DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Ranking web

5 de diciembre de 2014

Métodos Numéricos
Trabajo Práctico Nro. 2

Integrante	LU	Correo electrónico
Martin Carreiro	45/10	martin301290@gmail.com
Kevin Kujawski	459/10	kevinkuja@gmail.com
Juan Manuel Ortíz de Zárate	403/10	jmanuoz@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

1. Resumen

Los sitios web a medida que fueron creciendo en cantidad en la época de los 90's, se complicó el acceso a ellos y ,a menos que alguien te comentara o a través de publicidades, era muy dificil acceder a la información deseada. Es por eso que se produjo el auge de los buscadores, que a partir de palabras claves, podrían devolverte sitios que puedan llegar a responder tu pregunta o decirte algo al respecto. Un primer problema de entrada, es que, como todo en la vida, la calidad de dicho contenido puede no ser el deseado y existan mejores. Durante este trabajo repasaremos 3 algoritmos conocidos de ranqueo de páginas web, veremos los resultados y los compararemos.
Una vez que sepamos cómo funcionan y cómo ordenan y ubican los resultados, intentaremos responder a la pregunta: cuáles son los pasos a seguir para poder mejorar tu sitio y que salga con mejor puntaje que la competencia.
Durante el desarrollo del trabajo práctico comprobaremos que la forma de posicionarse en las búsquedas depende, en todos los algoritmos, de los links que otras páginas tengan a tu sitio (y viceversa), por lo tanto lo que concluiremos es que si el cliente quiere posicionarse bien, en el menor tiempo posible, habrá que recomendarle que negocie con otras páginas web para ser apuntadas por estas. La diferencia es que no necesariamente serán las mismas, esto dependerá de que método de búsqueda se esté utilizando.

2. Introducción teórica

2.1. Algoritmos de búsqueda

Los buscadores web son aquellos sitios que dada una palabra o frase encuentran las páginas que tienen lo que el usuario estaba buscando en una gran cantidad de casos. El secreto del éxito de estos buscadores no es sólo que encuentran el sitio con la palabra o frase deseada sino que encuentran el mejor resultado para esta.
Esto lo hace mediante métodos de rankeo que no son nada sencillos y es por eso que estaremos analizando 3 algoritmos distintos para hacerlo: PageRank, HITS e Indeg. Describiremos la idea teórica de cada uno, sus implementaciones y analizaremos cualitativa y temporalmente sus resultados. Es decir cuanto tardan en ejecutarse y que tan buenos son los resultados devueltos para distintas búsquedas. Procedamos entonces a explicar la idea de cada uno.

2.1.1. PageRank

El algoritmo de rankeo de sitios llamado **PageRank** lo desarrollaron los fundadores de Google, Page y Brin, y fue tan revolucionario que cambió la forma en que se empezó a buscar en la web.
El primer punto de este algoritmo es mirar a la web como un grafo dirigido, donde cada nodo representa un sitio y los vértices van de un nodo a otro si un sitio contiene un link direccionando a este otro. A partir de esto se define una *matriz de vínculos* M, y como su nombre lo indica, esta representará las conectividades de los sitios pero con la particularidad de que tendrá información acerca de cuanto puntaje le asigna un sitio a otro, es decir, m_{ij} será $1/G_j$ (siendo G_j el grado de salida del sitio j) si j tiene un link a i y 0 en otro caso. De esta forma, la importancia de cada sitio esta medida por la calidad de sitios que linkean a cada uno, y a su vez la calidad esta conformada por la cantidad de sitios a los que redirijo, ya que si un sitio redirige a pocos sitios a cada uno estará otorgándole un puntaje cercano al 1, y en caso contrario el puntaje que otorgará sera cercano al 0 y no tendrá mucho peso a la hora de que los sitios de destino obtengan mayor importancia gracias a este.
Fácilmente se puede deducir que todas las columnas de la matriz resultante van a sumar 1, por lo que será una matriz *"estocastica por columnas"*. Luego el problema de encontrar el ranking de cada cada sitio es equivalente a encontrar un $x \in \mathbb{R}^n$ tal que $Px = x$, es decir, encontrar (suponiendo que existe) un autovector asociado al autovalor 1 de una matriz cuadrada, tal que $x_i \geq 0$ y $\sum_{i=1}^{n} x_i = 1$.
Uno puede abstraerse a la noción del *navegante aleatorio*. Este vendría a representar cualquier usuario común que navega por la web a través de los links y que elige cualquiera de ellos con igual probabilidad.
Un detalle importante que se debe considerar es que para que el PageRank funcione no puede haber nodos desconectados, ya que de ser así la matriz de vínculos no sería estocastica por columnas debido a que la columna de la matriz que representa a los sitios que linkea estarían todos en 0. La solución a esto vendría a ser la inversa del nodo, y es considerar a que el sitio esta conectado a **todos** los demás sitios, por lo que será equiprobable la navegación a cada uno de ellos. Para tal modificar se le debe hacer una modificación a la matriz original, con lo que si definimos $v \in \mathbb{R}^{n \times n}$, con $v_i = 1/n$ y $d \in \{0,1\}^n$ donde $d_i = 1$ si $G_i = 0$, y $d_i = 0$ en caso contrario, siendo n la cantidad total de sitios web. La nueva matriz de transición será:

$$\begin{aligned} D &= vd^t \\ M_p &= M + D. \end{aligned}$$

Además debemos considerar la noción de *teletransportación*. Este vendría a representar cuando un usuario decida visitar una página cualquiera del conjunto, independientemente de si esta se encuentra o no referenciada por esa otra página. Para ello, consideramos que esta decisión se toma con una probabilidad

$c \geq 0$, y podemos incluirlo al modelo de la siguiente forma:

$$\begin{aligned} E &= v\bar{1}^t \\ M_f &= cM_p + (1-c)E, \end{aligned}$$

donde $\bar{1} \in \mathbb{R}^n$ es un vector tal que todas sus componentes valen 1. Probabilísticamente, la componente x_j del vector solución (normalizado) del sistema $M_f x = x$ representa la proporción del tiempo que, en el largo plazo, el navegante aleatorio pasa en la página j.
Para llegar a la solución nos basaremos en el Método de la Potencia, que consiste en multiplicar la matriz sucesivamente a un vector inicial arbitrario hasta que la diferencia entre multiplicaciones sea ínfima. Esto se verá en la sección Desarrollo donde veremos la forma de implementación de la misma.

Una vez calculado el ranking, el resultado será el ranking de todas las páginas del conjunto.

2.1.2. HITS

Este algoritmo fué pensado por Kleinberg [2] y su esencia está en la separación conceptual que hace entre un nodo *autoridad* y uno *Hub*. Vale aclarar que en este caso también entenderemos a un nodo como parte del grafo que modela a la internet y que representa a un sitio web en particular.
Autoridad serían aquellos sitios que tienen mayor importancia dentro de un tema específico y Hubs los que apuntan a estos sitios. En palabras mas terrenales las autoridades serían los que tienen las mejores respuestas y los hubs los que conocen a las mejores autoridades. En la práctica se entiende por autoridades a aquellos nodos que son apuntados por una gran cantidad de sitios y por Hubs a los nodos a apuntan a muchos otros. Una web es mejor Autoridad si es apuntada por buenos Hubs y viceversa, un Hub es mejor si apunta a las mejores autoridades.
Para modelarlo computacionalmente pensamos a la red como una matriz $A \in \{0,1\}^{n \times n}$ donde cada fila y columna representan a un nodo y tienen un 1 si el nodo fila apunta al nodo columna o un 0 en caso contrario. Luego Kleinberg [2] nos señala que debemos crear un vector x e y para agrupar las autoridades y hubs respectivamente. Estos vectores nos dice que los obtengamos tomando un x0 e y0 inciales con todos sus valores 1 y realizando este cálculo:

$$x = A^t y \quad (1)$$
$$y = Ax, \quad (2)$$

n veces (para un n definido) o hasta obtener un error menor a la tolerencia deseada. El error lo obtenemos calculando la norma manhattan entre el vector obtenido en el paso actual y el previo a este (mas adelante explicaremos como es la norma esta).
Luego de esto nos quedan en cada vector ordenados en el vector x las mejores autoridades y en el y los mejores hubs.

