Lista zadań. Nr 4. 11 kwietnia 2019

Algorytmy i Struktury Danych

IIUWr. II rok informatyki.

1. (0pkt) Rozwiąż wszystkie zadania dodatkowe.
2. (1pkt) Ułóż algorytm znajdujący najtańszą drogę przejścia przez tablicę, w którym oprócz ruchów dopuszczalnych w wersji problemu prezentowanej na wykładzie, dozwolone są także ruchy w górę i w dół tablicy.
3. (2pkt) Rozważmy następujące operacje na ciągach:
   - *insert(x,i,a)* - wstawienie a pomiędzy i-tym i $(i+1)$-szym elementem x-a;
   - *delete(x,i)* - usunięcie i-tego elementu x-a;
   - *replace(x,i,a)* - zastąpienie i-tego elementu x-a przez a.

   Jak łatwo zauważyć, dla każdych dwóch ciągów x i y istnieją sekwencje powyższych operacji przekształcające x w y. Jeśli każdej operacji przypiszemy koszt (nieujemną liczbę rzeczywistą) możemy mówić o minimalnym koszcie przekształcenia x w y (koszt ten nazywa się *odległością edycyjną* ciągów x i y).

   Ułóż algorytm, który dla danych dwóch ciągów znajdzie ich odległość edycyjną.
4. (1pkt) Zbiór $I \subseteq V$ zbioru wierzchołkow w grafie $G = (V, E)$ nazywamy *zbiorem niezależnym*, jeśli żadne dwa wierzchołki z I nie są połączone krawędzią. Ułóż algorytm, który dla zadanego drzewa T znajduje najliczniejszy zbiór niezależny jego wierzchołków.
5. (2pkt) Na trzyelementowym zbiorze $A = \{a, b, c\}$ określono operację $\odot$. Nie jest ona ani przemienna ani łączna. Ułóż algorytm, który dla danego ciągu $x_1 \odot x_2 \odot \ldots \odot x_n$, gdzie x_i są symbolami ze zbioru A, rozstrzyga, czy można w nim tak rozstawić nawiasy, by wartość otrzymago wyrażenia wynosiła a.
6. (2pkt) Dany jest graf pełny $G = (V, E)$ z nieujemnymi wagami na krawędziach oraz ciąg wszystkich jego wierzchołków $C = \langle v_1, \ldots, v_n \rangle$. Początkowo w wierzchołku v_1 znajdują się dwa pionki. W kolejnych ruchach masz przesunąć pionki według następujących zasad:
   - w każdym ruchu przesuwasz jeden pionek,
   - pionek stojący w wierzchołku v_i możesz być przesunąć do wierzchołka v_j jedynie wtedy, gdy $j > i$ (czyli do wierzchołka znajdującego się dalej w ciągu C),
   - wszystkie wierzchołki grafu muszą być odwiedzone przez co najmniej jeden pionek,
   - po ostatnim ruchu obydwa pionki znajdują się w wierzchołku v_n.

   Ułóż algorytm obliczający ciąg ruchów pionków minimalizujący sumę długości dróg przebytych przez pionki (przez długość drogi rozumiemy sumę wag jej krawędzi).
7. (1pkt) Ułóż algorytm, który dla danych liczb całkowitych $a_1, \ldots, a_n \in \langle -C..C \rangle$ sprawdza, czy zbiór $\{1, 2, \ldots, n\}$ można podzielić na trzy rozłączne podzbiory I, J, K, takie, że

$$\sum_{i \in I} a_i = \sum_{j \in J} a_j = \sum_{k \in K} a_k.$$

8. (1pkt) Na każdym polu szachownicy o wymiarach $4 \times n$ znajduje się jedna liczba naturalna. Ułóż algorytm, który umieszcza na szachownicy kamyki w taki sposób, że:

- na każdym polu znajduje się co najwyżej jeden kamień,
- jeśli na polu P znajduje się kamyk, to na polach mających wspólny bok z P nie ma kamyków,
- suma liczb z pól, na których leżą kamyki jest maksymalna.

9. (2pkt) Ułóż algorytm, który znajduje LCS dwóch ciągów długości n w czasie $O(n^2)$ używając pamięci rozmiaru $O(n)$.

10. (2pkt) Ułóż algorytm rozwiązujący poniższy problem triangulacji wielokąta wypukłego:

    PROBLEM:
    Dane: ciąg par liczb rzeczywistych $(x_1, y_1), \ldots, (x_n, y_n)$, określających kolejne wierzchołki n-kąta wypukłego P
    ZAŁOŻENIE: dane są określone poprawnie.
    Zadanie: Znaleźć zbiór S nieprzecinających się przekątnych, które dzielą P na trójkąty, taki, że długość najdłuższej przekątnej w S jest możliwie najmniejsza.

11. (**Z** 3pkt) Dana jest permutacja $p_1, p_2, \ldots, p_n$ oraz liczba k. Chcemy skonstruować k parami rozłącznych rosnących podciągów tej permutacji, które mają jak największą sumę długości. Ułóż algorytm, który rozwiązuje ten problem w czasie wielomianowym. Częściowe punkty można uzyskać za ułożenie algorytmu, który wypisuje tylko największą możliwą sumę długości, ale nie konstruuje odpowiadających jej podciągów.

