Lista 3
Zadanie 3

Treść

Tablica A indeksowana $[1, \ldots, n]$ do pewnego momentu k jest posortowana malejąco, a po k rosnąco. Podaj algorytm znajdujący $min(A)$ o złożoności obliczeniowej $O(\log n)$.

Rozwiązanie

Będziemy operować na przedziałach $[L, R]$ z $L\ i\ R$ włącznie, $L\ <=\ R, L\ i\ R$ to indeksy elementów z tablicy.

Sprawdzamy czy tablica nie jest pusta, jeśli jest możemy zakończyć algorytm.

Na początek przyjmujemy $[L, R]\ =\ [1, n]$.

Powtarzamy dopóki $L \neq R$:

- Bierzemy $index$ = $\lfloor\frac{L+R}{2}\rfloor$
- Jeśli $A[index]\ <\ A[index\ +\ 1]\ \ R\ =\ index$ (może istnieć element mniejszy od $A[index]$, ale na pewno nie leży w $[index + 1, n]$)
- W przeciwnym przypadku $L\ =\ index\ +\ 1$ ($A[index]$ nie jest najmniejszy i na pewno w $[1, index]$ go nie ma)

Kończymy pętle z przedziałem $[x, x]$, czyli jednoelementową podtablicą i element $A[x]$ jest rozwiązaniem.

Zawsze możemy sprawdzać $index + 1$, ponieważ w pętli tablice są przynajmniej dwuelementowe i $\left\lfloor\frac{x+x+1}{2}\right\rfloor = x$, czyli w przypadku dwuelementowym $index\ =\ L$ i $index\ +\ 1\ =\ R$.

Dowody

1) Poprawność przez indukcje, gdzie k oznacza przesunięcie w tablicy tzn. chcemy pokazać poprawność algorytmu bez względu na wybór indeksów ($[1,2,3]$ czy $[5,6,7]$ nie powinno robić różnicy).
   - Dla tablicy pustej i jednoelementowej sytuacja jest trywialna

   - $n = 2$ i niech m oznacza element minimalny, $L = 1+k, R = 2+k, index = 1+k$:
   $[m, x]$ $m < x$, więc $R = 1+k$ i schodzimy do poprawnej tablicy $[1+k, 1+k]$
   $[x, m]$ $x > m$, więc $L = 2+k$ i schodzimy do poprawnej tablicy $[2+k, 2+k]$

   - $n = 3$ $L = 1+k, R = 3+k, index = 2+k$
   $[m, x, y]$ $x < y$, więc $R = 2+k \rightarrow [m, x]$
   $[x, m, y] m < y, R = 2+k \rightarrow [x, m]$
   $[x, y, m]$ $y > m, L = 3+k$ schodzimy do poprawnej sytuacji jednoelementowej

   - Niech dla $3 < n < j$ algorytm działa

   - $n = j, L = 1+k, R = n+k$, x jest elementem pod $index$
   $[\dots, m, \dots, x, y, \dots]$ x może być m, $x < y, R = index \rightarrow$ redukuje do mniejszej tablicy z rozwiązaniem
   $[\dots, x, y, \dots, m, \dots]$ y może być $m, x > y, L = index + 1 \rightarrow$ redukuje do mniejszej z m w środku
   Algorytm redukuje tablice poprawnie do mniejszej, a dla mniejszych tablic algorytm działa, więc indukcja spełniona.
   Dla n parzystego redukujemy zawsze do $\frac{n}{2}$
   Dla n nieparzystego redukujemy do $\left\lfloor\frac{n}{2}\right\rfloor$ lub $\left\lfloor\frac{n}{2}\right\rfloor + 1$
   W ostateczności zredukujemy do sytuacji $n = 3$ lub $n = 2$, które dają poprawny wynik.

2) Złożoność $O(\log n)$:
   Jeden podział to jedno porównanie, dla każdego podziału redukujemy długość tablicy średnio o 2, więc $\frac{n}{2^k} \approx 1$, gdzie k to ilość podziałów=porównań, czyli $k = \log_2 n$.
   Bardziej formalnie:
   $$T(n) = aT(\frac{n}{b}) + f(n)$$
   Na jeden podział potrzebujemy jedno porównanie:
   $$a = 1, b = 2, \log_b a = 0 , f(n) = \Theta\left(n^{(\log_b a)}\right) = \Theta(1)$$

   Z Master Theorem wiemy, że w takiej sytuacji:

   $$T(n) = \Theta\ (n^{(\log_b a)} * \log n) = \Theta(\log n) = O(\log n)$$