Data oddania projektu:
08.05.2019

Prowadzący:
mgr inż. Marcin Ochman
Wykonawca:
Julia Krzeszowska 241615

SPRAWOZDANIE

PROJEKT nr 3 - Grafy

1. **Cel projektu**

Zapoznanie się z algorytmem Dijkstrty oraz algorytmem Bellmana-Forda służącymi do rozwiązania problemu najkrótszej drogi (ścieżki) w grafie.

2. **Przebieg projektu**

Został napisany program, w którym zostały zaimplementowane wyżej wymienione algorytmy dla dwóch sposobów przechowywania informacji o wierzchołkach w grafie. W projekcie znajdują się następujące klasy:

- klasa *NeighbourList*, która implementuje listę sąsiadów
- klasa *NeighbourMatrix*, która implementuje macierz sąsiedztwa
- klasa *Pair*, która przechowuje parę dowolnych obiektów
- klasa *Timer* służąca do pomiaru czasu
- klasa *Tests* służąca do pojedynczych testów obu algorytmów
- klasa *Algorythms*, w której zostały zaimplementowane algorytmy wyszukujące

Dodatkowo w projekcie znajduje się plik *main.cpp*, w którym wykonano zadane testy dla wybranego algorytmu (algorytm Bellmana-Forda).

Dla pięciu różnych liczb wierzchołków V = {50, 100, 150, 200, 250} oraz gęstości grafu E = {25%, 50%, 75%, 100%} została zbadana efektywność algorytmu Bellmana-Forda oraz Dijkstry. Wygenerowano po 100 losowych instancji, a wyniki uśredniono.
Poniższa tabele zawierają uśrednione wyniki czasu działania algorytmu dla różnej liczby wierzchołków oraz gęstości grafu.
W tabeli *Tab.1* zamieszczono wyniki dla listy sąsiadów, a w tabeli *Tab.2* dla macierzy sąsiedztwa testując algorytm Bellmana-Forda. Czasy podane są w sekundach.

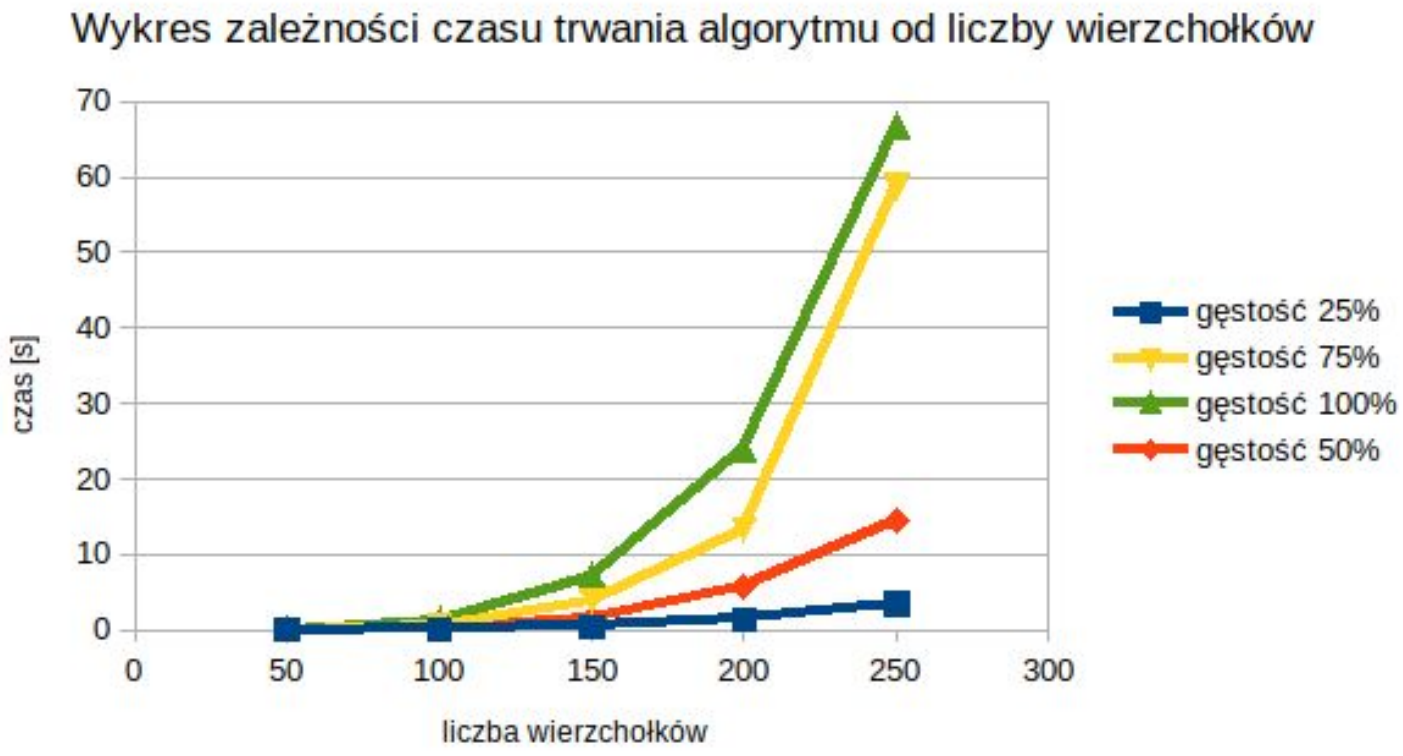
	50	100	150	200	250
25%	0,006533	0,087111	0,0425149	1,410529	3,536144
50%	0,02125	0,320066	1,67996	5,775814	14,449422
75%	0,043874	0,699003	3,903159	13,310085	58,815426
100%	0,074871	1,258657	7,132448	23,888158	66,647557

