Wrocław 24.04.19 r.

Struktury danych i złożoność obliczeniowa

Wydział Elektroniki
Kierunek: Informatyka
Nr albumu: 241390
Imię i Nazwisko: Paweł Parczyk
Grupa zajęciowa: Wtorek TP 13-15
Kod grupy: E05-00d
Prowadzący: dr inż. Zbigniew Buchalski

Temat

Badanie efektywności operacji dodawania, usuwania oraz wyszukiwania elementów w rożnych strukturach danych

Ocena:

Data: 30.04.19 r.

Sprawozdanie z przebiegu eksperymentu

Wstęp: Projekt zakładał napisanie czterech struktur danych. Są nimi tablica dynamiczna, lista, kopiec i drzewo czerwono-czarne, a następnie sprawdzenie czasów działania poszczególnych funkcji na zaimplementowanych strukturach danych. Na tablicy i liście należało sprawdzić czasy dla operacji dodawania, usuwania i wyszukania elementów. Usuwanie i dodawanie były rozdzielone na 3 różne procedury: początku, z końca i ze środka struktury. Na kopcu i drzewie należało przeprowadzić funkcje dodawania na początek i wyszukiwanie elementu.

1. **Tablica dynamiczna:**
   a) Dodawanie elementu: złożoność tej operacji wynosi O(1), gdy tablica może zostać rozszerzona bez potrzeby realokacji pamięci. Zrealizowana tablica miała zajmować jak najmniej miejsca więc za każdym dodaniem trzeba realokować pamięć co zmienia klasę złożoności do O(n).
   b) Usuwanie elementu: złożoność wynosi O(n) gdyż trzeba przesunąć wszystkie elementy.
   c) Wyszukanie elementu: złożoność wynosi O(n) gdyż trzeba przeiterować po wszystkich elementach**.**
2. **Lista jednokierunkowa:**
   a) Dodawanie elementu: złożoność klasy O(1) gdy wstawiany element ma się znaleźć na początku, w przeciwnym wypadu złożoność wynosi O(n) ponieważ trzeba przeiterować w poszukiwaniu właściwej pozycji, skrajnym przypadkiem wstawiania w środku jest wstawienie elementu na końcu.
   b) Usuwanie elementu: podobnie jak przy dodawaniu, usuwanie z początku ma złożoność O(1) a każde inne złożoność O(n).
   c) Wyszukiwanie elementu: O(n) wyszukanie oznacza przejście listy aż do szukanego elementu.
3. **Kopiec zaimplementowany z wykorzystaniem tablicy**
   a) Dodawanie elementu: złożoność O(n log n). Dodanie nowej wartości wymaga sprawdzenia warunku kopca
   b) Usunięcie elementu: złożoność O(n log n).
   c) Wyszukanie elementu: O(n) w przypadku implementacji na tablicy wyszukiwać można tak samo jak w tablicy
4. **Drzewo czerwono-czarne**
   a) Dodawanie elementu: O(h) zależy od głębokości drzewa.
   b) Usuwanie elementu: podobnie jak przy dodawaniu złożoność obliczeniowa usuwanie wynosi O(h) zależy od głębokości drzewa.
   c) Wyszukiwanie elementu: w drzewie zrównoważonym jakim jest drzewo czerwono-czarne wyszukiwanie ma złożoność O(log(2) n), ponieważ zależy od głębokości drzewa, które jest możliwe najmniejsze.

Plan eksperymentu: Eksperyment składał się z 6 testów z populacjami różnej wielkości. W każdym teście zostało wygenerowanych 100 populacji o jednakowej wielkości, następnie stworzono nowe struktury przeprowadzono test wyszukiwania i usunięto elementy z struktur.

Otrzymane wyniki czasowe zostały uśrednione w celu usunięcie błędów losowych.
Do przeprowadzenia eksperymentu zostały wybrane populacje o wielkości 100, 200, 500, 1000, 2000 i 5000 elementów. Liczby te zostały dobrane w oparciu o możliwości maszyny, na której były przeprowadzane testy.

