PROJEKTOWANIE ALGORYTMÓW I METODY SZTUCZNEJ INTELIGENCJI – PROJEKT 2

10.05.2020

Aleksander Górecki, 249003

Prowadzący zajęcia:

Mgr inż. Marta Emirsajłow
Termin zajęć: Piątek 9:15

1. Wstęp teoretyczny

Projekt drugi poświęcony był grafom, dwóm powszechnym sposobom ich implementacji oraz zależności między sposobem implementacji grafu, a efektywnością czasową algorytmu Bellmana-Forda, szukającego najkrótszej ścieżki do każdego z wierzchołków grafu. Główna część projektu opierała się na pomiarze, a następnie uśrednieniu czasu pracy wybranego algorytmu dla zestawu 100 losowo przygotowanych grafów, w zależności od ilości wierzchołków grafów w zestawie, oraz ich gęstości, przekładającej się na ilość krawędzi łączących wierzchołki.

2. Sposoby reprezentacji grafu

Istnieje wiele sposobów reprezentacji grafu jako struktury danych, w projekcie skupiono się jednak tylko na 2 wybranych sposobach, liście sąsiedztwa oraz macierzy sąsiedztwa.

a) **Lista sąsiedztwa:**
   Jeden z najpowszechniejszych sposobów reprezentacji grafu, często najefektywniejsza forma przechowywania informacji o grafie biorąc pod uwagę efektywność czasową i pamięciową. Opiera się na strukturze danych jaką jest lista, dla każdego wierzchołka grafu tworzona jest lista krawędzi wychodzących z tego wierzchołka.
b) **Macierz sąsiedztwa:**
   Jest to prostsza do zaimplementowania forma grafu, jednak często mniej opłacalna niż reprezentacja grafu w postaci listy sąsiedztwa. Opiera się na dwuwymiarowej tablicy, której indeksy odpowiadają wierzchołkom. Dana komórka macierzy trzyma wskaźnik do krawędzi, jeśli taka istnieje między danym zestawem wierzchołków, lub NULL gdy między wierzchołkami nie ma bezpośredniego połączenia.

3. Wybrany algorytm

W opisie zadania projektowego wymieniono dwa algorytmy, z których należało wybrać jeden i zbadać jego efektywność w zależności od charakterystyk grafu. Wybrany został algorytm Bellmana-Forda. Ma on podobne zastosowanie jak algorytm Dijkstry, ale różni się swoimi ograniczeniami i własnościami. Przede wszystkim, algorytm Bellmana-Forda ma gorszą złożoność czasową od algorytmu Dijkstry oraz wyklucza istnienie ścieżki o ujemnej wadze. W związku z tym, jego częstym zastosowaniem jest właśnie wyszukiwanie ścieżek o ujemnym koszcie, a nie samo szukanie najkrótszej drogi do poszczególnych wierzchołków.

4. Przebieg badań i uzyskane wyniki

Badanie efektywności algorytmu polegało na zmierzeniu czasu działania algorytmu dla zestawu 100 losowych grafów, oraz uśrednieniu go. Badano efektywność dla 4 gęstości grafu; 25%, 50%, 75% i 100% (graf pełny) oraz dla 5 różnych ilości wierzchołków; 10, 25, 50, 100, 150 (wybrano takie liczby, w związku z tym, że polecenie nie narzucało konkretnych ilości wierzchołków, a akurat te liczby pozwalały na uzyskanie najczytelniejszych wykresów). Uśredniony czas wykonania podany jest w **mikrosekundach**.

a) Ogólny spis wszystkich czasów wykonania, w zależności od reprezentacji grafu, ilości wierzchołków i gęstości:

