# Lista 3 zadanie 8

Wiktor Hamberger
308982

28 kwietnia 2020

Idea rozwiązania tego zadania jest mocno wzorowana na scalaniu posortowanych tablic w algorytmie sortowania przez scalanie. Skoro po scaleniu $T_1, T_2, T_3$ nasza mediana znajdzie się na pozycji $\lceil \frac{3n}{2} \rceil$ (każda z tablic ma n elementów), to oznacza że przy scalaniu tych tablic w "mergesortowy" sposób, po wybraniu $\lceil \frac{3n}{2} \rceil$ najmniejszych elementów dostastaniemy medianę – będzie ona $\lceil \frac{3n}{2} \rceil$-szym najmniejszym elementem, więc nawet nie musimy przeglądać wszystkich tablic do końca. Dodatkowo, żeby zaoszczędzić pamięć, nie potrzebujemy nigdzie zapisywać scalonej tablicy – interesuje nas tylko konkretny element, więc jedyne co będziemy musieli pamiętać, $T_1, T_2, T_3$, to trzy wskaźniki/indeksy, które będą nam mówiły, na jakim etapie przetwarzania tych tablic jesteśmy oraz jedną zmienną pamiętającą ile elementów musimy jeszcze przejść, żeby uzyskać medianę.

**Data:** $T_1, T_2, T_3$ – tablice, n – rozmiar tablic
**Result:** Mediana $\{T_1, T_2, T_3\}$
**Function** *mediana*$(T_1, T_2, T_3, n)$ **is**
    $i_1 := 0$
    $i_2 := 0$
    $i_3 := 0$
    **while** $i_1 + i_2 + i_3 < \lceil \frac{3n}{2} \rceil$ **do**
        **if** $T_1[i_1] \leq T_2[t_2]$ ***and*** $T_1[i_1] \leq T_3[t_3]$ ***and*** $i_1 < n$ **then**
            $i_1 += 1$;
        **else if** $T_1[i_1] \leq T_2[t_2]$ ***and*** $T_1[i_1] \leq T_3[t_3]$ ***and*** $i_1 < n$ **then**
            $i_2 += 1$;
        **else**
            $i_3 += 1$;
        **end**
    **end**
    **return** *min*$(T_1[i_1], T_2[i_2], T_3[i_3])$
**end**

Zmienne $i_1, i_2, i_3$ oznaczają ile elementów z każdej tablicy "wzięliśmy" do naszej scalonej, posortowanej tablicy. W każdym obrocie pętli wybieramy najmniejszy spośród $T_1[i_1], T_2[i_2], T_3[i_3]$, myślimy o nim jak o elemencie włożonym do scalonej tablicy i zwiększamy odpowiednie $i_k$. Skoro suma $i_1 + i_2 + i_3$

oznacza ile elementów sumarycznie włożyliśmy do scalonej tablicy, to medianę znajdziemy gdy będziemy próbowali scalić element o numerze $\lceil \frac{3n}{2} \rceil$, dlatego też przed nim kończymy pętlę i wybieramy do zwrócenia minimum spośród $T_1[i_1], T_2[i_2], T_3[i_3]$.

**Dowód poprawności**

Załóżmy że liczba $m$ którą zwrócił nasz algorytm nie jest medianą, jest nią natomiast liczba $p$. Musi więc zachodzić $m < p$ lub $m > p$.

- **m<p**: Z działania algorytmu i faktu posortowania tablic wiemy, że w $T_1, T_2, T_3$ istnieje conajmniej $\lceil \frac{3n}{2} \rceil - 1$ liczb mniejszych lub równych $m$, więc jest tam conajmniej $\lceil \frac{3n}{2} \rceil$ mniejszych od p. Dlatego p nie jest medianą.
- **p<m**: Analogicznie zauważmy, że w $T_1, T_2, T_3$ istnieje conajmniej:
  - $2 \mid n$: $\lceil \frac{3n}{2} \rceil$ liczb większych bądź równych $m$, więc istnieje $\lceil \frac{3n}{2} \rceil + 1$ liczb większych od $p$ – $p$ nie jest medianą;
  - $2 \nmid n$ $\lceil \frac{3n}{2} \rceil - 1$: liczb większych bądź równych $m$, więc istnieje $\lceil \frac{3n}{2} \rceil$ liczb większych od $p$ – $p$ nie jest medianą;

Z tego wynika, że nie istnieje takie $p$, więc algorytm działa poprawnie. Pozostaje jeszcze kwestia złożoności obliczeniowej. W każdym obrocie pętli algorytm wykona stałą liczbę operacji arytmetycznych i porównań – niech będzie to $c$. Pętla wykona $\lceil \frac{3n}{2} \rceil$ obrotów, więc złożoność to $O(\lceil \frac{3n}{2} \rceil) * c = O(n)$.