Informe

Laura Isabel Olivero

3/04/2025

1. Introducción

El siguiente informe describe en detalle el funcionamiento de un código en C++ que implementa un sistema de planificación de procesos utilizando múltiples colas (MLQ). El programa simula la ejecución de procesos mediante algoritmos de planificación Round Robin (RR) y First-Come-First-Served (FCFS), permitiendo la administración y cálculo de tiempos.

2. Estructura General del Código

El programa se organiza en varias clases y funciones principales:

- **Clase** `Process`: Representa un proceso y almacena atributos como el tiempo de llegada, tiempo de ejecución (BrushTime), prioridad, y otros tiempos relevantes para el análisis (tiempo de espera, tiempo de respuesta, etc.).
- **Clase** `Queue`: Implementa una cola de procesos. Admite dos modos de ejecución:
  - **Round Robin (RR)**: Utiliza un quantum de tiempo específico para simular la ejecución de procesos en forma cíclica.
  - **First-Come-First-Served (FCFS)**: Ejecuta los procesos de acuerdo con el orden de llegada, sin interrupciones.
- **Clase** `MLQ`: Administra múltiples colas de procesos (en este caso, dos de Round Robin y una de FCFS). Coordina la ejecución secuencial de las colas y calcula promedios de los distintos tiempos (espera, finalización, respuesta y turnaround).

- **Función** `main`: Se encarga de leer un archivo de entrada con la información de los procesos, asignarlos a la cola correspondiente según un parámetro (cola), y finalmente ejecutar el sistema de planificación. Al finalizar, genera un archivo de salida (`salida.txt`) con los resultados de la ejecución y los promedios calculados.

3. Descripción Detallada de las Clases

3.1. Clase `Process`

La clase `Process` encapsula la información de cada proceso:

- **Atributos:**
  - `arrivalTime`: Tiempo de llegada del proceso.
  - `BrushTime`: Tiempo de ejecución restante.
  - `originalBT`: Tiempo de ejecución original (para calcular métricas).
  - `cola`: Identifica la cola a la que pertenece el proceso.
  - `priority`: Prioridad del proceso.
  - `waitingTime, turnAroundTime, completeTime, runTime`: Tiempos calculados durante la simulación.
  - `id`: Identificador del proceso.
- **Métodos:**
  - Métodos `get...` para obtener los atributos.
  - `actRun(int a)`: Registra el tiempo de inicio de ejecución.
  - `actBrush(int a)`: Actualiza el tiempo de ejecución restante.
  - `actCT(int a)`: Actualiza el tiempo de finalización y calcula los tiempos de turnaround y espera.

3.2. Clase `Queue`

La clase `Queue` administra una cola de procesos y ofrece dos modos de planificación:

- **Atributos:**
  - `procesos`: Cola de procesos en espera.

- `finalizado`: Cola de procesos que han finalizado su ejecución.
- `roundRobin`: Bandera para determinar el tipo de algoritmo.
- `quantum`: Tiempo máximo de ejecución para cada proceso en Round Robin.
- `Tfin`: Tiempo total al finalizar la cola.

- **Métodos:**
  - `agregarProceso(Process p)`: Agrega un proceso a la cola.
  - `ejecutar(int time)`: Decide qué algoritmo utilizar (RR o FCFS) basado en la bandera `roundRobin`.
  - `RR(int time)`: Implementa el algoritmo Round Robin. Cada proceso se ejecuta por un tiempo igual al mínimo entre el quantum y el tiempo de ejecución restante; si no termina, se vuelve a colocar en la cola.
  - `FCFS(int time)`: Ejecuta los procesos en orden de llegada sin interrupciones, es decir, hasta que finalice completamente.

3.3. Clase `MLQ`

La clase `MLQ` representa la estructura de múltiples colas:

- **Atributos:**
  - `colas`: Vector de colas de procesos.
  - Variables para acumular los tiempos para el cálculo de promedios.
- **Métodos:**
  - `agregarColas(Queue p)`: Agrega una cola al vector.
  - `ejecutar()`: Ejecuta cada cola de manera secuencial, actualizando el tiempo global.
  - `promedios(ostream &out)`: Calcula y muestra los promedios de los tiempos clave.
  - `mostrar(ostream &out)`: Muestra la información de cada proceso finalizado.

4. Flujo del Programa y Lógica de Ejecución

4.1. Lectura del Archivo de Entrada

La función `main`:

- Verifica que se haya pasado el nombre del archivo de entrada.
- Abre y lee el archivo línea por línea.
- Cada línea contiene los parámetros de un proceso.
- Según el parámetro que indica la cola, el proceso se agrega a la cola correspondiente: dos colas con algoritmo Round Robin (con quantum 3 y 5) y una cola con FCFS.

4.2. Ejecución del Sistema de Planificación

Posteriormente:

- Se crea una instancia de la clase `MLQ` y se agregan las colas en el orden deseado.
- Se ejecuta el sistema de planificación; cada cola se procesa secuencialmente y se actualiza el tiempo global.
- Cada proceso se evalúa según el algoritmo de su respectiva cola y se actualizan sus tiempos.

4.3. Salida de Resultados

Finalmente:

- Se genera un archivo de salida (`salida.txt`) donde se imprime:
  - La información de cada proceso (identificador, tiempos originales, tiempos de llegada, cola, prioridad, tiempos calculados, etc.).
  - Los promedios del WT, CT, RT y TAT.

5. Conclusión

El código implementa un sistema de planificación de procesos basado en múltiples colas (MLQ) utilizando algoritmos Round Robin y FCFS. Se destaca la lectura estructurada de los datos de entrada, la asignación de procesos a distintas colas según un parámetro, y la simulación de la ejecución con cálculo de métricas de rendimiento.