Sistemas Operativos: Proyecto semestral

Prof. Carlos Faúndez

2020 - 1

1 Descripción general

En este proyecto semestral se le pide aplicar los conocimientos de Estructura de Datos, planificación, creación, permutación y sincronización de procesos, y uso de señales.

Para esto debe implementar un programa en lenguaje C bajo el sistema operativo Linux (o cualquier distribución de Linux) que, para ciertos algoritmos de planificación, sea capaz de llevar acabo la ejecución de procesos encolados y así, todos los procesos tengan oportunidad de ejecutar en CPU.

2 Enunciado

Se le pide crear un programa en lenguaje C ejecutable en Linux (o alguna distribución de Linux) que se encargue de la planificación de procesos, es decir, el uso de CPU. Estos procesos que se van encolando, es un mismo programa llamado "loop"[1] que está junto a este enunciado.

Los algoritmos de planificación que se deben implementar son: **FIFO, Round Robin (RR) y Prioridades no expulsivo**.

Para todos los algoritmos de planificación, se necesita cuantos procesos se van a crear/encolar. Asuma que todos los procesos entran al mismo instante de tiempo y los tiempos de ráfaga son desconocidos, estos dependen del valor final del `for` en "loop" y la velocidad de su máquina.

El programa debe crear estos procesos y permutarlos a "loop". Además de tener un control sobre estos para decidir cuando se debe detener un proceso para que se pueda ejecutar otro, según el algoritmo de planificación correspondiente.

Para el caso de Round Robin, asegúrese que exista un valor de Quantum (Q), este valor es definido al momento de ejecutar su programa. Considere el valor del Quantum en segundos.

Para el caso de Prioridades no expulsivo, debe existir un arreglo de valores de prioridades definido al momento de ejecutar su programa.

[1] Programa que es un simple `for` que va iterando hasta un valor aleatorio.

3 Condiciones de implementación

1. **Argumentos por consola**: El algoritmo, el número de procesos y los demás parámetros dependiendo del algoritmo deben ser definidos por consola.

   Ejemplos:

   `miprograma -F 8`: Ejecuta FIFO con 8 procesos

   `miprograma -R 5 -Q 2`: Ejecuta RR con 5 procesos con Q = 2 seg.

   `miprograma -P 3 -p 2 5 1`: Ejecuta prioridades no expulsivo con 3 procesos con valores de prioridades 2, 5 y 1 respectivamente.

2. **Uso de Estructuras de datos eficiente**: Debe implementar al menos una estructura de datos en su programa y usarla como tal. Además del uso de al menos un algoritmo eficiente que opere sobre una estructura de datos.

3. **Uso de señales**: Debe limitarse en el uso de señales para tener control sobre los procesos.

4 Condiciones de informe

Su informe debe explicar a grandes rasgos que es lo que hace, como lo hace y como se usa. Además debe contestar las siguientes preguntas:

1. ¿Qué estructuras de datos utilizaron? ¿Por qué?
2. ¿Qué algoritmo eficiente utilizaron para alguna estructura de datos? ¿Para qué?
3. ¿Qué señales utilizaron para el control de los procesos? ¿En qué situaciones los utilizan?

5 Condiciones de entrega

1. **Fecha de entrega: Semana del 10 de agosto**
2. El equipo de trabajo es en parejas.
3. Debe entregar el informe y su código bien identado y comentado en un archivo comprimido `.zip`. La entrega se hace a través de Moodle, en la sección correspondiente. Con que uno de los integrantes lo entregue es suficiente.
4. Si existe copia o plagio, se clasificará con la nota mínima. Se pueden basar en artículos o foros de internet, pero no se acepta copias exactas.
5. Si implementa más de un código fuente (más de un archivo `.c` con sus respectivos `.h`) se recomienda el uso de `Makefile`