Algorytmy i Struktury Danych, 10. ćwiczenia

2022-12-21

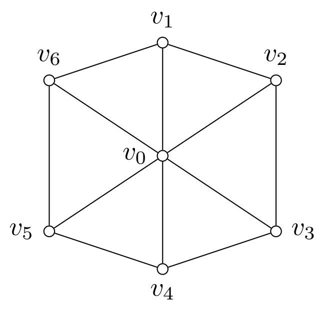
Zadanie 10.1

a) Niech G będzie grafem $n+1$ wierzchołkowym, $n > 4$, w którym jeden wierzchołek jest połączony ze wszystkimi innymi, a podgraf rozpięty na pozostałych n wierzchołkach jest cyklem elementarnym. Ile jest różnych drzew przeszukiwania w głąb w grafie G – dwa drzewa się różnią, jeśli istnieje wierzchołek, który w obu drzewach ma różnych rodziców. Opisz sposób obliczania tej liczby.

b) Niech $G = (V, E)$ będzie grafem dwuspójnym o co najmniej trzech wierzchołkach, a u jego wyróżnionym wierzchołkiem. Dobrą orientacją grafu G z wierzchołka u nazywamy graf skierowany otrzymany z G w następujący sposób: uruchomiamy algorytm przeszukiwania w głąb z wierzchołka u, a następnie orientujemy krawędzie drzewa przeszukiwania od ojca do syna, a krawędzie niedrzewowe od potomka do przodka. Dany jest graf zorientowany H z wyróżnionym wierzchołkiem u. Zaproponuj algorytm, który stwierdzi, czy H jest dobrą orientacją pewnego grafu G z wierzchołka u.

Rozwiązanie:
Punkt a)

Przykładowy graf dla $n = 5$:



Wynik to: $n(n-1)^2 + 2n$:

- $2n$ dla drzew rozpoczynających się w centrum grafu (korzeń jest ustalony na v_0, możemy na n sposobów wybrać pierwszy wierzchołek z cyklu v_i na dwa sposoby możemy wybrać kierunek w którym kontynuujemy DFS)
- $n \cdot (n-1)^2$ dla drzew rozpoczynających się na cyklu:

– $2n \cdot \sum_{j=2}^{n-2}(n-1-j)$ — $i \neq j$ i $(v_i, v_j) \notin E$: zaczynamy w v_i idziemy do v_j (idąc w lewo lub w prawo) idziemy do v_0 a następnie do v_k,

– $2n \cdot (n-2)$ — $i \neq j$ i $(v_i, v_j) \in E$ (i używamy tej krawędzi): zaczynamy w v_i idziemy do v_j (idąc po 1 krawędzi) idziemy do v_0 a następnie do v_k,

– n — $i \neq j$ i $(v_i, v_j) \in E$ (i NIE używamy tej krawędzi): zaczynamy w v_i idziemy do v_j (idąc po n-1 krawędziach) idziemy do v_0 i kończymy

– $n(n-1)$ — $i = j$: zaczynamy w v_i idziemy do v_0 a następnie do v_k następnie DFS odwiedza resztę cyklu.

Zadanie 10.2

Dane jest drzewo z korzeniem T, które jest DFS-drzewem rozpinającym pewnego n-wierzchołkowego grafu G. Wierzchołki drzewa są identyfikowane z ich numerami DFS wyznaczających kolejność ich pierwszych odwiedzin. Dla każdego wierzchołka i różnego od korzenia, $t[i]$ jest numerem rodzica i w drzewie T. Wartość $t[.]$ dla korzenia jest równa 0. Zaproponuj efektywny algorytm, który

a) sprawdzi, czy graf G może być grafem dwuspójnym wierzchołkowo, a jeśli odpowiedź jest pozytywna, to poda

b) minimalną liczbę krawędzi w grafie G,

c) maksymalną liczbę krawędzi w grafie G.

Rozwiązanie: Punkt a) Jeśli korzeń T ma więcej niż jednego syna to G nie jest dwuspójny.
Punkt b) każdy liść musi mieć jakąś krawędz powrotną, jeśli połączymy każdy liść z korzeniem to spełnimy wszystkie warunki dwuspójności, odpowiedź: $(n - 1 + L)$ gdzie L to liczba liści.
Punkt c) oprócz krawędzi z punktu b możemy dodać wszystkie, które nie naruszają DFS, czyli wszystkie do przodków w drzewie DFS.

