PRACTICA 5

Objetivo General

Reforzar los conocimientos sobre el manejo de hilos utilizando la API de C llamada pthread.

Objetivos Particulares:

- Desarrollar un programa de juguete en **C PURO** que permita al alumno acercarse al desarrollo de aplicaciones paralelas mediante el uso de hilos.
- El programa va a consistir en encontrar 3 de las características más comunes en número enteros, las cuales son que un número sea perfecto, fuerte y primo,
- Plantear una arquitectura paralela de maestro y esclavo para desarrollar la aplicación pseudo paralela de esta práctica.

Documentos a entregar

Esta práctica se tiene que realizar de forma individual y se debe entregar los siguientes documentos:

- El programa desarrollado en C previamente compilado y probado de manera previa para evitar contratiempos en la sesión de revisión de la práctica. Esto es muy importante ya que no se admiten cambios o modificaciones de este en la sesión de revisión de la práctica.
- Informe individual de cómo se abordaron cada uno de los problemas planteados (30 % de la calificación). El informe debe contener las mismas secciones de las practicas pasadas y debe realizarse a **MANO**, esto es, todo se tiene que escribir a mano sin la ayuda de ningún editor de texto y debe contener Caratula, Introducción, Desarrollo y Conclusiones. En la sección de Desarrollo se debe poner el diagrama de flujos para determinar cada una de las características que se mencionaron en los objetivos particulares.

Plazo de entrega

La hora y fecha límite para enviar el programa será el domingo 09 de junio del 2019 y estos junto con el informe serán revisados en el laboratorio el lunes 10 de junio del 2019.

Nota. No se recibirá ninguna práctica fuera de ese horario, sin ninguna excepción. EL TRABAJO DEBE SER TOTALMENTE INDIVIDUAL Y SIN REPLICAS

Especificaciones de las partes que conforman la práctica:

Como se mencionó anteriormente esta práctica tiene por objetivo retomar y reforzar el concepto de hilos dentro el panorama exclusivo del curso de Sistemas Operativos, por lo que omitiremos el paradigma en el cual realmente se explota el uso de estas herramientas de programación que es el de la programación concurrente. Es por ello que omitiremos el uso de mecanismos de sincronización como los son semáforos y candados y solo plantearemos el uso de una arquitectura pseudo aleatoria.

Vamos a revisar brevemente las principales arquitecturas de programación que se encuentran reportadas en la literatura, para las cuales nos vamos a valer de la clasificación de Flynn Michael. En esta clasificación se establece como se pueden agrupar el número de instrucciones, así como el flujo de datos que es utilizado para el procesamiento de la información, teniendo las siguientes arquitecturas:

- SISD (Single Instruction and Single Data Stream). En esta arquitectura se sigue la forma de programación tradicional, en la cual una serie de instrucciones es ejecutada una a una sobre una serie de datos obtenidos de memoria.
- SIMD (Single Instruction and Multiple Data Streams). Significa que todas las unidades paralelas comparten la misma instrucción, pero la realizan en diferentes elementos de datos. Esta arquitectura se puede plantear una analogía del mundo real, en este caso pensemos en el deporte de remo de equipos el cual consiste en la propulsión de una embarcación sobre el agua, con o sin timonel, mediante la fuerza muscular de varios remeros. En este caso hay varias entidades realizando una misma instrucción al mismo tiempo sobre diferentes datos que serían los remos que tendrían asignados a su cargo, esta instrucción se debe realizar de forma sincronizada ya que cualquier error en el ritmo establecido puede afectar el resultado final de la carrera de remos.
- MISD (Multiple Instruction and Single Data Stream). Este modelo es similar a SIMD, sin embargo, en este caso se tiene un conjunto de instrucciones que pueden ser ejecutadas con fragmentos de datos compartidos.
- MIMD (Multiple Instruction and Multiple Data Stream). En este caso se combinan cada una de las arquitecturas que describimos anteriormente ya que ahora se pueden pensar en una aplicación que tiene múltiples instrucciones pueden ser ejecutadas en diferentes conjuntos de datos.

