Algorytmy i Struktury Danych

1. (1pkt) Danych jest n odcinków $I_j = \langle p_j, k_j \rangle$, leżących na osi OX, $j = 1, \ldots, n$. Ułóż algorytm znajdujący zbiór $S \subseteq \{I_1, \ldots, I_n\}$, nieprzecinających się odcinków, o największej mocy.

2. (1pkt) System ma do wykonania n zleceń. Dla każdego zlecenia znamy czasy $p_j, k_j \in R_+$ - odpowiednio początku i końca realizacji zlecenia. Każde zlecenie ma być wykonywane przez jeden procesor. Ułóż algorytm obliczający, ile procesorów jest potrzebnych do wykonania wszystkich zleceń.

3. (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 ($i = 1, 2 \ldots n$). (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 $i_1, i_2, \ldots, i_n$, 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.

4. (2pkt) Udowodnij poprawność algorytmu Sollina. Zaproponuj jego implementację. Jaka jest jej złożoność czasowa?

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

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

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

   Rozważmy następujący problem. Dany jest n-elementowy zbiór $\{w_1, \ldots, w_n\}$ 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 $\forall_{1 \leq i,j \leq 3}\ a_i \neq b_j$. Mówimy, że zbiór parami rozłącznych trójek uporządkowanych $C = \{u_1, \ldots u_m :\ \text{gdzie}\ \forall_{i=1,\ldots,m}\ u_i \in \mathcal{N} \times \mathcal{N} \times \mathcal{N}\}$

   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 dodatkowe dla grupy zaawansowanej

1. (2pkt) Na wykładzie przedstawiono zachłanny algorytm dla następującego problemu *Pokrycia zbioru*:

Dane:	rodzina $S = \{S_1, \ldots, S_k\}$ podzbiorów zbioru $\{1, \ldots, n\}$ oraz funkcja $c : S \to \mathcal{R}_+$.
Rozwiązanie:	podrodzina $S' \subseteq S$, taka, że $\cup_{T \in S'} T = \{1, \ldots, n\}$.
Cel:	Znaleźć rozwiązanie optymalne, tj. minimalnym koszcie, zdefiniowanym jako $c(S') = \sum_{T \in S'} c(T)$.

Przedstawiona strategia konstruuje rozwiązanie startując od $S' = \emptyset$. W kolejnej iteracji do S' dodawany jest ten z podzbiorów S_i, dla którego wartość wyrażenia $c(S_i)/|S_i \setminus \cup_{T \in S'} T|$ jest minimalna. Postępujemy tak do czasu aż $\cup_{T \in S'} T = \{1, \ldots, n\}$ (ten moment nastąpi, ponieważ zakładamy, że $\cup_{T \in S} T = \{1, \ldots, n\}$.

- Udowodnij, że powyższy algorytm daje rozwiązania, które są co najwyżej $\log n$ razy gorsze od rozwiązania optymalnego.
- Pokaż, że istnieją dane, dla których rozwiązania znajdowane przez powyższy algorytm są $\Omega(\log n)$ gorsze od rozwiązań optymalnych.

Uwaga: jeśli powyższy algorytm był analizowany na innych Twoich zajęciach, uprzedź o tym fakcie prowadzącego.

Zadania dodatkowe - 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) Udowodnij, że spójny graf, którego wszystkie krawędzie mają różne wagi, posiada dokładnie jedno minimalne drzewo spinające.

3. (1pkt) Udowodnij poprawność algorytmu Prima.

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) 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?

6. (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.

7. (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.

8. (1pkt) Przypomnij sobie algorytm Dijkstry znajdowania najkrótszych ścieżek od zadanego wierzchołka do wszystkich pozostałych wierzchołków. Udowodnij jego 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