Universidad Simón Bolívar
Departamento de Computación y
Tecnología de la Información
CI-3825: Sistemas de Operación I

Prof. Fernando Torre Mora
Prof. Carlos Gómez
Prof. Fernando Lovera

Proyecto 2 Enero – Marzo 2018

Buscador-indizador de archivos

1 Introducción

Cuando un usuario utiliza actualmente una función de búsqueda en un sistema operativo, este programa en realidad revisa la palabra buscada en un índice, no directamente en el sistema de archivos. El índice almacena correspondencias entre palabras y rutas (por ejemplo, almacena "tarea" correspondiente a "/home/operativos/tarea.tar.gz" y "/home/basedatos/tarea1.tar.gz") por lo que generalmente permite dar una respuesta a la solicitud del usuario. Sin embargo, esto requiere dos programas separados: un indizador y un buscador; y el buscador sólo puede encontrar los archivos que fueron creados antes de que se corriera por última vez al indizador (el cual puede ejecutarse de manera muy infrecuente).

A la búsqueda de la siguiente gran innovación, la Fundación Linux propone un nuevo buscador de archivos que, a diferencia de los buscadores actuales, actualizaría el índice cada vez que es llamado. Esto permitiría que cada llamada sucesiva se más rápida, dando mejor servicio al usuario, y evitando dedicar recursos del sistema a indizar cuando el usuario podría estarlos necesitando para correr procesos usuario.

2 Características del indizador

El buscador-indizador propuesto buscaría el término de búsqueda en su índice, cargado desde un archivo; y, en un hilo paralelo, buscaría en el directorio desde el que fue llamado el programa, y sus subdirectorios. Si un archivo es encontrado que no ha sido agregado al índice, el programa deberá agregarlo en los términos de búsqueda correspondientes y luego, actualizar el archivo de índice. Si uno de los términos coincide con el término de búsqueda, el programa imprime la ruta completa de este archivo junto con las rutas completas de los archivos que resultaron de revisar el índice. Las rutas de archivo deben ir separadas sólo por un salto de línea. Una vez impresos los resultados, y actualizado el índice, siguiendo la filosofía UNIX, el programa deberá cerrar. El programa no deberá realizar ninguna otra impresión para permitir su uso en *Shell scripts*.

La Fundación Linux ha definido que los índices deben ser tablas de hash. Una tabla de hash es una estructura de datos que indica una correspondencia entre una clave, llamada hash, y una estructura de datos. La Fundación desea que el hash sea calculado a partir de los términos de búsqueda, y quepa en un `int`. Debe definir un formato de archivo que permita almacenar y cargar las mismas.

Los términos de búsqueda deben ser palabras completas pertenecientes al nombre del archivo: si un archivo se llama "Ricitos de Oro.pdf", debe poderse encontrar con los términos "ricitios", "de", "oro" y "pdf"; pero no con los términos "ricito" o "de oro".

No se deberán borrar entradas del índice aunque el archivo correspondiente haya sido borrado.

3 Requerimientos del proyecto

El programa recibirá, como único argumento obligatorio, el **término de búsqueda**. Adicionalmente, debe poder manejar cualquiera de los siguientes argumentos opcionales:

FLAG	LONGFORM	DESCRIPCIÓN
`-d <carpeta>`	`--dir <carpeta>`	Establece `<carpeta>` como el directorio desde el cual inicia la búsqueda. Por defecto, se inicia en el directorio actual.
`-m <altura>`	`--max <altura>`	Establece `<altura>` como el número de niveles de subdirectorios que se explorará para la actualización del índice. Por defecto, la altura máxima es de 20
`-i <archivo>`	`--index <archivo>`	Establece `<índice>` como el archivo de índice a usar Debe definir un nombre de índice para usar por defecto si este *flag* no es recibido y manejar la posibilidad de que el mismo no se haya creado o esté vacío.
`-u`	`--noupdate`	Indica que no se debe entrar en directorios que ya estén en el índice, buscando qué agregar al índice; sólo en los directorios que se encuentren nuevos.
`-a`	`--noadd`	Indica que no se debe entrar en directorios que no estén en el índice, buscando qué agregar al índice; sólo directorios que ya estén en él. Si se reciben ambos *flags* (por ejemplo, `-ua`), no se recorren los directorios; sólo se consulta el índice

El programa debe garantizar exclusión mutua sobre la tabla de hash del índice. El programa debe usar hilos del estándar POSIX. Puede usar tantas tablas de hash como considere necesario. Puede usar cualquier otra estructura de datos vista en clase.

4 Requerimientos del informe

Debe entregar un informe que contenga:

- Introducción y estructura del informe
- Explicación de cómo compilar y correr su programa
- Explicación de la estrategia de creación de hilos usada
- Explicación de la estrategia de exclusión mutua usada
- Explicación de la función de hash usada
- Explicación de su formato de archivo
- Cualquier otra explicación que considere necesaria
- Conclusiones y lecciones aprendidas

La estrategia explicada debe coincidir con la estrategia implementada.

5 Evaluación

Se asignarán:

- 5 puntos por su informe
- 2 puntos por su estrategia de creación de hilos y su implementación adecuada
- 2 puntos por su estrategia de exclusión mutua y su implementación adecuada
- 1 punto por su función de hash y su implementación adecuada
- 1 punto por su diseño de estructura de datos para almacenar el índice (tabla de hash y estructuras contenidas) y su implementación adecuada
- 1 punto por su formato de archivo y su implementación adecuada
- 1 puntos por su acceso correcto a los *i-node* de directorios y archivos
- 1 puntos por ejecución correcta y completa
- 1 punto por seguir convenciones de C en UNIX

El programa debe compilar y correr sin errores en las computadoras de la universidad.

6 Entrega

Debe entregar su código e informe en un archivo compactado de Linux (.tar, .tgz, etc.) libre de archivos intermedios o ejecutables. Deberá subirlo al Moodle de la materia en la sección marcada como "  Proyecto 2" hasta el viernes, 23 de marzo a las 7:30 a.m. Sólo deberá efectuar una entrega por grupo.