Diogo Rodrigues - 2380242
Marcos Vinicius de Quadros - 2380560

Algoritmos de Substituição de Páginas

Relatório técnico do projeto solicitado pelo professor Rodrigo Campiolo na disciplina de Sistemas Operacionais do Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Universidade Tecnológica Federal do Paraná – UTFPR

Departamento Acadêmico de Computação – DACOM

Bacharelado em Ciência da Computação – BCC

Campo Mourão

Julho / 2023

Sumário

1 Introdução 3
2 Atividade Proposta 3
3 Implementação 3
3.1 Como Compilar e Executar 3
3.2 Representação da Página 4
3.3 Arquivo de Configuração 4
3.4 NRU - Not Recently Used 5
3.5 FIFO - (First in, First out) 5
3.6 Segunda Chance 5
3.7 Estruturas Auxiliares 6
4 Conclusões 6
5 Referências 6

1 Introdução

A eficiente utilização da memória é um dos principais objetivos do sistema operacional ao lidar com a gerência de memória. Nesse contexto, a técnica de memória virtual é empregada, permitindo que apenas uma parte do processo esteja carregada na memória, viabilizando a execução de processos maiores que a capacidade de memória disponível sem maiores problemas. Um requisito essencial para o funcionamento dessa técnica é a implementação de algoritmos de substituição de páginas, responsáveis por selecionar qual página será substituída, liberando seu espaço no quadro de memória.

Este trabalho tem como objetivo principal a implementação dos algoritmos utilizados pelo sistema operacional, com foco específico nos algoritmos de substituição de páginas FIFO (First In, First Out), NRU (Not Recently Used) e o algoritmo da Segunda Chance.

2 Atividade Proposta

Com base no conteúdo apresentado durante as aulas teóricas, foi proposta a atividade de desenvolvimento de um simulador de memória virtual e algoritmos de substituição. O simulador oferece a opção de escolha entre três algoritmos e, com base nos dados fornecidos como entrada, realiza os cálculos e procedimentos necessários para simular a memória virtual. Os dados de entrada incluem o tamanho da RAM, o tamanho das páginas, o tamanho do processo (sendo este maior que a RAM), o algoritmo a ser simulado e o processo em si, que consiste em uma sequência de operações e endereços a serem processados (variando de 0x0 até o tamanho do processo)

3 Implementação

3.1 Como Compilar e Executar

Para compilar o programa, abra o prompt de comando no diretório onde se encontra o arquivo 'main.c' e execute o comando 'make'. Esse comando realizará a compilação do arquivo e gerará um executável denominado 'main.exe'. Para executar o programa, digite no prompt de comando "./main.exe ", seguido do arquivo de configurações no padrão informado na seção 3.3. A Figura 1 é um exemplo de compilação e execução.

```
diogo@ubuntu:~/Documentos/PROJETOS/SO_PROJETO_FINAL/arquivos$ make
gcc main.c -o main.exe
diogo@ubuntu:~/Documentos/PROJETOS/SO_PROJETO_FINAL/arquivos$ ./main.exe second_entrada.conf
```

Figura 1 – Exemplo de Compilação

3.2 Representação da Página

Para representar as páginas de um endereço lógico de forma mais conveniente no desenvolvimento do simulador, foi utilizado uma struct. Essa estrutura contém as informações necessárias para a execução dos algoritmos, que são: o número do frame (caso a página não esteja carregada, é atribuído o valor -1), o bit M (que indica se uma página foi modificada recentemente e é utilizado pelo algoritmo NRU), o bit R (que indica se uma página foi acessada recentemente e é utilizado pelo algoritmo NRU e pela técnica de Segunda Chance), e o bit V (que indica se uma página está em algum frame e é utilizado pelos três algoritmos).

3.3 Arquivo de Configuração

Para que os algoritmos de substituição funcionem corretamente, é necessário que as configurações sejam fornecidas por meio de um arquivo ".conf", como mostra a Figura 2.

```
./main.exe second_entrada.conf
```

Figura 2 – Exemplo de Arquivo de Configuração

A Figura 3 ilustra um exemplo de como o arquivo de configuração deve ser formatado, garantindo que a ordem dos parâmetros seja preservada, conforme exemplificado no arquivo de configuração de referência. É importante seguir essa estrutura para garantir que todas as configurações sejam interpretadas corretamente pelo sistema.

implementação > entrada3.conf

```
1  page_size=8
2  ram_size=16
3  size_process=32
4  algoritmo=3
5  2 0
6  2 4
7  2 8
8  2 c
9  1 0
10 2 10
11 2 13
12 2 4
13 1 0
14 2 8
```