Dicho procedimiento debe aplicarse sobre una subred (denominada root-set) que debe calcularse previamente. Para esto mediante algún buscador simple, basado en texto por ejemplo, se acota la red a una determinada cantidad de nodos. Luego se le agregan aquellos que apuntan a algún nodo de dicha sub-red y los apuntados por esta, para finalmente sí, a este root-set aplicarle HITS. En este trabajo supondremos que este acotamiento previo ya fué efectuado.

2.1.3. Indeg

A modo de comparación para los experimentos, agregamos este algoritmo que resulta un poco inocente. Supongamos que tenemos una red de páginas, el peso, o importancia, de cada una será el promedio de la cantidad de links existentes en otras páginas del mismo conjunto hacia esta sobre la cantidad de links en total. El cálculo del mismo es lineal en la cantidad de referencias y su complejidad espacial es un vector de tamaño igual que el conjunto inicial de páginas. Este vector final es el resultado.

$$[k_1/n, k_2/n, ..., k_n/n] \quad 0 \leq k_i \leq n \quad n = cantTotalLinks$$

2.2. Método De La Potencia

Durante el desarrollo de este trabajo utilizaremos el método de la potencia, conocido para aproximar los autovalores y autovectores de una matriz. Como el informe no esfuerza su concentración en cómo funciona este, recomendamos al lector informarse a cerca del mismo en [4].

2.3. Norma Manhattan

A lo largo de esta presentación, utilizaremos la distancia L1 entre dos vectores, también llamada Norma Manhattan. La norma Manhattan es la suma de la diferencia coordenada a coordenada en modulo:

$$\|v - w\|_1 = \sum_{i=1}^{n} |v_i - w_i|$$

El uso del mismo estará detallado en el desarrollo de cada algoritmo presentado.

2.4. Matriz Estocástica

Una matriz estocástica es aquella cuya suma de columnas o filas da 1, puede ser estocástica *por columnas* o *por filas* respectivamente. Formalmente es estócastica si:

$$\sum_j P_{i,j} = 1 \quad (por\ filas) \tag{3}$$

ó

$$\sum_i P_{i,j} = 1 \quad (por\ columnas) \tag{4}$$

2.5. Matriz Dispersa

Se define como matriz dispersa a aquella a la que la mayoría de sus elementos son cero. Ejemplo:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & a_{04} \\ 0 & a_{11} & a_{12} & 0 & 0 \\ 0 & 0 & 0 & a_{23} & 0 \\ 0 & 0 & 0 & a_{33} & 0 \\ a_{40} & 0 & 0 & 0 & 0 \end{bmatrix}$$

2.6. Almacenamiento de matrices dispersas

La matriz dispersa al tener la propiedad de tener muy pocos valores distintos de cero es conveniente solo guardar estos y asumir el resto como cero. Existen varias estructuras como Dictionary of Keys (DOK), Compressed Sparse Row (CSR) o Compressed Sparse Column (CSC) pensadas para optimizar el espacio y las operaciones con estas estructuras de datos.
Veamos las características de cada una:

2.6.1. CSC

Compressed Column Storage es una forma de almacenamiento para las matrices dispersas que se basa en el concepto de almacenar solo los elementos no-nulos de la matriz e información adicional para poder recorrerla y operar con ella normalmente.
En este caso su forma de almacenamiento consta de 3 arreglos y, como su nombre lo indica, destina un arreglo principal para almacenar los valores no-nulos de cada columna contiguos, y utiliza otro dos arreglos, uno del mismo tamaño del anterior que almacena el número de fila de cada elemento y otro de menor tamaño que contendrá las posiciones del arreglo principal en donde se producen los saltos de columna, que indica cuando un valor pertenece a la columna siguiente.

Veamos un ejemplo, suponiendo que tenemos la siguiente matriz:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 4 \\ 9 & 1 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 7 & 0 & 6 & 2 \\ 5 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Los arreglos quedarán asi:

$$valores = \begin{pmatrix} 9 & 5 & 1 & 7 & 3 & 2 & 6 & 4 & 2 \end{pmatrix}$$

$$indice\ filas = \begin{pmatrix} 2 & 5 & 2 & 4 & 2 & 3 & 4 & 1 & 4 \end{pmatrix}$$

$$puntero\ columnas = \begin{pmatrix} 1 & 3 & 5 & 6 & 8 & 10 \end{pmatrix}$$

2.6.2. CRS

Compressed Row Storage es otra forma de almacenamiento, análoga a CSC, con la diferencia de que en el segundo arreglo al que nos referimos anteriormente almacena el numero de columna de cada elemento y

el tercer arreglo contiene las posiciones del arreglo principal en donde se producen saltos de fila.

Siguiendo el ejemplo anterior los arreglos quedarían:

$$valores = \begin{pmatrix} 4 & 9 & 1 & 3 & 2 & 7 & 6 & 2 & 5 \end{pmatrix}$$

$$indice\ columnas = \begin{pmatrix} 5 & 1 & 2 & 3 & 4 & 2 & 4 & 5 & 1 \end{pmatrix}$$

$$puntero\ filas = \begin{pmatrix} 1 & 2 & 5 & 6 & 9 & 10 \end{pmatrix}$$

2.6.3. DOK

Dictionary of Keys se basa en otro concepto de almacenamiento de los valores no-nulos distinto de los anteriores. Como su nombre lo indica, es un diccionario en donde solo se almacenan los valores no-nulos y su posición como clave del mismo, cosa que nos asegura que no haya dos valores para una misma posición. Es decir, por cada elemento no nulo de la matriz el diccionario tendrá un elemento $< (fila, columna), valor >$

Continuando el ejemplo que usamos en las secciones anteriores, los elementos del diccionario serán:

$< (1,5), 4 >$
$< (2,1), 9 >$
$< (2,2), 1 >$
$< (2,3), 3 >$
$< (3,4), 2 >$
$< (4,2), 7 >$
$< (4,4), 6 >$
$< (4,5), 2 >$
$< (5,1), 5 >$

2.6.4. Elección

Como las matrices que vamos a manejar son en su mayoría dispersas debíamos decidir una forma optima de almacenarlas . Entre las tres formas presentadas anteriormente terminamos eligiendo el Dictionary Of Keys. Tomamos esta decisión en función de lo que necesitábamos hacer y los tiempos que teníamos para hacerlo, y por tal motivo vimos que el DOK nos daba una mejor visualización de los datos. Sin bien recorrer el DOK es mas complicado que en CSC o CRS porque los elementos no tienen un orden especifico, este tiene un fácil acceso a cada posición de la matriz sin necesidad de realizar cálculos extras, ventaja que termino pesando para la toma de decisión. Otra característica que peso para la elección fue que las complejidades para las operaciones necesarias con la matriz eran muchos menores, y aún teniendo en cuenta que el almacenamiento requerido es mayor que para el de los otros dos, la diferencia no es tan grande y optamos por priorizar lo otro, ya que además cumple todas las expectativas y necesidades que requerimos para elaborar el informe y realizar los experimentos.

