DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico Nº 2

"Ohhh solo tiran π-edras..."

Segundo cuatrimestre de 2015 — Métodos Numéricos

Integrante	LU	Correo electrónico
Frizzo, Franco	013/14	francofrizzo@gmail.com
Martínez, Manuela	160/14	martinez.manuela.22@gmail.com
Rabinowicz, Lucía	105/14	lu.rabinowicz@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Resumen

En este trabajo, se busca solucionar dos problemas sobre elaboración de *rankings*: la clasificación de páginas web para ordenar los resultados de un motor de búsqueda en Internet, y la de equipos según su desempeño en ligas deportivas. Para resolverlos, se presentarán los algoritmos *PageRank* y *GeM*, respectivamente. Ambos proporcionan una solución representando los datos en forma matricial y buscando luego el autovector principal de esta matriz. Se realizan pruebas experimentales sobre estos métodos, variando sus parámetros de entrada, y comparándolos con algoritmos alternativos en términos de eficiencia y efectividad.

Palabras clave: *PageRank*, motores de búsqueda, *rankings* deportivos, cálculo de autovectores

Índice

1. Introducción

En la actualidad, Internet ocupa un lugar central en la vida cotidiana de millones de personas, y una de las principales tareas que estas llevan a cabo en línea es la búsqueda de información. A eso se debe el rol central que ocupan los buscadores, y la gran importancia de contar con algoritmos que permitan indexar y clasificar la información disponible en la red y, muy especialmente, seleccionar de este enorme volumen de datos solo aquellos que sean más relevantes para un determinado contexto de búsqueda.

Por otro lado, las partidos de las grandes ligas deportivas movilizan a enormes masas de seguidores a lo largo y a lo ancho de todo el mundo, y además, importantes intereses económicos dependen de sus resultados. Por lo tanto, es de gran interés que estos resultados puedan definirse con criterios justos y objetivos, que no sea preseten a la especulación y sean robustos ante posibles intentos de manipulación.

Si bien a primera vista estos dos problemas pueden parecer poco relacionados, poseen características en común: en ambos casos, se busca dar un orden de importancia a una determinada colección de elementos, organizándolos en un *ranking* en base información estadística que puede obtenerse sobre las relaciones existentes entre ellos.

En este trabajo, se introducirá uno de los métodos que ha sido utilizado con gran éxito para resolver el primero de los problemas mencionados: el algoritmo de *PageRank*, presentado en 1998 por Sergey Brin y Larry Page[1], que constituye una parte esencial del motor del reconocido buscador Google. Tras una introducción teórica a los conceptos matemáticos que fundamentan este método, se presentará una implementación del mismo y se expondrán resultados experimentales con el fin de analizar su eficiencia y su capacidad para resolver el problema planteado.

Por otra parte, y de forma paralela, se procederá a abordar la situación planteada por el problema de la clasificación de equipos en ligas deportivas. Se mostrará que, tal y como se expone en Govan et al.[4], *PageRank* puede ser adaptado sin mayores dificultades para emplearse como criterio de clasificación en este contexto, y se analizará el caso particular de su utilización para elaborar la tabla de posiciones del torneo de Primera Divisón de la Asociación del Fútbol Argentino (AFA), realizando experimentos para evaluar su desempeño en comparación con el sistema de puntajes actual, y la forma en la que su funcionamiento ve afectado por variaciones en los parámetros del algoritmo.

2. Desarrollo

2.1. Conceptos teóricos

En esta sección, enunciamos a modo introductorio los principales conceptos que funcionan como sustento teórico de los métodos numéricos que emplearemos. Nuestro objetivo no es profundizar en los mismos, dado que ya existen excelentes trabajos realizados que abordan cada uno de ellos más en detalle.

Consideremos un sistema que puede tomar diferentes estados, dentro de un conjunto finito de ellos. Un *proceso estocástico* es una sucesión de estos estados. Si consideramos un caso particular de proceso estocástico, en que la probabilidad de que el sistema se encuentre en un estado dado en el momento $k+1$ queda completamente determinada por su estado en el momento k, tenemos una *cadena de Márkov*.

Supongamos que el sistema tiene n estados posibles, y denominemos p_{ij} a la probabilidad de que el sistema pase del estado j al estado i. Entonces podemos acomodar estos valores en una matriz $\mathbf{P} \in \mathbb{R}^{n \times n}$, que llamaremos *matriz de transición* de la cadena.

Notemos que todas las entradas de la matriz, dado que indican valores de una probabilidad, son no negativas, y además, para $j = 1, \cdots, n$, tenemos que $\sum_{i=1}^{n} p_{ij} = 1$. A una matriz que cumpla con estas condiciones la denominaremos *matriz estocástica por columnas*, y, como enuncian Bryan y Leise en [2], cumplen con la importante propiedad de tener a 1 como autovalor dominante; es decir, 1 es autovalor y además, para cualquier otro $\lambda \in \mathbb{R}$ autovalor de una de estas matrices, se cumple que $|\lambda| < 1$. Si, además, una matriz estocástica cumple que todas sus entradas son estrictamente positivas, entonces podemos afirmar que el espacio de autovectores asociados al autovalor 1 tiene dimensión 1.

En este trabajo, será de interés para nosotros poder calcular este autovector. Se cuentan con métodos muy diversos para realizarlo, pero muchos de ellos son excesivamente costosos, especialmente cuando las matrices son de dimensiones considerables. Afortunadamente, para matrices con las características que acabamos de mencionar, el autovector principal puede calcularse mediante un método iterativo, muy sencillo de formular, conocido como *método de la potencia*. Este método se basa en el siguiente resultado: si x es el autovector principal de una matriz $\mathbf{A}$, y v es un vector inicial cualquiera, entonces $\lim_{k \to \infty} \mathbf{A}^k v = x$. Es decir, partiendo de un vector inicial y multiplicando repetidas veces por la matriz $\mathbf{A}$, el resultado eventualmente convergerá al autovector buscado. Para una demostración de este hecho, como así también una exposición más detallada del proceso, puede consultarse [5, Sección 3]. Como podrá verse en la sección siguiente, y comprobarse luego experimentalmente, este algoritmo resulta muy apropiado para resolver los problemas planteados en este trabajo, y es capaz de aprovechar características propias de las matrices con las que trataremos para mejorar su eficiencia.

2.2. Métodos utilizados

2.2.1. *PageRank*: autovectores para elaborar un *ranking* de páginas web

Cualquier usuario de Internet que haya realizado alguna búsqueda en Google habrá notado que, en la inmensa mayoría de los casos, la información deseada no se encuentra demasiado lejos de los primeros resultados arrojados por el buscador. ¿Cómo es clasificada la ingente cantidad de datos disponibles en la Web para lograr presentar al usuario aquellos que tienen mayor relevancia? El punto clave reside en aprovechar la información que cada página web brinda

acerca de las demás a través de los vínculos o *links* existentes entre ellas. La idea en que se basa el criterio adoptado es simple: cuantas más páginas tengan *links* que apunten hacia otra página dada, más alta será la probabilidad de que esta última contenga información relevante; y mucho más lo será si los enlaces provienen de orígenes que son, a su vez, considerados importantes.

La World Wide Web, como cualquier otra red de páginas unidas a través de enlaces, puede ser considerada como un grafo dirigido, con las primeras representadas por los nodos y los segundos, por las aristas. Como punto de partida para elaborar el *ranking*, recurriremos a una versión modificada de la matriz de adyacencia de este grafo, donde para cada página el peso de los enlaces salientes será inversamente proporcional a su cantidad. Denotemos n_j a la cantidad de *links* salientes que posee la página j, y L_i al conjunto de páginas que poseen un link que apunta hacia i. Entonces, si llamamos $\mathbf{A}$ a esta matriz, tenemos que $a_{ij} = \frac{1}{n_j}$ si $j \in L_i$, y $a_{ij} = 0$ en caso contrario.

Nuestro objetivo es calcular un *ranking* $x = (x_1, \cdots, x_n)$, donde x_i es el puntaje asignado a la página i. Una manera de hacerlo siguiendo el criterio antes mencionado es planteando el sistema de ecuaciones

$$x_i = \sum_{j \in L_i} \frac{x_j}{n_j} \qquad i = 1, \cdots n$$

Notemos que este sistema corresponde a la representación matricial $\mathbf{A}x = x$. Es decir, el problema se reduce a encontrar un autovector de $\mathbf{A}$ asociado al autovalor 1. ¿Podemos garantizar la existencia de una solución a este sistema, y en tal caso, su unicidad?

Analizando la matriz $\mathbf{A}$, podemos notar que, por un lado, $0 \leq a_{ij} \leq 1 (\forall\ i, j)$, y por otra parte, si $n_j > 0$, entonces $\sum_{i=1}^{n} a_{ij} = 1$, y si, por el contrario, $n_j = 0$, entonces $\sum_{i=1}^{n} a_{ij} = 0$. Conceptualmente, cuando una página tiene *links* salientes, los pesos de los mismos suman 1, mientras que las columnas cuyos valores suman 0 corresponden a las páginas que no tienen enlaces salientes (o *nodos colgantes*). Por lo tanto, si consideramos una red sin nodos colgantes, tenemos que $\mathbf{A}$ es una matriz estocástica por columnas; esto garantiza la existencia de un autovector asociado al autovalor 1 y, nos dice, además, que cualquier otro autovalor λ de $\mathbf{A}$ deberá cumplir $|\lambda| < 1$. El modelo construido representa un proceso de Márkov; como explican Brin y Page[1], este puede interpretarse como el itinerario de un *navegante aleatorio*, que posicionado en una página j cualquiera, se dirige a cualquiera de sus links con probabilidad $\frac{1}{n_j}$.

Para resolver el problema de los nodos colgantes, consideramos que cuando el navegante llega a uno de estos nodos, selecciona equiprobablemente una página cualquiera de la red para continuar su recorrido. Matematicamente esto puede representarse con una nueva matriz $\mathbf{P_1} = \mathbf{A} + \mathbf{D}$, donde $\mathbf{D} = vd^t$, con

$$v_i = \frac{1}{n} \qquad \text{y} \qquad d_i = \begin{cases} 1 & \text{si } n_i = 0 \\ 0 & \text{si no} \end{cases}$$

$\mathbf{P_1}$ sí es una matriz estocástica por columnas, lo cual nos asegura que encontraremos solución. Más aún, dado que 1 es el autovalor principal de $\mathbf{P_1}$, sabemos que podremos hacerlo aplicando el método de la potencia. Ahora bien, es altamente deseable que la solución hallada sea única, para que las páginas queden clasificadas de manera única. Con el sistema tal y como está planteado, no es difícil encontrar ejemplos en los que existe más de una solución posible (ver [2]). Afortunadamente, esto también se resuelve de manera sencilla mediante el agregado al modelo de un *factor de amortiguación* o *teletransportación*[1] c, $0 < c < 1$, que permite definir

[1]El nombre *factor de teletransportación* hace referencia aque, desde la interpretación del navegante aleatorio

$$\mathbf{P_2} = c\mathbf{P_1} + (1-c)\mathbf{E}$$

donde $\mathbf{E} \in \mathbb{R}^{n\times n}$ tiene $\frac{1}{n}$ en todas sus posiciones. Notemos que, dado que $\mathbf{P_1}$ y $\mathbf{E}$ son matrices estocásticas por columnas, $\mathbf{P_2}$, que es un promedio ponderado de ambas, también lo es; y que, además, $\mathbf{P_2}$ tiene todos sus coeficientes estrictamente positivos. Es decir, $\mathbf{P_2}$ representa una cadena de Márkov ergódica, y por lo tanto, el sistema $\mathbf{P_2}x = x$ tiene solución única.

