

Trabalho final

Bruno Cesar da Silva Claudino(brunoclaudino@gmail.com)

Bruno Rodrigues Caputo (brunorcx@hotmail.com)

DCC403-Sistemas Operacionais 2019.1-Turma 01

Universidade Federal de Roraima



DCC -Departamento de Ciência da Computação - Bloco V

Campus Universitário do Paricarana - Aeroporto

69310-000 Boa vista, RR



Simulador de Memória Virtual

resumo: *Este trabalho consiste na implementação de um simulador de memória virtual. Para isso, serão introduzidos conceitos básicos de sistemas operacionais, em seguida, será apresentado um programa na linguagem C como exemplificação.*

 <p>UFRR</p>	<p><Simulador de Memória Virtual> por <Bruno Cesar da Silva Claudino, Bruno Rodrigues Caputo> - 2019.1 Boa vista, 26/06/2019</p>	 <p>Departamento de CIÊNCIA DA COMPUTAÇÃO</p>
--	---	--

SUMÁRIO

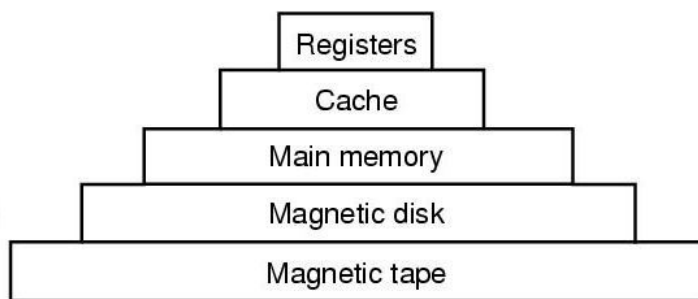
1 Introdução	2
1.1 Gerenciador de memória	2
1.2 Modos de tratar e endereçar a memória	2
1.3 Algoritmos de substituição de páginas	3
2 Funcionamento do simulador	3
3 Referências Bibliográficas	5

	<p align="center"><Simulador de Memória Virtual> por <Bruno Cesar da Silva Claudino, Bruno Rodrigues Caputo> - 2019.1 Boa vista, 26/06/2019</p>	
--	---	--

1 Introdução

Idealmente, podemos dizer, que a memória deveria ter as seguintes características: grande(em tamanho), rápida(em acesso), não volátil(dados persistentes) e de baixo custo. Contudo, não temos ainda uma memória como essa.

Respeitando a hierarquia de memória, podemos observar que duas características principais. A primeira é que quanto mais próximo do CPU a memória for, usualmente, terá um acesso mais rápido enquanto que em níveis mais distantes teremos maior tamanho da memória, na figura a seguir o CPU fica logo acima da memória de registradores.





1.1 Gerenciador de memória

Para cada tipo de memória conseguimos ter um gerenciador que ficará responsável por organizar os espaços livres e ocupados, fazendo alocação e localização de processos e/ou dados na memória em questão. Não se pode esquecer, dos problemas de *swapping*, para chaveamento entre memória principal com o disco, e memória principal e memória cache.

1.2 Modos de tratar e endereçar a memória

Existem diversos modos de se tratar a gerência de memória quando se tem múltiplos programas, uma das mais simples é fazer a divisão da memória em tamanho fixo, não necessariamente iguais, e ao encontrar um *job*, colocar na fila na sua respectiva partição.

 <p>UFRR</p>	<p align="center"><Simulador de Memória Virtual> por <Bruno Cesar da Silva Claudino, Bruno Rodrigues Caputo> - 2019.1 Boa vista, 26/06/2019</p>	 <p>Departamento de CIÊNCIA DA COMPUTAÇÃO</p>
--	--	---

1.3 Algoritmos de substituição de páginas

Existem diversos algoritmo para resolução deste problema, dentre eles vale ressaltar:

