Universidad Simón Bolívar
Depto. de Computación y T.I.
Sistemas de Operación (CI-3825)
Abril-Julio 2.022

Proyecto 2 (12%)

Objetivos

- Familiarizarse con uso de procesos y de hilos.
- Desarrollar una aplicación con procesos cooperantes.
- Diseñar esquemas de comunicación.
- Resolver problemas de región crítica.

Descripción general del problema

Ud debe desarrollar una aplicación que ordene de forma ascendente los enteros almacenados en los archivos ubicados en una jerarquía de directorios. En otras palabras, como entrada tendrá un árbol de directorios, su aplicación debe ubicar en el árbol los archivos con extensión txt, ordenarlos por separado y luego, ir mezclando[1] estas secuencias ordenadas en una secuencia ordenada más grande. En otras palabras, se generará un solo archivo de enteros ordenados.

Puede asumir que:

- Los archivos de extensión txt serán regulares.
- no habrá archivos enlaces fuertes ni simbólicos.
- los archivos tendrán sólo los caracteres 0-9 y salto de línea. Por ejemplo, el siguiente archivo tiene 8 enteros:

```
2
345
22
77
23456786453
11
56
44
```

Su aplicación utilizará los siguientes procesos cooperantes: un Lector, varios Ordenadores, varios Mezcladores y un Escritor. A continuación se muestra un pequeño

1 No debe concatenar archivos ordenados y volverlos a ordenar. Debe combinarlos, considerando que ya, cada uno, está ordenado.

bosquejo de su estructura general.

- **Lector**: Se encargará de:
  - crear los procesos/hilos restantes
  - inicializar los mecanismos de comunicación necesarios
  - encontrar los archivos con extensión txt
  - Por cada archivo no procesado:
    - asignarlo a un Ordenador desocupado.
  - indicarle a los Mezcladores y al Escritor que deben realizar el último paso.
  - destruir los procesos/hilos creados.
  - Terminar

  Note que :
  - este proceso lleva control de los Ordenadores y Mezcladores desocupados.
  - cuando todos los Ordenadores y Mezcladores estén desocupados y no haya más archivos que procesar, el Lector le notificará a los Mezcladores que comiencen a mandar sus secuencias al Escritor.

- **Ordenadores:**
  - recibe un archivo a la vez
  - lo ordena usando el algoritmo de selección.
  - informa al Lector que terminó
  - recibe un Mezclador desocupado
  - pasa la secuencia ordenada al Mezclador desocupado
  - Espera o pide otro archivo.

- **Mezcladores**:
  - inicializa una secuencia vacía, que está ordenada.
  - recibe una secuencia ordenada
  - La mezcla con la secuencia que ya tiene ordenada
  - Notifica al Lector que terminó

  Cuando el Lector le indique, debe enviar su secuencia ordenada al Escritor.

- **Escritor**:
  - espera hasta que el Lector le indique que va a comenzar el último paso.
  - recibe una secuencia por cada Mezclador,
  - Mientras exista una secuencia que no esté vacía
    - selecciona el siguiente menor, entre los menores de cada secuencia.
    - lo escribe en el archivo de salida.

Note que este último paso es similar a lo que hacen los mezcladores. Sin embargo, **no** se debe generar una secuencia nueva y después escribir en disco. En su lugar, lo que se debe hacer es, verificar el primer elemento de cada secuencia, determinar cuál es el menor, y ese menor es el que se escribe en el archivo inmediatamente.

Restricciones de implementación

Su aplicación debe cumplir con las siguientes restricciones:

- Los procesos deben comunicarse entre sí usando pipes no nominales o señales.
- Los hilos deben comunicarse entré sí usando memoria compartida o señales.
- Los hilos pueden usar semáforos, los procesos no pueden usarlos.

Las sintaxis de los comandos son:

```
# ordenaproc <num Ordenadores> <num Mezcladores> <raiz> <archivo salida>
# ordenahilo <num Ordenadores> <num Mezcladores> <raiz> <archivo salida>
```

donde

- `<num Ordenadores>`: número de procesos/hilos que ordenarán los enteros.
- `<num Mezcladores>`: Número de procesos/hilos que mezclarán secuencias ordenadas.
- `<raíz>`: es la raíz del árbol que se debe procesar
- `<archivo salida>`: Nombre del archivo dónde quedará la secuencia ordenada con todos los enteros.

Informe

- Realice un informe en el que explique los problemas de condición de carrera encontrados y cómo fueron resueltos.
- Implemente una versión monolítica de la aplicación descrita, es decir, ordenamiento y mezcla usando un solo proceso.

  ```
  # ordena <raiz> <archivo salida>
  ```

  La idea es usarlo como control en el experimento. Re use código de las otras dos versiones.
- Muestre un pequeño estudio experimental sobre el desempeño de las tres versiones de su aplicación, cada una será una serie diferente. Debe incluir una gráfica en $\mathbb{R}^3$:
  - **Eje X**: número de enteros ordenados.
  - **Eje Y**: número de Ordenadores + número de Mezcladores.
    - Varíe este parámetro para determinar si realmente impacta en el tiempo total para los mismos tamaño de entrada.
  - **Eje Z**: Tiempo total.

Recomendaciones

Entender bien el problema antes de diseñar su propuesta.

- Lea la información dada por las páginas de manual del sistema sobre las funciones

que debe usar.

- Diseñe **toda** su solución antes de comenzar a programar.
- **Programe en forma estructurada.**
- Vaya documentado su código a medida que lo vaya generando.
- Trabaje en forma ordenada!
- Si tiene alguna duda, con tiempo aclárela directamente con su profesor.

Entrega del proyecto:

Realización: Individual

Digital: Hasta las 11:59pm del jueves de semana 10.

Cada estudiante debe colocar en un archivo que con nombre carnet.tar.gz:

- los fuentes de su tarea (archivos .c y .h). Estos deben estar bien identificados, documentados y estructurados. Este conjunto será usado para compilar y correr su tarea.
- Un archivo de nombre *codigo.pdf* con los fuentes de su tarea. Exactamente la misma versión que los .c y .h. Éste será usado para evaluar el código de su tarea.
- El Makefile que pueda compilar/enlazar su aplicación.
- Un informe explicando manejos de regiones criticas y las gráficas de desempeño.

Note que debe estar subscrito al espacio canvas para poder optar a esta opción, ***no espere al día de la entrega para notificar que tiene problemas o que no se ha registrado***.