Algorytmy i Struktury Danych

1. (0pkt) Rozwiąż wszystkie zadania dodatkowe.

2. (1pkt) Jak zmieni się złożoność problemu przynależnośc 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. (1pkt) Algorytm znajdujący parę najmniej odległych punktów w trakcie przeglądania tablicy Y' sprawdza 7 punktów. Dokonaj modyfikacji algorytmu redukujących tę liczbę. Oczywiście modyfikacje te nie powinny zwiększać asymptotycznej złożoności algorytmu.

5. (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 nieprzecinających się przekątnych, które dzielą P na trójkąty i których sumaryczna długość jest minimalna.

6. (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 $O(n^2)$.

7. (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.

Zadania grupy zaawansowanej

1. (3pkt) *Gramatyką rosnącą* nazywamy taką gramatykę, w której każda produkcja ma lewą stronę krótszą od prawej. Ułóż algorytm, który dla dowolnej, ale ustalonej, gramatyki rosnącej G sprawdza, czy dane słowo należy do $L(G)$.

2. (2pkt) Jaka jest złożoność problemu przynależności do języka generowanego przez gramatykę rosnącą, gdy gramatyka jest daną wejściową?

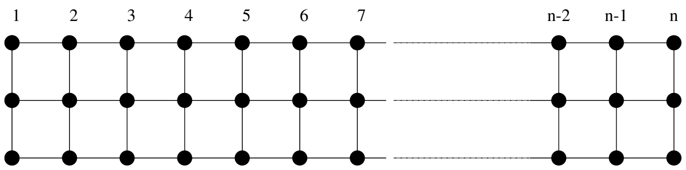
3. (3pkt) Ułóż algorytm sprawdzający przynależność słowa do języka liniowego, który wykorzystuje pamięć rozmiaru $f(n) \in o(n)$ Dla przypomnienia $o(n)$ jest klasą tych funkcji f, dla których $lim_{n \to \infty} f(n)/n = 0$).

Zadania dodatkowe - nie będą rozwiązywane w czasie ćwiczeń

1. (1pkt) Uzupełnij podany na wykładzie algorytm sprawdzający przynależnosc słowa do języka generowanego przez bezkontekstową gramatykę w normalnej postaci Chomsky'ego tak, by w przypadku pozytywnej odpowiedzi wypisywał jego wyprowadzenie.
2. (2pkt) Napisz w pseudopascalu 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).
3. (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

4. (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*?
5. (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
6. (2pkt) Zmodyfikuj algorytm znajdujący najdłuższy wspólny podciąg dwóch ciągów n elementowych, tak by działał w czasie $O(n^2)$ i używał $O(n)$ pamięci.
7. (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.

Krzysztof Loryś