AiSD, Konsultacje, 2021–11–16

Tomasz Waleń

Wydział Matematyki, Informatyki i Mechaniki, UW

1. Kolokwium 2017/18, Zadanie 1

W liczbowym, różnowartościowym ciągu $\langle a_1, a_2, \ldots, a_n \rangle$, $n > 2$, element a_i, $1 < i < n$, nazywamy lokalnym ekstremum gdy jest mniejszy lub większy od obu sąsiadów, tzn.

albo $a_{i-1} > a_i < a_{i+1}$, albo $a_{i-1} < a_i > a_{i+1}$

a) (3 punkty) Udowodnij, że każdy algorytm sortujący przez porównania 4-elementowe ciągi z co najwyżej jednym lokalnym ekstremum wymaga wykonania w pesymistycznym przypadku co najmniej 4 porównań.

b) (4 punkty) Zaproponuj algorytm sortowania 4-elementowych ciągów z co najwyżej 1 lokalnym ekstremum za pomocą co najwyżej 4 porównań

c) (4 punkty) Zaproponuj algorytm, asymptotycznie optymalny ze względu na liczbę porównań, sortujący ciągi o co najwyżej k lokalnych ekstremach dla zadanego k, $0 < k < n$. Dowiedź optymalności swojego rozwiązania.

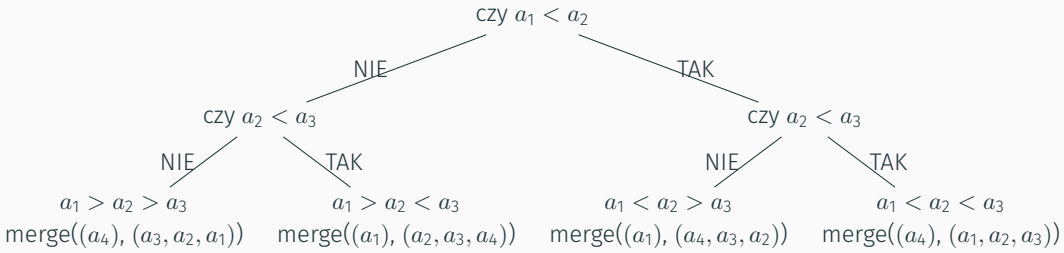
Rozwiązanie

a) wystarczy pokazać, że jest > 8 ciągów z co najwyżej jednym lokalnym ekstremum:

1: (1, 2, 3, 4) 2: (1, 2, 4, 3) 3: (1, 3, 4, 2) 4: (1, 4, 3, 2) 5: (2, 1, 3, 4) 6: (2, 3, 4, 1) 7: (2, 4, 3, 1) 8: (3, 1, 2, 4) 9: (3, 2, 1, 4) 10: (3, 4, 2, 1) 11: (4, 1, 2, 3) 12: (4, 2, 1, 3) 13: (4, 3, 1, 2) 14: (4, 3, 2, 1)

c) $O(n \log k)$ – scalanie k uporządkowanych ciągów (za pomocą kopca rozmiaru k).

Rozwiązanie



1. Kolokwium 2018/19, Zadanie 2

Wiadomo, że każdy algorytm wyznaczający przez porównania dwa elementy – największy i najmniejszy – w ciągu długości $n = 2k$, wykonuje w pesymistycznym przypadku co najmniej $3k - 2$ porównania. Udowodnij powyższe dla $n = 4$ i zaproponuj optymalny algorytm w tym przypadku.

Rozwiązanie

(Knuth, Tom 3, ćwiczenie 16, strona 231). Niech (a, b, c, d) oznacza stan obliczeń algorytmu,

- a — liczba elementów, które nie były jeszcze porównywane,
- b — liczba elementów, które były porównywane i nie przegrały żadnego porównania,
- c — liczba elementów, które były porównywane i przegrały wszystkie porównania,
- d — liczba elementów, które wygrały co najmniej jedno porównanie, i przegrały co najmniej jedno porównanie.

Dla $n = 4$ algorytm wyznaczający MIN-MAX startuje w stanie $(4, 0, 0, 0)$ kończy w stanie $(0, 1, 1, 2)$.

1. Kolokwium 2018/19, Zadanie 3

Elementy w tablicy $a[1..n]$, $n > 0$, są rozmieszczone w porządku kopcowym typu MIN. Dla $k > 0$ tablicę a rozszerzono o k elementów $a[n+1], ..., a[n+k]$. Zaproponuj algorytm, który w czasie $O(k + \log^2 n)$ zbuduje kopiec typu MIN na całej tablicy $a[1..n+k]$.

Uwaga: rozpocznij swoje rozważania przy założeniu, że $n = 2^{h-1}$, dla pewnego $h > 0$.

Rozwiązanie

- jeśli $k = \Omega(n)$ to zbuduj w czasie $O(k)$ kopiec na elementach $a[1, \ldots, n+k]$,
- w przeciwnym przypadku:
  - musimy poprawić poprawność kopca dla $O(k + \log n)$ węzłów,
  - wykonaj operację downheap (idąc od dołu do góry) dla wszystkich wierzchołków w których poddrzewie występują węzły $a[n+1, \ldots, n+k]$
  - to zajmuje czas $O(k)$ (węzły w ktorych obu poddrzewach występują węzły z $a[n+1, \ldots, n+k]$)
  - oraz $O(\log^2 n)$ ponieważ, jest $O(\log n)$ węzłów, które mają tylko w jednym poddrzewie klucze z $a[n+1, \ldots, n+k]$.

Zadanie nieznane

Udowodnij, że do posortowania n-elementowego ciągu, który zawiera k różnych elementów $a_1, \ldots, a_k$ i każdy element a_i występuje n_i razy, trzeba wykonać co najmniej $\log(\frac{n!}{(n_1! \cdot \ldots \cdot n_k!)})$ porównań.

Rozwiązanie

Ile jest ciągów spełniających warunki zadania?

$$\binom{n}{n_1} \cdot \binom{n-n_1}{n_2} \cdot \binom{n-n_1-n_2}{n_3} \cdot \ldots \cdot \binom{n-n_1-\ldots-n_{k-1}}{n_k}$$

$$\frac{n!}{n_1!(n-n_1)!} \cdot \frac{(n-n_1)!}{n_2!(n-n_1-n_2)!} \cdot \frac{(n-n_1-n_2)!}{n_3!(n-n_1-n_2-n_3)!} \cdots$$

1. Kolokwium 2015/16, Zadanie 2

Ile wynosi maksymalna liczba zamian w fazie budowy kopca (od dołu) dla 2-uporządkowanego ciągu $a[1..n]$, gdzie $n = 2^k - 1$, dla pewnego $k > 0$? Odpowiedź uzasadnij.

Rozwiązanie

Dla $n = 15$ to będzie:

$$8, 1, 9, 2, 10, 3, 11, 4, 12, 5, 13, 6, 14, 7, 15$$

Ogólnie to będzie przeplot ciągów:

- $2^{k-1}, \ldots, 2^k - 1$
- $1, \ldots, 2^{k-1} - 1$

Taka tablica a wymaga $2^{k-1} - 1$ zamian.