DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico I

PageRank

Métodos Numéricos

Primer Cuatrimestre de 2018

Integrante	LU	Correo electrónico
Serio, Franco Agustìn	215/15	francoagustinserio@gmail.com
Ruiz Ramirez, Emanuel David	343/15	ema.capo2009@hotmail.com
Goldstein, Brian Uriel	27/14	brai.goldstein@gmail.com
Fadel, Uriel	104/14	uriel.fadel@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. Introducción Teórica

Palabras Claves: Eliminación Gaussiana - PageRank - Modelo del navegante aleatorio

Resumen: El siguiente trabajo propone que dado un conjunto de paginas web con enlaces entre estas, obtener y analizar rankings de paginas web utilizando PageRank para saber la importancia de las paginas. Teniendo en cuenta el Modelo del navegante aleatorio para poder capturar el comportamiento de una persona cuando navega en la web. Y se hara énfasis en la cantidad de paginas, cantidad de enlaces entre ellas y en la posibilidad que una persona decida ir a una pagina cualquiera.

1.1. Método de Eliminación Gaussiana clásico

El método de eliminación gaussiana para resolver sistemas lineales consiste en llevar el sistema A.x=b al sistema A'.x=b', donde A' es una matriz triangular superior.

Ademas se sabe que el conjunto de soluciones del nuevo sistema es equivalente al original.

Para poder pasar del primer sistema al segundo, se tienen las siguientes operaciones sobre filas de la matriz:

- Multiplicar una de las filas por una constante no nula
- Sumar un multiplo de una de las filas a otra fila

Se podría agregar como tercera operación poder Intercambiar 2 filas pero esto solo es necesario cuando se hace uso de la eliminación gaussiana con pivoteo que sera explicado en la sección Desarrollo.

2. Desarrollo

Para saber que **la matriz (4) del enunciado** es equivalente a la matriz **e.z^t+p.W.D**. Se hacen los siguientes operaciones:

- Cada posición de la matriz W se puede escribir como w_{ij}
- Si se multiplica W.D, el elemento d_{jj} multiplica a la col$_j$(W), lo cual se escribe $w_{ij}.d_{jj} = \begin{cases} w_{ij}/c_j & si \quad c_j \neq 0 \\ 0 & si \quad c_j = 0 \end{cases}$
- Si se multiplica por un escalar p.W.D, este escalar multiplica a cada elemento de W.D y esto se escribe $p.w_{ij}.d_{jj} = \begin{cases} p.w_{ij}/c_j & si \quad c_j \neq 0 \\ 0 & si \quad c_j = 0 \end{cases}$
- Si se le suma una matriz e.z^t+p.W.D, esta matrices se suman posición a posición obteniendo $(e.z^t)_{ij} + p.w_{ij}.d_{jj} = 1.z_j + p.w_{ij}.d_{jj} = \begin{cases} (1-p)/n & si \quad c_j \neq 0 \\ 1/n & si \quad c_j = 0 \end{cases} + \begin{cases} p.w_{ij}/c_j & si \quad c_j \neq 0 \\ 0 & si \quad c_j = 0 \end{cases} = \begin{cases} (1-p)/n + p.w_{ij}/c_j & si \quad c_j \neq 0 \\ 1/n & si \quad c_j = 0 \end{cases}$

La siguiente matriz I-p.W.D escrita posición a posición es (I-p.W.D)$_{ij} = \begin{cases} 1 & si \quad i = j \\ -p.w_{ij}/c_j & si \quad c_j \neq 0 \wedge i \neq j \\ 0 & si \quad c_j = 0 \wedge i \neq j \end{cases}$

La suma de los módulos de los elementos de la col j es $\sum_{i=1}^{n} |(I - p.W.D)_{ij}|$ y hay dos posibilidades:

- si $c_j = 0$ la sumatoria da 1, porque como la cantidad de links salientes de la pagina j es 0, entonces hay ceros en toda la columna excepto en la posición (j,j).

  (I-p.W.D)$_{jj}$=1 y la suma del resto de los elementos de la columna j es 0. Por lo tanto la matriz es estrictamente diagonal dominante por columnas(e.d.d).

