# Quicksort

niedziela, 16 czerwca 2024 13:13

## Algorytm zrandomizowany

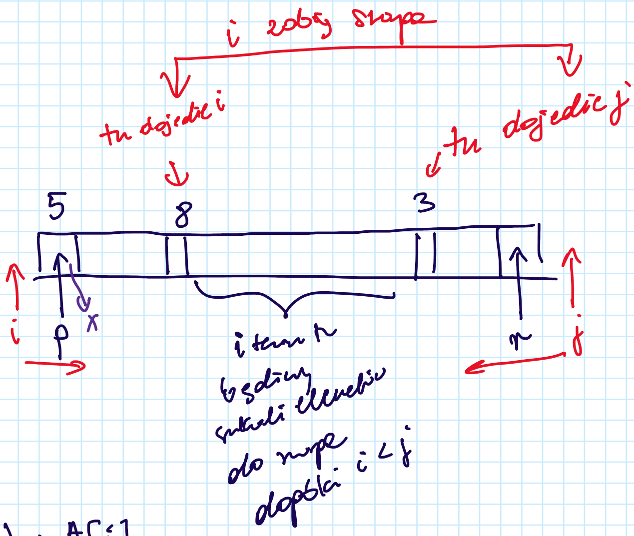
parametry odkąd do kąd sortujemy

```
procedure quicksort (A[1...n], p, r)
    if r-p jest małe then insert-sort (A[p...r])
    else choosepivot (A,p,r)
         q ← partition (A,p,r)   → czas liniowy od liczby elementów
         quicksort (A,p,q)
         quicksort (A,q+1,r)
```

Kluczowy w quicksorcie jest wybór pivota

wybór pivota będzie w czasie stałym

```
procedure partition (A[1...n], p, r)
    x ← A[p]
    i ← p-1      (wartowniki)
    j ← r+1
    while i < j do
        repeat j ← j-1 until A[j] ≤ x
        repeat i ← i+1 until A[i] ≥ x
        if i < j then zamień miejscami A[i] z A[j]
        else return j
```




Czyli partition działa w czasie $\Theta(r-p)$

## Wybór pivota:

1) pierwszy element → słabe gdy tablica posortowana wtedy $O(n^2)$
2) mediana → możemy ją znaleźć w czasie liniowym ale ma dużą stałą więc tak nie chcemy
3) mediana z małej próbki: biorę pierwszy ostatni i środkowy element liczę z nich medianę → znacznie przyspiesza algorytm
4) randomizowany:

```
procedure choosepivot (A[1...n], p, r)
    i ← random (p, q)
```

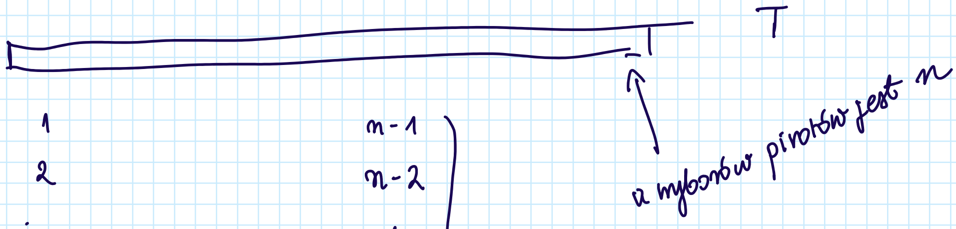
procedure choosepivot(A[1...n], p, r)

$i \leftarrow random(p, q)$

zamień $A[p]$ i $A[i]$ miejscami

generalnie losowy wybór jest spoko

## Analiza procedury (z losowym wyborem Pivota)




| | |
|---|---|
| 1 | $n-1$ |
| 2 | $n-2$ |
| $\vdots$ | $\vdots$ |
| $n-1$ | 1 |

$\left.\begin{matrix} n-1 \\ n-2 \\ \vdots \\ 1 \end{matrix}\right\} n-1$

rang elementu – liczba indeksów w tablicy zawierających elementy mniejsze od x

przykład:

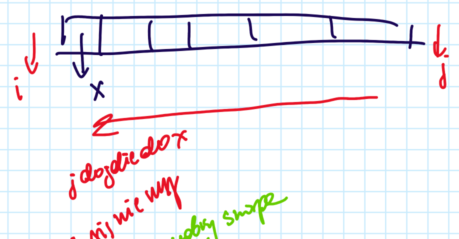
| | | | | |
|---|---|---|---|---|
| 7 | 1 | 2 | 4 | 5 |

$rank(1, A) = 1$

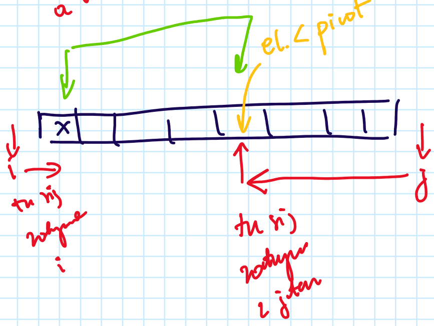
$rank(4, A) = 3$

$rank(7, A) = 5$

1° $rank(pivot, A) = 1$ skutkuje podziałem $1, n-1$




2° $rank(pivot, A) = 2$




i tu dojedzie j-tem ... się zrobi → podział na 1 i $n-1$

i to dlaczego jest ten ostatni więc 2 razy → podział na 1 i n-1

↓ cyli to jest ten powtarzający się przypadek którego szukamy

Dla kolejnych przypadków będzie:

