Algorytmy i Struktury Danych

IIUWr. II rok informatyki.

1. (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?

2. (2pkt) Danych jest n prostych $l_1, l_2, \ldots l_n$ na płaszczyźnie ($l_i = a_i x + b_i$), takich że żadne trzy proste nie przecinają się w jednym punkcie. Mówimy, że prosta l_i jest widoczna z punktu p jeśli istnieje punkt q na prostej l_i, taki że odcinek $\overline{pq}$ nie ma wspólnych punktów z żadną inną prostą l_j ($j \neq i$) poza (być może) punktami p i q.

   Ułóż algorytm znajdujący wszystkie proste widoczne z punktu $(0,+\infty)$.

3. (1,5pkt) *Otoczką wypukłą* zbioru P, punktów na płaszczyźnie, nazywamy najmniejszy wielokąt wypukły zawierający (w swoim wnętrzu lub na brzegu) wszystkie punkty z P. Naturalny, oparty na zasadzie dziel i zwyciężaj, algorytm znajdowania otoczki wypukłej dla zbioru P, dzieli P na dwa (prawie) równoliczne podzbiory (np. "pionową" prostą), znajduje rekurencyjnie otoczki wypukłe dla tych podzbiorów, a następnie scala te otoczki. Podaj algorytm wykonujący tę ostatnią fazę algorytmu, tj. algorytm scalania dwóch otoczek wypukłych.

4. (1,5pkt) Dane jest drzewo binarne (możesz założyć dla prostoty, że jest to pełne drzewo binarne), którego każdy wierzchołek v_i skrywa pewną liczbę rzeczywistą x_i. Zakładamy, że wartości skrywane w wierzchołkach są różne. Mówimy, że wierzchołek v jest minimum lokalnym, jeśli wartość skrywana w nim jest mniejsza od wartości skrywanych w jego sąsiadach.

   Ułóż algorytm znajdujący lokalne minimum odkrywając jak najmniej skrywanych wartości.

5. Dane jest nieukorzenione drzewo z naturalnymi wagami na krawędziach oraz liczba naturalna C.

(a) (2pkt) Ułóż algorytm obliczający, ile jest par wierzchołków odległych od siebie o C.

(b) (**Z** 2,5pkt) Jak w punkcie (a), ale algorytm ma działać w czasie $O(n \log n)$.

Uwaga: Można zadeklarować tylko jeden z punktów (a), (b).

6. (1,5pkt) 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?

7. (1pkt) *Inwersją* w ciągu $A = a_1, \ldots, a_n$ nazywamy parę indeksów $1 \leq i < j \leq n$, taką że $a_i > a_j$. Pokaż jak można obliczyć liczbę inwersji w A podczas sortowania.

8. (1,5pkt) 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).

9. (2pkt) Niech P_n będzie zbiorem przesunięć cyklicznych ciągu n-elementowego o potęgi liczby 2 nie większe od n. Pokaż konstrukcję sieci przełączników realizujących przesunięcia ze zbioru P_n.

   Uwagi:

   - Możesz założyć, że n jest potęgą dwójki albo szczególną potęgą dwójki, albo ...
   - Sieć Beneša-Waksmana jest dobrym rozwiązaniem wartym 0pkt (tzn. nic niewartym).