- LRU: Algoritmo Menos utilizado Recentemente, como o próprio nome diz ele se vale do conceito em que páginas muito usadas por último, provavelmente irão ser usadas novamente. Faz a troca da página que permaneceu sem ter muito uso pelo maior tempo. Custo elevado por precisar manter uma lista encadeada com todas as páginas mais recentes na memória. Utiliza um contador para identificar uso das páginas.
- Random: Enquanto houver páginas vazias na memória elas serão preenchidas, assim que a página estiver completamente ocupada uma nova página aleatória deverá ser substituída
- FIFO: utiliza a ideia geral da estrutura de dados lista, o sistema operacional mantém uma fila das páginas correntes na memória. A página no início da fila é a mais antiga e a página no final é a mais nova. Quando temos um *page fault*, a página do início será removida e a inserção se dará no início da fila. Pode ser ineficiente, já que uma página que está em uso constante pode ser retirada

2 Funcionamento do simulador

O simulador terá memórias fixas para cada programa, os quais deverão ser informadas na sua inicialização(quadros), bem como o tipo de algoritmo de substituição de páginas(conceitualmente chamado quando a página ficar cheia), tamanho da página e a memória total física.

Sua estrutura se dá através de lista encadeada, com o processo, endereço e um ponteiro para a próxima página.

Para compilar o programa basta utilizar o seguinte comando:



```
gcc -o MemSimulator MemSimulator.c
```

Para executar o programa deve-se utilizar o seguinte comando:

```
./MemSimulator lru arquivo.log 4 128
```

Seu código apresenta, inicialmente, comparações simples para verificar o tamanho máximo e mínimo das páginas e da memória, como pode ser visto a seguir:

```
if (pageSize < 2 || pageSize > 64){
    printf("ERRO: O tamanho de cada pagina deve estar entre 2 e 64.");
    return 0;
}
```

 <p>UFRR</p>	<p align="center"><Simulador de Memória Virtual> por <Bruno Cesar da Silva Claudino, Bruno Rodrigues Caputo> - 2019.1 Boa vista, 26/06/2019</p>	 <p>Departamento de CIÊNCIA DA COMPUTAÇÃO</p>
--	---	---



```
if (memSize < 128 || memSize > 16384){
    printf("ERRO: O tamanho da memoria deve estar entre 128 e 16384.");
    return 0;
}
```

Em seguida fará a leitura do arquivo, verificando o “r” ou “w” para confirmar leitura e escrita respectivamente. Até finalmente chegar na função de escrita de endereço, a qual tem como verificação as páginas usadas e determina a necessidade de criar uma nova página ou substituir.

```
void WriteAddress(char value[9]){
    if (usedPages < numPages){
        AddNewPage(tmpAddress);
    }
    else{
        faults++;
        ReplacePage(tmpAddress);
    }
}
```

Por fim, temos a proposta principal do código, em sua utilização de substituição de páginas que irá fazer de acordo com cada tipo especificado anteriormente.

```
void ReplacePage(char value[9]){
    if (strcmp(alg, "lru") == 0){
        LRU(value);
    }
    else if (strcmp(alg, "random") == 0){
        Random(value);
    }
    else if (strcmp(alg, "fifo") == 0){
        FIFO(value);}}}
```

 <p>UFRR</p>	<p><Simulador de Memória Virtual> por <Bruno Cesar da Silva Claudino, Bruno Rodrigues Caputo> - 2019.1 Boa vista, 26/06/2019</p>	 <p>Departamento de CIÊNCIA DA COMPUTAÇÃO</p>
--	---	--

3 Referências Bibliográficas

A. S. Tanenbaum, *Modern operating systems 3 e*. Prentice Hall, 2008.

UNIVESP, “Sistemas Operacionais - Aula 17 - Introdução ao Gerenciamento de Memória,” *YouTube*, 13-Jun-2017. [Online]. Available: <https://www.youtube.com/watch?v=Q8ZqjEafmNc&t=11s>.

“Virtual memory,” *Wikipedia*, 05-Jun-2019. [Online]. Available: https://en.wikipedia.org/wiki/Virtual_memory.