- si $c_j \neq 0$ la sumatoria da $1 + \sum_{\substack{i=1 \\ i \neq j}}^{n} |-p.w_{ij}/c_j| = 1 + \sum_{\substack{i=1 \\ i \neq j}}^{n} |-p|.|w_{ij}|/|c_j| = 1 + |-p|/|c_j| \sum_{\substack{i=1 \\ i \neq j}}^{n} |w_{ij}| =$

  $1 + p/c_j \sum_{\substack{i=1 \\ i \neq j}}^{n} w_{ij} = 1 + (p/c_j).c_j = 1 + p$ porque p y c_j son mayores estricto a 0, w_{ij} es 0 o 1, y $\sum_{\substack{i=1 \\ i \neq j}}^{n} w_{ij}$ es c_j, es decir la cantidad de links de la pagina j.

  (I-p.W.D)$_{jj}$=1 y la suma del resto de los elementos de la columna j es p. Por enunciado p<1, por lo tanto la matriz es e.d.d por columnas.

En cualquiera de los dos casos anteriores por ejercicio 12 de Practica 2, se deduce que la matriz I-p.W.D tiene factorizacion LU sin pivoteo. Esto quiere decir que **si aplicasemos Eliminación Gaussiana sobre esta matriz, obtenemos una matriz con ceros abajo de la diagonal sin haber tenido que pivotear**.

Por el ejercicio 17a, $|| - p.W.D||_1 = max_j \sum_{i=1}^{n} |-(p.W.D)_{ij}| = p$

Como $|| - p.W.D||_1$=p < 1. Por el ejercicio 21a de la Practica 2, I-p.W.D es inversible y por el ejercicio 21b de la Practica 2, $||(I - p.W.D)^{-1}||_1 = \frac{1}{1 - || - p.W.D||_1} = \frac{1}{1-p}$

Como I-p.W.D es inversible podemos calcular su numero de condición:
$\kappa_1(I - p.W.D) = ||I - p.W.D||_1.||(I - p.W.D)_1^{-1}||$ por desigualdad triangular
$\leq (||I||_1 + || - p.W.D||_1).||(I - p.W.D)^{-1}||_1$
$= (1 + p).||(I - p.W.D)^{-1}||_1$

$\leq (1+p).\frac{1}{(1-p)}$
$=\frac{1+p}{1-p}$
Si se considera a $\frac{1+p}{1-p}$ como una función de p
$\lim_{p\to 0^+} \frac{1+p}{1-p} = \frac{1}{1} = 1$ y $\lim_{p\to 1^-} \frac{1+p}{1-p} = \frac{1}{0^+} = +\infty$
$\left(\frac{1+p}{1-p}\right)' = \frac{1.(1-p)-(1+p).(-1)}{(1-p)^2} = \frac{(1-p)+(1+p)}{(1-p)^2} = \frac{2}{(1-p)^2}$
Como $(1-p)^2 > 0$, la derivada es > 0 en (0,1), entonces la función es estrictamente creciente en (0,1).

Por lo tanto, **la matriz va a estar bien condicionada** cuando su numero de condición esta cerca de 1 y esto sucede **cuando p esta cerca de 0**. Por otro lado, cuando p se acerca a 1, $||I - p.W.D||_1$ podría estar lejos de 1, es decir que $I - p.W.D$ podría estar mal condicionada.

Se utilizara eliminación gaussiana para transformar el sistema de ecuaciones (6) del enunciado en uno equivalente.

2.1. Análisis de algoritmos utilizados

2.1.1. Algoritmo de Gauss:

El procedimiento consiste en iterar por cada fila de la matriz buscando dejar en 0 todas las posiciones por debajo de la diagonal.

Sea $A \in R^{nxn}$. Y sea A^k, $0 \leq k \leq n-1$, la matriz luego de realizados k pasos del procedimiento,. En la primer iteración, realiza el siguiente cálculo:

$Fila_i = Fila_i - \frac{a_{i1}}{a_{11}}.Fila_1$, con 2 ≤ i ≤ n

De esta manera logra poner en 0 a todas las posiciones A_{i1} con $i > 1$. Luego, para la iteración k:

$Fila_i^k = Fila_i^{k-1} - \frac{a_{ik}^{k-1}}{a_{kk}^{k-1}}.Fila_k^{k-1}$, con 2 ≤ k < i

Como se obtuvo una matriz triangular superior se usa *Backward-substitution*, que resuelve este tipo de matrices para obtener el vector solución.

2.1.2. Algoritmo de Gauss con pivoteo parcial:

Sea $A \in R^{nxn}$. Tiene el mismo objetivo que el algoritmo anterior con la diferencia que en el paso k-esimo antes de dejar ceros por debajo de la posición (k,k). Se busca $_{k\leq i\leq n}^{max} |a_{ik}|$ para intercambiar la fila donde esta ese elemento con la fila k. Una vez hecho esto se procede a dejar ceros debajo de la posición (k,k) como se explica en la sección anterior.