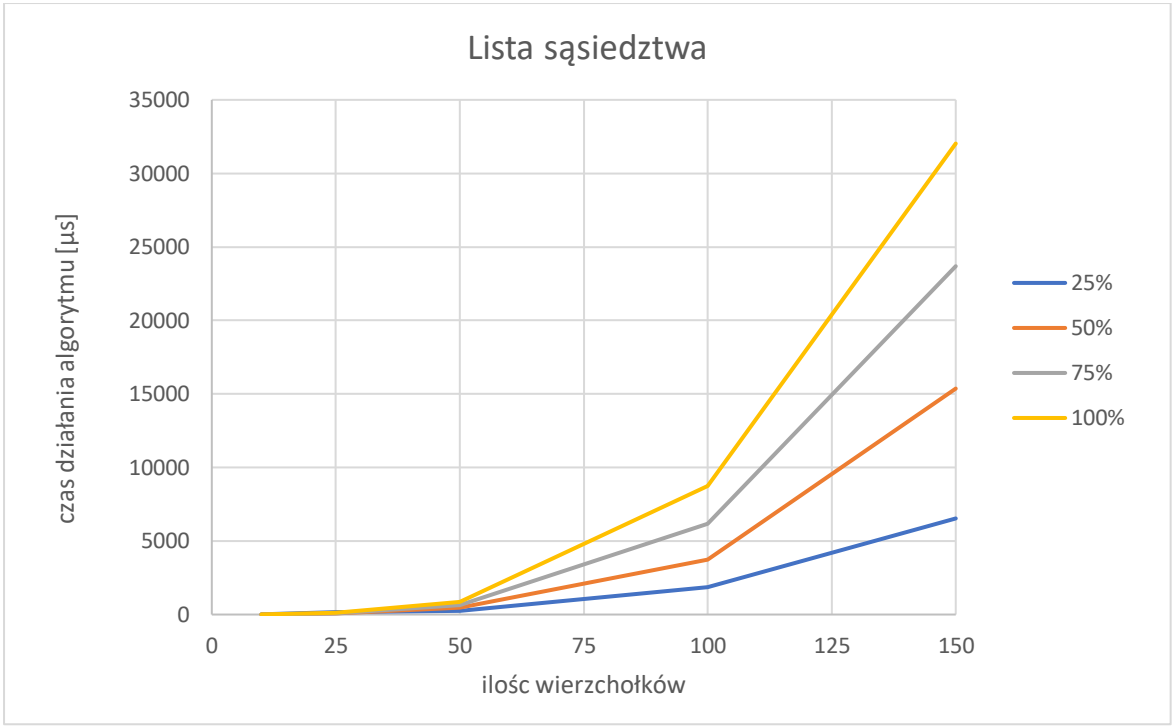
ilość wierzchołków

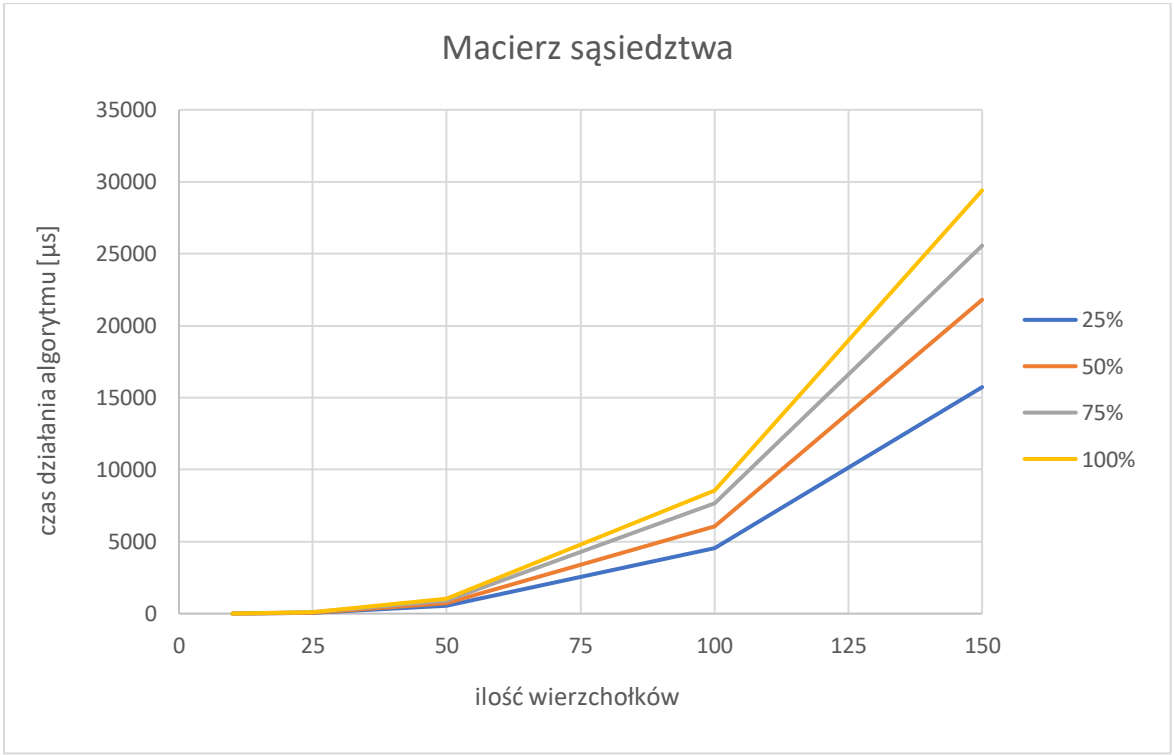
gęstość grafy	**Lista sąsiedztwa**	10	25	50	100	150
	25%	14	155	262	1853	6535
	50%	5	59	453	3750	15365
	75%	6	85	653	6193	23687
	100%	7	107	852	8762	32029