3. Desarrollo

A continuación detallamos cómo fue el desarrollo de los algoritmos presentados previamente

3.1. Page Rank

El algoritmo de PageRank lo dividimos en dos etapas, primero la inicialización en donde se crea la matriz estocástica y luego la corrida en donde se itera y calcula el pagerank hasta que la diferencia de norma entre los vectores sea menor que la tolerancia establecida.

Inicialización:

Algorithm 1 inicializar(c, dim, links)

```
1: v_inicial = vector(dim, 0);                ▷ creo un vector de dim elementos y lo inicializo en 0
2: v_inicial[0] = 1;                          ▷ pongo el primer elemento en 1
3: M_f = DOK(dim);                            ▷ la matriz en forma de DictionaryOfKeys representará al M_f
4: desconectados = vector()                   ▷ creo un vector para almacenar los nodos que no tienen salidas
5: for cada link en links do
6:     if link tiene salidas then
7:         n = link.cantidadDeSalidas;
8:         for cada salida del nodo do
9:             M_f.definir(salida, link, 1/n );   ▷ A cada salida del link actual le doy puntaje de 1/n
10:    else
11:        desconectados.agregar(link);
```

Calculo del PageRank:

Algorithm 2 calcular()

```
1: v = v_inicial
2: while norma > tolerancia do                ▷ hasta que converja
3:     w = v          ▷ guardo el vector anterior para calcular la norma Manhattan más adelante
4:     sumaDesconectados = 0
5:     for cada link desconectado: linkDesconectado do
6:         sumaDesconectados += v.elemento(linkDesconectado)   ▷
7:     v = M_f v                              ▷ aplico una iteraci'n del método de la potencia
8:     for cada elemento de v: v_i do
9:         v_i = c*(v_i+sumaDesconectados)+((1-c)/links.tamaño) ;   ▷
10:    norma = manhattan(v,w)                 ▷ calculo la norma Manhattan
```

3.2. HITS

Este también lo dividimos en la etapa en la etapa de inicialización y de cálculo de sus vectores. En la primera creamos la matriz estocástica e inicializamos los vectores y en la segunda calculamos los mismos hasta que iteremos k veces o la diferencia obtenida sea menor que la tolerancia.

Algorithm 3 inicializar(grafo)

```
1: dok = nuevoDOK()
2: for cada nodo del grafo do
3:     nodosLlegada = nodo.obtenerNodosALosQueLlega();
4:     for nodo2 de nodosLlegada do
5:         dok.definir(nodo1,nodo2,1)
6: vectorHubs = nuevoVectorConUnos(grafo.cantNodos())
7: vectorAutoridades = nuevoVectorConUnos(grafo.cantNodos())
```

Algorithm 4 calculoAutoridadesYHubs(tolerancia)

```
1: dokTranspuesto = dok.transponer();
2: for 1 a K do
3:     vectorHubsPrevio = vectorHubs;
4:     vectorAutoridadesPrevio = vectorAutoridades;
5:     vectorHubs = dokTranspuesto X vectorHubs;
6:     vectorHubs.normalizar();
7:     vectorAutoridades = dok X vectorAutoridades;
8:     vectorAutoridades.normalizar();
9:     toleranciaAutoridades = NormaManhattan(vectorAutoridades,vectorAutoridadesPrevio);
10:    toleranciaHubs = NormaManhattan(vectorHubs,vectorHubsPrevio);
11:    if toleranciaHubs < tolerancia||toleranciaAutoridades < tolerancia then
12:        break;
```

3.3. Indeg

La implementación de este algoritmo es bastante simple. Tomamos un vector inicial con ceros del tamaño de las páginas y recorremos todas las referencias de cada página hacia al resto, y por cada una de los sitios a los que visita, le sumamos en 1/cantidadLinksTotal su puntaje en el vector inicial.

Algorithm 5 calcular(links)

```
1: v_inicial = vector(links.size, 0);          ▷ creo un vector de dim elementos y lo inicializo en 0
2: totalAmountOfDomains = links.size;
3: for cada link en links do
4:     if link tiene salidas then
5:         for cada salida del nodo do
6:             v[salida] = v[salida] + 1/totalAmountOfDomains;   ▷ A cada salida del link actual le sumo el puntaje 1/totalAmountOfDomains
```

