Projektowanie algorytmów i metody sztucznej inteligencji
Projekt 2 - Grafy - sprawozdanie

Igor Zieliński 248944
Prowadzący: mgr inż. Marcin Ochma

Wtorek $15^{15} - 16^{55}$

1 Wstęp

Przedmiotem projektu było zaimplementowanie oraz badanie efektywności algorytmu Dijkstry dla dwóch reprezentacji grafu: za pomocą macierzy sąsiedztwa i listy sąsiedztwa.

2 Opis przebiegu eksperymentu

Sprawdzono poprawność implementacji porównując wyniki z algorytmem Bellmana-Forda. Wygenerowano po 100 testów dla każdego z rodzajów: graf N wierzchołków o gęstości D, gdzie N należało do zbioru {100, 500, 1000, 5000, 10000}, a D należało do zbioru {25%, 50%, 75%, 100%}. Następnie zmierzono czasy działania algorytmu dla obydwu reprezentacji grafu i uśredniono je.

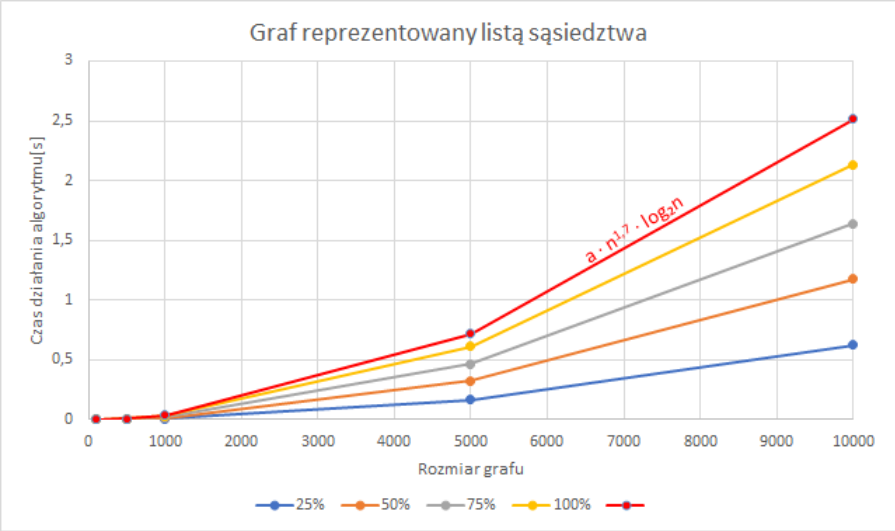
3 Wyniki eksperymentu

rozmiar \ gęstość	25%	50%	75%	100%
100	0,000304	0,000398	0,000494	0,000574
500	0,002506	0,004643	0,006357	0,007979
1000	0,008279	0,016241	0,022212	0,028952
5000	0,165734	0,325995	0,46593	0,613573
10000	0,624864	1,17445	1,637606	2,130871

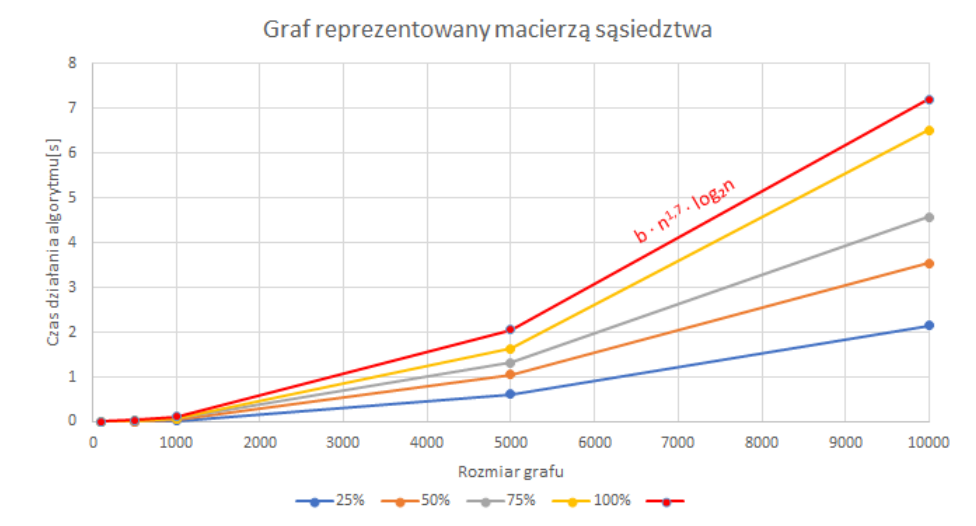
Rysunek 1: Tabela wyników uśrednionych dla Grafu z listą sąsiedztwa

rozmiar \ gęstość	25%	50%	75%	100%
100	0,000663	0,000942	0,001271	0,001404
500	0,0075	0,012107	0,017742	0,020108
1000	0,026082	0,04541	0,064107	0,074035
5000	0,61251	1,057388	1,327549	1,637055
10000	2,153306	3,540761	4,581333	6,515085