2.2.2. *GeM*: adaptando *PageRank* para clasificar equipos en ligas deportivas

El algoritmo presentado en la sección anterior no solo ha resultado sumamente exitoso para resolver el problema para el que fue ideado; también existen otros problemas que pueden resolverse con la misma idea básica. Como ya mencionamos anteriormente, uno de ellos es la elaboración de *rankings* de equipos para ligas deportivas a partir de los resultados obtenidos en los partidos.

En Govan et al.[4], los autores mencionan algunos métodos empleados para este fin, y los comparan frente a uno desarrollado por ellos mismos: *GeM*, que está fuertemente basado en el algoritmo de *PageRank*. La idea es la siguiente: una temporada de una liga deportiva también puede entenderse como un grafo dirigido. Los nodos representan a los equipos, y una arista entre los nodos i y j quiere decir que el equipo i perdió al menos una vez contra el equipo j. El peso de las aristas es proporcional a la diferencia absoluta en el marcador del partido al que representan (la suma de ellas, si se trata de más de un partido), normalizadas de forma tal que la suma de los pesos de todas las aristas salientes de cada nodo sea igual a 1.

A partir del grafo anteriormente mencionado, puede construirse la matriz $\mathbf{A}$ de adyacencia del grafo; y, de la misma manera que en el caso de la clasificación de páginas en buscadores, pueden derivarse de ella las matrices $\mathbf{P_1}$ y $\mathbf{P_2}$ para resolver los problemas de los nodos colgantes (que representan, en este caso, a los equipos invictos) y de la posible existencia de más de una solución. El sistema resultante es completamente análogo al obtenido para calcular *PageRank*, por lo que no detallaremos los pormenores matemáticos que permiten encontrar su solución. Sí deben tenerse en cuenta las inevitables diferencias implementativas que se derivarán de los distintos contextos de uso en los que se desempeñarán ambos algoritmos, que trataremos con más profundidad en la próxima sección.

2.3. Implementación

Ambos métodos fueron implementados en lenguaje C++, reutilizando el código en los casos en que fue posible, pero atendiendo a las diferencias en los contextos de aplicación esperados para cada uno, especialmente a la hora de decidir las estructuras de datos utilizadas.

En los dos casos, una vez leídos los datos de entrada, se crea la matriz correspondiente al sistema a resolver. A continuación, partiendo de un vector de probabilidad uniforme ($v \in \mathbb{R}^n$, con $v_i = \frac{1}{n}$ para $i = 1, \cdots, n$), se aplican sucesivas iteraciones del método de la potencia. Como criterio de detención, se estima la convergencia del método en cada iteración; para esto, se calcula la diferencia Manhattan entre el vector obtenido en cada repetición y en la inmediatamente anterior, y se la compara con un umbral de tolerancia, proporcionado como parámetro al

antes mencionada, la alteración que se efectúa al sistema puede interpretarse como agregarle la posibilidad de que en cada paso, con probabilidad c, el navegante se "teletransporte" a una página elegida al azar, independientemente de la existencia de un *link* que conecte esta con su posición actual.

momento de la ejecución. El algoritmo se detiene cuando la distancia obtenida está por debajo del umbral.

2.3.1. Implementación de *PageRank*

En el caso del algoritmo *PageRank*, se tuvieron especialmente en cuenta dos hechos: por un lado, el contexto de uso hace esperar que deban manejarse enormes cantidades de datos, por lo que es importante hacer hincapié en la eficiencia de la implementación; por otra parte, en una red con una estructura similar a la de la World Wide Web, cabe esperar que la cantidad de links presentes en cada página sea muy escasa en comparación con el número total de páginas. Esto permite optimizar la forma en la que se almacena la información: en particular, la matriz $\mathbf{A}$ correspondiente al sistema, que contiene los datos sobre los *links* de la red, es *esparsa*, es decir, gran parte de sus coeficientes son 0. Dado que las alteraciones posteriores de esta matriz (las efectuadas a fin de solucionar el problema de los nodos colgantes y el de las posibles múltiples soluciones), a partir de las cuales se obtiene la matriz $\mathbf{P_2}$, pueden ser aplicadas *ad-hoc* al momento de operar, se decidió aprovechar la ventaja mencionada y limitarse a almacenar en memoria la matriz $\mathbf{A}$.

La representación elegida fue CSC (*Compressed Sparse Column*). Esta representación consiste en tres vectores. El primero de ellos (`vals`) almacena todos los valores no nulos de la matriz, recorriéndola por columnas; el segundo (`ind_filas`), de igual longitud que el primero, contiene el índice de la fila en que se encuentra el elemento respectivo en `vals`; el tercero (`ptr_cols`), tiene una entrada por columna, y señala en qué posición de `vals` se encuentra el primer valor correspondiente a cada una de ellas. Esta representación fue elegida por sobre DOK (*Dictionary of Keys*) por considerarla más eficiente, tanto en memoria como temporalmente, y por sobre CSR (*Compressed Sparse Row*) porque permite distinguir eficientemente las columnas que solo contienen ceros (que requieren en este caso un tratamiento especial): en la presente implementación, se las representa colocando el valor -1 en la posición correspondiente del vector `ptr_cols`.

El único inconveniente que se encontró con la representación elegida es que resulta muy impráctica a la hora de cargar los datos. Es por eso que como paso intermedio entre la lectura de datos y la creación de la matriz, se utilizó una estructura temporal que se asemeja más al formato de DOK: un vector correspondiente a las columnas, cuyas entradas son a su vez vectores conteniendo los índices de la columna respectiva donde deberá colocarse un valor no nulo.

La implementación del método de la potencia no presentó grandes complicaciones: consiste en un ciclo que inicia con un vector con todas sus componentes iguales a $\frac{1}{n}$ y lo multiplica, en cada iteración, por la matriz del sistema. Implementar este producto implicó algo más de complejidad, ya que la matriz por la que se debe multiplicar es diferente a la almacenada en memoria. La multiplicación se realiza columna a columna, siguiendo el orden natural de la estructura elegida para la matriz. Se utiliza un vector acumulador para calcular el resultado, que se inicializa con todas sus posiciones en 0. Luego, para $i = 1, \cdots, n$, se utiliza la matriz $\mathbf{A}$ para calcular el producto entre la columna i de la matriz $\mathbf{P_2}$ por el valor v_i del vector de entrada, y el resultado se suma al vector acumulador. Para cada columna pueden presentarse dos casos, que deben tratarse de forma distinta:

1. Si la columna i de $\mathbf{A}$ solo contiene valores nulos (es decir, `ptr_cols[i-1] = -1`), tendremos que, para todo $j = 1, \cdots, n$, $p_{2ij} = c(\frac{1}{n}) + (1 - c)(\frac{1}{n}) = \frac{1}{n}$. Luego, el producto de esta columna por v_i puede calcularse directamente, y será un vector con todas sus componentes iguales a $\frac{v_i}{n}$.

2. Si, por el contrario, en la columna i de $\mathbf{A}$ hay valores no nulos, para $j = 1, \cdots, n$, tendremos que $p_{2ij} = c(a_{ij}) + (1-c)(\frac{1}{n})$. Es decir, el producto de esta columna por v_1 puede calcularse en dos pasos: primero, consideramos un vector con todas sus componentes iguales a $(1-c)(\frac{1}{n})$, y luego, para los índices j tales que $a_{ij} \neq 0$, le sumamos $c(a_{ij})$ a la componente correspondiente de dicho vector.

2.3.2. Implementación de *GeM*

La implementación del método *GeM* fue considerablemente más simple, debido a que, si bien ya no es válida la hipótesis de que la matriz obtenida será esparsa, sí puede asumirse que la instancias a tratar serán considerablemente menores. Dado que el algoritmo es básicamente el mismo, gran parte del código de *PageRank* pudo reutilizarse; las principiales modificaciones estuvieron en la forma de representación de la matriz y, por consiguiente, en la implementación del producto de esta con un vector.

Para el almacenamiento de la matriz, se decidió utilizar un vector de vectores (cada uno de estos representando una fila de la matriz), por considerar que brindaba un equilibrio adecuado entre eficiencia y facilidad de implementación. Dado que ahora se guardan en memoria todos los coeficientes, se decidió almacenar $\mathbf{P_2}$ en lugar de $\mathbf{A}$. De esta forma, las operaciones que transforman la segunda en la primera solo se hacen una vez, durante la carga de los datos, y se incrementa la simplicidad y la eficiencia de la función encargada de calcular el producto entre la matriz y el vector. Esta última función opera ahora de la forma tradicional, multiplicando cada fila de $\mathbf{P_2}$ por el vector proporcionado y almacenando el resultado en la componente correspondiente del resultado.

3. Experimentación

En esta sección, se presentan las pruebas experimentales que se realizaron, junto con los resultados obtenidos y una discusión de los mismos.

En primer lugar, se detallan las experiencias relacionadas con el método de *PageRank* y su aplicación en el desarrollo de motores de búsqueda. Estos experimentos se hallan orientados a evaluar, por un lado, el rendimiento y la convergencia del otro, y por otra parte, la calidad de los resultados obtenidos.

A continuación, se encuentran las pruebas relacionadas con respecto al método *GeM*. Estas se encuentran focalizadas en la elaboración de *rankings* para ligas futbolísticas, estudiando la factibilidad de la aplicación del algoritmo en este contexto particular, y diversas alternativas para resolver la dificultad que se presenta a la hora de tener en cuenta los empates.

Cabe señalar que con los archivos fuente del trabajo práctico se incluye una serie de scripts, en lenguajes Bash y Python, que permiten reproducir por completo los experimentos realizados, como así también los gráficos que se incluyen en este informe. Estos se encuentran dentro del directorio `exp`, y llevan los nombres `expi.sh`, siendo `i` el número de experimento.

3.1. Experimentación sobre *PageRank* y páginas web

3.1.1. Experimento 1: Tiempo de ejecución de *PageRank*

Presentación

El objetivo de este experimento fue extraer conclusiones acerca de la variación en el tiempo de cómputo requerido por el algoritmo de *PageRank* para redes de diferentes tamaños, en función de la cantidad de páginas y de la cantidad de *links* de las mismas.

Metodología, datos y parámetros del experimento

Se realizaron pruebas para medir el rendimiento temporal según dos parámetros:

(a) Cantidad de páginas de la red. Se consideraron redes generadas artificialmente con las siguientes cantidades de nodos: {400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000, 4000, 6000, 8000, 10000, 20000, 40000, 60000, 80000, 100000}, estableciendo en todas ellas la misma cantidad de enlaces, 100000.

(b) Cantidad de enlaces de la red. Se consideraron redes generadas artificialmente con una cantidad fija de 1050 nodos, variando la cantidad de enlaces establecidos entre ellos, tomando los valores {1000, 5000, 10000, 20000, 40000, 60000, 80000, 100000, 150000, 200000, 300000, 400000, 500000, 600000, 800000, 1000000}.

En todos los casos, para la generación de las instancias de prueba, se distribuyó uniformemente entre las páginas la cantidad de *links* salientes; las páginas hacia los que estos apuntan fueron determinadas al azar.

El parámetro de amortiguación c se mantuvo fijo con el valor 0.85 (el utilizado normalmente por el buscador Google[1]), mientras que el umbral de tolerancia para la detención del método se fijó en 0.001.

