Wtorki 16:50
Grupa I3
Kierunek Informatyka
Wydział Informatyki
Politechnika Poznańska

Algorytmy i struktury danych

Sprawozdanie z zadania w zespołach nr. 2
prowadząca: dr hab. inż. Małgorzata Sterna, prof PP

Algorytmy Grafowe

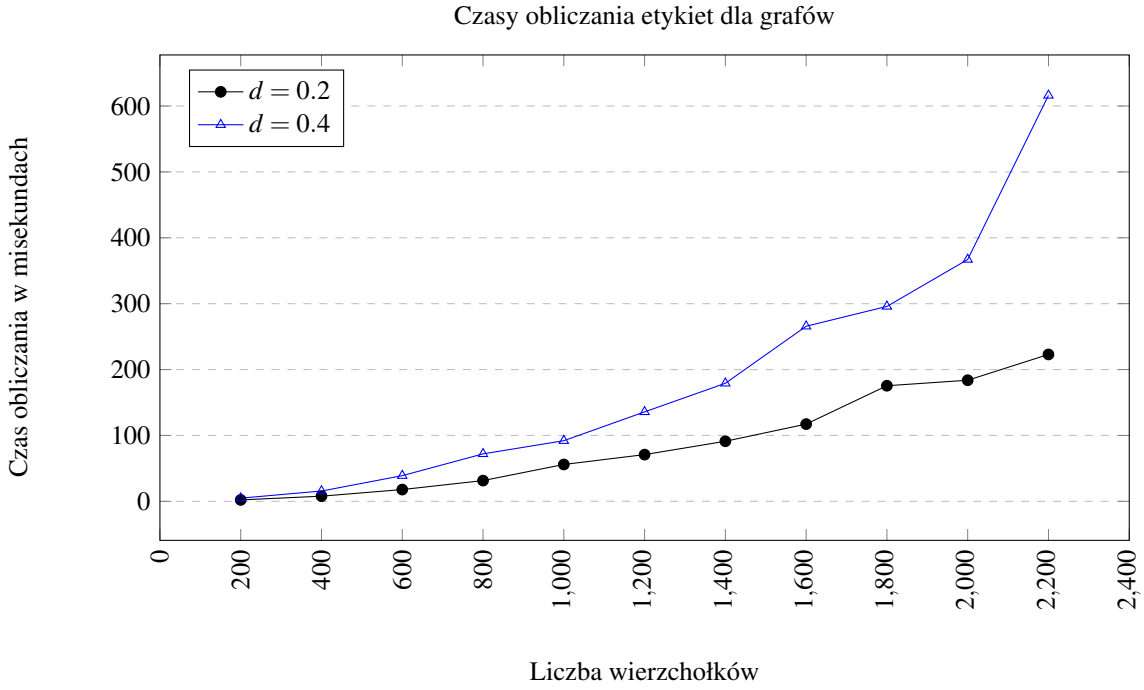
autorzy:
Piotr Więtczak nr indeksu 132339
Tomasz Chudziak nr indeksu 136691

7 maja 2018

1 Opis implementacji

Do implementacji wybranych struktur danych uzyliśmy języka C++, a do pomiarów czasu klasy std :: chrono :: high_resolution_clock z biblioteki < chrono >.