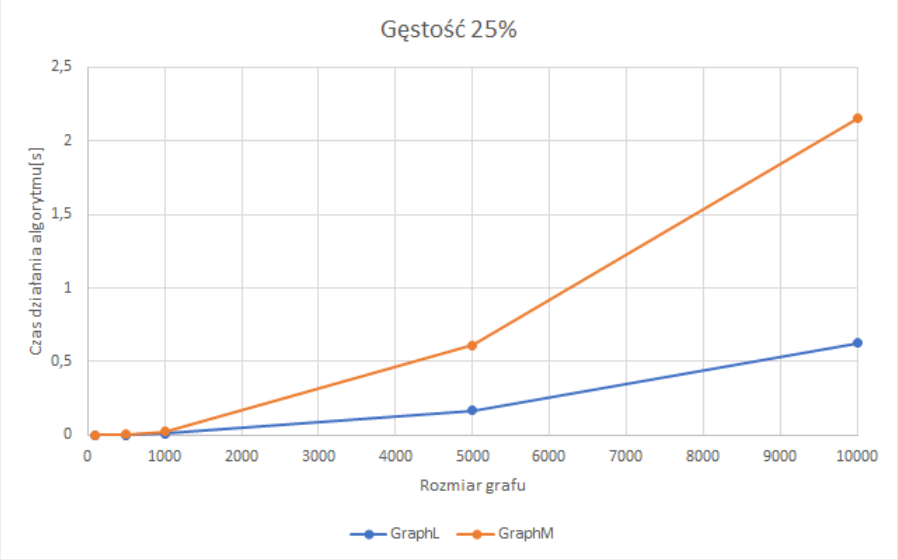
Rysunek 2: Tabela wyników uśrednionych dla Grafu z macierzą sąsiedztwa

