Algorísmica Avançada

Parcial I de Pràctiques

Grup A - divendres de 10 a 12. Prof. Carles Franquesa

28 d'octubre de 2011

Grau en Enginyeria Informàtica

Universitat de Barcelona

Per fer aquest parcial no heu de crear cap fitxer nou. Tant les rutines auxiliars com les instàncies de prova que es mostren al final de l'enunciat estan penjades al campus. Descarregueu-les, i renomeneu l'arxiu font en python al format *P1_A_nom_cognoms.py*. El parcial es composa de tres exercicis que un cop resolts caldrà penjar de nou al campus, en el mateix únic fitxer, en el qual ha de figurar-hi també el vostre nom en la primera línia. En tots tres, l'entrada de dades és un graf, que es llegirà d'un fitxer de text amb la rutina `llegir_graf()` que com podeu observar, també la teniu disponible en les rutines auxiliars al final de l'enunciat.

1. *Graf Eulerià.* Un graf és *eulerià* si tots els seus nodes tenen grau parell. El grau d'un node és el nombre de veïns. Feu una funció `euleria(G)`, que retorni un booleà indicant si un graf és eulerià, o no. Analitzeu l'eficiència de la funció. (10 punts)

2. *Ponts.* Una aresta és un *pont* en un graf si a l'eliminar-la, el nombre de components connexes augmenta. Feu una funció `pont(G,u,v)`, que retorni un booleà indicant si l'aresta (u, v) és un pont en el graf G, o no. Analitzeu l'eficiència de la funció. (30 punts)

3. *Cicle Eulerià.* Els grafs eulerians contenen cicles eulerians. Un *cicle eulerià* és un cicle que passa per totes les arestes sense repetir-ne cap (encara que pot repetir nodes). Feu un programa que donat un graf, comprovi si és eulerià, i si ho és, que ens digui una seqüència de nodes que formin un cicle eulerià, és a dir, que passi per totes les arestes sense repetir-ne cap. Per això cal començar a partir de qualsevol node i creuar qualsevol aresta de la seva adjacència sempre que no sigui un pont, si és possible, i si no hi ha més remei, creuar el pont. Un cop creuada qualsevol aresta, ja sigui pont o no, cal borrar-la del graf, i repetir l'acció a partir del node actual. El procés s'acaba quan en el graf ja no hi ha cap aresta. Per resoldre aquest exercici podeu utilitzar les rutines auxiliars que es donen tot seguit. (60 punts)

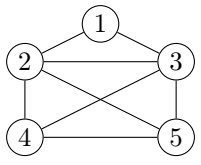
```
def llegir_graf():                                          #Θ(V + E)
    nom = input("Dona'm un nom pel graf:  ")                #Θ(1)
    G = nx.read_adjlist(nom)                                #Θ(V + E)
    return G                                                #Θ(1)

def dfs_visit(G,v):                                         #Θ(E/V)
    global vistos
    vistos.append(v)                                        #Θ(logV)
    for u in G.neighbors(v):                                #Θ(E/V)
        if u not in vistos:                                 #Θ(logV)
            dfs_visit(G,u)                                  #Θ(E/V)

def dfs(G):                                                 #Θ(V + E)
    global vistos
    q = 0                                                   #Θ(1)
    vistos = []
    for v in G:                                             #Θ(V)
        if v not in vistos:                                 #Θ(logV)
            q = q + 1                                       #Θ(1)
            dfs_visit(G,v)                                  #Θ(E/V)

    return q                                                #Θ(1)

def numero_de_components(G):                                #Θ(V + E)
    return dfs(G)                                           #Θ(1)
```

Rutines auxiliars.

entrades i sortides

Seguidament es mostren dos exemples. A la primera columna hi ha el graf, a la segona el contingut del fitxer de text que el representa, i en la tercera, la sortida que hauria de donar el vostre programa. Utilitzeu, per conveniència, el vértex 1 com a node inicial.

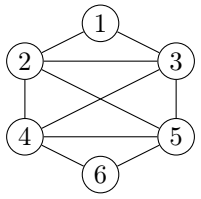
1.




```
1 2 3
2 3 4 5
3 4 5
4 5
```

no es un graf euleria

2.




```
1 2 3
2 3 4 5
3 4 5
4 5 6
5 6
```

['1', '3', '2', '5', '3', '4', '5', '6', '4', '2', '1']