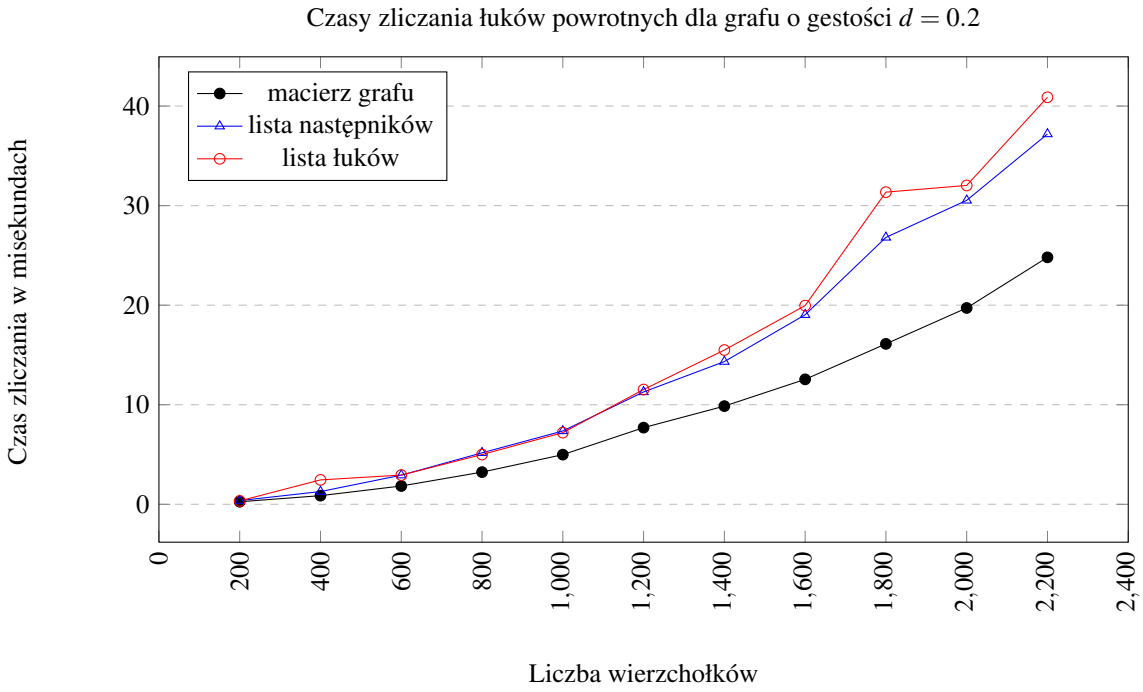
2 Obliczanie etykiet

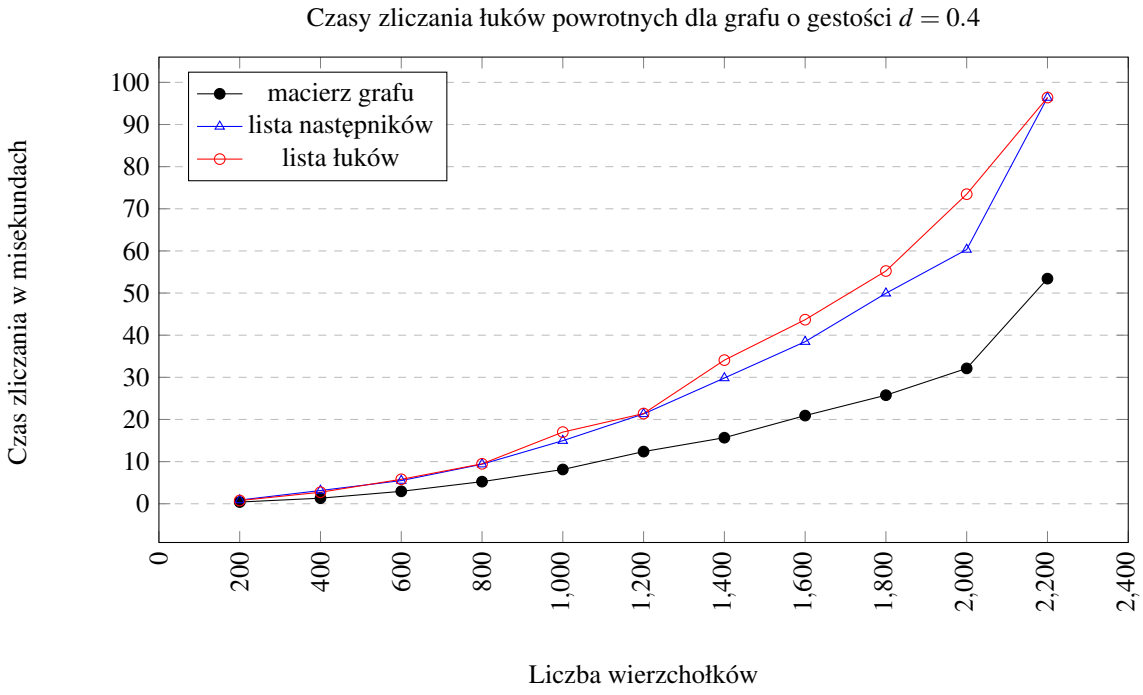


Metoda sortowania topologicznego opiera się na algorytmie DFS – przeszukiwania w głąb. Algorytm ten polega na odwiedzeniu wszystkich wierzchołków. W pierwszej kolejności wybiera on wierzchołki o najmniejszym możliwym numerze względem całej ścieżki, jaką już przebył. Jeżeli nie ma już dostępnych wierzchołków, to kończy ścieżkę. Następnie ze wszystkich dostępnych wierzchołków wybiera ten najmniejszy nieodwiedzony i powtarza algorytm. Kończy się on natomiast , gdy wszystkie wierzchołki zostały odwiedzone. Złożoność obliczeniowa w tym przypadku to O(n+m). Efektywność algorytmu DFS zależy od reprezentacji grafu. Najczęściej wykonywaną operacją przez ten algorytm jest przeszukiwanie listy poprzedników w celu znalezienia kolejnego wierzchołka. Wynika z tego, że najkorzystniejszą strukturą będzie lista następników, następnie macierz sąsiedztwa, dla których złożoność będzie wynosiła O(n). Mniej do tego algorytmu nadaje się lista łuków i lista poprzedników, dla których złożoność wynosi kolejno O(m) i O(n+m). Najmniej korzystną strukturą jest macierz incydencji, dla której ta operacja może trwać O(n*m). Duzy wpływ na czas trwania tej operacji ma równie gęstość grafu. Im ten jest gęstszy, tym algorytm musi sprawdzić większą ilość wierzchołków, czego konsekwencją jest dłuższy czas pracy.

3 Liczba łuków powrotnych

4 Czasy zliczania łuków powrotnych





5 Porównania poznanych reprezentacji grafu (macierzy sąsiedztwa, listy następników, listy poprzedników, listy łuków, macierzy incydencji)

5.1 Złożoność pamięciowa.

Ze wszystkich poznanych struktur najmniej miejsca zajmuje lista łuków O(m), następnie lista poprzedników oraz następników O(n+m). Większe zapotrzebowanie na ten zasób ma macierz sąsiedztwa $O(n^2)$. Najgorzej wypada macierz incydencji O(n*m), pomimo tak dużych wymagań ma ona jednak dość dużą zaletę, tylko ta forma będzie potrafiła w pełni zaprezentować hipergraf. Jednakże w przypadku grafów rzadkich będzie to duża wada.

