Universidad Simón Bolivar
Sistemas de Operación
Proyecto 1

Ricardo Churión
Carnet: 11-10200

Douglas Torres
Carnet : 11-11027

Informe

Implementación: Se utilizan 3 estrucutras enlazadas bases, una estructura amigos, que es una lista de nombres, una estructura lista que es un nombre con una lista de amigos asociada y una estructura comparacion que es el nombre de una tupla y dos listas de amigos asociadas.

El map recibe la lista que se leyo del archivo de entrada y devuelve una estructura del tipo comparacion. El reduce recibe esta estructura y la transforma en una nueva lista que ahora tiene el nombre de el par de amigos y la lista de amigos comunes asociada.

Se trato de tener un funciones.c con las funciones, pero haciendo esto los threads no corrian con las funciones por lo que se coloco toda la informacion en un solo .c

Detalles de hilos y procesos:

Hilos: Las funciones map y reduce recibian mas de un argumento. Como para llamar a un hilo solo se le puede pasar un argumento, se creo una estructura hilos que constaba de una lista enlazada de argumentos para poder pasarselos todos de una sola vez.

Ademas se noto que los hilos no pueden tener returns, sino que se utiliza la funcion ptrhead_exit() para mandarlo al hilo principal y en este se usa pthread_join() para recibir la informacion.

Adicionalmente, se declararon dos semaforos del tipo mutex para lidiar con las secciones criticas que se encontraron, una en map y una en reduce ya que los hilos trataban de modificar las estructuras al mismo tiempo.

Procesos: En el programa donde se implementan procesos se encontraron diversos errores que no pudieron ser solucionados. Uno de ellos fue un “Bus error”, que se ocacionaba al intentar escribir el archivo de salida. Se investigo sobre el problema y se determino que este era ocacionado por intentar acceder a una direccion no permitida. Con esta informacion no se pudo resolver el problema. Otro inconveniente que aparecio fue el uso de los semaforos debido a que siempre daba error la declaracion de los mismos y nunca se pudo probar su correctitud.

A continuacion se presenta la tabla de tiempos que se obtuvieron al ejecutar el programa en una computadora con procesador AMD Dual-Core C60, ejecutado en SO Debian 7 de 64 bits con un archivo de pruebas de 7 lineas. Para esto se uso el comando time de bash y se tomaron las mediciones de las secciones “real”,“user” y “sys”, que consisten en el tiempo real que elapso ejecutandose el programa, en la que los hilos pasaron ejecutandose en el CPU en modo usuario y modo kernel respectivamente.

Cantidad Hilos	Real (time)	User (time)	Sys (time)
1	0m0.019S	0m0.000S	0m0.004S
5	0m0.011S	0m0.004S	0m0.004S
10	0m0.011S	0m0.000S	0m0.008S

Los resultados son consistentes con la teoria ya que con un solo hilo toma mucho tiempo en relacion al caso de 5 hilos, pero con el de 10 hilos se empieza a perder rendimiento porque se dividen demasiadas instrucciones en pocas lineas.

No se pudo hacer el makefile, por lo tanto, para compilar el friendfindh.c debe ejecutarse en el terminal: gcc -lpthread -o pruebah friendfindh.c y para ejecutarlo se usa la sintaxis del enunciado.

Para compilar el friendfindp.c debe ejecutarse en el terminal: gcc -o pruebap -lrt friendfindp.c y tambien se ejecuta con la sintaxis del enunciado.