ćwiczenia (licencjat wieczorowy): lista zadań nr 3 *23 października 2007 r.*

ALGORYTMY I STRUKTURY DANYCH

SORTOWANIE LINIOWE, SCALANIE

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

1. [$**$] W n–elementowej tablicy zapisane są liczby wymierne postaci $\frac{p}{2^q}$, gdzie $0 \leq p, q \leq 10$. W oparciu o algorytm *counting–sort* skonstuuj metodę, która posortuje te liczby w liniowym czasie.

2. [$*$] Jak należy zmodyfikować algorytm *bucket–sort*, aby w najgorszym przypadku działał w czasie $O(n \log n)$?

3. [$**$] Skonstruuj w oparciu o algorytm *radix–sort* metodę, która pozwala posortować n liczb całkowitych ze zbioru $\{0, 1, \ldots, n^{-}1\}$ w czasie $O(n)$.

4. [$**$] Dane są dwa posortowane ciągi $A = (a_0, a_1, \ldots, a_{m-1})$ i $B = (b_0, b_1, \ldots, b_{m-1})$ zapisane w jednej $(m + n)$–elementowej tablicy (najpierw ciąg A a potem B). Pokaż, jak zaimplementować algorytm *scalania* takich ciągów, który będzie korzystał z pomocniczego bufora o rozmiarze $\min\{m, n\}$.

5. [$***$] Dane są dwa posortowane ciągi $A = (a_0, a_1, \ldots, a_{m-1})$ i $B = (b_0, b_1, \ldots, b_{m-1})$ zapisane w jednej $(m + n)$–elementowej tablicy (najpierw ciąg A a potem B). Zaprojektuj rekurencyjny algorytm, który będzie scalał *stabilnie* takie ciągi i nie będzie używał dodatkowych buforów pomodniczych na dane. Czas działania twojego algorytmu powinien być nie gorszu niż $O(n \log n)$.

6. [$**$] W przedstawionej na wykładzie rekurencyjnej wersji algorytmu *sortowania przez scalanie* dane wejściowe o rozmiarze n były dzielone na dwie równe z dokładnością do 1 części o rozmiarach odpowiednio $\lfloor \frac{n}{2} \rfloor$ i $\lceil \frac{n}{2} \rceil$. Jak zmieni się złożoność czasowa tego algorytmu, gdy dane wejściowe będziemy dzielić na trzy części o rozmiarach $\lfloor \frac{n}{3} \rfloor$, $\lfloor \frac{n+1}{3} \rfloor$ i $\lfloor \frac{n+2}{3} \rfloor$?