La arquitectura que se propone para esta práctica es la MISD, ya que vamos a tener tres algoritmos diferentes ejecutándose sobre el mismo conjunto de datos, y nos referimos a que tendremos un mismo arreglo en el cual se busquen cada una de las características de números enteros mencionadas en los objetivos de esta práctica. El diagrama que muestra la arquitectura pseudo paralela propuesta para esta práctica se puede observar en la Figura 1.

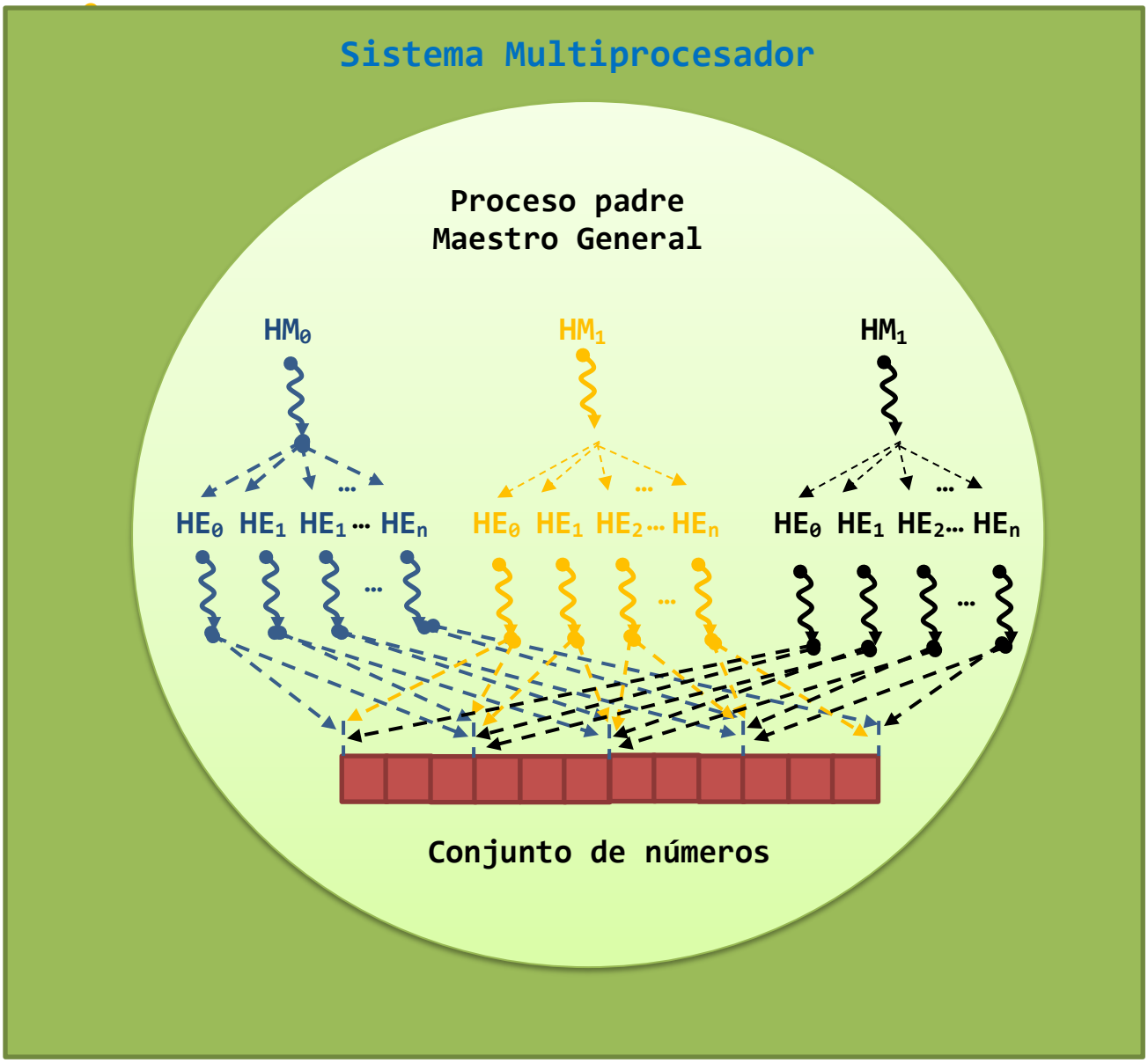

Figura 1. Arquitectura pseudo paralela propuesta

En la Figura 1 tenemos:

- $HM_i \; \forall \; i = 1,2,3$: representan los hilos que tendrán el rol de Hilos Maestros dentro de la arquitectura propuesta para la práctica.
- $HE_i \; \forall \; i = 1,2, \ldots, n$: representan los hilos que tendrán el rol de Hilos Esclavos y serán lanzados por el conjunto de Hilos Maestros dentro de la arquitectura propuesta para esta práctica.

Otras cosas que hay que señalar de la arquitectura son las siguientes:

1. El proceso padre será el que se cree cuando el programa en C se compile y se cree el ejecutable y tendrá el rol del Maestro General.
2. La cantidad de Hilos Maestros estará fija ya que solo se necesitan 3, los cuales revisaran cuantos números enteros son perfectos, fuertes y primos.
3. La cantidad de Hilos Esclavos será variada por cada uno de los Alumnos y obtendrán sus conclusiones acerca de cual es la mejor configuración de estos la cual será reportada en su informe correspondiente a esta práctica.
4. En el esquema de la Figura 1 se puede observar que de manera genérica se tiene un Conjunto de Números, este se obtendrá en base a un parámetro de entrada que será leído en la ejecución del programa, el cual será detallado a continuación.

Como podemos ver el esquema de la Figura 1 nos da la información de base para poder desarrollar esta práctica, sin embargo, vamos a enunciar cada uno de los pasos que se necesitan implementar para llevarla a cabo.

