Lista 2, zadanie 9 - Tomasz Woszczyński

Treść: Dla ważonego drzewa $T = (V, E;\ c)$, gdzie $c : V \to \mathbb{R}_+$, określamy jego zewnętrzną długość $EL(T)$ jako

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

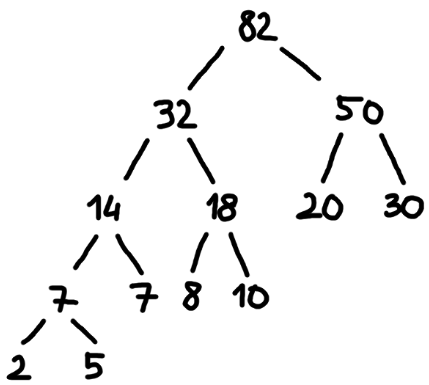
gdzie $d(v)$ jest długością ścieżki od korzenia do liścia v (mierzoną liczbą krawędzi na ścieżce). Rozważmy następujący problem. Dany jest n-elementowy zbiór $\{w_1, \ldots, w_n\}$ dodatnich liczb rzeczywistych. Zadaniem jest znalezienie ważonego drzewa binarnego T o n liściach, takiego, że każda liczba w_i jest wagą dokładnie jednego liścia oraz T ma minimalną wagę $EL(T)$ pośród wszystkich drzew o tej własności.

Informacje: Mamy dane $\{w_1, \ldots, w_n\}$ i wiemy, że $\forall i : w_i \in \mathbb{R}_+$, $c(v)$ jest wagą liścia v, a $d(v)$ długością ścieżki od korzenia do liścia v. Niech Q będzie kolejką priorytetową z wagami, dla każdego wierzchołka możemy stworzyć pewnego rodzaju strukturę postaci $v(w_i, \text{NULL}, \text{NULL})$, gdzie w_i jest wagą wierzchołka, a dwie ostatnie wartości (tj. NULL) są odpowiednio lewym i prawym poddrzewem wierzchołka v. Disclaimer do $Q.pop()$: bierzemy wartość pierwszego elementu, przypisujemy go do zmiennej l lub r, a następnie usuwamy pierwszy element w Q.

Algorytm:

Wrzuć do Q wagi liści;
while $|Q| > 1$ **do**
  $l := Q.pop()$;
  $p := Q.pop()$;
  $Q.push(l.w + p.w, l, p)$;
end
return $Q.pop()$;

Przykład: Weźmy liście o wagach $\{5, 7, 2, 10, 8, 20, 30\}$. Wtedy kolejka przed wejściem w pętle $Q = \{2, 5, 7, 8, 10, 20, 30\}$. Po pierwszym przejściu pętli będziemy mieć $Q = \{7, 7, 8, 10, 20, 30\}$, po drugim $Q = \{8, 10, 14, 20, 30\}$, po trzecim $Q = \{14, 18, 20, 30\}$, aż w kolejce zostanie jeden element (korzeń drzewa), który umieścimy na samym końcu w procesie budowania drzewa. Poniżej rysunek przedstawia uzyskane drzewo T (budujemy je od dołu).



Złożoność czasowa: Algorytm działa w czasie $O(nlogn)$, jako że pętla działa w czasie $O(n)$, a wstawienie elementu do kolejki prioretytowej w czasie $O(logn)$. Mnożąc te dwie wartości (jako że wstawienie elementu odbywa się w każdej iteracji pętli) daje nam wcześniej wspomnianą złożoność $O(nlogn)$.

Dowód znajdowania minimalnego drzewa: Niech T będzie pełnym drzewem binarnym, oznaczmy jego zewnętrzną długość jako $EL(T)$, zbiór jego liści jako $L(T)$ oraz weźmy dwa wierzchołki u, v o najmniejszych wagach. Możemy wtedy skonstruować drzewo T' takie, że:

- jego zbiór liści to $L(T') = L(T) - \{u, v\} \cup \{w\}$, gdzie w to nowo powstały wierzchołek poprzez połączenie dwóch o najmniejszych wagach,
- $w(w) = w(u) + w(v)$,
- $EL(T') \leqslant EL(T) - w(u) - w(v)$ oraz $EL(T') = EL(T) - w(u) - w(v)$, gdy u i v są rodzeństwem.

Podważmy istnienie drzewa T'. Gdy wierzchołki u oraz v są sąsiadami, możemy w prosty sposób stworzyć drzewo T' z T poprzez usunięcie u oraz v i dodanie w, czyli nowego liścia będącego ich rodzicem. Wtedy

$$\begin{aligned} EL(T') &= EL(T) - w(u) \cdot d(u) - w(v) \cdot d(v) + w(w) \cdot d(w) = \\ &= EL(T) - w(u) \cdot x - w(v) \cdot x + w(w) \cdot (x-1) = \\ &= EL(T) - w(u) - w(v) \end{aligned}$$

zakładając, że $x = d(u) = d(v) = d(w) + 1$, a więc długości ścieżek od korzenia do u oraz do v są równe.
W przeciwnym wypadku załóżmy bez straty ogólności, że $d(u) \geqslant d(v)$ i niech r będzie rodzeństwem u, który niekoniecznie jest liściem. Zamieniamy wtedy miejscami v z r, a wiedząc że $d(r)$ jest co najmniej tak długa jak ścieżka z u, to zewnętrzna długość $EL(T)$ może tylko zmaleć. Wtedy wykonujemy kroki opisane powyżej.

Dowód znajdowania optymalnego wyniku: Załóżmy, że drzewo T uzyskujemy poprzez działanie naszego algorytmu i weźmy inne drzewo U, dla którego zbiory liści i wag są równe tym zbiorom drzewa T. Wtedy $EL(T) \leqslant EL(U)$. Udowodnimy ten lemat indukcyjnie. Dla dwóch liści w drzewie jest to oczywiste, gdyż zachodzi równość. W przeciwnym wypadku mamy do czynienia z większą ilością liści w drzewie, wybierzmy więc liście u oraz v wybrane przez algorytm. Konstruujemy wtedy drzewa T' oraz U', takie że

- $EL(T') = EL(T) - w(u) - w(v)$,
- $EL(U') \leqslant EL(U) - w(u) - w(v)$.

T' jest drzewem tworzonym przez algorytm, załóżmy że ma on $n-1$ liści, wtedy możemy skorzystać z założenia indukcyjnego, dzięki czemu otrzymamy $EL(T') \leqslant EL(U')$, a więc ostatecznie mamy

$$\begin{aligned} EL(T) &= EL(T') + w(u) + w(v) \leqslant \\ &\leqslant EL(U') + w(u) + w(v) \leqslant \\ &\leqslant EL(U) \end{aligned}$$

co kończy dowód. ■