Segundo Trabalho[1] de INF 1019 – 2018.2

Data de entrega: 2ª.feira, 10 de dezembro de 2018

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 dois *algoritmos de Substituição de Páginas:* o Not-Recently-Used (NRU) e o Least-Recently-Used (LRU).

O simulador receberá como entrada um arquivo que conterá a sequência de endereços de memória acessados por um programa real (de fato, apenas uma parte da sequência completa de acessos a memória). 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, será definido o tamanho da memória (em quadros) para aquele programa e qual o algoritmo de substituição de páginas que 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 (*page faults*) 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/NRU)
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 a 32 KB)
d) O tamanho total de memória física hipoteticamente disponível pode variar de 1 MB a 16 MB.

Ou seja, uma invocação do simulador:

```
sim-virtual  LRU arquivo.log 16 128
```

indicaria um tamanho de página de 16KB e uma memória fisica de 128 KB.

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 (www.inf.puc-rio.br/~seibel/...)

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= add >> s`**. Por exemplo, se o tamamnho da página/quadro for 8 KB, obtém-se o índice da página com a instrução: **`page= addr>>13`**. No entanto, o seu simulador deve funcionar para diferentes tamanhos de página (vide parâmetro c de **sim-virtual, n**a seção 2), ou seja, você precisará implementar um código que calcula *s* a partir do valor do parâmetro 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), 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 houver um Page-fault, ou seja, uma nova página precisar ser carregada na memória física. Nesse caso, a página contida no quadro correspondente será: (a) sobre-escrita, se só tiver acessada para leitura, ou (b) se tiver sido modificada, precisará ser escrita de volta o disco – na partição de paginação. Tanto os Page-faults como essa escrita de páginas sujas deve ser registrado 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 a 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 tiveram que ser escritas de volta no disco (lembre-se que 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 8 128
Executando o simulador...
Arquivo de entrada: arquivo.log
Tamanho da memoria fisica: 128 KB
Tamanho das páginas: 8 KB
Alg de substituição: lru
Numero de Faltas de Páginas: 520
Numero de Paginas escritas: 352
```

Simulador em modo depuração

O programa (simulador) a ser entregue precisar gerar apenas o relatório final descrito anteriormente. Porém, recomenda-se fortemente que o simulador funcione também em modo ``depuração'' onde o mesmo explicite o que é feito a cada acesso à memória, por exemplo se foi "page hit", "page fault", e se a substituição teve que envolver uma cópia da página "suja" para disco. Para tal, deve-se ter um quinto parâmetro de **sim-virtual**, chamado "debug" ou "-D", que indique o uso do modo depurador. Recomendo também que vocês implementem um modo passo-a-passo (modo interativo), para que o usuário possa acompanhar o funcionamento da simulação.

O algoritmo OPTimo

No relatório de entrega do trabalho você deve descrever o que seria um algoritmo ótimo (OPT) para a substituição de páginas, que pode usar qualquer informação ou estatística sobre os índices de página sendo requisitados, e inclusive pode "olhar para o futuro", ou seja, descobrir as páginas que serão acessadas no futuro mais distante (em termos do número de instruções), e quais delas terão um acesso de leitura ou escrita.
Com essa liberdade de "vidente", tente mostrar que OPT terá um desempenho melhor do que NRU e o LRU para todos os logs.

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 breve descrição do seu algoritmo OPT
- Uma análise do desempenho dos 2 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 dois tamanhos de página.

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

Observações finais

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