2.2. Detalles de Implementación

Utilizamos C++.

La estructura de las matrices ralas fue un entero para la cantidad de filas y otro para la cantidad de columnas. Un vector del tamaño de la cantidad de filas y otro de la cantidad de columnas. Para ambos vectores en cada posición había un map con clave entero y significado double. Para el primer vector, cada posición del vector representaba una fila de la matriz cuyos elementos son distintos de 0, es decir que el map tiene como clave la posición columna y como significado al elemento. Para el segundo, cada posición representaba una columna de la matriz cuyos elementos son distintos de 0, es decir que el map tiene como clave la posición fila y como significado al elemento. Donde entre ambos vectores hay consistencia para que representen la matriz.

Los aspectos fundamentales que nos llevaron a tomar esta determinación fueron:

1. Permite acceder rápidamente a los elementos del matriz (ya que la matriz es rala).
2. Utilizar *precisión doble* asegura menos error de redondeo. La densidad de valores representables aumenta considerablemente en contraposición a su par de *precisión simple*.
3. Con la estructura elegida para la esparsa, la matriz puede entrar en memoria.
4. Como una matriz rala tiene pocos elementos distintos a 0, si se quisiera hacer un recorrido por filas o columnas, se puede usar los iteradores para saltar al siguiente elemento distinto a 0
5. Para la matriz esparsa, cuando se quiera hacer un recorrido por filas o columnas el uso de map garantiza que los iteradores pueden avanzar en orden.

Para recorrer la matriz M, se tuvo en cuenta la cantidad de ceros en la misma, para no efectuar operaciones de más, ni recorrer los componentes en 0.

Para poder acceder a un elemento que esta en la posición (x,y): se accede a la posición 'x' del vector cuya longitud es la cantidad de filas, obteniendo la fila 'x' que es un map. En este map, se busca la clave 'y'. En caso de encontrarla se devuelve el significado de esa clave, es decir el elemento que esta en esa posición. Si no se la encuentra, se devolverá 0 lo cual quiere decir que en esa posición había un 0.

$\text{máx}_{\{P_t^\circ(j)\}}$ $\overset{aaa}{bb}$

2.3. Metricas de experimentos

Para el experimento cuantitativo, se hizo uso de la funcion clock() de c++ para poder medir el tiempo que tardaba el procedimiento de interes. La funcion clock() devuelve la cantidad de ticks de reloj que pasaron hasta el momento que se lo invoca. Para conseguir números aleatorios se uso la función rand y srand (seed random) de la stdlib. Todos estos experimentos se corrieron en las pcs de los laboratorios.

3. Experimentación

3.1. Resultados Cuantitativos

3.1.1. Hipótesis 1

Hipótesis 1: A pesar de que no era requerido en éste trabajo práctico, se implementó una función de eliminación gaussiana con pivoteo. Para las matrices de este trabajo practico, eliminación gaussiana sin pivoteo tardaría menos en terminar que eliminación gaussiana con pivoteo. Donde ambos algoritmos realizan la misma tarea de dejar ceros abajo de la diagonal. El único test que estuvo malogrado fue el test_aleatorio donde la eliminación gaussiana sin pivoteo tardó más que con pivoteo. Ésto creo que se debe a nuestra implementación de la matriz rala, mas precisamente con el map.En matrices mas grandes como las de 15_segundos y 30_segundos ya sí se puede apreciar la diferencia en clocks de la eliminación gaussiana sin y con pivoteo.

3.1.2. Experimentación de la hipótesis 1

El experimento de correspondiente a la **Hipótesis 1** fue el siguiente:

- Tomamos como entrada matrices que crecen en tamaño y con distinta cantidad de links. Estas matrices fueron las provistas por los test.
- A cada matriz se le aplico primero E.G. con pivoteo y después sin pivoteo. En ambos casos se guardo el tiempo que tardaba. La idea fue intercalar método de gauss para evitar que queden almacenadas funciones en cache que puedan ser usadas a posterior. La secuencia en la que se aplico E.G fue aleatorio_desordenado, aleatorio, 15_segundos y 30_segundos.
- Los tiempos se guardaron en un csv y con estos se hicieron los gráficos. Los cuales indican por ejemplo que para una matriz a la cual se le aplico E.G. con pivoteo o sin pivoteo le tomo x cantidad de clocks.