Tab.1

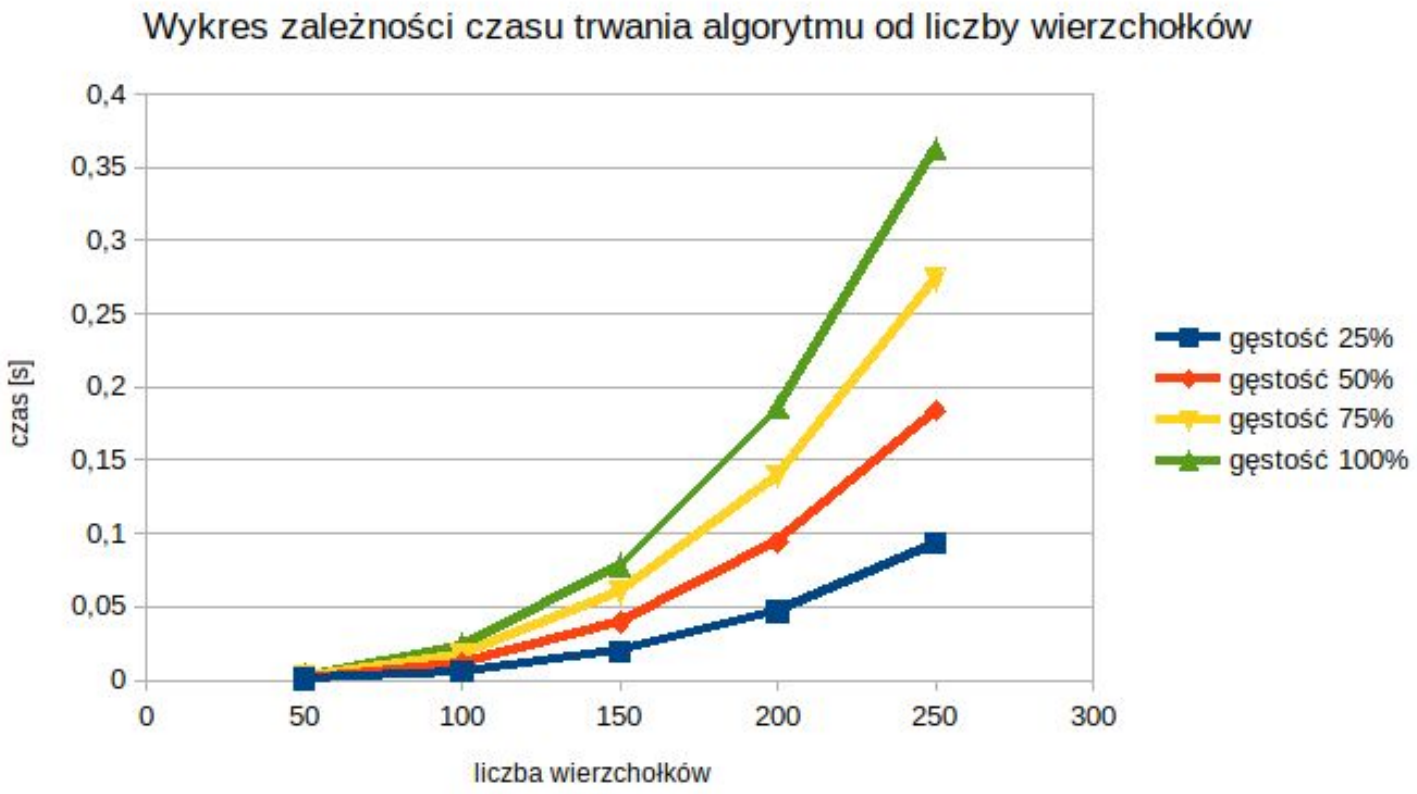
	50	100	150	200	250
25%	0,000748	0,005861	0,019639	0,047069	0,09322
50%	0,00148	0,011868	0,039259	0,093598	0,183848
75%	0,002193	0,017398	0,060168	0,139411	0,274036
100%	0,002904	0,023254	0,077999	0,185345	0,362434

Tab.2

Na podstawie *Tab.1* i *Tab.2* wygenerowano wykresy *Wyk.1* i *Wyk.2*.