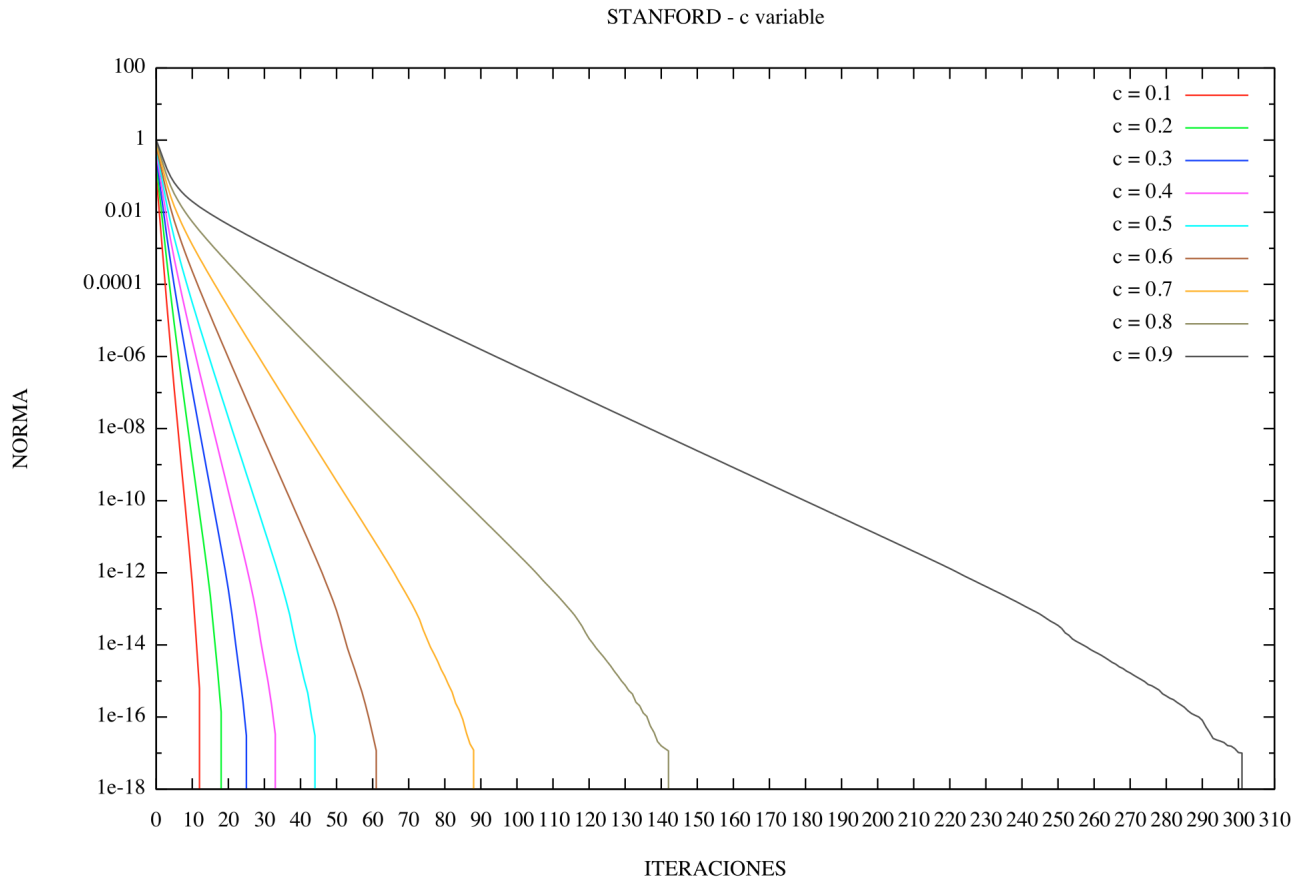
4. Experimentación Y Resultados

Los algoritmos previamente detallados tienen muchos puntos a testear. Veamos cada uno de estos en forma detallada y con los resultados intentemos responder a la pregunta original: ¿qué estrategia debo tomar para posicionar mi sitio en internet?

- Convergencia de Page Rank
- Convergencia de Hits
- Comparación de tiempos

4.1. Convergencia de Page Rank

La convergencia de dicho algoritmo ocurrirá cuando la norma Manhattan de los vectores de la iteración anterior y la del actual sea cero(o a un valor relativamente cerca, esta cercanía estará dada por una tolerancia que para los casos presentados son cero). Es ahí cuando tendremos la respuesta final.
Para evaluar el comportamiento de la norma manhattan variaremos la probabilidad del navegante aleatorio, el cual de ahora en más lo denotaremos como el parámetro **c**.
A continuación se muestran los resultados de como evoluciona la norma a lo largo de las iteraciones y como varía la misma con distintos c, y que luego discutiremos más adelante. Cabe aclarar que expresamos los valores de la norma en escala logarítmica para una mejor visualización y para que se obtenga un mejor entendimiento de como disminuye de a varias magnitudes en cada iteración.

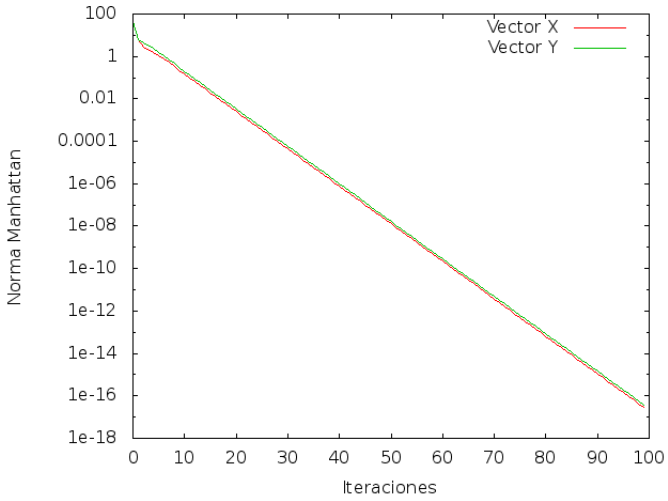


De los casos de prueba presentados, agregamos solo este ya que el resto de los resultados eran iguales. Claramente notamos que la convergencia se retrasa más a medida que se agranda el c.

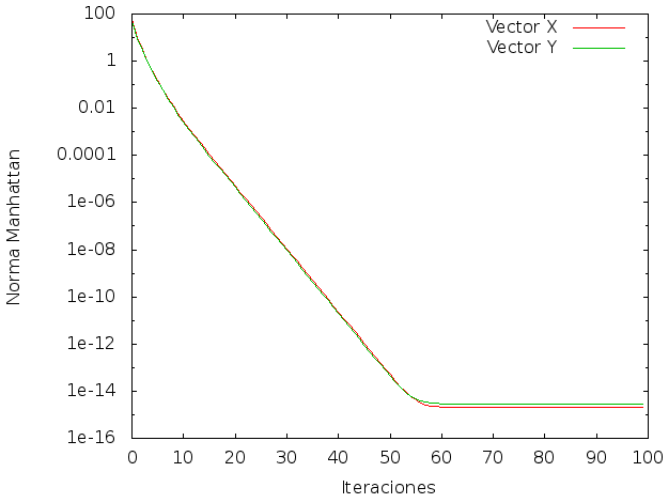
4.2. Convergencia de HITS

La convergencia de dicho algoritmo ocurrirá cuando la norma Manhattan de los vectores x e y (que contienen el puntaje de los sitios de autoridad y los de hubs respectivamente) comparados con los de la iteración anterior sea cero para alguno de los dos(o a un valor relativamente cerca). Es ahí cuando tendremos la respuesta final.
Al igual que el algoritmo anterior utilizamos una tolerancia igual a cero.

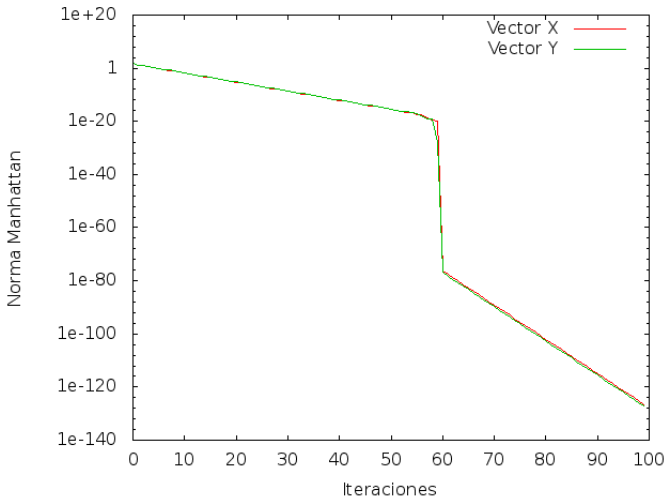
A continuación se muestran los resultados para cuatro instancias distintas, 3 medianas y una grande, de como evoluciona la norma a lo largo de las iteraciones en ambos vectores :



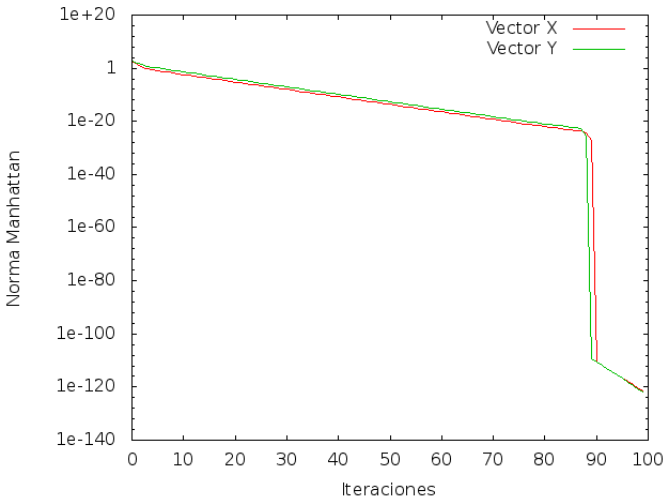
(a) Abortion expanded



(b) Genetic expanded



(c) Movies expanded



(d) Standford

En los últimos dos gráficos podemos observar drásticos descensos del valor de la norma cerca de la iteración 60 en el primero y de la 90 en el segundo. Esto es debido a que el valor de la norma es tan bajo que ya no es medible. Por lo tanto consideraremos que justo antes de esos descensos la norma ya convirgió.

