TP1: PageRank

Métodos Numéricos

Departamento de Computación, FCEyN,Universidad de Buenos Aires.

Contexto - Motores de búsqueda

¿ Qué debe hacer un motor de búsqueda?¿Qué tareas debe poder resolver?

- Explorar la red e identificar todas las páginas con acceso público.
- Almacenar la información obtenida, para realizar búsquedas eficientemente.
- Determinar un **orden** de las páginas según su **importancia**, para presentar la información con un **orden de relevancia**.

En el TP nos centraremos en este último Ãtem.

Cómo determinar un orden de importancia

- ¿Qué significa que una página sea importante?

  "Automated search engines that rely on keyword matching usually return too many low quality matches."

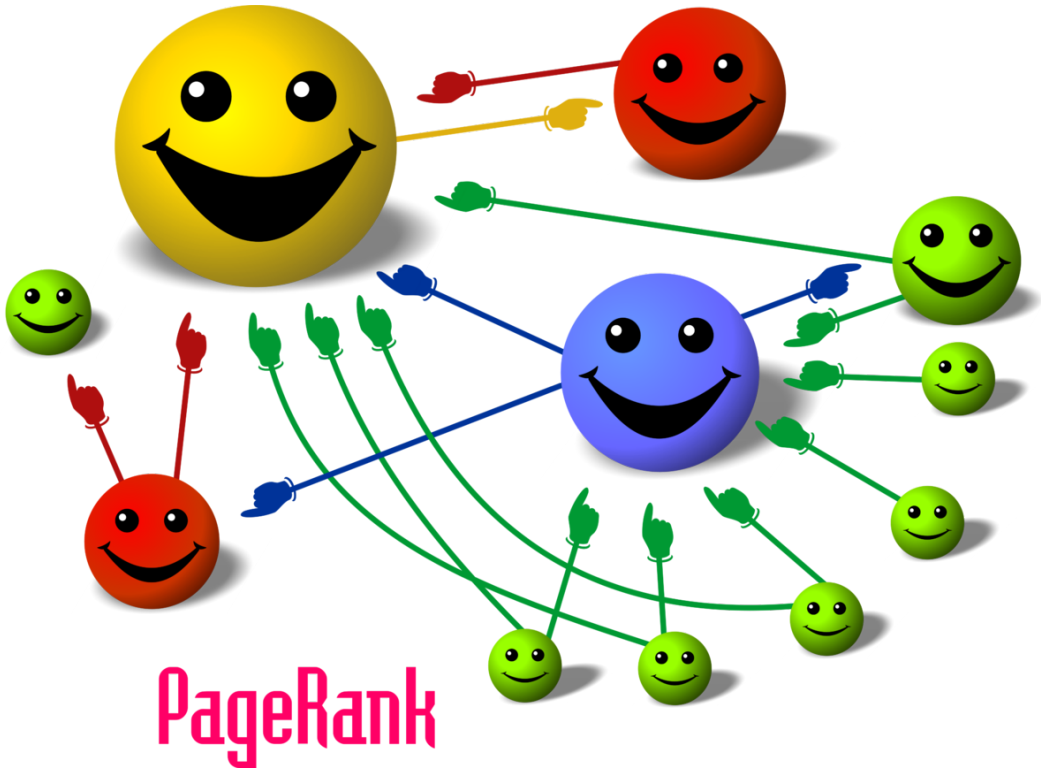
- ¿Cómo podemos utilizar la estructura de la web para ayudarnos?

  The citation (link) graph of the Web is an important resource that has largely gone unused in existing Web search engines

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.

Usando el grafo de la web

- Vamos a usar los links de la web para ayudarnos
- ¿Importará la cantidad de links?¿Cómo? Â¿Todos los links *valen* los mismo?
- Contaremos la **cantidad** y **calidad** de los links que apuntan a una determinada página.



Pagerank - El modelo

- Sea el conjunto de páginas web: $\{1, \ldots, n\}$.
- Definimos la **matriz de conectividad** $W \in \{0,1\}^{n \times n}$ como:

$$W_{ij} = \begin{cases} 1 & \text{si la página } j \text{ tiene un link a la página } i \\ 0 & \text{si no} \end{cases}$$

- Además, ignoramos los autolinks, $w_{ii} = 0 \ \forall i \in W$
- Informalmente: "En la fila i están las que apuntan a i y en la columna j están las webs apuntadas por j."
- Para una página $j \in W$, definimos su **grado** como:

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

Es decir, la cantidad de links que *salientes* de j.

