Proyecto Final - Reporte

Carlos Garcia Geronis A01748393

Lenguajes de Programación

Instituto Tecnológico de Estudios Superiores de

Monterrey

Proyecto Final-Reporte

Introducción

Para este proyecto se tenía como objetivo principal el entender cómo funciona el procesamiento a través de múltiples hilos, y cómo estos interactúan entre sí. Para lograr comunicar esto, hemos creado una herramienta que permite crear múltiples hilos con diferentes roles: Productores, que se encargan de generar tareas y operaciones matemáticas, y Consumidores, que se encargan de leer las tareas generadas de un bloque de memoria y solucionarlas mientras van saliendo. Debido a la naturaleza de los hilos, estos procesos no deberían esperar por cada uno y funcionar de manera independiente.

Contribuciones

Para este proyecto yo me encargué de hacer que los threads de Producer y Consumer hagan cada una lo que tiene que hacer. Para empezar, modifique la clase Producer para que genere adecuadamente las strings que contienen las operaciones matemáticas en formato de Scheme. Una vez se generaron esas strings, son enviadas al buffer para ser almacenadas hasta que las resuelva un Consumer. Luego, la clase Consumer espera a que el buffer tenga alguna operación para resolver, y cuando existe una, la lee, parsea y resuelve. Una vez resuelta, el Consumer elimina esa operación del buffer. Después, cambié la clase ProducerConsumer para que se encargue de generar y manejar los Threads. Para lograrlo, almacena los Threads en múltiples arreglos y los termina con un flag cuando se decide detener el software. Por último, validé que los procesos iniciaran y terminaran correctamente, e hice algunos cambios misceláneos para asegurar que la aplicación sea ligeramente más robusta.

Luego hice la integración de la GUI con el resto de las clases, me dedique a que al presionar el botón de inicio reciba cada uno de los datos ingresados por el usuario a través de los elementos de la GUI, siendo estos: jSpinner para productores, consumidores y el rango de valores máximo, así como jTextfields para los tiempos de espera, el tamaño del buffer y el rango de valores mínimo, así como checkboxes para seleccionar los operadores a utilizar. Me encargue de que una vez que el programa tiene los valores, este válida que sean entradas dentro del rango de valores permitidos y que no haya campos vacíos, en caso de haber algún valor fuera de rango o espacio en blanco se pasan valores por default al constructor de producerConsumer que es el que se encarga de correr el resto de clases. Para pasar el valor de los checkboxes de los operadores le asigne a cada uno de ellos un valor entero en caso de ser seleccionados, para la suma 1 para la resta 2 para la multiplicación 4 y la división 8, después sume los cuatro valores y el resultado lo transforme a byte de manera que pudiera enviarlo a traves del constructor de producerConsumer y este a su vez lo pase a producer para obtener los operadores a utilizar. Al final, ligué el botón detener a la función stop del producer consumer

El buffer se implementa como una Queue qué guarde los strings producidos. Use una queue ya qué tiene un sistema FIFO, lo cual hace qué el primer producto qué entra también sea el primero consumido. Otra parte qué realice fue la función produce, en donde se revisa primero si la queue está llena hacienda una comparación de su tamaño con el tamaño del buffer qué se dio en la interfaz para luego hacer un buffer.add y agregar el producto el cual es recibido como parámetro en la función, la función consume qué revisa si la queue está vacía para luego hacer buffer.poll y quitar el elemento en cabeza. Se usa wait() y notifyAll() para manejar el problema de varios consumidores intentando consumir el mismo producto. NotifyAll() se usa específicamente para despertar todas las threads y wait() para esperar a qué se vacíe o se llene la queue.

Al final me encargue principalmente de que nuestras tablas muestran la información de las tareas realizadas y tareas por hacer.

Se modificaron las clases de Producer Consumer y ProducerConsumer para que mandaran la información de la operación, el identificador del consumidor, productor y el resultado al modelo de JTable en la GUI.

También utilice la clase Buffer para que la barra de progreso represente a esta misma y muestra también el número de tareas realizadas.