Wyk.1 - wykres dla listy sąsiadów



Wyk.2 - wykres dla macierzy sąsiedztwa

W tabeli *Tab.3* zamieszczono wyniki dla listy sąsiadów, a w tabeli *Tab.4* dla macierzy sąsiedztwa testując algorytm Dijkstry. Czasy podane są w sekundach.

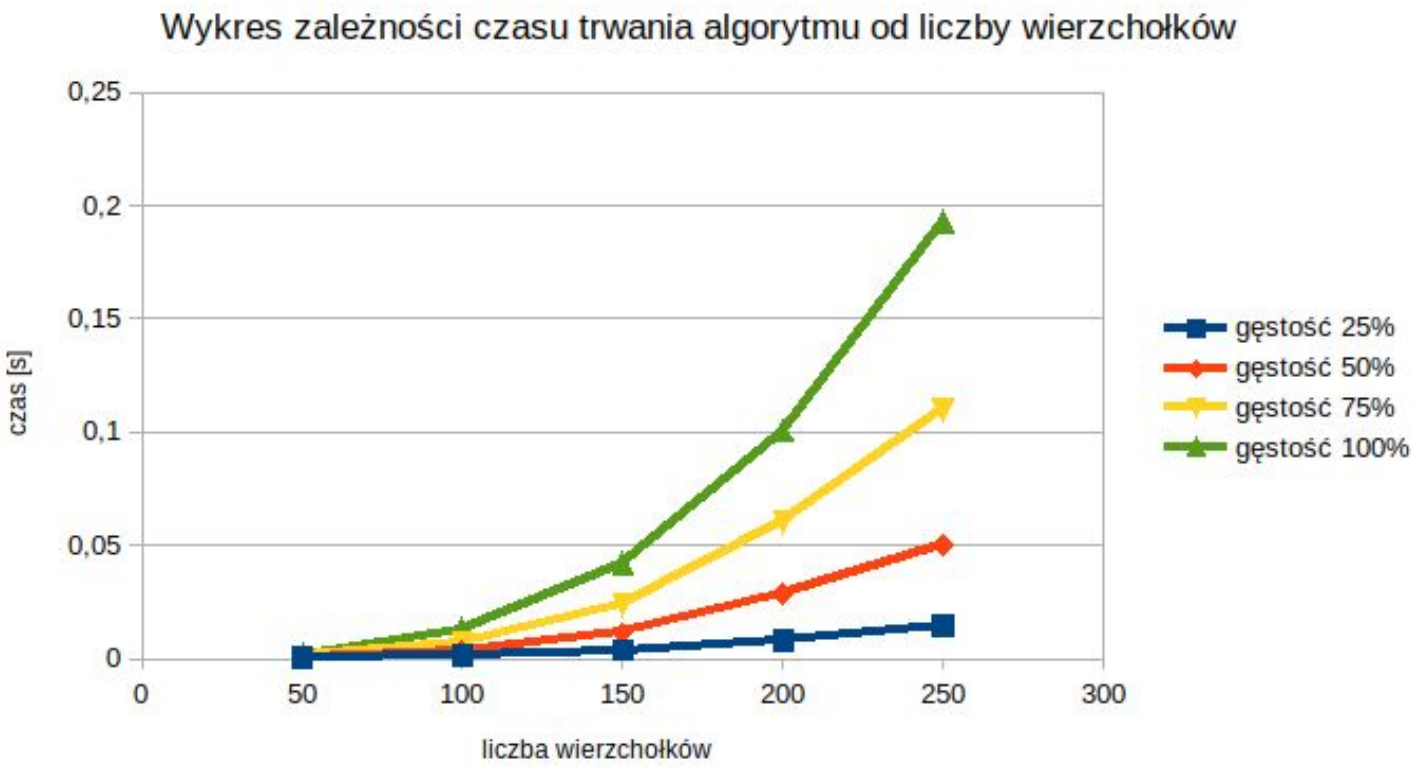
	50	100	150	200	250
25%	0,000522	0,001395	0,003711	0,008068	0,014694
50%	0,000689	0,003747	0,,011342	0,028579	0,050360
75%	0,001167	0,007461	0,024073	0,060269	0,109989
100%	0,001813	0,012705	0,041834	0,100206	0,192455

