Curso de Tecnologia em Sistemas de Computação
Disciplina de Sistemas Operacionais
Professores: Valmir C. Barbosa e Felipe M. G. França
Assistente: Alexandre H. L. Porto

Quarto Período
AD2 - Primeiro Semestre de 2011

Atenção: ADs enviadas pelo correio devem ser postadas cinco dias antes da data final de entrega estabelecida no calendário de entrega de ADs.

Atenção: Tem havido muita discussão sobre a importância de que cada aluno redija suas próprias respostas às questões da AD2. Os professores da disciplina, após refletirem sobre o assunto, decidiram o seguinte: Cada aluno é responsável por redigir suas próprias respostas. Provas iguais umas às outras terão suas notas diminuídas. As diminuições nas notas ocorrerão em proporção à similaridade entre as respostas. Exemplo: Três alunos que respondam identicamente a uma mesma questão terão, cada um, 1/3 dos pontos daquela questão.

Observação: A questão 6 é opcional. Se você tentar resolvê-la, você poderá ganhar até 2,0 pontos adicionais na nota da prova, mas a sua nota ainda será limitada a 10,0 pontos.

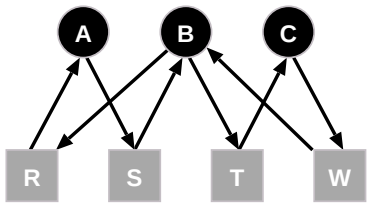
Nome -
Assinatura -

---

1. (2,0) Suponha que três processos A, B e C estejam em execução no sistema operacional, e que existam quatro recursos R, S, T e W não-preemptivos inicialmente não alocados a nenhum dos processos. Suponha ainda que um processo somente obtenha um conjunto de recursos se todos estiverem livres, e que ele seja bloqueado se um desses recursos não estiver disponível. Se somente o processo que possuir um recurso puder liberá-lo mostre, para a sequência de pedidos dada a seguir, o grafo de recursos obtido e, caso existam impasses informe, para cada um deles, os processos e recursos envolvidos.

   - A requisita R;
   - B requisita S e W;
   - C requisita T;
   - B requisita R e T;
   - A requisita S;
   - C requisita W.

   Resp.: A seguir mostramos o grafo de recursos obtido depois de o sistema operacional processar a sequência de pedidos dada na questão. Como podemos ver por esse grafo, existem dois impasses (ou ciclos), um envolvendo os processos A e B e os recursos R e S (o ciclo A-S-B-R-A) e um outro envolvendo os processos B e C e os recursos T e W (o ciclo B-T-C-W-B).



2. (2,0) Suponha que um computador possua 4GB de memória alocados em blocos de 32KB. Quantos KB são necessários para construir um mapa de bits para a memória livre? Suponha que o mapa não é armazenado nessa mesma memória.

Resp.: Como a memória do computador tem 4GB ou 4194304KB, como cada bloco (ou unidade de alocação) tem tamanho de 32KB, e finalmente como o mapa de bits não é armazenado na memória, então existem 131072 blocos na memória. Logo, o mapa de bits gasta 131072 bits ou 16384 bytes ou 16KB de espaço, pois no mapa temos um bit para cada bloco de memória indicando se ele está usado ou livre.

3. (2,0) Suponha que um computador tenha uma memória virtual de 256KB com páginas virtuais de 8KB, e que as informações dessas páginas virtuais sejam armazenadas em tabelas de páginas com um único nível. Suponha ainda que a memória física do computador possa armazenar somente um quarto das páginas virtuais. Responda às seguintes perguntas, justificando a sua resposta.

   (a) (1,0) Como ficaria a divisão em bits do endereço virtual? E como ficaria a divisão em bits do endereço físico?

   Resp.: -Como a memória virtual tem 256KB e como as páginas virtuais têm 8KB de tamanho, então existe um total de 32 páginas virtuais. Logo, o endereço virtual, que tem 18 bits porque 256KB é igual a 262144 ou 2^{18} bytes, é dividido do seguinte modo: os 5 bits superiores do endereço indicam o número da página virtual, pois $32 = 2^5$; e os 13 bits inferiores restantes indicam o deslocamento dentro da página virtual (porque 8KB é igual a 8192 ou 2^{13} bytes).
   -O tamanho da memória física é de 64KB, pois ela pode somente armazenar um quarto dos 256KB da memória virtual. Agora, como o tamanho da moldura de página é igual a 8KB, o mesmo tamanho da página virtual, então existe um total de $64/8 = 8$ molduras de página. Logo, o endereço físico, que tem 16 bits porque 64KB é igual a 65536 ou 2^{16} bytes, é dividido do seguinte modo: os 3 bits superiores definem o número da moldura de página, pois $8 = 2^3$; e os 13 bits inferiores restantes definem o deslocamento dentro da moldura (porque 8KB é igual a 2^{13} bytes).