Ademas algo que pone en cierta igualdad a ambos algoritmos es que usan indices para moverse por toda la matriz.

Los **resultados del experimento anterior**, en el cual se compara el tiempo que tardan ambos métodos de Eliminación Gaussiana, se pueden visualizar en los siguientes gráficos:

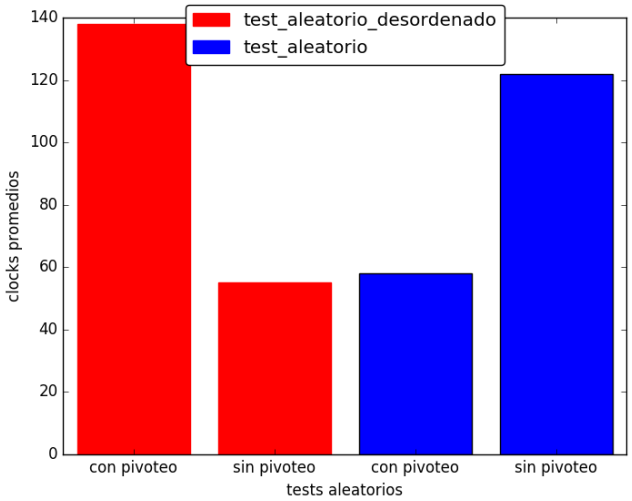

Figura 1: Tiempo que tardan los métodos para la entradas aleatorias

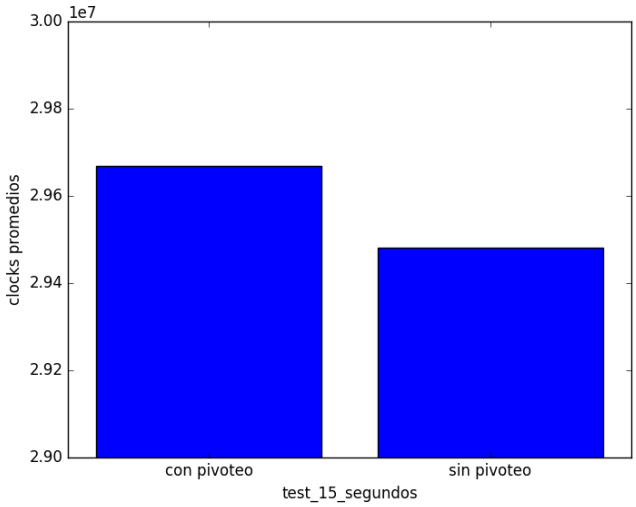

Figura 2: Tiempo que tardan los métodos para la entrada de 15_segundos

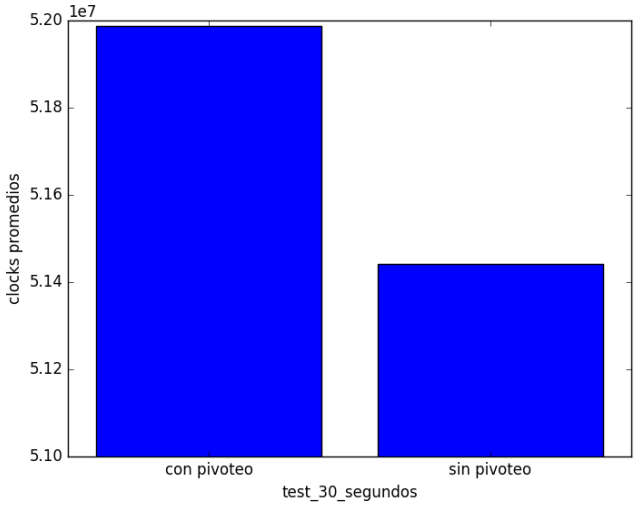

Figura 3: Tiempo que tardan los métodos para la entrada de 30_segundos

3.1.3. Discusión Experimento 1 cuantitativo

Para la experimentación de la **Hipótesis 1** se puede decir que:
Por los gráficos de las figuras 1, 2 y 3, se puede ver que E.G sin pivoteo tarda menos que E.G con pivoteo. Lo cual se debería a que uno no realiza en cada paso el pivoteo y el otro si. Pivotear implicaría un costo lineal para buscar el elemento máximo en una columna y otro lineal para intercambiar la fila. Esto agregaría a E.G con pivoteo un costo lineal en peor caso por paso.