ilość wierzchołków

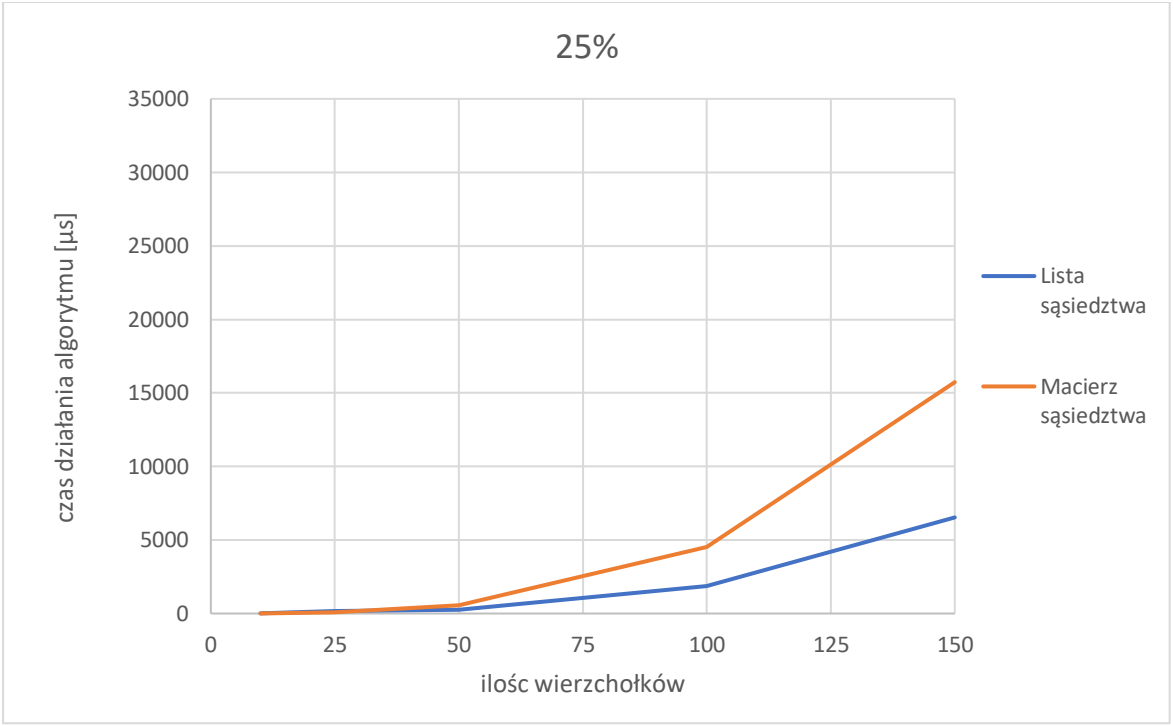
gęstość grafu	**Macierz sąsiedztwa**	10	25	50	100	150
	25%	4	66	561	4529	15733
	50%	5	80	735	6043	21802
	75%	5	95	897	7664	25562
	100%	6	106	1022	8538	29402

