Sistemas Operativos 2/2023
Laboratorio 3

Profesores:
Cristóbal Acosta (cristobal.acosta@usach.cl)
Fernando Rannou (fernando.rannou@usach.cl)
Ayudantes:
Ricardo Hasbun (ricardo.hasbun@usach.cl)
Leo Vergara (leo.vergara@usach.cl)

I. Objetivos Generales

Este laboratorio tiene como objetivo aplicar técnicas de programación imperativa mediante lenguaje C, como la recepción de parámetros mediante `getopt` y compilación mediante `Makefile` sobre sistema operativo Linux. Además de aplicar conocimiento adquiridos en cátedra sobre Concurrencia, Multihebrado, y llamados al SO Linux de manera exitosa.

II. Objetivos Específicos

1. Usar las funcionalidades de `getopt()` como método de recepción de parámetros de entradas.
2. Uso del Makefile para compilación por partes de programas.
3. Utilizar recursos de pthread para sincronizar hebras y proteger recursos compartidos.
4. Implementar soluciones de exclusión mutua por software.
5. Construir funciones de lectura y escritura de archivos.
6. Practicar técnicas de documentación de programas.

III. Enunciado

III.A. El bombardeo de partículas

En este laboratorio, se busca simular la energía depositada en un material por partículas de alta energía que lo impactan (ejemplo: partículas en un satélite). El programa a desarrollar calculará y registrará la energía en Joules que cada partícula va depositando en algún punto del material.

Para facilitar el trabajo, se utilizará un modelo de una dimensión (1D). Entonces, se usará un arreglo para llevar registro de las energías acumuladas.

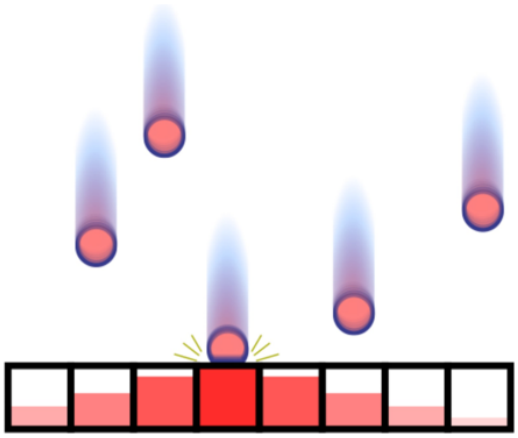

Figure 1. Acumulación de energías por partículas.

Por cada partícula simulada, se entrega su posición de impacto j y la energía potencial E_j que trae. La energía se deposita en la celda j y las celdas vecinas según la siguiente ecuación:

$$E_i = E_i + E_{j,i}$$

donde

$$E_{j,i} = \frac{10^3 \times E_j}{N\sqrt{|j-i|+1}}$$

donde N es el número de celdas del material. Note que es posible que la posición de impacto sea más grande que N. Esto es válido, y significa que aunque la partícula impactó fuera del material medido, su energía igualmente se registra en la zona de interés.

En caso de que la energía depositada sea muy pequeña, esta no se registrará, para lo cual se utilizará un umbral que se calcula como `MIN_ENERGY` $= 10^{-3}/N$, por lo que se deposita energía solo en el caso que $E_{j,i} \geq$ `MIN_ENERGY`.

III.B. Los archivos de entrada y de salida

Un bombardeo de partículas se especifica en un archivo de entrada, el cual contiene el número de partículas, y para cada partícula, la posición de impacto y la energía (E_j). El siguiente es un ejemplo de un archivo de bombardeo:

```
6
4 81
8 10
10 100
7 35
11 12
5 8
```

La primera línea indica la cantidad de partículas y en las siguentes entrega la descripción de cada una de ellas. Es importante destacar que en este archivo `NO` se especifica el número de celdas, ya que este parámetro se ha ingresado por argumento de línea de comando.

Para el archivo de salida, después de haber impactado todas las partículas, se debe escribir en la primera línea la posición del material con máxima energía y la cantidad acumulada. En las siguientes líneas, se debe indicar cada posición en orden con su respectiva energía acumulada (se deben escribir todas las celdas del material). Por ejemplo, haciendo uso de la entrada anterior, teniendo un material con 35 celdas, el resultado sería el siguiente, donde se logra ver en la celda 10 la máxima energía y luego la energía acumulada en todas las posiciones del material. Por ejemplo:

```
10 4732.568359
0  2537.520752
1  2745.225830
2  3027.479004
3  3456.707031
...
```

IV. El programa

IV.A. Lógica de solución

En este laboratorio se simulará el impacto de partículas en un material, que con base en la ecuación (ver sección III) y un archivo de texto, calcule cuál es la celda con mayor energía depositada.

1. El proceso principal (hebra madre) recibirá por linea de comandos el nombre del archivo de entrada, el nombre del archivo de salida, el tamaño del chunk que leerá cada hebra y la cantidad de hebras que deberán ser generadas por la hebra madre para realizar los cálculos necesarios.

2. La hebra madre solo debe abrir el archivo de entrada y las hebras hijas deberán leer el archivo.

3. Los cálculos relacionados con el bombardeo deben ser obtenidos por las hebras hijas. Por otro lado, la hebra madre debe esperar a que todas las hijas terminen su trabajo.

4. Cuando una hebra este libre, trata de leer chunk lineas del archivo de entrada. Este acceso debe ser exclusivo, es decir, solo una hebra hija puede estar leyendo el archivo a la vez.

5. Una vez que la hebra lee el chunk, esta se encargara de actualizar los datos para cada celda de material, para lo cual debe existir una o varias estructuras compartidas las cuales registren los datos de las expresiones.

