Alexandru-Ilie Popa
Pol Casacuberta Gil

CAIM Lab, Session 5: Pagerank

Eleccions de disseny

Estructura de dades que utilitzem:

- ➔ Airport: classe amb els atributs de cada aeroport (codi, nom, rutes, diccionari ruta-
- ➔ pes, out-weight i PageIndex)
- ➔ Edge: classe amb els atributs de cada ruta (origen, destinació, nombre de rutes)
- ➔ edgeHash: hash de les rutes i pes de cada ruta
- ➔ airportList: llista d'aeroports
- ➔ airportHash: hash key de cada IATA code a cada Airport
- ➔ PageRank: llista dels page ranks

Implementació del computePageRanks

Primer inicialitzem el vector P de tal forma que la suma dels seus elements sigui 1.
A cada iteració actualitzem el pagerank de cada node, el nombre d'iteracions dependrà d'una condició de parada (tolerància).

Hem escollit que la condició de parada sigui quan dues P's en iteracions consecutives estiguin prou a prop. Aquesta decisió és deguda al fet que fixar un nombre d'iteracions concret pot passar que o bé no siguin prou iteracions i perdem en precisió o bé són massa i perdem en eficiència. També sabem que cada vegada que és fa menor la diferència entre iteracions, sabem que estem convergint cap a una solució. Això és degut al fet que estem calculant una aproximació del pagerank teòric.

Per a assegurar-nos de què l'algoritme que hem programat sigui correcte hem fet diverses comprovacions. Una comprovació fàcil i ràpida de veure que el programa va ben encaminat és que a cada iteració la suma dels elements a P és 1 o proper a 1 per culpa d'errors d'arrodoniment.

També hem comprovat de forma visual que els page ranks coincideixin amb les rutes és a dir que pageranks molt elevats no continguin menys rutes entrants que pageranks molt petits, això evidentment no de forma exhaustiva però si un parell d'exemples per a comprovar la seva versemblança.

Problema trobat en fer la pràctica

En acabar el codi i en executar-lo per primera vegada, la suma en cada iteració dels elements del vector P no s'acostava a 1. Hem tornat a llegir l'enunciat i ens vam adonar que no teníem en compte els nodes que no tenen arestes incidents o sortints.
Ho hem solucionat calculant l'aportació dels nodes desconnexos sense haver de modificar el graf:

-Tots els nodes que no tenen arestes de sortida, si els connectem a tots els altres nodes, tindriem n-1 arestes, el dumping no varia, per tant, tindríem: v * $\frac{L}{n-1}$, on v és el nombre de nodes del graf

-També hem de tenir en compte la part ∑Pk[j]. En la primera iteració val el mateix que el valor a què està inicialitzat el vector P, és a dir $\frac{1}{n}$. Per a les altres val l'aportació total dels nodes desconnexos més el factor de teleportació:$\frac{1-L}{n}$ + ∑Pk[j] *v * $\frac{L}{n-1}$.

El cost total de preprocessar els fitxers i omplir les estructures de dades és O(n + m).
El cost per inicialitzar el vector P és com a molt O(n).
El cost total del bucle while és O(n + m). Ja que la suma de PageRanks per als nodes sense connexió és O(n). Per a cada aeroport calculem el seu nou rang accedint als seus veïns entrants O(m) (m són el número d'aristes) utilitzant valors ja calculats en temps constant. Com que la condició de parada no convergirà a un nombre més gran d'iteracions de n^2 llavors el cost total és O(n + m).

Experiment: damping factor

Hem provat diverses maneres d'inicialitzar el vector P, per a totes d'elles podem assegurar que la suma de tots els elements del vector és 1.

- $\frac{1}{n}$: Tot el vector està inicialitzat a $\frac{1}{n}$
- Un 1: Un element del vector està a 1 els altres a 0.
- $\sqrt{n}$ de $\frac{1}{\sqrt{n}}$: $\sqrt{n}$ elements del vector estan inicialitzats a $\frac{1}{\sqrt{n}}$.

	L = 0.2		L = 0.4		L = 0.6		L = 0.8		L = 0.85		L = 0.9	
	(s)	#it	(s)	#it	(s)	#it	(s)	#it	(s)	#it	(s)	#it
Un 1	0.72	17	1.47	27	2.21	44	4.44	93	5.02	125	5.15	188
$\frac{1}{n}$	0.59	12	1.28	22	2.15	38	3.16	87	4.29	119	5.71	184
$\sqrt{n}$ de $\frac{1}{\sqrt{n}}$	0.76	16	1.50	25	2.28	42	3.91	87	4.65	117	4.83	175

Hem vist que a mesura que augmenta el dumping factor l'algoritme tarda més a convergir i realitza un major nombre d'iteracions. Però també hem pogut observar que augmenta la precisió.
A mesura que disminuïm el dumping factor, obtenim els resultats més ràpidament, però perdem precisió i obtenim resultats més inexactes.

Experiment: closeness and speed of convergence

Per veure millor el nombre de decimals a tenir en compte hem triat com a dúmping factor 0.8 i hem provat de variar el nombre de decimals per a cada forma d'inicialitzar el vector P.

	L = 0.8																	
	ε=10^-2			ε=10^-4			ε=10^-6			ε=10^-8			ε=10^-10			ε=10^-12		
	s	PR	#it	s	PR	#it	s	PR	#it	s	PR	#it	s	PR	#it	s	PR	#it
Un 1	0.32	5.306092275256503e-05	7	1.06	5.299417064815061 4e-05	22	1.84	5.299417 06412294 6e-05	37	2.47	5.299417064122946 e-05	53	2.43	5.299417 06412294 6e-05	73	2.99	5.29941706412294 6e-05	93
$\frac{1}{n}$	0.04	9.450671661020844e-05	1	0.32	5.306092275256503 e-05	7	1.16	5.299417 06415069 9e-05	25	1.89	5.299417064122946 e-05	46	2.20	5.299417 06412294 6e-05	66	2.64	5.29941706412294 6e-05	87
$\sqrt{n}$ de $\frac{1}{\sqrt{n}}$	0.13	5.785758473056505e-05	3	0.78	5.299418321636606e-05	15	1.59	5.299417 06412296 1e-05	32	2.13	5.299417064122946 e-05	48	2.13	5.299417 06412294 6e-05	66	2.72	5.29941706412294 6e-05	87

Les conclusions que n'hem extret és que a mesura que augmenta el nombre d'iteracions els resultats són més precisos fins a arribar al punt de convergència i en aquest punt encara que es realitzin més iteracions no té efecte en la precisió dels resultats. Per aquesta raó ens hem quedat amb ε=10^-12