Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Desarrollo de Aplicaciones Distribuidas
Semestre I-2016

Informe de Análisis sobre Resultados
Fase 3

Integrantes:

- **David Fernández C.I. 24.313.648**

Problema

Se requiere realizar un contador de palabras implementando las funciones Map y Reduce de manera que las mismas puedan ser utilizadas en entornos distribuidos.

Se pueden ingresar datos a la aplicación mediante un archivo de texto sin formato ó a través de un campo de texto multilínea (textarea). La palabra a buscar se ingresará a través de un campo de texto de una línea (input).

Análisis de los Resultados Obtenidos

Después de haber realizado la implementación de la solución propuesta en el análisis de la fase, se obtuvieron unos resultados satisfactorios al ejecutar la aplicación.

Se implementó una interfaz principal (**Home.html**) en la cual primero hay que seleccionar la forma en la que se van a ingresar los datos (mediante 2 radiobutton), y luego aparecerá ya sea un campo de texto si se seleccionó la opción de **Escribir en campo de texto** o un botón para subir un archivo si se seleccionó la otra opción (**Subir archivo .txt**).

Al haber hecho lo descrito anteriormente lo único que falta es ingresar en el campo de texto la palabra que hay que buscar y darle al botón de **Realizar Búsqueda.**

La información suministrada en los campos del formulario son obtenidos mediante las funciones PHP respectivas (por ejemplo **$_POST['texto']**) y luego a cada información se le pasa la función **probar(),** la cuál elimina todos esos caracteres innecesarios y previene la SQL Injection.

Dependiendo del método de entrada (**campo de texto** o **subida de archivo**) se realizará lo siguiente:

- **Campo de texto:** se llama a la función **map_text()** y se le pasa por parámetro la entrada recibida. Dentro primero se llama a una función **quitar()** la cual elimina tanto las comas, como otros signos de puntuación. Luego la función se encarga de hacer el **map**, es decir; que agarra cada palabra y la convierte en una tupla **(pal , 1)**. Cada una de estas tuplas se le pasa a la función **reduce()**, que finalmente agrupa cada una de las tupla y las suma dependiendo de cuantas sean iguales. El resultado de esto se guarda en un arreglo asociativo que contendrá en el **índice k** la cantidad de apariciones de la **palabra k** en el texto. El resultado es mostrado en el campo de texto de respuesta.

- **Subida de archivo:** en este caso primero copiamos el **archivo .txt** al servidor y lo abrimos mediante **fopen()** con modo de solo lectura. Se invoca a la función **map()** que va a realizar lo mismo que en el método anterior, solo que se va leyendo línea por línea el archivo de entrada. Se obtienen las tuplas y son pasadas a la función **reduce()** que las agrupa y las suma. El resultado de esto se guarda en un arreglo asociativo que contendrá en el **índice k** la cantidad de apariciones de la **palabra k** en el texto. El resultado es mostrado en el campo de texto de respuesta.

En cuanto a la conexión a la base de datos, existe un archivo PHP llamado **dba.php** en el cual se pueden hacer las modificaciones respectivas dependiendo de el manejador de base de datos que se necesite usar.

Finalmente se puede observar que los resultados obtenidos y la implementación realizada concuerdan con el análisis que se hizo previamente.