Dolne ograniczenie złożoności czasowej sortowania przez porównania

Definicja *Sortowanie przez porównania* to takie sortowanie, w którym końcowa permutacja sortowanych elementów jest osiągana wyłącznie na podstawie porównań sortowanych elementów ze sobą.

Definicja *Drzewo decyzyjne dla sortowania* to drzewo binarne, w którego węzłach są porównania sortowanych elementów, którego krawędzie są etykietowane wynikami porównań i którego liście zawierają uporządkowania sortowanych elementów wynikające z porównań na ścieżce od korzenia do danego liścia.

Teza Każdy algorytm sortujący przez porównania, dla dowolnej ale ustalonej ilości sortowanych elementów można przedstawić jako drzewo decyzyjne dla sortowania.

Twierdzenie Każdy algorytm sortujący przez porównania ma czas pesymistyczny $\Omega(n \lg n)$, gdzie n to ilość sortowanych elementów.

Idea dowodu

algorytm przedstawiamy jako drzewo decyzyjne dla sortowania n elementów

złożoność pesymistyczna jest co najmniej taka jak wysokość drzewa (oznaczmy ją przez h) czyli długość najdłuższej ścieżki od korzenia do liścia

$$2^h \geq \text{ ilość liści } \geq n!$$

co daje po zlogarytmowaniu

$$h \geq lg\, n!$$

Daje się oszacować: $h \geq c\, n \lg n$ czyli $h = \Omega(n \lg n)$

Twierdzenie Każdy algorytm sortujący przez porównania ma średni czas $\Omega(n \lg n)$, gdzie n to ilość sortowanych elementów.

Sortowania w czasie liniowym

Te algorytmy *nie należą* do klasy algorytmów sortujących przez porównania. Sortują używając innych sposobów niż porównywanie sortowanych elementów, nakładając jednak pewne wymagania na dane, które mogą sortować.

Sortowanie przez zliczanie (counting-sort)

Uwaga: tablice indeksowane od 1

```
Counting-sort(A,B,k)
// A - tablica do sortowania
// B - wynik sortowania
// k - zakres warości w tablicy A
// C - pomocnicza lablica "liczników"

  for i=1 to k
     C[i]=0
  for j=1 to A.length
     C[A[j]] = C[A[j]]+1;
     // teraz C[i] == ilość wartości równych "i" w tablicy A
  for i=2 to k
     C[i]=C[i]+C[i-1]
     // teraz C[i] == ilość wartości mniejszych lub równych  "i" w A
  for j=A.length downto 1
     B[C[A[j]]]=A[j]
     C[A[j]]=C[A[j]]-1
```

Stwierdzenie Pesymistyczny czas sortowania przez zliczanie jest $\Theta(n+k)$, gdzie n to rozmiar sortowanej tablicy a k to zakres sortowanych wartości (wartości w zakresie od 1 do k).

Wniosek Przy oznaczaniach jak powyże, jeżeli k jest stałą lub jest $O(n)$ to pesymistyczny czas sortowania przez zliczanie jest $\Theta(n)$.

Sortowanie pozycyjne (Radix-sort)

```
Radix-sort(A,d,k)
// A - tablica do sortowania zawierająca
// liczby d-cyfrowe
// k - zakres wartości cyfr (niekoniecznie
//     dziesiątkowy system liczenia)

for j=d downto 1
   sortuj A według cyfry z pozycji j-tej
   stabilnym algorytmem sortowania, gdzie
   pozycje liczymy od najbardziej znaczącej cyfry,
   czyli od przodu liczby
```

Stwierdzenie Pesymistyczny czas sortowania pozycyjnego, gdzie każdą pozycję sortujemy przez zliczanie jest $\Theta(d(n+k))$, gdzie n to rozmiar sortowanej tablicy, d to ilość cyfr w liczbie (liczby d-cyfrowe), a k to zakres cyfr (cyfry w zakresie od 0 do $k-1$).

Wniosek Przy oznaczaniach jak powyżej, jeżeli k i d są stałymi to pesymistyczny czas sortowania pozycyjnego, gdzie każdą pozycję sortujemy przez zliczanie, jest $\Theta(n)$.