Investigación 02: Algoritmos de planificación de procesos

Maestro:

Javier Rosales Martinez

Materia:

Seminario de Solución de Problemas de Sistemas Operativos

Sección:

D06

Alumno:

Alejandro Covarrubias Sánchez

Código:

221350192

Planificación de procesos

La planificación de procesos es una función crucial dentro de los sistemas operativos. Su propósito es gestionar y controlar la ejecución de los procesos en un sistema, asegurando que el uso del procesador (CPU) sea eficiente y equitativo entre todas las tareas que lo requieren. Los procesos son programas en ejecución, y como generalmente hay varios procesos en un sistema, la planificación decide el orden en que estos se ejecutarán y por cuánto tiempo.

Los principales objetivos de la planificación de procesos son asegurar que el procesador esté ocupado la mayor parte del tiempo, garantizar que los procesos interactivos reciban atención rápidamente, reducir tiempos de espera, tiempos de ejecución de procesos, etc. y asegurar que ningún proceso quede injustamente ignorado.

Algoritmos de planificación

Para lograr la planificación más eficiente y equitativa, existen diferentes algoritmos que buscan determinar el orden de ejecución de los procesos tomando en cuenta diferentes parámetros. Los más comunes son FIFO, SFJ, Prioridades y Round Robin.

First In, First Out

Es uno de los algoritmos más simples de planificación. En FIFO o FCFS (First Come, First Served), los procesos se ejecutan en el orden en que llegan a la cola de procesos listos. El proceso que llega primero será el primero en recibir tiempo de CPU, y continuará ejecutándose hasta que termine o entre en estado de espera.

- Ventaja: Es sencillo de implementar.
- Desventaja: Puede tener un efecto negativo conocido como problema del convoy: si un proceso largo llega primero, hará esperar a los procesos más pequeños que llegaron después, lo que puede aumentar el tiempo de espera promedio.

Shortest Job First

El algoritmo de SJF (trabajo más corto primero) selecciona para su ejecución el proceso que tiene la duración más corta estimada. Es ideal para sistemas donde se conoce de antemano cuánto tiempo durará cada proceso.

- Ventaja: Minimiza el tiempo promedio de espera, ya que los procesos cortos se ejecutan antes, lo que generalmente reduce la cola de espera.
- Desventaja: No es adecuado para sistemas en tiempo real o cuando no se conoce la duración de los procesos de antemano. También sufre de inanición: los procesos largos pueden esperar indefinidamente si siempre llegan procesos más cortos.

Prioridad

En este algoritmo, cada proceso tiene asignada una prioridad, y el proceso con la prioridad más alta será el primero en ejecutarse. Para esto también deberá determinarse si se sigue una lógica ascendente o descendente, donde la prioridad más alta será la que tenga el número mayor o menor respectivamente. Las prioridades pueden ser asignadas estática o dinámicamente, y pueden estar basadas en diferentes factores, como la importancia del proceso o su tiempo de espera en cola.

- Ventaja: Es útil en sistemas donde algunos procesos tienen mayor importancia que otros, como en sistemas de tiempo real.
- Desventaja: Puede ocurrir un fenómeno conocido como inanición o hambre (starvation), donde un proceso de baja prioridad nunca recibe CPU si constantemente llegan procesos de mayor prioridad. Este problema se puede mitigar con una técnica llamada envejecimiento, donde se incrementa la prioridad de los procesos a medida que pasa el tiempo.

Round Robin

Este es un algoritmo más justo y equitativo, especialmente útil en sistemas interactivos. En Round Robin, cada proceso recibe una cantidad fija de tiempo, llamada quantum de tiempo o time slice. Si el proceso no termina durante ese quantum de tiempo, es interrumpido y colocado al final de la cola de procesos listos, mientras el siguiente proceso en la cola recibe su turno de ejecución.

Características clave:

- Ventaja: Asegura que todos los procesos reciban tiempo de CPU de manera equitativa, lo que reduce la espera de procesos interactivos.
- Desventaja: Si el quantum de tiempo es demasiado pequeño, el sistema puede gastar mucho tiempo cambiando de proceso (lo que se conoce como overhead de conmutación de contexto). Si es demasiado largo, se convierte en un algoritmo similar a FIFO.

Bibliografía

Departamento de Ciencias e Ingeniería de la Computación, UNS. (2020). *PLANIFICACIÓN DE PROCESOS*. Universidad Nacional del Sur. https://cs.uns.edu.ar/~so/data/apuntes/SO-2020-mod%2007.pdf

Vera Amaro, G. H. (n.d.). *2.4 Planeación de procesos*. Instituto Consorcio Clavijero. https://cursos.clavijero.edu.mx/cursos/182_so/modulo2/contenidos/tema2.4.3.html

Wolf, G. (2016). *Planificación de procesos: Algoritmos de planificación*. Sistemas Operativos - Gunnar Wolf. https://sistop.gwolf.org/html/03_planificacion_de_procesos.html