Zadanie 10.3

Marszrutą w grafie G nazywamy każdy skończony ciąg wierzchołków grafu taki, że każde dwa kolejne wierzchołki są połączone krawędzią w tym grafie. Marszruta jest zamknięta, gdy rozpoczyna się i kończy w tym samym wierzchołku.

Powiemy, że graf G jest eulerowski, jeśli istnieje w nim marszruta zamknięta, w której każda krawędź z grafu pojawia się dokładnie raz. Marszrutę o takiej własności nazywamy cyklem Eulera.

Zaproponuj algorytm, który w czasie liniowym sprawdza, czy dany graf nieskierowany jest eulerowski i jeśli tak, to znajduje w nim cykl Eulera.

Rozwiązanie:
Cykl Eulera w grafie skierowanym G istnieje wtedy i tylko wtedy gdy:

- dla każdego wierzchołka $v \in V(G)$ mamy $indeg(v) = outdeg(v)$.

- nieskierowana wersja grafu G (tzn. taka w której ignorujemy zwrot krawędzi), jest spójna,

Algorytm:

- $C = \emptyset$
- tak długo jak G nie jest pusty, oblicz dowolny cykl i dołącz go do C,

Zadanie 10.4

Dane jest n-wierzchołkowe drzewo z korzeniem T z wagami na krawędziach (liczby całkowite). Dla każdego wierzchołka v różnego od korzenia dane są rodzic $p[v]$ w drzewie i waga $w[v]$ krawędzi $v - p[v]$. Przyjmujemy też, że wierzchołki są ponumerowane w porządku "preorder" i utożsamiamy je z tymi numerami - v oznacza zarówno wierzchołek, jak i jego numer.

Zaproponuj algorytm, który w czasie $O(n + k)$ udzieli odpowiedzi na k zapytań postaci:

$W(u; v)$:: ile wynosi waga ścieżki (suma wag krawędzi na ścieżce) z u do v, gdy wiadomo, że u jest przodkiem v?

Rozwiązanie:

- policz $s[u]$ — suma waga na krawędziach na ścieżce od u do korzenia,
- $W(u, v) = s[v] - s[u]$

Zadanie 10.5

Kaktusem nazywamy graf, w którym każda dwuspójna składowa jest krawędzią lub cyklem.

a) Zaprojektuj wydajny czasowo algorytm, który dla danego kaktusa G i wskazanych wierzchołków u i v obliczy liczbę różnych ścieżek elementarnych z u do v.

b) Załóżmy, że krawędziom kaktusa przypisano całkowitoliczbowe wagi. Zaprojektuj wydajny czasowo algorytm, który w ważonym kaktusie $G = (V, E)$ znajduje minimalną wagę DFS drzewa rozpinającego zakorzenionego w zadanym wierzchołku s.

Uwaga: DFS drzewo rozpinające, to drzewo ukorzenione i takie, że krawędzie niedrzewowe łączą tylko potomków z przodkami w tym drzewie.

Zadanie 10.6

Grafy trójkątne to grafy spójne, w których każda dwuspójna składowa jest trójkątem (cyklem długości 3).

a) Udowodnij, że każdy graf trójkątny jest 3-kolorowalny.

b) Zaproponuj efektywny algorytm 3-kolorowania grafów trójkątnych.

c) Zaproponuj efektywny algorytm obliczania rozmiaru najliczniejszego skojarzenia w danym grafie trójkątnym.

Rozwiązanie: Punkt a) Znajdź dowolny "zewnętrzny" trójkąt v_1, v_2, v_3. Pokoloruj rekurencyjnie graf po usunięciu krawędzi $(v_1, v_2), (v_2, v_3), (v_3, v_1)$ i izolowanych wierzchołków. Takie pokolorowanie można łatwo uzupełnić o kolorowanie v_1, v_2, v_3 gdyż co najwyżej jeden z wierzchołków otrzymał kolor, więc pozostałe możemy pokolorować pozostałymi dwoma wolnymi kolorami.

Punkt b) Policz drzewo DFS grafu G, zauważmy, że w grafie trójkątnym wszystkie krawędzie powrotne (u, v) łączą wierzchołki o różnicy wysokości dokładnie 2 ($h[u] - h[v] = 2$). Więc jeśli nadamy wierzchołkom kolory $c(v) = h[v]$ mod 3 otrzymamy poprawne kolorowanie

Punkt c) Jak w dowodzie a) tylko dodajemy do skojarzenia krawędź z trójkąta, która nie sąsiaduje z pozostałym grafem.