4.3. Tiempos

Inicialmente hicimos pruebas con dos grafos dados por la cátedra, uno chico y uno grande. Los resultados obtenidos fueron los siguientes:

- HITS Abortion(2000 nodos) = 0.694 segundos
- PAGE RANK Abortion(2000 nodos) = 0.104 segundos
- INDEG Abortion(2000 nodos) = 0.032 segundos
- HITS Berkstan(685230 nodos) = 751.63 segundos
- PAGE RANK Berkstan(685230 nodos) = 89.23 segundos
- INDEG Berkstan(685230 nodos) = 8.123 segundos

Esto parecería indicarnos que tanto para grafo grandes como chicos, PageRank es más rápido que HITS. Para corroborar esto pensamos en hacer mas pruebas con instancias de distintos tamaños.
El siguiente gráfico muestra la evolución del tiempo de computo en función del tamaño de la red para cada algoritmo. La red utilizada en todos los casos es una red estrella en la que todos los nodos (o sitios) apuntan al primero de ellos. Utilizamos este tipo de grafo a modo de ejemplo para tener un caso incial de prueba.

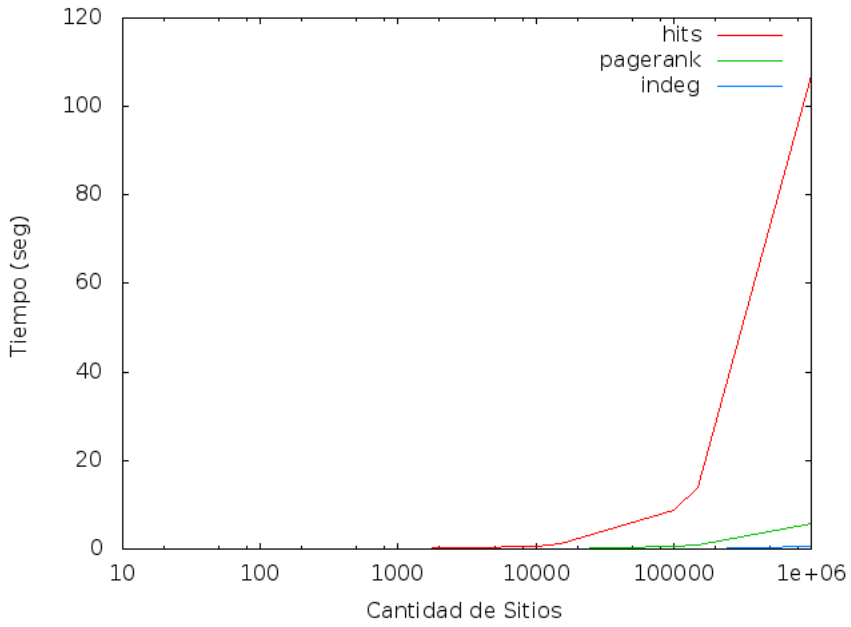


Figura 2: Tiempo de ejecución en función del tamaño de la red

Como esto no alcanza para sacar ninguna conjetura debido a que es para un caso particular pensamos en hacer mas pruebas. Lo que provoca que el cálculo sea mas o menos complejo reside en variables desconocidas por nosotros y la cátedra debido a su complejidad. Por lo tanto pensamos en realizar el siguiente test lo mas genérico posible. Para eso hicimos un script (*testTiempos.cpp*) que que va generando grafos aleatoriamente de un tamaño dado y les aplica los 3 algoritmos midiendo sus tiempos de computo.

Su pseudocógdigo sería:

Algorithm 6 TestGrafosRandom

```
1:  for i de 10 a 11000 do
2:      grafo = crearGrafoSinAristas(cantidadNodos=i);
3:      for por cada nodo de grafo do
4:          cantidadAristas = random(entre 0 e i);
5:          for j de 0 a cantidadAristas do
6:              nodo2 = obtenerUnNodoRandom();
7:              conectar(nodo,nodo2);
8:      medirTiempo(pageRank(grafo));
9:      medirTiempo(HITS(grafo));
10:     medirTiempo(indeg(Grafo));
```

El resultado obtenido fue el siguiente:

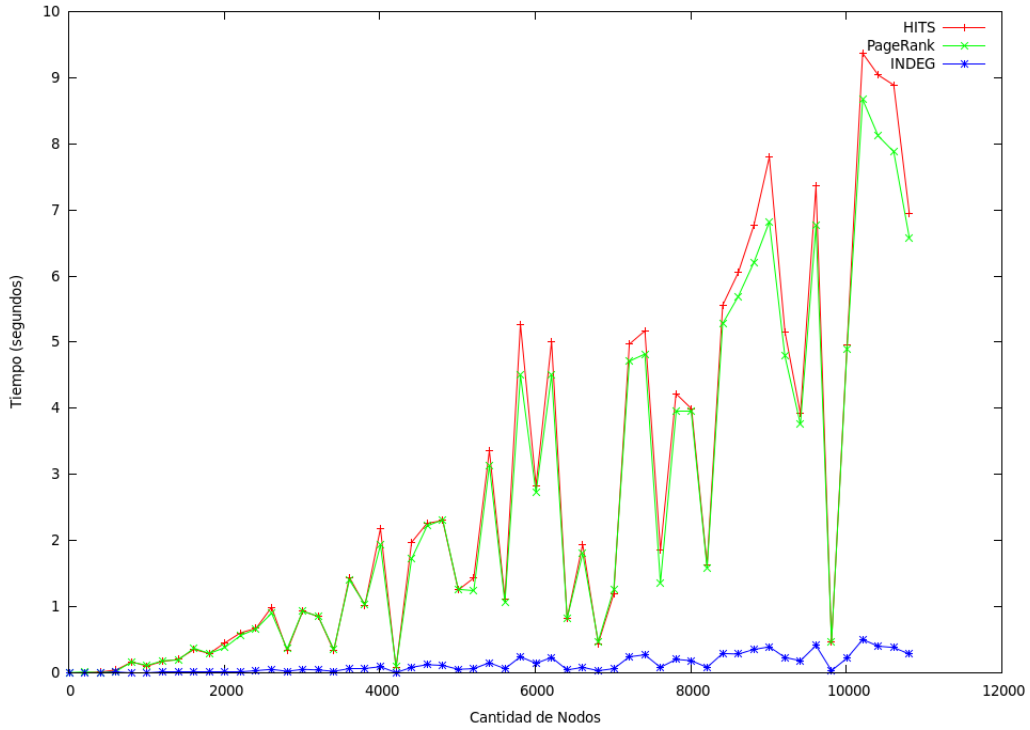


Figura 3: Tiempo de ejecución en función de la cantidad de nodos

5. Discusión

5.1. Convergencia de PageRank

Es importante notar que para distintos tamaños de redes las iteraciones que toma para converger son muy parecidas para el mismo c. Acá solo mostramos dos ejemplos (uno grande y uno chico) pero probamos otros casos y nos dió lo mismo.
Así como es lo mismo para distintas redes (fijando el c), es notable como para C chicos las iteraciones son pocas, sin embargo, el crecimiento es exponencial (o más a veces) en relación al crecimiento del c. Aquí nos referimos a la cantidad de veces que realizamos el método de la potencia.

