Métodos Numéricos
1er Cuatrimestre 2019
Trabajo Práctico 1

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Sistemas de ecuaciones lineales

Introducción

El motor del buscador de Google, desde hace más de 20 años, utiliza el *ranking de Page* o *PageRank* [1] como uno de los criterios para ponderar la importancia de los resultados de cada búsqueda. Calcular este ranking requiere simplemente resolver un sistema de ecuaciones lineales donde la cantidad de ecuaciones e incógnitas del sistema es igual al número de páginas consideradas... ¿Simplemente?

Modelado del problema

Dado un determinado conjunto de n páginas web definimos la *matriz de conectividad* $\mathbf{W}$ poniendo $w_{ij} = 1$ si la página j tiene un link a la página i y $w_{ij} = 0$ si no. Además $w_{ii} = 0$ pues ignoramos los *autolinks*. De esta forma, la matriz $\mathbf{W}$ puede resultar extremadamente rala y muy grande de acuerdo al tamaño del conjunto de páginas.

Para cada página $j = 1 \dots n$, definimos su *grado* como:

$$c_j = \sum_{i=1}^{n} w_{ij} \tag{1}$$

Es decir, la cantidad de links *salientes* de j. Típicamente c_j es un número mucho menor que n.

Se busca que el ranking sea mayor en las páginas *importantes*. Heurísticamente, una página es importante cuando recibe muchos "votos" de otras páginas, es decir, links. Pero no todos los links pesan igual: los links de páginas más importantes valen más. Pocos links de páginas importantes pueden valer más que muchos links de páginas poco importantes. Y los links de páginas con muchos links valen poco. Por lo tanto, una forma de calcular la importancia o *puntaje* x_i de la página i es:

$$x_i = \sum_{j=1}^{n} \frac{x_j}{c_j} w_{ij} \tag{2}$$

Es decir, la página j le aporta a i su puntaje ponderado por cuántos links salientes tiene. También, se puede definir la matriz de puntajes $\mathbf{M} = \mathbf{WD}$, donde $\mathbf{D}$ es una matriz diagonal con elementos d_{jj} de la forma:

$$d_{jj} = \begin{cases} 1/c_j & \text{si} \quad c_j \neq 0 \\ 0 & \text{si} \quad c_j = 0 \end{cases},$$

[1] Denominado así por Larry Page, uno de los fundadores de Google, otrora joven científico actualmente devenido en multimillonario. Ver artículo original del 1998 (citado más de 17800 veces) en [1]

Lo cual nos permite calcular el ranking de todas las páginas como:

$$\mathbf{M}\ \mathbf{x} = \mathbf{x} \tag{3}$$

donde $\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_n)^T$. Luego, la ecuación 2 corresponde al elemento i-ésimo $(\mathbf{M}\ \mathbf{x})_i$.

Este modelo tiene un problema y es que no logra capturar el comportamiento errático del usuario mientras surfea la red redes.

Modelo del navegante aleatorio

Un enfoque alternativo es considerar el modelo del *navegante aleatorio*. El navegante aleatorio empieza en una página cualquiera del conjunto, y luego en cada página j que visita elige con probabilidad $p \in (0, 1)$ si va a seguir uno de sus links, o con probabilidad $1 - p$, si va a pasar a otra página cualquiera del conjunto. Una vez tomada esa decisión, si decidió seguir un link de la página j elige uno al azar con probabilidad $1/c_j$, mientras que si decidió pasar a otra página cualquiera entonces elige una al azar con probabilidad $1/n$). Cuando la página j no tiene links salientes, es decir $c_j = 0$, elige al azar una página cualquiera del conjunto. Por lo tanto, se espera que luego de mucho surfear el navegante aleatorio va a estar en páginas importantes con mayor probabilidad.

Formalmente, la probabilidad de pasar de la página j a la página i es:

$$a_{ij} = \begin{cases} (1-p)/n + (p\ w_{ij})/c_j & \text{si} \quad c_j \neq 0 \\ 1/n & \text{si} \quad c_j = 0 \end{cases}, \tag{4}$$

y sea $\mathbf{A} \in \mathbb{R}^{n \times n}$ a la matriz de elementos a_{ij}. Entonces el *ranking de Page* es la solución del sistema:

$$\mathbf{A}\ \mathbf{x} = \mathbf{x} \tag{5}$$

que cumple $x_i \geq 0$ y $\sum_i x_i = 1$.

