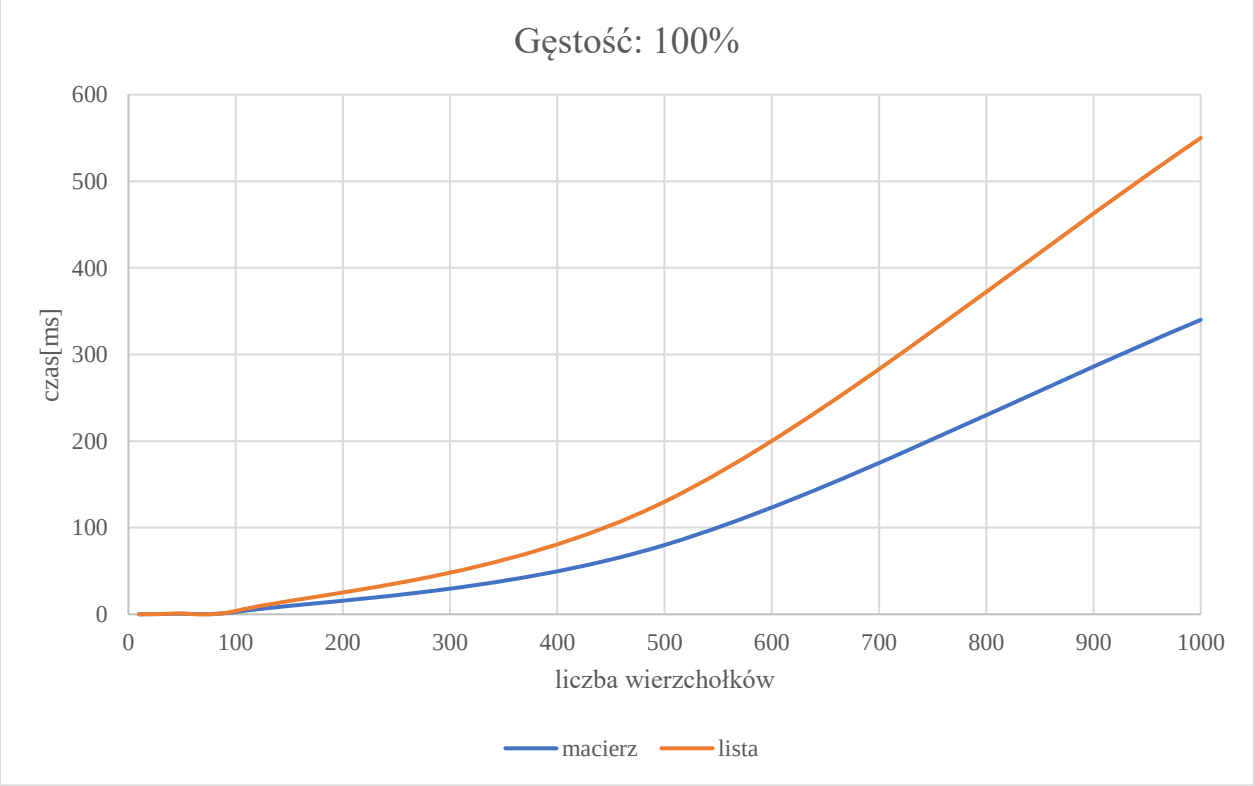
Wykres 3. Zależność czasu wyszukiwania najkrótszych ścieżek od ilości wierzchołków grafu dla dwóch reprezentacji grafu.



Wykres 4.



Wykres 5.



Wykres 6.

Wnioski

Zaimplementowany algorytm Dijkstry działa bez zarzutów dla obu reprezentacji grafów. Algorytm jest bardziej efektywny dla małych gęstości grafu dla reprezentacji za pomocą list sąsiedztwa, w przypadku większych gęstości algorytm działa szybciej, gdy graf ma macierz sąsiedztwa. Dla obu reprezentacji, złożoność czasowa jest bliska O(ElogV) (E – liczba krawędzi, V – liczba wierzchołków) która jest oczekiwana w przypadku implementacji kolejki przez kopiec.

Literatura

Maksymalna wartość danego typu
https://en.cppreference.com/w/cpp/types/numeric_limits
//25.04.2019//

Budowa klasy GrafMacierz (koncept)
https://www.youtube.com/watch?v=BcaZIxaS-F4
//27.04.2019//

Lista jednokierunkowa "Data Structures And Algorithms"
//28.04.19//

Kolejka priorytetowa w formie kopca, implementacja "Data Structures And Algorithms"
//02.05.19//

Algorytm Dijkstry, implementacja "Data Structures And Algorithms"
//03.05.19//

Dijkstra niektóre elementy
https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-using-priority_queue-stl/
//03.05.19//

Zapis danych do pliku tekstowego
http://www.algorytm.edu.pl/pliki-tekstowe/zapis-do-pliku.html
//05.05.19//

Odczyt danych z pliku tekstowego
https://www.p-programowanie.pl/matura-z-informatyki/odczyt-danych-z-pliku-c/
//05.05.19//

Pomiar czasu działania algorytmu Dijkstry
https://www.techiedelight.com/measure-elapsed-time-program-chrono-library/
//06.05.19//