Universidad Central de Venezuela

Facultad de Ciencias

Escuela de Computación

Asignatura: Cálculo Científico (6105)

Estudiante: Naranjo Sthory Alexanyer Antonio

Cédula de identidad: V – 26.498.600

**Tarea 6: El autovector de 458 billones de dólares (hasta ahora)**

Para el desarrollo de la solución al problema planteado en el enunciado del ejercicio, se optó por investigar, trabajar e implementar el algoritmo ***PageRank*** desarrollado por los fundadores de Google, *Larry Page* y *Serguéi Brin*. Como siempre, se adjunta además del presente informe, una carpeta con el nombre ***Codes*** donde se almacenan los códigos fuentes escritos bajo el lenguaje de programación Matlab/Octave y una carpeta con el nombre ***Images*** con las imágenes de las gráficas que se hayan generado para sustentar lo explicado en cada sección del actual documento.

***Algoritmo PageRank:* ¿Qué es?**

Una de las razones por las que *Google TM* es un motor de búsqueda tan eficaz es el algoritmo *PageRank TM* desarrollado por los fundadores de Google, *Larry Page* y *Serguéi Brin*, cuando eran estudiantes de posgrado en la Universidad de Stanford. El PageRank se determina enteramente por la estructura de enlaces de la ***World Wide Web*** (www). Se vuelve a calcular una vez al mes y no tiene en cuenta el contenido real de las páginas web ni las o consultas individuales. Entonces, para cualquier consulta concreta, Google encuentra las páginas de la que coinciden con esa consulta y las enumera en el orden de su PageRank.

Imagínese que navega por la web y pasa de una página a otra eligiendo al ***azar*** un enlace saliente de una página para llegar a la siguiente. Esto puede llevar a callejones sin salida en páginas sin enlaces salientes, o ciclos alrededor de camarillas de páginas interconectadas. Así que, una cierta fracción de las veces, simplemente se elige una página al azar de la Web. Este paseo aleatorio teórico se conoce como ***cadena de Markov*** o ***proceso de Markov***. La probabilidad límite de que un navegante aleatorio infinitamente dedicado visite una página es su PageRank. Una página tiene un alto rango si otras páginas con alto rango la enlazan a ella.

***Algoritmo PageRank:* Análisis Teórico**

Sea el conjunto de páginas web a las que se puede llegar siguiendo una cadena de hipervínculos que comienza en una página raíz, y que es el número de páginas en . Para Google, el conjunto varía en realidad con el tiempo, pero en junio de 2004, superaba los .

Sea la matriz de conectividad de una parte de la web, es decir, si hay un hipervínculo a la página desde la página y en caso contrario. La matriz puede ser enorme, pero es muy dispersa. Su *j-enésima* columna muestra los enlaces de la *j-enésima* página. El número de ***nonzeros*** (valor positivo o negativo, no igual a cero) en es el número total de hipervínculos en .

Sean y , las sumas de las filas y columnas de :

Las cantidades y son el grado de entrada y el grado de salida de la página .

Sea la probabilidad de que el paseo aleatorio siga un enlace. Un valor típico es . Entonces es la probabilidad de que se elija una página arbitraria y es la probabilidad de que se elija una página aleatoria concreta.

Sea la matriz de cuyos elementos son,

Obsérvese que A proviene de escalar la matriz de conectividad por sus sumas de columna. La j-enésima columna es la probabilidad de saltar de la página a las demás páginas de la web. Si la página es un callejón sin salida, es decir, no tiene enlaces de salida, entonces asignamos una probabilidad uniforme de a todos los elementos de su columna. La mayoría de los elementos de son iguales a , la probabilidad de saltar de una página a otra sin seguir un enlace.

Si y , entonces

La matriz A es la matriz de probabilidad de transición de la cadena de Markov. Sus elementos están todos estrictamente entre cero y uno y las sumas de sus columnas son todas iguales a uno. Un importante resultado de la teoría de matrices conocido como *teorema de Perron-Frobenius* se aplica a estas matrices. Concluye que una solución no nula de la ecuación,

Existe y es único dentro de un factor de escala. Si este factor de escala se elige de forma que,

Entonces es el vector de estado de la cadena de Markov y es el PageRank de Google. Los elementos de son todos positivos y menores que uno.

Además, el vector es la solución del sistema lineal singular y homogéneo,

***Algoritmo PageRank:* Análisis Práctico (Calculo para hallar *x*)**

Para un modesto, una forma fácil de calcular en Matlab/Octave es empezar con alguna solución aproximada, como los PageRanks del mes anterior, o,

A continuación, basta con repetir la declaración de la asignación,

Hasta que los vectores sucesivos coincidan dentro de una tolerancia determinada. Esto se conoce como el ***método de la potencia*** y es casi el único enfoque posible para muy grandes.

En la práctica, las matrices y nunca se forman realmente. Un paso del método de potencia se haría mediante una pasada por una base de datos de páginas web, actualizando los recuentos de referencia ponderados generados por los hipervínculos entre páginas.

La mejor manera de calcular el PageRank en Matlab/Octave es aprovechar la estructura particular de la matriz de Markov. Este es un enfoque que preserva la dispersión de . La matriz de transición puede escribirse como sigue,

Donde es la matriz diagonal formada a partir de los recíprocos de los grados exteriores,

Donde además, es el n-vector de todos los unos, y es el vector con componentes,

La matriz de rango uno, , da cuenta de las elecciones aleatorias de las páginas web que no siguen los enlaces.

La ecuación,

Puede ser reescrita de la siguiente manera,

Donde,

No conocemos el valor de porque depende del vector desconocido , pero podemos tomar temporalmente .

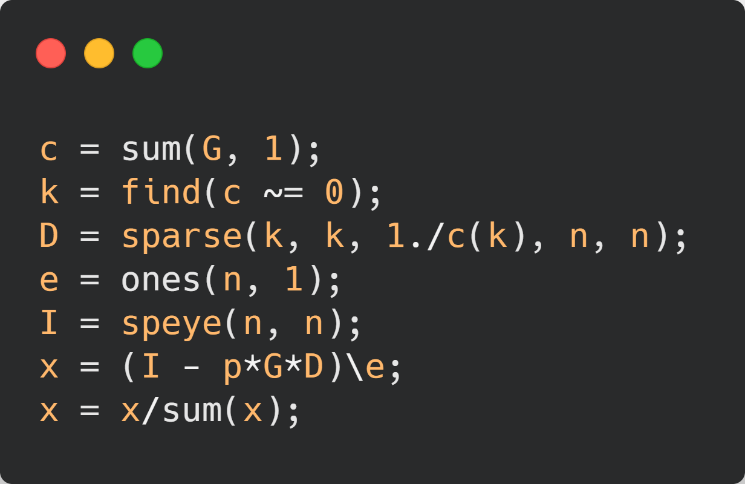
Mientras sea estrictamente menor que uno, la matriz de coeficientes es no singular y la ecuación,

Puede resolverse para . Entonces, la resultante puede reescalarse de forma que,

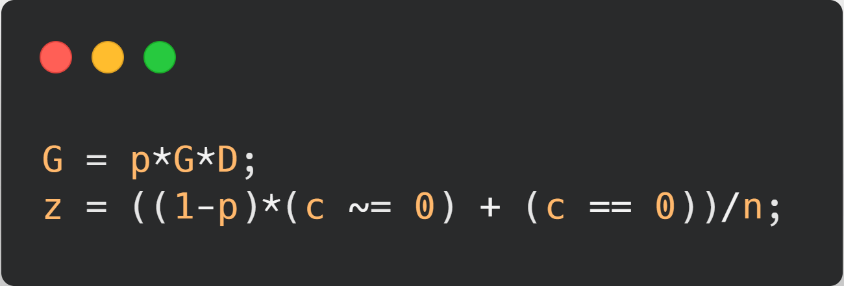
Obsérvese que el vector no interviene realmente en este cálculo.

***Algoritmo PageRank:* Análisis Práctico (Métodos alternativos)**

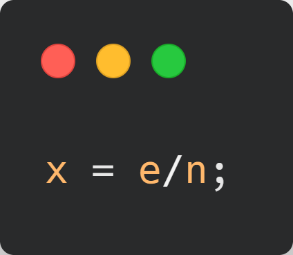
Lo último mencionado en la sección anterior puede ser escrito en código Matlab/Octave de la siguiente manera,



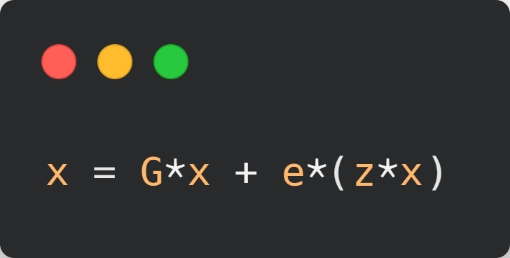
El *método de la potencia* también puede implementarse de forma que no se produzca la matriz de Markov y así preservar la dispersión. Para ello calculamos,



Iniciamos con,

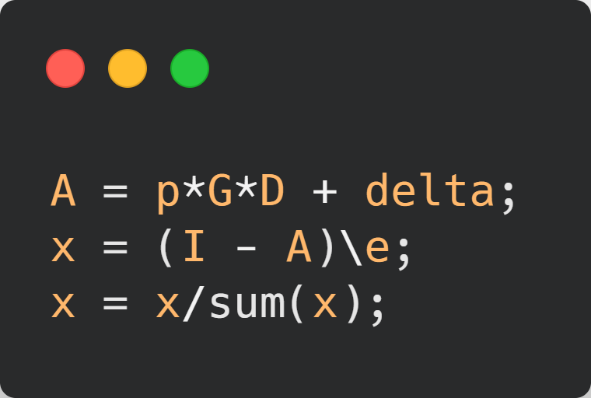


Entonces, se repite la siguiente sentencia,



Hasta que se establezca con varios decimales.

También es posible utilizar un algoritmo conocido como ***iteración inversa***.



A primera vista, esto parece una idea muy mala. Como es teóricamente singular, con el cálculo exacto algún elemento diagonal del factor triangular superior de debería ser cero y este cálculo debería fallar. Pero con el error de redondeo, la matriz calculada probablemente no es exactamente singular. Incluso si es singular, el redondeo durante la eliminación gaussiana probablemente impedirá que haya elementos diagonales exactamente nulos.

Sabemos que la eliminación gaussiana con pivoteo parcial siempre produce una solución con un residuo pequeño, en relación con la solución calculada, incluso si la matriz está mal condicionada. El vector que se obtiene con la operación de barra invertida, suele tener componentes muy grandes. Si se reescala por su suma, el residuo se escala por el mismo factor y se vuelve muy pequeño. En consecuencia, los dos vectores y son iguales entre sí con un error de redondeo. En esta situación, la resolución del sistema singular con la eliminación gaussiana explota, pero lo hace exactamente en la dirección correcta.

***Algoritmo PageRank:* Código Fuente**

***Algoritmo PageRank:* Caso de Prueba**