CAIM Lab, Session 5: Pagerank

Moreno Oya, Daniel
Cidraque Sala, Carles

INTRODUCCIÓN

En esta práctica nos piden que hagamos un script donde hemos de calcular el pagerank de una lista de aeropuertos construyendo una red definidas por rutas entre los aeropuertos. Originalmente el pagerank se utiliza para páginas web pero en este caso lo utilizamos para aeropuertos (las aristas tendrán peso, el cual es el número de rutas entre los aeropuertos).

IMPLEMENTACIÓN

Primero hemos implementado las estructuras de datos necesarias para guardar los aeropuertos y las rutas de la manera más eficiente posible. Las estructuras de datos que hemos creado son las siguientes:

- **Airport :** clase con los siguientes atributos:
    - **code:** código IATA del aeropuerto
    - **name:** ciudad y país del aeropuerto
    - **routes**: set de los aeropuertos que lo tienen como destino en una ruta
    - **routeHash**: un map donde cada llave es aeropuerto destino del de la clase y cada valor es el peso de la arista
    - **outdegree**: número de rutas que tiene donde el aeropuerto es origen
    - **pageRank**: valor del pagerank del aeropuerto

- **Edge**: clase con los siguientes atributos:
    - **origin**: código del aeropuerto origen
    - **weight**: número de veces que hay esa ruta

- **edgeList:** lista de edges. Representa todas las rutas
- **edgeHash**: edgeHash[(origen, destinación)] = weight
- **airportList**: lista de todos los aeropuertos
- **airportHash**: airportHash[codigo] = airport

Con esta implementación de la red de rutas entre aeropuertos nos permite hacer el cálculo del pagerank en tiempo lineal respecto al número de aeropuertos.

El código dado nos mostraba un pseudocódigo para realizar el cálculo del pagerank. Debíamos decidir el valor del damping factor (el cual iremos modificando para realizar experimentos) y también cuando paramos la ejecución del cálculo. Hemos puesto dos condiciones para decidir cuando parar el cálculo:

- Número máximo de iteraciones
- Cuando dos cálculos de iteraciones consecutivas convergen

El vector P del pseudocódigo representa el vector con los pagerank calculados. En nuestra implementación es el atributo pageRank de la clase Airport. El vector Q se utiliza para ir almacenando el cálculo de los pagerank en cada iteración.

Uno de los principales problemas que nos encontramos era encontrar la manera de asegurarnos que la suma de los pagerank diera aproximadamente 1. Pero aplicando la

fórmula dada para calcular el pagerank nos dimos cuenta que se cumplía esta condición. El principal problema era los aeropuertos con outweight igual a 0, es decir, los aeropuertos aislados que no eran ni origen ni destino en ninguna ruta. En cada iteración le asignamos un valor a cada aeropuerto para asegurar que la suma sea aproximadamente 1.

EXPERIMENTACIÓN

En este apartado haremos experimentos modificando los diferentes parámetros para ver si los resultados varían y cómo afecta al tiempo de ejecución y al número de iteraciones que realiza el algoritmo. Los parámetros que modificaremos serán:

- El número máximo de iteraciones, que sirve para parar el cálculo del pagerank en caso de que el número de iteraciones llegue al máximo.
- Epsilon, parámetro que marca la máxima diferencia entre dos cálculos de iteraciones consecutivas para que converjan.
- Damping factor

Número máximo de iteraciones

Para la realización de este experimento hemos fijado la epsilon a 1×10^{-5} y el damping factor a 0.85.

		Valor de pagerank	
Máx. Iteraciones	**Segundos**	**Aeropuerto más popular**	**Aeropuerto menos popular**
1	0.02657	0.00518017	0.00002613
5	0.12388	0.00356422	0.00002613
10	0.23872	0.00355103	0.00002613
22	0.51667	0.00354837	0.00002613
25	0.51667	0.00354837	0.00002613

El comportamiento del tiempo de ejecución parece trivial respecto al número de iteraciones que hace el programa para el cálculo del pagerank. Si aumentamos el número máximo de iteraciones el algoritmo tendrá mayor tiempo de ejecución pero también tendrá más precisión en los cálculos. También podemos observar si el número máximo de iteraciones es mayor que 22 los resultados no se modifican porque el algoritmo converge en la iteración 22 y el algoritmo para.

Epsilon

Para la realización de este experimento hemos fijado el damping factor a 0.85 y el número máximo de iteraciones a 1000.

		Valor de pagerank		
Epsilon	**Segundos**	**Aeropuerto más popular**	**Aeropuerto menos popular**	**Iteracions**
1×10^{-1}	0.00222	0.00017422	0.00017422	0
1×10^{-5}	0.52078	0.00354837	0.00002613	22
1×10^{-10}	2.18719	0.00354807	0.00002613	93
1×10^{-15}	3.83906	0.00354807	0.00002613	164

La epsilon es uno de los parámetros responsables de la finalización del bucle que realiza el cálculo de los pagerank. Si la epsilon es grande entonces los resultados serán menos precisos pero harà menos iteraciones y el tiempo de ejecución será menor. En cambio, si vamos disminuyendo la epsilon los resultados serán cada vez con más precisión y el algoritmo hará más iteraciones y , por lo tanto, tendrá mayor tiempo de ejecución. Como podemos ver para epsilon = 1×10^{-1} el algoritmo no hace ninguna iteración y cada aeropuerto tiene un pagerank = 1/n.

Damping factor

Para la realización de este experimento hemos fijado epsilon a 1×10^{-10} y el número máximo de iteraciones a 2000.

		Valor de pagerank		
Damping factor	**Segundos**	**Aeropuerto más popular**	**Aeropuerto menos popular**	**Iteracions**
0.1	0.169068	0.00073221	0.00015679	7
0.25	0.26238942	0.00146534	0.00013066	11
0.5	0.51281023	0.00246245	0.00008711	22
0.75	1.23466014	0.00324983	0.00004355	53
0.85	2.1632726	0.00354807	0.00002613	93

0.9	3.3089466	0.00381352	0.00001742	143
0.95	7.0860567	0.00408869	0.00000871	294
0.99	34.825273	0.00426428	0.00000174	1499

Como podemos ver el algoritmo tendrá mayor tiempo de ejecución si vamos aumentando el damping factor. Por lo tanto, el algoritmo tarda más en que los resultados de dos iteraciones consecutivas converjan, ya que el número máximo de iteraciones es bastante grande. También vemos que si vamos aumentando el damping factor los resultados son cada vez más precisos.