ćwiczenia (licencjat wieczorowy): lista zadań nr 4+5 *6 listopada 2007 r.*

# ALGORYTMY I STRUKTURY DANYCH

## PODZIAŁ, SORTOWANIE SZYBKIE, WYBÓR $k$–TEGO ELEMENTU

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

1. [$*$] Czasem mamy do czynienia z danymi, których wartości często się powtarzają. Jak zmodyfikować procedurę podziału, aby dokonywała ona tak zwanego *trójpodziału* względem wybranego elementu $x$: na początku ma się znaleźć blok z elementami $< x$, potem blok z elementami $= x$, a na końcu blok z elementami $> x$. Czas działania twojego algorytmu powinien być liniowy.

2. [$**$] W $n$–elementowej tablicy $A[0 \ldots n-1]$ zapisany jest nieuporządkowany ciąg $(a_0, a_1, \ldots, a_{n-1})$. Zaprojektuj rekurencyjny algorytm, który *stabilnie* rozdzieli dane w tej tablicy względem określonej wartości piwota $x$ i nie będzie używał dodatkowych buforów pomodniczych na dane. Czas działania twojego algorytmu powinien być nie gorszy niż $O(n \log n)$.

3. [$*$] Rozważmy algorytm *sortowania szybkiego*, w którym do podziału zastosujemy procedurę *Sedgewick–partition*. Pokaż, że jeśli wszystkie elementy sortowanej tablicy mają taką samą wartość, to wtedy czas działania procedury *quick–sort* wynosi $\Theta(n \, logn)$.

4. [$*$] Rozważmy algorytm *sortowania szybkiego*, w którym podział danych następuje względen piwota wziętego z początku tablicy. Pokaż, że jeśli elementy sortowanej tablicy są ustawione w porządku malejącym, to wtedy czas działania procedury *quick–sort* wynosi $\Theta(n^2)$.

5. [$**$] Jak zmodyfikować algorytm *quick–sort*, aby w każdym przypadku głębokość wywołań rekurencyjnych była nie większa niż $\log n$ dla danych rozmiaru $n$?

   *Wskazówka.* Jedno z wywołań rekurencyjnych rozwiń w miejscu.

6. [$**$] Wiemy, że dolna granica na liczbę wykonywanych porównań przez dowolny algorytm znajdujący minimum w $n$–elementowym zbiorze wynosi $n-1$. Dolna granica na liczbę wykonywanych porównań przez dowolny algorytm znajdujący vice–minimum w $n$–elementowym zbiorze wynosi $n + \lceil \log n \rceil - 2$. Skonstruuj algorytm znajdujący vice–minimum w nieuporządkowanym $n$–elementowym zbiorze pamiętanym w tablicy, działający optymalnie.

7. [$*$] Mamy dostęp do algorytmu *czarna skrzynka*, który w liniowym czasie znajduje medianę nieuporządkowanego ciągu liczb. W jaki sposób, korzystając z *czarnej skrzynki*, wyznaczyć dowolny $k$–ty co do wielkości element w nieuporządkowanym ciągu liczb. Twój algorytm powinien działać w czasie liniowym.

8. [$**$] Opisz algorytm, który wyznaczy medianę spośród 5 elementów, wykonując w najgorszym przypadku co najwyżej 6 porównań.

9. [$***$] Czas działania *algorytmu magicznych piątek* można oszacować zliczając porównania, które on wykonuje. Liczbę tych porównań można określić następującym wzorem:

$$T(n) \leq \begin{cases} \frac{3}{8}n^2 & \text{dla} \quad n < 15 \\ 6\lfloor \frac{n}{5} \rfloor + T(\lfloor \frac{n}{5} \rfloor) + n + T(n - 3\lceil \frac{n-4}{10} \rceil) & \text{dla} \quad n \geq 15 \end{cases}$$

   Oszacuj z dokładnością do stałego czynnika liczbę wykonywanych przez ten algorytm porównań.

   *Komentarz.* Przypomnienie algorytmu magicznych piątek. Dla małych danych ($n < 15$) wystarczy posortować połowę tablicy algorytmem bąbelkowym. Dla większych danych ($n \geq 15$) wykonujemy cztery kroki: *(i)* wyznaczmy mediany ciągów 5–elementowych za pomocą 6 porównań, *(ii)* rekurencyjnie wyznaczany medianę spośród tych median, *(iii)* dokonujemy trójpodziału względen wyznaczonej wartości *(iv)* i rekurencyjnie rozwiązujemy problem dla jednej z części.