RELATÓRIO - SIMULADOR DE GERÊNCIA DE PROCESSOS

Daniel Brenner Seitenfus

1 IMPLEMENTAÇÃO DAS FILAS

A implementação das filas foi realizada com base na apostila dos profs. Celes e Rangel. Foi utilizado o conceito de listas simplesmente encadeadas, com a criação de duas estruturas: *No* e *Fila*.

1.1 No

Responsável por definir cada elemento da lista, possuindo um ponteiro para o processo (Process *process) e um ponteiro para o próximo nó (no* prox).

1.2 Fila

A estrutura Fila armazena os ponteiros para os nós de início e fim. É necessário manter essas referências pois a manipulação das filas, em especial a fila FIFO, requer a inserção de um novo elemento no final da fila e a remoção do elemento que está no início.

1.3 Métodos

Como já mencionado, a fila implementada implementa o conceito FIFO, então alguns métodos são necessários para isso ser possível, são eles: *criar, inserir, retirar, vazia* e *possui*.

1.3.1 Função “criar”

A função criar inicializa uma nova fila, alocando memória para a estrutura Fila e definindo os ponteiros início e fim como NULL.

1.3.2 Função “inserir”

A função inserir adiciona um novo processo ao final da fila. Ela aloca memória para um novo nó, define o ponteiro *process* para o processo a ser inserido e ajusta os ponteiros da fila de forma que o novo nó seja o último.

1.3.3 Função “retirar”

A função retirar remove e retorna o primeiro processo da fila. Se a fila não estiver vazia, ela atualiza o ponteiro de início para o próximo nó e, se necessário, ajusta o ponteiro de fim.

1.3.4 Função “vazia”

A função vazia verifica se a fila está vazia, retornando verdadeiro se o ponteiro de início for NULL.

1.3.4 Função “possui”

A função possui verifica se um determinado processo está presente na fila, percorrendo todos os nós e comparando os ponteiros dos processos.

2 IMPLEMENTAÇÃO DO MODELO DE PROCESSO

A estrutura PCB contém os seguintes campos:

- **pid:** identificador único do processo.
- **tInicio:** tempo de início do processo.
- **pc:** contador de “programa”, indicando a posição atual no ciclo de execução.
- **cyclesLength:** tamanho do array de ciclos entre CPU e I/O.
- **cycles:** array de inteiros representando os ciclos de CPU e I/O.
- **state:** estado atual do processo.
- **tDevice:** tempo restante no dispositivo.
- **deviceTime:** array que acumula o tempo gasto em cada dispositivo.
- **waitingTime:** tempo total em que o processo esteve aguardando na fila de prontos.
- **tEnd:** tempo em que o processo terminou.

3 ESTRUTURA DO SIMULADOR

3.1 Funções Principais

3.1.2 init

Função de inicialização que lê os dados de entrada, cria os processos e dispositivos, e inicia a simulação.

3.1.2 run

Função principal que executa o loop de simulação, onde cada iteração representa um ciclo de execução.

3.1.3 scheduleLongTerm

Responsável pelo escalonamento a longo prazo, verificando a inserção na fila ready para os processos que se encontram em estado “new/ready”; as filas de dispositivos e a fatia de tempo da CPU.

3.1.4 scheduleShortTerm

Implementa o escalonador de curto prazo, selecionando o próximo processo a ser executado.

3.1.4 handleDevices

Gerencia o funcionamento dos dispositivos, incluindo a movimentação de processos entre estados e filas.

3.2 Procedimentos executados em cada ciclo de execução

A simulação avança em ciclos discretos, onde cada ciclo representa uma unidade de tempo do CPU. Em cada ciclo, o simulador executa, de forma resumida, os seguintes procedimentos:

3.2.1. Verificação de novos processos checkNewProcesses

Verifica se há processos que devem iniciar nesse ciclo (com base em tInicio). Processos com tInicio igual ao tempo atual (tCPU) são movidos para o estado NEW_READY.

3.2.2. Escalonamento a Longo Prazo “scheduleLongTerm”

Admite processos do estado *NEW_READY* para a fila de prontos (*readyQueue*), mudando seu estado para *READY*. Verifica a fila de dispositivos, liberando aqueles que já foram atendidos. Verifica se o processo em execução (*scheduledProcess*) completou seu ciclo de CPU atual.

3.2.3 Escalonamento a Curto Prazo “scheduleShortTerm”

Se não há processo escalonado, seleciona o próximo processo da fila de prontos (readyQueue) para executar.

2.4. Execução do processo atual

Se há um processo em execução (scheduledProcess), decrementa o tempo restante do ciclo de CPU atual (scheduledProcess->cycles[scheduledProcess->pc]--). Incrementa o timeSlice. Verifica se o timeSlice atingiu o limite: Se sim, preempta o processo, mudando seu estado para READY e reinserindo-o na fila de prontos. Atualiza o scheduledProcess e o timeSlice.

Se não há processo em execução: Incrementa o tempo de CPU ocioso (cpuIdleTime).

2.5. Gerenciamento dos dispositivos handleDevices:

Para cada dispositivo: Se o dispositivo está ocioso e há processos na sua fila, inicia o atendimento do próximo processo: Remove o processo da fila do dispositivo (device->queue) e Configura o tempo de atendimento (tDevice) com base em attendanceTime.

Se o dispositivo está atendendo um processo: Decrementa o tempo restante de atendimento (tDevice--).