Lista de Exercícios - Sistemas Operacionais
Aula 10: *Gerenciamento de Memória - Parte 3*

Professores: Felipe M. G. França e Valmir C. Barbosa
Assistente: Alexandre H. L. Porto

1. Quanto tempo leva para carregar um programa de 64KB de um disco cujo tempo médio de busca é 10ms, cujo tempo de rotação é 8ms e cujas trilhas armazenam 1MB,

   (a) para um tamanho de página de 2KB?

   (b) para um tamanho de página de 4KB?

   (c) para um tamanho de página de 64KB?

   As páginas são distribuídas aleatoriamente no disco. Baseado nos resultados obtidos, por que as páginas usadas na prática são tão pequenas? Cite duas desvantagens de usarmos uma página de 64KB ao invés de uma de 4KB.

   Resp.: Primeiramente vamos notar que, em média, o tempo gasto para ler uma página aleatória do disco é de 14ms, pois deveremos esperar em média pela metade do tempo de rotação do disco para ler esta página (4ms), e o tempo médio de busca é de 10ms. Agora, como cada trilha armazena 1MB, isto é, 1024KB, e como o tempo de rotação do disco é de 8ms, então a taxa de transferência de dados será de 128KB/ms (isto é, transferimos 128KB em 1ms). Então:

   (a) Para uma página com tamanho de 2KB, o tempo de carga desta página será de 14ms mais o tempo necessário para copiá-la, que será de $2/128 = 0.015625$ms. Agora, como o tamanho do programa é de 64KB, este terá 32 páginas, e com isso, o tempo para carregá-lo será de 14.015625×32 ms, isto é, 448.5ms.

(b) Para uma página com tamanho de 4KB, o seu tempo de carga será de 14ms mais o seu tempo de cópia, que será de $4/128 = 0.03125$ms. Agora, como temos 16 páginas no programa, pois o seu tamanho é de 64KB, o tempo para carregar o programa será de 14.03125×16 ms, isto é, 224.5ms.

(c) Para uma página com tamanho de 64KB, o tempo de carga da página será de 14ms mais o tempo gasto para copiá-la, que será de $64/128 = 0.5$ms. Agora, como o programa terá somente uma página, pois o seu tamanho é igual ao tamanho da página, o tempo para carregar o programa será de 14.5ms.

Como podemos notar pelos resultados anteriores, quanto maior a página, menor é o tempo de carga desta página. O motivo de se usar páginas pequenas é devido a dois fatores:

(a) O desperdício com a fragmentação interna, tanto na memória como no disco ao salvar uma página (como vimos na Aula 10, em média, a metade da última página alocada a uma área de um processo não é usada). No caso da página de 64KB (que desperdiça em média 32KB) o desperdício será 16 vezes maior do que o da página de 4KB (que desperdiça em média 2KB);

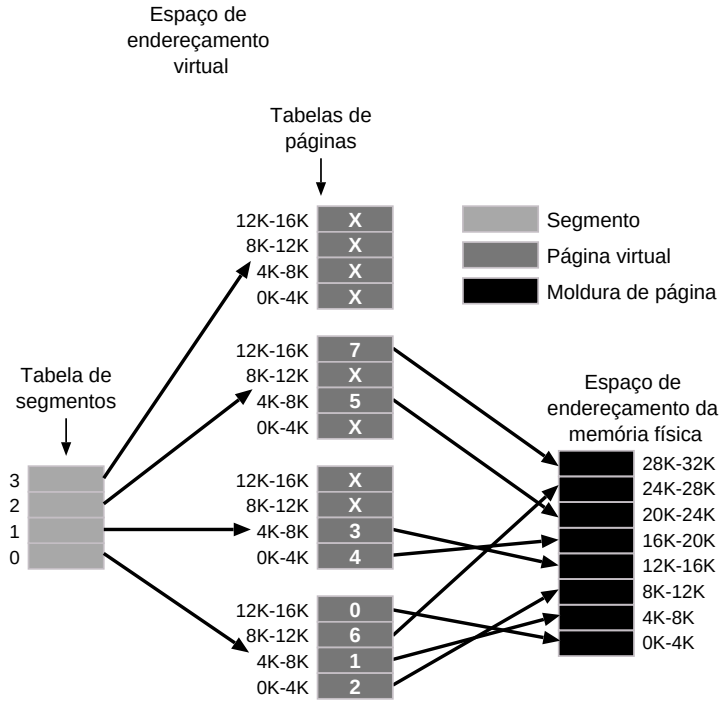
(b) Quanto maior o tamanho das páginas, menor a sua quantidade, o que pode reduzir o desempenho do sistema, caso o conjunto funcional de um processo esteja disperso por todo o seu espaço de endereçamento. Suponha que temos um espaço de endereçamento virtual de 16MB e um espaço de endereçamento físico de 1MB. Suponha ainda que o processo use ativamente 256 regiões de memória de 4KB distribuídas uniformemente por todo o espaço de endereçamento virtual. Logo, a distância entre as posições iniciais de duas regiões consecutivas no espaço de endereçamento virtual será de 64KB. Além disso, teremos 256 molduras de página se usarmos páginas virtuais com 4KB e somente 16 molduras se usarmos páginas com 64KB. Com isso, a taxa de falhas de página será cerca de 16 vezes maior se usarmos páginas com 64KB, pois poderemos colocar, para ambos os tamanhos de página, somente uma região em cada moldura de página.

