Tablice mieszające

Marcin Ochman

27 kwietnia 2014

1 Opis problemu

Tablice asocjacyjne to wygodna struktura danych, która pozwala dostać się do danych przy użyciu klucza. Istnieje kilka różnych pomysłów, jak taką strukturę zaimplementować.U mnie można znaleźć trzy różne rozwiązania:

- tablica mieszająca z dowiązaniem łańcuchowym
- użycie *std::vector*, klucze są dodawane tak, że są zawsze posortowane
- użycie drzewa przeszukiwań binarnych

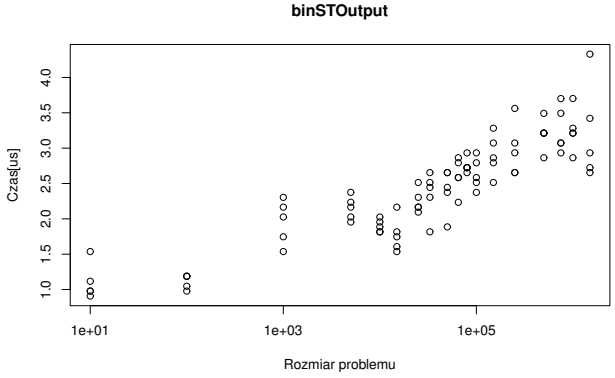
Każde z tych rozwiązań ma swoje wady i zalety, jednak średni czas znalezienia wartości przechowywanej pod podanym kluczem wynosi $\mathcal{O}(lg(n))$ (przy braku kolizji w tablicy mieszającej, drzewo binarne bliskie zrównoważenia).

2 Sposób implementacji funkcji mieszającej dla łańcuchów znaków

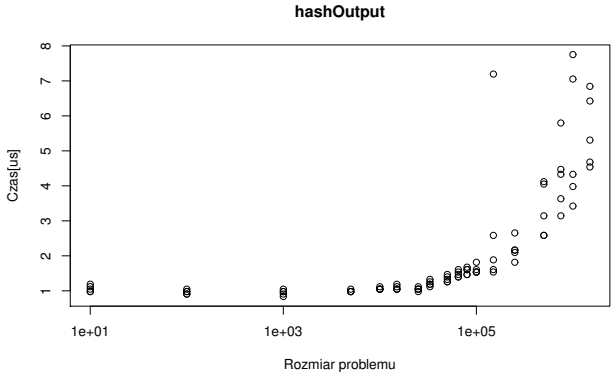
Funkcja mieszająca zaimplementowana jest w bardzo prosty ale sprawdzony sposób. Każdy kod litery jest mnożony przez wagę, która jest odpowiednim indeksem litery w łańcuchu powiększonym o jeden. Wszystko jest sumowane, a następnie brana jest tylko reszta z dzielenia przez 10000.

3 Wyniki przeprowadzonych testów

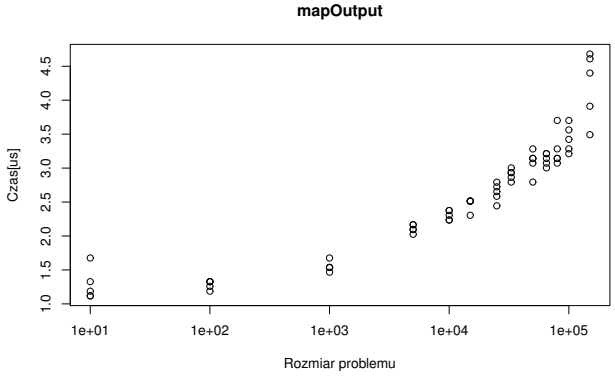
Test polegał na zmierzeniu czasu dostępu do wartości, która została dodana na samym końcu. Ze względu na to, że implementacja na *std::vector* dodaje element w czasie liniowym, trwa to bardzo długo, dlatego ograniczyłem się w tym przypadku do 150000 elementów. Dla pozostałych tablic maksymalny rozmiar to 1,5 miliona elementów. Wyniki testów są przedstawione na poniższych wykresach



Rysunek 1: Wyniki dla drzewa przeszukiwań binarnych



Rysunek 2: Wyniki dla tablicy mieszającej



Rysunek 3: Wyniki dla tablicy asocjacyjnej zbudowanej na *std::vector*

4 Wnioski oraz uwagi

Na podstawie przeprowadzonych testów można wysnuć następujące wnioski:

- Widać, że czas dostępu nie zależy tylko od rozmiaru problemu, sposób uporządkowania danych oraz ich różnorodność jest również istotna, stąd duże rozbieżności w czasie dostępu do elementu dla takiego samego rozmiaru danych.
- Tablica mieszająca, dla dużego rozmiaru ma gorszy czas od drzewa przeszukiwań binarnych, ponieważ występują kolizje i należy dodatkowo szukać w listach w czasie liniowym
- Implementacja na *std::vector* i na drzewie przeszukiwań, co można w przybliżeniu zaobserwować na wykresach, ma czas $\mathcal{O}(log(n)$, tablica mieszająca ma problemy dla liczniejszych zbiorów, występują kolizje, jest to złożoność $\mathcal{O}(nlog(n))$
- Implementacja na *std::vector* nie nadaje się do przechowywania bardzo dużej ilości danych, ponieważ złożoność dodania elementu do tablicy jest liniowa