3° $\quad 2, \; n-2$

4° $\quad 3, \; n-3$

itd

→ ppb wystąpienia tych przypadków to $\frac{1}{n}$

a podziału $1, n-1$ jest $\frac{1}{n}$

Oczekiwany czas procedury quick sort:

$T(1) = 1$

$$T(n) = \frac{1}{n}\left[\left(T(1) + T(n-1)\right) + \sum_{d=1}^{n-1}\left(T(d) + T(n-d)\right)\right] + \Theta(n)$$

Wynik procedury *partition* w oczywisty sposób zależy od wartości $rank(A[p], A[p..r])$. Gdy jest ona równa $i$ (dla $i = 2, \ldots, n$), wynikiem *partition* jest $p+i-2$. Ponadto, gdy $rank(A[p], A[p..r]) = 1$, wynikiem jest $p$. Tak więc zmienna $q$ z procedury *quicksort* przyjmuje wartość $p$ z prawdopodobieństwem $2/n$, a każdą z pozostałych wartości (tj. $p+1, p+2, \ldots, r-1$) z prawdopodobieństwem $1/n$. Stąd oczekiwany czas działania procedury *quicksort* wyraża się równaniem

$$\begin{cases} T(1) = & 1 \\ T(n) = & \frac{1}{n}\left[(T(1) + T(n-1)) + \sum_{d=1}^{n-1}(T(d) + T(n-d))\right] + \Theta(n) \end{cases}$$

Zmienna $d = q - p + 1$ oznacza długość pierwszej z podtablic.

Ponieważ $T(1) = \Theta(1)$ a $T(n-1)$ w najgorszym przypadku jest równe $\Theta(n^2)$, więc

$$\frac{1}{n}(T(1) + T(n-1)) = O(n).$$

To pozwala nam pominąć ten składnik, ponieważ będzie on uwzględniony w ostatnim członie sumy. Tak więc:

$$T(n) = \frac{1}{n}\sum_{d=1}^{n-1}(T(d) + T(n-d)) + \Theta(n).$$

W tej sumie każdy element $T(k)$ jest dodawany dwukrotnie (np. $T(1)$ raz dla $q = 1$ i raz dla $q = n-1$), więc możemy napisać:

$$T(n) = \frac{2}{n}\sum_{k=1}^{n-1}T(k) + \Theta(n) \tag{1}$$

Ponieważ mamy silne przesłanki, by przypuszczać, że rozwiązanie tego równania jest rzędu $\Theta(n \log n)$, ograniczymy się do sprawdzenia tego faktu. Niech

$$T(n) = \frac{2}{n}\sum_{k=1}^{n-1}T(k) + \Theta(n) \leq an\log n + b$$

dla pewnych stałych $a, b > 0$. Naszym zadaniem jest pokazanie, że takie stałe $a$ i $b$ istnieją.

Bierzemy $b$ wystarczająco duże by $T(1) \leq b$. Dla $n > 1$ mamy:

$$T(n) \;=\; \frac{2}{n}\sum_{k=1}^{n-1}(ak\log k + b) + \Theta(n) \leq \frac{2a}{n}\sum_{k=1}^{n-1}k\log k + \frac{2b}{n}(n-1) + \Theta(n)$$

Proste oszacowanie $\sum_{k=1}^{n-1} k \log k$ przez $\frac{1}{2}n^2 \log n$ nie prowadzi do celu, ponieważ musimy pozbyć się składnika $\Theta(n)$. Oszacujmy więc $\sum_{k=1}^{n-1} k \log k$ nieco staranniej:

**Fakt 2** $\sum_{k=1}^{n-1} k \log k \leq \frac{1}{2}n^2 \log n - \frac{1}{8}n^2$

Dowód. Rozbijamy sumę na dwie części:

$$\sum_{k=1}^{n-1} k \log k = \sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k + \sum_{k=\lceil n/2 \rceil}^{n-1} k \log k$$

Szacując $\log k$ przez $\log \frac{n}{2}$ dla $k < \lceil \frac{n}{2} \rceil$ oraz przez $\log n$ dla $k \geq \lceil \frac{n}{2} \rceil$, otrzymujemy:

3

Spośób typu "taki sobie"

KLO

$$\sum_{k=1}^{n-1} k \log k \quad \leq \quad ((\log n) - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k = \log n \sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k \leq$$
$$\frac{1}{2}n(n-1)\log n - \frac{1}{2}(\frac{n}{2} - 1)\frac{n}{2} \leq \frac{1}{2}n^2 \log n - \frac{1}{8}n^2$$

□

Teraz możemy napisać

$$\frac{2a}{n}\left(\frac{1}{2}n^2 \log n - \frac{1}{8}n^2\right) + \frac{2b}{n}(n-1) + \Theta(n) \leq an \log n - \frac{a}{4}n + 2b + \Theta(n) =$$
$$an \log n + b + \left(\Theta(n) + b - \frac{a}{4}n\right)$$

Składową $(\Theta(n) + b - \frac{a}{4}n)$ możemy pominąć, dobierając $a$ tak, by $\frac{a}{4}n \geq \Theta(n) + b$. Zauważmy, że taki dobór zależy jedynie od stałej $b$ oraz od stałej ukrytej pod $\Theta$, a więc za $a$ można przyjąć odpowiednio dużą stałą.