Pero el caso imprevisto que contradice a la hipótesis es el caso de la matriz aleatoria en 1. Donde se puede observar que E.G con pivoteo tarda menos que sin pivotear. Se cree que esto se debería a que como el tamaño de la matriz es chico, para los casos chicos puede ser que no siempre se cumpla la **Hipótesis 1**.

3.1.4. Hipótesis 2

Hipótesis 2: Para el segundo experimento, se hicieron matrices personalizadas con números random. Lo que se quería demostrar en este experimento es cómo aumentan los clocks promedios a medida que aumentan las páginas y a medida que aumentan la densidad del grafo en sí. Estas matrices random se hicieron de 500, 1000, 2000 y hasta 4000 páginas para poder sobrepasar la cantidad de páginas que ofrece la cátedra. A su vez, las densidades elegidas son de 0.1 a 0.9 con intervalos de 0.1. La densidad se definió como cantidad de links dividido la cantidad total de páginas.

3.1.5. Experimentación de la hipótesis 2

El experimento correspondiente a la **hipótesis 2** fue el siguiente:

- Se generaron matrices de los tamaños indicados en la hipótesis. Y se insertaron 1's de acuerdo a la densidad de la hipótesis. Es decir, se creo una matriz de 500 paginas con densidad 0.1, luego otras del mismo tamaño con densidad 0.2, asi hasta llegar a 0.9. Luego se replico este procedimiento con el resto de los tamaños
- Para cada matriz de un tamaño y densidad se probo cuanto tardaba PageRank.
- Y ese tiempo fue enviado a un .csv y se hizo el gráfico. El cual indica que dada una densidad y tamaño, cuanto tiempo tarda en terminar.

Los **resultados del experimento anterior** se pueden ver en el siguiente gráfico:

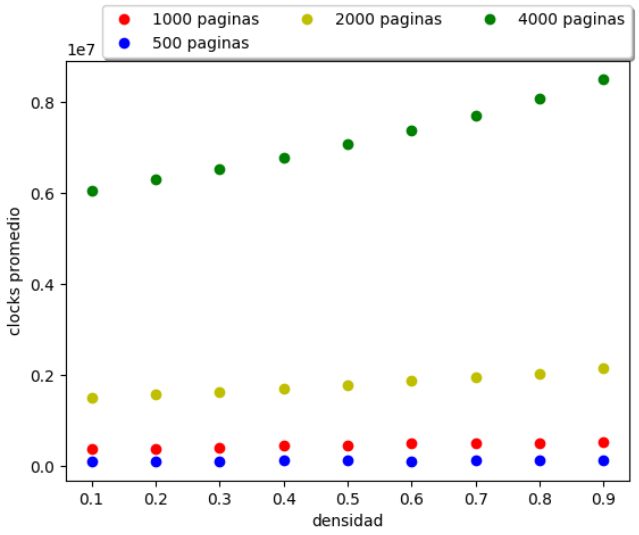

Figura 4: Tiempo que PageRank tarda al variar densidad y tamaño de la matriz

3.1.6. Discusión Experimento 2 cuantitativo

Como se puede apreciar en la figura 4, el aumento de la densidad tiene mucha más incidencia en las matrices con mayor tamaño. Por ejemplo, en el caso de 4000 páginas, al cambiar la densidad los clocks suben mucho más abruptamente que en los otros 3 casos.
Por otro lado, el aumento del tamaño de la matriz tiene un efecto exponencial más que lineal en los clocks promedios que tarda el programa en terminar.

3.1.7. Conclusiones Cuantitativas

Se puede concluir que a medida que crece la matriz y la densidad con la trabaja PageRank va a tardar mas en terminar. Y que E.G sin pivoteo tarda menos que E.G con pivoteo para matrices de gran tamaño.

3.2. Resultados Cualitativos

3.2.1. Planteo de experimentos

En esta sección nos proponemos analizar el algoritmo de manera de entender el criterio con el que el algoritmo ordena los rankings, discutir y observar si estos resultados coinciden con el orden intuitivo que podriamos tener nosotros y experimentar como influye la topologia del grafo y el parametro p en la calidad de los puntajes obtenidos.

