ćwiczenia (licencjat wieczorowy): lista zadań nr 13 *15 stycznia 2008 r.*

ALGORYTMY I STRUKTURY DANYCH

METODA „DZIEL I ZWYCIĘŻAJ”

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

1. [∗] Rozważmy pewną modyfikację *sortowania głupiego*, która polega na przepychaniu na koniec (połączonym z sortowaniem) $\frac{1}{4}n$ największych elementów (operację tą powtarzamy trzykrotnie):

```
procedure STOOGE–SORT–4 (key T[0 ... n−1])
{
    if n < 8 then { BUBBLE–SORT (T); return; }
    p1 ← ⌈n/4⌉; p2 ← 2⌈n/4⌉; p3 ← 3⌈n/4⌉;
    /* pierwszy etap przepychania z sortowaniem */
    STOOGE–SORT-4 (t[0 ... p2−1]);
    STOOGE–SORT–4 (t[p1 ... p3−1]);
    STOOGE–SORT–4 (t[p2 ... n−1]);
    /* drugi etap przepychania z sortowaniem */
    STOOGE–SORT–4 (t[0 ... p2−1]);
    STOOGE–SORT–4 (t[p1 ... p3−1]);
    /* trzeci etap przepychania z sortowaniem */
    STOOGE–SORT–4 (t[0 ... p2−1]);
}
```

   Uzasadnij poprawność tego algorytmu i oszacuj jego złożoność korzystając z *uniwersalnego twierdzenia o rekurencji*.

2. [∗∗] Dane są dwie posortowane n-elementowe tablice liczb oraz liczba całkowita k z zakresu $1 \leq k \leq 2n$. Opisz algorytm znajdowania k-tej co do wielkości liczby spośród liczb zapisanych w obu tablicach. Czas działania twojego algorytmu powinien być rzędu $O(\log n)$. Uzasadnij jego poprawność.

3. [∗∗] Jak uogólnić algorytm mnożenia długich liczb metodą „dziel i zwyciężaj”, aby długości liczb nie musiały być potęgami 2, a także by ich długości mogły być różne.

   Uwaga. W zadaniu nie chodzi o to, by dopisać tyle zer, aby długości liczb powiększyły się do potęg 2.

4. [∗∗] Niech dane będą dwie długie liczby w systemie d-arnym: jedna o długości m i druga o długości n. Załóżmy, że $m \leq n$. Gdy $m \simeq n$, to algorytm pisemny oblicza iloczyn takich liczb w czasie $O(n^2)$, a algorytm z wykładu oparty na technice „dziel i zwyciężaj” potrzebuje tylko $O(n^{\log 3})$ czasu. Ale gdy m jest znacznie mniejsze niż n, to również ten ostatni algorytm jest nie do zaakceptowania. Skonstruuj algorytm, który dla przypadku gdy $m \ll n$ będzie działał w czasie $O(n\, m^{\log \frac{3}{2}})$.

5. [∗∗∗] Zmodyfikuj algorytm mnożenia długich liczb metodą „dziel i zwyciężaj” w ten sposób, aby liczba była dzielona na trzy części zamiast na dwie. Ile mnożeń krótszych liczb musisz wtedy wykonać? Jaka jest złożoność czasowa twojego algorytmu?

   Uwaga. Asymptotyczny czas działania algorytmu z podziałem na trzy części powinien być lepszy niż z podziałem na dwie części.

6. [∗∗] Niech A będzie macierzą o wymiarach $k\,n \times n$ a B macierzą o wymiarach $n \times k\,n$, gdzie $k \in \mathbf{N}$. Jak szybko można obliczyć iloczyn $A \cdot B$ korzystając z algorytmu Strassena? O ile lepszy jest ten wynik od tradycyjnego mnożenia macierzy? Odpowiedz na te same pytania dla iloczynu $B \cdot A$.

7. [∗∗] Wyznacz wartość progową (*threshold*) opłacalności stosowania algorytmu Strassena w stosunku do tradycyjnego algorytmu mnożenia macierzy. Za operacje dominujące przyjmij działania arytmetyczne na elementach macierzy (dodawanie, odejmowanie i mnożenie).