5.2. Convergencia de HITS

En todos los casos podemos observar que tanto el vector de hubs como el de autoridades convergen de forma muy similar, sólo en la instancia grande hay una pequeña diferencia pero es bastante despreciable. Por otro lado podemos ver que los casos en los que mas drásitca es la convergencia (abortion y genetic) los valores inciales de la norma manhattan son muy altos (alrrededor de 100), provocando asi que se equiparen con las que comienzan en valores mas bajos pero convergen mas lentamente (movies y standford). En estos dos últimos casos además podemos notar grandes saltos de convergencia pasando en pocas iteracion de $1e^{20}$ a menos de $1e^{80}$, entiendiendo, aca sí, que la diferencia es totalmente despreciable y el valor obtenido ya ha convergido. De todas formas consideramos que puede ser un punto de interés para analizar mas en profundidad ya que más allá de decir que entendemos de eso, no sabríamos explicar porque se produce ese salto.

5.3. Comparación de Tiempos

En todos los resultados el tiempo de PageRank fué menor o igual al de HITS. En los primeros dos casos (el de la cátedra y el grafo estrella) fué significativamente menor. En cambio con los grafos random se mantuvieron más cerca aunque también en muchos casos pagerank estuvo por debajo y en ninguno por arriba.
En el caso random podemos ver que el tiempo fluctúa, es decir, no se mantiene en crecimiento constante como el anterior. Esto es porque si bien la cantidad de nodos va creciendo puede no suceder lo mismo con la cantidad de aristas. Además, como dijimos antes, la complejidad algórtimica reside en variables desconocidas por nosotros, por lo que una instancia podría estar resultando muy compleja mientras que una mas grande mucho más simple debido a esto. De todas formas la idea de esta prueba era simplemente poder corrobar de forma mas genérica que el tiempo de cómputo de pageRank es menor al de HITS. Cosa que efectivamente podemos terminar concluyendo.
Finalmente cabe destacar que, como era de esperar, INDEG es la que posee menor tiempo de cómputo de forma significativa en todas las pruebas. Hasta en el caso random las distintas instancias aleatorias lo van haciendo fluctuar muy poco. Como explicamos antes esto se debe a la simplicidad algorítimca que posee.

5.4. Comportamiento Esperado

A continuación veremos en redes pequeñas como se comporta cada algoritmo para ver si su comportamiento es el esperado.

5.4.1. PageRank

Para mostrar un ejemplo del comportamiento del PageRank generamos una red de 11 nodos y lo corrimos con un $c = 0.85$.

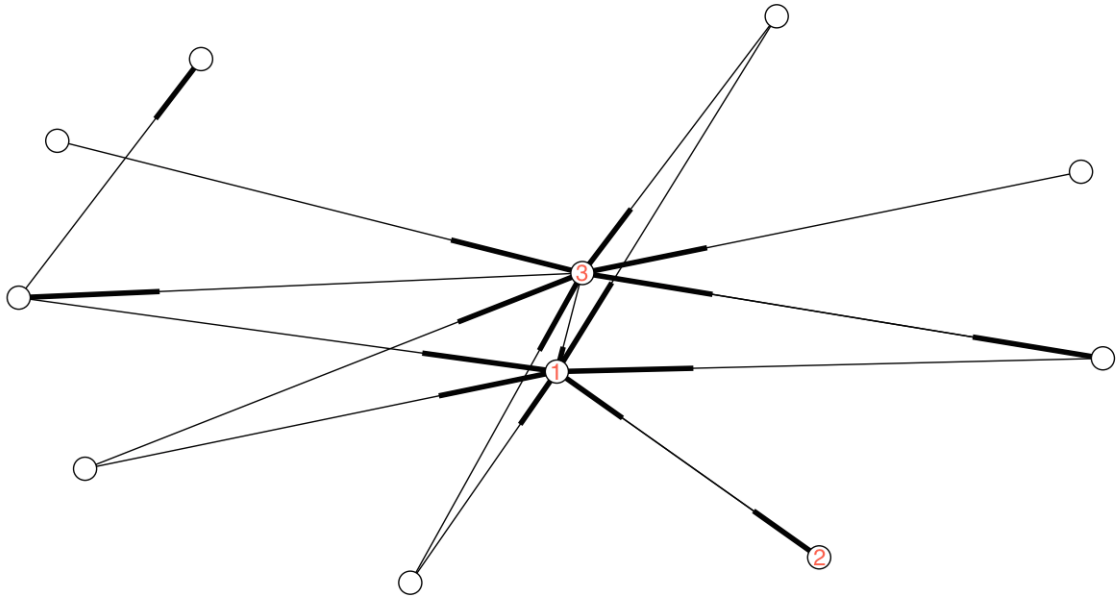


Figura 4: Red de 11 nodos, c=0.85

Lo particular de esta red es que uno *ingenuamente* podría pensar que los dos sitios centrales 1 y 3 van a ser los que mas PageRank obtengan, pero esa suposición se basaría en que el algoritmo solo tiene en cuenta los grados de entrada de cada nodo. El resultado real que se obtiene de esta red es que el orden de PageRank se da por el 1, 2 y 3 (los demás nodos no son importantes para ilustrar el comportamiento). El nodo 2 le gana al 3 ya que la diferencia sustancial es que el 1 que tiene un alto valor lo apunta únicamente al 2, es decir, le da todo el peso que él tiene, mientras que el nodo 3 a pesar de tener muchos sitios que lo apuntan estos son sitios de muy bajo valor de los cuales solo tiene nodos de salida.

5.4.2. HITS

Dada la siguiente red, veamos que nos devuelve HITS:

Resultado obtenido:

$$\begin{bmatrix} & Autoridad & Hub \\ Nodo1 & 0{,}000000 & 0{,}383092 \\ Nodo2 & 0{,}967054 & 0{,}000000 \\ Nodo3 & 0{,}000000 & 0{,}383092 \\ Nodo4 & 0{,}180008 & 0{,}454401 \\ Nodo5 & 0{,}180008 & 0{,}454401 \\ Nodo6 & 0{,}000000 & 0{,}383092 \\ Nodo7 & 0{,}000000 & 0{,}383092 \end{bmatrix}$$

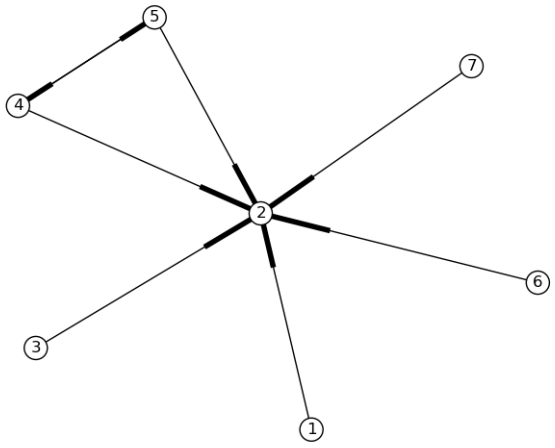


Figura 5: Red de 7 nodos

Efectivamente podemos observar que en la columna de autoridades el nodo 2 es el mayor ya que es el que mas apuntado esta y todos aquellos que tienen 0 es porque no son apuntados por ninguno. Por otro lado en la columna de hubs podemos ver que los nodos 4 y 5 son los que mayor valor tienen ya que son los que mas apuntan a otros nodos con 2 salidas.

5.4.3. INDEG

Veamos el comportamiento dada esa pequeña red.

