Sortowanie przez scalanie

S. Płaneta

16 kwietnia 2015

1 Zadanie

Zadaniem było zaimplementowanie algorytmu sortowania przez scalanie i dokonanie jego analizy.

2 Teoria

2.1 Złożoność pamięciowa:

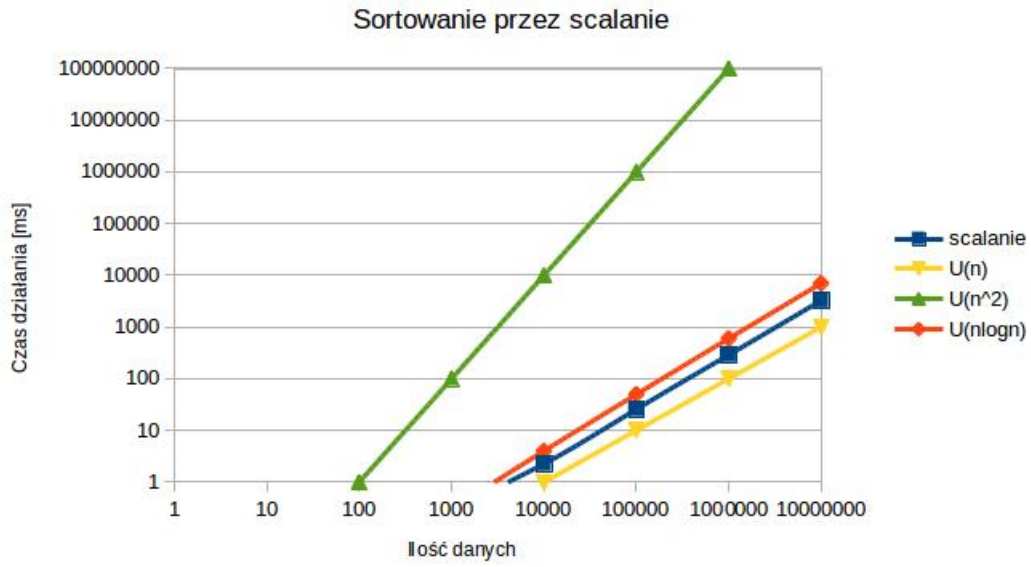
Algorytm w każdym przypadku ma złożoność obliczeniową równą $O(n)$ - jest to spowodowane koniecznością alokowania pamięci na dodatkową tablicę. Sortowanie nie odbywa się w miejscu, co jest niewątpliwie wadą tego rozwiązania.

2.2 Złożoność obliczeniowa:

Samo scalanie, niezależnie od przypadku ma zawsze złożoność $O(n)$, dlatego złożoność obliczeniowa całego algorytmu Mergesort'a, niezależnie od przypadku należy do klasy $O(n \log n)$.

3 Wyniki

Po przetestowaniu algorytmu losowymi danymi otrzymano następujący wykres:



Dane pomiarowe:

Ilość danych	Czas sortowania [ms]
1	0.0003
10	0.0011
100	0.0106
1000	0.2428
10000	2.2586
100000	25.6503
1000000	289.9
10000000	3252.6

4 Wnioski

Zgodnie z założeniami, analizując wykres można zauważyć, że algorytm sortowania przez scalanie ma złożoność obliczeniową asymptotycznie równą $O(n \log n)$

Największą wadą testowanego algorytmu jest jego złożoność pamięciowa: - potrzebny dodatkowy obszar pamięci na przechowywanie kopii podtablic do scalenia.

Algorytm ten ma też swoje zalety:
- stabilność - złożoność obliczeniowa asymptotycznie równa $O(n \log n)$ bez

względu na rodzaj danych wejściowych
- szybkość działania - złożoność czasowa $O(n \log n)$ w wielu przypadkach jest rozsądnym i wystarczającym czasem sortowania
- łatwość implementacji

Średnio algorytmy Mergesort'a jest wolniejszy od Quicksort'a, jedynie w przypadkach pesymistycznych spiuje się lepiej. (Jednak w Quicksorcie dość łatwo jest się przed nimi zabezpieczyć)