2. Suponha que a memória do computador possua 8 molduras de página, e que as molduras 0, 1 e 5 estejam sendo usadas por um processo A, e as molduras restantes por um outro processo B. Suponha ainda que as páginas tenham sido carregadas nestas molduras nos tempos 23, 4,

12, 8, 15, 17, 34 e 2, respectivamente, e que o sistema operacional use o algoritmo FIFO. Qual seria a moldura cuja página seria substituída se o sistema usasse a política de alocação de páginas global? E se fosse usada a política de alocação de páginas local?

Resp.: -Quando usamos a política de alocação global, todas as páginas serão consideradas ao ocorrer uma falha de página. Com isso, o algoritmo FIFO, que ordena as páginas em ordem crescente de acordo com o tempo da carga, ordenará todas as páginas na seguinte ordem: 7, 1, 3, 2, 4, 5, 0 e 6. Logo, se ocorrer uma falha de página, a página 7 do processo B será a escolhida para ser substituída.
-Quando usamos a política de alocação local, somente as páginas do processo que gerou a falha de página serão consideradas. Com isso, se o processo A gerou a falha de página, o algoritmo FIFO primeiramente ordenará as páginas 0, 1 e 5 de A em ordem crescente de acordo com o tempo da carga, na seguinte ordem: 1, 5 e 0. Depois disso, escolherá a página de A com menor tempo da carga para ser substituída, isto é, a página 1. Agora, se a falha foi gerada pelo processo B, primeiramente as páginas 2, 3, 4, 6 e 7 de B serão ordenadas pelo algoritmo FIFO, na seguinte ordem: 7, 3, 2, 4 e 6. Finalmente, a página com o menor tempo da carga, a 7, será a escolhida para ser substituída.

3. Suponha que o sistema operacional use a técnica de segmentação com paginação, e que o computador tenha um espaço de endereçamento virtual dividido como na figura dada a seguir. Responda:



(a) Para cada segmento, que faixas de endereços gerariam falhas de página ao acessá-las, se o "X" em uma entrada de uma tabela de páginas indicasse que a página não está na memória?

Resp.: Para o segmento 0, vemos que não ocorrerão falhas de páginas ao acessarmos um endereço deste segmento, pois todas as suas páginas estão mapeadas na memória. Para o segmento 1, vemos que uma falha de página será gerada se acessarmos os endereços de 8192 até 16383 (faixas 8K-12K e 12K-16K). Para o segmento 2, vemos que uma falha de página será gerada ao acessarmos os endereços de 0 até 4095 (faixa 0K-4K) ou os endereços de 8192 até 12287 (faixa 8K-12K). Finalmente, o acesso a qualquer endereço do segmento 3 irá gerar uma falha de página, pois nenhuma das páginas deste segmento está mapeada na memória.

(b) Suponha que o processo A esteja usando os segmentos 1 e 3, e que o processo B esteja usando os segmentos 0 e 2. Suponha ainda que a ordem de carga das páginas nas molduras tenha sido a seguinte: 1, 7, 5, 2, 0, 6, 3 e 4. Qual será a página substituída quando o processo A acessar algum endereço do segmento 3, se o sistema operacional usar o algoritmo FIFO para substituir as páginas com

uma política de alocação global? E se a política for local?

Resp.: Como vimos na Aula 9, no algoritmo FIFO, a página carregada há mais tempo na memória será a escolhida para ser substituída. E como podemos ver pela figura, as molduras 3 e 4 estão associadas ao processo A, e as outras molduras estão associadas ao processo B. Então:

- Se a política de alocação global for usada, todas as molduras deverão ser consideradas pelo algoritmo FIFO. Agora, como a ordem 1, 7, 5, 2, 0, 6, 3 e 4 está de acordo com a ordem crescente do tempo de carga das páginas nas molduras, então a página 1 do segmento 0 (associado ao processo B), que está na moldura 1, será a escolhida para ser substituída.