Zadania dodatkowe do rozwiązania samodzielnego lub podczas repetytorium

1. (1pkt) Uzupełnij podany na wykładzie algorytm sprawdzający przynależność słowa do języka generowanego przez bezkontekstową gramatykę w normalnej postaci Chomsky'ego tak, by w przypadku pozytywnej odpowiedzi wypisywał jego wyprowadzenie.

2. (1pkt) Jak zmieni się złożoność problemu przynależność słowa do języka generowanego przez bezkontekstową gramatykę w normalnej postaci Chomsky'ego, jeśli gramatyka także będzie daną wejściową?

3. (2pkt) *Gramatyką liniową* nazywamy gramatykę bezkontekstową, w której prawe strony produkcji zawierają co najwyżej jeden symbol nieterminalny. Ułóż algorytm sprawdzający przynależność słowa do języka liniowego, który wykorzystuje pamięć rozmiaru $O(n)$.

4. (2pkt) Dana jest szachownica $n \times n$ i pozycje pionów na niej (może być ich nawet $O(n^2)$). Ułóż algorytm znajdujący prostokątny fragment szachownicy o największym polu, na którym nie znajduje się ani jeden pion. Twój algorytm powinien działać w czasie $O(n^2)$.

5. (2pkt) Napisz w pseudopascalu lub pseudoC++ dwie procedury:

   (a) drukującą ciąg nazw macierzy wraz z poprawnie rozstawionymi nawiasami wyznaczającymi optymalną kolejność mnożenia macierzy,

   (b) drukującą ciąg instrukcji postaci $A \leftarrow B \times C$, prowadzących do obliczenia w optymalny sposób iloczynu macierzy (A jest nazwą macierzy roboczej, a B i C - są nazwami macierzy wejściowych lub wcześniej obliczonych macierzy roboczych).

6. (1pkt) Udowodnij, że liczba wywołań rekurencyjnych w poniższej procedurze obliczającej minimalny koszt pomnożenia macierzy jest $\Theta(3^n)$.

   function $minmat(i,j)$
       if $i = j$ **then** **return** 0
       $ans \leftarrow \infty$
       for $k \leftarrow i$ **to** $j-1$ **do**
           $ans \leftarrow \min(ans,\ \ d_{i-1}d_kd_j + minmat(i,k) + minmat(k+1,j))$
       return ans

7. (2pkt) Jak wiesz, liczba wszystkich poprawnych rozstawień n par nawiasów (a więc i sposobów pomnożenia n macierzy) jest równa n-tej liczbie Catalana.

   - Wykaż, że liczba ta rośnie szybciej niż 3^n.
   - Czy potrafisz wskazać poprawne rozstawienie nawiasów, które nie jest rozważane przez procedurę *minmat*?

8. (2pkt) Dany jest zbiór n przedmiotów $A = \{a_1, \ldots, a_n\}$. Dla każdego przedmiotu znamy jego wagę $w(a_i)$ oraz jego cenę $c(a_i)$; obie te liczby są naturalne a $c(a_i))$ jest nie większe od n^2. Dana jest ponadto liczba naturalna P. Ułóż algorytm znajdujący podzbiór S zbioru przedmiotów, taki, że suma wag przedmiotów z S nie przekracza P, a suma ich cen jest możliwie największa. W jakim czasie działa Twój algorytm?

9. (1pkt) Pokaż jak obliczyć długość elementów LCS używając jedynie $2\min(m,n)$-elementowej tablicy c plus $O(1)$ dodatkowej pamięci. Następnie pokaż, jak to zrobić używając $\min(m,n)$-elementowej tablicy c plus $O(1)$ dodatkowej pamięci

10. (2pkt) Zmodyfikuj algorytm znajdujący najdłuższy wspólny podciąg dwóch ciągów n elementowych, tak by działał w czasie $O(n^2)$ i używał $O(n)$ pamięci.

11. (1pkt) Zmień podany na wykładzie algorytm znajdujący najtańszą drogę przejścia przez tablicę tak, by znajdował drogę o drugim co do wielkości koszcie.

12. (2pkt) Podwójną drabiną rozmiaru n nazywamy graf przedstawiony na poniższym rysunku.

Rysunek 1: *Podwójna drabina n-elementowa*

Uogólnij na podwójne drabiny podany na wykładzie algorytm znajdujący liczbę drzew rozpinających o k krawędziach wyróżnionych.

13. (2pkt) Nad pewną rzeką płynącą ze wschodu na zachód zamieszkały bobry: na południowym brzegu - panowie; na północnym - panie. Tak się złożyło, że panów bobrów jest tyle samo co pań i ponadto każdy pan zapałał uczuciem do jednej pani (szczęśliwie każdy do innej). Każdy z panów chce zbudować groblę prowadzącą z jego żeremia do żeremia wybranki. Problem w tym, że groble nie mogą się krzyżować. Rada starszych postanowiła ustalić, ile maksymalnie grobli może powstać. Ułóż algorytm, który wykona to zadanie.

Krzysztof Loryś