5.2 Test łuku.

Najszybszą pod tym względem okazuje się macierz grafu O(1), dzięki sprawdzaniu tylko jednej wartości w strukturze. Drugą pod tym względem jest lista łuków i macierz incydencji O(m). Słabiej pod tym względem wypada lista następników i poprzedników O(n).

5.3 Sprawdzanie następników.

Najbardziej wydajną okazuje się lista następników i macierz grafu O(n). Warto by dodać, że dla tylko dla przypadku pesymistycznego czas tych dwóch struktur jest taki sam, w każdym innym lista następników jest szybsza. Wolniejsze okazuje się lista łuków i lista poprzedników z kolejno O(m) i O(n+m). Najwolniejsza okazuje się macierz incydencji O(n*m).

5.4 Sprawdzanie poprzedników.

Najlepszą reprezentacją do tego testu okazuje się lista poprzedników O(n). Drugą co do wydajności jest macierz sąsiedztwa. Wolniejsza okazuje się lista łuków O(m) oraz lista następników O(n+m). Po raz kolejny najmniej wydają okazuje się macierz incydencji O(m*n).

5.5 Zbiór łuków.

Optymalną strukturom do tego testu okazuje się lista łuków O(m). Druga co do wydajności jest lista następników i poprzedników O(m+n). Bardzo słabo wypada macierz sąsiedztwa $O(n^2)$. Najgorzej jednak z tym zadaniem radzi sobie macierz incydencji O(n*m).

Spis treści

1 Opis implementacji **1**

2 Obliczanie etykiet **1**

3 Liczba łuków powrotnych **2**

4 Czasy zliczania łuków powrotnych **2**

5 Porównania poznanych reprezentacji grafu (macierzy sąsiedztwa, listy następników, listy poprzedników, listy łuków, macierzy incydencji) **3**

5.1 Złożoność pamięciowa. 3

5.2 Test łuku. 3

5.3 Sprawdzanie następników. 3

5.4 Sprawdzanie poprzedników. 3

5.5 Zbiór łuków. 3