UNIVERSIDADE FEDERAL DE SANTA CATARINA
INE5412 – Sistemas Operacionais I
Profs. Giovani Gracioli e Márcio Bastos Castro
2023/02

Simulação de Algoritmos de Substituição de Páginas

O presente trabalho tem por objetivo escrever um programa para simular o comportamento de algoritmos de subsituição de páginas, usados pelos mecanismos de paginação dos sistemas operacionais modernos. O programa deve simular os seguintes algoritmos de substituição de páginas:

- FIFO (First In, First Out)
- OPT (Algoritmo ótimo)
- LRU (Least Recently Used)

O programa deve receber como parâmetro o número de quadros (frames) disponíveis na memória RAM (usando os parâmetros argc/argv na main) e ler da entrada padrão (stdin – usando cin por exemplo) a sequência de referência às páginas (uma referência por linha), conforme exemplo abaixo:

```
./simulador 4 < referencias.txt

ou

cat referencias.txt | ./simulador 4
```

No exemplo acima, a memória RAM tem 4 quadros e o arquivo referencias.txt contém as referências de acesso para as páginas, sendo uma por linha. A entrada deve ser finalizada quando for encontrado um caracter vazio indicando final de arquivo (usar !feof(stdin) para parar o laço de entrada).

A saída da simulação deve ter o número de quadros, o número de acesso a memória e o número de falta de páginas verificado para cada algoritmo, de acordo com o exemplo a seguir:

```
4 quadros
30 refs
FIFO: 17 PFs
LRU: 15 PFs
OPT: 11 PFs
```

Atenção: use exatamente esse formato de saída.

São disponibilizados 3 arquivos de teste contendo referências de acesso às páginas. Um dos

arquivos, “vsim-gcc.txt.gz”, possui um trecho de *tracing* de acessos à memória do compilador GCC (obtido e adaptado desta página).

Formato de Entrega e Avaliação

O trabalho em duplas deverá ser entregue no Moodle no dia especificado pela tarefa. Todos os arquivos contendo o código do trabalho, bem como Makefile e um **relatório** apresentando **sucintamente** a solução e seu projeto (i.e., projeto OO, com diagramas UML), deverão ser submetidos pelo Moodle.

A avaliação se dará em 4 fases:

1. Avaliação de compilação: compilar o código enviado. Caso haja erros de compilação, a nota do trabalho será automaticamente zerada.

2. Avaliação de execução: para validar que a solução executa corretamente sem falhas de segmentação. Caso haja falhas de segmentação, a nota é zerada. Será também avaliado o uso de variáveis globais (-5 pontos) e vazamentos de memória (-20%).

3. Desempenho do programa de simulação também será avaliado e comparado. O trabalho que realizar a simulação mais rapidamente terá a nota no quesito em 100%. Os demais serão ranqueados até o mínimo de 60%. A avaliação de desempenho será, obviamente, realizada no mesmo computador.

4. Avaliação da organização do código: busca-se nesta fase avaliar a organização do código orientado a objetos e o seguimento das diretrizes do trabalho (saída, contexto, processo, CPU, troca de contexto, algoritmos de escalonamento). Deve-se usar classes e objetos e não estilo de programação baseado em procedimentos (como na linguagem C). Alguns itens para avaliação são: (i) funcionamento do programa; (ii) saída do programa (conforme especificação); (iii) clareza do código (utilização de comentários e nomes de variáveis adequadas); (iv) qualidade do relatório; (v) compilação sem warnings; (vi) sem vazamento de memória e (vii) modelagem do software desenvolvido com diagramas UML.

5. A quarta fase consiste na apresentação do trabalho em dia e horário agendado pelo professor. Durante as apresentações, o professor irá avaliar o **conhecimento individual dos alunos sobre os conteúdos teóricos e práticos vistos em aula e sobre a solução adotada no trabalho**. A nota atribuída à cada aluno i no trabalho ($NotaTrabalho_i$) será calculada da seguinte forma, onde A_i é a nota referente à apresentação do aluno i e S é a nota atribuída à solução do trabalho:

$$NotaTrabalho_i = \frac{A_i \, x \, S}{10}$$

Plágio não será tolerado em nenhuma hipótese ao longo dos trabalhos, acarretando em nota 0 a todos os envolvidos.

Caso a dupla apresente a saída da simulação utilizando uma interface gráfica, terá um bônus de até 2 pontos na nota do trabalho.

Referências

- Disciplina de Sistemas Operacionais do curso de Ciência da Computação da UFPR. Professor Carlos Maziero.
- Sistemas Operacionais Modernos. 3ª edição. Andrew S. Tanembaum. Pearson. 2010.