Relatório
Algoritmos de Substituição de Páginas
Sistemas Operacionais I

Universidade Federal de Santa Catarina - UFSC

William Silveira Figueiredo
21203999
Yuiti Kanekiyo Leite
21202127

Introdução

O gerenciamento de memória é um desafio para Sistemas Operacionais, em especial, a substituição adequada de páginas quando a memória física está cheia. O mau gerenciamento pode resultar em baixo desempenho ou falhas do sistema.

Nesse contexto, este relatório detalha a criação de um programa que emula algoritmos de substituição de páginas. O software foi desenvolvido utilizando linguagem C++ seguindo os princípios de SOLID da Programação Orientada a Objetos, e implementa três algoritmos: FIFO (First In, First Out), LRU (Least Recently Used) e OPT (Algoritmo Ótimo). Para sua execução, são necessários dois parâmetros: o número de quadros disponíveis na RAM e um arquivo de texto com o mapeamento de páginas referenciadas. Ao final do processamento, o programa fornece um resultado contendo o número de quadros utilizados, os acessos à memória e as faltas de páginas para cada algoritmo.

FIFO (First In, First Out)

O algoritmo FIFO baseia-se no princípio de que a primeira página que entrou na memória será a primeira a ser substituída. Na implementação, foi utilizada uma fila que opera no princípio “primeiro a entrar, primeiro a sair” a fim de rastrear a ordem em que as páginas foram inseridas na memória. Sempre que uma página é inserida na memória, ela também é inserida no final da fila. E sempre que é necessário remover uma página da memória (porque a memória está cheia), a página a ser removida é sempre a do início da fila, que é a que está na memória há mais tempo.

Este método garante que as páginas sejam removidas na ordem em que foram inseridas. Abaixo seguem os diagramas necessários para auxiliar na compreensão de tal comportamento.

LRU (Least Recently Used)

O LRU é baseado em timestamps (ou último checkpoint). A ideia principal é associar o checkpoint de tempo a cada página no instante atual sempre que ela é referenciada. O checkpoint de tempo indica a idade ou a última vez que ela foi referenciada. Assim, cada referência de página recebe uma marcação do tempo a cada vez que ela é referenciada. O mapa timestamps armazena os checkpoints de tempo mais recentes de cada página. Portanto, sempre que uma página é referenciada, independente de ser uma falta de página ou não, o checkpoint de tempo dessa página é atualizado com o valor de tempo do instante atual.

Quando ocorre uma falta de página e a memória está cheia, é necessário substituir a página que foi usada menos recentemente. Isso é feito analisando o mapa timestamps. Assim, busca-se a página com o checkpoint de tempo mais antigo (ou seja, página que foi referenciada há mais tempo). Logo que a página é identificada, ela é removida de memória física e seu checkpoint de tempo removido do meu mapa timestamps.

Este método garante que as páginas sejam substituídas com base no uso mais recente, garantindo que as páginas ativamente usadas permaneçam na memória física, enquanto as páginas menos recentemente usadas sejam candidatas à substituição. Abaixo seguem os diagramas necessários para auxiliar na compreensão de tal comportamento.

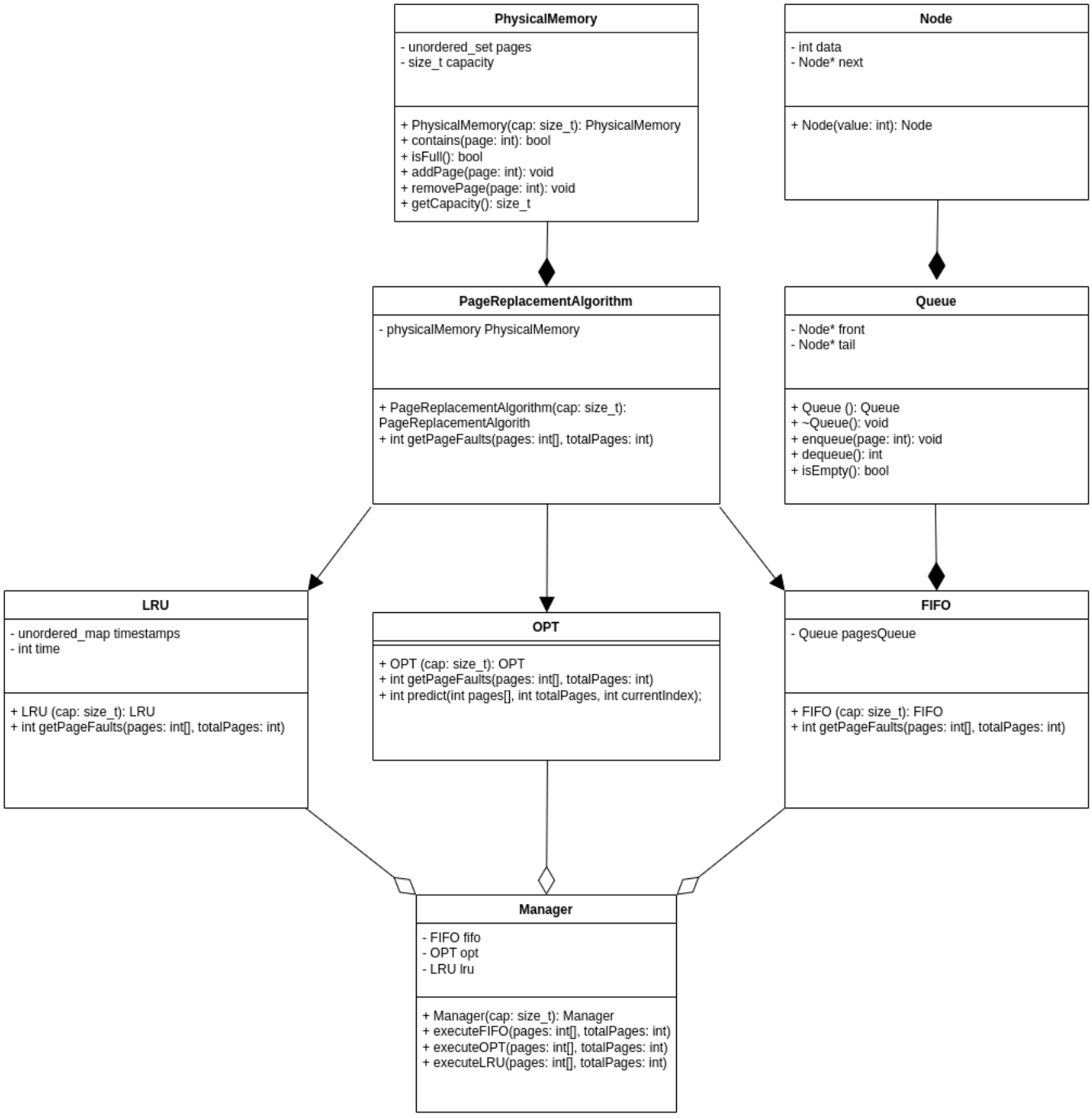
OPT (Algoritmo Ótimo)