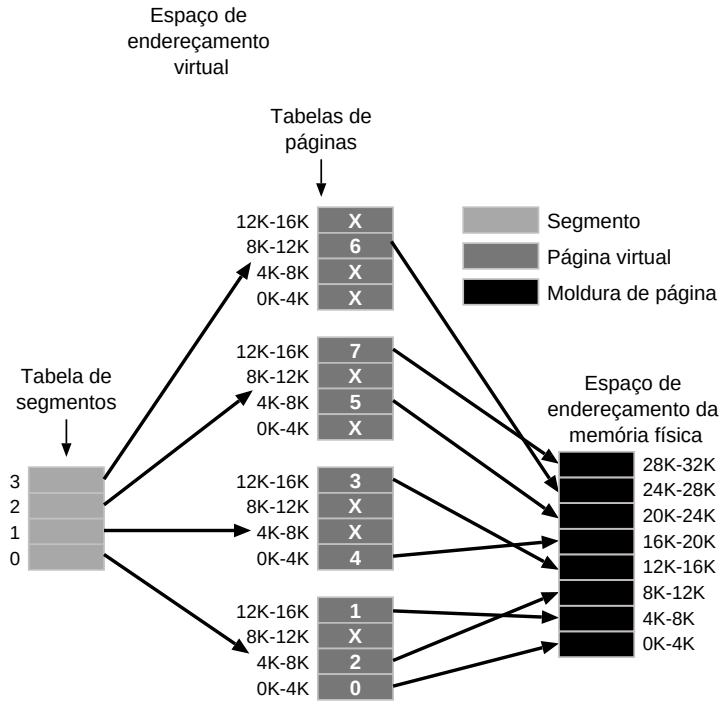
   (b) (1,0) Suponha que tenha sido alocada, a um processo, a metade das molduras de página disponíveis, e que essas molduras inicialmente estejam vazias. Quantas falhas de página ocorrerão se o

sistema operacional usar o algoritmo LRU, e se o processo acessar, em sequência, as páginas virtuais 15, 27, 7, 0, 9, 0, 15, 7, 27 e 9?

Resp.: Como vimos na Aula 9, no algoritmo LRU as páginas são primeiramente ordenadas, em ordem crescente, de acordo com o tempo do seu último acesso. A página a ser substituída é a primeira página segundo essa ordenação, isto é, a página não acessada há mais tempo. Na tabela dada a seguir mostramos, em cada linha, o que ocorre ao acessarmos as páginas na ordem dada no enunciado. Para cada uma dessas linhas mostramos na primeira coluna a página que é acessada, na segunda coluna a ordem em que as páginas devem ser escolhidas e finalmente, na terceira coluna, se o acesso gera uma falha de página. A primeira página segundo a ordenação, que é aquela a ser substituída, é mostrada em negrito. Como podemos ver por esta tabela, teremos 8 falhas de página.

Páginas	Ordenação				Ocorreu uma falha?
15	**15**				Sim
27	**15**	27			Sim
7	**15**	27	7		Sim
0	**15**	27	7	0	Sim
9	**27**	7	0	9	Sim
0	**27**	7	9	0	Não
15	**7**	9	0	15	Sim
7	**9**	0	15	7	Não
27	**0**	15	7	27	Sim
9	**15**	7	27	9	Sim

4. (2,0) Suponha que um computador utilize a técnica de segmentação com paginação. Na figura a seguir mostramos as tabelas de páginas para os 4 segmentos de 16KB do computador. Usando essas tabelas, forneça os endereços virtuais bidimensionais (segmento, endereço dentro do segmento) correspondentes aos endereços físicos dados e os endereços físicos correspondentes aos endereços virtuais bidimensionais dados:



(a) (0,5) (0, 16383)

(b) (0,5) 25752

(c) (0,5) (2, 5116)

(d) (0,5) 12345

Resp.: Pela figura vemos que o espaço de endereçamento de cada segmento possui 16K, com endereços de 14 bits variando de 0 até 16383. Já o espaço de endereçamento físico possui 32K, com endereços variando de 0 até 32767. Como as páginas possuem 4K de tamanho, cada segmento é dividido em 4 páginas virtuais e o espaço de endereçamento físico é dividido em 8 molduras de página. Logo, um endereço de um segmento é dividido do seguinte modo: 2 bits para o número da página virtual e 12 bits para o deslocamento. De modo similar, o endereço físico é dividido do seguinte modo: 3 bits para o número da moldura de página e 12 bits para o deslocamento. A seguir mostramos como converter cada um dos endereços físicos para os endereços virtuais bidimensionais correspondentes e vice-versa. Mostramos também, entre parênteses, cada endereço na base binária, separando os dois campos

do endereço pelo caractere "|".