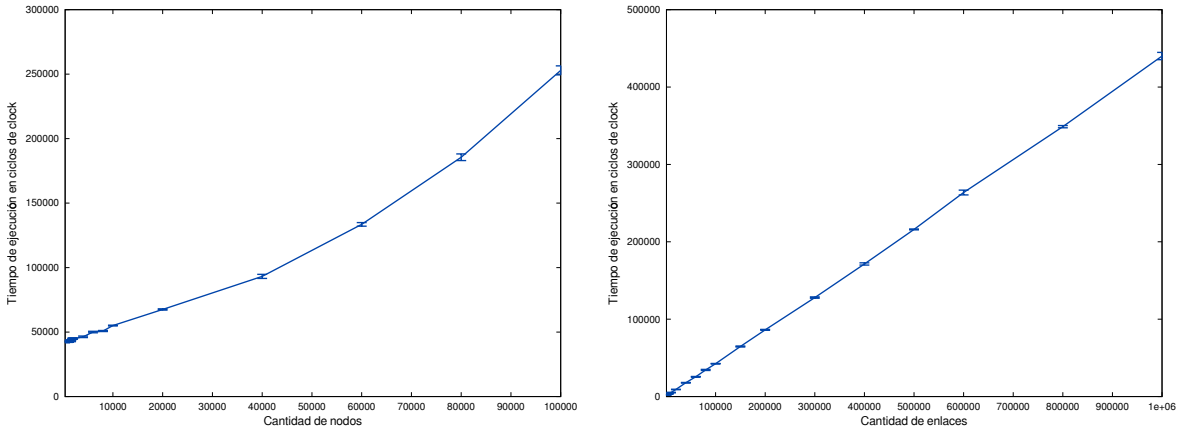
En cada caso, se midió el tiempo necesario para la construcción de la matriz que modela la red y para la ejecución de todas las iteraciones necesarias del método. Para realizar las mediciones se utilizaron las funciones provistas a tal efecto por la biblioteca estándar de C++. Con el fin de evitar posibles errores en las mismas, cada una se repitió 20 veces, considerando luego el promedio entre los valores obtenidos.

Hipótesis

Es importante considerar en este punto la forma en que la implementación del algoritmo *PageRank* aprovecha la propiedad de la matriz que modela la red ($\mathbf{P_1}$) de ser esparsa. Una inspección al código de dicha implementación revela que la única operación costosa que se realiza en cada iteración es el producto entre la dicha matriz y el vector resultante de la iteración anterior. Dentro de este producto existen dos ciclos no anidados; uno de ellos itera sobre todas las posiciones no nulas de la matriz, es decir, linealmente en la cantidad de *links* de la red, para realizar el producto del vector con estos coeficientes. El otro ciclo se realiza sobre todas las posiciones del vector, cuya cantidad es igual a la cantidad de nodos de la red; cabe destacar que este ciclo es extremadamente simple, consistiendo únicamente en un par de sumas.

Teniendo en cuenta lo anterior, cabe esperar que el tiempo de ejecución dependa linealmente tanto de la cantidad de nodos de la red como de la cantidad de *links* de la misma, pero con una constante mucho mayor en este último caso.

Resultados obtenidos y discusión



Resultados arrojados por el experimento 1. El gráfico de la izquierda representa el tiempo promedio de ejecución del algoritmo *PageRank* en función de la cantidad de nodos de la red, y el de la derecha, en función de la cantidad de *links* de la misma. Se representa también el desvío estándar de las mediciones..

Como puede observarse en los gráficos que se incluyen, los resultados obtenidos corroboraron la hipótesis de que el tiempo de ejecución del algoritmo aumentaría linealmente con la cantidad de links de la red evaluada. No obstante, la relación con la cantidad de nodos de la red parece no ser lineal, sino de un orden mayor. Aquí entra en juego, muy probablemente, un hecho que no fue tomado en cuenta al elaborar las hipótesis: la naturaleza iterativa del método, que no permite determinar cuántas veces se ejecutará hasta detenerse a partir de una simple inspección

de los ciclos del algoritmo. Es decir, hay que tener en cuenta que un incremento en la cantidad de nodos de la red puede tener un efecto no solo en la complejidad de cada iteración particular, sino en la cantidad total de iteraciones requeridas para alcanzar un resultado.

En efecto, al considerarse una red con más nodos, los valores del vector que se espera obtener luego de cada iteración disminuyen en magnitud, ya que siempre deben sumar 1, al tiempo que aumenta su cantidad. La evidencia empírica obtenida parece indicar que este incremento progresivo del tamaño de la red causó efectos en el cálculo de la norma Manhattan entre los resultados de iteraciones sucesivas, ralentizando la convergencia del algoritmo y alterando el comportamiento que había sido previsto. Este efecto, por otra parte, no parece tener influencia cuando la variable modificada es la cantidad de *links*.

3.1.2. Experimento 2: Convergencia de *PageRank*

Presentación

Por medio de este conjunto de pruebas experimentales, se buscó analizar la convergencia lograda por el algoritmo *PageRank* en cada iteración. Como parámetro de la misma, se consideró la diferencia entre el resultado arrojado por dicha iteración y el obtenido en la inmediatamente anterior, en términos de la norma Manhattan de esta diferencia. Teniendo en cuenta que este mismo parámetro es el empleado como criterio de parada del método, se buscó extraer conclusiones sobre la relación entre la tolerancia establecida para dicho criterio y la cantidad de iteraciones necesarias para completar el algoritmo. Además, se analizó la influencia del valor del parámetro c en la convergencia del método.

Metodología, datos y parámetros del experimento

A los efectos del experimento, se consideraron las siguientes tres instancias de redes, una de tamaño pequeño y otras dos con cantidad de nodos mayor:

(a) Red pequeña, formada por 50 páginas web reales y la estructura de *links* determinada por las mismas, que al momento del experimento contenía 160 enlaces. La lista de sitios utilizados puede encontrarse en el archivo `exp/exp2-a-webs.txt`, y el grafo generado, en `exp/exp2-a-graph.txt`.

(b) Red mediana, formada por las páginas de la Universidad de Stanford (stanford.edu) y los enlaces entre ellas. Comprende 281903 páginas y 2312497 enlaces. Los datos, extraídos de [6], fueron recopilados en 2002.

(c) Red mediana, formada por las páginas de la Universidad de Notre Dame (nd.edu) y los enlaces entre ellas. Comprende 325729 páginas y 1497134 enlaces. Los datos, extraídos de [6], fueron recopilados en 1999.

Está claro que el tamaño de estas redes no es comparable con el de aquellas que aparecen en el escenario de un buscador web actual (el primer índice de Google, en 1998, contenía 26 millones de páginas [1]), pero se las consideró suficientes para evaluar el comportamiento del método sin incrementar desmesuradamente los tiempos de ejecución, dejando abierta la posibilidad de realizar, en el futuro, pruebas con instancias mayores.

Sobre las tres instancias, se ejecutó el algoritmo *PageRank* hasta que la diferencia entre dos iteraciones sucesivas alcanzó el umbral de tolerancia de 0.001 según la norma Manhattan, registrando el valor obtenido para la misma en cada iteración.

En cuanto al parámetro c, el experimento se repitió para cada instancia con los valores {0.85, 0.90, 0.95, 0.99}. Se desestimaron valores menores de c porque, como se argumenta en [5], estos producen resultados poco relevantes y aumentan demasiado la posibilidad de manipulación del método.

Hipótesis

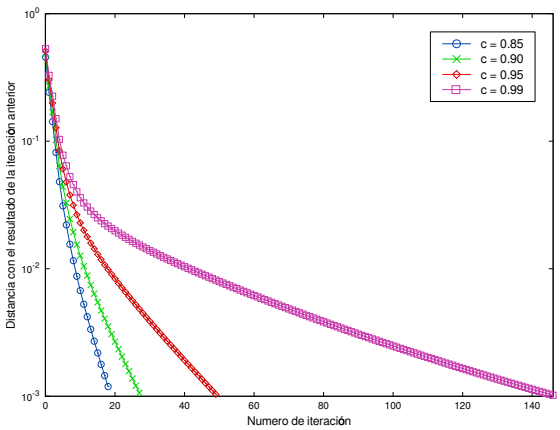
Dada la naturaleza iterativa del método, resulta razonable esperar que el ritmo de convergencia del método disminuya a medida que el resultado obtenido se acerca, de forma asintótica, al estado de equilibrio. Esto significa que el costo requerido en cantidad de iteraciones para aumentar la precisión del resultado obtenido se volverá cada vez mayor, apareciendo, en función de los requerimientos específicos del contexto de aplicación, un umbral mínimo para la tolerancia a partir del cual no valdrá la pena continuar ejecutando el método.

En cuanto al valor de c, es importante recordar que valores pequeños de este parámetro determinan, a la hora de construir la matriz $\mathbf{P_2}$ sobre la que se aplica el algoritmo, un mayor peso relativo de la matriz de amortiguación ($\mathbf{E}$) con respecto a la que contiene los pesos reales de cada enlace ($\mathbf{P_1}$). En otras palabras, conforme disminuye el valor de c, cabe esperar que la matriz $\mathbf{P_2}$ sea más homogénea, o que se "se parezca más" a $\mathbf{E}$.

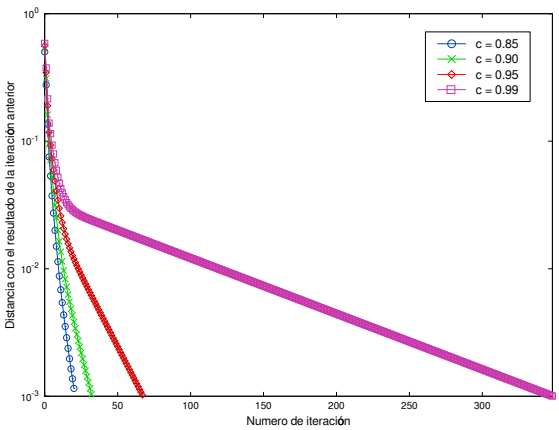
Ahora bien, el vector inicial con el que se realiza la primera iteración es un vector homogéneo, que, como puede deducirse fácilmente, es el resultado esperado en caso de que la matriz de la red sea exactamente $\mathbf{E}$. Por lo tanto, es razonable esperar que ante valores menores de c, el vector inicial sea una mejor aproximación del resultado buscado, produciendo que el método converja más rápidamente.

Por otra parte, los resultados del experimento anterior parecen sugerir que el método tiende a converger más lentamente para redes con más nodos, conjetura que se espera corroborar con las instancias evaluadas en esta prueba.

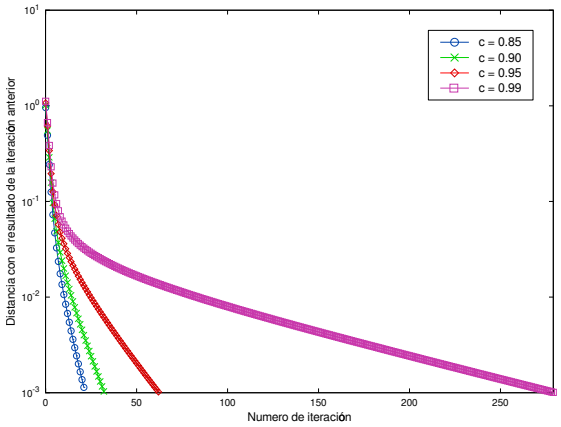
Resultados obtenidos y discusión



Instancia (a)



Instancia (b)



Instancia (c)

Resultados arrojados por el experimento 2. Los gráficos representan la diferencia Manhattan entre el resultado de cada iteración y el de la inmediatamente anterior durante la ejecución del algoritmo, para las distintas instancias y valores de c considerados.

La primer conclusión, que resulta evidente de una sencilla observación de los gráficos obtenidos, es que el valor de c tiene, efectivamente, una gran influencia en la convergencia del método: cuando este pasa de 0.95 a 0.99, por ejemplo, la cantidad de iteraciones requeridas para alcanzar el umbral fijado de tolerancia aumenta a casi el triple, en el menos extremo de los tres casos.

También se corrobora en los gráficos que la convergencia del método es cada vez más lenta conforme avanzan las iteraciones. Se observa, especialmente en el caso de $c = 0{,}99$, que la convergencia es rápida hasta que se alcanza un punto de inflexión, a partir del cual queda relativamente estancada. Su aproximación a una línea recta en los gráficos, cuyo eje y posee escala logarítmica, sugiere que su comportamiento asintótico es un decrecimiento exponencial.

Un interesante hecho a notar es que la evidencia empírica contradice la hipótesis de la existencia de una relación inversa entre la cantidad de nodos de la red estudiada y la velocidad de convergencia del método: la instancia (b), pese a poseer considerablemente menos nodos que la instancia (c), requirió para todos los valores estudiados de c una mayor cantidad de iteraciones para alcanzar el umbral de tolerancia aceptado. No es posible, sin embargo, descartar por completo la posibilidad de que tamaño de la red y convergencia estén relacionadas; es necesario tener en cuenta que en el experimento anterior se comparaban redes con la misma cantidad de enlaces y una estructura semejante, características que no poseen las instancias contrastadas en esta prueba, por lo que se considera pendiente para investigaciones futuras la realización de nuevas pruebas donde se estudie la interferencia de estos factores con los resultados observados.

3.1.3. Experimento 3: Análisis cualitativo de los resultados

Presentación

Por medio de estas pruebas, se pretende realizar un estudio cualitativo de los resultados arrojados por el método *PageRank*. A modo de contraste, y para realizar una evaluación comparativo, se implementó un método de clasificación alternativo más simple, denominado In-deg. Este consiste en ordenar las páginas considerando simplemente la cantidad de *links* entrantes

hacia cada una de ellas. En particular, el puntaje de una página se calcula como el cociente entre la cantidad de enlaces que recibe y el total de los mismos en la red.

Metodología, datos y parámetros del experimento

Para esta experiencia se construyeron artificialmente tres redes pequeñas, pensadas con el objetivo de poner de manifiesto y observar las diferencias entre los dos algoritmos planteados. Estos casos de prueba, que ilustra la Figura 1, son los siguientes:

(a) La Web 1 cuenta con ocho nodos. Seis de los mismos (nodos 3 a 8) se encuentran enlazados de forma tal que cada uno de ellos recibe enlaces de exactamente dos de los otros. Uno de los nodos restantes (nodo 1) recibe un enlace de cada uno de estos seis. El otro (nodo 2) recibe solo un enlace de este último.

(b) La Web 2 también posee ocho nodos. Uno de ellos (nodo 1) recibe enlaces de todos los demás. Otro nodo (nodo 2) solo recibe un enlace del nodo 1. El nodo 3 recibe enlaces de los últimos cinco nodos (del 4 al 8). Estos, por su parte, no tienen enlaces entrantes.

(c) La Web 3 se compone de dos subredes no conectadas, una de 4 nodos y otra de 2 nodos. Dentro de ellas, cada nodo recibe exactamente un enlace entrante desde otro nodo.

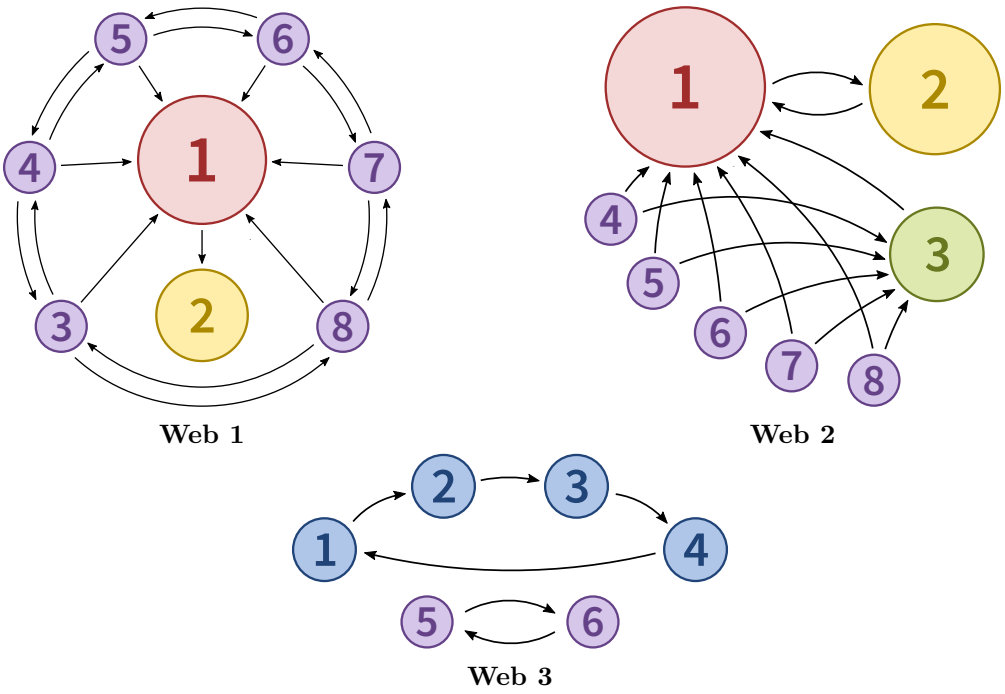


Figura 1: Redes utilizadas como datos de entrada para el Experimento 3. El tamaño con el que aparece representado cada nodo corresponde al *ranking* que la hipótesis planteada prevé como resultado de *PageRank*.

Hipótesis

Las redes estudiadas pretenden poner de manifiesto las principales características distintivas del algoritmo *PageRank*.

La primera de ellas presenta un nodo (el nodo 2) que recibe un solo enlace, menos que todos los demás. No obstante, este es un enlace de gran peso, ya que procede del nodo 1, quien recibe a su vez links de todos los nodos restantes. Cabe esperar que esta página sea ignorada por IN-DEG, pero no por *PageRank*, que seguramente la ubicará por encima de los otros nodos satélites.

La segunda red presenta un escenario más sutil. Aquí la diferencia se espera entre los nodos 2 y 3: el segundo recibe más enlaces, por lo que será más relevante para IN-DEG; el primero, sin embargo, recibe un enlace del nodo 1, que ocupa una posición central en la red. Cabe esperar que esto eleve su posición en *PageRank*. También es de interés que el nodo 2 de esta red, a diferencia del de la red anterior, tiene un enlace saliente hacia 1. Dado que esto tiende a incrementar más el puntaje de 1 en *PageRank*, se espera que resulte perjudicial para el puntaje de 2.

La tercera red es un caso simple de una red no conexa. Para IN-DEG, todas las páginas tendrán el mismo puntaje, ya que poseen exactamente un *link* entrante. *PageRank*, sin embargo, podría establecer una diferencia a favor de la subred mayor, teniendo en cuenta el *factor de teletransportación*; en caso de ser un navegante aleatorio al momento de teletransportarse, es mayor la probabilidad de moverse a una página de la subred de mayor tamaño.

Resultados obtenidos y discusión

Ranking Web 1 - *PageRank*			*Ranking* Web 1 - In-deg		
Pos.	Nodo	Puntaje	Pos.	Nodo	Puntaje
1	2	0.216403	1	1	0.315789
2	1	0.205549	2	3	0.105263
3	3	0.096341	3	4	0.105263
4	4	0.096341	4	5	0.105263
5	5	0.096341	5	6	0.105263
6	6	0.096341	6	7	0.105263
7	7	0.096341	7	8	0.105263
8	8	0.096341	8	2	0.052632

Ranking Web 2 - *PageRank*			*Ranking* Web 2 - In-deg		
Pos.	Nodo	Puntaje	Pos.	Nodo	Puntaje
1	1	0.447829	1	1	0.538462
2	2	0.399828	2	3	0.384615
3	3	0.058594	3	2	0.076923
4	4	0.018750	4	4	0.000000
5	5	0.018750	5	5	0.000000
6	6	0.018750	6	6	0.000000
7	7	0.018750	7	7	0.000000
8	8	0.018750	8	8	0.000000

Ranking Web 3 - *PageRank*			*Ranking* Web 3 - In-deg		
Pos.	Nodo	Puntaje	Pos.	Nodo	Puntaje
1	1	0.166667	1	1	0.166667
2	2	0.166667	2	2	0.166667
3	3	0.166667	3	3	0.166667
4	4	0.166667	4	4	0.166667
5	5	0.166667	5	5	0.166667
6	6	0.166667	6	6	0.166667

Los primeros dos casos reflejaron resultados similares a los planteados en la hipótesis, mostrando hasta qué punto *PageRank* tiene en cuenta el peso de los links por encima de la cantidad de los mismos. El caso de la Web 1 es especialmente destacable, ya que allí la página 2, que recibe tan solo un enlace, quedó por encima de todas las demás, incluso de la página 1, que recibe 6. In-deg, en cambio, la relegó al último lugar.

En la Web 2, en cambio, pese a que el escenario es similar, la página 2 quedó en segunda posición. Esto corrobora la hipótesis de que agregar links salientes puede jugar en contra de la posición de una página en el *ranking* de *PageRank*.

En cuanto a la Web 3, quedó claramente refutada la hipótesis de que las subredes recibirían diferentes puntuaciones de acuerdo a su tamaño. Tanto para In-deg como para *PageRank*, todas las páginas recibieron idéntico puntaje. El escenario de redes no conexas es especialmente interesante por resultar menos intuitivo, y además, porque refleja la estructura real de la Internet, donde se presenta este tipo de subredes. Por lo tanto, profundizar en la experimentación con estos casos es un tema de interés para investigaciones posteriores.

3.2. Experimentación sobre *GeM* y ligas deportivas

3.2.1. Experimento 1

Presentación

En este experimento se verá la diferencia entre los métodos de *GeM* y el de la AFA para determinar el *ranking* de una liga deportiva. Para ello se tomará como parámetro de entrada una tabla con los partidos, sus respectivas fechas, y sus resultados. En la misma, deberá existir un equipo que ocupe una de las primaras posiciones en la tabla de *rankings* que haya perdido contra otro que ocupa una de las últimas. Observaremos la diferencia entre el *ranking* generado por cada uno de los métodos.

Hipótesis

Cuando se juega un partido entre dos equipos, en el método de *GeM* se tendrá en cuenta que tan fuerte son los equipos involucrados y cual fue la diferencia de goles en el partido. En cambio, en el método de la AFA el ganador recibirá 3 puntos sin importar los resultados ni las posiciones que los mismos tenían hasta el momento. Esto puede generar diferencias en el *ranking* ya que cada método considera distinta información para determinar las nuevas posiciones.

Por otro lado, si un equipo que se encuentra en las últimas posiciones de la tabla juega contra uno que está en las primeras y gana, para el método de la AFA es exactamente igual que haya jugado con cualquier otro. Para *GeM*, al tener en cuenta que tan fuertes son los equipos, esto generará que el nuevo vencedor quede en una mejor posición en la tabla. Así, en este método, entre dos fechas se pueden generar saltos, es decir, un equipo puede pasar de tener una muy mala posición en la tabla a estar dentro de los mejores solo por haber ganado un partido contra un equipo fuerte. En el método de la AFA no pasa ya que sólo se le sumarán 3 puntos al equipo triunfador.

Datos de entrada

Resultados del Torneo de Primera División del Fútbol Argentino hasta la Fecha 23. El valor de c es `0.85` y el valor de tolerancia es `0.00001`. Se eligió el valor de c y el de la tolerancia arbitrariamente, procurando que los casos de prueba no fueran excesivamente grandes para no prolongar innecesariamente la duración de las pruebas. .

Resultados

Ranking Liga Deportiva - *GeM*			*Ranking* Liga Deportiva - AFA		
Pos.	Equipo	Puntaje	Pos.	Equipo	Puntaje
1	Boca Juniors	0.080913	1	San Lorenzo	0.054289
2	Aldosivi	0.069879	2	Boca Juniors	0.053203
3	River Plate	0.069065	3	Racing Club	0.049946
4	San Lorenzo	0.058309	4	Rosario Central	0.048860
5	San Martín (SJ)	0.051788	5	River Plate	0.047774
6	Racing Club	0.051674	6	Independiente	0.041260
7	Rosario Central	0.046326	7	Belgrano	0.041260
8	Quilmes	0.045206	8	Estudiantes (LP)	0.041260
9	Newell's Old Boys	0.043581	9	Tigre	0.040174
10	Vélez Sarsfield	0.039828	10	Banfield	0.040174
11	Gimnasia y Esgrima (LP)	0.035738	11	Lanús	0.039088
12	Estudiantes (LP)	0.034629	12	Gimnasia y Esgrima (LP)	0.038002
13	Belgrano	0.033908	13	Quilmes	0.034745
14	Banfield	0.032996	14	San Martín (SJ)	0.034745
15	Unión	0.028015	15	Unión	0.033659
16	Defensa y Justicia	0.027041	16	Tempreley	0.030402
17	Lanús	0.026036	17	Argentinos Juniors	0.028230
18	Independiente	0.023666	18	Newell's Old Boys	0.028230
19	Tigre	0.023078	19	Aldosivi	0.028230
20	Sarmiento	0.022266	20	Vélez Sarsfield	0.027144
21	Olimpo	0.021956	21	Defensa y Justicia	0.026059
22	Crucero del Norte	0.021713	22	Sarmiento	0.026059
23	Arsenal	0.018678	23	Olimpo	0.024973
24	Argentinos Juniors	0.017951	24	Colón	0.024973
25	Tempreley	0.017670	25	Godoy Cruz	0.023887
26	Huracán	0.013740	26	Huracán	0.022801
27	Godoy Cruz	0.013541	27	Atlético de Rafaela	0.021716
28	Atlético de Rafaela	0.011738	28	Arsenal	0.018458
29	Colón	0.011141	29	Nueva Chicago	0.015201
30	Nueva Chicago	0.007928	30	Crucero del Norte	0.015201

Discusión

Como podemos ver en las tablas de *rankings*, los órdenes no son exactamente iguales. Por ejemplo, mirando el equipo 1 (Aldosivi), en el *ranking* de la AFA se encuentra en la posición 20 y cuando usamos el método de *GeM* éste está en la posición 2. Esto se debe a que Aldosivi le ganó tanto al equipo 24 (San Lorenzo) como al 7 (Boca Juniors), que se encontraban primero y segundo en la tabla de posiciones de la AFA, respectivamente. Por esto, en *GeM* sube muchas posiciones mientras que con la AFA sólo se le otorgan 3 puntos por cada uno de los dos partidos.

3.2.2. Experimento 2

Presentación

Con este experimento se busca mostrar como varía el ranking de los equipos a medida que van pasando las fechas de la liga para diferentes valores de c. Para ello se tomará la primer fecha y se medirá el ranking hasta el momento. Luego tomaremos la liga completa y se evaluará el ranking final. Esto se repetirá para cada uno de los valores de c. Luego se compararán las variaciones entre el inicio y el fin de la liga para cada uno de los valores.

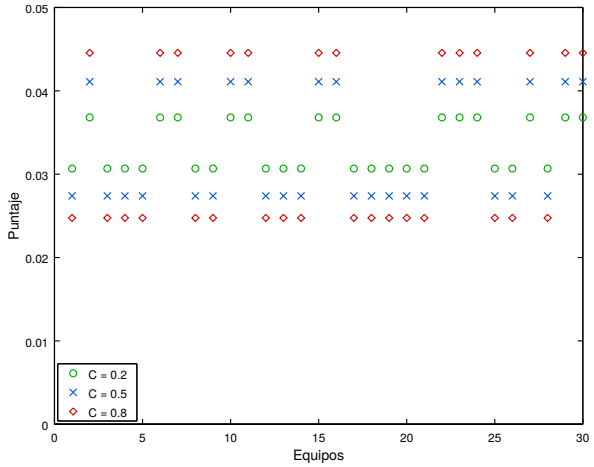
Hipótesis

Suponemos que cuando el grafo es muy poco conexo, es decir, cuando no hay demasiadas conexiones entre los equipos, como es al inicio de la liga, el c cobra relevancia. Por lo tanto, al inicio, para diferentes valores de c va a haber mucha variación en los rankings de los equipos. En cambio, cuando el grafo es mas conexo, es decir, cuando los equipos están muy relacionados entre sí, como sucede al final de la liga, cuando ya se jugaron todas las fechas, la importancia del c disminuye, por lo que suponemos que van a ser muy similares los rankings finales para los diferentes valores de c.

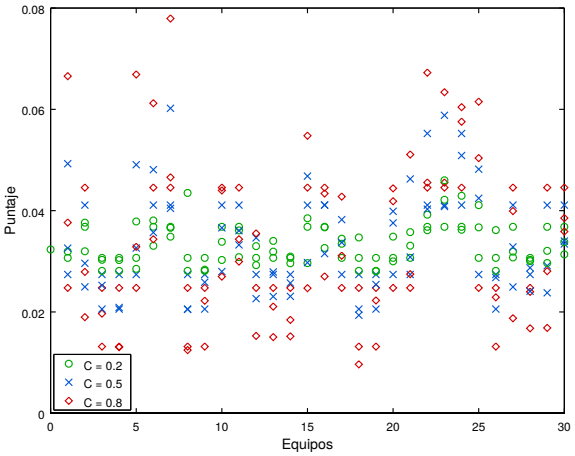
Datos de entrada

Se tomará como valor de tolerancia 0.00001 y los valores de c serán 0.2, 0.5 y 0.9. La liga utilizada se encuentra en el archivo matches.txt. Se eligió el valor de la tolerancia arbitrariamente, procurando que los casos de prueba no fueran excesivamente grandes para no prolongar innecesariamente la duración de las pruebas. Los valores de c elegidos son valores intermedios del total de valores que puede tomar (c no puede ser mayor que 1).

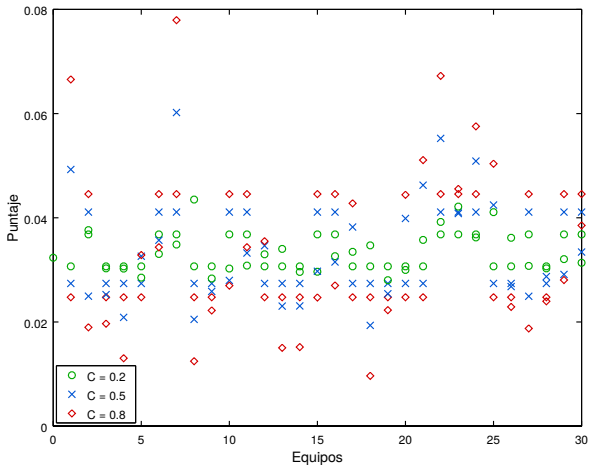
Resultados



1 fecha



5 fechas



23 fechas

Resultados arrojados por el experimento 2. Los gráficos muestran la evolución del *ranking* obtenido mediante *GeM* para la Liga de Primera División de la AFA, según el transcurso de las fechas.

Discusión

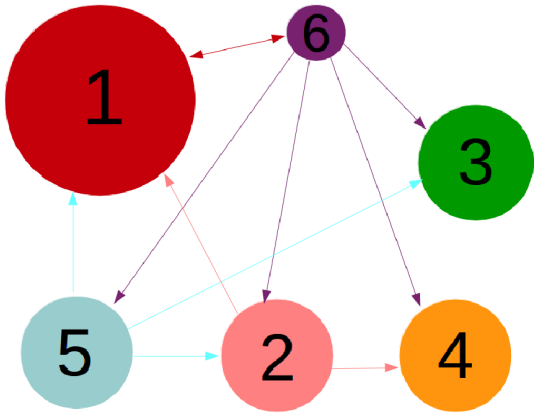
Como se puede observar en los gráficos, en la instancia inicial de la liga los diferentes valores de c generan variaciones mayores entre los puntajes de cada equipo que cuando tomamos el ranking final. Además podemos observar que para valores mas chicos de c, los resultados son más homogéneos. Esto se debe a que al decrementar c, incrementamos (1-c) que es la probabilidad de que un equipo juegue con otro y gane. Ésta probabilidad es para todos los equipos igual. Si la aumentamos, entonces ésta tomará más importancia generando así un sistema en el que los puntajes de los diferentes equipos sean mucho mas cercanos.

3.2.3. Experimento 3: Casos de empate

Presentación

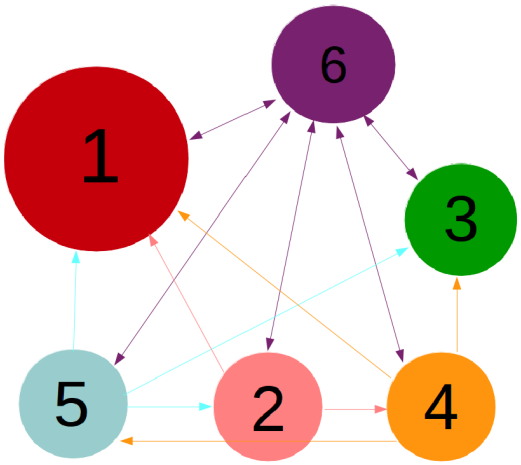
Cuando dos equipos se enfrentan y empatan, podemos tomar diferentes criterios para asignarles el puntaje correspondiente a ese partido. En este experimento, analizaremos tres posibles métodos de asignación de puntaje. En el primero no se le otorgarán puntos a ninguno de los dos equipos. Es decir, empatar y no haber jugado va a valer lo mismo al momento de medir el ranking final de la liga deportiva. En el segundo de los métodos que se analizará, el puntaje que cada equipo recibirá por haber empatado será pasado por parámetro al momento de armar la tabla de puntuaciones. Éste valor será el mismo para los dos equipos y se denominará k.

El objetivo de este experimento es determinar cual de los tres métodos presentados resuelve mejor la situación del empate. Para ello de tomaron diferentes ligas que constan de 6 equipos que juegan entre si a lo largo de 6 fechas. Cada una de ellas presenta diferentes situaciones de empate. En la liga 2 el equipo 1 le gana a todos menos con el 6 que empata. El equipo 6 pierde con todos menos con el 1 que empata. El equipo 2 le gana al 5 y al 3 y pierde con el 4. El 3 le gana al 4 y al 5 y pierde con el 2. El 4 gana contra el 2 y pierde con el 3 y el 5. Por último el 5 gana contra el 4 y pierde contra el 2 y el 3.



Liga 2

En la liga 4 el equipo 1 gana con todos los que juega excepto cuando juega con el 6 que empata. El equipo 6 empata todos. El resto de los equipos quedan iguales que en la liga 2.

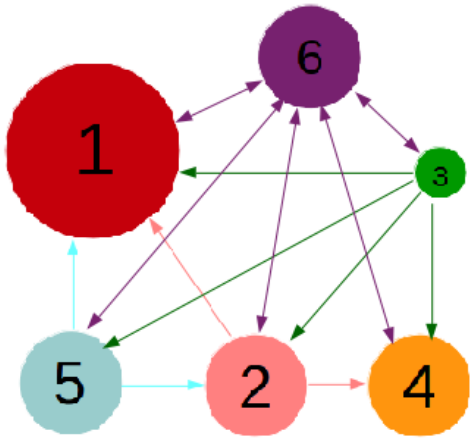


Liga 4

Por último en la liga 5 el equipo 1 gana con todos los que juega excepto cuando juega con el 6 que empata. El equipo 6 empata todos. El equipo 3 pierde con todos los que juega excepto cuando juega contra el 6 que empata. El resto de los equipos quedan iguales que en la liga 2.

Hipótesis

Suponemos que en el método en el que se les da 0 puntos a los equipos empatados lo que va a pasar es que para dos equipos en los cuales uno empata todos los partidos en los que juega y otro que pierde (es indistinto si cuando juegan entre ellos empatan o pierde el primero) ambos quedarán con igual puntaje, lo que es injusto para el equipo que empata. Suponemos que esto se va a poder observar bien en la liga 5, ya que el equipo 3 pierde todos los partidos menos uno que lo empata (cuando juega contra el 6) y el 6 que empata todos. Ambos equipos quedarán



Liga 5

últimos en el ranking final. Además si un equipo fuerte juega con uno débil y estos dos empatan entonces para ambos será lo mismo que no haber jugado, lo que no es justo para el equipa débil. Esto se puede ver en la liga 2 ya que el equipo 1 vence contra todos menos con el 6, con quien empata. El equipo 6 pierde con todos menos con el 1. En el resultado final el equipo 6 obtendrá 0 puntos a pesar de haber empatado con el equipo mas fuerte de la liga.

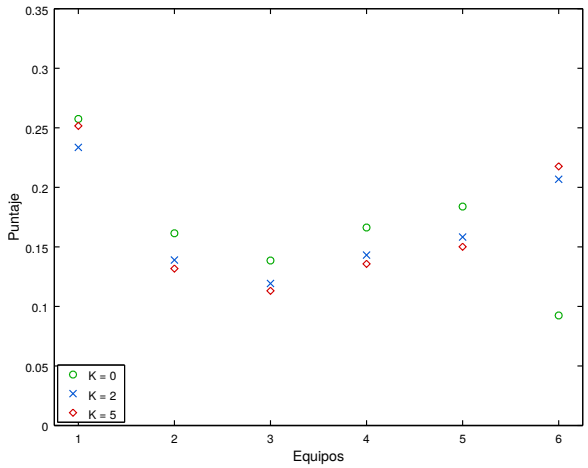
En el método en el cual se le otorga una cantidad de puntos distinta de 0 al empate se supone que cuando un equipo débil le gane a uno fuerte va a subir mucho en el ranking. Esto es así ya que el peso que le da el enlace del equipo fuerte al débil es muy grande. Lo que va a pasar cuando estos equipos jueguen es que para el que lleva la delantera va a ser casi indistinto perder que empatar mientras que para el otro empatar lo favorece demasiado. Esto se puede observar en la liga 2. Suponemos que el equipo 6 (quien perdió todos menos uno que empató) quedará en una posición mucho mejor en el ranking por haber empatado con el 1 (quien gano todos los otros partidos que jugó). Además en este caso veremos que sucede con los rankings a medida que aumentamos el puntaje otorgado a los equipos que empatan. Suponemos que para valores muy grandes del empate puede pasar que sea mas conveniente empatar que ganar. Esto puede suceder ya que para determinar los puntos que se obtienen cuando se gana un partido hay que calcular la diferencia de goles. Si empatar otorga muchos puntos entonces posiblemente la diferencia de goles sea menor que lo que se puede obtener al empatar. Esto se puede observar en la liga 4. Suponemos que para valores de empate muy grandes el equipo 6 (quien empato todos) superará en el ranking final al equipo 1 (quien gano todos).

En el tercer método, el valor de C2 representa la importancia que se le da a la matriz de empate. Suponemos que al aumentar este valor, cuando empaten dos equipos en los cuales uno sea débil y el otro fuerte, la cantidad de posiciones que ascenderá el equipo débil será mayor. Suponemos que con valores de C2 intermedios no ocurrirá que el equipo débil que empató con el fuerte ascienda demasiado en el ranking pero si se le otorgarán puntos por ello. Esto se puede observar en la liga 5 donde suponemos que se observará que empatar es mejor que perder pero peor que ganar y en la liga 2 donde el equipo 6 no va a tomar puna posición en el ranking muy elevada solo por haber vencido a un equipo fuerte pero tampoco quedará con cero puntos.

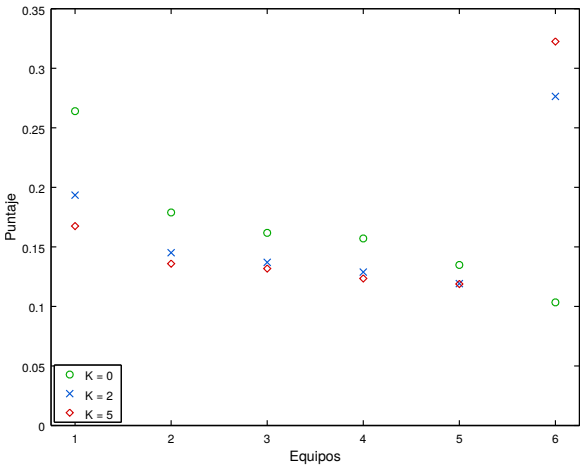
Datos de entrada

Se tomará como valor de tolerancia 0.00001 y el valor de c 0.85. Se eligió el valor de la tolerancia y el de c arbitrariamente, procurando que los casos de prueba no fueran excesivamente grandes para no prolongar innecesariamente la duración de las pruebas. Las 3 ligas utilizadas se encuentran en la carpeta exp3-partidos. Los valores de puntajes que se les darán a los equipos que empaten que se pasarán como parámetro en el segundo método son 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 y 10. Los valores de c que se pasarán para el tercer método serán 0.1, 0.15, 0.2, 0.25, 0.3, 0.35 y 0.4. Los valores de k y c que se observan en los gráficos son los valores más representativos.

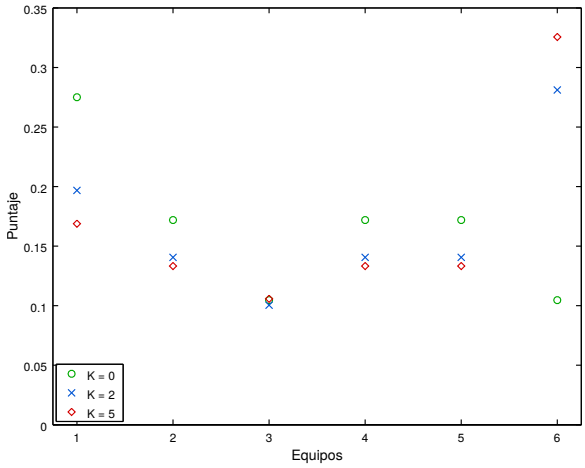
Resultados



Liga 2

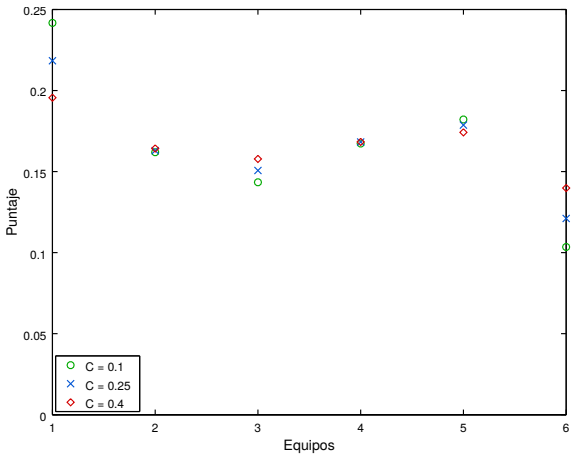


Liga 4

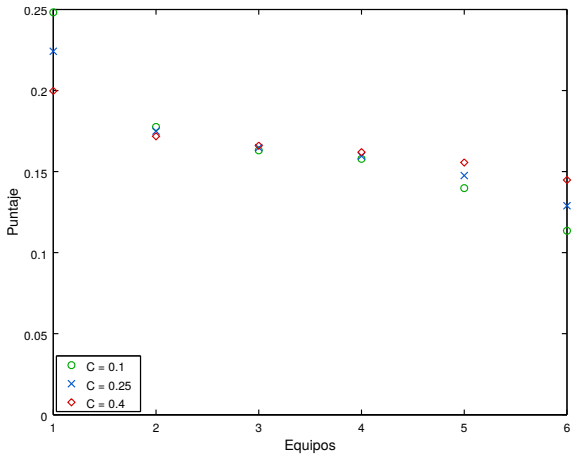


Liga 5

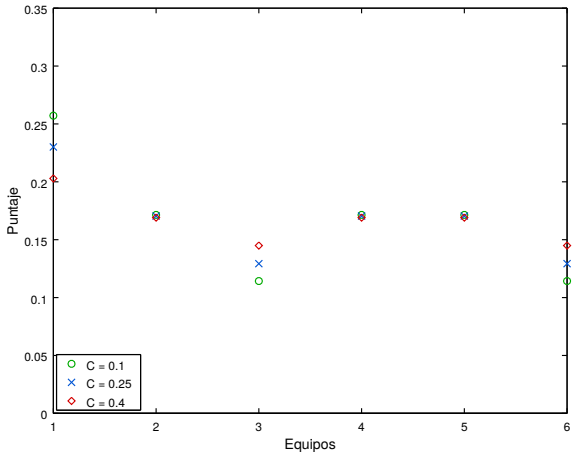
Resultados arrojados por el experimento 2 para el método (2).



Liga 2



Liga 4



Liga 5

Resultados arrojados por el experimento 2 para el método (3).

Discusión

Como se puede ver en el gráfico de la liga 2 modificando el k, el equipo 6 tiene muy poco puntaje cuando el k vale cero ya que al haber perdido todos y tener únicamente un empate, si éste vale cero entonces su puntaje es el mas bajo. Cuando el k aumenta, entonces se le dan puntos por el empate. Como este empate fue con un equipo fuerte entonces deja al equipo 6 muy bien posicionado en la tabla.

En el gráfico de la liga 4 modificando k puede verse como a pesar de que el 1 ser el equipo con mas partidos ganados, al ponerle valores mas altos al puntaje que obtienen los equipos cuando empatan que la diferencia de goles que tiene el equipo mas fuerte, entonces el que empato logra tomar posiciones más altas en la tabla.

En el gráfico de la liga 5 modificando k puede observarse cuando el k vale 0 que los equipos 3 y 6 quedan empatados a pesar de que uno haya perdido en todos y el otro empatado. En cambio cuando se le da puntos a los equipos que empatan el equipo 6 sube mucho en el ranking. Incluso llega a superar al equipo que gano todos los partidos. Dejamos a experimentos futuros

determinar cuál es el valor que deberá tomar k para que no suceda que supere al equipo vencedor.

Cuando usamos el método 3, al que le pasamos por parámetro un C2 que será el nivel de importancia que se le da a la matriz de empate explicada anteriormente, podemos observar que los puntajes que obtienen los equipos que empatan y los que pierden es exactamente el mismo, por lo tanto este método no cumple con el objetivo de lograr que un equipo que empata todos los partidos tenga mejor posición en el ranking que aquel que pierde pero que al mismo tiempo empatar contra un equipo fuerte no genere que el otro ascienda mucho en la tabla de posiciones. Además se puede ver que para los diferentes valores de C2 lo único que se modifica es el puntaje de los equipos sin alterar el ranking. Cuando el C2 toma valores mas grandes se les asigna mayor puntaje a los equipos que empataron, dejando con menos puntos a los que ganaron.

También se puede observar que para los equipos que no empatan con ninguno, la variación del C2 no les afecta. Esto es así ya que el C2 sólo modifica la matriz de empates. En el caso de los equipos que no empataron nunca el valor correspondiente de la matriz para ellos es cero. Al multiplicarlo por el C2 sigue siendo cero. Entonces ésta variación no modifica su puntaje.

Concluimos que el criterio mas acertado de los que analizamos hasta el momento es otorgarle al empate una determinada cantidad de puntos sin que genere que empatar se mejor que ganar pero que tampoco suceda que si un equipo fuerte juega contra uno débil y empatan éste ultimo obtenga cero puntos.

4. Conclusiones

Durante la realización de este trabajo, se pudo observar un ejemplo más de cómo las herramientas teóricas que brinda el álgebra lineal pueden tener aplicaciones sumamente prácticas. El éxito del método de *PageRank* es indiscutible, ya que su gran precisión a la hora de seleccionar resultados de relevancia fue crucial para posicionar al buscador Google por sobre sus competidores y convertirlo en el más utilizado a nivel mundial.

En cuanto a su adaptación para la clasificación en ligas deportivas, *GeM*, se encontraron múltiples razones por las que puede resultar muy efectivo. A diferencia del método utilizado por la AFA, toma en cuenta las posiciones en las que se encuentran los equipos a enfrentarse. Esto genera que si un equipo que se encuentra en una posición del *ranking* muy baja gana un partido con uno que esta entre los primeros, el vencedor subirá más en la tabla que lo que podría subir si le dan solamente una cantidad fija de puntos. Por otro lado en *GeM*, importa cual es la diferencia de goles, en cambio, con el método de la AFA es indistinta.

Una de las características que se observa al utilizar el método *GeM* es que, comparando las posiciones entre dos fechas consecutivas, puede pasar que se genere un salto con algún equipo. Es decir, un equipo puede pasar de estar en las últimas posiciones de la tabla a una de las primeras, lo cual podría generar sorpresas, en ocasiones desagradables.

Utilizando la información que nos brindan los experimentos realizados, podríamos corregir el método *GeM* tomando un buen criterio ante la situación de empate. Por lo que pudimos observar a lo largo del trabajo una buena opción es darle un determinado puntaje a cada uno de los equipos empatados evitando que éste sea demasiado alto y pueda suceder que empatar sea mas conveniente que ganar. Por otro lado, si fuera cero entonces hay determinados casos, como cuando un equipo fuerte empata contra uno débil, en donde no dar puntos puede ser muy injusto para el equipo endeble. En este caso lo que no podemos evitar es que al empatar un equipo débil con uno fuerte el débil ascienda demasiadas posiciones en el ranking de la liga. Por ello se decidió experimentar que sucede si tomamos un parámetro que determine la importancia que se le dará a la situación de empate. En este trabajo notamos que de esta forma empatar y perder genera la misma cantidad de puntos. Dejamos a trabajo futuro determinar el por que de esta situación.

El método de *GeM* tiene la fortaleza de ser robusto ante intentos de especular con los resultados, ya que toma en cuenta no solo si ganan o pierden los equipos si no que se basa en más información para determinar el *ranking*. No obstante, su complejidad también es una de las desventajas que puede atribuirsele: claramente, el método utilizado por la AFA es mas sencillo, y, como las ligas deportivas son vistas por muchas personas, puede generar problemas que el método utilizado para calcular posiciones en la tabla sea extremadamente difícil de comprender.

Apéndices

A. Enunciado del trabajo práctico

Contexto y motivación

A partir de la evolución de Internet durante la década de 1990, el desarrollo de motores de búsqueda se ha convertido en uno de los aspectos centrales para su efectiva utilización. Hoy en día, sitios como Yahoo, Google y Bing ofrecen distintas alternativas para realizar búsquedas complejas dentro de un red que contiene miles de millones de páginas web.

En sus comienzos, una de las características que distinguió a Google respecto de los motores de búsqueda de la época fue la calidad de los resultados obtenidos, mostrando al usuario páginas relevantes a la búsqueda realizada. El esquema general de los orígenes de este motor de búsqueda es brevemente explicado en Brin y Page [1], donde se mencionan aspectos técnicos que van desde la etapa de obtención deinformación de las páginas disponibles en la red, su almacenamiento e indexado y su posterior procesamiento, buscando ordenar cada página de acuerdo a su importancia relativa dentro de la red. El algoritmo utilizado para esta última etapa es denominado PageRank y es uno (no el único) de los criterios utilizados para ponderar la importancia de los resultados de una búsqueda. En este trabajo nos concentraremos en el estudio y desarrollo del algoritmo PageRank.

Por otro lado, las competencias deportivas, en todas sus variantes y disciplinas, requieren casi inevitablemente la comparación entre competidores mediante la confección de *Tablas de Posiciones* y *Rankings* en base a resultados obtenidos en un período de tiempo determinado. Estos ordenamientos de equipos están generalmente (aunque no siempre) basados en reglas relativamente claras y simples, como proporción de victorias sobre partidos jugados o el clásico sistema de puntajes por partidos ganados, empatados y perdidos. Sin embargo, estos métodossimples y conocidos por todos muchas veces no logran capturar la complejidad de la competencia y la comparación. Esto es particularmenteevidente en ligas donde, por ejemplo, todos los equipos no juegan la misma cantidad de veces entre sí.

A modo de ejemplo, la NBA y NFL representan dos ligas con fixtures de temporadas regulares con estas características. Recientemente, el Torneo de Primera División de AFA se suma a este tipo de competencias, ya que la incorporación de la *Fecha de Clásicos* parece ser una interesante idea comercial, pero no tanto desde el punto de vista deportivo ya que cada equipo juega contra su *clásico* más veces que el resto. Como contraparte, éstos rankings son utilizados muchas veces como criterio de decisión, como por ejemplo para determinar la participación en alguna competencia de nivel internacional, con lo cual la confección de los mismos constituye un elemento sensible, afectando intereses deportivos y económicos de gran relevancia.

El problema, Parte I: PageRank y páginas web

El algoritmo PageRank se basa en la construcción del siguiente modelo. Supongamos que tenemos una red con n páginas web $Web = \{1, \ldots, n\}$ donde el objetivo es asignar a cada una de ellas un puntaje que determine la importancia relativa de la misma respecto de las demás. Para modelar las relaciones entre ellas, definimos la *matriz de conectividad* $W \in \{0,1\}^{n \times n}$ de forma tal que $w_{ij} = 1$ si la página j tiene un link a la página i, y $w_{ij} = 0$ en caso contrario. Además, ignoramos los *autolinks*, es decir, links de una página a sí misma, definiendo $w_{ii} = 0$. Tomando

esta matriz, definimos el grado de la página j, n_j, como la cantidad de links salientes hacia otras páginas de la red, donde $n_j = \sum_{i=1}^{n} w_{ij}$. Además, notamos con x_j al puntaje asignado a la página $j \in Web$, que es lo que buscamos calcular.

La importancia de una página puede ser modelada de diferentes formas. Un link de la página $u \in Web$ a la página $v \in Web$ puede ser visto como que v es una página importante. Sin embargo, no queremos que una página obtenga mayor importancia simplemente porque es apuntada desde muchas páginas. Una forma de limitar esto es ponderar los links utilizando la importancia de la página de origen. En otras palabras, pocos links de páginas importantes pueden valer más que muchos links de páginas poco importantes. En particular, consideramos que la importancia de la página v obtenida mediante el link de la página u es proporcional a la importancia de la página u e inversamente proporcional al grado de u. Si la página u contiene n_u links, uno de los cuales apunta a la página v, entonces el aporte de ese link a la página v será x_u/n_u. Luego, sea $L_k \subseteq Web$ el conjunto de páginas que tienen un link a la página k. Para cada página pedimos que

$$x_k = \sum_{j \in L_k} \frac{x_j}{n_j}, \qquad k = 1, \ldots, n. \tag{1}$$

Definimos $P \in \mathbb{R}^{n \times n}$ tal que $p_{ij} = 1/n_j$ si $w_{ij} = 1$, y $p_{ij} = 0$ en caso contrario. Luego, el modelo planteado en (1) es equivalente a encontrar un $x \in \mathbb{R}^n$ tal que $Px = x$, es decir, encontrar (suponiendo que existe) un autovector asociado al autovalor 1 de una matriz cuadrada, tal que $x_i \geq 0$ y $\sum_{i=1}^{n} x_i = 1$. En Bryan y Leise [2] y Kamvar et al. [5, Sección 1] se analizan ciertas condiciones que debe cumplir la red de páginas para garantizar la existencia de este autovector.

Una interpretación equivalente para el problema es considerar al *navegante aleatorio*. Éste empieza en una página cualquiera del conjunto, y luego en cada página j que visita sigue navegando a través de sus links, eligiendo el mismo con probabilidad $1/n_j$. Una situación particular se da cuando la página no tiene links salientes. En ese caso, consideramos que el navegante aleatorio pasa a cualquiera de las página de la red con probabilidad $1/n$. Para representar esta situación, definimos $v \in \mathbb{R}^{n \times n}$, con $v_i = 1/n$ y $d \in \{0, 1\}^n$ donde $d_i = 1$ si $n_i = 0$, y $d_i = 0$ en caso contrario. La nueva matriz de transición es

$$\begin{aligned} D &= v d^t \\ P_1 &= P + D. \end{aligned}$$

Además, consideraremos el caso de que el navegante aleatorio, dado que se encuentra en la página j, decida visitar una página cualquiera del conjunto, independientemente de si esta se encuentra o no referenciada por j (fenómeno conocido como *teletransportación*). Para ello, consideramos que esta decisión se toma con una probabilidad $c \geq 0$, y podemos incluirlo al modelo de la siguiente forma:

$$\begin{aligned} E &= v \bar{1}^t \\ P_2 &= c P_1 + (1 - c) E, \end{aligned}$$

donde $\bar{1} \in \mathbb{R}^n$ es un vector tal que todas sus componentes valen 1. La matriz resultante P_2 corresponde a un enriquecimiento del modelo formulado en (1). Probabilísticamente, la componente

x_j del vector solución (normalizado) del sistema $P_2x = x$ representa la proporción del tiempo que, en el largo plazo, el navegante aleatorio pasa en la página $j \in Web$. Denotaremos con π al vector solución de la ecuación $P_2x = x$, que es comúnmente denominado *estado estacionario*.

En particular, P_2 corresponde a una matriz *estocástica por columnas* que cumple las hipótesis planteadas en Bryan y Leise [2] y Kamvar et al. [5], tal que P_2 tiene un autovector asociado al autovalor 1, los demás autovalores de la matriz cumplen $1 = \lambda_1 > |\lambda_2| \geq \cdots \geq |\lambda_n|$ y, además, la dimensión del autoespacio asociado al autovalor λ_1 es 1. Luego, π puede ser calculada de forma estándar utilizando el método de la potencia.

Una vez calculado el ranking, se retorna al usuario las t páginas con mayor puntaje.

El problema, Parte II: PageRank y ligas deportivas

Existen en la literatura distintos enfoques para abordar el problema de determinar el *ranking* de equipos de una competencia en base a los resultados de un conjunto de partidos. En Govan et al. [4] se hace una breve reseña de dos ellos, y los autores proponen un nuevo método basado en el algoritmo PageRank que denominan GeM[2]. Conceptualmente, el método GeM representa la temporada como un red (grafo) donde las páginas web representan a los equipos, y existe un link (que tiene un valor, llamado peso, asociado) entre dos equipos que los relaciona modelando los resultados de los posibles enfrentamientos entre ellos. En base a este modelo, Govan et al. [4] proponen calcular el ranking de la misma forma que en el caso de las páginas web.

En su versión básica, que es la que consideraremos en el presente trabajo, el método GeM (ver, e.g., [4, Sección GeM Ranking Method]) es el siguiente[3]:

1. La temporada se representa mediante un grafo donde cada equipo representa un nodo y existe un link de i a j si el equipo i perdió al menos una vez con el equipo j.

2. Se define la matriz $A^t \in \mathbb{R}^{n\times n}$

$$A^t_{ji} = \begin{cases} w_{ji} & \text{si el equipo } i \text{ perdió con el equipo } j, \\ 0 & \text{en caso contrario,} \end{cases}$$

   donde w_{ji} es la diferencia absoluta en el marcador. En caso de que i pierda más de una vez con j, w_{ji} representa la suma acumulada de diferencias. Notar que A^t es una generalización de la matriz de conectividad W definida en la sección anterior.

3. Definir la matriz $H^t_{ji} \in \mathbb{R}^{n\times n}$ como

$$H^t_{ji} = \begin{cases} A^t_{ji} / \sum_{k=1}^{n} A^t_{ki} & \text{si hay un link } i \text{ a } j, \\ 0 & \text{en caso contrario.} \end{cases}$$

4. Tomar $P = H^t$, y aplicar el método PageRank como fue definido previamente, siendo π la solución a la ecuación $P_2x = x$. Notar que los páginas sin links salientes, en este contexto se corresponden con aquellos equipos que se encuentran invictos.

5. Utilizar los puntajes obtenidos en π para ordenar los equipos.

[2] Aunque no se especifica, asumimos que el nombre se debe a las iniciales de los autores.

[3] Notar que en artículo, Govan et al. [4] lo definen sobre la traspuesta. La definición y las cuentas son equivalentes, simplemente se modifica para mantener la consistencia a lo largo del enunciado.

En función del contexto planteado previamente, el método GeM define una estructura que relaciona equipos dependiendo de los resultados parciales y obtener un ranking utilizando solamente esta información.

Enunciado

El objetivo del trabajo es experimentar en el contexto planteado utilizando el algoritmo PageRank con las variantes propuestas. A su vez, se busca comparar los resultados obtenidos cualitativa y cuantitativamente con los algoritmos tradicionales utilizados en cada uno de los contextos planteados. Los métodos a implementar (como mínimo) en ambos contexto planteados por el trabajo son los siguientes:

1. *Búsqueda de páginas web:* PageRank e In-deg, éste último consiste en definir el ranking de las páginas utilizando solamente la cantidad de ejes entrantes a cada una de ellas, ordenándolos en forma decreciente.

2. *Rankings en competencias deportivas:* GeM y al menos un método estándar propuesto por el grupo (ordenar por victorias/derrotas, puntaje por ganado/empatado/perdido, etc.) en función del deporte(s) considerado(s).

El contexto considerado en 1., en la búsqueda de páginas web, representa un desafío no sólo desde el modelado, si no también desde el punto de vista computacional considerando la dimensión de la información y los datos a procesar. Luego, dentro de nuestras posibilidades, consideramos un entorno que simule el contexto real de aplicación donde se abordan instancias de gran escala (es decir, n, el número total de páginas, es grande). Para el desarrollo de PageRank, se pide entonces considerar el trabajo de Bryan y Leise [2] donde se explica la intución y algunos detalles técnicos respecto a PageRank. Además, en Kamvar et al. [5] se propone una mejora del mismo. Si bien esta mejora queda fuera de los alcances del trabajo, en la Sección 1 se presenta una buena formulación del algoritmo. En base a su definición, P_2 no es una matriz esparsa. Sin embargo, en Kamvar et al. [5, Algoritmo 1] se propone una forma alternativa para computar $x^{(k+1)} = P_2 x^{(k)}$. Este resultado debe ser utilizado para mejorar el almacenamiento de los datos.

En la práctica, el grafo que representa la red de páginas suele ser esparso, es decir, una página posee relativamente pocos links de salida comparada con el número total de páginas. A su vez, dado que n tiende a ser un número muy grande, es importante tener en cuenta este hecho a la hora de definir las estructuras de datos a utilizar. Luego, desde el punto de vista de implementación se pide utilizar alguna de las siguientes estructuras de datos para la representación de las matrices esparsas: *Dictionary of Keys* (dok), *Compressed Sparse Row* (CSR) o *Compressed Sparse Column* (CSC). Se deberá incluir una justificación respecto a la elección que consdiere el contexto de aplicación. Además, para PageRank se debe implementar el método de la potencia para calcular el autovector principal. Esta implementación debe ser realizada íntegramente en C++.

En función de la experimentación, se deberá realizar un estudio particular para cada algoritmo (tanto en términos de comportamiento del mismo, como una evaluación de los resultados obtenidos) y luego se procederá a comparar cualitativamente los rankings generados. La experimentación deberá incluir como mínimo los siguientes experimentos:

1. Estudiar la convergencia de PageRank, analizando la evolución de la norma Manhattan

(norma L_1) entre dos iteraciones sucesivas. Comparar los resultados obtenidos para al menos dos instancias de tamaño mediano-grande, variando el valor de c.

2. Estudiar el tiempo de cómputo requerido por PageRank.

3. Para cada algoritmo, proponer ejemplos de tamaño pequeño que ilustren el comportamiento esperado (puede ser utilizando las herramientas provistas por la cátedra o bien generadas por el grupo).

Puntos opcionales:

1. Demostrar que los pasos del Algoritmo 1 propuesto en Kamvar et al. [5] son correctos y computan P_2x.

2. Establecer una relación con la proporción entre $\lambda_1 = 1$ y $|\lambda_2|$ para la convergencia de PageRank.

El segundo contexto de aplicación no presenta mayores desafíos desde la perspectiva computacional, ya que en el peor de los casos una liga no suele tener mas que unas pocas decenas de equipos. Más aún, es de esperar que en general la matriz que se obtiene no sea esparsa, ya que probablemente un equipo juegue contra un número significativo de contrincantes. Sin embargo, la popularidad y sensibilidad del problema planteado requieren de un estudio detallado y pormenorizado de la calidad de los resultados obtenidos. El objetivo en este segundo caso de estudio es puramente experimental.

En función de la implementación, aún cuando no represente la mejor opción, es posible reutilizar y adaptar el desarrollo realizado para páginas web. También es posible realizar una nueva implementación desde cero, simplificando la operatoria y las estructuras, en C++, MATLAB o PYTHON.

La experimentación debe ser realizada con cuidado, analizando (y, eventualmente, modificando) el modelo de GeM:

1. Considerar al menos un conjunto de datos reales, con los resultados de cada fecha para alguna liga de alguún deporte.

2. Notar que el método GeM asume que no se producen empates entre los equipos (o que si se producen, son poco frecuentes). En caso de considerar un deporte donde el empate se da con cierta frecuencia no despreciable (por ejemplo, fútbol), es fundamental aclarar como se refleja esto en el modelo y analizar su eventual impacto.

3. Realizar experimentos variando el parámetro c, indicando como impacta en los resultados. Analizar la evolución del ranking de los equipos a través del tiempo, evaluando también la evolución de los rankings e identificar características/hechos particulares que puedan ser determinantes para el modelo, si es que existe alguno.

4. Comparar los resultados obtenidos con los reales de la liga utilizando el sistema estándar para la misma.

Puntos opcionales:

1. Proponer (al menos) dos formas alternativas de modelar el empate entre equipos en GeM.

Parámetros y formato de archivos

El programa deberá tomar por línea de comandos dos parámetros. El primero de ellos contendrá la información del experimento, incluyendo el método a ejecutar (`alg`, 0 para PageRank, 1 para el método alternativo), la probabilidad de teletransportación c, el tipo de instancia (0 páginas web, 1 deportes), el *path* al archivo/directorio conteniendo la definición de la red (que debe ser relativa al ejecutable, o el path absoluto al archivo) y el valor de tolerancia utilizado en el criterio de parada del método de la potencia.

El siguiente ejemplo muestra un caso donde se pide ejecutar PageRank, con una probabilidad de teletransportación de 0.85, sobre la red descripta en `test1.txt` (que se encuentra en el directorio `tests/`), correspondiente a una instancia de ranking aplicado a deportes y con una tolerancia de corte de 0,0001.

```
0 0.85 1 tests/red-1.txt 0.0001
```

Para la definición del grafo que representa la red, se consideran dos bases de datos de instancias con sus correspondientes formatos. La primera de ellas es el conjunto provisto en SNAP [6] (el tipo de instancia es 0), con redes de tamaño grande obtenidos a partir de datos reales. Además, se consideran las instancias que se forman a partir de resultados de partidos entre equipos, para algún deporte elegido por el grupo.

En el caso de la base de SNAP, los archivos contiene primero cuatro líneas con información sobre la instancia (entre ellas, n y la cantidad total de links, m) y luego m líneas con los pares i, j indicando que i apunta a j. A modo de ejemplo, a continuación se muestra el archivo de entrada correspondiente a la red propuesta en Bryan y Leise [2, Figura 1]:

```
# Directed graph (each unordered pair of nodes is saved once):
# Example shown in Bryan and Leise.
# Nodes: 4 Edges: 8
# FromNodeId    ToNodeId
1   2
1   3
1   4
2   3
2   4
3   1
4   1
4   3
```

Para el caso de rankings en ligas deportivas, el archivo contiene primero una línea con información sobre la cantidad de equipos (n), y la cantidad de partidos totales a considerar (k). Luego, siguen k lńeas donde cada una de ellas representa un partido y contiene la siguiente información: número de fecha (es un dato opcional al problema, pero que puede ayudar a la hora de experimentar), equipo i, goles equipo i, equipo j, goles equipo j. A continuación se muestra el archivo de entrada con la información del ejemplo utilizado en Govan et al. [4]:

```
6 10
```

```
1 1 16 4 13
1 2 38 5 17
1 2 28 6 23
1 3 34 1 21
1 3 23 4 10
1 4 31 1 6
1 5 33 6 25
1 5 38 4 23
1 6 27 2 6
1 6 20 5 12
```

Es importante destacar que, en este último caso, los equipos son identificados mediante un número. Opcionalmente podrá considerarse un archivo que contenga, para cada equipo, cuál es el código con el que se lo identifica.

Una vez ejecutado el algoritmo, el programa deberá generar un archivo de salida que contenga una línea por cada página (n líneas en total), acompañada del puntaje obtenido por el algoritmo PageRank/In-deg/método alternativo.

Para generar instancias de páginas web, es posible utilizar el código Python provisto por la cátedra. La utilización del mismo se encuentra descripta en el archivo README. Es importante mencionar que, para que el mismo funcione, es necesario tener acceso a Internet. En caso de encontrar un bug en el mismo, por favor contactar a los docentes de la materia a través de la lista. Desde ya, el código puede ser modificado por los respectivos grupos agregando todas aquellas funcionalidades que consideren necesarias.

Para instancias correspondientes a resultados entre equipos, la cátedra provee un conjunto de archivos con los resultados del Torneo de Primera División del Fútbol Argentino hasta la Fecha 23. Es importante aclarar que los dos partidos suspendidos, River - Defensa y Justicia y Racing - Godoy Cruz han sido arbitrariamente completados con un resultado inventado, para simplificar la instancia. En función de datos reales, una alternativa es considerar el repositorio DataHub [3], que contiene información estadística y resultados para distintas ligas y deportes de todo el mundo.

Fechas de entrega

- *Formato Electrónico:* Martes 6 de Octubre de 2015, 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 `[TP2]` seguido de la lista de apellidos de los integrantes del grupo.
- *Formato físico:* Miércoles 7 de Octubre de 2015, a las 18 hs. en la clase práctica.

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

Referencias

[1] Sergey Brin y Lawrence Page. "The anatomy of a large-scale hypertextual Web search engine". En: *Computer Networks and ISDN Systems* 30.1-7 (abr. de 1998), págs. 107-117. ISSN: 01697552. DOI: 10.1016/S0169-7552(98)00110-X. URL: http://linkinghub.elsevier.com/retrieve/pii/S016975529800110X.

[2] Kurt Bryan y Tanya Leise. "The Linear Algebra behind Google". En: *SIAM Review* 48.3 (2006), págs. 569-581.

[3] *DataHub*. http://datahub.io.

[4] Angela Y. Govan, Carl D. Meyer y Rusell Albright. "Generalizing Google's PageRank to Rank National Football League Teams". En: *Proceedings of SAS Global Forum 2008*. 2008.

[5] Sepandar D. Kamvar y col. "Extrapolation methods for accelerating PageRank computations". En: *Proceedings of the 12th international conference on World Wide Web*. WWW '03. Budapest, Hungary: ACM, 2003, págs. 261-270. ISBN: 1-58113-680-3. DOI: 10.1145/775152.775190. URL: http://doi.acm.org/10.1145/775152.775190.

[6] *Stanford Large Network Dataset Collection*. http://snap.stanford.edu/data/\#web.