Lista 6 z AiSD
Zadanie 3

Kamil Banaś 308262

16 czerwca 2020

1 Treść

Podaj algorytm sprawdzający izomorfizm drzew nieukorzenionych.

2 Rozwiązanie

W zadaniu zakładamy, że oba drzewa mają n wierzchołków (jeżeli mają różną liczbę, to nie mogą być izomorficzne).

Znamy algorytm sprawdzający izomorfizm drzew ukorzenionych (z wykładu). Wiemy, że algorytm taki działa w czasie $O(n)$. Wystarczy zatem pokazać jakiś sposób ukorzenienia w wierzchołku wspólnym dla obu drzew.

Mamy dwie możliwości: wybranie centroidów lub wybranie centrów.

2.1 Centroid

Centroid - wierzchołek, którego usunięcie dzieli drzewo na las, w którym każde drzewo ma co najwyżej połowę wierzchołków oryginalnego drzewa. W drzewie mogą być maksymalnie dwa centroidy.

2.1.1 Algorytm

Przedstawmy algorytm wyszukiwania centroidu w drzewie:

Puśćmy DFS od arbitralnie wybranego korzenia. Dla każdego rozpatrywanego wierzchołka sprawdźmy, czy wielkość dowolnego poddrzewa jest większa od $n/2$. Jak już sprawdziliśmy wszystkie poddrzewa, to od n odejmujemy sumę wszystkich poddrzew oraz aktualnego wierzchołka i ponownie porównujemy z $n/2$ - dzięki temu uzyskujemy wielkość poddrzewa zawierającego korzeń dla tego wierzchołka. Taką sumę zapisujemy w tablicy $size$ jako wielkość podrzewa z danym wierzchołkiem jako korzeniem. Jeżeli uzyskaliśmy jakąkolwiek liczbę większą od $n/2$ - aktualny wierzchołek nie jest centroidem, w przeciwnym przypadku jest.

W kwestii implementacyjnej: jest do standardowy DFS z tablicą $size$ trzymającą wielkość danego podrzewa.

2.1.2 Dowód poprawności

Udowodnijmy, że poprawnie wybraliśmy centroid lub dwa centroidy powyższym algorytmem. Widzimy, że zarówno w przypadku wybrania złego centroidu lub nie wybrania aktualnego centroidu błędem byłoby złe policzenie poddrzew - czyli niepoprawne obliczenie tablicy $size$. Wiemy, że DFS odwiedza każdy wierzchołek i wykonuje obliczenia rekurencyjnie od liści. Zatem niezmiennikiem będzie poprawne wartości w tablicy $size$.

- dla liścia l mamy $size[l] = 1$
- dla dowolnego drzewa zakładamy, że poddrzewa mają dobrze wyliczone wielkości w tablicy $size$. Zatem wielkością takiego drzewa jest suma $size$ dla poddrzew dodać korzeń - tak samo, jak w algorytmie.

2.1.3 Złożoność

Wyszukanie centroidu sprowadza się do puszczenia DFS po drzewie - otrzymujemy złożonośc $O(n)$.

2.2 Centrum

Centrum - wierzchołek, którego największa odległość do dowolnego wierzchołka jest jak najmniejsza. W drzewie mogą być maksymalnie dwa centra. Centrum leży na najdłuższej ścieżce w drzewie.

2.2.1 Algorytm

Możemy spróbować szukać najdłuższej ścieżki i środkowego wierzchołka/środkowych wierzchołków, ale zaproponuję inny algorytm.

Zasadniczo w algorytmie usuwamy za każdym razem wszystkie liście w drzewie, dopóki nie zostaną jeden lub dwa wierzchołki, które będą szukanymi centrami.

Szczególniej to na kolejkę wrzucamy wszystkie liście. W momencie usunięcia liścia sprawdzamy, czy jego sąsiadujący wierzchołek stał się liściem; jak tak, to wrzucamy ten wierzchołek na kolejkę i ustawiamy jego wartość jako wartość usuniętego wierzchołka dodać jeden. Jak kolejka stanie się pusta to wypisujemy wierzchołki (jeden lub dwa) o największej wartośći.

2.2.2 Dowód poprawnośći

Wystarczy udowodnić następujący lemat:

Lemat: Usunięcie wszystkich aktualnych liści z drzewa nie zmienia jego centrum/centrów.
Dowód: Usunięcie liści nie wpływa na odległości pomiędzy pozostałymi wierzchołkami. Każda najdłuższa ścieżka w drzewie została skrócona o dwa wierzchołki - zatem najdłuższa ścieżka nie została zmieniona jak i pozycja środkowego/środkowych wierzchołków na niej, czyli centrum drzewa.

2.2.3 Złożoność

Każdy wierzchołek zostaje dodany do kolejki i z niej usunięty, przy czym jest wykonywana stała liczba działań/przepisań, zatem złożość czasowa to $O(n)$.

2.3 Ostateczny Algorytm

Dla obu drzew znajdujemy centroid/centroidy lub centrum/centra w czasie $O(n)$. Jeżeli drzewa nie mają takiej samej liczby centroidów lub centrów, to nie są izomorficzne. W przeciwnym przypadku wystarczy wykonać algorytm sprawdzający izomorfizm drzew ukorzenionych, gdzie korzenie to znalezione centroidy/centra. Jeżeli są tylko jedne w drzewach, to wykonujemy algorytm tylko raz. W przeciwnym przypadku z jednego drzewa wybieramy jeden centroid/centrum i w razie potrzeby wykonujemy algorytm dla obu centroidów/centrów drugiego drzewa.

Złożonośc czasowa to $O(n)$.