3.2.2. Instancias generadas:

Para correr experimentos en esta seccion generamos 3 instancias que nos resultaron interesantes y que corrimos junto a los grafos provistos por la catedra.
grafo_denso: generamos un grafo de 100 nodos con una densidad alta de aristas (mas de 5000). Nos pareció importante evaluar el comportamiento en un grafo denso y los provistos por la catedra no lo son.
triangular_sup: generamos un grafo cuya matriz asociada es triangular superior (excepto la diagonal que tiene 0). la topologia de este grafo es una cadena en la que el cada nodo tiene aristas salientes a todos los elementos siguientes de la cadena (y por lo tanto aristas entrantes solo de los elementos anteriores).
grafo_patologico: Este grafo es similar al anterior pero con la diferencia de que en este se agrega una arista desde el ultimo elemento de la cadena al primero.

3.2.3. Experimento 1

En Este experimento ordenamos los nodos del grafo en ”buckets”por cantidad de aristas entrantes y de cada bucket hacemos un promedio de sus rankings.
Una vez calculados los promedios por de todos los nodos agrupados por cantidad de aristas entrantes, graficamos esos promedios teniendo como eje de abscisas la cantidad de Links entrantes y como ordenadas el ranking.
el objetivo del experimento es buscar correlación entre la cantidad de links Entrantes a un nodo y su ranking por lo que si se observara una linea podriamos concluir correlacion. De no observarse linea o crecimiento sostenido al incrementar la cantidad de linksEntrantes podemos concluir que no hay correlacion entre el grado entrante y el ranking de una pagina.

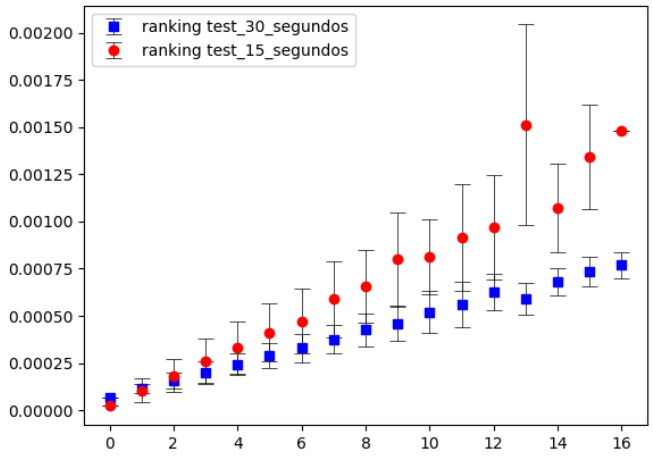

Figura 5: Rankings de los grafos de 30 y 15 seg

observaciones: Se Puede ver en la figura 4 como ambos grafos parecen mostrar un crecimiento del ranking lineal en la cantidad de links entrantes. Esto sugiere correlación.

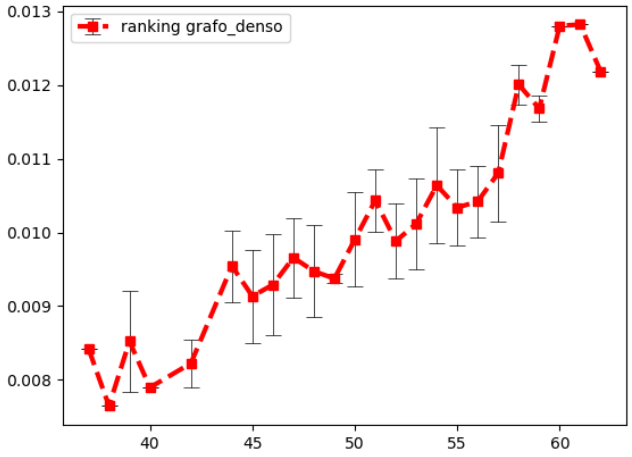

Figura 6: Rankings del grafo Denso

observaciones: Se puede ver en el grafico que el grafo_denso tambien muestra el mismo patron de asemejar un crecimiento lineal al largo plazo (si bien hay picos locales). Esto sugiere correlación.

