Algorytmy i Struktury Danych, 6. ćwiczenia

2018-11-07

Spis treści

1 d-kopce

d–kopiec do drzewo zupełne o stopniu d z porządkiem kopcowym (min w korzeniu). Należy pokazać, że poszczególne operacje wykonuje się w czasie:

- Min — $O(1)$
- DeleteMin — $O(d \cdot \log_d(n))$
- DecreaseKey — $O(\log_d(n))$

Koszt implementacji algorytmu Dijkstry, przy użyciu d–kopców: $O(nd \cdot \log_d(n) + m \cdot \log_d(n))$.

Zanalizować jak należy dobrać d w zależności od m i n (jeśli za d weźmiemy $max(2, \lceil m/n \rceil)$ to dostajemy $O(\frac{m \log n}{\log m/n})$).

2 System różnych reprezentantów

Dana jest rodzina n niepustych podzbiorów zbioru $\{1, 2, \ldots, n\}$, z których każdy to całkowitoliczbowy przedział postaci $[i, j]$, $i \leq j$. Zaprojektuj efektywny algorytm sprawdzania, czy zadana rodzina posiada system różnych reprezentantów, a jeśli tak, to podaje jeden z nich.
Możemy udowodnić, że następujący algorytm zachłanny rozwiązuje problem:

- dane: n przedziałów $[l_i, r_i]$,
- niech K oznacza kopiec zawierający przedziały uporządkowane rosnąco według prawych końców, początkowo kopiec jest pusty
- $y = 1$

- for $i \in 1, \ldots, n$ do:
  - dodaj do kopca wszystkie przedziały postaci $[i, r_k]$,
  - jeśli kopiec nie jest pusty, to niech $[l_r, r_k] = ExtractMin(K)$
  - jeśli $y > r_k$ to zakończ algorytm — BRAK ROZWIĄZANIA
  - w przeciwnym przypadku, przydziel jako reprezentanta $[l_r, r_k]$ wartość $max(l_r, y)$
  - $y := max(l_r, y) + 1$

3 Rozgłaszanie komunikatów

Dane drzewo T, należy obliczyć czas potrzebny na przesłanie komunikatów do wszystkich węzłów drzewa. Przesłanie komunikatu po jednej krawędzi zajmuje 1 jednostkę czasu.
Algorytm $O(n \log n)$:

- jeśli wierzchołek jest liściem to $czas = 0$,
- wpp. rekurencyjnie oblicz czas potrzebny na rozgłoszenie w poddrzewach,
- posortuj malejąco otrzymane czasy: $t_1, \ldots, t_k$
- $czas = max\{i + t_i : 1 \leq i \leq k\}$

Aby otrzymać algorytm $O(n)$ trzeba sprytnie obliczać wartości atrybutu $czas$.

- $Q = \{$ liście $T \}$,
- while $root \notin Q$ do
  - $x = Q.extractMin()$
  - dodaj $x.czas$ do kolejki $parent(x)$,
  - jeśli $parent(x)$ ma już pełną listę poddrzew, to policz $parent(x).czas$ i dodaj $parent(x)$ do kolejki.

Kolejkę Q można zaimplementować w tablicy (i–ty element tablicy zawiera listę wierzchołków o wartości $x.czas = i$). Sumarycznie operacje $extractMin$ zajmą czas $O(n)$. Dodawanie do kolejki zajmuje czas $O(1)$.

4 Dijkstra z ograniczonymi wagami

Aby otrzymać czas $O(NW + M)$ potrzebujemy kolejki priorytetowej o następujących czasach wykonania poszczególnych operacji:

- ExtracMin — $O(W)$
- DecreaseKey — $O(1)$

Wystarczy zauważyć, że jeśli do jakiegoś wierzchołka istnieje droga, to jej długość jest $\leq NW$. Czyli potrzebujemy tablicy NW elementowej (i–ty element tablicy zawiera listę nieodwiedzonych wierzchołków w odległości i od wierzchołka początkowego).