Corso di Algoritmi e Strutture Dati
Anno Accademico 2018–2019
Prof. Giovanni Pighizzini

Esercizi

6. Grafi

Esercizio 6.1
Sia G un grafo (orientato o non orientato) con n vertici. Supponete che esista almeno un cammino da un vertice x e un vertice y. Stabilite un limite superiore per la lunghezza del più corto cammino tra x e y.

Esercizio 6.2
Sia M la matrice di adiacenza di un grafo G orientato con n vertici $v_1, v_2, \ldots, v_n$, i.e., per $i, j = 1, \ldots, n$, $M_{i,j} \in \{0, 1\}$ e $M_{i,j} = 1$ se e solo se G contiene l'arco (v_i, v_j) ($M_{i,j}$ indica l'elemento di riga i e colonna j di M). Dimostrate che per ogni intero $k \geq 0$, $M^k_{i,j} = 1$ se e solo se G contiene un cammino di lunghezza k da v_i a v_j, dove M^k indica la k-esima potenza di M calcolata utilizzando le operazioni booleane di disgiunzione e congiunzione al posto di somma e prodotto.
Suggerimento: si può procedere per induzione su k.

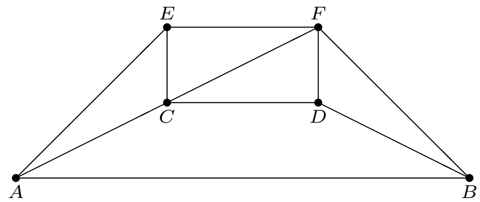
Esercizio 6.3
Come nell'esercizio 6.2 considerate la matrice di adiacenza M di un grafo orientato G e la k-esima potenza M^k di M, calcolata questa volta come matrice di interi (cioè con le usuali operazioni di somma e prodotto, al posto delle operazioni booleane). Cosa rappresenta il valore del generico elemento $M^k_{i,j}$?

Esercizio 6.4
Sia G un grafo orientato con n vertici $v_1, v_2, \ldots, v_n$, rappresentato mediante una matrice di adiacenza M. Trovate un metodo per determinare, a partire da M, una matrice N con n righe e n colonne e con elementi in $\{0, 1\}$, tale che $N_{i,j} = 1$ se e solo se G contiene un cammino (di una qualsiasi lunghezza) dal vertice v_i al vertice v_j.
Suggerimento: utilizzate i risultati degli esercizi 6.1 e 6.2.

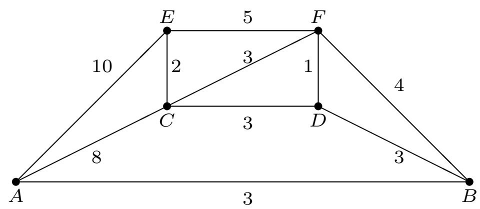
Esercizio 6.5
Sia G un grafo orientato con n vertici $v_1, v_2, \ldots, v_n$, rappresentato mediante una matrice di adiacenza M. Trovate un metodo per individuare, utilizzando la matrice N descritta nell'esercizio 6.4, tutti i vertici che appartengono alla stessa componente fortemente connessa di un vertice v_i.

Esercizio 6.6
Scrivete una versione iterativa dell'algoritmo per la visita in profondità di un grafo non orientato.
Suggerimento: è utile servirsi di una struttura a pila nella quale caricare i vertici già raggiunti e dei cui archi incidenti si debba iniziare o completare l'ispezione.

Esercizio 6.7
Individuare almeno 3 differenti alberi ricoprenti per il seguente grafo:



Esercizio 6.8
Considerate il grafo pesato rappresentato nella figura seguente:



Applicate l'algoritmo di Kruskal per individuare un albero ricoprente di costo minimo. Elencate, in successione, gli archi che vengono selezionati dall'algoritmo. Esistono altri alberi ricoprenti di costo minimo?

Esercizio 6.9
Svolgete l'esercizio 6.8, utilizzando l'algoritmo di Prim al posto di quello di Kruskal.

Esercizio 6.10
Determinate un albero ricoprente di *costo massimo* per il grafo dell'esercizio 6.8.

Esercizio 6.11
Dimostrate che se un grafo non orientato G è una foresta di k alberi se e solo se G contiene $n - k$ archi ed è privo di cicli.

Esercizio 6.12
Una *foresta ricoprente* di un grafo non orientato è una foresta contenente un albero ricoprente per ogni componente connessa del grafo. Supponete di applicare l'algoritmo di Kruskal a un grafo non orientato, pesato e non connesso. Cosa si ottiene? E se si applica l'algoritmo di Prim?

Esercizio 6.13
Progettate un algoritmo basato sulla tecnica union-find per determinare le componenti connesse di un grafo non orientato. Valutate poi il tempo di calcolo.
Suggerimento: Ispiratevi all'uso della tecnica union-find nell'implementazione dell'algoritmo di Kruskal.