1. El programa desarrollado en C debe ser compilado en un formato formal con el siguiente comando, pensando que el programa lleva por título Propiedades_Numero.c

```
$ gcc -Wall -Werror Propiedades_Numero.c -o
                Propiedades_Numeros
```

2. En la ejecución del programa compilado debe incluirse el comando time y el limite de los números que se pretenden revisar. Por facilidad siempre probar intervalos de base 10, al igual que probar con esta base el numero de hilos lanzados, un ejemplo de ejecución seria con el siguiente comando:

```
$ time ./Propiedades_Numeros 1000000
```

3. Para mas facilidad en la revisión de esta practica se establece el siguiente estándar. Definir 4 funciones que regresen y reciban un objeto las cuales son:
   - void * t_control_total(void *arg); Función que ejecutará cada hilo maestro y podrá crear sus propios hilos esclavos en la búsqueda de las propiedades establecidas para esta práctica, el HM_0 buscara los números perfectos, el HM_1 buscara los números fuertes y el HM_2 buscara los números primos
   - void * t_perfectos(void *arg); Función para buscar los números perfectos dentro de un intervalo definido en base al numero de hilos maestros lanzados por cada hilo maestro.
   - void * t_fuertes(void *arg); Función para buscar los números fuertes dentro de un intervalo definido en base al número de hilos maestros lanzados por cada hilo maestro

- void * t_primos(void *arg); Función para buscar los números primos dentro de un intervalo definido en base al número de hilos maestros lanzados por cada hilo maestro

4. La implementación de la lógica del main() y las funciones que definimos anteriormente son completamente libres, en cuestión de variables y algorítmica en general la cual será explicada a detalle en la revisión de la práctica.