Práctica 04: Algoritmos de Planificación 2

Maestro:

Javier Rosales Martinez

Materia:

Seminario de Solución de Problemas de Sistemas Operativos

Sección:

D06

Alumno:

Alejandro Covarrubias Sánchez

Código:

221350192

Antecedentes

Los algoritmos de planificación son métodos utilizados en sistemas operativos para gestionar cómo y cuándo se ejecutan los procesos, o tareas, en un procesador. Su principal objetivo es distribuir eficientemente el tiempo de CPU entre los procesos para maximizar el rendimiento del sistema.

Existen varios tipos de algoritmos de planificación, como el algoritmo de Prioridad, que ejecuta los procesos según su importancia, pero puede causar que los de baja prioridad esperen indefinidamente. Por otro lado, Round Robin mejora la equidad asignando a cada proceso un tiempo fijo para ejecutarse antes de pasar al siguiente, asegurando que todos reciban atención, aunque puede no ser óptimo para procesos cortos.

Metodología

El programa está escrito en Python e implementa y simula cuatro algoritmos de planificación de procesos: FIFO (First-In, First-Out) y SJF (Shortest Job First), descritos en la práctica anterior y añadiendo Prioridad y Round Robin. La solución está diseñada para leer un archivo que contiene la información de los procesos y luego “ejecutarlos” utilizando cada uno de los algoritmos. En este caso, la “ejecución” es la impresión en pantalla del proceso, mostrando los tiempos de inicio y finalización de cada proceso. No se utilizaron librerías externas en este programa, solo funciones básicas del lenguaje Python, como la función `open()` para manejar archivos.

La solución está diseñada con una función independiente para cada algoritmo de planificación. La función principal `main()` gestiona la lectura de los datos de procesos desde un archivo de texto, añadiendo cada línea a una lista de procesos donde cada uno es una tupla con tres valores: el nombre del proceso, su duración y su prioridad y luego ejecuta cada algoritmo para observar y comparar los diferentes tiempos de inicio y finalización dependiendo del algoritmo utilizado.

`priority_scheduling()` implementa la planificación por prioridad. Similar a la función por SJF de la práctica 03, pero los procesos son ordenados por su prioridad antes de ejecutarse. Los procesos con prioridad más baja (valor numérico más pequeño) se ejecutan primero.

```
def priority_scheduling(processes):
  """
  Ordena la lista de procesos según su prioridad, y los ejecuta mostrando el tiempo de inicio y finalización
  """
  current_time = 0
  processes_sorted = sorted(processes, key=lambda x: x[2])

  for process in processes_sorted:
    start_time = current_time
    end_time = current_time + process[1]
    print(f"{process[0]}\t| Inicio: {start_time}s | Fin: {end_time}s | Prioridad: {process[2]}")
    current_time = end_time
```

Por último, `round_robin()` recibe un "quantum" o tiempo fijo de ejecución por proceso como parámetro. En esta función, los procesos se ejecutan por intervalos de tiempo, si el tiempo de ejecución del proceso es mayor al quantum, se reintroduce en la cola de ejecución con el tiempo restante.

```
def round_robin(processes, quantum):
  """
  Ejecuta cada proceso únicamente durante el tiempo indicado, hasta finalizarlos.
  La lista no se ordena antes de procesarse.
  """
  current_time = 0
  ready_queue = processes.copy()

  while ready_queue:
    process = ready_queue.pop(0)
    name, duration, _ = process
    if duration <= quantum:
      start_time = current_time
      end_time = current_time + duration
      print(f"{process[0]}\t| Inicio: {start_time}s | Fin: {end_time}s")
      current_time = end_time
    else:
      start_time = current_time
      end_time = current_time + quantum
      print(f"{process[0]}\t| Inicio: {start_time}s | Fin: {end_time}s")
      current_time = end_time
      ready_queue.append((name, duration - quantum, _))
```

Conclusión

Utilizando como base el programa de la práctica anterior, este implementa la simulación del resto de los principales algoritmos de planificación de procesos en sistemas operativos, leyendo un archivo de texto que contiene información sobre procesos (nombre, duración y prioridad) y simulando su ejecución.

Igual que con el programa de la práctica anterior, las funciones de simulación y la lectura de datos desde el archivo funcionan correctamente. Sin embargo, aún se pueden implementar mejoras, tanto a las funciones de planificación, como mejorar Round Robin para calcular tiempos de espera y de retorno, como al resto del programa, añadiendo visualizaciones más detalladas, simular entornos multiprocesador, o agregar una interfaz de usuario.

Referencias

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