1. Scheduling o Planificación de Procesos

1.1. Scheduling o Planificación de Procesos

time slice o time quantum: período de tiempo que el kernel le otorga a un proceso. Para S.O de tipo Time Sharing.

1.2. Time Sharing

Los Sistemas Operativos llamados **time sharing** surgen de la idea de los programadores de tener *"toda una computadora para uno mismo"*.

Planificador o Scheduler: Es un componente del sistema operativo que se encarga de decidir qué proceso se ejecuta en cada momento.

1.3. Números y el Workload

El **Workload** es carga de trabajo de un proceso corriendo en el sistema.

1.4. Métricas de Planificación

políticas de planificación o scheduling

1. **Rurnaround time:** tiempo en el cual el proceso se completa menos el tiempo de arribo al sistema:

$$T_{Turnaround} = T_{CompletionTime} - T_{ArrivalTime} \tag{1}$$

2. **Response time:** tiempo que tarda el sistema en responder a una solicitud:

$$T_{Response} = T_{FirstResponse} - T_{ArrivalTime} \tag{2}$$

1.5. Políticas Para Sistemas Mono-procesador

Se estudiarán las políticas de planificación para un sistema que posea un solo procesador o CPU con un solo núcleo de procesamiento.

1. **First In, First Out (FIFO)**
2. **Shortest Job First (SJF)**
3. **Shortest Time-to-Completion (STCF)**
4. **Round Robin (RR)**
5. **Multi-Level Feedback Queue (MLFQ)**

First In, First Out (FIFO)

Este algoritmo asigna la CPU al proceso que llegó primero y lo ejecuta hasta que termina o se bloquea. Luego, el siguiente proceso en la cola es seleccionado y se le asigna la CPU.

Shortest Job First (SJF)

Este algoritmo asigna la CPU al proceso que tiene el menor tiempo de ejecución. Se ejecuta el proceso de duración mínima, una vez finalizado esto se ejecuta el proceso de duración mínima y así sucesivamente.

Shortest Time-to-Completion (STCF)

En este caso el planificador se adelanta y si hay un proceso que puede terminar antes, ejecuta ese primero, posponiendo la ejecución del que se estaba ejecutando. Esto lo realiza con un context. switch. Es pre-empitve.

Round Robin (RR)

La idea del algoritmo es bastante simple, se ejecuta un proceso por un período determinado de tiempo **(slice)** y transcurrido el período se pasa a otro proceso, y así sucesivamente cambiando de proceso en la cola de ejecución [Round Robin Paper].

Con este método de planificación, la métrica de *response* mejora, pero la de *turnaround* empeora, ya que se atrasa el retorno de todos los programas.

1.6. Multi-Level Feedback Queue (MLFQ)

Multi-Level Feedback Queue (MLFQ) es un algoritmo de planificación de procesos en sistemas operativos. Este algoritmo utiliza múltiples colas de prioridad para asignar la CPU a los procesos. Los procesos se colocan en la cola de prioridad más baja al llegar y se ejecutan hasta que se bloquean o terminan. Si un proceso no se completa en la cola de prioridad más baja, se mueve a la siguiente cola de prioridad más alta.

MLQF intenta atacar principalmente 2 problemas:

- Intenta optimizar el turnaround time mediante la ejecución de la tarea mas corta primero.(No sabemos cual es)
- Intenta hacer que el sistema tenga un tiempo de respuesta interactivo para los usuarios.(Minimizar el response time)

1.7. MLQF: Las reglas básicas

Las 2 reglas básicas de MLFQ:

- **REGLA 1:** si la prioridad (A) es mayor que la prioridad de (B), (A) se ejecuta y (B) no.
- **REGLA 2:** si la prioridad de (A) es igual a la prioridad de (B), (A) y (B)se ejecutan en Round-Robin.
- **REGLA 3:** Cuando un proceso entra en el sistema, se le asigna la prioridad más alta.
- **REGLA 4:** Una vez que una tarea usa su asignación de tiempo en un nivel dado (independientemente de cuantas veces haya renunciado al uso de la CPU) su prioridad se reduce: (Por ejemplo baja un nivel en la cola de prioridad)
- **REGLA 4a:** Si una tarea usa un time slice mientras se esta ejecutando su prioridad se reduce de una unidad (baja la cola una unidad menor)
- **REGLA 4b:** Si una tarea renuncia al uso de la CPU antes de un *time slice* completo se queda en el mismo nivel de prioridad.
- **REGLA 5:** Después de cierto periodo de tiempo S, se mueven las tareas a la cola con mas prioridad.

La clave para la planificación MLFQ subyace entonces en cómo el planificador setea las prioridades. En vez de dar una prioridad fija a cada tarea, MLFQ varia la prioridad de la tarea basándose en su comportamiento observado REGLA 3, 4a, 4b.

PROBLEMA Con este Approach de MLFQ **Starvation o indefiniciónes**: Si una tarea es muy larga, puede que nunca llegue a ejecutarse. Para solucionar esto se utiliza la REGLA 5.

1.8. Planificación: Proportional Share

La planificación por proporcional share es un tipo de planificación de procesos que trata de garantizar que cada proceso obtenga un cierto porcentaje de tiempo de CPU según su prioridad. Una forma de implementar este tipo de planificación es mediante el lottery scheduling, que consiste en realizar sorteos periódicos para determinar qué proceso debe ejecutarse a continuación; los procesos que deben ejecutarse más a menudo deben tener más posibilidades de ganar el sorteo.

La ventaja de este método es que utiliza la aleatoriedad para evitar casos extremos, reducir el estado necesario y acelerar la decisión. La desventaja es que no garantiza un reparto exacto del tiempo de CPU, sino solo probabilístico.