Krystian Matusiak 249460
Czwartek 9:00-11:00
Dr inż. Łukasz Jeleń

Projektowanie Algorytmów i Metody Sztucznej Inteligencji

Projekt 2 - Grafy

1. Opis projektu

Projekt opierał się na przetestowaniu działania jednego z algorytmów wyznaczającego najkrótszą drogę między wierzchołkiem startowym a resztą wierzchołków w grafie. Testowanym algorytmem był algorytm Dijkstry. Dla potrzeb tego algorytmu graf musiał być nieskierowany, ważony (wagi nieujemne).

Graf będzie reprezentowany przez listę oraz macierz sąsiedztwa. Dla każdej z reprezentacji przeprowadzono testy w zależności od gęstości :

- 25%
- 50%
- 75%
- 100%

oraz dla różnych ilości wierzchołków:

- 10
- 20
- 30
- 40
- 50

Ze względu na liczne “wycieki” pamięci, test zrealizowano na małej ilości wierzchołków oraz dla małej ilości powtórzeń, tj. dla 40 pomiarów, z których wyliczono średni czas trwania algorytmu.

Maksymalna możliwa ilość krawędzi w grafi wyliczana jest ze wzoru :

$$\frac{n(n-1)}{2}$$

gdzie n oznacza ilość wierzchołków.

2. Opis algorytmu

Algorytm Dijkstry stosuje się w grafach z wagami nieujemnymi. Znajduje on najkrótszą drogę między wierzchołkami grafu tworząc “chmurę” ze wszystkich wierzchołków. Pesymistyczna złożoność obliczeniowa olgorytmu wynosi:

- O(E + VlogV)

gdzie V oznacza liczbę wierzchołków, natomiast E liczbę krawędzi.

3. Budowa programu

Program wykonujący test efektywności algorytmu napisano obiektowo. Stworzono klasę Wierzcholek oraz Krawedz, które będą przechowywane w napisanej obiektowo liście dwukierunkowej. Utworzono również klasy Macierz_Sas oraz Lista_Sas, tj. listę oraz macierz sąsiedztwa, które reprezentowały dany graf. Utworzono także klasę działającą jako kolejka priorytetowa, która przechowywała elementy klasy Para, przechowująca wierzchołek oraz klucz o danym priorytecie. Algorytm istnieje w postaci funkcji zwracającej czas trwania algorytmu.

Wyniki pomiaru czasu, jak również ścieżki między wierzchołkami, zapisano w kilku plikach, gdyż ze względu na wykorzystywaną pamięć nie można było w jednym procesie przeprowadzić testów.

4. Wyniki testów

Tabela 1. Wyniki testów trwania algorytmu dla listy sąsiedztwa.

Ilosc wierzchołków	Lista 25% Czas [ms]	Lista 50% Czas [ms]	Lista 75% Czas [ms]	Lista 100% Czas [ms]
10	0,009	0,010	0,011	0,014
20	0,032	0,058	0,080	0,118
30	0,125	0,232	0,358	0,569
40	0,314	0,559	0,916	1,519
50	0,744	1,445	2,318	3,952

Wykres 1. Zależność czasu trwania algorytmu od ilości wierzchołków dla listy sąsiedztwa.