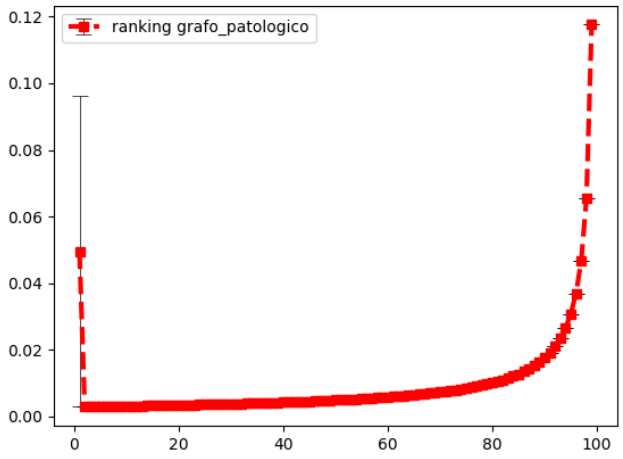

Figura 7: Rankings del caso patologico

En este grafo se puede observar como inducimos un crecimiento exponencial en el ranking para los nodos mas adelantados en la cadena, logrando asi que mientras mas adelante estas (recuerdo que siempre tenes entrantes de todos los anteriores) mas aristas entrantes tenes y mas ranking. La unica excepción es el primer nodo que como le llega todo el peso del ultimo termina siendo el nodo con mas ranking (lranking 0.8).

El gráfico de la matriz triangular superior es igual a este ultimo en forma con la excepción de que el primer nodo no sobresale siendo el de menor ranking. No lo graficamos para no sobrecargar el informe con gráficos redundantes.

3.2.4. Conclusiones experimento 1

Concluimos a partir de los experimentos observados que hay una correlacion clara entre la cantidad de aristas entrantes que tiene un nodo y su ranking asociado. Si bien también el algoritmo permite que nodos con pocas aristas entrantes tengan un ranking alto por el metodo de ponderado de los links.

3.2.5. Experimento 2

En este Experimento pretendemos estudiar el comportamiento del algoritmo de acuerdo a propiedades del grafo y del p elegido.
Como ya observamos en el experimento 1 correlación entre Cantidad de links entrantes y ranking, veremos como varia el ranking en funcion de este y de P.
Para esto corremos el algoritmo haciendo variar el P de a 0.1 y plantearemos los resultados obtenidos para ese P ordenados por cantidad de linksEntrantes en un heatmap donde en las abscisas pondremos la cantidad de links entrantes y en la ordenadas los diferentes valores de P. Ya sabemos (por Exp 1) que al menos para algunos valores de P el grafico deberia colorearse de izquierda a derecha de manera que a la izquierda los colores sean mas claros y a la derecha mas oscuros por la correlación ya establecida. Ahora queremos ver que como incide principalmente el P es decir En la coloración de abajo hacia arriba.
Se tomo el grafo de 15 seg por ser un grafo representativo de la realidad (poco denso, ninguna estructura particular) y no ser extremadamente grande para asi poder correr el algoritmo con todos los diferentes p en un tiempo razonable.

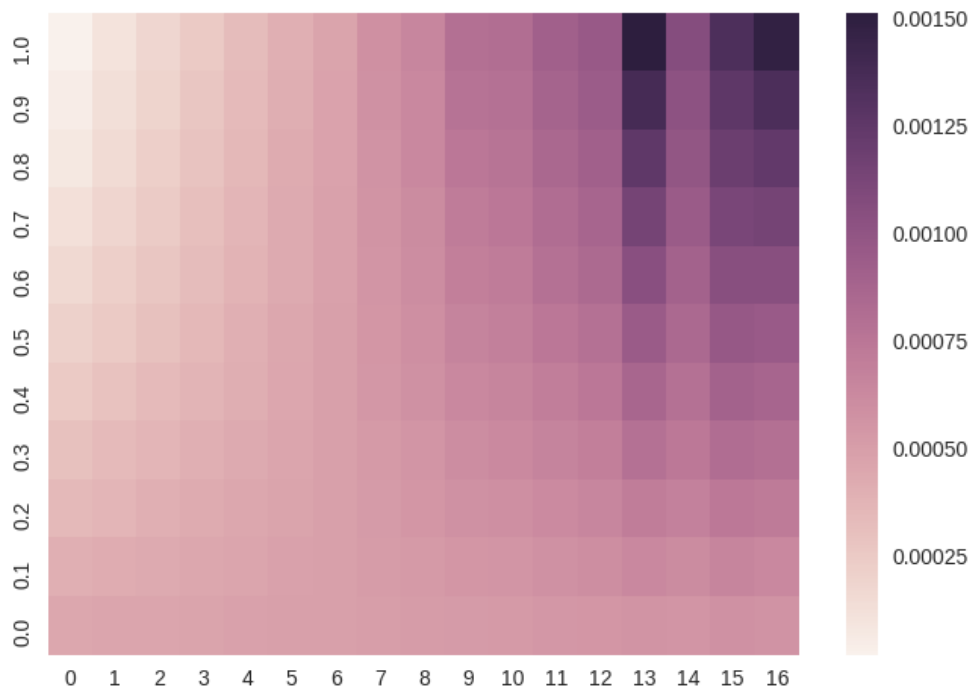

Figura 8: cantidad de links entrantes en las abscisas, p en las ordenadas, ranking el coloreo

