27.03.2018r.

Struktury danych i złożoność obliczeniowa

Projekt: Struktury danych

Prowadzący: dr inż.. Dariusz Banasiak  
Autor: Wojciech Wójcik  
Indeks: 235621  
Wtorek 15:15 TN

1. Wstęp

Głównym celem było samodzielne zaimplementowanie danych struktur i przeprowadzenie na nich odpowiednich eksperymentów.

Zaimplementowane zostały następujące struktury danych:

* Tablica
* Lista dwukierunkowa
* Kopiec binarny(typu maksimum – element maksymalny w korzeniu)

Eksperymenty miały na celu bezpośrednie porównanie wyżej wymienionych struktur poprzez mierzenie czasu wykonywania podstawowych operacji. Ma to na celu pokazanie do czego dana struktura nadaje się najlepiej. W tym celu został też stworzony interfejs umożliwiający masowe testowanie i generowanie danych. Zostało przeprowadzone ponad 500 eksperymentów, w których został zmierzony czas każdej operacji każdej struktury.

1. Plan eksperymentu:

Wykorzystane środowisko programistyczne: Visual Studio 2017

Język: C i C++

W programie zostały zaimplementowane struktury danych., oraz interfejs umożliwiający testowanie.  
Dane są generowane do i wczytywane z pliku.  
Testy wykonane dla następującej ilości elementów: 100, 1000, 20000, 50000 oraz 1000000.  
Mierzenie czasu zostało wykonane za pomocą QueryPerformanceCounter.   
Mierzenie czasu było uruchamiane zaraz przed i odczytywanie zaraz po uruchomieniu mierzonej funkcji.

Dla każdej struktury została zaimplementowana metoda dodawania elementu, usunięcia elementu i wyszukanie elementu. Dodatkowo dla tablic i listy rozpatrzono osobne operacje dodawania na początku, w dowolnym miejscu i końcu tablicy. Z każdym pomiarem miejsce wstawiania, usuwania elementu oraz dane były generowane losowo.

1. Krótki opis struktur

* **Lista dwukierunkowa** – posiada wskaźnik na poprzedni i następny element. Usuwanie i dodawanie odbywa się poprzez modyfikację wskaźników poprzedniego i następnego elementu w liście. Wskaźniki pierwszego i ostatniego elementu wskazują na wskaźnik zerowy (null pointer).  
  Tworzenie tej struktury odbywa się poprzez dodawanie elementu na początek, jest to dużo szybsze (podczas prób przy dodawaniu elementu na koniec przy tworzeniu struktury czas jej tworzenia wynosił ponad 10s, dlatego porzuciłem dalsze testowanie.)
* **Tablica dynamiczna** – przechowuję wskaźnik na tablicę danych. Realizuje operację modyfikacji struktury zawsze poprzez utworzenie nowej tablicy i przepisanie do niej wartości uwzględniając daną operację.
* **Kopiec typu maksimum** – jest zaimplementowany na podstawie tablicy dynamicznej. Zostały dodane metody pozwalające przechodzić po strukturze w poprawny sposób (wyliczanie indeksu dziecka itp.). Został zaimplementowany w niej interfejs umożliwiający dodanie, usunięcie oraz szukanie, jednak korzysta on z metod tablicy dynamicznej. Interfejs ten również czuwa by zawsze były spełnione warunki kopca.

1. Wyniki

Średnie wyniki wczytywań struktur (wyniki w milisekundach):

|  |  |  |  |
| --- | --- | --- | --- |
| Rozmiar | Średnia z dynamicznej tablicy | Średnia z kopca | Średnia z listy dwukierunkowej |
| 100 | 0,4857398 | 0,5518034 | 0,4360011 |
| 1000 | 5,454463 | 5,706945 | 4,326826 |
| 20000 | 234,0677 | 253,5184 | 93,57667 |
| 50000 | 1088,252 | 1099,091 | 234,0363 |
| 100000 | 3805,921111 | 3818,308 | 461,2496 |

Lista dwukierunkowa(wyniki w milisekundach):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rozmiar | Średnia z dodanie na start | Średnia z usuniecie ze startu | Średnia z dodanie na końcu | Średnia z usuniecie z końca |
| 100 | 0,001455795 | 0,000798886 | 0,001585056 | 0,001434607 |
| 1000 | 0,001455795 | 0,0007671 | 0,009107726 | 0,007798146 |
| 20000 | 0,001618963 | 0,000873053 | 0,197231617 | 0,178922908 |
| 50000 | 0,001895596 | 0,000878447 | 1,220678727 | 1,253461191 |
| 100000 | 0,002252562 | 0,000991721 | 4,430942 | 4,859641 |

|  |  |  |  |
| --- | --- | --- | --- |
| Rozmiar | Średnia z dodanie na pozycji | Średnia z usuniecie z pozycji | Średnia z wyszukiwania |
| 100 | 0,001006555 | 0,001000197 | 0,000544599 |
| 1000 | 0,004653459 | 0,004678888 | 0,002828946 |
| 20000 | 0,051624571 | 0,047314398 | 0,067636201 |
| 50000 | 0,186205391 | 0,085699443 | 0,416772267 |
| 100000 | 0,58362115 | 0,155583541 | 1,920920612 |