6. El acceso a estas estructuras compartidas también debe ser exclusivo. Note que es muy probable que varias hebras estén actualizando estas estructuras a la vez.

7. Una vez que una hebra termina de actualizar los datos para una línea especifica, está vuelve a intentar leer chunk lineas del archivo de entrada. Esto se realiza hasta que no se puedan leer mas lineas.

8. Como el numero de hebras no necesariamente divide al número de lineas del archivo de entrada, ocurrirá el caso de que una hebra leerá menos lineas que chunk cuando llegue al final del archivo. También existe la posibilidad de que una hebra intente leer y se encuentre con el final del archivo. La solución para esta problemática debe ser implementada por usted.

9. En caso de que se emplee el flag -D, se debe mostrar por la terminal el gráfico solicitado junto con la cantidad de lineas leídas por cada hebra.

10. Finalmente, la hebra madre escibe los resultados con el mismo formato solicitado en el laboratorio 1 y laboratorio 2.

IV.B. Línea de comando

La ejecución del programa tendrá los siguientes parámetros que deben ser procesados por `getopt()`:

- `-N`: es el número de celdas
- `-H`: es el número de hebras a generar
- `-i`: es el archivo con el bombardeo (archivo de entrada)
- `-o`: es el archivo de salida
- `-c`: es el número de chuck, es decir, cantidad de líneas a leer por chunk.
- `-D`: bandera o flag que permite indicar si se quiere ver por consola la cantidad de celdas de material.

Por ejemplo:

```
$ ./lab3 -N 5 -H 3 -i input.txt -o output.txt -c 10 -D
```

En el caso que se utilice la flag -D se debe imprimir por consola un gráfico simple y normalizado con las energías. Por ejemplo, utilizando un N=10 y la flag -D presente, se obtiene:

```
0   8881.3223 |oooooooooooooooooooooooooooooooooooooooooooooooo
1   9608.2910 |ooooooooooooooooooooooooooooooooooooooooooooooooooo
2  10596.1777 |ooooooooooooooooooooooooooooooooooooooooooooooooooooo
3  12098.4746 |oooooooooooooooooooooooooooooooooooooooooooooooooooooooo
4  15066.8076 |ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
5  13584.3311 |ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
6  13256.4805 |oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
7  14255.6436 |ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
8  13870.8066 |oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
9  14156.3018 |ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
```

IV.C. Requerimientos

Como requerimientos no funcionales, se exige lo siguiente:

- Debe funcionar en sistemas operativos con kernel Linux.
- Debe ser implementado en lenguaje de programación C.
- Se debe utilizar un archivo Makefile para compilar los distintos targets.
- Realizar el programa utilizando buenas prácticas, dado que este laboratorio no contiene manual de usuario ni informe, es necesario que todo esté debidamente comentado.
- Los programas se encuentren desacoplados, es decir, que se desarrollen las funciones correspondientes en otro archivo .c para mayor entendimiento de la ejecución.
- La solución debe implementar :
  (a) `pthread_create()`
  (b) `pthread_join()`
  (c) `pthread_mutex_init()`
  (d) `pthread_mutex_lock()`
  (e) `pthread_mutex_unlock()`

V. Entregables

El laboratorio es en parejas. Si se elije una pareja está no podrá ser cambiada durante el semestre. Se descontará 1 punto (de nota) por día de atraso con un máximo de tres días, a contar del cuarto se evaluará con nota mínima. Debe subir en un archivo comprimido ZIP (una carpeta) a USACH virtual con los siguientes entregables:

- `Makefile`: Archivo para compilar los programas.
- `lab3.c`: Archivo principal del laboratorio, contiene el main. Se concentra en obtener y validar los datos entregados como argumentos del programa. Si los datos son validos, debe abrir el archivo de entrada, crear y esperar por las hebras y dar formato a los resultados finales en base a funciones auxiliares.
- `funciones.c y funciones.h` : Archivo con funciones, en el .c el desarrollo de la función y en el . h las cabeceras. Recuerde que todas las funciones deben estar comentadas, explicadas de forma entendible especificando sus entradas, funcionamiento y salida. Si una función no esta explicada se bajara puntaje.
- `Otros:` Cualquier otro archivo fuente que se vea necesario para la realización del lab. Pueden ser archivos con funciones, estructuras de datos, etc.
- Trabajos con códigos que hayan sido copiados de un trabajo de otro grupo serán calificados con la nota mínima.

Recuerde que todas las funciones deben estar comentadas, explicadas de forma entendible especificando sus entradas, funcionamiento y salida. Si una función no está explicada se bajará puntaje. Se deben comentar todas las funciones de la siguiente forma:

```
// Entradas: explicar qué se recibe
// Salidas: explicar qué se retorna
// Descripción: explicar qué hace
```

El archivo comprimido (al igual que la carpeta) debe llamarse: RUTESTUDIANTE1_RUTESTUDIANTE2.zip

Ejemplo 1: 19689333k_186593220.zip

NOTA 1: El archivo debe ser subido a uvirtual en el apartado "Entrega Lab3".

NOTA 2: Cualquier diferencia en el formato del laboratorio que es entregado en este documento, significara un descuento de puntos.

NOTA 3: SOLO UN ESTUDIANTE DEBE SUBIR EL LABORATORIO.

NOTA 4: En caso de solicitar corrección del lab está será en los computadores del Diinf, es decir, si funciona en eso computadores no hay problema.

NOTA 5: Cualquier comprimido que no siga el ejemplo 1, significara un descuento de 1 punto de nota.

NOTA 6: La fecha de entrega es la única disponible, no hay posibilidad de moverla.

VI. Fecha de entrega

Domingo 10 de Diciembre, 2023.