Observaciones: Se puede ver en el gráfico como la coloración de izquierda a derecha es la esperada para los P grandes (a partir de $0{,}5$), a partir de de $0{,}8$ para arriba se acentúan demasiado las diferencias si bien se mantiene el orden.
Para p chico 0 , $0{,}1$, $0{,}2$ no logra marcarse bien las diferencias de puntaje y esto podría traer errores ya que los mínimos errores de precisión terminarían dándole el mismo ranking a paginas que no correspon-

de.
por otro lado se observa en el extremo superior izquierdo como las paginas con pocos links entrantes quedan relegadas a casi nada de ranking para P muy grandes, mientras que en el extremo superior derecho el ranking se vuelve muy alto.
Por estas observaciones se sugiere un p que balancee los rankings de modo que la diferencia no sea atroz por tener solo unos algunos links entrantes mas (recuerdo que estos links entrantes se correlacionan a tener .[el] favor"del algoritmo).

3.2.6. Conclusiones experimento 2

Por lo observado podemos concluir que: la correlacion entre cantidad de links y ranking se mantiene independientemente del p (excepto quizás para $p = 0$) pero esta correlacion se acentua mas cuando el p es mas grande.
Concluimos tambien que para que el algoritmo devuelva rankings significativos es recomendable tener un p relativamente grande (mayor o igual a $0{,}6$) y para que la diferencia de unos pocos links entrantes no impacte demasiado en el ranking sugerimos un p no muy cercano a uno. Planteamos el p optimo en el rango $[0{,}6;0{,}7]$.

3.2.7. Preguntas del enunciado

1. ¿Cómo es el ranking obtenido en cada caso de acuerdo a la estructura del grafo páginas?
El ranking obtenido, funciona de manera similar para grafos de distintos tamaños Aunque es posible generar casos patologicos donde el ranking no funciona como uno esperaría intuitivamente.
Por poner un ejemplo tal vez uno esperaría que la cantidad de links entrantes refleje por lo menos parcialmente el ranking sin embargo es posible realizar casos (como el patologico) el el que un nodo con un único link entrante se lleva el 80 % del puntaje total.

2. ¿Qué conclusiones pueden sacar de la interpretación de los resultados?
Como esta expandido en detalle en las conclusiones de los experimentos 1 y 2 (cualitativos). Pudimos concluir por un lado la correlacion entre la cantidad de links entrantes de un nodo y su ranking, si bien tambien vimos es posible generar caso patologicos que rompan esta regla.Por otro lado pudimos concluir del experimento 2 que se verifica el experimento 1 para todo p y que un p chico ($< 0{,}2$) asigna rankings similares a todos los nodos de modo que errores de precisión numérica se volverían mucho mas conflictivos en estos casos.
También concluimos que un P muy cercano a 1 traeria el problema de acentuar mucho la diferencia (en numero) de ranking entre paginas relativamente cercanas (en orden de ranking). Por estas observaciones concluimos un p idoneo al rededor del $0{,}7$.

3.Respecto del ranking de Page: ¿Funciona cómo era esperado? ¿Hubo sorpresas? ¿Qué pueden concluir sobre su significado? ¿Diran que es un buen ranking?
Funciona como era esperado. No hubo sorpresas. Es un ranking util que si bien valora la cantidad de links tambien los pondera de manera que no es facil falsear el sistema inventando paginas que apunten a a tuya porque si son paginas de poco valor y relativamente pocas en una red tan grande como internet, no influirán mucho.
Adicionalmente es un buen algoritmo para tener un punto de comparación contra otros algoritmos de rankeo de paginas que se quieran implementar.
Es un buen ranking siempre y cuando la topologia del grafo no sea una especialmente diseñada para hacerlo andar mal. Es decir para un grafo como podria encontrarse en la realidad , es un buen algoritmo.

4. ¿Cómo es la calidad de los rankings?
Si esta bien elegido el P la calidad de los rankings es buena, en especial cuando el grafo crece en cantidad de nodos. Si el P esta mal elegido, podrian generarse problemas en los que muchas paginas que merecen rankings distintos terminen con el mismo, debido a que por ejemplo un P bajo implica una distribucion de puntajes bantante uniforme (si bien ordenada de la misma forma que con cualquier otro p).

4. Apéndice