Program testujący został napisany w języku c++ z wykorzystaniem technik programowania obiektowego, następnie został skompilowany z użyciem kompilatora g++. Testy zostały przeprowadzone na komputerze zarządzanym przez system *Linux Ubuntu 16.04*. Do pomiaru czasów realizacji funkcji została użyta linuxowa funkcja *gettimeofday z biblioteki <sys/time.h>*, której rozdzielczość wynosi jedna milisekunda.

Eksperyment

Poniżej zaprezentowane zostały czasy wykonania poszczególnych operacji na strukturach. Pomiar każdej z funkcji składa się z dwóch tabel. Pierwsza prezentuje sumaryczne czasy poszczególnych operacji. Tzn: komórka w pierwszej tabeli [dodaj (koniec) x 100] zawiera informację ile trwało dodanie 100 elementów na koniec tablicy dynamicznej. Druga tabela zawiera średni czas pojedynczego wykonania operacji tzn: komórka [dodaj (koniec) x 100] zawiera informację ile średnio trwało dodanie elementu do tablicy dynamicznej.

Tablica dynamiczna:

Sumaryczne czasy operacji dla całych struktur							
ilość danych	Dodaj [micro s] (koniec)	Usuń [micro s] (koniec)	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Dodaj [micro s] (wybrane miejsce)	Usuń [micro s] (wybrane miejsce)	Wyszukaj [micro s]
100	931	880	890	953	991	967	15
200	1638	1614	1626	1630	1747	1646	16
500	6184	6304	6367	6352	7024	6462	26
1000	23102	23859	23415	23202	25428	23077	48
2000	88603	89787	88856	88865	97441	88876	93
5000	552975	559642	553718	549692	597396	553414	237

Średni czas realizacji pojedynczej funkcji dla przebiegów w różnych populacjach							
ilość danych	Dodaj [micro s] (koniec)	Usuń [micro s] (koniec)	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Dodaj [micro s] (wybrane miejsce)	Usuń [micro s] (wybrane miejsce)	Wyszukaj [micro s]
100	9,31	8,8	8,9	9,53	9,91	9,67	0,15
200	8,19	8,07	8,13	8,15	8,735	8,23	0,08
500	12,368	12,608	12,734	12,704	14,048	12,924	0,052
1000	23,102	23,859	23,415	23,202	25,428	23,077	0,048
2000	44,3015	44,8935	44,428	44,4325	48,7205	44,438	0,0465
5000	110,595	111,9284	110,7436	109,9384	119,4792	110,6828	0,0474

Lista jednokierunkowa:

Sumaryczne czasy operacji dla całych struktur							
ilość danych	Dodaj [micro s] (koniec)	Usuń [micro s] (koniec)	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Dodaj [micro s] (wybrane miejsce)	Usuń [micro s] (wybrane miejsce)	Wyszukaj [micro s]
100	331	282	31	34	215	140	7
200	1144	980	54	59	745	457	12
500	6847	5727	132	148	4480	2493	37
1000	29476	26425	264	307	18504	9553	65
2000	153140	150405	515	657	99717	54144	129
5000	1125051	1113327	1332	1715	749940	473738	321

Średni czas realizacji pojedynczej funkcji dla przebiegów w różnych populacjach							
ilość danych	Dodaj [micro s] (koniec)	Usuń [micro s] (koniec)	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Dodaj [micro s] (wybrane miejsce)	Usuń [micro s] (wybrane miejsce)	Wyszukaj [micro s]
100	3,31	2,82	0,31	0,34	2,15	1,4	0,07
200	5,72	4,9	0,27	0,295	3,725	2,285	0,06
500	13,694	11,454	0,264	0,296	8,96	4,986	0,074
1000	29,476	26,425	0,264	0,307	18,504	9,553	0,065
2000	76,57	75,2025	0,2575	0,3285	49,8585	27,072	0,0645
5000	225,0102	222,6654	0,2664	0,343	149,988	94,7476	0,0642

Kopiec:

Sumaryczne czasy operacji dla całych struktur			
ilość danych	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Wyszukaj [micro s]
100	230	1920	451
200	460	6714	1023
500	1166	40615	3015
1000	2416	169823	6824
2000	4918	702661	15050
5000	13513	4619594	44203

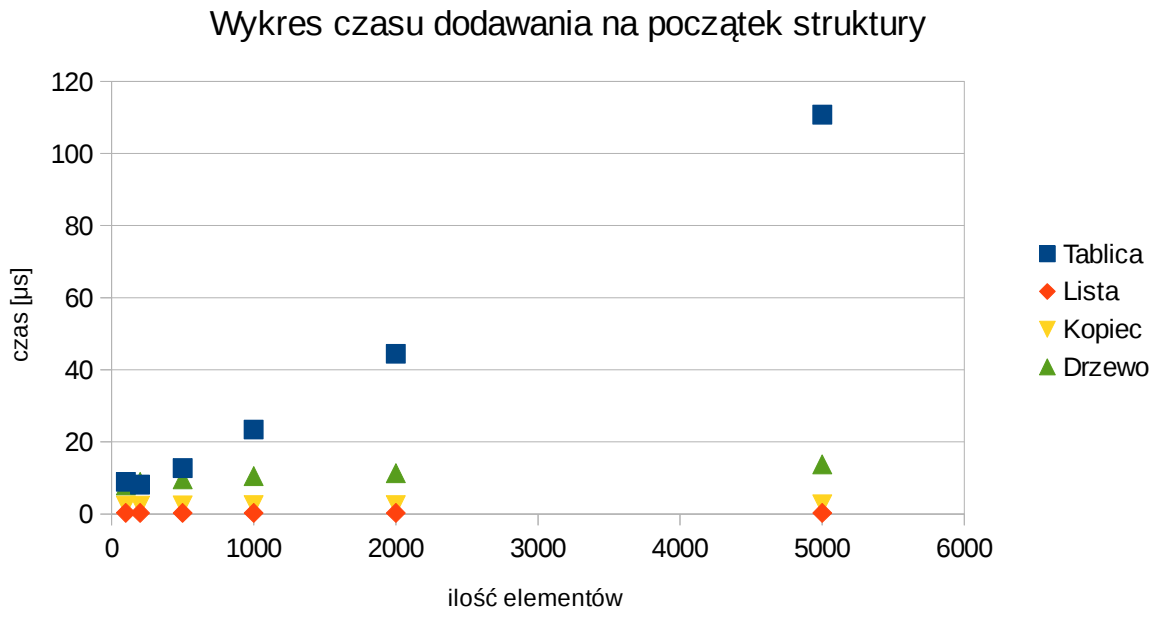
Średni czas realizacji pojedynczej funkcji			
ilość danych	Dodaj [micro s] (początek)	Usuń [micro s] (początek)	Wyszukaj [micro s]
100	2,3	19,2	4,51
200	2,3	33,57	5,115
500	2,332	81,23	6,03
1000	2,416	169,823	6,824
2000	2,459	351,3305	7,525
5000	2,7026	923,9188	8,8406

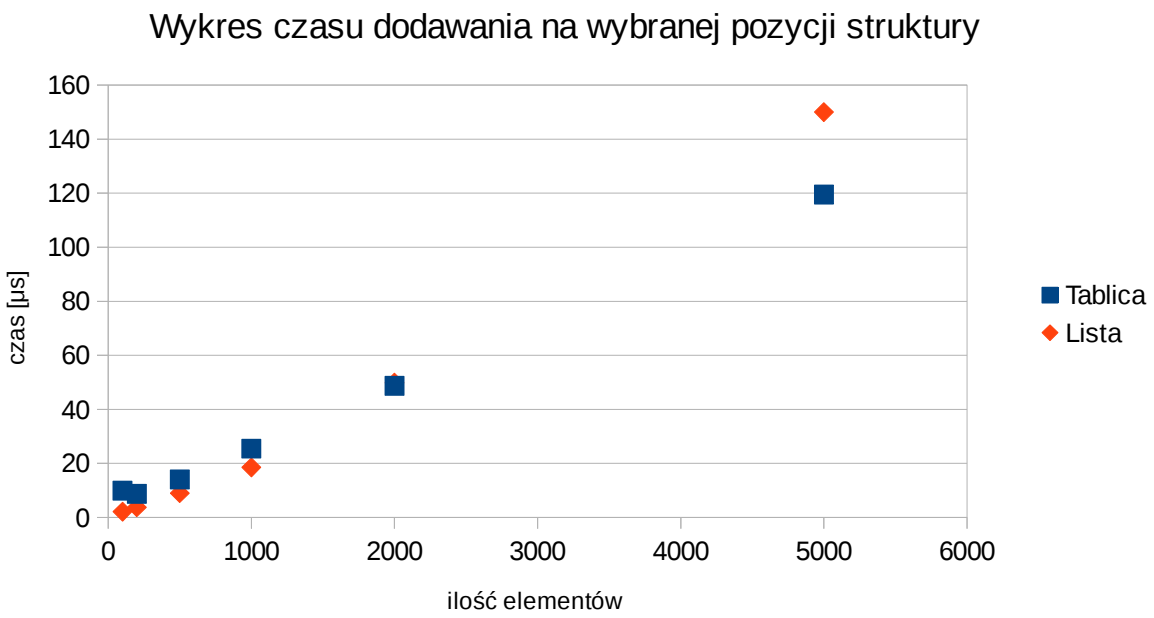
Drzewo czerwono-czarne:

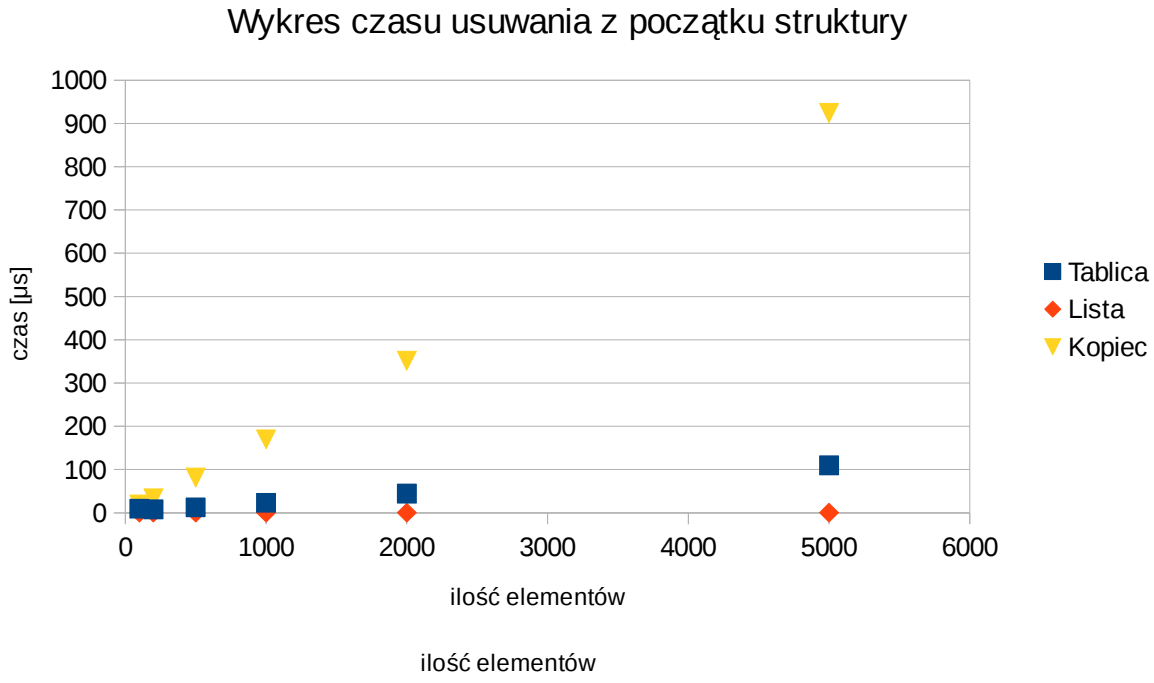
Sumaryczne czasy operacji dla całych struktur		
ilość danych	Dodaj [micro s] (początek)	Wyszukaj [micro s]
100	792	440
200	1780	1042
500	4823	3080
1000	10465	7191
2000	22685	16405
5000	68682	53305

Średni czas realizacji pojedynczej funkcji		
ilość danych	Dodaj [micro s] (początek)	Wyszukaj [micro s]
100	7,92	4,4
200	8,9	5,21
500	9,646	6,16
1000	10,465	7,191
2000	11,3425	8,2025
5000	13,7364	10,661

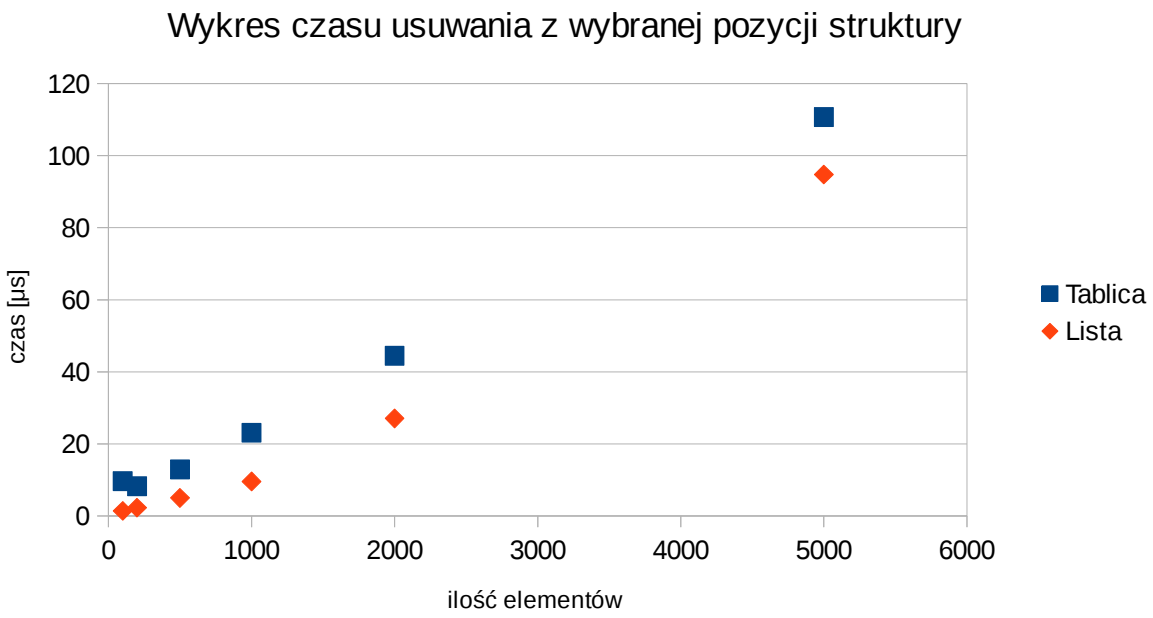
Z uzyskanych danych wygenerowano wykresy w celu oszacowania faktycznej złożoności obliczeniowej zaimplementowanych struktur danych.

