UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Politècnica Superior d'Enginyeria de Vilanova i la Geltrú

RECUPERACIÓ DE LA INFORMACIÓ (REIN)

Sessió 4 laboratori: PageRank

En aquesta sessió:

- Usarem dos fitxers de text per construir una xarxa d'aeroports i vols.
- Programarem una versió del càlcul del *PageRank* per aquestes dades.

1 Fitxers d'entrada

Tenim dos fitxers descarregats des d'OpenFlights:[1]

`airports.txt` conté una llista d'aeroports del món. Els primers camps són: identificador d'OpenFlights, nom de l'aeroport, nom de la principal ciutat a la que serveix, país, 3 lletres del codi IATA, 4 lletres del codi ICAO i altres dades. Només els aeroports principals tenen codis IATA. Com a exemple, les dues primeres línies d'aquest fitxer són:

```
1,"Goroka","Goroka","Papua New Guinea","GKA","AYGA",-6.081689,145.391881,5282,10,"U"
2,"Madang","Madang","Papua New Guinea","MAG","AYMD",-5.207083,145.7887,20,10,"U"
```

`routes.txt` conté una llista de rutes aèries del món. Els primers camps són: codi de la línia aèria, codi de línia aèria d'OpenFlights, codi de l'aeroport d'origen (3 lletres del codi IATA o 4 lletres del codi ICAO), el mateix amb el codi d'OpenFlights, codi de l'aeroport destinació (3 lletres del codi IATA o 4 lletres del codi ICAO) i altres dades. En el cas de que tingueu dubtes sobre el contingut del fitxer o amb els codis, podeu consultar la pàgina d'OpenFlights.

```
2B,410,AER,2965,ASF,2966,,0,CR2
2B,410,AER,2965,GOJ,4274,,0,CR2
```

Per aquesta tasca treballarem només amb codis IATA i amb aeroports que tinguin codis IATA. Ignorarem els aeroports que no tinguin codis IATA i també els codis propis d'OpenFlights.

[1] Les persones que han fet la descàrrega han fet un donatiu de part vostra.

2 PageRank

Volem calcular el valor de *PageRank* dels aeroports usant una xarxa definida per les rutes entre els aeroports. A partir dels fitxers anteriors podem construir un graf on els vèrtexs són els aeroports IATA i les arestes tripletes de la forma (i, j, k) on i i j són codis IATA i k és el nombre de rutes entre i i j. Noteu que, a diferència de la versió vista a classe, les arestes tenen pesos, per tant, el grau de sortida de i, $out(i)$, s'ha de calcular com la suma de tots els pesos de totes les arestes de sortida de i:

$$out(i) = \sum_{(i,j,k) \in E} k$$

A classe hem explicat un mètode iteratiu per calcular el *PageRank* usant notació matricial. Un notació alternativa pel mateix mètode és:

```
1. n = nombre de vèrtexs de G;
2. P = qualsevol vector de mida n i suma 1 (per exemple, vector ple de 1/n);
3. L = el factor d'amortiment escollit, entre 0 i 1;
4. while (no condició acabament) {
5.       Q = vector de mida n ple de 0;
6.       for i in 0..n-1 {
7.           Q[i] = L * sum { P[j] * w(j,i) / out(j) :
8.                            hi hagi una aresta (j,i) a G }
9.                  + (1-L)/n;
10.      }
11.      P = Q;
12. }
```

La condició d'acabament pot ser un nombre fix d'iteracions o bé quan dues `P` d'iteracions consecutives variïn poc (convergència). És feina vostra pensar si això significa una diferència de 2 decimals, 4, 16, o depèn de n.

El factor d'amortiment queda al vostre criteri; sovint es troba entre $0,8$ i $0,9$. També podeu investigar si triar un valor o altre per aquest factor d'amortiment, afecta la vostra solució i/o el temps de còmput.

3 Tasca

1. Abans de començar, estigueu segurs de que enteneu la versió del càlcul del *PageRank* proposada: Què significa el terme `(1-L)/n` en la línia 9 del pseudocodi? Què representen les variables `P` i `Q` del codi?

2. Escriviu un programa, usant el llenguatge de programació que vulgueu, que:

- llegeixi el fitxer `airports.txt` i `routes.txt` en estructures de dades adients,
- calculi el valor de *PageRank* de cada aeroport, i
- mostri una llista de parells (*page rank*, nom d'aeroport) ordenada de forma decreixent pel valor de *page rank*.

Les estructures de dades que probablement necessitareu són:

- Taules (arrays, hash, ...) que relacionin codis d'aeroports, noms d'aeroports i índexs amb vèrtexs.
- Un graf de forma que permeti recuperar de forma eficient totes les tripletes (i, j, k) per a una j donada. No seria bona idea (ineficient en temps i memòria) construir una matriu $n \times n$, on n és el nombre d'aeroports.

Qüestions importants:

- Una comprovació senzilla del funcionament del vostre algorisme és que després de cada iteració la suma dels elements de `P` ha de ser 1. Si no ho és, alguna cosa heu fet malament en la vostra implementació.
- L'eficiència de la solució és important. El temps d'execució per una iteració hauria de ser lineal respecte el nombre d'arestes i vèrtexs. Ocupar memòria o temps de forma quadràtica respecte el nombre de vèrtexs, no seria una bona solució. En aquest graf (i en la majoria de grafs on s'aplica el *PageRank*), el nombre d'arestes és de $O(n)$, no de $O(n^2)$. L'execució de l'algorisme en aquest graf hauria de trigar segons, no minuts.
- No es garanteix que tots els aeroports del primer fitxer tinguin rutes d'entrada i de sortida. Recordeu que aquests casos distorsionen l'algorisme del *PageRank*. Preneu les precaucions que siguin necessàries quan implementeu l'algorisme, però no els elimineu: tenen un pes de PageRank encara que tinguin 0 arestes de sortida, i ha de ser calculat. Tampoc seria correcte afegir les arestes des d'aquests nodes cap a tots els altres nodes perquè això faria créixer el nombre d'arestes a n^2. Modifiqueu el pseudocodi anterior per afegir l'efecte d'aquestes arestes (virtuals) de forma eficient: quantes n'hi ha i quan de pagerank afegeixen en total a cadascun dels vèrtexs en particular?
- També pot succeir que alguns codis d'aeroport que apareguin en les rutes no apareguin en el fitxer d'aeroports. Aquests sí que podeu eliminar-los.

4 Lliuraments

Per aquesta activitat no haureu lliurar un informe però sí mostrar a classe el resultat que ha produït el vostre programa. Se us faran preguntes concretes sobre decisions preses a l'hora de fer la implementació, per poder avaluar l'activitat.