Figura 3 – Exemplo de entrada

3.4 NRU - Not Recently Used

A função desse algoritmo é substituir a página com a classe mais baixa. As classes são divididas em:

- Classe 0: bit R=0 e bit M=0
- Classe 1: bit R=0 e bit M=1
- Classe 2: bit R=1 e bit M=0
- Classe 3: bit R=1 e bit M=1

Para a implementação do algoritmo, foram definidas três etapas distintas. A primeira etapa consiste em verificar a disponibilidade de um quadro livre na memória. Caso exista um quadro livre, a página é inserida nesse quadro. A segunda etapa é executada caso não haja quadros livres disponíveis. Nessa etapa, é realizada uma análise das páginas presentes na memória para determinar qual delas possui a menor classe. Em seguida, essa página é removida da memória. A terceira e última etapa do algoritmo é responsável por adicionar a nova página na memória.

O algoritmo utiliza uma função chamada "update-bits()"que tem como objetivo zerar os bits R após um determinado número de acessos, a fim de simular a interrupção de relógio. Esse intervalo é definido pela variável "refresh-interval"declara na função main.

3.5 FIFO - (First in, First out)

Consiste em remover a página mais antiga. Para que isso seja possível, o algoritmo mantém uma **fila** das páginas acessadas, facilitando a remoção da mais antiga (primeira da fila) e inserção da nova página (última da fila).

Para a implementação do FIFO, dividimos suas funções em 3 etapas.

- 1° - Verifica se existe um quadro livre na memória e, caso exista, adiciona a página nesse quadro.
- 2° - Caso a primeira etapa não seja verdadeira, remove a página que esta na primeira posição da fila.
- 3° - Adiciona a nova página ao final da fila.

3.6 Segunda Chance

Consiste em remover a página mais antiga com bit R=0. Para facilitar essa organização de páginas, o algoritmo utiliza uma lista encadeada, adicionando a página acessada recentemente ao final da lista. Esse algoritmos foi dividido em 4 partes.

- 1° - Verifica se existe algum quadro livre na memória e, caso exista, referencia a pagina para esse quadro.
- 2° - Busca a página mais antiga com bit R=0.
- 3° - Adiciona a página ao final da lista e referencia ela ao quadro que a página vitima estava referenciada. Se ela ja estava na lista, é removida e inserida novamente.
- 4° - A página mais antiga(página vitima) é removida da lista.

3.7 Estruturas Auxiliares

Para a correta implementação dos algoritmos FIFO e Segunda Chance, fez-se necessário o uso de Tipos Abstratos de Dados (TADs), como filas e listas encadeadas. Essas estruturas são utilizadas pelos algoritmos para identificar uma página "vítima"de forma eficiente e adequada ao contexto.

4 Conclusões

Cada algoritmo possui características específicas que podem ser úteis em determinados cenários, mas menos apropriadas em outros. Por exemplo, o algoritmo FIFO, que remove as páginas na ordem em que chegam, é adequado quando as páginas de uma tabela são acessadas em tempos bem distantes. Por outro lado, em situações onde algumas páginas da tabela são acessadas com alta frequência, o uso do FIFO pode não ser vantajoso, pois páginas frequentemente acessadas seriam removidas.

Portanto, a escolha do algoritmo adequado depende de uma análise cuidadosa dos padrões de acesso às páginas. É essencial compreender as características de cada algoritmo para utilizá-las de forma otimizada, maximizando sua eficiência nos diferentes contextos de uso. A adequada seleção do algoritmo garantirá um melhor desempenho no gerenciamento da memória e atenderá às necessidades específicas de cada situação.

5 Referências

GEEKSFORGEEKS. *Not Recently Used (NRU) page replacement algorithm.* 2023. Disponível em: <https://www.geeksforgeeks.org/not-recently-used-nru-page-replacement-algorithm/>https://www.geeksforgeeks.org/not-recently-used-nru-page-replacement-algorithm/. Acesso em: 04.06.2023. Nenhuma citação no texto.

GEEKSFORGEEKS. *Page Replacement Algorithms in Operating Systems.* 2023. Disponível em: <https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/>https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/. Acesso em: 04.06.2023. Nenhuma citação no texto.

GEEKSFORGEEKS. *Second Chance (or Clock) Page Replacement Policy*. 2023. Disponível em: <https://www.geeksforgeeks.org/second-chance-or-clock-page-replacement-policy/>https://www.geeksforgeeks.org/second-chance-or-clock-page-replacement-policy/. Acesso em: 04.06.2023. Nenhuma citação no texto.