Lista zadań. Nr 2. 16 lutego 2005

ALGORYTMY I STRUKTURY DANYCH

IIUWr. II rok informatyki.

1. (2pkt) Taśma magnetyczna zawiera n plików o długościach $l_1, l_2, ..., l_n$. Znamy częstości p_i żądań załadowania i-tego pliku do pamięci ($i = 1, 2 \dots n$). (Oczywiście $\sum_{i=1}^{n} p_i = 1$.) Po wykonaniu każdej operacji załadowania pliku, taśma jest przewijana na początek. Jeśli pliki pamiętane są w kolejności $i_1, i_2, \dots, i_n$, to średni czas ładowania pliku wynosi

$$\bar{T} = c \sum_{j=1}^{n} \left[p_{i_j} \sum_{k=1}^{j} l_{i_k} \right],$$

gdzie stała c zależy jedynie od gęstości zapisu i szybkości przewijaka taśmy (zakładamy, że oba te parametry są stałe). Chcemy zminimalizować wielkość $\bar{T}$. Podaj zachłanny algorytm znajdujący optymalne rozmieszczenie plików na taśmie.

2. (2pkt) Udowodnij poprawność algorytmu Sollina. Zaproponuj jego implementację. Jaka jest jej złożoność czasowa?

3. (2pkt) Rozważamy grafy skierowane, w których każda para wierzchołków połączona jest przynajmniej jedną krawędzią. Podaj algorytm wyznaczający dla takiego grafu *ścieżkę Hamiltona*, tj. ścieżkę przechodzącą dokładnie jeden raz przez wszystkie wierzchołki. Udowodnij, że Twój algorytm działa poprawnie.

4. (2pkt) Udowodnij, że w grafie nieskierowanym o n wierzchołkach, w którym każdy wierzchołek ma co najmniej $n/2$ sąsiadów, istnieje ścieżka Hamiltona. Podaj algorytm, który dla takich grafów znajduje tę ścieżkę.

5. (2pkt) Dla ważonego drzewa $T = (V, E; c)$, gdzie $c : V \to \mathcal{R}_+$, określamy jego *zewnętrzną długość* $EL(T)$ jako:

$$EL(T) = \sum_{v-\text{liść } T} c(v) \cdot d(v),$$

gdzie $d(v)$ jest długością ścieżki od korzenia do liścia v (mierzoną liczbą krawędzi na ścieżce).

Rozważmy następujący problem. Dany jest n-elementowy zbiór $\{w_1, \dots, w_n\}$ dodatnich liczb rzeczywistych. Zadaniem jest znalezienie ważonego drzewa binarnego T o n liściach, takiego, że każda liczba w_i jest wagą dokładnie jednego liścia oraz T ma minimalną wagę $EL(T)$ poŚród wszystkich drzew o tej własności.

6. (2pkt) Trójki uporządkowane $\langle a_1, a_2, a_3 \rangle$ i $\langle b_1, b_2, b_3 \rangle$ są *rozłączne*, jeśli $\forall_{1 \le i,j \le 3}\ a_i \neq b_j$. Mówimy, że zbiór parami rozłącznych trójek uporządkowanych $C = \{u_1, \dots u_m : \text{ gdzie } \forall_{i=1,\dots,m}\ u_i \in \mathcal{N} \times \mathcal{N} \times \mathcal{N}\}$

pokrywa odcinek $[1, n]$, jeśli każda liczba naturalna z tego odcinka występuje jako składowa jednej z trójek zbioru C.

Ułóż algorytm, który dla zadanych liczb naturalnych n, k, l wyznacza zbiór C uporządkowanych trójek pokrywający odcinek $[1, n]$ o następującej własności:

$$\forall_{\langle a,b,c \rangle \in C}\ [b = a + k \text{ i } c = b + l] \text{ lub } [b = a + l \text{ i } c = b + k].$$

Czy dla każdych danych taki zbiór istnieje?

Zadania nadobowiązkowe - nie będą rozwiązywane w czasie ćwiczeń

1. (1pkt) Wykaż, że zachłanny algorytm wydawania reszty, w sytuacji gdy monety mają nominały $c^0, c^1, \ldots, c^k$ dla pewnych stałych naturalnych $c > 1$ i $k \geq 1$, daje optymalne rozwiązanie.

2. (1pkt) Przypomnij sobie algorytmy Prima i Kruskala znajdowania minimalnego drzewa rozpinającego. Udowodnij poprawność jednego z nich.

3. (1pkt) Ułóż algorytm rozwiązujący następujący problem szeregowania zadań dla c procesorów:
   Dane: t_i - czas obsługi i-tego zadania ($i = 1, 2, ..., n$).
   Problem: każde z zadań przypisać do jednego z c procesorów oraz ustalić kolejność wykonywania zadań przez każdy z procesorów tak, by zminimalizować wartość:
   $T = \sum_{i=1}^{n}$ (czas przebywania i-tego zadania w systemie).

   Udowodnij, że Twój algorytm zawsze znajduje optymalne rozwiązanie.

   Jaki jest czas działania Twojego algorytmu?

4. (1pkt) Ułóż algorytm rozwiązujący następujący problem szeregowania zadań dla c procesorów:
   Dane: t_i - czas obsługi i-tego zadania ($i = 1, 2, ..., n$).
   Problem: każde z zadań przypisać do jednego z c procesorów oraz ustalić kolejność wykonywania zadań przez każdy z procesorów tak, by zminimalizować wartość:
   $T = \sum_{i=1}^{n}$ (czas przebywania i-tego zadania w systemie).

   Udowodnij, że Twój algorytm zawsze znajduje optymalne rozwiązanie.

   Jaki jest czas działania Twojego algorytmu?

5. (1pkt) Przypomnij sobie algorytm Dijkstry znajdowania najkrótszych ścieżek od zadanego wierzchołka do wszystkich pozostałych wierzchołków. Udowodnij jegoi poprawność.

Problemy (chyba) trudne

1. Podaj warunek konieczny i wystarczający na to, by dla danego ciągu nominałów monet podany na wykładzie algorytm zachłanny wydawania reszty był poprawny.

2. Jaka jest złożoność następującego problemu?
   Problem.
   dane: ciąg $C = c_1, \ldots, c_k$
   wynik: "TAK" - jeśli zachłanny algorytm wydawania reszty dla ciągu C i dowolnej wartości reszty daje optymalne rozwiązanie;
   "NIE" - w przeciwnym przypadku.

Krzysztof Loryś