Mikołaj Korobczak

Algorytmy i struktury danych

Zadanie 6 lista 1

Wrocław, 6 kwietnia 2020

Prowadzący: mgr Adam Kunysz

1. Treść zadania

Dany jest niemalejący ciąg n liczb dodatnich $a_1 \leqslant a_2 \leqslant \cdots \leqslant a_n$. Wolno nam modyikować ten ciąg za pomocą następujących operacji: wybieramy dwa elementy a_i, a_j spełniające $2a_i \leqslant a_j$ i wykreślamy je oba z ciągu. Ułóż algorytm obliczający, ile co najwyżej elementów możemy w ten sposób usunąć.

2. Algorytm

Algorytm będzie zachłanny:
Dane: $A[1, \ldots, n]$

```
i = 1
j = (n div 2) + 1
counter = 0
while j <= n do
    if 2*A[i] <= A[j] then
        A[i] = -1
        A[j] = -1
        counter = counter + 1
        i = i + 1
    endif
    j = j + 1
done
return counter
```

3. Dowód poprawności

Weźmy dowolne rozwiązanie optymalne O. Załóżmy, że w tym rozwiązaniu usunięte zostało k par.

Lemat 1.

Weźmy dwa zbiory L i R takie, że dla dowolnej usuwanej pary (a_i, a_j) takiej, że $2a_i \leqslant a_j$, $a_i \in L$ i $a_j \in R$. Wtedy można zmienić te punkty w taki sposób, że:

1. $\forall_{a_i \in L} a_i < \min R$,
2. zbiór L jest postaci
$$L = \{a_1, a_2, \ldots, a_k\}$$
taki, że $\forall_{a_1 \leqslant a_i \leqslant a_k} a_i \in L$,

3. zbiór R to

$$R = \{a_{m+i}, a_{m+j}, \ldots, a_{m+l}\} \qquad \text{dla } m = \lfloor \frac{n}{2} \rfloor + 1 \text{ i pewnych } 0 \leqslant i < j < l.$$

Dowód.

1. Weźmy dowolne dwie pary punktów $(a_i, a_i'), (a_j, a_j')$ takie, że $a_i' \leqslant a_j$. Wtedy (a_i, a_j) oraz (a_i', a_j') są poprawnymi parami więc można je poukładać w naszych zbiorach w ten sposób.
2. Weźmy dowolny punkt a_i dla $i < k$ taki, że $a_i \notin L$. Weźmy najmniejszy punkt $a_j \in L$ taki, że $i < j \leqslant k$. Wtedy można punkt sparowany z nim sparować z naszym a_i, natomiast a_j sparować z parą kolejnego punktu z L i tak aż każdy będzie miał parę. Więc a_i będzie należał do L.
3. Załóżmy, że zbiór R składa się z punktów $(a_i, a_j, \ldots, a_k)$ takich, że $i < m$. Wtedy możemy zamienić te punkty na większe takie, że punkty zaczynają się od a_{m+l} dla $l \geqslant 0$. Wiemy, że jest to możliwe ponieważ $k \leqslant \frac{n}{2}$. W ten sposób dostajemy żądany zbiór R.

□

Korzystając z lematu 1 wystarczy pokazać, że nasz algorytm znajduje punktom $a_1, a_2, \ldots, a_k$ parę. Dowód będzie przez indukcję.

Dowód.
Baza:
a_1 zostanie sparowane z najmniejszym a_j takim, że $j \geqslant m = \lfloor \frac{n}{2} \rfloor + 1$, czyli znajdzie parę z najmniejszym punktem z R.
Krok:
Weźmy dowolne $i \leqslant k$ i załóżmy, że punkty $a_1, a_2, \ldots, a_{i-1}$ znajazły pary, z kolejnymi najmniejszymi punktami $a_1', a_2', \ldots, a_{i-1}'$ takimi, że są $\geqslant a_m$. Wtedy punkt a_i znajdzie parę z najmniejszym punktem po punkcie a_{i-1}', który spełni wymaganą nierówność. Taki punkt na pewno istnieje ponieważ w szczególności algorytm może brać punkty ze zbioru R, a wtedy napewno sparuje wszystkie punkty. □

Jednocześnie warto zaznaczyć, że algorytm nie może znaleść więcej niż k par, ponieważ nie jest to możliwe sprawdzając za każdym razem zadaną nierówność.

4. Złożoność czasowa i pamięciowa

Złożoność pamięciowa to $O(1)$ ponieważ rezerwujemy pamięć tylko dla zmiennych $i, j, counter$. Jeżeli założymy, że musimy zarezerwować pamięć dla naszej tablicy (ja zakładam, że dostajemy tablicę już gdzieś w pamięci), to złożoność pamięciowa to $O(n)$.
Złożoność czasowa to $O(n)$ poniważ w najgorszym przypadku przebiegamy po naszej tablicy A od $A[1]$ do $A[\lfloor \frac{n}{2} \rfloor]$ oraz od $A[\lfloor \frac{n}{2} \rfloor + 1]$ do $A[n]$.