|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Wykonawca: Grzegorz Kopciński | | | | | | | | |
| Laboratorium Podstaw Informatyki | | | | | | | | |
| Temat: | Metody dostępu do danych | | | | | | Nr.lab.: 5 | |
| Rok akademicki | termin | Rodz. studiów | grupa | Data lab. | Data oddania sprawozdania | prowadzący | | ocena |
| 2011/2012  semestr letni |  | dzienne | 2 | 31.05 | 14.06 | Robert Tutajewicz | |  |

2

Dane nie posortowane: Wyszukiwano 100 elementów. Min = 1, Max = 189, średnio 93,06

Dane posortowane: Wyszukiwano 100 elementów. Min = 9, Max = 197, średnio 104,33

Posortowanie elementów nie ma wielkiego wpływu na efektywność wyszukiwania, a nawet w posortowanej tablicy średnie wyszukiwanie trawa dłużej. Posortowanie tablicy ma znaczenie tylko dla niskich wartości danych.

3

Wyszukiwano 100 elementów. Min = 2, Max = 8, średnio 6,61

Metoda ta jest wyraźnie lepsza od przeszukiwania sekwencyjnego. Takiego też rezultatu należało się spodziewać. W metodzie tej nawet jeśli szukany klucz znajduje się na końcu tablicy, nie jest konieczne testowanie wszystkich elementów(tak jak miałoby to miejsce w przeszukiwaniu sekwencyjnym). Również maksymalna liczba dostępów do pamięci jest zdecydowanie niższa.

4

Drzewo (nie wywarzone): Wyszukiwano 100 elementów. Min = 1, Max = 15, średnio 8,33

Drzewo (wyważone): Wyszukiwano 100 elementów. Min = 2, Max = 8, średnio 6,61

Drzewa wyważone okazały się nieznacznie lepsze w średnim dostępie do danych. Za to maksymalny potrzebny dostęp do pamięci zmniejszył się prawie dwukrotnie.

5

Elementy należy wpisać w kolejności: 16, 8, 24, 4, 12, 20, 28, 2, 6, 10, 14, 18, 22, 26, 30, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31.

6

Roz. strony 2: Dostępy do pamięci: Min = 2, Max = 9, średnio 6,81. Dostępy do dysku: Min = 1, Max = 4, średnio 3,5

Roz. strony 4: Dostępy do pamięci: Min = 1, Max = 10, średnio 6,58. Dostępy do dysku: Min = 1, Max = 3, średnio 2,7

Roz. strony 10:Dostępy do pamięci: Min = 3, Max = 9, średnio 6,89. Dostępy do dysku: Min = 1, Max = 2, średnio 1,94

Roz. strony 20:Dostępy do pamięci: Min = 3, Max = 9, średnio 6,87. Dostępy do dysku: Min = 1, Max = 2, średnio 1,93

Roz. Strony 200: Dostępy do pamięci: Min = 2, Max = 8, średnio 6,61. Dostępy do dysku: Min = 1, Max = 1, średnio 1

Im większy rozmiar strony tym mniej dostępów do dysku, co oznacza mniejszy czas wyszukiwania. W związku z tym, optymalny ze względu na czas wyszukiwania rozmiar strony wynosi 200. Takie rozwiązanie byłoby jednak sprzeczne z ideą B-drzew, gdyż wtedy wszystkie dane znajdowałyby się pamięci operacyjnej(a przecież B-drzewa stosuje się po to by tylko część danych była w pamięci operacyjnej). W takim wypadku kompromisowym rozwiązaniem jest rozmiar strony równy 10. Wtedy ilość dostępów zarówno do dysku jak i do pamięci jest stosunkowo niska, a także zajęcie pamięci operacyjnej nie jest wysoka.

7

W każdym wierzchołku można zamieścić maksymalnie 2m elementów, w związku z czym każdy wierzchołek może mieć 2m+1 potomków. Na pierwszym poziomie można zamieścić 2m elementów, na następnym 2m(2m+1). Zatem na każdym poziomie można umieścić (2m+1) razy więcej elementów niż na poziomie poprzednim. Maksymalna liczba elementów jest sumą maksymalnych liczb elementów na każdym poziomie. Wyraża się ona wzorem:

Korzeń może mieć minimalnie 1 element i 2 potomków, a każdy inny węzeł minimalnie m elementów i m+1 potomków. W związku z tym na drugim poziomie może być minimalnie 2m elementów, a na każdym następnym (m+1) razy więcej w stosunku do poprzedniego. Suma wszystkich elementów wyraża się wzorem:

8

Roz. strony 2: Dostępy do pamięci: Min = 6, Max = 10, średnio 8,07. Dostępy do dysku: Min = 4, Max = 4, średnio 4

Roz. strony 4: Dostępy do pamięci: Min = 5, Max = 10, średnio 7,41. Dostępy do dysku: Min = 3, Max = 3, średnio 3

Roz. strony 10: Dostępy do pamięci: Min = 4, Max = 9, średnio 7,02. Dostępy do dysku: Min = 2, Max = 2, średnio 2

Roz. strony 20: Dostępy do pamięci: Min = 4, Max = 9, średnio 7,05. Dostępy do dysku: Min = 2, Max = 2, średnio 2

Roz. Strony 200: Dostępy do pamięci: Min = 2, Max = 8, średnio 6,61. Dostępy do dysku: Min = 1, Max = 1, średnio 1

Podobnie jak w B-drzewach zwiększanie rozmiaru strony powoduje zmniejszenie czasu wyszukiwania. Zatem optymalny rozmiar strony wynosi 200. Jeśli jednak chcemy ograniczyć zajęcie pamięci operacyjnej, należałoby wybrać rozmiar strony równy 10, dla którego liczba dostępów do dysku i do pamięci jest stosunkowo niska.

9

Minimalnie lepsze okazały się B-drzewa. Różnica jest jednak niewielka.

10

Dzielenie przez rozmiar tablicy, sondowanie liniowe z krokiem 1

Rozmiar 220: Min = 1, Max = 118, średnio 6,19

Rozmiar 240: Min = 1, Max = 15, średnio 2,82

Rozmiar 260: Min = 1, Max = 28, średnio 1,88

Rozmiar 280: Min = 1, Max = 14, średnio 1,65

Dzielenie przez rozmiar tablicy, sondowanie liniowe z krokiem 7

Rozmiar 260: Min = 1, Max = 39, średnio 4,32

Dzielenie przez rozmiar tablicy, podwójne mieszanie zależne

Rozmiar 260: Min = 1, Max = 15, średnio 2,25

Dzielenie przez rozmiar tablicy, podwójne mieszanie niezależne

Rozmiar 260: Min = 1, Max = 11, średnio 2,24

Mieszanie Fibonacciego, sondowanie liniowe z krokiem 1

Rozmiar 260: Min = 1, Max = 41, średnio 2,32

Podział, składanie i dzielenie, sondowanie liniowe z krokiem 1

Rozmiar 260: Min = 1, Max = 28, średnio 1,88

Wycięcie 3 cyfr klucza, sondowanie liniowe z krokiem 1

Rozmiar 260: Min = 1, Max = 189, średnio 93,06