Lista zadań. Nr 7. 6 czerwca 2020

Algorytmy i Struktury Danych

IIUWr. II rok informatyki.

1. (2pkt) Niech $A = \{a_1, a_2, \ldots, a_n\}$ będzie zbiorem kluczy, (takim, że $\forall_{i=1,\ldots,n-1} a_i < a_{i+1}$), które chcemy pamiętać w słowniku stałym. Znamy także ciąg $p_1, \ldots, p_n$ prawdopodobieństw zapytania o poszczególne klucze. Przyjmujemy, że $p_1 + p_2 + \ldots + p_n = 1$, a więc do słownika nie będą kierowane zapytania o klucze spoza słownika. Chcemy zaimplementować słownik jako drzewo BST.

   Ułóż algorytm znajdujący takie drzewo, które minimalizuje oczekiwany czas wykonywania operacji na słowniku.

   - (**P**) Algorytm ma działać w czasie $O(n^3)$.
   - (**Z**) Algorytm ma działać w czasie $O(n^2)$.

2. (2pkt) Ułóż algorytm rozwiązujący problem znajdowania najbliżej położonej pary punktów na płaszczyźnie oparty na następującej idei. Niech d będze odległością pomiędzy parą najbliżej położonych punktów spośród punktów $p_1, p_2, \ldots, p_{i-1}$. Sprawdzamy, czy p_i leży w odległości mniejszej niż d, od któregoś z poprzednich punktów. W tym celu dzielimy płaszczyznę na odpowiednio małe kwadraty, tak by w każdym z nich znajdował się nie więcej niż jeden punkt. Te "zajęte" kwadraty pamiętamy w słowniku.

   Twój algorytm powinien działać w oczekiwanym czasie liniowym. Jeśli nie potrafisz zbudować algorytmu opartego na powyższej idei, możesz opracować algorytm oparty na innej (ale spełniający te same wymagania czasowe).

3. (1pkt) Oblicz jaka jest oczekiwana liczba pustych list po umieszczeniu n kluczy w tablicy haszującej o n elementach.

4. (1pkt) Przeprowadź analizę zamortyzowanego kosztu ciągu operacji *insert, deletemin, decrease-key, meld, findmin* wykonywanych na kopcach Fibonacciego, w których kaskadowe wykonanie operacji *cut* wykonywane jest dopiero wtedy, gdy wierzchołek traci trzeciego syna.

5. (**Z** 2pkt) Rozważ implementację kopców Fibonacciego, w której zamiast przechowywać w każdym wierzchołku bit oznaczający, że jedno z jego dzieci zostało już odcięte, za każdym razem patrzymy na (świeży) losowy bit. Udowodnij, że oczekiwany zamortyzowany czas operacji *insert* , *decrease-key* jest stały, a zamortyzowany oczekiwany czas operacji *delete-min* jest polilogarytmiczny względem liczby już wykonanych operacji.

Zadania dodatkowe - do samodzielnego rozwiązywania

1. (1pkt) Uzasadnij stwierdzenie, że funkcje haszujące potrzebne w konstrukcji słownika stałego (podanej na wykładzie) mogą być efektywnie znalezione.

2. (2pkt) Opracuj algorytm, który dla danych macierzy A, B i C o wymiarach $n \times n$ sprawdza, czy C jest równa iloczynowi $a \cdot B$. Twój algorytm powinień dawać poprawną odpowiedź z dużym prawdopodobieństwem.