Autor: Eduardo Murillo

Taller 12

Mutexes

Introducción

En este taller usaremos sincronización hilos para ver coordinar la ejecución de hilos.

Descripción

En este taller, implementaremos un programa, que nos permitirá buscar el número de ocurrencias de distintas palabras en un documento de texto. El programa será multihilos para acelerar el proceso.

./buscar ruta hilos palabra1 palabra2 palabra3

Se provee la siguiente función, para saber de antemano el número de líneas del documento y el tamaño de cada línea.

```
#define MAX 1000000

int numero_lineas(char *ruta, int *tam_lineas){
        if(ruta != NULL){
                FILE* ar = fopen(ruta,"r");
                int lineas = 0;
                int tam_linea;
                while(!feof(ar)){
                        tam_linea++;
                        char c = getc(ar);
                        if(c == '\n'){
                                if(tam_lineas != NULL){
                                        tam_lineas[lineas] = tam_linea;
                                }
                                lineas++;
                                tam_linea = 0;


                        }
                }
                fclose(ar);
                return lineas;
        }
        return -1;
}
```

Con esta información, Uds. podrán particionar el archivo y darle una parte a cada hilo. Su programa inicialmente creará un arreglo, donde guardará las palabras a buscar:

char palabras[…]

y un arreglo donde llevaremos la cuenta de cada palabra:

int num_palabras[…]

Autor: Eduardo Murillo

Crearemos el número de hilos que nos diga el segundo argumento. Luego, haremos que cada hilo trabaje sobre secciones de igual tamaño del archivo. Usaremos las librerías string.h y stdio.h:

Para abrir un archivo:

FILE* fp = fopen("ruta","r");

Para leer una línea del archivo a la vez:

char *linea = fgets(fp, buf, tamano);

Esta función devuelve NULL cuando ya no hay más líneas.

Para obtener cada palabra de la línea:

strtok(string, " ,.!?:;"); //separadores son espacios, comas y símbolos.

strtok(NULL, " ,.!?:;"); //En cada llamada se devolverá una palabra. Cuando no haya más, devolverá NULL.

Para empezar a leer el archivo en la parte que nos corresponde:

fseek(fp, offset, SEEK_SET)

Noten que varios hilos estarán modificando el arreglo **num_palabras**. Su programa imprimirá de manera constante (cada segundo), el número de cada palabra que se ha encontrado hasta el momento.

Entregable

Repositorio Git con Makefile. Tarea grupal.