Por lo tanto, el elemento i-ésimo $(\mathbf{A}\mathbf{x})_i$ es la probabilidad de encontrar al navegante aleatorio en la página i sabiendo que x_j es la probabilidad de encontrarlo en la página j, para $j = 1 \ldots n$.

Luego, la matriz $\mathbf{A}$ puede reescribirse como:

$$\mathbf{A} = p\ \mathbf{W}\ \mathbf{D} + \mathbf{e}\ \mathbf{z}^T,$$

donde $\mathbf{D}$ es una matriz diagonal de la forma

$$d_{jj} = \begin{cases} 1/c_j & \text{si} \quad c_j \neq 0 \\ 0 & \text{si} \quad c_j = 0 \end{cases},$$

$\mathbf{e}$ es un vector columna de unos de dimensión n y $\mathbf{z}$ es un vector columna cuyos componentes son:

$$z_j = \begin{cases} (1-p)/n & \text{si} \quad c_j \neq 0 \\ 1/n & \text{si} \quad c_j = 0 \end{cases}.$$

Así, la ecuación (5) puede reescribirse como

$$\mathbf{M}\,\mathbf{x} = \gamma\,\mathbf{e} \tag{6}$$

donde la matriz $\mathbf{M} = \mathbf{I} - p\ \mathbf{W}\ \mathbf{D}$ y la constante $\gamma = \mathbf{z}^T\ \mathbf{x}$ funciona como un factor de escala. Se puede demostrar[2] que resolver este sistema lineal mediante el siguiente procedimiento brinda un método eficaz para calcular el ranking de Page.

Cálculo del ranking de Page

El procedimiento para calcular el ranking de Page consiste en:

1. Suponer $\gamma = 1$.
2. Resolver el sistema lineal: $\mathbf{M}\,\mathbf{x} = \gamma\,\mathbf{e}$.
3. Normalizar el vector $\mathbf{x}$ de manera que $\sum_i x_i = 1$.

Enunciado

Se **debe** implementar un programa en `C` o `C++` que realice el cálculo del ranking de Page según el procedimiento descripto anteriormente.

Como parte **obligatoria** se pide implementar lo siguiente:

1. El método de Eliminación Gaussiana (EG) *sin* permutaciones.
2. Una estructura de matrices ralas que sea eficiente en espacio y en tiempo para la tarea que se busca realizar.
3. Herramientas de entrada/salida para la lectura de archivos con los datos de los conjuntos de páginas y escritura del ranking de Page. Se debe respetar el formato que se describe más abajo.

Previamente, **se deberá** estudiar las características de la matriz involucrada y responder a lo siguiente:

1. ¿Por qué la matriz $\mathbf{A}$ definida en (4) es equivalente a $p\ \mathbf{W}\ \mathbf{D} + \mathbf{e}\ \mathbf{z}^T$? Justificar adecuadamente.
2. ¿Cómo se garantiza la aplicabilidad de EG sin permutaciones de filas o columnas? ¿Qué tipo de matriz resulta $\mathbf{M}$?
3. Calcular el número de condición de $\mathbf{M}$ con norma 1 ¿Cómo influye el valor de p?
   Para este punto se *sugiere* consultar el material de la cátedra y la bibliografía, en particular [3].

[2] Esto se puede ver en el artículo [5] pero no es el objetivo de este trabajo práctico.

Además, **se pide** realizar un informe utilizando como guía las pautas de laboratorio de la materia conteniendo la experimentación pedida en la siguiente sección. Es importante incluir en la sección desarrollo del informe del trabajo práctico, las alternativas consideradas y descartadas para cada uno de los métodos utilizados.

Se **recomienda** fuertemente en todos los casos comparar los resultados intermedios utilizando python, Matlab/Octave o R.

Casos de test

La cátedra proveerá un conjunto de tests con archivos de entrada y salida esperada. Para aprobar el trabajo, los mismos **deberán** funcionar correctamente en sus implementaciones con una tolerancia máxima de 10^{-4} medida en error absoluto con respecto a los valores de ranking proporcionados.

Además, cada test **deberá** correr en las computadoras del laboratorio en un tiempo de similar en orden de magnitud al indicado.