(a) Endereço virtual (0, 16383) (11 | 111111111111): para esse endereço do segmento 0, vemos que estamos acessando a palavra 4095 da página virtual 3 (com endereços de 12288 até 16383). Pela figura, vemos que essa página virtual 3 está mapeada na moldura de página 1 (com endereços de 4096 até 8191). Logo, o endereço físico é 4096 (o primeiro endereço da moldura) mais 4095 (o deslocamento), isto é, 8191.

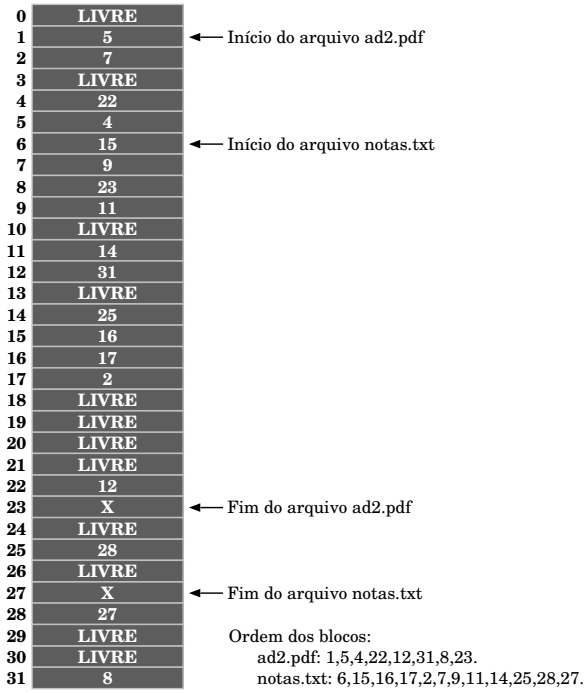
(b) Endereço físico 25752 (110 | 010010011000): para esse endereço, vemos que estamos acessando a palavra 1176 da moldura de página 6 (com endereços de 24576 até 28671). Agora, como vemos na figura que a página virtual 2 do segmento 3 (com endereços de 8192 até 12287) está mapeada nessa moldura, então a segunda componente do endereço virtual bidimensional é 8192 (a primeira palavra da página) mais 1176 (o deslocamento), isto é, 9368. Logo, o endereço virtual bidimensional é $(3, 9368)$.

(c) Endereço virtual (2, 5116) (01 | 001111111100): em relação a esse endereço do segmento 2, vemos que foi acessada a palavra 1020 da página virtual 1 (com endereços de 4096 até 8191). A figura nos diz que essa página está mapeada na moldura de página 5 (com endereços de 20480 até 24575). Logo, o endereço físico é 20480 (o endereço inicial da moldura) mais 1020 (o deslocamento), isto é, 21500.

(d) Endereço físico 12345 (011 | 000000111001): no caso desse endereço, estamos acessando a palavra 57 da moldura de página 3 (com endereços de 12288 até 16383). Pela figura, vemos que a página virtual 3 do segmento 1 está mapeada nessa moldura (com endereços também de 12288 até 16383), então a segunda componente do endereço virtual bidimensional também é 12345 (12288 + 57). Portanto, o endereço virtual bidimensional é $(1, 12345)$.

5. (2,0) Suponha que um computador tenha um disco com 32 blocos numerados de 0 até 31, e que o sistema operacional use a técnica de alocação por lista encadeada baseada em um índice ao gerenciar os blocos do disco. Responda às seguintes perguntas justificando a sua

resposta, supondo que a alocação dos blocos do disco seja dada pela tabela a seguir:



(a) (0,5) Qual é o número mínimo de bytes necessários para armazenar toda a tabela no disco?

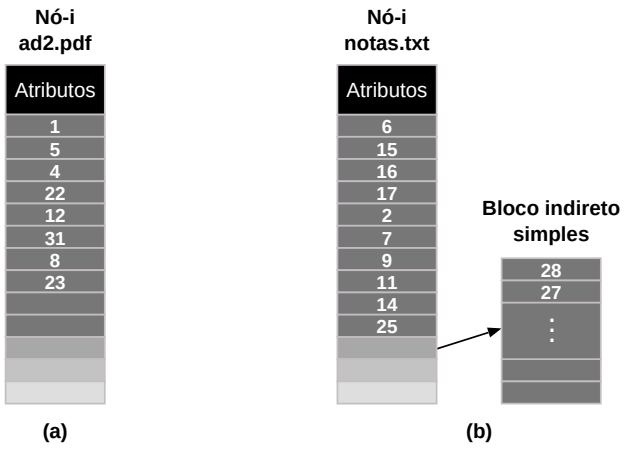
Resp.: Para usarmos o número mínimo de bytes, cada bloco do disco deve ser endereçado com o menor número possível de bits. Como temos 32 ou 2^5 blocos, então precisamos de pelo menos 5 bits para representar um endereço (que, neste caso, varia de 0 até 31). Agora, como o número de entradas na tabela usada pela técnica de alocação da questão é igual ao número de blocos do disco, então o espaço mínimo necessário para armazenar a tabela é de $5 \times 32 = 160$ bits, o que significa que o número mínimo de bytes necessários é de $160/8 = 20$ bytes.