Resultado obtenido:

$$\begin{bmatrix} & Puntaje \\ Nodo1 & 0{,}000000 \\ Nodo2 & 0{,}714285 \\ Nodo3 & 0{,}000000 \\ Nodo4 & 0{,}142857 \\ Nodo5 & 0{,}142857 \\ Nodo6 & 0{,}000000 \\ Nodo7 & 0{,}000000 \end{bmatrix}$$

Este algoritmo es bastante claro de interpretar y deja todo caso trivial. El nodo 2 posee mucha más calidad de sitio ya que es el más apuntado, y en el segundo puesto empatando el nodo 4 y 5, por ser aquellos con más sitios apuntándolos, excpetuando el nodo 2. El resto de los nodos no reciben ningún tipo de link hacia ellos, por lo que su puntaje es cero.

5.5. Comparación de calidad

En esta sección procederemos a discutir sobre la calidad de resultados que obtenemos de cada algoritmo y luego los compararemos entre si.
Como el objetivo de este trabajo práctico esta enfocado al ranking web que se le asigna a los distintos

sitios de internet, consideramos como buenos resultados aquellos que aparecerían en la primer página de los buscadores, es decir, los primero 10 resultados serán los que consideraremos para el análisis.

5.5.1. PageRank

Según el paper de Bryan y Leise, quienes proponen el algoritmo, lo más común es que el valor del navegante aleatorio sea de 0.15. Por lo tanto creemos que con este valor es donde aparecerán los mejores resultados, pero también veremos que sucede con valores de 0.5 y 0.85, ya que estos valores indican por un lado que la probabilidad del navegante entre quedarse e irse es equiprobable y por otro lado es el inverso de lo que ellos consideran como el valor más común. En valores de 0 y 1 no tendrían sentido el análisis ya que por un lado daría la matriz original y por el otro una matriz equiprobable.
El caso de prueba que utilizaremos es el dado por la cátedra, **Death Penalty**, y lo elegimos ya que es un tema bastante discutido donde se pueden encontrar resultados interesantes.

Resultados con un c=0.15

1. http://www.amnesty.org
   Amnesty International On-line: human rights website

2. http://www.quaker.org/fcadp
   Friends Committee to Abolish the Death Penalty

3. **No relacionado con el tema**
   http://www.santegidio.org/solid/pdm/pdm.htm
   Sito spostato

4. http://www.aclu.org/issues/death/hmdp.html
   Death Penalty and the ACLU

5. http://sun.soci.niu.edu/ critcrim/dp/dp.html
   Death Penalty Information (from: http://www.soci.niu.edu/ critcrim)

6. http://www.ncadp.org National Coalition To Abolish the Death Penalty
7. http://www.ccadp.org Canadian Coalition Against the Death Penalty
8. http://www.deathpenalty.org Death Penalty Focus
9. http://www.smu.edu/ deathpen Death Penalty News & Updates
10. http://www.derechos.org/dp Death Penalty Links

Resultados con un c=0.5

1. http://www.amnesty.org
   Amnesty International On-line: human rights website

2. http://www.quaker.org/fcadp
   Friends Committee to Abolish the Death Penalty

3. **No relacionado con el tema**
   http://www.santegidio.org/solid/pdm/pdm.htm
   Sito spostato

4. http://www.aclu.org/issues/death/hmdp.html
   Death Penalty and the ACLU

5. **No relacionado con el tema**
   http://www.web.amnesty.org/rmp/dplibrary.nsf/index?openview
   Empty title field

6. http://sun.soci.niu.edu/ critcrim/dp/dp.html
   Death Penalty Information (from: http://www.soci.niu.edu/ critcrim)

7. http://www.ccadp.org
   Canadian Coalition Against the Death Penalty

8. http://www.ncadp.org
   National Coalition To Abolish the Death Penalty

9. http://www.smu.edu/ deathpen
   Death Penalty News & Updates

10. http://www.derechos.org/dp
    Death Penalty Links

Resultados con un c=0.85

1. http://www.amnesty.org
   Amnesty International On-line: human rights website

2. **No relacionado con el tema**
   http://www.web.amnesty.org/rmp/dplibrary.nsf/index?openview
   Empty title field

3. http://www.quaker.org/fcadp
   Friends Committee to Abolish the Death Penalty

4. **No relacionado con el tema**
   http://notes.nacdl.org
   Access to Members Only Areas

5. http://www.criminaljustice.org
   National Association of Criminal Defense Lawyers On-line

6. **No relacionado con el tema**
   http://www.ewg.org
   Environmental Working Group

7. **No relacionado con el tema**
   http://www.santegidio.org/solid/pdm/pdm.htm
   Sito spostato

8. http://www.aclu.org/issues/death/hmdp.html
   Death Penalty and the ACLU

9. http://sun.soci.niu.edu/ critcrim/dp/dp.html
   Death Penalty Information (from: http://www.soci.niu.edu/ critcrim)

10. **No relacionado con el tema**
    http://www.ablexbooks.com
    Ablex Publishing Welcome Page

En base a los resultados se puede ver como a medida que aumenta el c empiezan a aparecer resultados que poco tienen que ver con el tema directamente, ya que puede estar relacionado de alguna forma o no diferenciarse tanto del eje temático.
Nos pareció extraño que aparece siempre muy bien posicionado el sitio web Sito spostato, que nada tiene que ver con el tema de la pena de muerte, por lo tanto decidimos hacer un foco especial en este para ver porque sucedía esto y llegamos a la conclusión que es debido a que el factor mas determinante es que gran cantidad de sitios referidos al tema y a su vez bien posicionados (aunque fuera del top 10) apuntaban al mismo, y por lo tanto le daban bastante peso a Sito spostato.
Aunque se pueda ver que con un c menor los resultados tienen relación con el tema, deja claro que sitios como Google o todos aquellos que utilicen este tipo de algoritmos, trabajan bastante sobre el resultado del algoritmo para eliminar sitios de SPAM o mejorar la calidad todavía más.

5.5.2. HITS

Este analásis de calidad lo haremos sobre el tema death penalty. Veremos cuales son los primeros 5 resultados que devuelve HITS en cuanto a autoridades y hubs.

Resultados Hubs

1. http://www.clarkprosecutor.org/html/links/dplinks.htm
   Death Penalty Links
2. http://faculty.etsu.edu/blankenm/deathlinks.htm
   Death Penalty Links
3. http://coramnobis.com/portal/deathpen.html
   A Capital Defender's Toolbox: criminal defense death penalty litigation online resource center
4. http://info-s.com/deathpenalty.html
   The Info Service
5. http://members.xoom.com/ccadp/links.htm
   Canadian Coalition Against the Death Penalty - Collection of Links

Resultados Autoridades

1. http://sun.soci.niu.edu/ critcrim/dp/dp.html
   Death Penalty Information
2. http://www.aclu.org/issues/death/hmdp.html
   Death Penalty and the ACLU
3. http://www.ncadp.org
   National Coalition To Abolish the Death Penalty
4. http://www.smu.edu/ deathpen
   Death Penalty News and Updates
5. http://www.deathpenalty.org
   Death Penalty Focus

Aquí podemos observar claramente que la calidad además de ser, a priori, buena y correcta, tiene coherencia. La mayoría de los hubs sobre el tema son conjunto de links sobre pena de muerte, servicios de información o central de recursos sobre litigios en penas de muerte. Por otro lado, las autoridades son diarios con noticias y novedades, organizaciones enfocadas a eso o páginas institucionales (.edu). Ambos tienen sentido en sus categorías ya que es totalmente lógico que páginas institucionales sean fuentes propias de información que son citadas por otras (osea autoridades) o que una central de recursos linkee a muchos otros citios (osea un hub).
También es de destacar que ningún link pareciera ser spam, o sobre algo no relacionado.

