I.T.C.R.

Departamento de Computación
Prof: Ing. Erika Marín Schumann.

Sistemas Operativos

Planificador de CPU
II Semestre del 2016

Objetivos del proyecto

- Aprender a desarrollar un esquema de cliente-servidor.
- Conocer la comunicación por medio de sockets.
- Desarrollar un programa donde se utilicen threads.
- Poner en práctica la teoría del Planificador de CPU

Definición general

Este primer proyecto tiene como finalidad desarrollar un simulador de planificador de CPU. Este simulador debe implementar los siguientes algoritmos: FIFO, SJF, HPF y Round Robin con un quantum especificado por el usuario.

Además el proyecto será con un esquema cliente servidor que enviará mediante sockets la información de los procesos a ejecutar.

Obviamente, el estudiante debe investigar sobre el protocolo de comunicación de los sockets.

Se contará con un cliente que funciona de 2 maneras: automático o manual. El modo manual leerá desde un archivo la información de procesos (PID, burst, prioridad). El modo automático creará un proceso con estos valores al azar. Por cada proceso se creará un hilo que debe conectarse via sockets al planificador de CPU. Este podrá estar en otra computadora, oyendo desde algún puerto definido.

El planificador de CPU recibirá la información de los procesos, los pondrá en una cola de espera y empezará la "ejecución" de los procesos. Apenas se tenga un proceso en la cola de espera ya se iniciará la simulación de la ejecución del mismo.

Se espera que el Planificador de CPU tenga al menos 2 hilos: uno que reciba los mensajes del socket y ponga los procesos en la cola (Job scheduler) y otro que seleccione cual proceso tendrá el CPU (Cpu scheduler) y simule al CPU ejecutando. La ejecución de un proceso debe ser de la duración real del burst.

El Planificador de CPU no debe dejar de funcionar hasta que uno se lo indique. Al finalizar debe desplegar información resumen de los procesos ejecutados.

Se debe tener cuidado, de no dejar procesos sin matar o sockets abiertos.

Descripción Detallada

- Cliente Manual: Recibirá de parámetro un archivo con una lista de procesos. Por cada uno de estos procesos debe crear un thread que le envié al planificador la información. Debe haber un tiempo de sleep entre la lectura en el archivo de cada uno de los procesos. Esa duración será un tiempo random. La información que tendrá el archivo de entrada es la siguiente: (ejemplo)

PID	BURST	Prioridad
1	8	3
2	7	2

- Cliente Automático: Debe funcionar igual que el anterior, con un thread por proceso y un tiempo random en la creación de cada uno de los procesos. La diferencia es que la información de los procesos no la tomará de un archivo sino la "inventará" poniendo valores random en el Burst (1-20) y la Prioridad (1-5). Este programa debe seguir creando archivos mientras nadie lo detenga.
- Una vez enviada la información de los procesos al planificador, los threads del cliente debe morir.
- Comunicación: La comunicación entre los clientes y el Simulador debe hacerse por medio de sockets.
- Simulador: Se debe seleccionar el algoritmo a correr(FIFO, SJF, HPF, Round Robin) debe tener al menos los siguientes procesos (hilos)
  - Job Scheduler: Será un hilo que reciba los mensajes de los sockets y los vaya guardando en algún tipo de lista.
  - CPU Scheduler: Constantemente deberá verificar si hay procesos en la cola. Si los hay debe ponerlos a ejecutar. Esto significa que según el algoritmo que se haya seleccionado, deberá escoger a alguien de la cola, "Ejecutarlo" (durar su burst) y seleccionar a alguien más, según el criterio del algoritmo.

        - Debe quedar claro que mientras se ejecutan procesos, otros irán llegando a la cola.
        - Cada vez que hay un context switch y un proceso se ejecutará, se debe desplegar en pantalla (ej: Proceso 1 con burst x y prioridad x entra en ejecución)
        - Cada vez que un proceso termina completamente su ejecución y deja de estar en espera debe desplegarlo en pantalla.
        - En cualquier momento de la ejecución el usuario puede consultar la cola. Solo debe desplegar los procesos que están en espera. No los que ya hayan terminado.
        - Se recomienda que tenga buen control del estado de los procesos.
        - Para el Round Robin, debe tomar en cuenta siempre lo que lleve ejecutado.
        - Para el FIFO el criterio de selección será el PID.
        - No se implementará ningún algoritmo apropiativo (salvo el RR)
        - Debe haber alguna forma de detener la simulación. Cuando esto ocurra, el Simulador debe desplegar la siguiente información resumen:
            - Cantidad de procesos ejecutados
            - Cantidad de segundos con CPU ocioso.
            - Tabla de TAT y WT para los procesos ejecutados
            - Promedio de Waiting Time
            - Promedio de Turn Around Time
    - Puede ser que necesiten otros threads, para llevar tiempos, o para obtener comandos (despliegue cola / detener) la simulación o para lo que ocupen. No hay problema.
- Diseñe bien la información que debe contener de cada proceso. Piense que tendrá un tipo de PCB para que el CPU Scheduler sepa a quien seleccionar.
- Dependiendo de cómo guarde la información de los procesos, pudiera necesitar un semáforo.

Documentación

Se espera que sea un documento donde especifique el análisis de resultados del programa junto con unos casos de pruebas. El Análisis debe resumir el resultado de la programación, qué sirve, qué no sirve y aspectos que consideren relevantes.

Para las pruebas se espera que definan claramente cada prueba, cuáles son los resultados esperados y cuáles fueron los resultados obtenidos. No es necesario que sean grandes pero deben evaluar la funcionalidad completa del programa.

Deben especificar, además, como compilar su tarea.

Aspectos Administrativos

- El desarrollo de este programa debe de realizarse en grupos de exactamente dos personas.
- El trabajo se debe de entregar antes del 21 de setiembre de 2016, por correo electrónico.
- Deben entregar el código fuente junto con el ejecutable.
- Se debe de entregar la documentación IMPRESA para ser calificado.