Experimentación

Se deberá realizar tanto un análisis **cualitativo** como **cuantitativo** de los métodos vistos en el trabajo práctico.

Análisis cuantitativo

Se **pide**, analizar y reportar los errores obtenidos en el ranking de Page utilizando:

1. La aproximación de la solución x' obtenida: $|Ax' - x'| \approx 0$
2. El error absoluto con respecto a los test de la cátedra.

Análisis cualitativo

Se **deberán** estudiar los rankings obtenidos, en función de la estructura del grafo, y del valor de p, *relacionándolo* con el modelado del problema.

Para esto se deberán estudiar al menos los grafos provistos en los casos de tests de la cátedra. Además, cada grupo puede proponer instancias de prueba que crean pertinentes para analizar el método (entregando además los archivos correspondientes).

Para guiar el análisis, responder las siguientes preguntas:

1. ¿Cómo es el ranking obtenido en cada caso de acuerdo a la estructura del grafo páginas?
2. ¿Qué conclusiones pueden sacar de la interpretación de los resultados?

3. Respecto del ranking de Page: ¿Funciona cómo era esperado? ¿Hubo sorpresas? ¿Qué pueden concluir sobre su significado? ¿Dirían que es un buen ranking?

4. ¿Cómo influye el valor de p en la calidad de los rankings?

Programa ejecutable y entrada/salida

Los archivos de entrada y salida serán de texto plano, y deberán respetar el siguiente formato:

archivo entrada: En la primera línea un entero N, la cantidad total de páginas. En la segunda línea un entero M, la cantidad total de links. Luego siguen M líneas, cada una con dos enteros i j separados por un espacio ($1 \leq i, j \leq N$), indicando que hay un link de la página i a la página j.

archivo de salida: La primera línea deberá contener el valor de p utilizado. Luego, deberán seguir N líneas, donde la línea i contiene el valor del ranking de Page para la página i.

En el Cuadro 1 se pueden ver las primeras 8 líneas de un archivo de entrada que contiene una instancia con 30 páginas y 178 links entre ellas, y su salida obtenida como resultado de correr el programa con $p = 0{,}8$.

30	0.8
178	0.00042152
1 3	0.00026892
1 2	0.000230783
1 19	0.000329963
2 26	0.000117996
3 12	0.000359948
3 4	0.000367727
...	...

Cuadro 1: Izquierda: Ejemplo de un archivo de entrada. Derecha: archivo de salida.

El programa ejecutable **deberá** cumplir con el siguiente formato de uso:

```
./tp1 archivo p
```

Donde `archivo` es la ubicación del archivo de entrada a leer, y `p` es el valor de p a usar. El archivo de salida deberá ser escrito en la ubicación `archivo.out`.

Además, se **deberá** entregar un archivo `README` conteniendo la instrucciones de compilación y ejecución, y también ejemplos de invocación del programa.

Fechas de entrega

- *Formato Electrónico:* domingo 7 de abril hasta las 23.59 hs, enviando el trabajo (informe + código) a la dirección `metnum.lab@gmail.com`.
  - El subject del email debe comenzar con el texto `[TP1]` seguido de la lista de apellidos de los integrantes del grupo separados por punto y coma `;`. Ejemplo: `[TP1] Lennon; McCartney; Starr; Harrison`
  - Se ruega no sobrepasar el máximo permitido de archivos adjuntos de 20MB. Tener en cuenta al realizar la entrega de no ajuntar bases de datos disponibles en la web, resultados duplicados o archivos de backup.
- *Recuperatorio:* jueves 2 de mayo hasta las 23.59 hs, enviando el trabajo corregido a `metnum.lab@gmail.com`.

Importante: El horario es estricto. Los correos recibidos después de la hora indicada serán considerados re-entrega.

Referencias

[1] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[2] Kurt Bryan and Tanya Leise. The $25,000,000,000 eigenvector: The linear algebra behind google. *SIAM review*, 48(3):569–581, 2006.

[3] Richard L Burden, J Douglas Faires, and Albert C Reynolds. Análisis numérico, 2002.

[4] Sepandar D Kamvar, Taher H Haveliwala, Christopher D Manning, and Gene H Golub. Extrapolation methods for accelerating pagerank computations. In *Proceedings of the 12th international conference on World Wide Web*, pages 261–270. ACM, 2003.

[5] Amy N Langville and Carl D Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2004.