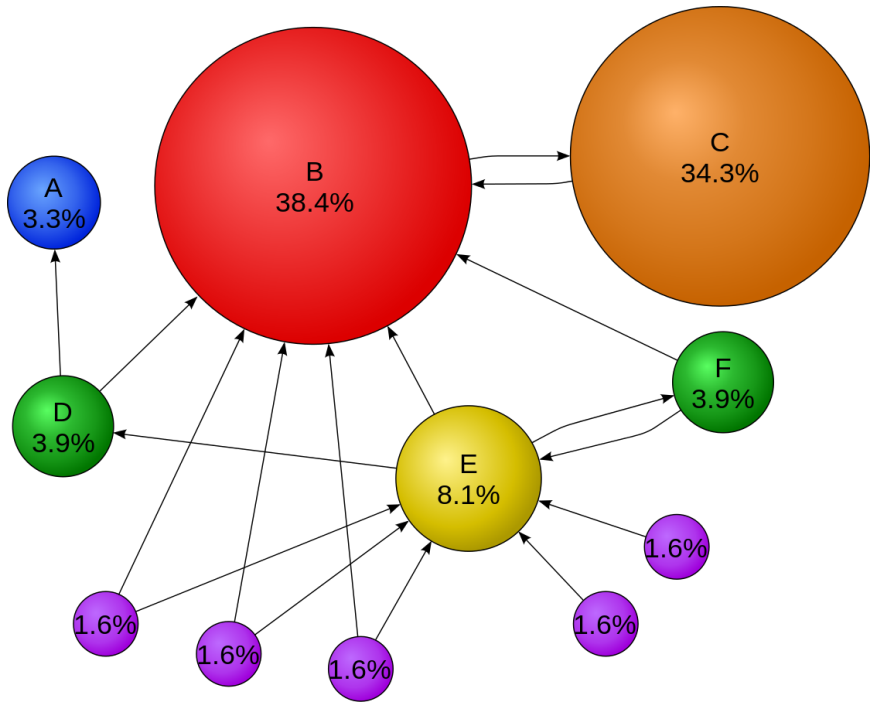
¿Cómo definimos puntajes?

- Llamamos x_i al **puntaje** asignado a la página i.
- Dadas $i, j \in \{1 \dots n\}$, el **aporte** del link $j \longrightarrow i$ a la página i como:

$$\frac{x_j}{c_j} w_{ij}$$

.

- Es decir, j le aporta a i su puntaje ponderado por cuántos links salientes tiene. Si no hay link o $c_j = 0$ le aporta cero.



¿Cómo calculamos puntajes?

Definimos el puntaje de la página i como:

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

El puntaje de una página depende del puntaje de otras.
Luego, se define la matriz de puntajes $\mathbf{R} = \mathbf{WD}$, donde $\mathbf{D}$ es una matriz diagonal de la forma:

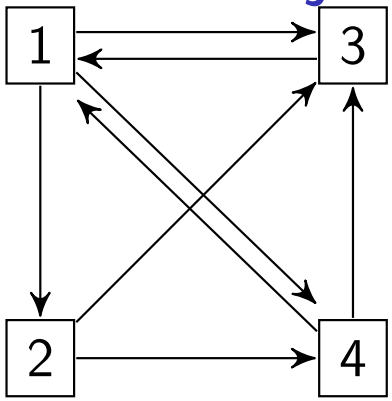
$$d_{jj} = \begin{cases} 1/c_j & \text{si} \quad c_j \neq 0 \\ 0 & \text{si} \quad c_j = 0 \end{cases},$$

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

$$\boxed{\mathbf{R}\ \mathbf{x} = \mathbf{x}} \quad \text{donde} \quad \mathbf{x} = (x_1, \ldots, x_i, \ldots, x_n)^T$$

Finalmente, tomaremos como solución al vector $\mathbf{x}$ normalizado para que sume 1.

Veamos un ejemplo



$$\mathbf{R} = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$c_1 = 3,\ c_2 = 2,\ c_3 = 1,\ c_4 = 2$$

Si $\mathbf{x}$ es el vector con los puntajes de las páginas:

$$(\mathbf{Rx})_i = fila_i(\mathbf{R})\mathbf{x} = \sum_{j=1}^{n} \frac{x_j}{c_j} w_{ij}$$

$$(\mathbf{Rx})_i = x_i \quad \forall i$$

Luego, la solución al sistema es $\mathbf{x} = [\frac{12}{31}, \frac{4}{31}, \frac{9}{31}, \frac{6}{31}]^T$

Para pensar...

- La página 3 es apuntada por todas las demás pero sin embargo no tiene el mayor puntaje. ¿Por qué?
- ¿Listo? ¿Qué pasa si una página j no tiene links salientes ?

Intuición: Navegante aleatorio

- El modelo anterior tiene un problema y es que no logra capturar el comportamiento errático del usuario mientras surfea la red redes.
- Un enfoque alternativo es considerar el modelo del *navegante aleatorio*.
- Pagerank se basa en la idea de un **navegante aleatorio**. Este inicia en una página cualquiera y *navega* por la web mediante los links, moviéndose de página en página con cierta probabilidad.

Navegante aleatorio

Recorrido aleatorio de páginas

- En cada página j que visita el navegante elige:
  1. con probabilidad $p \in (0,1)$ si va a seguir uno de sus links,
  2. con probabilidad $1-p$, si va a pasar a otra página cualquiera.
- Caso 1: Si decidió seguir un link de la página j elige uno al azar con probabilidad $1/c_j$,
- Caso 2: si decidió pasar a otra página cualquiera entonces elige una al azar con probabilidad $1/n$.
- Cuando la página j no tiene links salientes ($c_j = 0$), elige al azar una página cualquiera del conjunto.

Navegante aleatorio

- Definimos la matriz $\mathbf{A}$ donde la posición a_{ij} representa la probabilidad de pasar a la página i estando en la página j.
- Considerando el recorrido aleatorio, definimos las probabilidades a_{ij} de la matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$ como:

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

- Finalmente, el *PageRank* es la solución al sistema:

$$\boxed{\mathbf{Ax} = \mathbf{x}}$$

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

- En el TP, vamos a hallar $\mathbf{x}$ como la solución a un **sistema lineal** equivalente.

Rescritura matricial de la matriz $\mathbf{A}$

Definiciones:

- $\mathbf{W} \in \mathbb{R}^{n \times n}$, matriz de conectividad
- $\mathbf{D} \in \mathbb{R}^{n \times n}$, matriz diagonal tal que

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

- $\mathbf{z} \in \mathbb{R}^n$, vector definido como

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

- $\mathbf{e} = (1, \ldots, 1)^T$ vector de n unos.
- $p \in (0, 1)$ representa la probabilidad de que el navegante siga en uno de los links de la página actual.

Matriz $\mathbf{A}$ del recorrido aleatorio

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

Solución mediante sistemas lineales

El problema de PageRank se puede reescribir como:

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

donde la matriz $\mathbf{M} = \mathbf{I} - p\,\mathbf{W}\,\mathbf{D}$ y el valor $\gamma = \mathbf{z}^T\,\mathbf{x}$ funciona como un factor de escala.

Procedimiento para calcular el PageRank

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

Tareas de Implementación

1. Leer el enunciado
2. Implementar Eliminación Gaussiana (EG) *sin* permutaciones.
3. Implementar una estructura de matrices ralas
4. Pasar los test de la cátedra
5. Leer el enunciado

Experimentación e informe

1. Leer el enunciado
2. Análisis **cuantitativo**
   - 2.1 La aproximación de la solución x' obtenida: $|Ax' - x'| \approx 0$
   - 2.2 El error absoluto con respecto a los test de la cátedra.
3. Análisis **cualitativo**.
   - 3.1 Rankings en función de la estructura del grafo
   - 3.2 Influencia del valor de p
   - 3.3 Relacionar con el modelado del problema
4. Leer el enunciado

Sugerencia de avance para el TP

Vencimiento de tareas por fecha:

- 26 de Agosto:
  - Leer enunciado y pautas de laboratorio. Descargar material del TP.
  - Código: operaciones matriciales + I/O de datos de páginas
- 2 de Septiembre:
  - Código: eliminación gaussiana + PageRank
  - Informe: secciones introducción y desarrollo
- 9 de Septiembre:
  - Informe: sección resultados y discusión
- 11 de Septiembre (domingo): **entrega** por mail.
  - Informe: corrección ortográfica y gramatical
  - Código: README, comentarios y .zip para la entrega

Atención

- Utilizar los casos de test provistos por la cátedra para verificar los cálculos.
- Recuperatorio: domingo 9 de octubre hasta las 23.59 hs

Referencias

- Sergey Brin and Lawrence Page, The anatomy of a large-scale hypertextual Web search engine, *Computer Networks and ISDN Systems*, April 1998.
- Kurt Bryan and Tanya Leise, The linear algebra behind google. *SIAM Review*, 2006.
- 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*, New York, NY, USA, 2003. ACM.