Algorytmy i Struktury Danych

IIUWr. II rok informatyki.

1. (**P** 1pkt) Ułóż, oparty o zasadę dziel i zwyciężaj, algorytm obliczający największy wspólny dzielnik dwóch liczb, który wykorzystuje następującą własność:

$$gcd(a,b) = \begin{cases} 2gcd(a/2, b/2) & \text{gdy } a, b \text{ są parzyste,} \\ gcd(a, b/2) & \text{gdy } a \text{ jest nieparzyste a } b \text{ jest parzyste,} \\ gcd((a-b)/2, b) & \text{gdy } a, b \text{ są nieparzyste} \end{cases}$$

□

Porównaj złożoność tego algorytmu z algorytmem Euklidesa.

2. (2pkt) Przeanalizuj następujący algorytm oparty na strategii dziel i zwyciężaj jednoczesnego znajdowania maksimum i minimum w zbiorze $S = \{a_1, \ldots, a_n\}$:

```
Procedure MaxMin(S:set)
    if |S|= 1 then return {a1, a1}
    else
        if |S|= 2 then return (max(a1, a2), min(a1, a2))
        else
            podziel S na dwa równoliczne (z dokładnością do jednego elementu) podzbiory S1, S2
            (max1, min1) ← MaxMin(S1)
            (max2, min2) ← MaxMin(S2)
            return (max(max1, max2), min(min1, min2))
```

Uwaga: Operacja **return** $(\max(a_1, a_2), \min(a_1, a_2))$ wykonuje jedno porównanie.

- Jak pokażemy na jednym z wykładów każdy algorytm dla tego problemu, który na elementach zbioru wykonuje jedynie operacje porównania, musi wykonać co najmniej $\lceil \frac{3}{2}n - 2 \rceil$ porównania. Dla jakich danych powyższy algorytm wykonuje tyle porównań? Podaj wzorem wszystkie takie wartości.
- Jak bardzo może różnić się liczba porównań wykonywanych przez algorytm od dolnej granicy?
- Popraw algorytm, tak by osiągał on tę granicę dla każdej wartości n?

3. (1,5pkt) Podaj algorytm scalania dwóch otoczek wypukłych, będący częścią algorytmu znajdowania otoczki wypukłej zbioru punktów na płaszczyźnie, podanego na wykładzie.

4. (2pkt)
   - (**P**) Dane jest nieukorzenione drzewo z naturalnymi wagami na krawędziach oraz liczba naturalna C. Ułóż algorytm obliczający, ile jest par wierzchołków odległych od siebie o C.
   - (**Z**) Jak wyżej, ale algorytm ma działać w czasie $O(n \log n)$.

5. (2pkt) Macierz A rozmiaru $n \times n$ nazywamy macierzą Toeplitza, jeśli jej elementy spełniają równanie $A[i,j] = A[i-1, j-1]$ dla $2 \leq i, j \leq n$.

(a) Podaj reprezentację macierzy Toeplitza, pozwalającą dodawać dwie takie macierze w czasie $O(n)$.

(b) Podaj algorytm, oparty na metodzie "dziel i zwyciężaj", mnożenia macierzy Toeplitza przez wektor. Ile operacji arytmetycznych wymaga takie mnożenie?

6. (2pkt) *Medianą* n elementowego wielozbioru A nazywamy wartość tego elementu z A, który znalazłby się na pozycji $\lceil n/2 \rceil$ po uporządkowaniu A według porządku $\leq$. Ułóż algorytm, który dla danych uporządkowanych niemalejąco n-elementowych tablic $T1, T2, T3$ znajduje medianę wielozbioru A utworzonego ze wszystkich elementów tych tablic.

7. (2pkt) Jakie jest prawdopodobieństwo wygenerowania permutacji identycznościowej przez sieć Beneša-Waksmana, w której przełączniki ustawiane są losowo i niezależnie od siebie w jeden z dwóch stanów.

Zadania dodatkowe - nie będą rozwiązywane w czasie ćwiczeń

1. (0 pkt) Przypomnij sobie algorytm scalający dwie posortowane tablice U i V w czasie liniowym, tj. w czasie liniowo proporcjonalnym do sumy długości tych tablic.

2. (1 pkt) Złożoność podanego na wykładzie algorytmu sortowania przez scalenia wyraża się wzorem:

$$\begin{array}{ll} T(1) = a & \\ T(n) \leq T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + bn & \text{dla } n > 1 \end{array}$$

dla pewnych $a, b > 0$. Udowodnij, że $T(n) \in O(n \log n)$.

3. (1 pkt) Zamiast dzielić tablicę T na dwie połówki, algorytm sortowania przez scalanie mógłby dzielić ją na części o rozmiarach $\lceil n/3 \rceil$, $\lceil (n+1)/3 \rceil$ oraz $\lceil (n+2)/3 \rceil$, sortować niezależnie każdą z tych części, a następnie scalać je. Podaj bardziej formalny opis tego algorytmu i przeanalizuj czas jego działania.

4. (1pkt) Rozważ wersje algorytmu *muliply* dzielące czynniki na trzy i cztery części. Oblicz współczynniki w kombinacjach liniowych określających wartości c_i.

5. (1 pkt) Niech u i v będą liczbami o n i m cyfrach (odpowiednio). Załóżmy, że $m \leq n$. Klasyczny algorytm oblicza iloczyn tych liczb w czasie $O(mn)$. Algorytm *multiply* z wykładu potrzebuje $O(n^{\log 3})$ czasu, co jest nie do zaakceptowania gdy m jest znacznie mniejsze od n. Pokaż, że w takim przypadku można pomnożyć liczby u i v w czasie $O(nm^{\log(3/2)})$.

6. (1pkt) Udowodnij, że podana na wykladzie sieć przełączników (sieć Beneša-Waksmana) jest asymptotycznie optymalna pod względem głębokości i liczby przełączników.

7. (1pkt) Załóżmy, że przełączniki ustawiane są losowo (każdy przełącznik z jednakowym prawdopodobieństwem ustawiany jest w jeden z dwóch stanów). Sieć zbudowaną z takich przełączników można traktować jako generator losowych permutacji. Udowodnij, że nie istnieje sieć przełączników, generująca permutacje z rozkładem jednostajnym.

8. (2pkt) Przeanalizuj sieć permutacyjną omawianą na wykładzie (tzw. sieć Beneša-Waksmana)
   - Pokaż, że ostatnią warstwę przełączników sieci Beneša-Waksmana można zastąpić inną warstwą, która zawiera $n/2 - 1$ przełączników (a więc o jeden mniej niż w sieci oryginalnej) a otrzymana sieć nadal będzie umożliwiać otrzymanie wszystkich permutacji.
   - Uogólnij sieć na dowolne n (niekoniecznie będące potęgą liczby 2).

Krzysztof Loryś