Segundo Trabalho[1] de INF1316

1- Introdução e Objetivo

Neste trabalho, você deverá implementar um simulador de memória virtual, **sim-virtual**, em linguagem C. Neste simulador você criará as estruturas de dados e os mecanismos de mapeamento de memória (lógico -> físico) necessários para realizar a paginação, e deverá implementar três *algoritmos de Substituição de Páginas:* o **Least-Recently-Used (LRU),** o algoritmo de **Segunda Chance,** o algoritmo do relógio **(Clock)** o algoritmo de **substituição Ótimo** (substitui a página que será acessada no futuro mais remoto).

O simulador receberá como entrada um arquivo que conterá a sequência de endereços de memória acessados por um programa real. Esses endereços estarão no formato hexadecimal, seguidos por uma letra R ou W, para indicar se o acesso foi de leitura ou escrita. Ao iniciar o simulador, use como tamanho da memória para aquele programa 1 MB ou 2 MB e indique qual algoritmo de substituição de páginas será utilizado. O simulador deve, então, processar cada acesso à memória para atualizar os bits de controle de cada página/quadro, detectar falha de páginas (*pagefaults*) e simular o processo de substituição de páginas e carga de uma nova página no quadro de página escolhido. Durante a simulação, estatísticas devem ser coletadas, para gerar um relatório curto ao final da execução.

O simulador deverá ter os seguintes quatro argumentos de linha de commando:

a) O algoritmo de substituição de página sendo simulado (LRU/2nd chance/clock/ótimo)
b) O arquivo contendo a sequência de endereços de memória acessados (arq.log)
c) O tamanho de cada página/quadro de página em KB (tamanhos a serem suportados 8 e 32 KB)
d) O tamanho total de memória física hipoteticamente disponível que pode ser de 1MB e de 2 MB.

Ou seja, uma invocação do simulador:

```
sim-virtual  LRU arquivo.log 8 2
```

indicaria o algoritmo de substituição de páginas LRU, simulado para a entrada dada pelo arquivo.log, um tamanho de página de 8KB e uma memória fisica de 2 MB.

2- Formato do arquivo de entrada

Cada linha do arquivo (.log) conterá um endereço de memória acessado (em representação hexa-decimal), seguido das letras R ou W, indicando um acesso de leitura ou escrita, respectivamente.

---

[1] O enunciado foi adaptado de um trabalho prático do prof. Dorgival Guedes Neto (DCC/UFMG)

Por exemplo:

```
0044e4f8 R
0044e500 R
0044e520 R
0700ff10 R
2f6965a0 W
0044e5c0 R
00062368 R
```

Portanto, a leitura de cada linha do arquivo poderá ser feita usando o procedimento `da stdio: fscanf(file, "%x %c ", &addr, &rw),` onde o tipo de `addr` é `unsigned`.

Serão fornecidos quatro arquivos de entrada, que correspondem a sequências de acesso à memoria de uma execução real dos seguintes 4 programas considerados significativos: **compilador.log, matrix.log, compressor.log** e **simulador.log**, ou seja, um compilador, um programa científico, um compressor de arquivos e um simulador de partículas. Estes arquivos estarão disponíveis na página do curso (no EAD).

Obtendo a página a partir do endereço lógico

Para de obter o índice da página que corresponde a um endereçp lógico, `addr`, basta descartar os ***s*** bits menos significativos, ou seja, executar um shift para a direita: **`page= addr >> s`**. Por exemplo, se o tamanho da página/quadro for 8 KB, obtém-se o índice da página com a instrução: **`page= addr>>13 (8K requer 13 bits para representação)`**. No entanto, o seu simulador deve funcionar para diferentes tamanhos de página (vide parâmetro/item **c** de **sim-virtual,** na seção 1), ou seja, você precisará implementar um código que calcula ***s*** a partir do valor do parâmetro do item **c**).

Estruturas de dados

Como os endereços nos arquivos.log, são de 32 bits então, para páginas de 8 KB (as menores a serem suportadas pelo seu simulador), como 13 bits endereçam 8KB, pode haver 19 bits relativos ao índice de página, permitindo tabelas de página de meio MB entradas! Isso não seria muito eficiente na prática, mas para fins de simulação, e como está-se considerando apenas um programa em execução, isso não é um problema para a simulação.
A estrutura de dados para representar cada quadro físico deve conter os flags **R** (de página referenciada), flag **M** (de página modificada), e o **instante de último acesso de cada página em memória**. Os demais campos dependerão de cada um dos algoritmos implementados, e ficam a seu critério.

Implementação da Substituição de páginas

Os algoritmos de substituição de páginas só entram em ação quando todos os quadros de página estiverem ocupados, e se houver um Page-fault, ou seja, se uma nova página precisar ser carregada na memória física. Nesse caso, a página contida no quadro correspondente será: (a) sobre-escrita, se só tiver sido acessada para leitura, ou (b) se tiver sido modificada (página suja), precisará ser escrita de volta no disco (apenas em teoria, lembre-se que este trabalho é uma simulação) na partição de paginação. Tanto o total de Page-faults como o total de escrita de páginas sujas **devem ser registradas** na simulação.

Utilize um contador **`time`** (valor inicial = 0) que é incrementado a cada iteração do seu simulador, como forma de simular a passagem do tempo e registrar o momento de cada acesso à memória.

Formato de Saída da simulação

Como resultado, quando o último acesso à memória for processado, o simulador deverá gerar um pequeno relatório, contendo:

- A configuração utilizada (os quatro parâmetros de **sim-virtual**);
- O número de page faults (páginas carregadas na memória);
- O número de páginas ``sujas'' que seriam escritas de volta no disco (**lembre-seque páginas sujas que existam no final da execução não precisam ser escritas**).

Um exemplo de saída poderia ser da forma (valores completamente fictícios):

```
prompt> sim-virtual lru arquivo.log 8 2
Executando o simulador...
Arquivo de entrada: arquivo.log
Tamanho da memoria fisica: 2 MB
Tamanho das páginas: 8 KB
Algoritmo de substituição: LRU
Número de Faltas de Páginas: 520
Número de Páginas Escritas: 352
```

Entregáveis

Você deve entregar o(s) código(s) fonte do simulador (.c e .h), e um relatório sobre o seu trabalho. Não inclua os arquivos.log no kit de entrega!

O relatório deve conter:

- **Um resumo do projeto:** alguns parágrafos que descrevam a estrutura geral do seu simulador e as estruturas de dados relevantes (para implementação de cada algoritmo de substituição).
- **Uma análise do desempenho dos algoritmos de substituição de páginas para os 4 programas utilizados**, comparando o número de page-faults e de escritas de página quando o tamanho da memória física permanece fixo, mas o tamanho das páginas varia. Compare os resultados para os dois tamanhos de página. Compare os algoritmos de substituição de páginas e justifique os valores obtidos.
- **Uma descrição do algoritmo de substituição ideal (algoritmo ótimo)**, em pseudo-código.

Esse relatório com a análise de desempenho e definição do algoritmo ótimo é uma parte importante do trabalho e terá um peso significativo sobre a nota do trabalho.

Observações finais

- Os trabalhos podem ser feitos em grupos de 2 alunos. Indique claramente os integrantes do grupo no cabeçalho do simulador e no relatório.
- Os grupos poderão ser chamados para apresentações orais/demonstrações dos trabalhos.