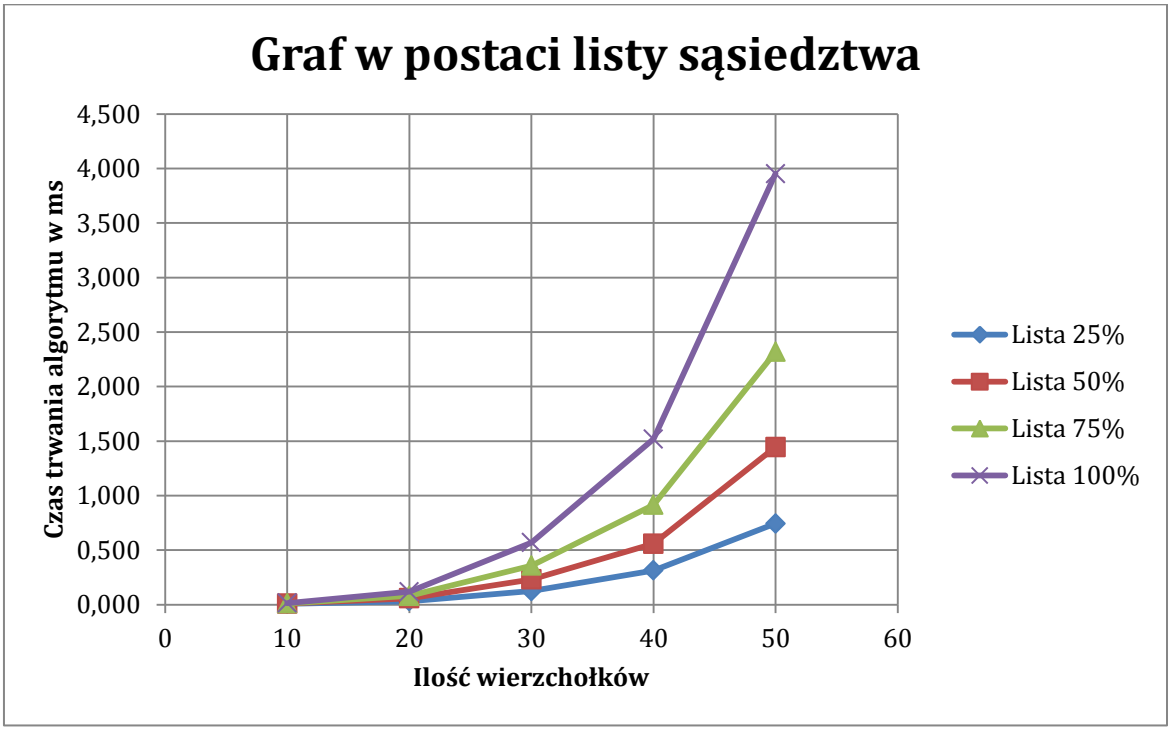
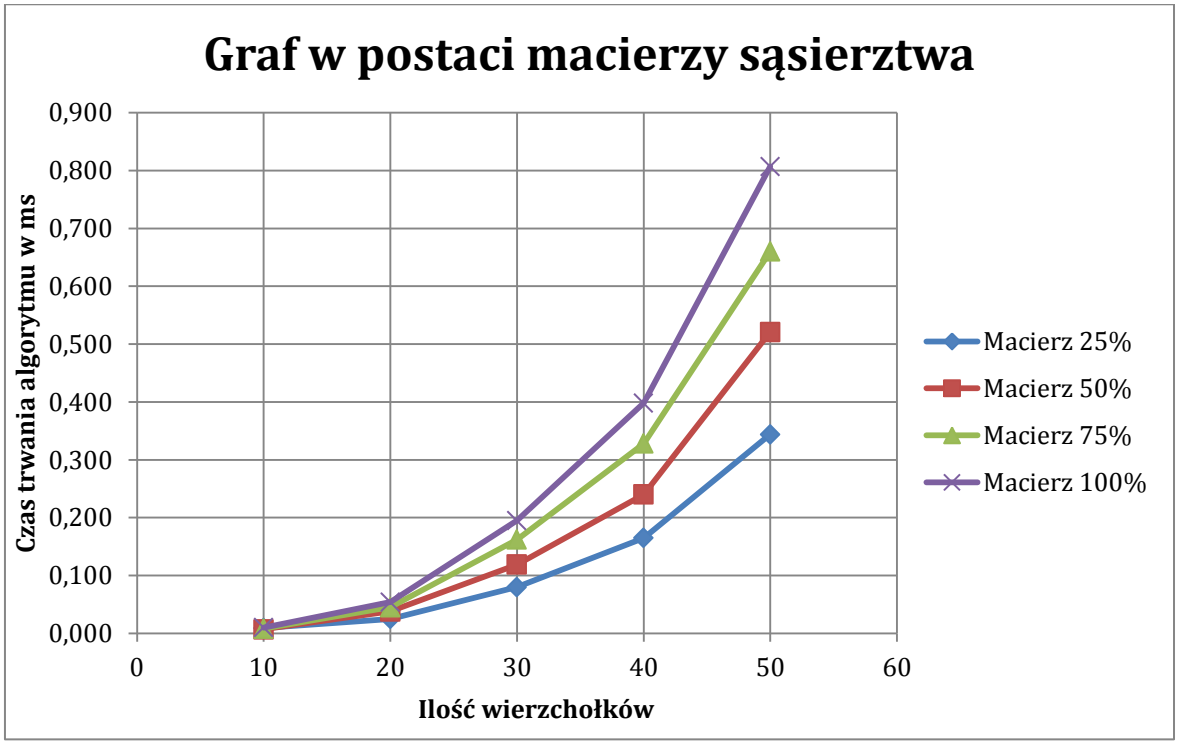


Tabela 1. Wyniki trwania algorytmu dla macierzy sąsiedztwa.

Ilosc wierzchołków	Macierz 25% Czas [ms]	Macierz 50% Czas [ms]	Macierz 75% Czas [ms]	Macierz 100% Czas [ms]
10	0,009	0,007	0,008	0,010
20	0,025	0,038	0,045	0,054
30	0,080	0,119	0,162	0,195
40	0,165	0,240	0,328	0,398
50	0,344	0,521	0,660	0,807

Wykres 2. Zależność czasu trwania algorytmu od ilości wierzchołków dla macierzy sąsiedztwa.



5. Wnioski

Analizując powyższe wykresy można zauważyć, iż w przypadku grafu, reprezentowanego przez macierz sąsiedztwa, czas trwania algorytmu niewiele zależy od gęstości grafu, w przeciwieństwie do listy sąsiedztwa. Im większa gęstość grafu tym dłuższy okres wyszukiwania najkrótszej drogi za pomocą algorytmu Dijkstry.

Ponadto zauważalna jest różnica w czasie trwania algorytmu. Dla macierzy sąsiedztwa przy 50 wierzchołkach oraz gęstości 100% pomiar trwał 0,807ms podczas gdy dla listy sąsiedztwa, przy tej samej gęstości oraz ilości wierzchołków, pomiar dochodził do 4 sekund.

6. Bibliografia

https://pl.wikipedia.org/wiki/Algorytm_Dijkstry
https://eduinf.waw.pl/inf/alg/001_search/0124.php