Sistemas Operativos
Práctica 2

Blanca Martín · Fernando Villar
Grupo 2201 · Pareja 10

17/03/2017

Índice

1. Introducción

En este documento se discutirán los ejercicios propuestos en la segunda práctica de Sistemas Operativos (grupo 2201), que orbitan alrededor de los conceptos relacionados con hilos, señales, manejadores y máscaras, así como con sus diversas funciones de definición y manipulación, sus usos y su funcionamiento.

2. Listado de código fuente

- ejercicio3a.c
- ejercicio3b.c
- ejercicio4.c
- ejercicio6.c
- ejercicio8.c
- ejercicio10.c

Todo ello regulado por el archivo "Makefile", que genera automáticamente los archivos .o y los ejecutables para cada ejercicio, y que cuenta con la posibilidad de efectuar un comando **make clean** para eliminar todos los archivos generados. Todos los ejercicios correspondientes a la parte de hilos llevan su complemento correspondiente en dicho "Makefile". La documentación completa del código fuente se presenta en el archivo PDF aportado en este mismo directorio, `"Documentación código fuente (Doxygen).pdf"`.

3. Ejercicio 3 (a y b)

En el primer apartado del ejercicio 3, se ha ejecutado una instrucción **fork**; en el código del hijo se ejecuta la instrucción que cuenta los primeros n números primos (siendo n un entero introducido como argumento) y se mata el proceso; el padre lo espera, y se calcula el tiempo que tarda en ejecutarse todo el programa, para cien procesos hijos generados secuencialmente. Por otro lado, en el segundo apartado, se crean cien hilos en los que, en cada uno, se ejecuta exactamente la instrucción de conteo de primos, contando también el tiempo.

El resultado de la ejecución es el siguiente:

```
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio3b 10000
Tiempo empleado: 263369 usecs.
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio3a 10000
Tiempo empleado: 339749 usecs.
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio3a 10000
Tiempo empleado: 342533 usecs.
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio3b 10000
Tiempo empleado: 248995 usecs.
```

Figura 1: Ejecución de ejercicios 3a y 3b

Como se puede apreciar, el método de los hilos es más rápido. La razón principal es que el sistema operativo no se toma tiempo para copiar los datos del programa cada vez que se crea un hilo nuevo, mientras que sí que lo tiene que hacer para la creación de nuevos procesos.

4. Ejercicio 4

Para pasar parámetros a los diferentes hilos generados, se ha usado una estructura en la que, además, y para responder a la pregunta planteada en el ejercicio, se incluye la estructura de información correspondiente al otro hilo. En esta estructura están:

- Identificador del hilo.
- Entero por el que se ha de multiplicar la matriz.
- Dimensión de la matriz.
- Matriz.
- Fila actual por la que se va leyendo.
- Estructura del otro hilo (de la otra matriz).

En la implementación se pide la dimensión de las matrices, los multiplicadores y las matrices, y se rellenan las estructuras; éstas se pasan a los hilos como parámetros, y en ellos, por una parte, se realiza la multiplicación, y por otra, se imprime cada fila del resultado y alguna información sobre la ejecución del hilo paralelo.

```
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio4
Introduzca dimensión de la matriz cuadrada:
2
Introduzca multiplicador 1:
3
Introduzca multiplicador 2:
4
Introduzca matriz 1:
1
2
3
4
Introduzca matriz 2:
4
3
2
1
Realizando producto:
Hilo 1 multiplicando fila 0 resultado 3 6  - el Hilo 2 va por la fila 0
Hilo 2 multiplicando fila 0 resultado 16 12  - el Hilo 1 va por la fila 1
Hilo 1 multiplicando fila 1 resultado 9 12  - el Hilo 2 va por la fila 1
Hilo 2 multiplicando fila 1 resultado 8 4  - el Hilo 1 va por la fila 2
```

Figura 2: Ejecución de ejercicio 4

5. Ejercicio 6

En el sexto ejercicio de la práctica se ha implementado un programa sencillo: se trata de crear un proceso hijo que imprima un mensaje por pantalla cada 5 segundos, hasta que el padre, que espera 30 segundos, manda una señal para que el proceso hijo finalice.

```
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio6
Soy el proceso hijo con PID: 17002
Soy el proceso hijo con PID: 17002
Soy el proceso hijo con PID: 17002
Soy el proceso hijo con PID: 17002
Soy el proceso hijo con PID: 17002
Soy el proceso hijo con PID: 17002
```

Figura 3: Ejecución de ejercicio 6

6. Ejercicio 8

De este ejercicio, aunque se recomienda la lectura de los comentarios dentro del código para entender la implementación, merece la pena mencionar el hecho de que se ha utilizado un manejador vacío para

evitar que, la última vez que el último proceso manda una señal al proceso raíz, se vuelva a imprimir el mensaje correspondiente al paso por cada proceso, actividad realizada en el manejador principal. Sólo afecta a esa señal dado que, a partir de ahí, todo son señales de terminación.

```
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio8 3 2
Hola PID=17220. Fecha y Hora: 17/03/17 05:01:37
Hola PID=17221. Fecha y Hora: 17/03/17 05:01:39
Hola PID=17222. Fecha y Hora: 17/03/17 05:01:41
Hola PID=17223. Fecha y Hora: 17/03/17 05:01:43
Hola PID=17220. Fecha y Hora: 17/03/17 05:01:50
Hola PID=17221. Fecha y Hora: 17/03/17 05:01:52
Hola PID=17222. Fecha y Hora: 17/03/17 05:01:54
Hola PID=17223. Fecha y Hora: 17/03/17 05:01:56
Muere PID=17221.
Muere PID=17222.
Muere PID=17223.
Muere PID=17220.
```

Figura 4: Ejecución de ejercicio 8

7. Ejercicio 10

Para la implementación de este último problema se han utilizado:

- Tres manejadores: uno para despertar al proceso A después de que haya terminado por leer la palabra FIN; otro para actuar de temporizador en el proceso B, realizando lecturas cada 5 segundos; y un último para enviar una señal del proceso B al A para que este último muera.
- Una máscara de señales, que contiene las tres señales correspondientes a los tres manejadores. Se utiliza para que el proceso A reconozca la señal de despertar enviada desde el proceso B, y para que el proceso B reconozca la señal de alarma que actúa de temporizador.
- Un fork, en donde el proceso hijo (A) realiza aperturas/clausuras y escrituras del fichero, mientras que el proceso padre (B) realiza aperturas/clausuras y lecturas del mismo fichero.
- Una función que genera números aleatorios para seleccionar palabras aleatorias de la frase propuesta.

Se recomienda, como en el ejercicio anterior, leer las explicaciones y los comentarios contenidos en el código.

```
sliezzan@debian-sliezzan:~/Documents/OperatingSystems/SOPER/02_P2$ ./ejercicio10
Proceso B lee una palabra: PROCESO
Proceso B lee una palabra: A
Proceso B lee una palabra: FICHERO
Proceso B lee una palabra: LEE
Proceso B lee una palabra: EN
Proceso B lee una palabra: FIN
Proceso B lee una palabra: CADENA
Proceso B lee una palabra: EL
Proceso B lee una palabra: FICHERO
Proceso B lee una palabra: A
Proceso B lee una palabra: EL
Proceso B lee una palabra: FICHERO
Proceso B lee una palabra: CADENA
Proceso B lee una palabra: UN
Proceso B lee una palabra: HASTA
Proceso B muere.
Proceso A muere.
```

Figura 5: Ejecución de ejercicio 10 (simplificado a 15 palabras leídas por B