O algoritmo é baseado na previsão no futuro de páginas que vão ser referenciadas. A ideia principal é substituir a página que não será usada pelo maior período de tempo no futuro. Embora seja impossível

prever o comportamento exato de um programa em execução, este algoritmo serve como uma referência para comparação a fim de avaliar outros algoritmos de substituição de páginas. A implementação realizada no projeto consiste em dois métodos principais: pesquisa de “page faults” e predição da melhor página a ser substituída no futuro próximo.

Quando uma falta de página ocorre, é necessário prever qual a melhor página a ser substituída no futuro através da seleção da página que será referência depois de todas as outras páginas na memória ou que não será mais referenciada novamente nos acessos de página futuro. Se uma página nunca vai ser referenciada no futuro, ela é selecionada imediatamente. Porém, se todas as páginas na memória física forem referenciadas no futuro, a removida será aquela referenciada o mais tarde possível nos próximos acessos. Abaixo seguem os diagramas necessários para auxiliar na compreensão de tal comportamento.

Diagrama de Classe



1. PhysicalMemory
    - Representa a memória física.

    - Atributos:
      > pages: conjunto não ordenados que contém as páginas
      > capacity: capacidade da memória física

    - Métodos:
      > contains: checa se a memória contém a página.
      > isFull: verifica se a memória está cheia.
      > addPage: adiciona a página à memória.
      > removePage: remove a página da memória.
      > getCapacity: pega a capacidade de frames suportada pela memória.

2. PageReplacementAlgorithm
    - Representa um algoritmo genérico e encapsula a memória física para ser utilizada na implementação de outros algoritmos.

    - Atributos:
      > physicalMemory: representação da memória física.

    - Métodos:
      > getPageFaults: abstração de método que calcula falhas de acesso à páginas na memória física.

3. LRU
    - Algoritmo com a substituição da página menos usada recentemente.

    - Atributos:
      > timestamps: mapa com checkpoints de tempo de quando cada página foi acessada pela última vez.

    - Métodos:
      > getPageFaults: método que calcula falha de acesso às páginas na memória física.
      > predict: prevê qual a melhor página a ser removida da memória física com base na próxima.

4. OPT
    - Algoritmo ótimo realiza as melhores substituições possíveis através da remoção de páginas da memória física que vão demorar mais a ser citadas novamente.

    - Métodos:
      > getPageFaults: método que calcula falha de acesso às páginas na memória física.

5. FIFO
    - O algoritmo de substituição de páginas FIFO ubstitui as páginas que foram adicionadas primeiro na memória física.

    - Atributos:
      > pagesQueue: fila referência para gerenciar a ordem que páginas foram adicionadas na memória física.

    - Métodos:
      > getPageFaults: método que calcula falha de acesso às páginas na memória física.

6. Queue
    - Estrutura de dados de lista ligada.

    - Atributos:
      > front: referência para o primeiro elemento da fila.
      > tail: referência para o último elemento da fila.

    - Métodos:
      > getPageFaults: método que calcula falha de acesso às páginas na memória física.
      > enqueue: adiciona um novo nó no final da fila.
      > dequeue: remove o nó item da fila.
      > isEmpty: verifica se a fila está vazia.

7. Node
    - Nó da fila.

    - Atributos:
      > data: o dado armazenado no nó.
      > next: referência ao próximo nó na fila.

8. Manager
    - O gerenciador controla a execução dos múltiplos algoritmos de substituição de páginas.

    - Atributos:
      > fifo: algoritmo FIFO.
      > opt: algoritmo ótimo.
      > lru: algoritmo LRU.

    - Métodos:
      > executeFIFO: método para execução do algoritmo FIFO.
      > opt: método para execução do algoritmo algoritmo ótimo.
      > lru: método para execução do algoritmo algoritmo LRU.