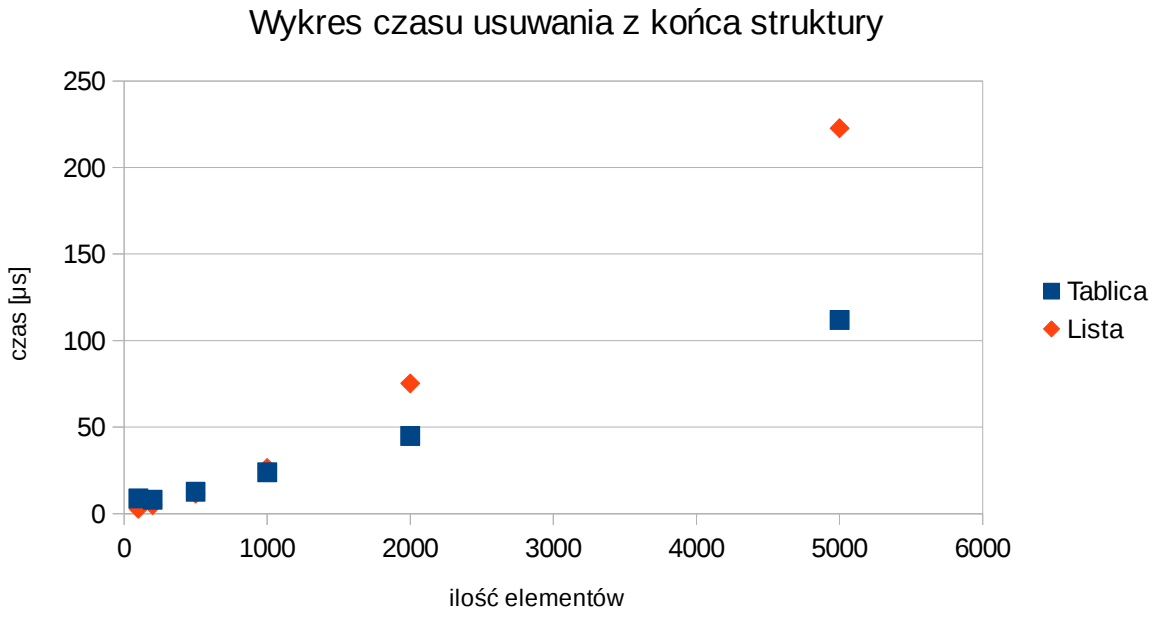


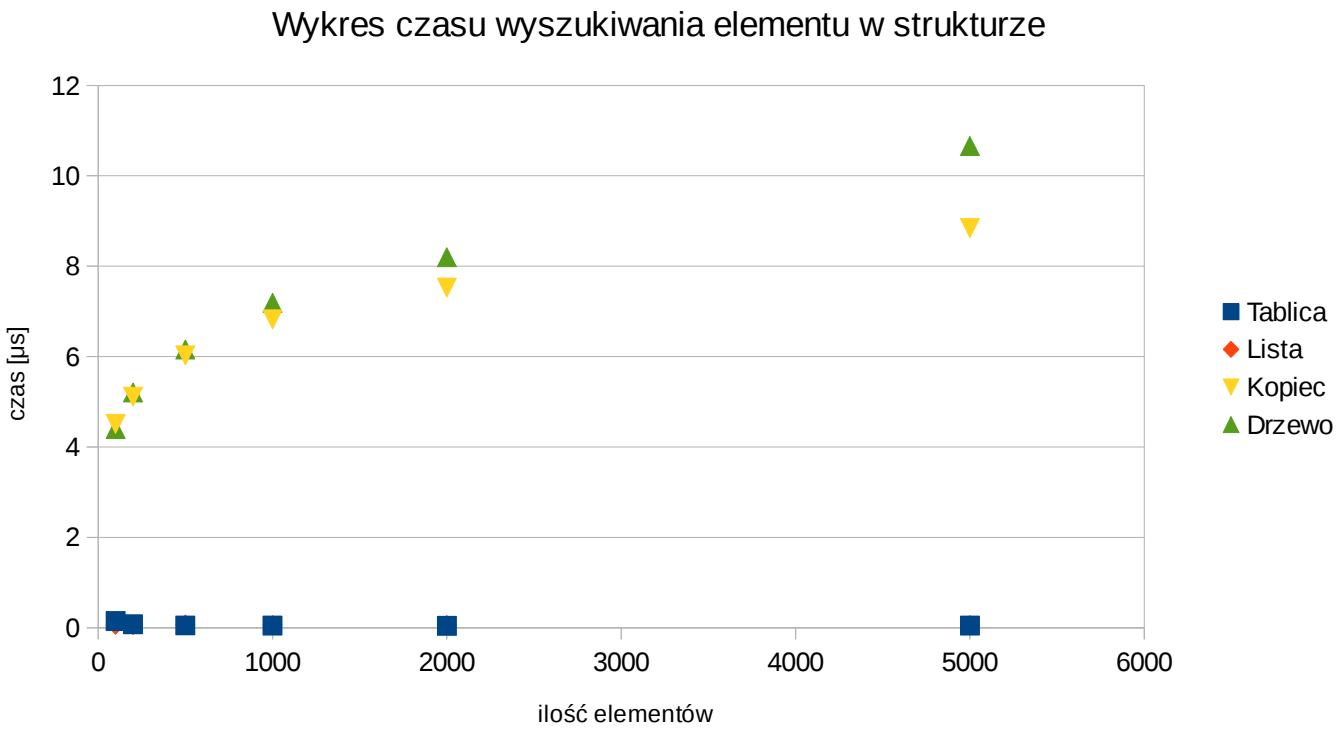




ilość elementów


Wykres czasu usuwania z wybranej pozycji struktury
120
100
80
60
40
20
0
czas [µs]
0
1000
2000
3000
4000
5000
6000
ilość elementów
Tablica
Lista


Wykres czasu usuwania z końca struktury
250
200
150
100
50
0
czas [µs]
0
1000
2000
3000
4000
5000
6000
ilość elementów
Tablica
Lista



Wnioski: Złożoność obliczeniowa zaimplementowanych struktur nie odbiega od złożoności obliczeniowych podanych w literaturze.

Z uwagi na różne cechy struktur każda znajduje zastosowanie przy różnych operacjach i działaniach. W zależności od charakteru przetwarzanych danych stosuje się inne struktury.

Zalety i wady:

Tablica: jest to struktura która charakteryzuje się bardzo szybkim dostępem do elementu, za to bardzo źle znosi modyfikację danych wewnątrz. Zajmuje jej to bardzo dużo czasu. Struktura ta mogłaby znaleźć zastosowanie wszędzie tam gdzie nie ma częstych modyfikacji ilości elementów.

Lista: lista jest strukturą idealną gdy często jest modyfikowana populacja, ale nie same wartości. Dodanie lub usunięcie danych odbywa się w bardzo szybkim tempie. Za to wyszukanie już jest znacznie wolniejsze od tablicy

Drzewo czerwono-czarne: drzewo czerwono czarne jest drzewem samo balansującym się co powoduje niezwykle dobre złożoności obliczeniowe wyszukiwania. Może byś stosowane wszędzie tam gdzie iterowanie po tablicy czy liście daje niewystarczająco szybkie rezultaty.

Podczas eksperymentów zaobserwowano bardzo niespodziewane czasy wyszukiwania w strukturach drzewiastych. Z wykresów widać że zaimplementowane algorytmy zachowały

pożądane złożoności obliczeniowe, ale czasy ich wykonania nawet dla bardzo małych populacji są niespodziewanie długie. Może to być spowodowane architekturą realizacji pracy równoległej. Funkcje wyszukujące w drzewie i kopcu zostały zaimplementowane jako funkcje rekurencyjne. Istnieje możliwość ze podczas wywołania kolejnych iteracji system przeskakiwał do innych zadań co spowodowało przestoje.

Kopiec: jako struktura która na szczycie zawiera największy element idealnie nadaje się do realizacji kolejki priorytetowej.

W eksperymentach udało się uzyskać lepszą złożoność obliczeniową niż znaleziona jako wzorcowa. W algorytmie wykorzystano przeglądanie drzewa w głąb z odrzucaniem poddrzew w których korzeń jest mniejszy od elementu wyszukiwanego wykorzystując cechę drzewa, że każde dziecko jest mniejsze od swojego rodzica. Na wykresach widać, ze drzewo ma lepszą złożoność od kopca więc nie jest to rozwiązanie idealne i tak lepsze od liniowego przeszukiwania tablicy.