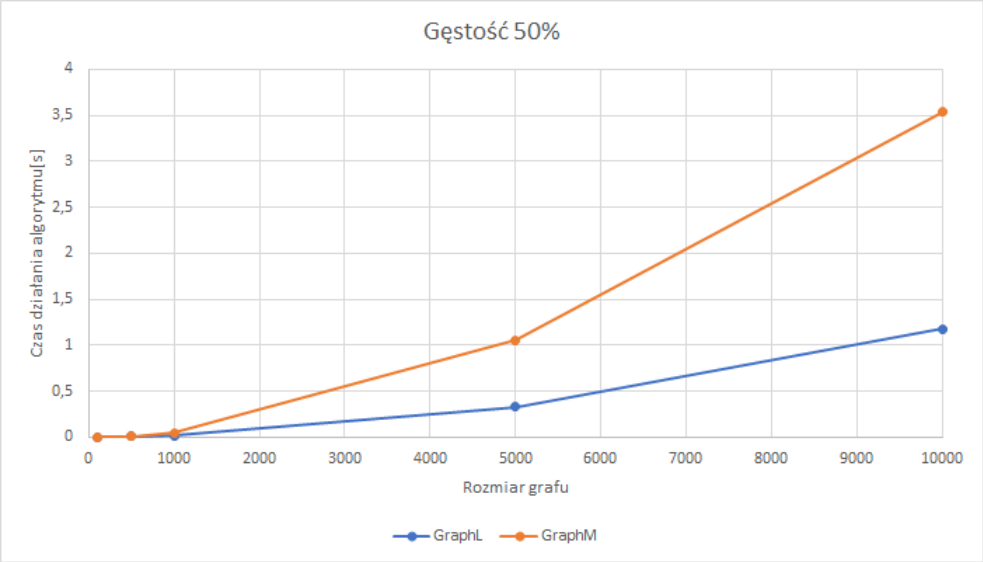


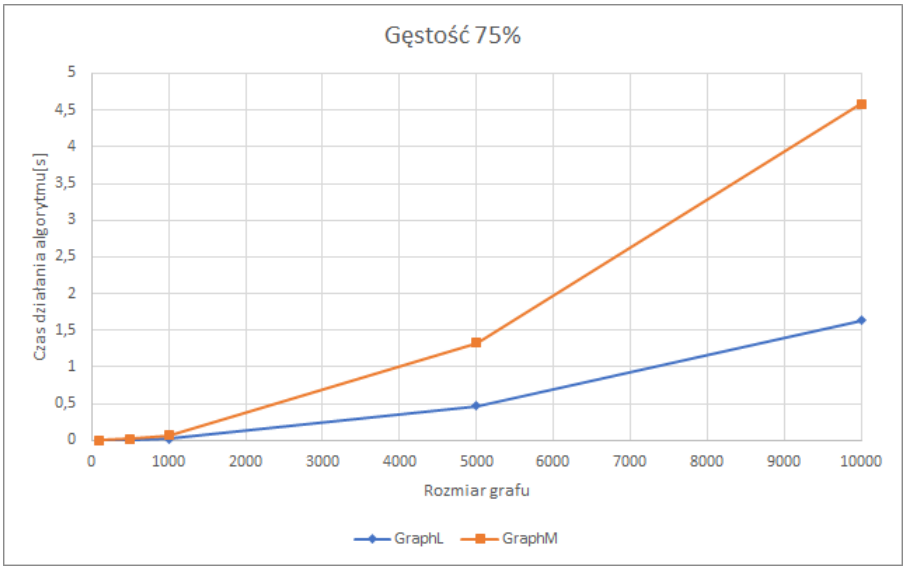
Rysunek 3: Stała $a = 3 \cdot 10^{-8}$

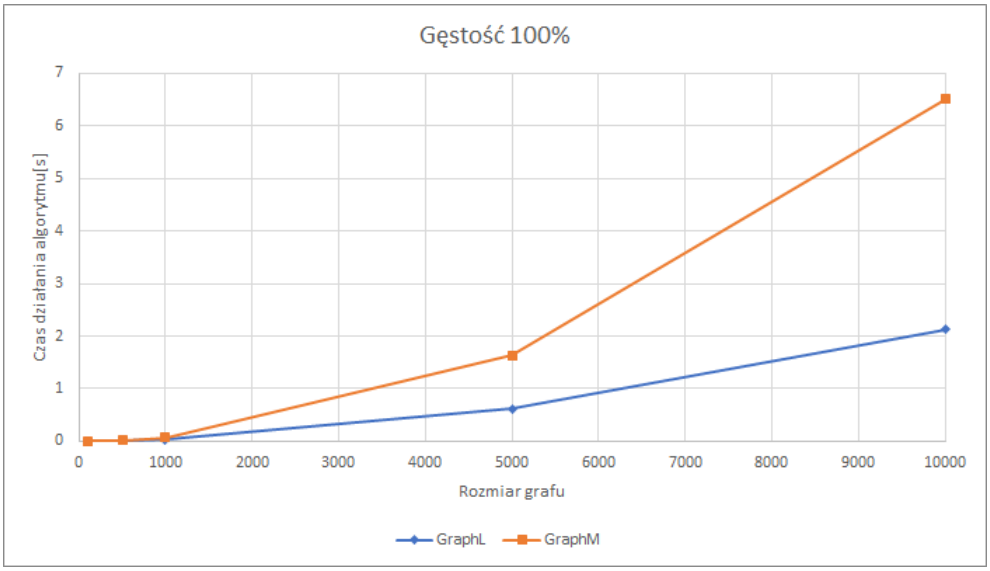


Rysunek 4: Stała $b = 8,6 \cdot 10^{-8}$




Gęstość 50%
Czas działania algorytmu[s]
Rozmiar grafu
GraphL
GraphM


Gęstość 75%
Czas działania algorytmu[s]
Rozmiar grafu
GraphL
GraphM


Gęstość 100%
Czas działania algorytmu[s]
Rozmiar grafu
GraphL
GraphM

4 Wnioski

1. Wraz ze wzrostem gęstości grafu wydłuża się czas działania algorytmu.

2. Ze złożoności algorytmu przedstawionej w źródle - $O((N + M) \cdot log_2 N)$ - można by wnioskować, że dla grafu pełnego wykres czasu działania algorytmu będzie przypominał $N^2 \cdot log_2 N$. Natomiast, jak pokazano na wykresach zbiorowych dla danych reprezentacji grafu, można ograniczyć taki wykres przez $N^{1,7} \cdot log_2 N$. Oznacza to, że pesymistyczna złożoność obliczeniowa jest przeszacowana i nieosiągalna dla nawet grafu pełnego.

3. Algorytm we wszystkich przypadkach działa wolniej dla macierzowej reprezentacji grafu. Wynika to z tego, że w algorytmie przeglądamy wszystkich sąsiadów wierzchołka v, która to operacja ma złożoność obliczeniową równą $O(n)$ dla macierzowej reprezentacji i $O(deg(v))$ dla listowej reprezentacji grafu. Dla garfu pełnego natomiast, różnica ta wynika z implementacji metody incidentEdges(v) w mojej realizacji. Dla macierzy należy utworzyć wektor sąsiadów, podczas gdy w listowej reprezentacji zwracana jest referencja.

5 Źródła

https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm