

Métodos de cálculo del vector PageRank

F. PEDROCHE*

Institut de Matemàtica Multidisciplinar
Universitat Politècnica de València.

pedroche@imm.upv.es

Resumen

Los algoritmos de búsqueda de información en internet, como el *PageRank* de Google, constituyen un ejemplo excelente de aplicación de las herramientas básicas del análisis matricial, las cadenas de Markov y el álgebra lineal numérica. En este trabajo se ilustran los métodos de solución que se usan para el cálculo del vector PageRank: el método de la potencia y sus derivados (extrapolación, adaptativo, Arnoldi y BlockRank), y los basados en la resolución de un sistema lineal por métodos iterativos (Jacobi, Gauss-Seidel, Schwarz aditivo, métodos de subespacios de Krylov).

Palabras clave : *Motores de búsqueda, PageRank, procesos de Markov, matrices no negativas, sistemas lineales, métodos iterativos, cálculo científico.*

Clasificación por materias AMS : *15A06, 65C40, 65F10, 65F15, 65F50*

1. Introducción

En mayo de 2005, una consulta en internet usando el motor de búsqueda Google¹ informaba de que se estaba realizando la petición sobre un total de 8,085 millones de páginas. Otra búsqueda, esta vez realizada el 26 de octubre de 2006, permite estimar que, al menos, hay unos 24,640 millones de páginas web indexadas² por Google. Estos datos dan una idea del tamaño y la velocidad de crecimiento de internet. Si a esto unimos las posibilidades de negocio

*Trabajo subvencionado por los proyectos DGI número MTM2004-02998 y DGI número MTM2007-64477.

¹Una idea de la popularidad de este tipo de búsquedas es que el verbo *to google* fue añadido oficialmente al *Oxford English Dictionary*, en junio de 2006, con el significado de *usar el buscador Google para obtener información en internet*.

²Estimación hecha a partir de la petición "+el**" OR "+the**" OR "+le**" OR "+der**" OR "+O**" OR "+il**". El signo más obliga a buscar el artículo *el, the*, etc. y los asteriscos son dos palabras cualesquiera; v. [53] para más trucos de Google.

mediante anuncios publicitarios³ que el mismo Google promueve [79], tenemos que es fundamental disponer de un sistema de clasificación de páginas rápido y fiable, para poner orden en toda esta maraña de datos que ha ido creciendo vertiginosamente; para un análisis sobre la estructura de la World Wide Web, véanse, por ejemplo, [22], [44], [55], [59], [66].

El *PageRank* [57], el método inicial de cálculo que usaron los fundadores de Google⁴ para clasificar las páginas web según su importancia, es objeto de constantes mejoras. La finalidad del método es la obtención de un vector, también llamado PageRank, que da la importancia relativa de las páginas⁵. Dado que el vector PageRank se calcula en función de la estructura de las conexiones de la web (v. [24] en un número anterior del Boletín de SEMA) se dice que es independiente de la petición de la persona que realiza la búsqueda. Algunas modificaciones del PageRank para hacer intervenir la petición se han propuesto en [33], [36], [60]. También hay modificaciones del PageRank que incorporan la utilización del botón de página anterior [70].

Desde el punto de vista de la persona que administra una página web se ha mostrado que la técnica para obtener un PageRank óptimo consiste en incluir enlaces a páginas importantes de nuestra misma comunidad web, mientras que los enlaces irrelevantes nos penalizan a nosotros y a toda nuestra comunidad [3]. Un análisis del algoritmo PageRank así como diversas propiedades se puede encontrar, por ejemplo, en [8], [10] [18], [23], [24], [37], [38], [47], [51], [71].

En este trabajo nos centraremos en los nuevos métodos de cálculo del vector PageRank haciendo especial hincapié en aquellos conceptos del álgebra⁶ lineal que intervienen en los modelos. Estos modelos se basan, por una parte, en desarrollar técnicas para acelerar el método clásico de la potencia (que había caído en el abandono y prácticamente casi ni se usaba) y, por otra, en usar una formulación con un sistema de ecuaciones lineales y aplicar entonces un esquema iterativo. El cálculo práctico, atendiendo a las estructuras de computación utilizadas, puede ser centralizado (utilizando un procesador) o en paralelo (usando múltiples procesadores).

El resto del trabajo se estructura de la manera siguiente. En la sección 2 se introduce el concepto de vector clasificador de páginas PageRank y en la sección 3 se revisa el modelo de Brin y Page para su cálculo. En la sección 4 se resumen los métodos de solución basados en el método de la potencia. En la sección 5 se resumen las características de los métodos basados en la escritura de un sistema de ecuaciones lineales. Finalmente, se dan unas conclusiones del trabajo.

³Los beneficios por publicidad en internet, en EEUU, han crecido un 36% en el primer semestre de 2006 [78].

⁴Véase [76] para conocer detalles sobre los inicios de la compañía Google Inc.

⁵Las personas registradas en Google pueden instalarse el indicador de PageRank (PR) incluido en la *Google Toolbar*. Con esto se puede conocer el PR, en una escala de 0 a 10, de las páginas que visitemos. Por ejemplo, un diario estatal tiene un PR en torno a 8, mientras que la página de una universidad tiene un PR alrededor de 5.

⁶El uso actual de la palabra *álgebra* proviene del título del libro *Kitab al-jabr wa l-muqabala*, (Libro de la reducción y la comparación) escrito en el siglo IX por Mohammed Ibn Musa al-Khwarizmi [65], [80]; las palabras *guarismo* y *algoritmo* derivan de su nombre [4].

2. El vector PageRank de Google

Una de las características del PageRank es que si uno navega aleatoriamente por internet y está un tiempo suficientemente grande paseando, entonces tendrá una gran probabilidad de encontrar las páginas con mayor PageRank. Para centrar ideas, consideremos un conjunto de cuatro páginas web como en la figura 1. Vemos que desde la página 1 podemos tomar un enlace a las páginas 2 ó 3. También se ve que desde las páginas 2 ó 4 se pueden acceder al resto de páginas.

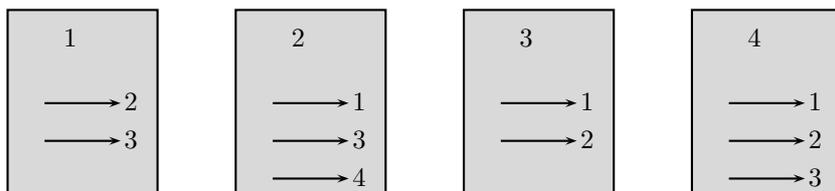


Figura 1: Una web con cuatro páginas mostrando sus enlaces salientes.

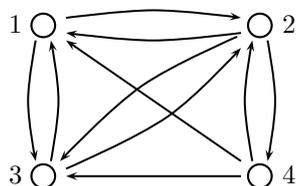


Figura 2: Grafo dirigido correspondiente a la web de la figura 1.

Esta estructura de enlaces entre las páginas se puede representar mediante un grafo dirigido, como el de la figura 2. Dado un conjunto de n páginas web definimos su matriz de conectividad G como la matriz cuadrada de orden n cuyos elementos, denominados g_{ij} , $1 \leq i, j \leq n$, valen 1 si hay enlace de la página j a la página i , con $i \neq j$, y 0 en otro caso. La matriz de conectividad correspondiente al grafo de la figura 2 viene dada entonces por:

$$G = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \quad (1)$$

Imaginemos que tenemos una persona (un o una surfista) que va saltando de manera aleatoria de unas páginas a otras. Queremos construir una matriz de transición P , cuyos elementos den las probabilidades condicionadas de salto. Para ello, vamos a asumir que, cuando se encuentra en una página, *tiene la misma probabilidad de elegir cualquier enlace saliente*. Esta elección es la base del modelo de Brin y Page. En nuestro ejemplo (figura 1), si el surfista está en la página 2 entonces tiene una probabilidad de $1/3$ de ir a cualquiera de las

páginas 1, 3 ó 4. Aplicando este razonamiento a las cuatro páginas obtenemos que la matriz de transición P del surfista aleatorio es en este caso:

$$P = \begin{bmatrix} 0 & 1/3 & 1/2 & 1/3 \\ 1/2 & 0 & 1/2 & 1/3 \\ 1/2 & 1/3 & 0 & 1/3 \\ 0 & 1/3 & 0 & 0 \end{bmatrix}. \quad (2)$$

Desde el punto de vista del cálculo es muy fácil obtener la matriz P a partir de la matriz G : basta dividir cada columna de G por la suma de los elementos de G en dicha columna, siempre que esta cantidad no sea cero, es decir, siempre que esta columna no corresponda a una página sin salida. Para formalizar este hecho, definamos el número de enlaces salientes (*out-degree*) de una página j como: $c_j = \sum_{i=1}^n g_{ij}$, $1 \leq j \leq n$. Ahora ya podemos construir la matriz P asociada a la estructura del conjunto de páginas web de acuerdo con nuestra hipótesis de probabilidad de saltos. Ha de ser $P = (p_{ij}) \in \mathbb{R}^{n \times n}$, tal que:

$$p_{ij} = \begin{cases} g_{ij}/c_j & \text{si } c_j \neq 0. \\ 0 & \text{en otro caso.} \end{cases} \quad 1 \leq i, j \leq n. \quad (3)$$

Es obvio que si $c_j \neq 0$, para todo j , entonces P es una matriz *estocástica por columnas*, es decir, la suma de cada columna vale 1 y cada elemento toma un valor entre 0 y 1. Como ejemplo, la matriz P dada por la ecuación (2) es estocástica. Además, esta matriz tiene otras propiedades matemáticas interesantes: su espectro (conjunto de valores propios) es: $\sigma(P) = \{1, -1/2, -1/3, -1/6\}$. Su radio espectral⁷ es $\rho(P) = 1$, y la matriz P es irreducible⁸ y primitiva⁹. Estas propiedades son importantes ya que el vector PageRank [57] es el vector límite de distribución de probabilidad de una *cadena de Markov ergódica*: el vector de estado límite¹⁰ existe, coincide con el *vector de estado estacionario* (un vector de probabilidad asociado al valor propio 1) y es independiente del vector de estado inicial. Esta propiedad del vector de estado límite se verifica, en particular, para matrices estocásticas y primitivas: $P \geq 0$, P irreducible y P sólo tiene un valor propio ($\lambda = 1$) de módulo el radio espectral $\rho(P) = 1$.

En este ejemplo, el vector de estado estacionario (que coincide con el de estado límite) resulta: $v_{est}^T = [2/7 \ 9/28 \ 2/7 \ 3/28] \approx [0,29 \ 0,32 \ 0,29 \ 0,10]$, y es el vector PageRank de las cuatro páginas. La interpretación es la siguiente: si un surfista aleatorio se mueve por un conjunto de cuatro páginas enlazadas como en la figura 2, entonces, para un tiempo

⁷Se define el *radio espectral* de una matriz cuadrada como el valor máximo, en valor absoluto, de sus valores propios.

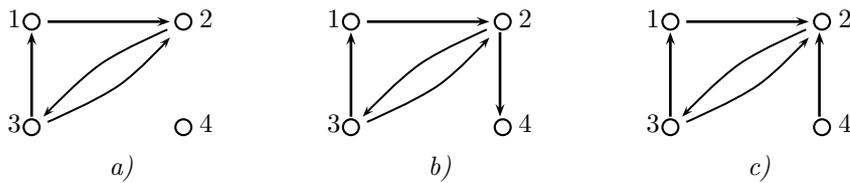
⁸Una matriz A es *irreducible*, si, y solamente si, su grafo es *fuertemente conexo*: para cada pareja de nodos (i, j) con $i \neq j$, se puede ir de i a j a través de una sucesión de arcos orientados. En otro caso, se dice que es *reducible*.

⁹Una matriz cuadrada A se dice que es *primitiva* si es irreducible, no negativa, y con $\rho(A)$ estrictamente mayor que cualquier otro valor propio.

¹⁰Dado un vector de probabilidad v_0 y una matriz estocástica P , el *vector de estado límite* es, si existe, $v_\infty = \lim_{k \rightarrow \infty} P^k v_0$.

suficientemente prolongado, lo más probable es encontrarlo en la página 2 (con una probabilidad de 0,32), mientras que la probabilidad de encontrarlo en las páginas 1 ó 3 es de 0,29. Dicho de otra forma, el surfista emplea un 32% de su tiempo visitando la página 2. Nótese que las páginas 1, 2 y 3 son citadas por el mismo número de páginas. Lo único que las diferencia es que la página 2 tiene un enlace a la página 4 que revierte posteriormente en un aumento de la probabilidad de la página 2. En este ejemplo la matriz P es primitiva, pero hay casos en los que la estructura de enlaces entre las páginas no conduce a una matriz estocástica y primitiva, como vemos en el siguiente ejemplo. En la sección 3 analizamos cómo conseguir que la matriz P sea estocástica y primitiva.

Ejemplo 1 Los grafos dirigidos siguientes:



tienen asociadas las matrices P , según la ecuación (3):

$$a) \begin{bmatrix} 0 & 0 & 0,5 & 0 \\ 1 & 0 & 0,5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad b) \begin{bmatrix} 0 & 0 & 0,5 & 0 \\ 1 & 0 & 0,5 & 0 \\ 0 & 0,5 & 0 & 0 \\ 0 & 0,5 & 0 & 0 \end{bmatrix}, \quad c) \begin{bmatrix} 0 & 0 & 0,5 & 0 \\ 1 & 0 & 0,5 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

La situación a) no la vamos a permitir, ya que si una página no tiene enlaces entrantes ni salientes no pertenece al conjunto de n páginas en el cual definimos las matrices G y P . La situación b) ilustra el concepto de nodo sin salida (dangling node): el nodo correspondiente a la página 4 no tiene enlaces salientes y por ello $c_4 = 0$ y la matriz P asociada no es estocástica. Solo la matriz asociada al grafo c) es una matriz estocástica. Las tres matrices son reducibles, ya que no todos los nodos son accesibles cuando se parte de un nodo arbitrario. Por tanto, ninguna de ellas es primitiva y no nos sirven para el modelo del surfista aleatorio tal como lo tenemos definido hasta ahora.

3. Modelo de Brin y Page

3.1. Sin contemplar nodos *dangling*

El modelo inicial para el cálculo del vector PageRank se basaba en calcular el vector estacionario de la matriz P de orden n , definida por (3), siempre que esta matriz fuera estocástica y primitiva. En este modelo no se contemplan los nodos sin salida¹¹ con lo cual c_j es no nulo para todo j y, en consecuencia, P es estocástica. Sin embargo, Brin y Page se dieron cuenta que la estructura de

¹¹Nos referimos a los *dangling nodes*, nodos sueltos o colgados. Brin y Page dicen que los tenían en cuenta al final de los cálculos, pero no dejan muy claro como lo hacían.

la web daba lugar a que P no fuera primitiva e introdujeron un nuevo modelo basado en una matriz estocástica P' que podemos escribir en la forma:

$$P' = \alpha P + (1 - \alpha)ve^T, \quad (4)$$

donde $0 < \alpha < 1$, $e^T \in \mathbb{R}^{1 \times n}$ es el vector de unos: $e^T = [1 \ 1 \ \dots \ 1 \ 1]$, y $v \in \mathbb{R}^{n \times 1}$ es el llamado vector de personalización o de teleportación (*personalization vector*, véase [1], [33]) y es un vector de distribución de probabilidad que se suele tomar como $v = \frac{1}{n}e$. El producto ve^T es una matriz de orden n . El parámetro α se denomina de amortiguamiento (*damping*) y se suele tomar $\alpha = 0,85$, ya que fue el que usaron originalmente Brin y Page [57]. Diversos trabajos actuales se centran en la influencia de α [6], [11], [12]. Se dice que en la ecuación (4), la matriz P' es una combinación lineal convexa de las matrices P y ve^T .

El término $(1 - \alpha)ve^T$, con v un vector de distribución de probabilidad positivo, da lugar a que todos los elementos de P' sean no nulos, con lo cual, P' es irreducible; véase en [51] otra forma de forzar la irreducibilidad. El efecto estadístico de este término es introducir saltos aleatorios que no dependen de las propiedades de enlace de la página. Valores de α próximos a uno ofrecen comportamientos más realistas pero pueden arruinar la irreducibilidad (en el límite $\alpha=1$) y aumentar el número de iteraciones del método de la potencia [47]. Nótese que $\alpha=1$ correspondería a usar la matriz de conectividad real de la web.

Por otra parte, es conocido [9], que una matriz irreducible y no negativa con algún elemento diagonal no nulo es primitiva. En consecuencia, si no hay nodos sin salida, la matriz P' es estocástica y primitiva, que es lo que se desea. Sin embargo, en internet hay páginas sin enlaces salientes y se han de incorporar al modelo; en la sección siguiente vemos una solución.

3.2. Modelo contemplando nodos sin salida

Cuando hay nodos que no tienen enlaces salientes hemos visto que en las columnas respectivas se tiene que $c_j = 0$ y estas columnas están llenas de ceros. En consecuencia P no será estocástica ni tampoco lo será P' (para ilustrar este hecho, úsese la matriz b) del ejemplo 1 con $\alpha = 0,85$ y $v = [1, 0, 0, 0]^T$). Se define entonces la nueva matriz [8], [20], [42], [47], [50], [51]:

$$\bar{P} = \alpha[P + vd^T] + (1 - \alpha)ve^T, \quad (5)$$

donde v y e son los mismos que en el modelo anterior (aunque este nuevo vector de personalización podría tomarse diferente) y el vector $d \in \mathbb{R}^{n \times 1}$ se define como: $d_i = 1$, si $c_i = 0$, y $d_i = 0$ en otro caso. De esta forma, la matriz \bar{P} , que generalmente se denomina *matriz Google*, es estocástica aunque haya nodos sin salida. La matriz vd^T actúa sobre las columnas pertenecientes a nodos sin salida, asignándoles una probabilidad de salto no nula. Por eso se dice que el modelo PageRank es un *modelo de paseo y salto*. La matriz \bar{P} tiene la desventaja de ser una matriz densa y enorme, pero, como veremos en la sección siguiente, no hará falta calcularla explícitamente.

4. Métodos basados en el método de la potencia

Hemos indicado que el vector PageRank, es decir, el vector estacionario de la cadena de Markov ergódica definida por \bar{P} , es el vector de distribución de probabilidad x que verifica el problema de valores y vectores propios:

$$\bar{P}x = x. \quad (6)$$

Desde que Brin y Page [57] anunciaron la aplicación del método de la potencia (*power method*) [77] para el cálculo del vector PageRank, diversos investigadores se pusieron a trabajar en formas de mejorar el funcionamiento del mismo [2], [10], [32], [33], [41], [64]. En las secciones siguientes presentamos el método de la potencia y diversas modificaciones del mismo.

4.1. El método de la potencia

En el cuadro 1 se muestra el algoritmo del método de la potencia para el cálculo de x que resuelve (6). Se ha usado la *norma uno*, $\|x\|_1 = \sum_{i=1}^n |x_i|$. Nótese que para un vector de probabilidad se tiene $\|x\|_1 = 1$, y las matrices estocásticas por columnas conservan esta norma, es decir $\|\bar{P}x\|_1 = \|x\|_1$. Para el criterio de parada, dado por la tolerancia ϵ , se ha usado también esta norma [13], [25]. Según el cuadro 1 una iteración del método de la potencia requiere el cálculo del producto matriz-vector Px^k . En general esto requiere del orden de n^2 operaciones, es decir, $O(n^2)$. Sin embargo, la matriz P es casi vacía. Algunos autores han indicado que el número medio de enlaces salientes por página es del orden de 7 [43] u 8 [35]. Es decir, cada columna de P solo tiene 7 u 8 elementos no nulos. Como consecuencia, el producto Px^k es del orden de $O(n)$ operaciones, es decir unos 25,000 millones de operaciones. Es importante destacar que este algoritmo no necesita construir explícitamente la matriz \bar{P} para resolver (6). Para aclarar este punto notemos primero que dado que P no es en general estocástica, porque tiene *dangling nodes*, se tiene que:

$$\|Px^0\|_1 = \|x^0\|_1 - d^T x^0. \quad (7)$$

Usando esta ecuación, el parámetro que hemos llamado γ en el algoritmo del cuadro 1 resulta ser, para $k = 0$:

$$\gamma = \|x^0\|_1 - \|x^1\|_1 = \|x^0\|_1 - \|\alpha Px^0\|_1 = (1 - \alpha) + \alpha d^T x^0,$$

donde hemos usado que $\|x^0\|_1 = 1$. De la anterior expresión se tiene que el producto γv vale: $\gamma v = (1 - \alpha)v + \alpha d^T x^0 v$ y como $d^T x^0$ es un escalar esto es lo mismo que: $\gamma v = (1 - \alpha)v + \alpha v d^T x^0$, con lo cual, la tercera línea del bucle del algoritmo del cuadro 1 resulta:

$$x^1 = \alpha Px^0 + (1 - \alpha)v + \alpha v d^T x^0 = \alpha [P + v d^T] x^0 + (1 - \alpha)v,$$

y como $e^T x^0 = 1$, podemos introducir este producto en el segundo término, quedando finalmente:

$$x^1 = \alpha [P + v d^T] x^0 + (1 - \alpha) v e^T x^0,$$

Inicialización: $x^0 = e/n, \quad k = 0$ repeat $x^{k+1} = \alpha P x^k$ $\gamma = \ x^k\ _1 - \ x^{k+1}\ _1$ $x^{k+1} = x^{k+1} + \gamma v$ $\delta = \ x^{k+1} - x^k\ _1$ $k = k + 1$ until $\delta < \epsilon$
--

Cuadro 1: Algoritmo 1. Método de la potencia.

y si comparamos con la ecuación (6) esto es, precisamente: $x^1 = \bar{P}x^0$. Con lo cual ya se ve que el algoritmo del cuadro 1 realiza las iteraciones del método de la potencia $x^{k+1} = \bar{P}x^k$ sin necesidad de construir explícitamente la matriz densa \bar{P} . Esto es muy importante ya que supone un ahorro tanto en tiempo de computación como de memoria para almacenar los cálculos. Y estos son dos de los factores críticos en cálculo científico.

Con respecto a la velocidad de convergencia del método de la potencia es conocido [69], [77] que viene determinado por el cociente entre los dos primeros valores propios (en módulo) de \bar{P} . En [47] se muestra que si $\bar{P} = P + vd^T$, tiene el espectro $\{1, \lambda_2, \lambda_3, \dots, \lambda_n\}$ entonces el espectro de \bar{P} , dada por (5), es $\{1, \alpha\lambda_2, \alpha\lambda_3, \dots, \alpha\lambda_n\}$. Como consecuencia, la velocidad de convergencia viene dada por $\frac{|\lambda_1(\bar{P})|}{|\lambda_2(\bar{P})|} = \frac{1}{\alpha|\lambda_2(P)|}$. Como \bar{P} es estocástica se tiene $|\lambda_2(\bar{P})| \leq 1$ y esto junto al hecho que $\alpha < 1$ hace que $\frac{|\lambda_1(\bar{P})|}{|\lambda_2(\bar{P})|} \gg 1$ y la velocidad de convergencia es alta¹². De hecho se ha informado de la convergencia del vector PageRank de Google con unas 50 a 100 iteraciones en cálculos que han durado varios días [49], [51], [57]. En [35] se muestra que sobre una web de 80 millones de nodos trabajando con un procesador AMD Athlon a 1,533MHz, unas 50 iteraciones tardan alrededor de 8 horas. Para más detalles acerca del método de la potencia aplicado a cadenas de Markov, véase [69]. En los apartados siguientes mostramos diversas técnicas para acelerar este método.

4.2. Método de la potencia con extrapolación

La idea de partida de este método se debe a Aitken y se remonta a 1937; v. [77]. Según Aitken, cuando se realizan iteraciones del tipo $x^{k+1} = Ax^k$ es de esperar que en una iteración determinada el vector x^{k-1} pueda aproximarse mediante una combinación lineal de los dos vectores propios dominantes (o sea, asociados a los valores propios dominantes) de A , que denotamos aquí como v_1 y v_2 . Es decir, Aitken asume que para una iteración $k - 1$, con $k > 1$, podemos

¹²Nótese que si \bar{P} es reducible, como suele pasar en la web, con $\lambda_2(\bar{P}) = 1$, entonces $\lambda_2(\bar{P}) = \alpha$. Por ello se suele decir que el segundo valor propio de la matriz Google es α .

Inicialización:	$x^0 = e/n, \quad k = 0$
repeat	$x^{k+1} = \alpha P x^k$ $\gamma = \ x^k\ _1 - \ x^{k+1}\ _1$ $\text{if } k + 1 == d + 2$ $x^{k+1} = \frac{x^{k+1} - \alpha^d x^{k+1-d}}{1 - \alpha^d}$ $x^{k+1} = x^{k+1} + \gamma v$ $\delta = \ x^{k+1} - x^k\ _1$ $k = k + 1$
until	$\delta < \epsilon$

Cuadro 2: Algoritmo 2. Método de la potencia con extrapolación.

escribir:

$$x^{k-1} = u_1 + \beta_2 u_2, \quad (8)$$

en donde β_2 es un coeficiente desconocido. Esta idea es la que se usa en [35] para presentar el método de la potencia con extrapolación para el cálculo del vector PageRank. Aquí se tienen las iteraciones $x^{k+1} = \bar{P}x^k$ y se asume $\lambda_2(\bar{P}) = \alpha$. Esto junto con la ecuación (8) nos conduce a:

$$x^k = \bar{P}x^{k-1} = \bar{P}(u_1 + \beta_2 u_2) = u_1 + \alpha\beta_2 u_2, \quad (9)$$

y de (8) y (9) podemos despejar $u_1 = \frac{x^k - \alpha x^{k-1}}{1 - \alpha}$. Nótese que esta fórmula permite calcular una aproximación del vector propio dominante u_1 a partir de x^k y x^{k-1} . La llamada *extrapolación simple* consiste en utilizar esta fórmula como aproximación a x^3 , es decir: $x^3 = \frac{x^3 - \alpha x^2}{1 - \alpha}$, para después seguir usando $x^4 = Ax^3$, $x^5 = Ax^4$, etc. Los autores de [35] muestran que la extrapolación simple no da buenos resultados y generalizan entonces este método a una extrapolación de grado d , que llaman extrapolación A^d , basada en la fórmula de corrección: $x^k = \frac{x^k - \alpha^d x^{k-d}}{1 - \alpha^d}$. Esta fórmula da lugar al algoritmo 2, mostrado en el cuadro 2. Nótese que la fórmula de extrapolación sólo se emplea una vez y su uso viene gobernado por el parámetro d . Tomar $d = 1$ corresponde a la extrapolación simple. Estos autores realizan experimentos sobre una web de 80 millones de nodos y concluyen que los mejores resultados, en tiempos de convergencia, se obtienen para $d = 6$, resultando una reducción del 30% del tiempo requerido por el método de la potencia clásico del algoritmo 1.

4.3. Método Adaptativo

Es conocido que la velocidad de convergencia de las componentes del vector PageRank no es uniforme; las componentes de una gran mayoría de páginas convergen muy rápidamente, mientras que un pequeño número, las que tienen mayor PageRank, tardan mucho más en converger. Así, Kamvar *et al* [41]

observaron que para una red de 280,000 nodos con 3 millones de enlaces, bastaban unas 16 iteraciones para la convergencia del 66 % de las componentes del vector PageRank. Sugirieron entonces que no hacía falta iterar en las componentes de x^k que ya han alcanzado la convergencia. Llamando $x^k = (x_1^k, x_2^k, \dots, x_n^k)$, un criterio de convergencia para asumir que la componente i -ésima de x ya ha convergido es, por ejemplo: $|x_i^{k+1} - x_i^k|/|x_i^k| < 10^{-3}$. Supongamos ahora, sin pérdida de generalidad, que cuando ya han pasado k iteraciones, tenemos que las componentes x_i^k con $i = 1, 2, \dots, p$ no han convergido mientras que el resto de componentes x_i^k con $i = p+1, \dots, n$ sí lo han hecho. Llamemos $x_N^k = (x_1^k, \dots, x_p^k)$, $x_C^k = (x_{p+1}^k, \dots, x_n^k)$, escribamos $x^k = \begin{bmatrix} x_N^k \\ x_C^k \end{bmatrix}$, consideremos el esquema iterativo:

$$x^{k+1} = Ax^k, \quad (10)$$

y hagamos una partición 2×2 en bloques de la matriz A compatible con la partición hecha en x^k , es decir: $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$, con $A_{11} \in \mathbb{R}^{p \times p}$, $A_{12} \in \mathbb{R}^{p \times (n-p)}$, $A_{21} \in \mathbb{R}^{(n-p) \times p}$, $A_{22} \in \mathbb{R}^{(n-p) \times (n-p)}$. Lo interesante de todo esto es que las matrices A_{21} y A_{22} pueden substituirse por ceros y la matriz A_{12} no es necesaria para posteriores iteraciones. Para ver esto basta con notar que las iteraciones (10) pueden escribirse en la forma:

$$x^{k+1} = \begin{bmatrix} x_N^{k+1} \\ x_C^{k+1} \end{bmatrix} = \begin{bmatrix} A_{11} & O \\ O & O \end{bmatrix} \begin{bmatrix} x_N^k \\ 0 \end{bmatrix} + \begin{bmatrix} A_{12}x_C^k \\ x_C^k \end{bmatrix}.$$

Nótese que en cada iteración $k+1, k+2, \dots$ sólo hay que calcular el producto $A_{11}x_N^k$ ya que el resto de variables corresponden al estado conocido (convergió) en la iteración k . Hay que indicar que en la práctica las componentes que convergen no estarán ordenadas como hemos supuesto aquí y habrá que identificarlas para considerar como nulas las correspondientes filas de la matriz de iteración A . En experimentos sobre webs reales se ha obtenido que unas 8 iteraciones es suficiente para asumir que la convergencia en ciertas componentes ya es válida. El proceso de adaptación se puede ir repitiendo cada 8 iteraciones, de manera que cada vez se eliminan más componentes de A . Con esta técnica se han obtenido mejoras, respecto del método de la potencia clásico, del orden del 22 % en tiempo de cálculo [41].

Rungsawang y Manaskasemsak llevan a cabo, en [62], una paralelización de este método, después de haber experimentado con la paralelización sobre el método usual de la potencia [54], [61]. En sus experimentos con el método de la potencia adaptativo en paralelo sobre una web de 28 millones de páginas, usando 32 procesadores en paralelo, consiguen aceleraciones del orden de 6 a 8 veces mayor que la conseguida con el método usual (método de la potencia no adaptativo). Los autores de este trabajo, que se está llevando a cabo actualmente, creen que todavía hay muchas opciones para mejorar los tiempos de ejecución. La matriz utilizada, así como otras matrices correspondientes a grafos de páginas de internet pueden obtenerse del servidor [81].

```

 $q^1 = q / \|q\|_2$ 
for  $j = 1$  to  $k$ 
   $z = Aq^j$ 
  for  $i = 1$  to  $j$ 
     $h^{i,j} = (q^i)^T z$ 
     $z = z - h^{i,j} q^i$ 
  end for
   $h^{j+1,j} = \|z\|_2$ 
  if  $h^{j+1,j} = 0$  break
   $q^{j+1} = z / \|h^{j+1,j}\|_2$ 
end for

```

Cuadro 3: $\text{Arnoldi}(A, q, k)$.

4.4. Algoritmo del tipo Arnoldi

El algoritmo de Arnoldi (v. cuadro 3) es un conocido método para construir una base ortonormal de un *espacio de Krylov*, concepto definido más adelante. Golub y Gerif [28] presentan una adaptación del llamado método de Arnoldi refinado, para el cálculo del vector PageRank. Este algoritmo se esquematiza en el cuadro 4, donde \tilde{I} denota una matriz identidad aumentada con una fila de ceros y $V^{*,k}$ indica la columna k -ésima de V . Estos autores muestran que el criterio de convergencia $\sigma_{\min}(H^{k+1,k} - \tilde{I}) < \epsilon$ es equivalente a $\|Aq - q\| < \epsilon$. Por tanto, aplicando el algoritmo a la matriz \tilde{P} se obtiene como resultado el vector PageRank. Los resultados numéricos presentados muestran que para $\alpha = 0,85$ no se mejoran los tiempos de cálculo del método de la potencia. Sin embargo, para $\alpha = 0,99$ el algoritmo es mucho mejor que el método de la potencia, llegando a necesitar menos de la mitad de iteraciones para $k = 16$. Para $\alpha = 0,999$ el método de la potencia necesita 7,000 iteraciones mientras que el algoritmo propuesto llega a la misma tolerancia con 432 iteraciones, con $k = 4$. Véase [28] para más detalles.

```

Repeat
   $[Q^{k+1}, H^{k+1,k}] = \text{Arnoldi}(A, q, k)$ 
  Compute  $H^{k+1,k} - \tilde{I} = U\Sigma V^T$ 
  Set  $v = V^{*,k}$ 
  Set  $q = Q^k v$ 
Until  $\sigma_{\min}(H^{k+1,k} - \tilde{I}) < \epsilon$ 

```

Cuadro 4: PageRank usando una variante del método de Arnoldi refinado.

4.5. Métodos de agregación/desagregación

Kamvar *et al* [42] proponen sacar partido a que la web se encuentra agrupada formalmente en servidores que concentran gran cantidad de enlaces. La idea consiste en calcular un vector PageRank local para cada servidor¹³. Estos vectores se usarán después para construir un vector PageRank asociado a toda la web que se utilizará como vector inicial para el cálculo estándar del vector PageRank. Por tanto este método puede considerarse como un método para calcular un vector inicial apropiado que aumente la velocidad de convergencia. Para calcular el vector PageRank local se puede hacer uso de cualquiera de los métodos mencionados anteriormente o cualquier otro, ya que si los servidores son pequeños los problemas de memoria para el cálculo del vector PageRank local no serán ahora un problema serio. El algoritmo propuesto en [42] se denomina *BlockRank* y consta de los pasos siguientes:

1. Separación de la web en k dominios (servidores o conjuntos de servidores). Por ejemplo, en la figura 3 mostramos una web separada en tres dominios, cada uno de ellos conteniendo sus páginas web correspondientes.
2. Cálculo del vector PageRank local $l_j, j = 1, \dots, k$ asociado a cada dominio, con $\|l_j\|_1 = 1, \forall j$.
3. Cálculo del vector BlockRank. Este es un vector PageRank asociado a la web considerada como bloques de páginas; en la figura 4 corresponde a un vector PageRank asociado a los nodos enumerados como 1, 2 y 3. Este vector se calcula asignando a los k bloques la matriz de Markov $B = L^T AS$, donde L es una matriz $n \times k$ formada con los k vectores PageRank locales de la siguiente forma:

$$L = \begin{bmatrix} l_1 & & & \\ & l_2 & & \\ & & \ddots & \\ & & & l_k \end{bmatrix}, \quad (11)$$

y S es una copia de L pero sustituyendo por unos los elementos de L no nulos. El vector BlockRank es entonces el vector b solución de $Bb = b$.

4. El vector inicial de iteraciones para el cálculo del PageRank global (o sea el asociado a la matriz A), se toma como $x^0 = Lb$.

Este método cuenta con muchas ventajas. Primero, dado que se divide el problema en problemas pequeños, el cálculo de cada PageRank local se puede realizar de forma eficiente y rápida ya que todos los datos pueden caber en la memoria caché de un ordenador estándar. Segundo, como los cálculos del PageRank local son independientes se pueden usar procesadores en paralelo. Tercero, por lo mismo que en el punto anterior, se puede estar calculando un

¹³Idea ya usada en [2] donde se habla de *Host graph*.

vector PageRank local de un cierto dominio y, a la vez, realizar una navegación (*crawl*) por otro dominio con el objeto de actualizar el grafo del mismo. Una última ventaja es que este método permite definir un vector de personalización por bloques atendiendo a los servidores. Definiendo un vector $v = (v_1, v_2, \dots, v_k)$ se puede penalizar la importancia de ciertos dominios y resaltar la de otros. Esto es más efectivo que diseñar un vector de personalización global discriminando millones de páginas individuales. Los autores de este método obtienen mejoras de hasta el 200 % en tiempo de computación cuando se comienza el algoritmo PageRank usual con el vector inicial que ofrece este método; v. [42] para más detalles.

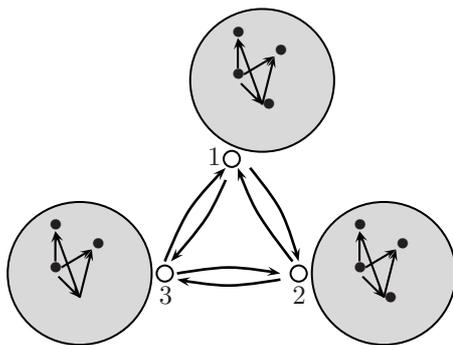


Figura 3: BlockRank sobre una web con tres dominios.

En 2004, Broder¹⁴ et al [14], a la sazón investigadores de IBM y Yahoo!, anunciaron que habían desarrollado una versión similar pero mejorada del método BlockRank, que denominaron *Método de Agregación de grafos*. Este método no calcula un vector inicial x^0 sino una aproximación al vector PageRank x . En sus experimentos dicen que logran aumentar la velocidad del método de la potencia clásico hasta en un 210 % y eso sin optimizar los códigos. Desafortunadamente, el artículo donde muestran los resultados consta de dos páginas y deja más dudas de las que resuelve. Otro investigador de Yahoo!, Pavel Berkhin [8], indica que este método de agregación constituye la técnica actual más práctica de aceleración y se ha venido usando con datos reales del año 2003 de la compañía AltaVista. En la sección 5 veremos que en ciertos casos existen mejores resultados para esta misma web mediante computación en paralelo [25], [26].

Los dos métodos anteriores pueden considerarse como métodos de agregación/desagregación [69]. En [48] y [49] se presenta el método iterativo de agregación/desagregación (IAD) para el cálculo del PageRank. La idea, en términos generales, es la siguiente: conocido el vector PageRank x (por ejemplo, el del mes pasado) y la matriz P que se usó entonces, se quiere emplear esta información para construir el vector inicial de una nueva ejecución del método de la potencia. El problema es que la red habrá cambiado, y hay que actualizar

¹⁴Desde noviembre de 2005 el doctor Andrei Broder es investigador de Yahoo!, donde ocupa el cargo de vicepresidente de nuevas tecnologías de búsqueda.

la matriz P . Los cambios pueden deberse a que se han añadido o eliminado páginas, a que han cambiado un poco los enlaces o a que unos pocos enlaces han cambiado mucho. Se calcula entonces la matriz P actualizada y a partir de ella se obtiene una matriz más pequeña, A , para la cual resulta más rápido calcular un vector PageRank x_A . Después, x_A se emplea para construir un vector x^0 de inicio del método de la potencia en la matriz P actualizada. Langville y Meyer muestran resultados, obtenidos con MATLAB sobre webs pequeñas, con mejoras, en tiempos de cálculo, de hasta el 83%. Indican que este método puede acelerarse con una técnica del tipo extrapolación como la comentada en la sección 4.2. En [40] se da un análisis de la convergencia de este método [48] y se muestra que puede considerarse como un método de la potencia preconditionado por una factorización parcial LU .

Por último, hay que citar a Ipsen y Selee [39] que presentan un algoritmo que agrupa todas las páginas sin salida y aplica entonces el método de la potencia. Su algoritmo es una generalización de los que acabamos de comentar y permite el uso del TrustRank [30] el algoritmo patentado por Google en 2005 para luchar contra el *web spamming*.

4.6. Método de la potencia particionado y en paralelo

En [13] se introduce un modelo para el cálculo del vector PageRank que usa el método de la potencia (como en el algoritmo 1 del cuadro 1) con un modelo del tipo *hipergrafo*. Aquí el elemento central es cómo se realizan los productos matriz-vector. Según estos autores, los modelos en paralelo suelen usar una distribución de los datos en los procesadores del tipo unidimensional, esto es, asignando a cada procesador filas o columnas enteras de la matriz poco densa. Los modelos hipergrafo utilizan otro mecanismo, llamado bidimensional, de distribución de los elementos no nulos de la matriz sobre los procesadores [74]. Los autores sostienen que la técnica es recomendable para el caso de cálculo con diferentes vectores de personalización. Consiguen reducir a la mitad el tiempo usado por iteración con respecto al método usado en [25] y sugieren que esta técnica puede desarrollarse también con una formulación con sistema lineal.

4.7. PageRank asíncrono

En [46], se presenta un método iterativo asíncrono para el cálculo del vector PageRank. Lo denominan PageRank asíncrono y se trata de una puesta en práctica del método de la potencia sin normalización, como en el algoritmo 1 del cuadro 1, mediante esquemas iterativos asíncronos. Los primeros resultados, sobre una web de 281,903 nodos muestran que las técnicas asíncronas pueden ser una buena alternativa en entornos con memoria distribuida. Se informa de aumento de velocidades (en tiempo de cálculo) de hasta 2,66 veces respecto de la que se obtiene con el método de la potencia usual (síncrono).

En la siguiente sección nos ocuparemos de las formulaciones que hacen uso de la escritura de un sistema de ecuaciones lineales para el cálculo del PageRank.

5. Métodos basados en un sistema lineal

Hemos visto que Brin y Page calcularon el vector PageRank utilizando el método de la potencia [57]. Más tarde [2], [10], [19], [56], el problema se formuló mediante un sistema de ecuaciones lineales. Los métodos directos de solución no son aconsejables dado los tamaños, incluso con matrices tan poco densas como las que aparecen. Por tanto, los métodos usados son iterativos; v. [15], [29], [31], [63], [73] para una descripción de métodos iterativos. El uso de estos métodos aplicados en este contexto ha puesto de manifiesto la importancia del parámetro de amortiguamiento α . Para $\alpha = 0,85$ ya se ha dicho que el método de la potencia funciona adecuadamente, al separar suficientemente los dos primeros valores propios [34], pero introduce una cantidad excesiva de saltos aleatorios. Veremos que para $\alpha > 0,85$ los métodos iterativos usados para la solución de un sistema lineal se han revelado como una alternativa para el cálculo del vector PageRank [25]. De hecho, los resultados que Arasu *et al* [2] mostraron en 2002 se obtuvieron usando el método iterativo de Gauss-Seidel con el parámetro $\alpha = 0,9$. Antes de describir con detalle algunos métodos, en la sección siguiente repasamos algunos conceptos sobre métodos iterativos.

5.1. Métodos iterativos para resolver un sistema lineal

De acuerdo con [31] decimos que un método iterativo para resolver el sistema de ecuaciones lineales $Ax = b$ es aquel que produce los vectores x^1, x^2, \dots , (llamados iteraciones) a partir de un vector inicial x^0 siguiendo una regla del tipo $x^{k+1} = \Phi(x^k)$, donde Φ es una función que depende de A y b . Se suele escribir $x^{k+1} = \Phi(x^k, b)$, $k = 0, 1, 2, \dots$ y se dice que \bar{x} es un punto fijo del método iterativo si $\bar{x} = \Phi(\bar{x}, b)$. Si Φ es una función continua respecto del primer argumento y existe el límite $\lim_{k \rightarrow \infty} x^k$ entonces este límite es un punto fijo de Φ . Un método se dice *convergente* si para todo b existe el límite \bar{x} y es independiente del valor inicial x^0 . El análisis teórico de la convergencia de los diversos métodos iterativos empleados para el cálculo del vector PageRank se basa en los dos teoremas siguientes [9].

Teorema 1 *Sea A una matriz invertible. Sea la descomposición $A = M - N$, con M una matriz invertible. Sea el esquema iterativo:*

$$x^{k+1} = M^{-1}Nx^k + M^{-1}b, \quad k = 0, 1, 2, \dots \quad (12)$$

Entonces, se cumple que este esquema iterativo converge a la única solución de $Ax = b$ para cualquier x^0 si, y solamente si, $\rho(M^{-1}N) < 1$.

El esquema iterativo (12) se suele escribir también en la forma:

$$x^{k+1} = Tx^k + c, \quad k = 0, 1, 2, \dots \quad (13)$$

donde $c = M^{-1}b$ y la matriz $T = M^{-1}N$ se denomina *matriz de iteración* del esquema iterativo. Cuando la matriz A no es invertible se hace uso del siguiente resultado [9], donde I denota la matriz identidad.

Teorema 2 *Sea A una matriz cuadrada singular. Se cumple que el esquema iterativo (13) aplicado a $Ax = b$ converge a alguna solución $x(x^0)$ para cada x^0 si, y solamente si, (1) $\rho \leq 1$, (2) si $\rho(T) = 1$ entonces $\text{rango}(I - T) = \text{rango}(I - T)^2$ y (3) si $\rho(T) = 1$ entonces si un valor propio tiene módulo 1 ha de ser el valor propio 1.*

Nótese que el método de la potencia consiste en resolver el sistema $Ax = 0$ con $A = I - P$, no invertible, usando el sistema iterativo (13) tomando $M = I$, $N = P$, lo que lleva a la matriz de iteración $T = P$. El teorema 2 muestra que este método es convergente cuando la matriz P es la matriz de transición de una matriz de Markov ergódica. De forma similar se analiza la convergencia de los métodos de Jacobi, Gauss-Seidel, SOR, etc. [5], [9], [63], [69], [73].

5.2. Cálculo del vector PageRank mediante un sistema lineal

La ecuación de valores y vectores propios (6) puede escribirse también como un sistema homogéneo de n ecuaciones lineales con n incógnitas:

$$(I - \bar{P})x = 0, \quad (14)$$

sujeto a la condición $x^T e = 1$. Se cumple que $I - \bar{P}$ es una M -matriz¹⁵ singular irreducible. Se conocen diversos métodos que pueden aplicarse para resolver (14), v. [69], pero no son aconsejables ya que \bar{P} es muy grande. Para evitar esta situación, se introdujo el siguiente enfoque [2], [19], [47], [56]: sustituyendo en la ecuación (6) el valor de \bar{P} dado por (5), se obtiene el sistema de ecuaciones lineales:

$$(I - \alpha\bar{P})x = (1 - \alpha)v, \quad (15)$$

donde $\bar{P} = P + vd^T$. En este caso se cumple que la matriz de coeficientes $(I - \alpha\bar{P})$ es una M -matriz no singular. De nuevo, como $\alpha\bar{P}$ sigue siendo una matriz grande, tampoco interesa escribirla explícitamente. Recientemente, tanto en [47] como en [19], se presentó una nueva estrategia. Ambos trabajos demuestran, de manera diferente, que el vector PageRank x , solución de (6), se puede obtener resolviendo el sistema de ecuaciones lineales *disperso* (o sea, con pocos elementos no nulos):

$$(I - \alpha P)y = v, \quad (16)$$

y haciendo posteriormente $x = y/(y^T e)$. En este caso la matriz de coeficientes $(I - \alpha P)$ es susceptible de ser calculada y almacenada, ya que tiene un gran número de elementos que son cero. Además, esta matriz es una M -matriz no singular, lo que implica que ciertos métodos iterativos como Jacobi, Gauss-Seidel y los de tipo Schwarz puedan aplicarse con garantías de éxito para la resolución del sistema (16). En la práctica, estos métodos se aplican en la modalidad de bloques, al descomponer el sistema de ecuaciones (16) en diversos subsistemas, que pueden tener variables comunes entre ellos (*overlapping subdomains*, [16], [58], [68], [72]). Entre las estrategias se usan reordenaciones de las matrices [63].

¹⁵Una matriz $A \in \mathbb{R}^{n \times n}$, con elementos no diagonales no positivos, se llama M -matriz si se puede escribir en la forma $A = sI - B$ con $B \geq 0$, y $s \geq \rho(B)$.

Los métodos iterativos que se emplean actualmente incluyen los métodos del tipo de *subespacios de Krylov* [67] y su uso en computación en paralelo [25], [27]. Si x^0 es una aproximación inicial a $Ax = b$, el *residuo* inicial es $r^0 = b - Ax^0$ y el *subespacio de Krylov* de dimensión m definido por A y r^0 es aquel cuyos vectores se obtienen mediante combinaciones lineales de los vectores $r^0, Ar^0, A^2r^0, \dots, A^{m-1}r^0$. Es decir: $\mathcal{K}_m(A, r^0) = \langle r^0, Ar^0, A^2r^0, \dots, A^{m-1}r^0 \rangle$. Los métodos iterativos basados en subespacios de Krylov obtienen una solución x^m del tipo $x^m = x^0 + v^m$ donde v^m pertenece a $\mathcal{K}_m(A, r^0)$. El criterio para obtener x^m es que se minimice alguna función. Por ejemplo, el método GMRES (Generalized Minimal RESidual) se basa en minimizar $\|b - Ax\|_2$ con $x \in \mathcal{K}_m(A, r^0)$. Aquí hemos usado la norma-2: $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$.

La convergencia de los métodos de Krylov puede mejorarse usando *precondicionadores* [7], [63]. Aquí precondicionar significa variar el sistema de ecuaciones inicial $Ax = b$ de manera que obtenemos uno equivalente que es más rápido de resolver por un método iterativo. Por ejemplo, una manera de precondicionar es mediante una matriz M invertible tal que el sistema $M^{-1}Ax = M^{-1}b$ sea fácil de resolver mediante un método iterativo; verbigracia, un método de Krylov. El problema principal de los métodos de Krylov para sistemas grandes, como en el caso de Google, es que se han de almacenar los vectores r^0, Ar^0, A^2r^0, \dots . Aquí la solución pasa por desarrollar una buena paralelización de los algoritmos.

5.3. Algunos trabajos recientes

En [25] se comparan el método de la potencia, el método iterativo de Jacobi y los métodos iterativos de Krylov siguientes: GMRES, BiCGSTAB (Biconjugate Gradient Stabilized), BiCG (Biconjugate Gradient), QMR (Quasi-Minimal Residual), CGS (Conjugate Gradient Squared) y CI (Chebyshev Iterations). Además, en algunas pruebas se usaron como precondicionadores los métodos de Jacobi, Jacobi por bloques y Schwarz aditivo. De las conclusiones del estudio destacamos las siguientes. Para las webs probadas los algoritmos QMR, CGS y CI no convergen. En los casos en que se usaron 60 procesadores ninguno de los métodos de Krylov pudo correr debido a limitaciones de memoria. Los métodos de la potencia y Jacobi tienen aproximadamente la misma velocidad de convergencia y el comportamiento más estable. El método de la potencia es el más rápido (en tiempo total de computación) para α en torno a 0,85. Los métodos GMRES y BiCGSTAB son los que ofrecen mejores resultados para $\alpha = 0,9$, $\alpha = 0,95$ y $\alpha = 0,99$. Consiguen reducir las 10 horas que cuesta un cálculo en serie similar a [14], a unos 35 minutos para el caso del método de la potencia en paralelo y a unos 28 minutos para el método BiCGSTAB con un *cluster* de 140 procesadores (70 máquinas) con 280 GB de memoria. En un trabajo posterior [26] muestran que con ciertas mejoras en el tratamiento del paralelismo consiguen bajar a 333 segundos en el caso del método de la potencia en paralelo y a 391 segundos con el método BiCGSTAB. Hay que resaltar que no indican el tiempo que cuesta almacenar la matriz en memoria. Véase también [21] para otros resultados con métodos de Krylov para calcular el PageRank.

En [45] usan un sistema de identificación de páginas con una 2-tupla de enteros positivos, uno para localizar el servidor (*HostID*) y otro para identificar la página dentro del servidor (*LocalID*). Trabajaron sobre una web de 63 millones de páginas y 801 millones de links distribuidos sobre 470,000 servidores. Estos servidores fueron repartidos en 8 bloques. Sus resultados muestran que con este sistema y utilizando el algoritmo de Gauss-Seidel en paralelo consiguen una velocidad 10 veces mayor que con el método de la potencia en serie.

En [16], la ecuación (16) se resuelve por el método aditivo de Schwarz [17]. Los resultados numéricos, basados en una web pequeña, muestran que el método aditivo de Schwarz con solapamiento es preferible al método de Jacobi por bloques, atendiendo tanto al número de iteraciones como al tiempo empleado para llevarlas a cabo. En general, los mejores resultados se obtiene cuando se realiza una permutación previa de la matriz Google de manera que los elementos queden agrupados en la diagonal.

En [50] se presenta un método llamado *PageRank reordenado*, basado en [52]. Se resuelve el sistema (16) mediante una reordenación previa que agrupa las columnas (filas, en su caso, ya que usan la matriz transpuesta) de ceros. Muestran que el problema se reduce a un sistema lineal muy pequeño, que resuelven con el método de Jacobi. Consiguen multiplicar por 6 la velocidad de cálculo para algunas web consideradas. Otro trabajo reciente es [75] en donde se utiliza un preconditionador del tipo multigrid algebraico (AMG) para una ecuación como (14). Se indica que se obtiene una aceleración significativa de la velocidad de convergencia.

Como se aprecia en esta breve revisión de métodos, hay todavía muchas variables donde encontrar posibilidades de mejora para el cálculo del vector PageRank de una manera más rápida, fiable, y segura.

6. Conclusiones

Los métodos de cálculo para el vector PageRank se pueden separar en dos: aquellos que usan el método de la potencia y los que se basan en un sistema de ecuaciones lineales. Para evitar el empleo de matrices densas se usa el método de la potencia escrito como en el algoritmo 1, cuadro 1, o un sistema de ecuaciones lineales como (16). Los métodos iterativos encuentran su utilidad en valores de $\alpha > 0,85$ donde el método de la potencia puede resultar muy lento. Los estudios analíticos y experimentales no han hecho nada más que comenzar. Las posibilidades de mejora son muy amplias dado que los métodos vistos ofrecen ventajas que son complementarias.

7. Agradecimientos

Deseo agradecer a Rafael Bru una lectura atenta y ciertos comentarios agudos sobre una versión anterior de este manuscrito. Asimismo agradezco a Enrique Fernandez-Cara y a un revisor (o revisora) anónimo las múltiples sugerencias que han enriquecido el presente trabajo.

Referencias

- [1] M. S. Aktas, M. A. Nacar y F. Menczer. Personalizing PageRank Based on Domain Profiles. *Proceedings of the sixth WEBKDD workshop: Webmining y Web Usage Analysis (WEBKDD04)*, in conjunction with the 10th ACM SIGKDD conference (KDD04), (B. Mobasher, B. Liu, B. Masand y O. Nasraoui, eds). Seattle, Washington, Estados Unidos. 2004.
- [2] A. Arasu, J. Novak, A. Tomkins y J. Tomlin. PageRank computation and the structure of the Web: experiments and algorithms. In *The Eleventh International WWW Conference*, ACM Press. Nueva York. Estados Unidos. 2002.
- [3] K. Avrachenkov y N. Litvak. The effect of new links on Google PageRank. *Informe Técnico RR-5256*, INRIA. 2004.
- [4] R. Baeza-Yates. Al-Khorezmi: Un Matemático Olvidado. *Ciencia al día*. Vol 1. No 1. 1998.
- [5] R. Barrett, M. W. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, y H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* SIAM, Filadelfia, Pensilvania, EE.UU, 1994.
- [6] L. Becchetti y C. Castillo . The Distribution of PageRank Follows a Power-Law only for Particular Values of the Damping Factor *WWW 2006*, Mayo 22-26, Edimburgo, Escocia. 2006.
- [7] M. Benzi. Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics* 182, 418-477. 2002.
- [8] P. Berkhin, A survey on PageRank computing. *Internet Mathematics*, vol. 2, No. 1: 73-120. 2005.
- [9] A. Berman y R. J. Plemmons. *Nonnegative matrices in the mathematical sciences*, SIAM, Filadelfia, Pensilvania, Estados Unidos. Reimpresión, 1994.
- [10] M. Bianchini, M. Gori y F. Scarselli. Inside PageRank. *ACM Transactions on Internet Technology (TOIT)*, Volume 5 , Issue 1, pp. 92-128. 2005
- [11] P. Boldi, M. Santini y S. Vigna. PageRank as a function of the damping factor. *Proceedings of the 14th international conference on World Wide Web*.pp. 557-566, Chiba, Japón. 2005.
- [12] P. Boldi. TotalRank: ranking without damping. *Proceedings of the 14th international conference on World Wide Web*.pp. 898-899, Chiba, Japón. 2005.
- [13] J.T. Bradley, D.V. de Jager, W.J. Knottenbelt y Trifunovic, A. Hypergraph partitioning for faster parallel PageRank computation. *Formal techniques for computer systems and business processes proceedings*. 3670: 155-171, 2005.

- [14] A. Broder, R. Lempel, F. Maghoul y J. Pederson. Efficient PageRank Approximation via Graph Aggregation, *Proceedings of the thirteenth World Wide Web Conference*. pp. 484-485. Nueva York, Estados Unidos. 2004.
- [15] R. Bru Métodos iterativos para resolver sistemas lineales. *Bol. Soc. Esp. Mat. Apl.*, vol. 24 , pp. 9-29, 2003.
- [16] R. Bru, F. Pedroche y D. B. Szyld. Cálculo del vector PageRank de Google mediante el método aditivo de Schwarz. En *Congreso de Métodos Numéricos en Ingeniería 2005. J.L. Pérez Aparicio et al (ed.)*, p. 263. Granada, España. 2005.
- [17] R. Bru, F. Pedroche y D. B. Szyld. Additive Schwarz Iterations for Markov Chains, *SIAM J. Matrix Anal. Appl.*, vol. 27, No. 2, pp. 445-458, 2005.
- [18] P. Craven. Google's PageRank Explained and how to make the most of it. <http://www.webworkshop.net/>. 2002.
- [19] G. M. Del Corso, A. Gulli y F. Romani. Fast PageRank Computation Via a Sparse Linear System. *Lecture notes in computer science*, 3243, 118-130. 2004.
- [20] G. M. Del Corso, A. Gulli y F. Romani. Exploiting web matrix permutations to speed up PageRank computation. *Informe técnico*. Instituto di Informatica e Telematica. Universidad de Pisa. Italia. 2004.
- [21] G. M. Del Corso, A. Gulli y F. Romani. Comparison of Krylov Subspace Methods on the PageRank Problem. *Journal of Computational and Applied Mathematics*, en prensa, 2006.
- [22] N. Eiron, K. S. McCurley y J. A. Tomlin. Ranking the web frontier. *Proceedings of the thirteenth World Wide Web Conference*. pp. 309-318. Nueva York, Estados Unidos. 2004.
- [23] A. Farahat, T. LoFaro, J. C. Miller, G. Rae y L. A. Ward. Authority Rankings from HITS, PageRank, and SALSA: Existence, Uniqueness, and Effect of Initialization. *SIAM Journal on Scientific Computing* Volume 27 , Issue 4, pp. 1181-1201. 2006.
- [24] P. Fernández. El secreto de Google y el álgebra lineal. *Bol. Soc. Esp. Mat. Apl.*, vol. 30, pp. 115-141, 2004.
- [25] D. Gleich, L. Zhukov y P. Berkhin. Fast parallel PageRank: A linear system approach. Informe técnico YRL-2004-038, Yahoo!. 2004.
- [26] D. Gleich y L. Zhukov. Scalable computing for power law graphs: experience with parallel pagerank. Informe técnico Yahoo!. 2005.
- [27] D. Gleich, L. Zhukov y P. Berkhin. Fast Parallel PageRank: Methods and Evaluations. *Poster. BASCD 2005*, University of San Francisco. California, Estados Unidos. 2005.

- [28] G. H. Golub y C. Greif. An Arnoldi-type algorithm for computing pagerank. *BIT Numerical Mathematics*. (on-line) 2006.
- [29] G. H. Golub y C. F. Van Loan. *Matrix Computations*, (3r ed.) Johns Hopkins, 1996.
- [30] Z. Gyöngyi, H. García-Molina, Pedersen, J. Combating web spam with TrustRank. *Proceedings of the 30th VLDB Conference* Toront, Canadá. 2004.
- [31] W. Hackbusch. *Iterative solution of large sparse systems*. Springer, Berlin, 1994.
- [32] T. H. Haveliwala. Efficient computation of PageRank. *Informe técnico 1999-31*, Computer Science Department, Stanford University, California, Estados Unidos. 1999.
- [33] T. H. Haveliwala. Topic-sensitive PageRank. In *The Eleventh International WWW Conference*. May 7-11, Honolulu, Hawaii, Estados Unidos. 2002.
- [34] T. H. Haveliwala y S. D. Kamvar. The second eigenvalue of the Google matrix. *Informe técnico*, Stanford University, California, Estados Unidos. 2003.
- [35] T. H. Haveliwala, S. D. Kamvar, D. Klein, C. Manning y G. Golub. Computing PageRank using power extrapolation. *Informe técnico*, Stanford University, California, Estados Unidos. 2003.
- [36] T. H. Haveliwala, S D. Kamvar y G. Jeh. An analytical comparison of approaches to personalizing PageRank. *Informe técnico*, Stanford University, California, Estados Unidos. 2003.
- [37] D. J. Higham. Google PageRank as mean playing time for pinball on the reverse web. *Applied Mathematics Letters*, 18. 1359-1362, 2005.
- [38] I.C.F. Ipsen y R.S. Wills. Mathematical Properties and Analysis of Google's PageRank *Bol. Soc. Esp. Mat. Apl.*, vol. 34, pp. 191-196, 2006.
- [39] I.C.F. Ipsen y T. M. Selee. PageRank Computation, with Special Attention to Dangling Nodes. *SIAM J. Matrix Anal. Appl.*, sometido.
- [40] I.C.F. Ipsen y S. Kirkland. Convergence Analysis of a PageRank Updating Algorithm by Langville and Meyer. *SIAM J. Matrix Anal. Appl.*, vol. 27, no. 4, pp. 952-67. 2006.
- [41] S. D. Kamvar, T . H. Haveliwala y G. H. Golub. Adaptive methods for the computation of PageRank. *Linear Algebra and its Applications*. 386, 51-65. 2004.
- [42] S. D. Kamvar, T. H. Haveliwala, C. D. Manning y G. H. Golub. Exploiting the block structure of the web for computing PageRank. *Informe Técnico*, Stanford University, California, Estados Unidos. 2003.

- [43] J. Kleinberg, S. R. Kumar, P. Raghavan, S. Rajagopalan y A. Tomkins. The web as a graph: measurements, models and methods. En *Proceedings of the international conference on combinatorics and computing*. 1999.
- [44] J. Kleinberg y S. Lawrence. The Structure of the Web. *Science*, Vol. 294, pp. 1849-1850, 2001.
- [45] C. Kohlschutter, P. A. Chirita y W. Nejdl. Efficient parallel computation of PageRank *Advances in information retrieval*. 3936: 241-252, 2006.
- [46] G. Kollias, E. Gallopoulos, y D. B. Szyld. Asynchronous iterative computations with Web information retrieval structures: The PageRank case. *Proceedings of the Conference on Parallel Computing 2005*. 13-16 Septiembre 2005, Málaga. (por aparecer).
- [47] A. N. Langville y C. D. Meyer. Deeper inside PageRank. *Internet Mathematics*, Vol. 1(3):335-380. 2005.
- [48] A. N. Langville y C. D. Meyer. Updating Markov Chains with an eye on Google's PageRank. *SIAM J. Matrix Anal. Appl.*, vol. 27, no. 4, pp. 968-987, 2006.
- [49] A. N. Langville y C. D. Meyer. Updating PageRank using the group inverse and stochastic complementation. *Informe técnico crsc02-tr32*, North Carolina State University, Mathematics Department. Carolina del Norte, Estados Unidos. 2002.
- [50] A. N. Langville y C. D. Meyer. A reordering for the PageRank problem. *SIAM J. Sci. Comput.* 27 (6) 2112-2120. 2006.
- [51] A. N. Langville y C. D. Meyer. A survey of eigenvector methods for web information retrieval. *SIAM Review* 47 (1) 135-161. 2005.
- [52] C. P-C. Lee, G. H. Golub y S. A. Zenios A Fast two-stage algorithm for computing PageRank and its extensions. *Informe técnico SCCM-2003-15*. Stanford University. 2003.
- [53] J. Long. *Hacking con Google*. Anaya Multimedia, Madrid, 2005.
- [54] B. Manaskasemsak y A. Rungsawang. Parallel PageRank computation on a gigabit PC cluster. *18th International Conference on advanced information networking and applications (AINA '04)*, volume 1. 273-277. 2004
- [55] C. B. Moler. The World's Largest Matrix Computation Google's PageRank is an eigenvector of a matrix of order 2.7 billion. *MATLAB News & Notes*. 2002.
- [56] C. B. Moler. *Numerical Computing with MATLAB*. SIAM. Filadelfia, Estados Unidos. 2004.

- [57] L. Page, S. Brin, R. Motwani y T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Technologies Project. 1999. Accesible en: <http://dbpubs.stanford.edu:8090/pub/1999-66>.
- [58] A. Quarteroni y A. Valli, A. *Domain Decomposition Methods for Partial Differential Equations*, Oxford Science Publications, Clarendon Press, Oxford, Reino Unido. 1999.
- [59] R. Raghavan y H Garcia-Molina. Crawling the hidden web. *Informe Técnico*, Stanford Digital Library Technologies Project <http://dbpubs.stanford.edu/pub/2000-36>. 2000.
- [60] M. Richardson y P. Domingos. The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank. In: *Advances in Neural Information Processing Systems 14*. 2002.
- [61] A. Rungsawang y B. Manaskasemsak. Partition-Based parallel PageRank algorithm In *3th International Conference on Information Technology and Applications. ICITA 2005..* vol. 2. pp. 57-62. 2005.
- [62] A. Rungsawang y B. Manaskasemsak. Parallel adaptive technique for computing PageRank. In *Proceedings of the 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP'06)*. pp. 15-50. 2006.
- [63] Y. Saad. *Iterative Methods for Sparse Linear Systems (Second edition)*. SIAM, Philadelphia. 2003.
- [64] S. Serra-Capizzano. Jordan Canonical Form of the Google Matrix: A Potential Contribution to the PageRank Computation *SIAM Journal on Matrix Analysis and Applications* Volume 27 , Issue 2, 305 - 312, 2005.
- [65] M. Serres (ed.). *A history of scientific thought: Elements of a history of science*. Blackwell Publishers Ltd, Oxford, R.U., 1995.
- [66] C. Silverstein. Mathematical Analysis of Hyperlinks in the World Wide Web. *SIAM 50th anniversary and 2002 annual meeting. July, 8-12*. Filadelfia, Pensilvania, Estados Unidos. 2002.
- [67] V. Simoncini y D. B. Szyld. Recent computational developments in Krylov Subspace Methods for linear systems. *Numerical Linear Algebra with Applications*, en prensa.
- [68] B. F. Smith, P. E. Bjørstad, y W. D. Gropp. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Cambridge, Reino Unido. 1996.
- [69] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, Nueva Jersey, Estados Unidos. 1994.

- [70] M. Sydow. Random surfer with back step. *WWW2004*, Mayo 17-22, Nueva York, EE.UU. 2004.
- [71] M. Thelwall. Can Google's PageRank be used to find the most important academic Web pages?. *Journal of Documentation*. Vol. 59. No. 2. pp. 205-217. 2003.
- [72] A. Toselli y O. B. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer, Heidelberg, Alemania. 2005.
- [73] R. S. Varga. *Matrix Iterative Analysis*. Springer. Berlín, Alemania. 2ª edición, 2000.
- [74] B. Vastenhouw y R. H. Bisseling. A two-dimensional data distribution method for parallel sparse matrix-vector multiplications. *SIAM Review* 47 (1) 67-95. 2005.
- [75] Virnik, E. An algebraic multigrid preconditioner for a class of singular M -matrices. *Proceedings of the 13th ILAS Conference*. 18-21 Julio, Amsterdam. Holanda. 2006.
- [76] D. Vise. *The Google Story*. Delacorte Press. Nueva York, EEUU, 2005.
- [77] J. H. Wilkinson *The algebraic eigenvalue problem*. Clarendon Press. Oxford, Gran Bretaña. 1965.
- [78] Los ingresos por publicidad 'online' aumentan, pero pueden frenarse este año. *ELPAIS.es*, 26-09-2006.
- [79] Soluciones publicitarias de Google, <http://www.google.es/intl/es/ads/>.
- [80] The MacTutor History of Mathematics archive, <http://turnbull.mcs.st-and.ac.uk/history/>.
- [81] The Stanford WebBase Project, <http://dbpubs.stanford.edu:8091/testbed/doc2/WebBase/>.