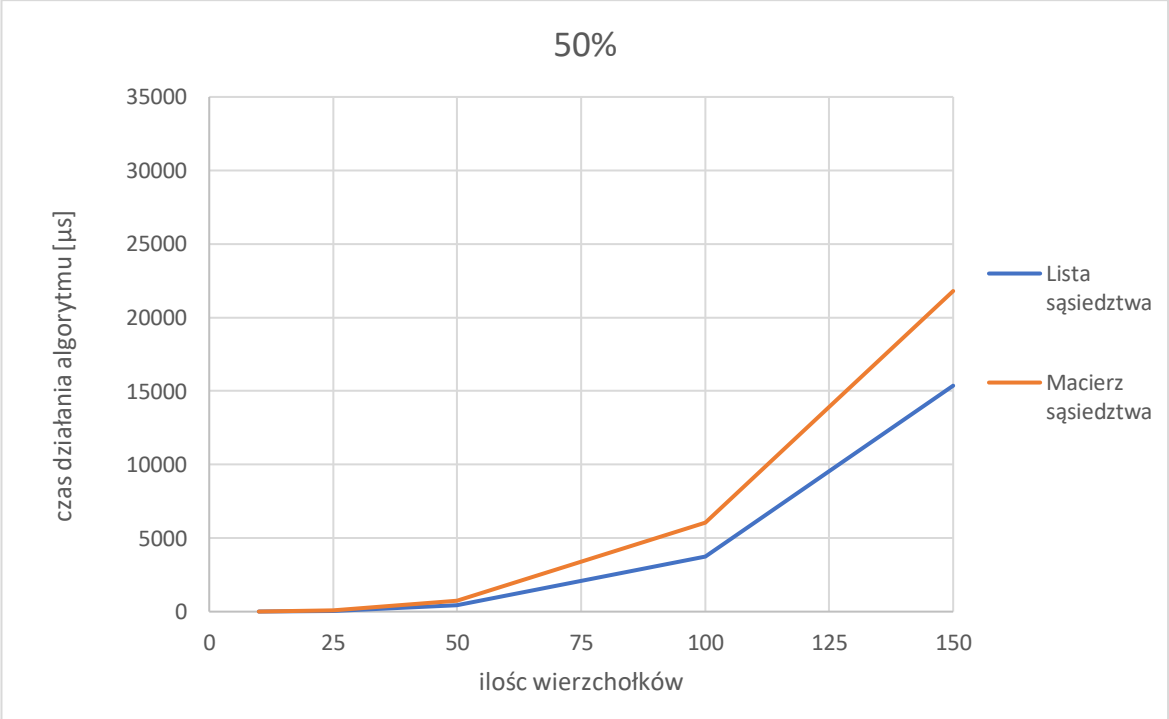
b) Zestawienie dwóch sposobów implementacji grafu, krzywe odpowiadają gęstościom grafu wymienionym w legendzie:

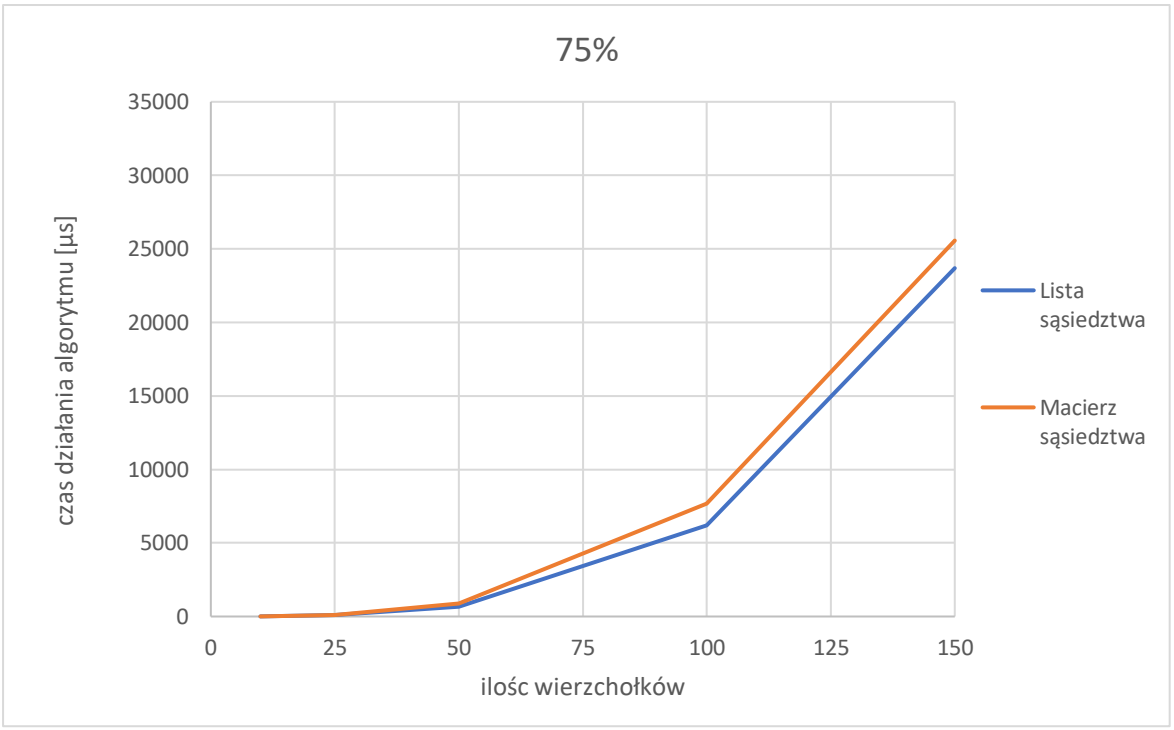


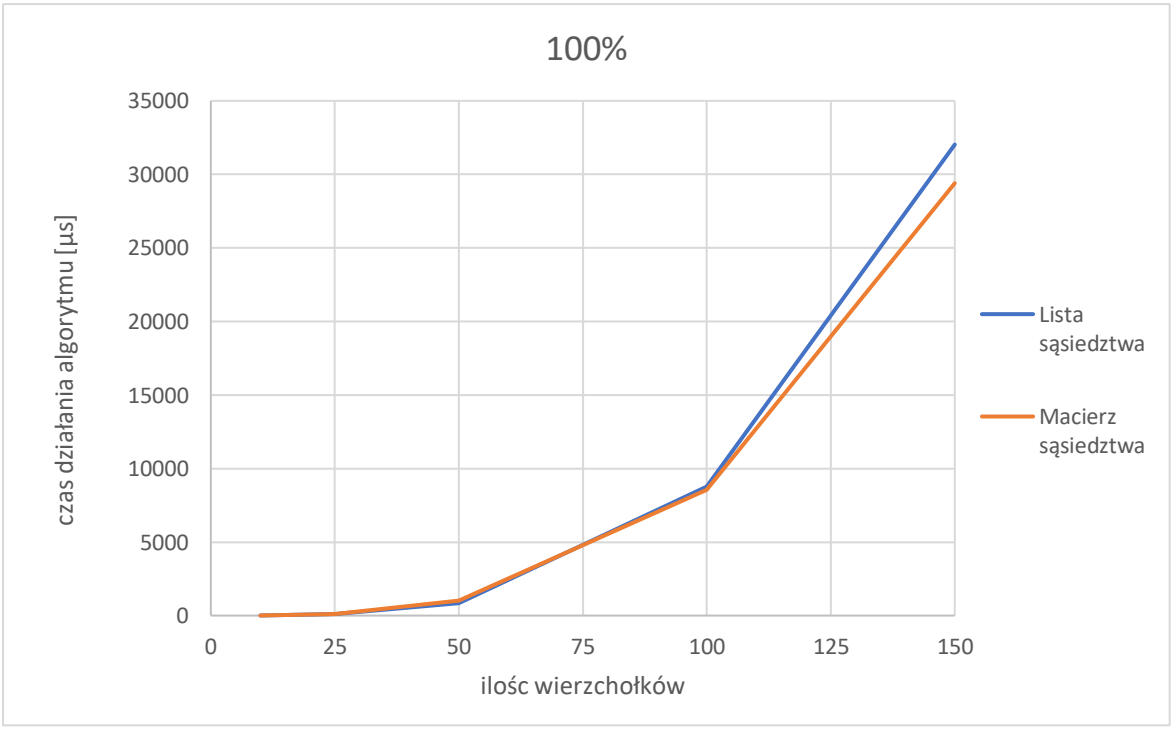


c) Zestawienie 4 gęstości grafu, krzywe odpowiadają danej implementacji grafu:






75%
35000
30000
25000
20000
15000
10000
5000
0
0
25
50
75
100
125
150
czas działania algorytmu [µs]
ilośc wierzchołków
Lista sąsiedztwa
Macierz sąsiedztwa


100%
35000
30000
25000
20000
15000
10000
5000
0
0
25
50
75
100
125
150
czas działania algorytmu [µs]
ilośc wierzchołków
Lista sąsiedztwa
Macierz sąsiedztwa

5. Wnioski i uwagi

- Wraz ze wzrostem gęstości grafu zaciera się różnica między efektywnością algorytmu dla listy i macierzy sąsiedztwa
- W związku z tym, że gęstość grafu wpływa na ilość krawędzi, czyli połączeń do sprawdzenia w trakcie działania algorytmu, w każdym z przypadków większa gęstość przekłada się na dłuższy czas pracy
- Pomimo nieuwzględnienia tych danych w projekcie, należy wspomnieć, że podczas testowania algorytmu dla reprezentacji macierzowej zużycie pamięci RAM rzeczywiście było znacząco wyższe niż dla reprezentacji w postaci listy sąsiedztwa

6. Literatura

- Data structures and algorithms in C++, Michael T. Goodrich

- https://pl.wikipedia.org/wiki/Reprezentacja_grafu
- http://algorytmy.ency.pl/tutorial/algorytm_bellmana_forda
- https://pl.wikipedia.org/wiki/Algorytm_Bellmana-Forda
- https://www.geeksforgeeks.org/graph-and-its-representations/
  https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm