

Piotr Karoń, 241 626

Wrocław, dn. 29.05.2019

Zadanie projektowe 2.

Badanie efektywności algorytmów grafowych w zależności od rozmiaru instancji oraz sposobu reprezentacji grafu w pamięci komputera.

PROWADZĄCY:

dr Jarosław Mierzwa

Spis treści

1	Założenia projektowe	3
1.1	Cel	3
1.2	Technologie	3
1.3	Przebieg eksperymentu	3
2	Złożoności czasowe algorytmów	3
2.1	Algorytm Kruskala	3
2.2	Algorytm Prima	3
2.3	Algorytm Dijkstry	4
3	Wyniki	4
3.1	Wykresy	4
3.2	Tabele	6
4	Podsumowanie	7
	Bibliografia	9

1 Założenia projektowe

1.1 Cel

Celem projektu jest zbadanie efektywności algorytmów Kruskala, Prima, Dijkstry i Bellmana-Forda w zależności od sposobu reprezentacji grafu i wielkości instancji.

1.2 Technologie

Do implementacji wymienionych struktur użyto języka *Kotlin* w wersji *Native*, która jest kompilowana do kodu maszynowego danej platformy.

1.3 Przebieg eksperymentu

Badania przeprowadzone zostały dla wierzchołków w liczbie: 10, 100, 1000, 10000, 30000, dla każdej liczby w gęstościach 25%, 50%, 75%, 99% osobno dla reprezentacji macierzowej i listowej. Każdy test wykonano 50 razy, a czas uśredniono z wszystkich prób.

2 Złożoności czasowe algorytmów

Przy tworzeniu poniższych opisów korzystano z Wprowadzenia do algorytmów autorstwa T. Cormen, C. Leiserson, R. Rivest [1], materiałów udostępnionych na stronie dr Tomasza Kapłona [3] oraz stronie I LO w Tarnowie [5]

2.1 Algorytm Kruskala

Czas działania algorytmu Kruskala dla grafu $G = (V, E)$ zależy od sposobu implementacji struktury zbiorów rozłącznych. W tym projekcie wykorzystano implementację lasu zbiorów rozłącznych z łączeniem według rangi i z kompresją ścieżek. Całkowity czas działania algorytmu wynosi $O(E \log_2 E)$.

2.2 Algorytm Prima

Szybkość działania algorytmu Prima zależy od sposobu implementacji kolejki priorytetowej przechowującej koszty dojścia do danego wierzchołka. W tym projekcie wykorzystano kopiec stworzony w projekcie nr 1, dzięki czemu otrzymano złożoność $O(E \log_2 V)$.

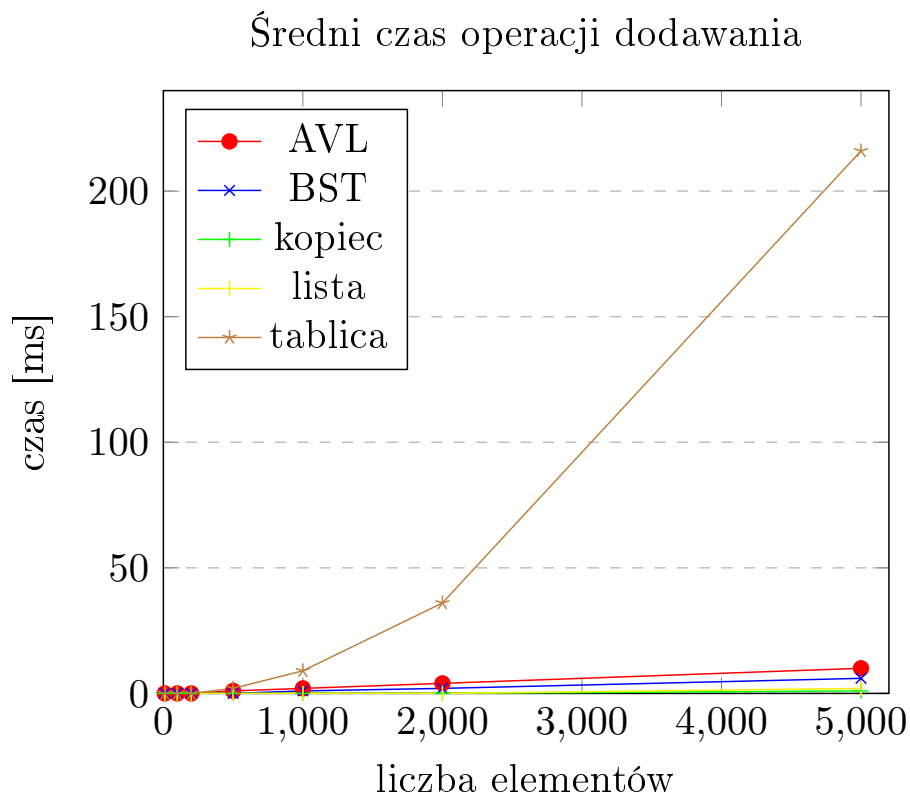
2.3 Algorytm Dijkstry

Algorytm Dijkstry działa poprawnie tylko jeśli w grafie nie znajdują się ścieżki o ujemnej wadze. Szybkość algorytmu zależy od sposobu implementacji kolejki priorytetowej przechowującej koszty dojścia do danego wierzchołka. W tym projekcie wykorzystano kopiec stworzony w projekcie nr 1, dzięki czemu otrzymano złożoność $O(E \log_2 V)$.

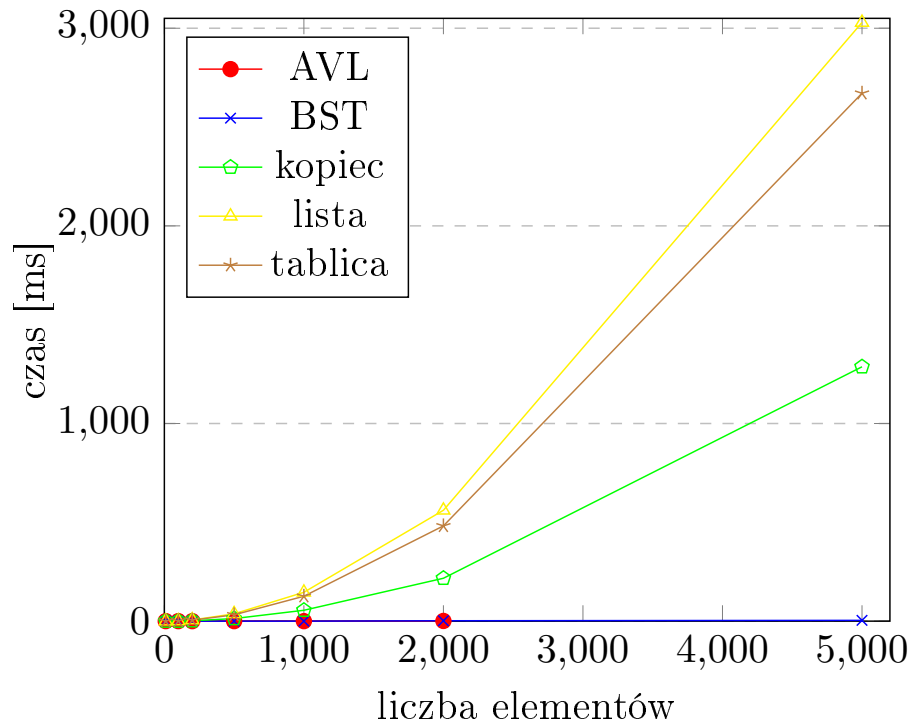
2.4 Algorytm Bellmana-Forda

3 Wyniki

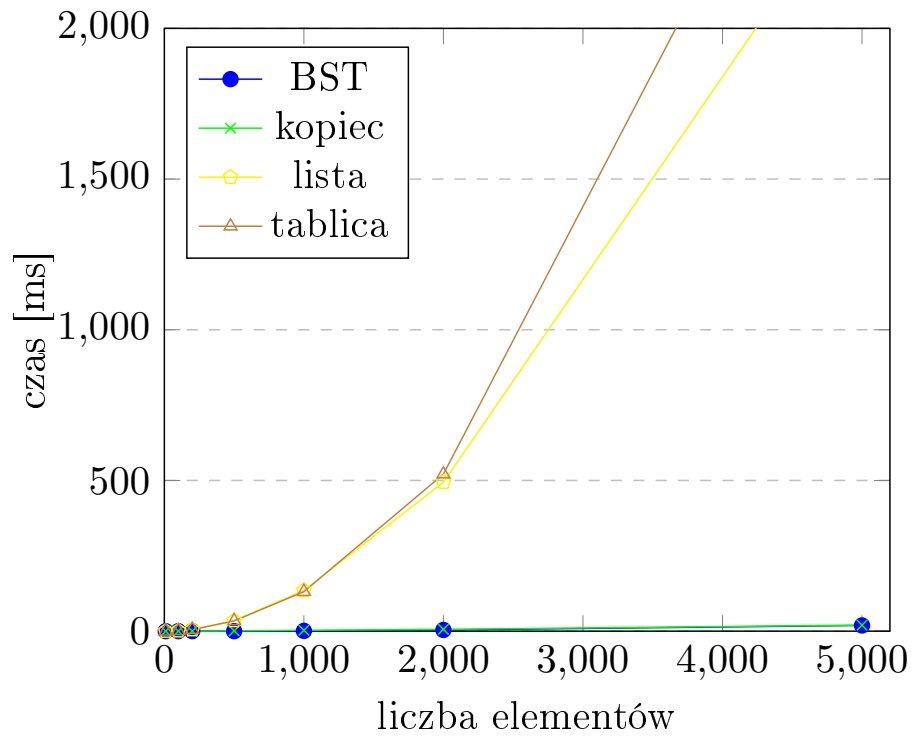
3.1 Wykresy

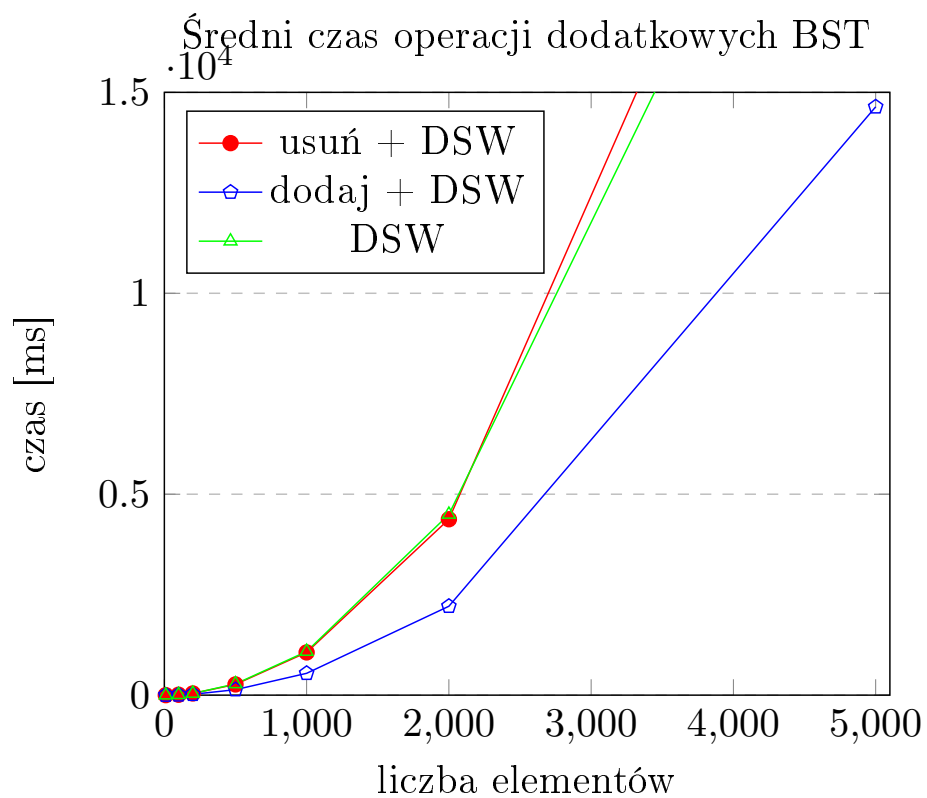


Średni czas operacji szukania



Średni czas operacji usuwania





3.2 Tabele

Tablica 1: Wyniki operacji dodawania w milisekundach

*	Tablica	Lista	Kopiec	Bst	Avl
10	0	0	0	0	0
100	0	0	0	0	0
200	0	0	0	0	0
500	2	0	0	0	1
1000	9	0	0	1	2
2000	36	0	0	2	4
5000	216	2	1	6	10

Tablica 2: Wyniki operacji szukania w milisekundach

*	Tablica	Lista	Kopiec	Bst	Avl
10	0	0	0	0	0
100	1	1	0	0	0
200	5	6	0	0	0
500	32	37	0	0	0
1000	126	146	0	1	0
2000	482	561	0	2	1
5000	2671	3029	1	6	2

Tablica 3: Wyniki operacji usuwania w milisekundach

*	Tablica	Lista	Kopiec	Bst	Avl
10	0	0	0	0	*
100	1	1	0	0	*
200	5	5	0	0	*
500	34	35	1	0	*
1000	131	135	3	1	*
2000	521	494	6	4	*
5000	3185	2512	20	19	*

Tablica 4: Wyniki operacji specjalnych BST

	Równoważenie	Usuwanie + DSW	Dodawanie + DSW
10	0	0	0
100	10	10	5
200	42	42	21
500	279	267	136
1000	1086	1063	543
2000	4497	4377	2212
5000	28523	26293	15005

4 Podsumowanie

Zaimplementowane algorytmy nie są optymalne. W większości jednak założona złożoność obliczeniowa sprawdziła się. Najbardziej obciążające były operacje na drzewie BST z wy-

korzystaniem algorytmu DSW. Równoważenie drzewa po każdym wstawieniu węzła nie jest dobrym pomysłem. Można to robić co kilka, kilkanaście wstawień – takie niezrównoważenie nie wpłynie bardziej na złożoność w przypadku innych operacji.

Bibliografia

- [1] T. Cormen, C. Leiserson, R. Rivest *Wprowadzenie do algorytmów*, Wydawnictwa Naukowo-Techniczne Warszawa, Wyd. IV, 2004
- [2] [1] str. 304.
- [3] `tomasz.kaplon.staff.iiar.pwr.wroc.pl/`, strona dr Tomasza Kapłona
- [4] `kotlinlang.org/docs/reference/native-overview.html`, dokumentacja języka Kotlin/Native
- [5] `eduinf.waw.pl`, materiały na stronie I LO w Tarnowie

