Tablica asocjacyjna

S. Płaneta

16 kwietnia 2015

1 Zadanie

Zadaniem było utworzenie tablicy asocjacyjnej na tablicy haszującej, dobranie odpowiedniego algorytmu haszującego, rozwiązanie problemu kolizji, a następnie przeanalizowanie złożoności obliczeniowej odwoływania się do elementów tablicy.

2 Przygotowanie

Wybór algorytmu haszującego:

Dobra funkcja haszująca powinna spełniać (w przybliżeniu) założenie prostego równomiernego haszowania. Spełnienie tego warunku jest zazwyczaj niemożliwe, ponieważ rzadko znany jest rozkład prawdopodobieństwa pojawiania się kluczy, a same klucze mogą pojawiać się niezależnie. Typowe podejście polega na takim doborze wartości funkcji haszującej, aby były one maksymalnie niezależne od możliwych wzorców mogących występować w danych.

Istnieje wiele znanych funkcji haszujących, jedne mniej, inne bardziej skomplikowane i efektywne, np.: Haszowanie modularne, Haszowanie przez mnożenie czy Haszowanie uniwersalne.

Jako funkcję haszującą wybrałem Haszowanie przez mnożenie, jako stałą mnożenia wybrałem liczbę polecaną przez Knutha:

$A \approx (sqrt5 - 1)/2 = 0,6180339887...$
$h(key) = \lfloor rozmiar * ((key * A) mod 1) \rfloor$

Wybrałem tę metodę haszowania ze względu na jej pożądane efekty działania, a jednocześnie prostotę implementacji. Funkcja haszująca nie może być zbyt skomplikowana, aby czasy jej obliczania nie przerastały czasów operacji na tablicach - spełnia ona również ten warunek.

Rozwiązanie problemu kolizji:

- **Metoda łańcuchowa** - wszystkie elementy, którym odpowiada ta sama pozycja w tablicy zostają umieszczone na jednej liście

- **Adresowanie otwarte** - jeżeli miejsce, które odpowiada kluczowi jest zajęte, szukamy kolejnego wolnego miejsca, powiększając wartość funkcji haszującej o tzw. współczynnik przyrostu

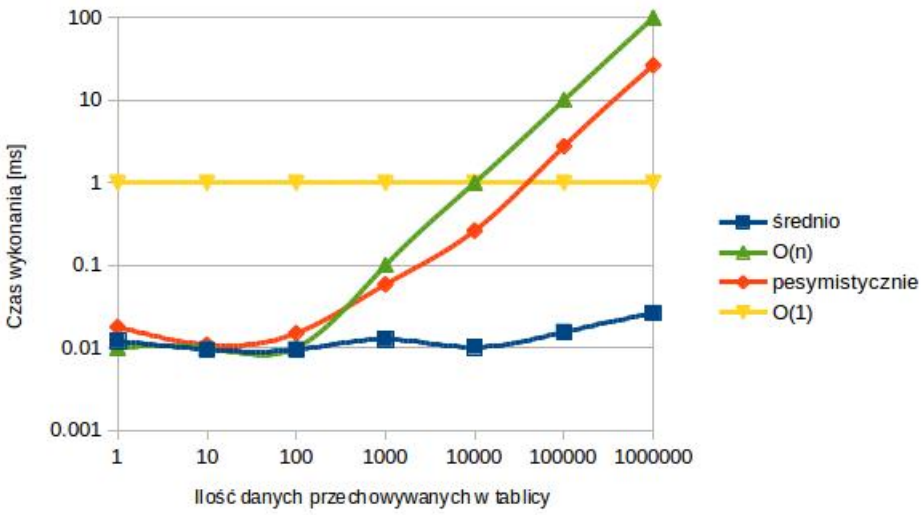
- **Współczynnik wypełnienia** - rozmiar tablicy jest na bieżąco regulowany

- **Haszowanie kukułcze** - zastosowanie dwóch tablic i dwóch odpowiadających im funkcji haszujących

Jako metodę rozwiązania problemu kolizji wybrałem Metodę łańcuchową - przy zastosowaniu odpowiedniej funkcji haszującej, średnia złożoność obliczeniowa wynosi O(1). Wykonywane są wtedy obliczenia, aby najpierw znaleźć odpowiednią listę - metoda h(key), której złożoność zawsze wynosi O(1). Następnie wykonywane są operacje na liście - wyszukanie elementu odpowiadającemu kluczwowi. Czas ten również średnio wynosi O(1) - jeżeli wartości są w miarę równiemiernie rozdystrybuowane po różnych listach. W przypadku pesymistycznym, gdy wszystkie elementy zostaną wpisane do jednej listy, jej złożoność obliczeniowa rośnie do O(n). Jest to spowodowane koniecznością przeszukania całej listy - jak już wiemy operacja ta ma złożoność O(n). Przypadek ten, w praktyce zdarza się bardzo rzadko, podczas testów, uzyskać się go udało jedynie w sposób kontrolowany.

3 Wyniki

Po przetestowaniu algorytmów otrzymano następujący wykres:



4 Wnioski

Zgodnie z założeniami, analizując wykres można zauważyć, że średnia złożoność obliczeniowa jest w klasie O(1), jednak w przypadku pesymistycznym wynosi ona O(n). Można temu przypadkowi łatwo zapobiec, dobierając odpowiednią funkcję haszującą, oraz rozsądny rozmiar tablicy w stosunku do ilości wczytywanych danych.

Wybór zarówno funkcji haszującej jak i metody rozwiązania problemu kolizji może być odpowiednio dobrany do wczytywanych danych (rodzaj kluczy, ilość danych).