- Se a política de alocação for local, então somente as molduras associadas ao processo A, ou seja, as molduras 3 e 4, deverão ser consideradas. Como neste caso a ordenação, de acordo com o tempo de carga, será 3 e 4, então a página 1 do segmento 1, que está na moldura 3, será a escolhida para ser substituída.

(c) Suponha que uma TLB tenha sido adicionada à MMU. Qual deveria ser o número de entradas desta TLB para que 5% dos acessos a endereços de um segmento fossem satisfeitos por ela?

Resp.: O enunciado desta questão ficou ambíguo, pois não foi dito se o endereço é o endereço virtual bidimensional, composto pelo par segmento, endereço linear dentro deste segmento, ou o endereço linear de um segmento. Logo, existem duas respostas corretas: a que considerar o endereço como o bidimensional ou a que considerar o endereço como o linear. A seguir damos as respostas para cada uma destas possibilidades:

- Para garantir que 5% dos acessos aos endereços lineares de cada um dos segmentos sejam satisfeitos pela TLB, então deveremos dividir o espaço da TLB em 4 partes iguais, e associar cada parte a um dos segmentos. Como cada segmento tem 16384 (16K) endereços lineares, então bastará que a TLB possua 820 entradas para cada segmento. Isso ocorrerá porque uma entrada da TLB dá o mapeamento de um dos endereços

lineares do segmento, e 5% de 16384 é aproximadamente igual a 820 (pois $16384 \times 0,05 = 819,2 \sim 820$). Logo, a TLB deverá ter 3280 entradas no total.

- Para garantir que 5% dos acessos aos endereços virtuais bidimensionais sejam satisfeitos pela TLB, então deveremos ter entradas suficientes para armazenar 5% destes endereços. Existem 65536 endereços virtuais, pois temos 4 segmentos com 16384 (16K) endereços lineares. Logo, a TLB deverá possuir 3277 entradas, pois uma entrada da TLB dá o mapeamento de um dos endereços virtuais, e 5% de 65536 é aproximadamente igual a 3277 (pois $65536 \times 0,05 = 3276,8 \sim 3277$).

4. Suponha que um sistema operacional use três segmentos com tamanho de 64KB. O primeiro segmento armazenará a pilha dos processos, o segundo armazenará o código dos processsos e, finalmente, o terceiro armazenará os dados dos processos. Suponha ainda que o sistema operacional use a segmentação com paginação, com páginas cujos tamanhos podem ser um dos seguintes: 512 bytes, 2KB, 8KB ou 16KB. Para quais tamanhos de página as pilhas, os códigos e os dados dos processos cujos tamanhos, em bytes, são dados na tabela a seguir podem ser totalmente armazenados na memória ao mesmo tempo? Justifique a sua resposta.

Processo	Pilha	Código	Dados
A	16384	8192	1024
B	8192	4096	2048
C	3072	6144	17408

Resp.: A seguir apresentamos uma tabela para cada um dos possíveis tamanhos de página. Em cada tabela mostramos, no seu título, o tamanho da página e o número total de páginas em cada segmento. Até a penúltima linha de cada tabela mostramos, usando o mesmo formato da tabela do enunciado, os números de páginas necessários a cada um destes tamanhos. Finalmente, na última linha de cada tabela mostramos, para cada uma das colunas, o número total de páginas necessárias. Para que o tamanho da página não possa ser usado, o número de páginas necessárias, em alguma coluna (segmento) da tabela associada ao tamanho, precisa ser maior do que o número de páginas definidas por este tamanho. Como os números de páginas necessários, em cada coluna de cada tabela, são todos menores ou iguais aos máximos correspondentes, então todos os tamanhos poderão ser usados para armazenar totalmente todos os três processsos na memória.

Tamanho de 512 bytes - 128 páginas

Processo	Pilha	Código	Dados
A	32	16	2
B	16	8	4
C	6	12	34
Total	54	36	40

Tamanho de 2KB - 32 páginas

Processo	Pilha	Código	Dados
A	8	4	1
B	4	2	1
C	2	3	9
Total	14	9	11

Tamanho de 8KB - 8 páginas

Processo	Pilha	Código	Dados
A	2	1	1
B	1	1	1
C	1	1	3
Total	4	3	5

Tamanho de 16KB - 4 páginas

Processo	Pilha	Código	Dados
A	1	1	1
B	1	1	1
C	1	1	2
Total	3	3	4