

## Hoja de ejercicios Map Reduce

25 de octubre de 2023

HIBD

Año 2023/2024

Máster TECI

Facultad de CC.

Matemáticas

### ▷ 1. Grado de los vértices de un grafo

Los grafos son estructuras que sirven para representar relaciones (aristas) entre objetos (vértices). Los grafos se utilizan como modelo en multitud de áreas y existe una extensa teoría que proporciona interesantes resultados y algoritmos. Los grafos pueden representarse de muchas formas diferentes, quizás la representación más conocida es la que utiliza una matriz de adyacencia. Sin embargo, en este ejercicio vamos a considerar que un grafo está representados como una listas de aristas.

**Simplificar** Estamos pensando en trabajar con grafos (más bien grandes), de los que obtenemos las aristas (relaciones entre objetos) de una forma distribuida y quizás propensa a errores o duplicaciones. Un primer problema que tenemos que resolver es la depuración y simplificación de los datos.

Suponemos que la lista de aristas original puede tener repeticiones, cambios de orden y ciclos en un vértice. Queremos generar una lista de aristas con las siguientes propiedades:

- No hay aristas repetidas.
- Los aristas muestran los nodos ordenados.
- No hay ciclos en los vértices.

Escribe un programa con el esquema map-reduce que resuelva el problema de simplificar una lista de aristas con respecto a los criterios anteriores.

**Ejemplo** Si la lista original es  $\{(B, B), (B, A), (C, A), (A, B), (A, D), (B, C)\}$ , un resultado podría que ser  $\{(A, B), (B, C), (A, D), (A, C)\}$ . Observa que puede haber distintas soluciones dependiendo del orden que se elija para clasificar los nodos.

**Grado de los vértices** Una medida que puede asignarse a los vértices es la cantidad de aristas a las que pertenece, lo que se conoce como el *grado* de un vértice.

**Ejemplo** Si la lista original es  $\{(B, B), (B, A), (C, A), (A, B), (A, D), (B, C)\}$ , y su versión simplificada es  $\{(A, B), (B, C), (A, D), (A, C)\}$ , el resultado tendría que ser:

$$\{(A, 3), (B, 2), (C, 2), (D, 1)\}$$

**Grado de los vértices agregado por arista** Escribe un programa con el esquema map-reduce que devuelva el grado de los vértices pero agregado por aristas. El formato de

la salida del algoritmo tiene que ser la lista de aristas con la información del grado de cada uno de los nodos de la arista:

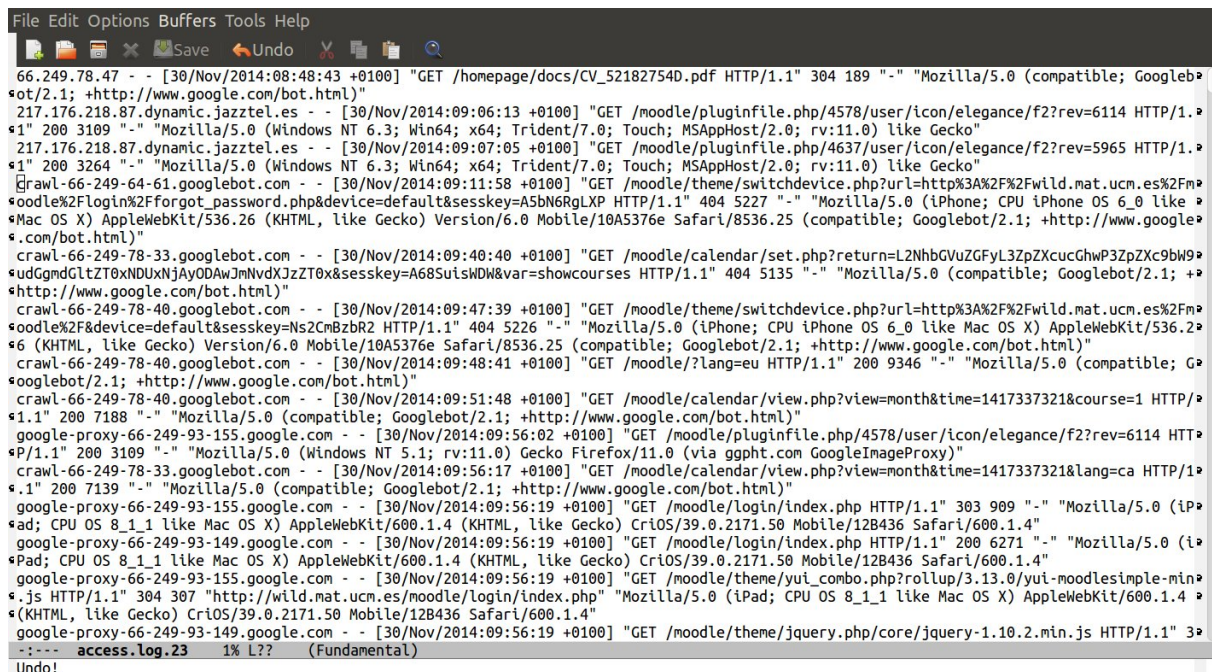
$$(\text{nodo}_1, \text{nodo}_2, \text{grado del nodo}_1, \text{grado del nodo}_2)$$

**Ejemplo** Si la lista original es  $\{(B, B), (B, A), (C, A), (A, B), (A, D), (B, C)\}$ , y su versión simplificada es  $\{(A, B), (B, C), (A, D), (A, C)\}$ , el resultado tendría que ser:

$$\{(A, B, 3, 2), (B, C, 2, 2), (A, D, 3, 1), (A, C, 3, 2)\}$$

## ▷ 2. Análisis de ficheros log de acceso a páginas web

Toda la actividad que realizan los usuarios en un servidor web queda reflejada en sus ficheros *log*. Estos ficheros guardan información textual como la que aparece en la figura siguiente:



```

File Edit Options Buffers Tools Help
66.249.78.47 - - [30/Nov/2014:08:48:43 +0100] "GET /homepage/docs/CV_52182754D.pdf HTTP/1.1" 304 189 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
217.176.218.87.dynamic.jazztel.es - - [30/Nov/2014:09:06:13 +0100] "GET /moodle/pluginfile.php/4578/user/icon/elegance/f2?rev=6114 HTTP/1.1" 200 3109 "-" "Mozilla/5.0 (Windows NT 6.3; Win64; x64; Trident/7.0; Touch; MSAppHost/2.0; rv:11.0) like Gecko"
217.176.218.87.dynamic.jazztel.es - - [30/Nov/2014:09:07:05 +0100] "GET /moodle/pluginfile.php/4637/user/icon/elegance/f2?rev=5965 HTTP/1.1" 200 3264 "-" "Mozilla/5.0 (Windows NT 6.3; Win64; x64; Trident/7.0; Touch; MSAppHost/2.0; rv:11.0) like Gecko"
crawl-66-249-64-61.googlebot.com - - [30/Nov/2014:09:11:58 +0100] "GET /moodle/theme/switchdevice.php?url=http%3A%2F%2Fwild.mat.ucm.es%2Fmoodle%2Flogin%2Fforgot_password.php&device=default&sesskey=A5bN6RgLP HTTP/1.1" 404 5227 "-" "Mozilla/5.0 (iPhone; CPU iPhone OS 6_0 like Mac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A5376e Safari/8536.25 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
crawl-66-249-78-33.googlebot.com - - [30/Nov/2014:09:40:40 +0100] "GET /moodle/calendar/set.php?return=L2NhbGVuZGFyL3ZpZXcucGhwP3ZpZXc9bW9udGmGdGltZT0xNDUxNjAyODAwJmNvdXJzZT0x&sesskey=A685uIsWDW&var=showcourses HTTP/1.1" 404 5135 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
crawl-66-249-78-40.googlebot.com - - [30/Nov/2014:09:47:39 +0100] "GET /moodle/theme/switchdevice.php?url=http%3A%2F%2Fwild.mat.ucm.es%2Fmoodle%2Fdevice=default&sesskey=A5bN6RgLP HTTP/1.1" 404 5226 "-" "Mozilla/5.0 (iPhone; CPU iPhone OS 6_0 like Mac OS X) AppleWebKit/536.26 (KHTML, like Gecko) Version/6.0 Mobile/10A5376e Safari/8536.25 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
crawl-66-249-78-40.googlebot.com - - [30/Nov/2014:09:48:41 +0100] "GET /moodle/?lang=eu HTTP/1.1" 200 9346 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
crawl-66-249-78-40.googlebot.com - - [30/Nov/2014:09:51:48 +0100] "GET /moodle/calendar/view.php?view=month&time=1417337321&course=1 HTTP/1.1" 200 7188 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
google-proxy-66-249-93-155.google.com - - [30/Nov/2014:09:56:02 +0100] "GET /moodle/pluginfile.php/4578/user/icon/elegance/f2?rev=6114 HTTP/1.1" 200 3109 "-" "Mozilla/5.0 (Windows NT 5.1; rv:11.0) Gecko Firefox/11.0 (via ggpht.com GoogleImageProxy)"
crawl-66-249-78-33.googlebot.com - - [30/Nov/2014:09:56:17 +0100] "GET /moodle/calendar/view.php?view=month&time=1417337321&lang=ca HTTP/1.1" 200 7139 "-" "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
google-proxy-66-249-93-155.google.com - - [30/Nov/2014:09:56:19 +0100] "GET /moodle/login/index.php HTTP/1.1" 303 909 "-" "Mozilla/5.0 (iPad; CPU OS 8_1_1 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) CriOS/39.0.2171.50 Mobile/12B436 Safari/600.1.4"
google-proxy-66-249-93-149.google.com - - [30/Nov/2014:09:56:19 +0100] "GET /moodle/login/index.php HTTP/1.1" 200 6271 "-" "Mozilla/5.0 (iPad; CPU OS 8_1_1 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) CriOS/39.0.2171.50 Mobile/12B436 Safari/600.1.4"
google-proxy-66-249-93-155.google.com - - [30/Nov/2014:09:56:19 +0100] "GET /moodle/theme/yui_combo.php?rollup/3.13.0/yui-moodlesimple-min.js HTTP/1.1" 304 307 "http://wild.mat.ucm.es/moodle/login/index.php" "Mozilla/5.0 (iPad; CPU OS 8_1_1 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) CriOS/39.0.2171.50 Mobile/12B436 Safari/600.1.4"
google-proxy-66-249-93-149.google.com - - [30/Nov/2014:09:56:19 +0100] "GET /moodle/theme/jquery.php/core/jquery-1.10.2.min.js HTTP/1.1" 304 2222 "http://wild.mat.ucm.es/moodle/theme/jquery.php/core/jquery-1.10.2.min.js" "Mozilla/5.0 (iPad; CPU OS 8_1_1 like Mac OS X) AppleWebKit/600.1.4 (KHTML, like Gecko) CriOS/39.0.2171.50 Mobile/12B436 Safari/600.1.4"
:-- access.log.23 1% L?? (Fundamental)
Undo!

```

Figura 1: Fragmento de un fichero log.

Mientras que un libro suele tener entre 4000 y 6000 líneas, en un solo día, un servidor apache de un sitio 'modesto' como es `wild.mat.ucm.es`, puede generar decenas de miles de líneas. De forma simplificada, podemos decir que por cada petición que hace un usuario del servidor, se registra una línea que contiene toda la información relacionada con la petición.

Los ficheros log tienen un formato muy concreto y son muy fácilmente legibles (aunque mirando la imagen anterior cueste creerlo!). Miremos una única línea:

147.96.18.203 - - [27/Apr/2015:10:15:32 +0200] "GET /moodle/ HTTP/1.1" 200 8944 Mozilla/5.0 (X11; Ubuntu; Linux x86\_64; rv:34.0) Gecko/20100101 Firefox/34.0"

La información se estructura por columnas. Vamos a comentar algunas columnas que serán de nuestro interés, para una información más detallada puede consultarse la documentación oficial (<http://httpd.apache.org/docs/1.3/logs.html#accesslog>).

- En la primera columna aparece la dirección ip del cliente, lo que llamaremos usuarios. En el ejemplo anterior 147.96.18.203.
- En la cuarta columna aparece la fecha y la hora en la que la petición de página se ha realizado, [27/Apr/2015:10:15:32 +0200].
- En la quinta columna aparece la petición de página concreta que el usuario ha hecho, "GET /moodle/ HTTP/1.1". De esta petición lo que más nos interesa es la página pedida, en este caso /moodle/.
- En la sexta columna aparece un código de respuesta, que indica si se ha podido responder a la petición del usuario, si la página no existe, etc. (véase, por ejemplo, [http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos\\_de\\_estado\\_HTTP](http://es.wikipedia.org/wiki/Anexo:C%C3%B3digos_de_estado_HTTP)). En el ejemplo anterior el código de respuesta es 200, que indica que la página solicitada ha sido enviada sin ningún problema.

El análisis de log es una actividad que interesa a muy diversos niveles: identificar las páginas más visitadas, la carga del servidor en las distintas horas del día, los navegadores utilizados por los usuarios... Vamos a hacerte algunas propuestas concretas:

**Accesos totales a páginas web** En primer lugar nos gustaría conocer qué páginas de nuestro servidor son las más accedidas. Escribe un programa paralelo que nos diga cuántos accesos exitosos (código de respuesta 200) ha tenido cada página.

**Errores en accesos a páginas web** Escribe un programa paralelo que nos diga cuántos accesos erróneos (código de respuesta distinto a 200) ha habido cada día.

**Usuarios por cada página web** Contar el número total de accesos de cada página es importante, pero también es importante la variedad, es decir, el número de usuarios diferentes que visitaron la página. Suponiendo que cada usuario queda identificado por una dirección ip, escribe un programa paralelo que nos diga el número de usuarios diferentes que ha tenido cada una de las páginas.

**Carga del servidor por fechas** Para empezar, se quiere conocer el número total de páginas que se sirven cada día. Escribe un programa paralelo que, para cada día, devuelva el número total de accesos exitosos (código de respuesta 200) a las páginas del servidor.

**Usuarios en un día** Contar el número total de accesos de cada página en un día es interesante, pero también es interesante saber el número de usuarios por día. Suponiendo que cada usuario queda identificado por una dirección ip, escribe un programa paralelo que nos diga el número de usuarios diferentes que ha habido cada día.

**Ranking de usuarios** Suponiendo que cada usuario queda identificado por una dirección ip, escribe un programa paralelo que nos diga los  $k$  usuarios que más accesos exitosos a páginas hayan realizado.

**Comportamiento por días** Supongamos que estamos interesados en analizar el *comportamiento* de los usuarios por días. Para cada usuario, nos gustaría saber qué grupos de páginas visita, con qué frecuencia se conecta, si tiene un patrón definido de navegación, si este patrón es parecido al de otros usuarios... Realmente no hace falta mucha imaginación para pensar posibles aplicaciones de un análisis de estos resultados.

El comportamiento de un usuario puede definirse con muy diversos niveles de detalle. Nosotros vamos a considerar simplemente que el comportamiento durante un día es el conjunto de páginas web que solicita.

Escribe un programa paralelo que, a partir de unos ficheros log, devuelva los comportamientos de cada usuario (el conjunto de páginas visitadas por día). Es decir, para cada usuario y cada fecha, el programa nos informe de las páginas que ha visitado.

**Comportamiento por días** Para seguir con el estudio de los comportamientos de usuarios, nos gustaría aglutinar la información y para cada usuario tener todos los comportamientos que hemos encontrado en los ficheros log analizados.

**Comportamiento por días** Como último detalle del estudio de los comportamientos, nos gustaría saber para cada usuario, las veces que cada comportamiento se repite.