IMIĘ I NAZWISKO Mariusz Dajczak
NR INDEKSU 200403
TERMIN czwartek 10:00-12:35
DATA 27.03.2014

PROJEKTOWANIE ALGORYTMOW I METODY SZTUCZNEJ INTELIGENCJI

SPRAWOZDANIE Z LABORATORIUM **Tablica asocjacyjna**

1. Wstęp

Tablica asocjacyjna to abstrakcyjny typ danych, który przechowuje pary elementów: klucz i wartość. Ten obiekt jest również nazywany słownikiem lub mapą. W bibliotekach C++ istnieje szablon std::map, który jest implementacją tablicy asocjacyjnej.

Celem tego ćwiczenia było zaimplementowanie tablicy asocjacyjnej na 3 różnych strukturach danych:

— lista z STL,
— binarne drzewo poszukiwań,
— tablica haszująca z funkcją mieszającą.

Następnie należało przetestować czas dostępu do elementu dla różnych implementacji.

2. Wyniki pomiarów

2.1. Binarne drzewo poszukiwań

Rozmiar problemu	Czas wykonania
10	18.5
100	20.9
1000	21.9
10000	20.1
100000	18
1000000	20.4
10000000	21.3

Tabela 1. Pomiary czasu dla implementacji na BST

2.2. Tablica haszująca

Rozmiar problemu	Czas wykonania
10	22.1
100	21.1
1000	25.8
10000	56.1
100000	475.9
1000000	4074.8
10000000	52219.1

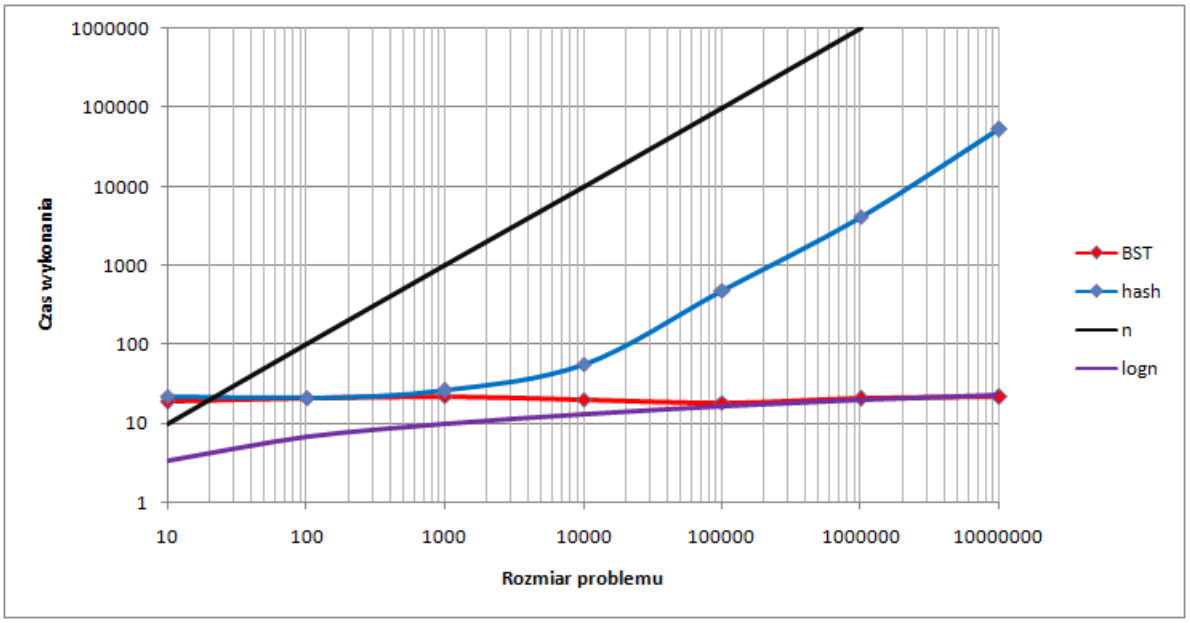
Tabela 2. Pomiary czasu dla implementacji na tablicy haszującej

3. Zestawienie

Na wykres naniosłem 2 funkcje odniesienia:

— $f(n) = n$ - kolor czarny

— $f(n) = logn$ - kolor fioletowy



Rysunek 1. Złożoność obliczeniowa dostępu do elementu

4. Wnioski

Po przeprowadzeniu symulacji widzimy, że dla małego rozmiaru danych wejściowych czas dostepu do elementu jest identyczny dla obydwu implementacji. Wraz z wzrostem rozmiaru problemu widoczna zaczyna być przewaga tablicy opartej na binarnym drzewie poszukiwań, której złożoność oblicziowa wynosi $logn$. W tablicy haszującej zaczynają powstawać kolizje, ponieważ elementów jest więcej niż rozmiar tablicy, i pod jednym jej indeksem znajduje się wiele kluczy (pod każdym indeksem znajduje się lista). W takim przypadku algorytm przeszukuje liniowo owe listy i w efekcie złożoność czasowa wzrasta do liniowej.

Implementacji na zwykłej liście nie udało mi się przetestować. Podczas próbnych testów na kontrolowanych danych wejściowych wszystko działało prawidłowo. Natomiast podczas testowania złożoności obliczeniowej dla wiekszych losowych danych program zawieszał się. Co najlepsze moment , w którym przestawał działać był uzależniony od danych wejściowych, dla każdego nowo wygenerowanego zestawu wysypywał się w innym momencie.