5.5.3. Indeg

Resultados

1. http://sun.soci.niu.edu/ critcrim/dp/dp.html
   Death Penalty Information (from: http://www.soci.niu.edu/ critcrim)

2. http://www.aclu.org/issues/death/hmdp.html
   Death Penalty and the ACLU

3. http://www.ncadp.org
   National Coalition To Abolish the Death Penalty

4. http://www.amnesty.org
   Amnesty International On-line: human rights website

5. http://www.smu.edu/ deathpen
   Death Penalty News & Updates

6. http://www.quaker.org/fcadp
   Friends Committee to Abolish the Death Penalty

7. http://www.law.cornell.edu/topics/deathpenalty.html
   LII: Law about...the Death Penalty

8. http://www.deathpenalty.org
   Death Penalty Focus

9. http://www.derechos.org/dp
   Death Penalty Links

10. http://www.cuadp.org
    CUADP

5.5.4. Comparación

Si comparamos los resultados del PageRank con un C alto claramente da mejores respuestas el HITS. Sin embargo en el paper de Bryan y Leise[3] se recomienda un C de 0.15, con este valor los resultados mejoran notoriamente aunque sigue habiendo algunos que no corresponden o son SPAM. Bajo este ejemplo podemos notar que tienen resultadoes muy similares, aunque Page Rank tiene SPAM en su resultado y HITS no. Esto tiene sentido en la medida que el mismo paper de HITS recomienda el uso de su algoritmo en sets chicos y a priori parece tener mejor resultado HITS. Esto a su vez es subjetivo pero nos parece interesante que sitios que parecen representar bien el tema como http://www.deathpenalty.org y http://www.ncadp.org aparecen en ambos resultados. En cuanto a INDEG, es claro que eligió los más apuntado, en este contexto obtuvo resultados buenos, pero no obstante notar que muy facilmente podría cambiar este resultado con pocos cambios en la subred

6. Conclusiones

6.1. PageRank

Este algoritmo resultó bastante sencillo de implementar, una gran parte del mismo fue la implementación de la matriz esparsa para mejorar tanto la performance espacial como la temporal en el cálculo del método de la potencia.
En cuanto a tiempos es bastante estable y podemos deducir bajo las pruebas realizadas que debería seguir siendo así para grafos aún más grandes. A contrapartida, la calidad si bien es muy buena para c=0.15, lo que confirma el comentario en el paper original, notamos que hay un trabajo grande por encima en los buscadores reales pero es un muy buen punto de partida. Nos referimos a posicionamiento por publicidad, eliminación de SPAM, etc.

6.2. HITS

Una parte importante sobre este algoritmo es que tarda mucho para nodos grandes, sin embargo no debemos olvidar que en su paper[2] Kleinberg habla de que este algoritmo debe ser aplicado no sobre toda la red sin sobre un subconjunto de la misma (*root set*) obtenido de una busqueda incial. Por lo tanto si acotamos el análisis a los grafos mas acotados podemos ver que el tiempo de computo es aceptable y hasta muy parecido al de page rank.

6.3. INDEG

Este algoritmo es bastante simple y en una red chica y confiable puede llegar a valer. Es muy rápido y en caso de necesitar algún dato rápido, es muy fácil de implementar. Igualmente tiene mucho peso la confiabilidad, ya que es muy simple de crecer tu puntaje, simplemente comprando un lugar mínimo en la mayor cantidad de páginas posibles.
En cuanto a estrategia para mejorar el posicionamiento de tu sitio bajo este algoritmo es conseguir la mayor cantidad de páginas para que te apunten sin importar qué calidad de sitio.

6.4. Mejor estrategia para comprar links

En esta sección intentaremos posicionar el sitio wachiturros dentro del data-set Death Penalty y cómo queda en cada estrategia propuesta.

6.4.1. PageRank

Intentaremos analizar distintas instancias de una red para analizar cual es la estrategia más conveniente a la hora de comprar links para aumentar el PageRank Como explicamos anteriormente, para el algoritmo de PageRank es más importante la calidad del sitio de entrada antes que la cantidad. Por lo tanto para encontrar la mejor estrategia intentaremos ver como se comporta la red para un sitio en especial variando los links que lo apunta. Para esto nos quedaremos con los primeros sitios de mayor pagerank de la red original y entre esos iremos variando entre diferentes conjuntos que tengan diferentes cantidades de links de salida y observando en que posición final queda nuestro sitio

Ahora hagamos las siguientes prueba, apuntemos a los Wachiturros por distintas combinaciones de las primeras 10 posiciones que muestran los resultados del set de datos y analicemos en la posición que queda el sitio en cada caso:

$$\begin{bmatrix} & Ranking \\ 10\ sitios & 12^{o} \\ 5\ primeros & 27^{o} \\ 5\ segundos & 65^{o} \\ Mejor\ sitio & 122^{o} \\ 5\ menos\ salidas & 16^{o} \\ 4\ menos\ salidas & 19^{o} \\ 3\ menos\ salidas & 34^{o} \end{bmatrix}$$

Como se puede ver por los resultados, el mejor posicionamiento se consigue cuando los 10 sitios nos apuntan pero a su vez es el que mayor costo tiene y comparando con los demás resultados podemos ver que se consiguen muy buenos resultados cuando priorizamos los nodos que tienen menos links de salidas, sobre todo el caso en que solamente con que 4 nodos de los 10 que no tienen salidas nos apuntan figuramos en una posición muy cercana a la lograda comprando todos los links y tienendo un costo bastante menor ya que además esos sitios no son los de posiciones altas lo que se traduce en un menor costo por cada uno.
Por lo tanto, se puede concluír que al momento de decir que sitios nos deben linkear, debemos optar por los que están bien posicionados pero a su vez tienen muy pocas salidas, ya que este último es un factor muy determinante en el algoritmo de PageRank.

6.4.2. Hits

Si el algoritmo aplicado en la red fuese HITS lo recomendable al cliente sería que negocie con los principales HUBS para que apunten a su sitio. Logrando así rankear mejor en la sección de Autoridades sobre el tema. No le recomendaríamos que negocie con las páginas autoridades ya que dificilmente estas accediesen debido a que de esta manera se estarían restando puntos en el ranking de autoridades.
Por ejemplo en el caso de death penalty habría que negociar con alguno de los 3 principales HUBS: clarkprosecutor.org, faculty.etsu.edu o coramnobis.com. Teniendo en cuenta el costo de cada uno, ya que si el primero costase el doble o mas que el segundo tal vez convendría más negociar con el segundo y tercero.

$$\begin{bmatrix} & Ranking \\ 3\ primeros\ autoridad & 675 \\ 3\ primeros\ hub & 143 \\ 1er\ hub & 273 \\ 2do\ y\ 3er\ hub & 215 \end{bmatrix}$$

Referencias

[1] http://personales.upv.es/p̃edroche/inv/ _docs/fpedrochev4(sema).pdf

[2] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. J. ACM, 46(5):604,632, September 1999.

[3] Kurt Bryan and Tanya Leise. The linear algebra behind google. SIAM Review, 48(3):569 581, 2006.

[4] Burdan y Faires, Numerical Analysis, 3rd edition, 453 467, 2002.

[5] Kamvar, Haveliwala - 2003 - Extrapolation methods for accelerating PageRank computations