Tab.3

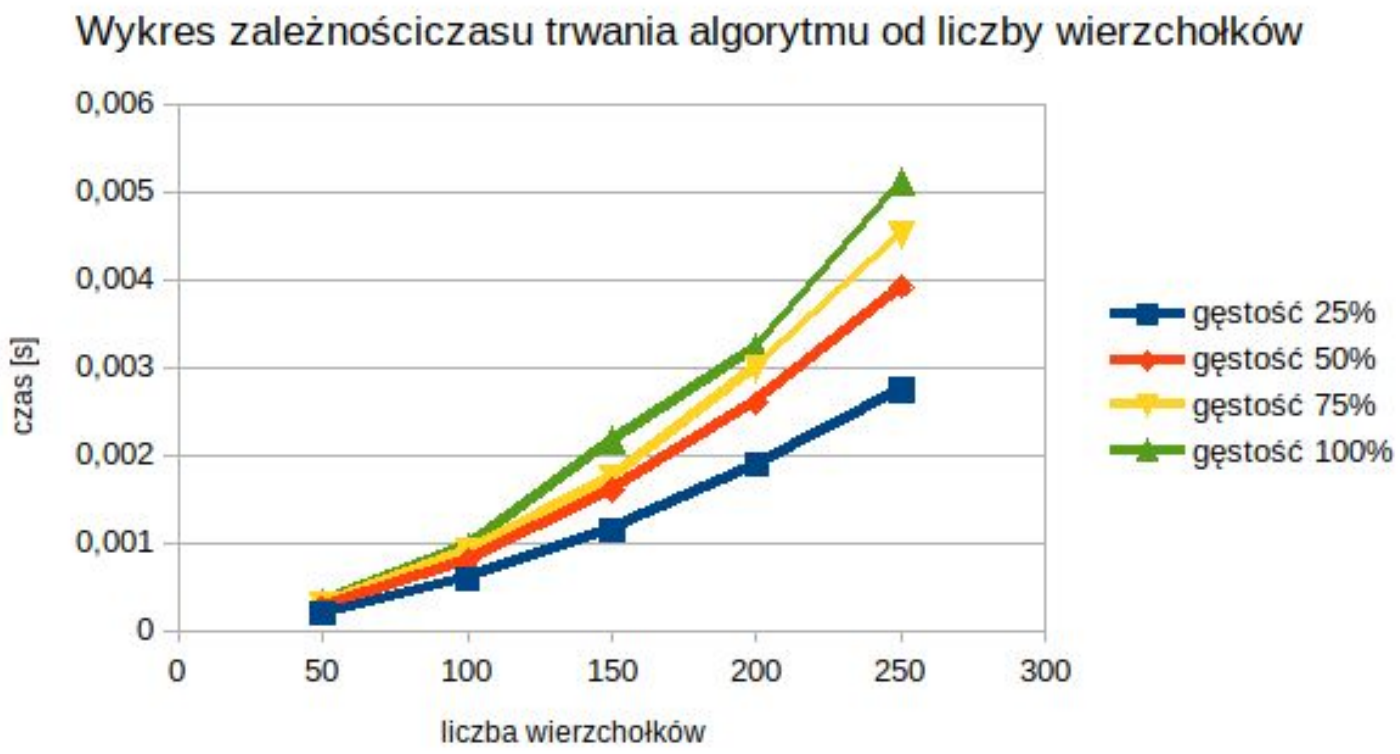
	50	100	150	200	250
25%	0,000200	0,000591	0,001151	0,001894	0,002744
50%	0,000262	0,000800	0,001603	0,002592	0,003915
75%	0,000299	0,000905	0,001753	0,002986	0,004517
100%	0,000313	0,000955	0,002160	0,003222	0,005197

Tab.4

Na podstawie *Tab.3* i *Tab.4* wygenerowano wykresy *Wyk.3* i *Wyk.4.*



Wyk.3 - wykres dla listy sąsiadów



Wyk.4 - wykres dla macierzy sąsiedztwa

3. Wnioski

Zaobserwowano, że zarówno algorytm Bellmana-Forda jak i Dijkstry działa szybciej dla macierzy sąsiedztwa niż dla listy sąsiadów. Z wykresów można odczytać, że złożoność obliczeniowa algorytmu Bellmana-Forda zgadza się ze złożonością podaną w literaturze, tj. O(VE). Także złożoność algorytmu Dijkstry zgadza się ze złożonością podaną w literaturze, tj. O(E+VlogV).