Kopiec typu maksimum (wyniki w milisekundach):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rozmiar | Średnia z wyszukiwania | Średnia z usuwania | Średnia z dodawania | Średnia z wyszukiwania |
| 100 | 0,008094812 | 0,003574856 | 0,004621671 | 0,008094812 |
| 1000 | 0,054748069 | 0,004562339 | 0,006825496 | 0,054748069 |
| 20000 | 1,422667715 | 0,046560012 | 0,021853877 | 1,422667715 |
| 50000 | 4,972723794 | 0,152579355 | 0,045277016 | 4,972723794 |
| 100000 | 11,11567888 | 0,30402805 | 0,088169913 | 11,11567888 |

Tablica dynamiczna (wyniki w milisekundach):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rozmiar | Średnia z dodanie na start | Średnia z usuniecie ze startu | Średnia z dodanie na końcu | Średnia z usuniecie z końca |
| 100 | 0,002600089 | 0,001133698 | 0,001078603 | 0,000979007 |
| 1000 | 0,004233884 | 0,002066084 | 0,001765177 | 0,001688891 |
| 20000 | 0,036093971 | 0,018338342 | 0,014214671 | 0,013678552 |
| 50000 | 0,091120421 | 0,039856076 | 0,034171598 | 0,037206869 |
| 100000 | 0,196494217 | 0,081874178 | 0,073834442 | 0,092946277 |

|  |  |  |  |
| --- | --- | --- | --- |
| Rozmiar | Średnia z dodanie na pozycji | Średnia z usuniecie z pozycji | Średnia z wyszukiwania |
| 100 | 0,001203628 | 0,00106165 | 0,000402622 |
| 1000 | 0,001900797 | 0,001773653 | 0,001775773 |
| 20000 | 0,016471457 | 0,01711143 | 0,028681498 |
| 50000 | 0,043829895 | 0,03631686 | 0,067541032 |
| 100000 | 0,094406301 | 0,075567844 | 0,149143759 |

1. Wnioski i analiza wyników
   1. Wczytywanie

Przy czytywaniu najwydajniejsza okazała się lista dwukierunkowa, ale tylko w przypadku, gdy tworzenie struktury odbywało się poprzez dodanie na początek. W innych strukturach nie miało to widocznego znaczenia.

Z uwagi na to że kopiec dziedziczy po dynamicznej tablicy, czas jego tworzenia jest minimalnie dłuższy, jest to spowodowanie dodatkowymi warunkami, które kontrolują poprawność kopca.

Ponadto, czasy tworzenia kopca i tablicy dynamiczna nie rosną liniowo, tak jak lista dwukierunkowa – jest to spowodowane, tym że, przy każdym dodaniu trzeba przekopiować całą zawartość struktury.

* 1. Operacja na strukturach

Wedle przypuszczeń najszybszą metodą dodawania elementów do struktury jest dodawanie na start w liście dwukierunkowej. Ciekawa jest różnica w tablicy dynamicznej przy dodawaniu na początek, a dodawaniu na koniec. Jest ono ponad 2 razy szybsze od dodania na początek (w przypadku rozmiaru struktury o rozmiarze 100000) – takiego wyniku nie spodziewałem się.

W przypadku Listy dwukierunkowej najszybszą operacjami są te wykonywane jak najbliżej początku.

Najszybsze wyszukiwanie daje nam tablica dynamiczna, jest to prawdopodobnie spowodowane ułożeniem danych w pamięci.   
Ciekawe za to są dość długie czasy wyszukiwani w kopcu, oczekiwałem dużo mniejszych. Jak widać dużo szybsze jest przeszukanie całej tablicy. Co prawda kopiec eliminuje pewne gałęzie drzewa, lecz dodatkowe warunki, jak i pobieranie danych z różnych miejsc pamięci skutecznie ograniczają tą strukturę.

W przypadku dodawania na określoną pozycję tablica dynamiczna również okazała się najszybsza.

1. Bibliografia
   1. Materiał z ćwiczeń dr inż. Jarosława Mierzwy
   2. <http://eduinf.waw.pl/inf/>