AISD - Lista 8 Zadanie 4

4 czerwca 2020

1 Zadanie

Podaj algorytm o złożoności liniowej dla następującego problemu:

Input: Graf $G = (V, E)$ z wagami na krawędziach (wagi mogą być ujemne), wierzchołek $s \in V$, drzewo $T = (V, E'), E' \subset E$.
Output: $True$ jeśli T jest drzewem najkrótszych ścieżek od s w grafie G, w przeciwnym wypadku $False$.

2 Rozwiązanie

Aby sprawdzić czy T jest drzewem najkrótszych ścieżek od s w grafie G wykorzystam otrzymane drzewo T do uzupełnienia tablicy odległości algorytmu *Bellmana-Forda*. Przyjmuję, że T faktycznie jest drzewem i nie występują w nim żadne cykle. Aby stworzyć tablicę odległości $D[]$ z T:

1. Ustawiam wszystkie wartości D na MAX_INT poza $D[s]$, które ustawiam na 0.

2. Wykonuję procedurę BFS rozpoczynając od s. Podczas odwiedzania każdego z sąsiadów x wierzchołka y ustawiam $D[x] := D[y] + length(y, x)$. Ponieważ przechodzimy przez skierowane drzewo, nie trzeba się przejmować sprawdzaniem czy dany wierzchołek był już odwiedzany.

Teraz gdy mamy tablicę $D[]$ możemy za jej pomocą wykonać procedurę *update* (relaksacji krawędzi) algorytmu *Bellmana-Forda*. Ponieważ procedura *update* znajduje możliwe poprawy do aktualnej tablicy odległości $D[]$, brak zmian po jej zastosowaniu informuje nas, że wszystkie najkrótsze ścieżki zostały odnalezione i można zakończyć algorytm *Bellmana-Forda*. Oznacza to też, że jeżeli po wykonaniu procedury *update* nastąpią jakiekolwiek zmiany tablicy $D[]$, tablica wygenerowana z drzewa T nie była tablicą minimalnych odległości, a to bezpośrednio dowodzi, że T nie jest drzewem najkrótszych ścieżek. W przeciwnym wypadku, jeżeli zmiany nie nastąpią, T jest drzewem najkrótszych ścieżek.

2.1 Algorytm

Q - kolejka FIFO
$D \leftarrow [MAX\_INT, ...]$
$D[s] \leftarrow 0$

Q.push(s)
while Q not empty **do** ▷ Procedura BFS aby uzyskać $D[]$
    $y \leftarrow Q$.pop()
    for all $(y, x) \in E'$ **do**
        $D[x] \leftarrow D[y] + length(y, x)$
        Q.push(x)
    end for
end while

for all $(u, v) \in E$ **do** ▷ Procedura *update* (relaksacja krawędzi)
    if $D[v] > D[u] + length(u, v)$ **then**
        return $False$ ▷ Zmiana podczas relaksacji, T nie jest SPT
    end if
end for
return $True$ ▷ Nie ma zmiany podczas relaksacji, T jest SPT

2.2 Złożoność

Podczas przejścia BFS przez drzewo T i stworzenia $D[]$ przechodzimy przez E' krawędzi i maksymalnie V wierzchołków. Daje to złożoność $O(V + E')$ dla tej operacji. Operację *update* musimy wykonać tylko raz dla E krawędzi, co daje złożoność $O(E)$. Sumując obydwie operacje dostajemy złożoność $O(V + E' + E)$. Ponieważ $E' \subset E$, możemy to uprościć do $O(V + E)$.