(b) (0,5) É possível realocar os arquivos segundo a alocação contígua sem modificar a alocação do primeiro bloco lógico?

Resp.: Não, como veremos a seguir. Como vimos na aula 11, na alocação contígua, cada arquivo deve ser armazenado em blocos consecutivos do disco. Pela figura vemos que os blocos físicos que armazenam os primeiros blocos lógicos dos arquivos ad2.pdf e notas.txt são, respectivamente, 1 e 6. Além disso, ad2.pdf tem 8 blocos e notas.txt tem 12 blocos. Logo, ad2.pdf deveria ser armazenado nos blocos de 1 até 8 e notas.txt deveria ser armazenado nos blocos de 6 até 17, o que não é possível, porque ambos precisarão ser armazenados nos blocos de 6 até 8.

(c) (1,0) Mostre como serão alocados os arquivos se agora o sistema operacional usar a técnica de alocação baseada em nós-i.

Resp.: Na figura a seguir mostramos como cada arquivo é alocado usando a técnica de alocação baseada em nós-i que vimos na aula 11. Nessa figura, mostramos o nó-i usado por ad2.pdf na parte (a) e o nó-i usado por notas.txt na parte (b). Na figura, somente preenchemos os campos dos nós-i associados aos blocos que foram usados (os campos com os atributos não precisavam ser preenchidos e não serão considerados na correção da questão).



6. (2,0) Suponha que um sistema operacional use uma cache para armazenar 4 blocos do disco, gerenciada pelo algoritmo LRU que vimos na aula 9, sendo que agora blocos, e não páginas, sejam substituídos. Su-

ponha ainda que ler um bloco da cache seja x vezes mais rápido do que lê-lo diretamente do disco. Se um processo acessar, em sequência, os blocos 0, 1, 4, 5, 4, 3, 4, 5 e 9, qual será o ganho de tempo em relação a ler todos estes blocos diretamente do disco, se cada bloco é lido do disco em t unidades de tempo? Justifique a sua resposta.

Resp.: Como o algoritmo usado para gerenciarmos a cache do disco é o LRU, então os blocos são ordenados em ordem crescente de acordo com a última vez em que foram acessados (para leitura ou para escrita). Além disso, quando a cache estiver cheia e precisarmos acessar um bloco que não esteja nela, o primeiro bloco segundo a ordenação será o escolhido para ser removido da cache. Na tabela a seguir mostramos de cima para baixo em cada linha, os blocos acessados, de acordo com a ordem de acesso dada na questão. Para cada linha da tabela, na primeira coluna damos o bloco acessado. Na segunda coluna damos a ordem, segundo o algoritmo LRU, na qual os blocos são escolhidos para serem substituídos, destacando em negrito o bloco a ser escolhido. Finalmente, na última coluna, damos o tempo gasto ao acessarmos o bloco. Suponha que t é o tempo para acessar um bloco diretamente do disco. O tempo de acesso, como podemos ver pelo enunciado da questão, é $\frac{t}{x}$ se o bloco estiver na cache ou t se o bloco não estiver na cache e precisar ser lido do disco. Pela tabela, vemos que o tempo total gasto para acessar todos os blocos é de $6t + \frac{3t}{x}$. Agora, como o tempo total gasto para acessar os blocos sem o uso da cache seria de $9t$, então o ganho é de $9t - \left(6t + \frac{3t}{x}\right) = 9t - 6t - \frac{3t}{x} = 3t - \frac{3t}{x} = 3t\left(\frac{x-1}{x}\right)$.

Páginas	Ordenação	Tempo de acesso
0	**0**	t
1	**0** 1	t
4	**0** 1 4	t
5	**0** 1 4 5	t
4	**0** 1 5 4	$\frac{t}{x}$
3	**1** 5 4 3	t
4	**1** 5 3 4	$\frac{t}{x}$
5	**1** 3 4 5	$\frac{t}{x}$
9	**3** 4 5 9	t