**Dane studenta:** Filip Kurasz 249458 27.03.2020

**Nazwa kursu:** Projektowanie algorytmów i metody sztucznej inteligencji

**Dane prowadzącego:** mgr inż. Marta Emirsajłow

**Termin zajęć:** piątek 13:15-15:00

**SPRAWOZDANIE**

**PROJEKT NR 1**

**ALGORYTMY SORTOWANIA**

1. Wprowadzenie

Należało wybrać trzy algorytmy sortowania, zaimplementować i przeanalizować ich efektywność. W poniższym sprawozdaniu wykorzystane zostały następujące algorytmy sortowania: szybkie (quicksort), przez scalanie (mergesort), introspektywne (introsort).

Wszystkie algorytmy testowane były na 100 tablicach o następujących rozmiarach:

10.000, 50.000, 100.000, 500.000, 1.000.000.

W dodatku dla każdego rozmiaru były wymuszane warunki początkowe:

Cała tablica wypełniona losowymi liczbami, tablica posortowana w 25%, w 50%, w 75%, w 95%, w 99%, w 99.7% oraz posortowana malejąco.

1. Opis badanych algorytmów
2. QuickSort

Sortowanie szybkie jest jednym z najpopularniejszych algorytmów sortowań działający na zasadzie „dziel i zwyciężaj”. Jest wydajny, a jego złożoność obliczeniowa jest rzędu , natomiast w przypadku pesymistycznym wynosi ona . Jest powszechnie używany ze względu na szybkość dzięki zastosowaniu rekurencji.

1. MergeSort

Rekurencyjny algorytm sortowania stosujący również zasadę „dziel i zwyciężaj”. Szczególnie przydatny przy danych dostępnych sekwencyjnie, na przykład w postaci listy jednokierunkowej. Jego złożoność obliczeniowa wynosi , w najgorszym przypadku również .

1. IntroSort

Odmiana sortowania hybrydowego, w której wyeliminowany został problem złożoności występującej w najgorszym przypadku algorytmu sortowania szybkiego. Głównym założeniem algorytmu jest obsługa najgorszego przypadku algorytmu sortowania szybkiego tak, aby zapewnić logarytmiczno-liniową złożoność obliczeniową. Jego złożoność obliczeniowa wynosi również dla przypadku najgorszego.

1. Przebieg badanych algorytmów
2. QuickSort – Wyniki w sekundach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ROZMIAR/PROCENT | 0 | 25 | 50 | 75 | 95 | 99 | 99,7 | ODWROTNIE |
| 10000 | 0,129 | 0,125 | 0,224 | 0,113 | 0,087 | 0,092 | 0,066 | 0,056 |
| 50000 | 0,696 | 0,692 | 1,631 | 0,544 | 0,429 | 0,374 | 0,358 | 0,273 |
| 100000 | 1,444 | 1,388 | 2,305 | 1,106 | 0,954 | 0,788 | 0,734 | 0,605 |
| 500000 | 7,514 | 7,042 | 11,609 | 5,9 | 5,543 | 3,752 | 3,678 | 3,327 |
| 1000000 | 15,278 | 14,192 | 50,526 | 11,76 | 9,029 | 8,616 | 7,159 | 6,108 |

1. MergeSort – Wyniki w sekundach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ROZMIAR/PROCENT | 0 | 25 | 50 | 75 | 95 | 99 | 99,7 | ODWROTNIE |
| 10000 | 0,126 | 0,127 | 0,138 | 0,105 | 0,076 | 0,067 | 0,063 | 0,041 |
| 50000 | 0,721 | 0,689 | 1,259 | 0,552 | 0,438 | 0,404 | 0,397 | 0,28 |
| 100000 | 1,475 | 1,479 | 4,655 | 1,115 | 0,895 | 0,808 | 0,724 | 0,609 |
| 500000 | 7,691 | 7,228 | 14,624 | 5,643 | 4,434 | 3,8 | 3,833 | 3,1 |
| 1000000 | 15,878 | 15,12 | 51,506 | 11,54 | 8,939 | 8,39 | 7,579 | 7,617 |

1. IntroSort – Wyniki w sekundach

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ROZMIAR/PROCENT | 0 | 25 | 50 | 75 | 95 | 99 | 99,7 | ODWROTNIE |
| 10000 | 0,168 | 0,232 | 0,171 | 0,135 | 0,113 | 0,156 | 0,101 | 0,101 |
| 50000 | 1,862 | 0,997 | 1,145 | 1,516 | 0,702 | 0,683 | 0,685 | 0,558 |
| 100000 | 2,711 | 3,505 | 2,398 | 2,492 | 2,139 | 1,983 | 1,878 | 1,493 |
| 500000 | 16,671 | 17,196 | 13,664 | 9,199 | 6,995 | 11,025 | 7,79 | 6,236 |
| 1000000 | 27,0641 | 24,286 | 22,654 | 21,112 | 15,452 | 14,674 | 14,087 | 10,422 |

1. Podsumowanie

QuickSort i MergeSort mają zbliżone czasy we wszystkich testach. Ponadto przy 50% wypełnionej tablicy oba, ale w szczególności QuickSort ma problemy z posortowaniem w porównaniu do innych warunków początkowych. Wraz z ilością posortowanych elementów na starcie, czas sortowania ulega skróceniu. Najszybszy czas sortowania notujemy dla tablicy posortowanej malejąco. IntroSort wykazuje ciągły spadek czasu sortowania. Nie zauważono znacznych odchyleń podczas testów tego rodzaju sortowania. Różnice w czasie mogły wynikać z braku wystarczającej optymalizacji.

1. Bibliografia

<https://en.wikipedia.org/wiki/Merge_sort>

<https://en.wikipedia.org/wiki/Quicksort>

<https://en.wikipedia.org/wiki/Introsort>

<https://www.geeksforgeeks.org/>

<https://stackoverflow.com/>

<https://www.youtube.com/channel/UC0RhatS1pyxInC00YKjjBqQ>