

IMIĘ I NAZWISKO    Mariusz Dajczak

NR INDEKSU        200403

TERMIN            czwartek 10:00-12:35

DATA              20.03.2014

# **PROJEKTOWANIE ALGORYTMOW I METODY SZTUCZNEJ INTELIGENCJI**

*SPRAWOZDANIE Z LABORATORIUM*

**Algorytmy sortujące**

## 0.1. Wstęp

Sortowanie jest jednym z podstawowych problemów informatyki. Polega na uporządkowaniu danego zbioru względem pewnych cech, np. posortowanie liczb od najmniejszej do największej. Istnieje wiele algorytmów sortowania. Począwszy od prymitywnego sortowania głupiego, poprzez liniowe, bąbelkowe, aż do algorytmów typu Quicksort lub mergesort. Cecha wyznaczającą efektywność algorytmu sortowania jest złożoność czasowa.

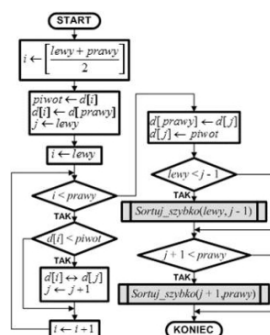
## 0.2. Cel ćwiczenia

Celem ćwiczenia jest zbadanie złożoności obliczeniowej algorytmów sortowania. W sprawozdaniu będę omawiał trzy algorytmy:

- sortowanie szybkie
- sortowanie przez scalanie
- sortowanie przez kopcowanie,

Wszystkie te algorytmy mają złożoność czasową  $n \log n$  i uważane są za jedno z najszybszych.

## 0.3. Quicksort



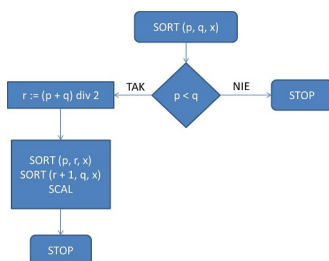
Rysunek 1. Schemat blokowy quicksort

Algorytm sortowania szybkiego jest uważany za najszybszy jeśli chodzi o dane losowe. Polega on na metodzie dziel i zwyciężaj. Polega to na tym, że zbiór danych zostaje podzielony na dwa podzbiory i każdy z nich jest sortowany niezależnie. W średnim przypadku jego złożoność obliczeniowa wynosi  $n \log n$ , natomiast w najgorszym  $n^2$ .

### Dane pomiarowe

Rozmiar	Powtórzenia	Czas
10	20	79.3
100	20	1738.25
1000	20	15227.4
10000	20	120922

## 0.4. Merge



Rysunek 2. Schemat blokowy sortowania przez scalanie

Algorytm sortowania przez scalanie polega na podzieleniu zbioru na jednoelementowe podzbiory, a następnie na łączeniu tych podzbiorów w większy zbiór posortowany.

### Dane pomiarowe

Rozmiar	Powtórzenia	Czas
10	5	541
100	5	5155.4
1000	5	76662.4
10000	5	1.37028e+006

## 0.5. Heapsort

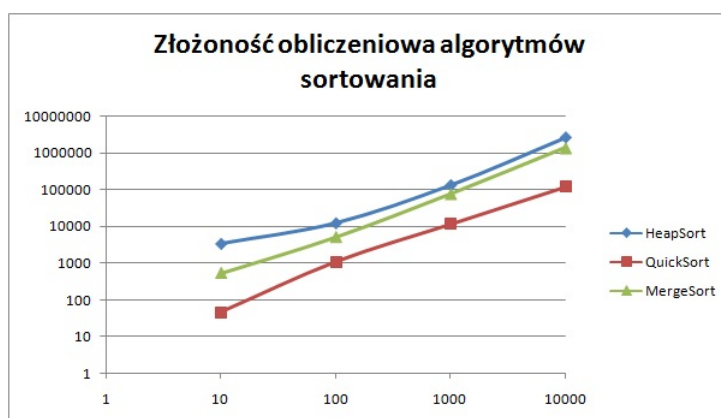
Algorytm sortowania poprzez kopcowanie jest najwolniejszym z algorytmów o złożoności  $n \log n$ . Polega na stworzeniu kopca binarnego z danych. Następnie wartość znajdująca się w korzeniu wedruje na początek tablicy wynikowej,

a ostatni element kopca przechodzi do korzenia. Kopiec jest porządkowany i operacja się powtarza, aż do kompletnego rozebrania kopca.

### Dane pomiarowe

Rozmiar	Powtórzenia	Czas
10	10	3417.3
100	10	12390.9
1000	10	132030
10000	10	2,58E+06

## 0.6. Zestawienie



Rysunek 3. Złożoność obliczeniowa algorytmów sortowania

## 0.7. Wnioski

Wszystkie algorytmy, które testowałem mają złożoność obliczeniową  $n \log n$ . Najszybszym jest QuickSort, wraz z wzrostem problemu jego przewaga rośnie. MergeSort oraz HeapSort mają zbliżoną charakterystykę złożoności czasowej.

QuickSort Jest również lepszy pod względem zapotrzebowania na pamięć. Działa on na oryginalnej tablicy z danymi, co powoduje, że praktycznie nie zużywa dodatkowych zasobów. Natomiast pozostałe dwa algorytmy tworzą tablice pomocnicze, co zwiększa złożoność pamięciową.