Karol Kliniewski 238896 Wrocław, 04.04.2018

Poniedziałek, 15:15-16:55 TP

Struktury danych i złożoność obliczeniowa

*„Badanie efektywności operacji dodawania, usuwania oraz wyszukiwania elementów w różnych strukturach danych.”*

Prowadzący: Dr inż. Zbigniew Buchalski

1. **Wstep**

Zgodnie z tematem projektu została zaimplementowana realizacja struktur danych: tablica, lista i kopiec binarny. Pamięć każdej ze nich jest przydzielana dynamicznie(w przypadku zmiany rozmiaru jest ponownie alokowana). Językiem użytym do napisania projektu jest C++ i została stworzona konsolowa aplikacja w wersji obiektowej.

1. **Złożoności**

Wyniki otrzymane w wyniku testów programu będzie można porównać z podręcznikowymi danymi na temat złożoności czasowej operacji na danych strukturach. A są to:

Tablica:

1. Dodawanie i usuwanie:

Operacje te polegają na relokacji pamięci o rozmiarze powiększonym lub zmniejszonym w innym miejscu i przepisaniu naszej zawartości do nowego miejsca. Złożoność czasowa takiego algorytmu to O(n) – zależy wprost proporcjonalnie od ilości elementów. W przypadku, gdy nie musielibyśmy zwalniać pamięci, to usunięcie pierwszego/ostatniego elementu wiązało by się ze zmianą indeksowania/zakresu indeksowania, co by znacznie skróciło operację, jednak założeniem tego zadania jest przydzielanie minimalnego fragmentu pamięci na każdą ze struktur.

1. Wyszukanie

W przypadku pesymistycznym(szukany element znajduje się na końcu) algorytm ma złożoność czasową O(n)

Lista:

1. Dodawanie i usuwanie

Operacje mają złożoność czasową zależną od miejsca, w którym chcemy dodać element. Aby dostać się do odpowiedniego punktu musimy „przeskoczyć” do sąsiedniego elementu po wskaźnikach od początku lub końca.

1. Wyszukanie

Podobnie jak w tablicy wyszukanie w przypadku pesymistycznym ma złożoność czasową O(n), gdyż przeszukuje strukturę od początku do końca i dużo zależy, w którym momencie napotka element.

Kopiec binarny:

1. Dodawanie i usuwanie

Operacje te po wykonaniu wymagają naprawienia struktury kopca, jeśli została ona naruszona. W pesymistycznym przypadku złożoność obliczeniowa jest zależna od liczby poziomów w drzewie i wynosi O(n\*log2n). Jeśli struktura nie została naruszona, wtedy element zostaje dodany do tablicy jako ostatni liść.

1. Wyszukanie

Wyszukanie w kopcu polega na tym samym co wyszukanie w tablicy i ma taką samą złożoność równą O(n)

1. **Pomiar czasu**

Pomiar czasu wykonano dla struktur zbudowanych z 1000, 5000, 10000, 20000 elementów, które zostały wygenerowane przy pomocy złożenia kilku funkcji rand() w połączeniu z resztą z dzielenia. Dzięki temu użyte rekordy były bardzo zróżnicowane, również ze względu na znak. Operacja mierzenia czasu została wykonana przy pomocy funkcji QueryPerformanceCounter z WindowsAPI opisanej w dokumencie dostępnym pod linkiem na stronie prowadzącego: http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pamsi/debug\_and\_time.pdf

1. **Wyniki pomiarów**

* Tablica

Usuwanie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,002 |
| 5000 | 0,0145 |
| 10000 | 0,0456 |
| 20000 | 0,0953 |

Dodanie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0022 |
| 5000 | 0,0185 |
| 10000 | 0,0218 |
| 20000 | 0,037 |

Wyszukanie w tablicy:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0014 |
| 5000 | 0,0132 |
| 10000 | 0,0424 |
| 20000 | 0,0893 |

* Lista

Dodanie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0022 |
| 5000 | 0,0185 |
| 10000 | 0,0218 |
| 20000 | 0,037 |

Usuniecie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0021 |
| 5000 | 0,012 |
| 10000 | 0,0211 |
| 20000 | 0,032 |

Wyszukanie w liście:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0014 |
| 5000 | 0,0122 |
| 10000 | 0,043 |
| 20000 | 0,0875 |

* Kopiec binarny

Dodanie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0022 |
| 5000 | 0,0220 |
| 10000 | 0,0312 |
| 20000 | 0,0412 |

Usuniecie elementów:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0021 |
| 5000 | 0,021 |
| 10000 | 0,029 |
| 20000 | 0,037 |

Wyszukanie w kopcu:

|  |  |
| --- | --- |
| Ilość elementów | średni czas[ms] |
| 1000 | 0,0024 |
| 5000 | 0,0114 |
| 10000 | 0,0411 |
| 20000 | 0,079 |

1. **Wnioski**

Wyniki pomiarów są porównywalne z wartościami spodziewanymi na podstawie literatury. Większe różnice by były dostrzegalne, jakby osobno rozpatrzyć dodawanie i usuwanie elementów szczególnie z początku i końca ze względu na sposób docierania do elementów w liście. Dla struktur budowanych na tablicy czas dodania/usunięcia jest taki sam niezależnie od miejsca, dla listy ma pozycja spory wpływ, a najdłużej są dodawane elementy do środka struktury. Największym problemem podczas implementacji okazało się wyświetlanie kopca. Bardzo dobrze wygląda przy liczbach jedno lub dwu cyfrowych. Prawie wszystkie pomiary mają wyniki rosnące zbliżenie do liniowości. Spodziewane były większe różnice w czasach operacji na strukturach, jednak średnie czasy nie odbiegają w znacznym stopniu od siebie.