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
AP2 - Primeiro Semestre de 2008

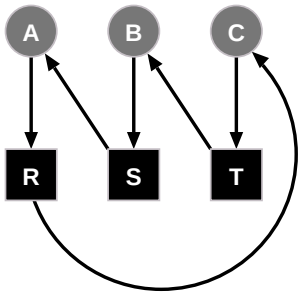
Nome -
Assinatura -

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1. (1.0) Suponha que obtivemos o grafo de recursos dado a seguir, em que três processos A, B e C compartilham três recursos não-preemptivos R, S e T. Como podemos ver pela figura, temos um impasse. Um aluno de sistemas operacionais propôs a seguinte solução para tentar evitar este impasse: um processo somente pode obter S se possuir R, e somente pode obter T se possuir S. Esta solução evita o impasse? Caso o evite, existe alguma limitação?



Resp:. - A solução evitará o impasse, pois não teremos mais um ciclo orientado no grafo como veremos a seguir. O processo A não poderá mais possuir S porque ele não possui R. Do mesmo modo, o processo B não poderá obter T pois não possui S. Finalmente, o processo C não ficará bloqueado, pois ele deseja T e deveria possuir S antes de ser bloqueado esperando por T (note que ele poderá obter S sem ser bloqueado, pois possui R).
- Existem duas limitações. Uma limitação é a de que o processo precisa possuir R para obter S. A outra é que o processo precisa possuir S para obter T, o que significa que para possuir T ele deve possuir R e S. Logo, o processo deverá obter recursos adicionais que ele pode não precisar e, com isso, estes recursos poderão ficar ociosos até que o processo os libere.

2. (1.0) Se pudermos armazenar até 1/4 das páginas virtuais nas molduras de página, qual será a relação entre os números de bits ocupados pelos espaços de endereçamento virtual e físico? Justifique a sua resposta.

Resp:. A relação pedida na pergunta é obtida comparando-se os números de bits dos endereços virtual e físico. Suponha que p é o

número de páginas virtuais, e que m é o número de molduras de página. Como os números de páginas e molduras são potências de 2, então $m = 2^{b_m}$ e $p = 2^{b_p}$, onde $b_m > 0$ é o número de bits do campo moldura de página do endereço físico, e $b_p > 0$ é o número de bits do campo página virtual do endereço virtual. Agora, como podemos armazenar 1/4 das páginas virtuais nas molduras de página, então temos que $p = 4 \times m = 2^2 \times m$. Com isso, $2^{b_p} = 2^2 \times 2^{b_m} = 2^{b_m+2}$, o que implica que $b_p = b_m + 2$. Finalmente, como os tamanhos das molduras de página e das páginas virtuais são idênticos, então o endereço virtual possui dois bits a mais do que o endereço físico.

3. (3.0) Suponha que um computador tem um espaço de endereçamento virtual de 24 bits, e que cada página virtual pode armazenar até 1024 bytes. Suponha ainda que podemos armazenar até a metade das páginas virtuais nas molduras de página. Responda:

   (a) (1.0) Quantas páginas existem no espaço de endereçamento virtual? E quantas molduras existem no espaço de endereçamento físico?

   Resp:. - O tamanho de uma página virtual é de $1024 = 2^{10}$ bytes, o que significa que o campo deslocamento do endereço virtual tem 10 bits. Agora, o número de bits do campo página virtual do endereço virtual é de 14, pois o endereço tem 24 bits de tamanho. Logo, o número de páginas virtuais é de 2^{14}, ou seja, 16384.
   - Como podemos armazenar até a metade das páginas virtuais nas molduras de página, então o número de molduras será exatamente a metade do número de páginas virtuais. Agora, como o número de páginas virtuais é de 16384, então temos 8192 molduras de página no espaço de endereçamento físico.

   (b) (1.0) Qual é o tamanho, em bits, do espaço de endereçamento físico?

   Resp:. Como vimos na resposta do item (a), temos $8192 = 2^{13}$ molduras de página. Agora, como os tamanhos das páginas virtuais e das molduras de página são idênticos, então o endereço físico

tem 23 bits, pois precisamos de 13 bits para representar o número da moldura de página, e 10 bits para representar o deslocamento dentro da moldura.

(c) (1.0) Suponha que duas molduras de página foram alocadas a um processo, e que ele acessa alternadamente três páginas virtuais a, b e c do seguinte modo: a, c, a, b, b e a. Se as molduras alocadas ao processo estiverem inicialmente vazias, quantas falhas de página ocorrerão se o sistema operacional usar o algoritmo LRU para substituir as páginas?

Resp:. Na tabela dada a seguir mostramos, na primeira coluna, a página que foi acessada. Nas colunas 2 e 3 mostramos as páginas das duas molduras alocadas ao processo, ordenadas em ordem crescente de acordo com o tempo do último acesso. Como estamos usando o algoritmo LRU, a página da coluna 2 sempre é a escolhida para ser removida. Finalmente, na coluna 4, dizemos se o acesso à página gerou ou não uma falha de página. A ordem em que as páginas são dadas na tabela é a mesma ordem do enunciado da questão. Como podemos ver pela tabela, somente 3 acessos geraram falhas de página.

Página acessada	Molduras		Ocorreu uma falha?
a	a	-	Sim
c	a	c	Sim
a	c	a	Não
b	a	b	Sim
b	a	b	Não
a	b	a	Não

4. (1.0) Suponha que um sistema operacional use o gerenciamento de memória por segmentação com paginação. Se cada segmento puder ser dividido em 32 páginas de 4KB, quantos segmentos existirão se um programa puder acessar até 4GB de memória?

Resp:. Como cada segmento é dividido em $32 = 2^5$ páginas de 4KB, ou seja, 2^{12} bytes, então o tamanho de cada segmento é de $2^5 \times 2^{12} = 2^{17}$

bytes. Agora, como o programa não pode acessar mais de 4GB, isto é, 2^{32} bytes, e como o tamanho de cada segmento é de 2^{17} bytes, então teremos $2^{32}/2^{17} = 2^{15} = 32768$ segmentos.

5. (3.0) Suponha que um computador possui um disco com 4MB de espaço total, e que cada bloco do disco tem 16KB de tamanho. Responda:

   (a) (1.0) Se usarmos a alocação contígua, poderemos sempre armazenar um arquivo de 1MB no disco se 3MB do disco já estiver sendo usado? Justifique a sua resposta.

   Resp:. Não pois, como vimos na Aula 11, os arquivos devem ser compostos por blocos consecutivos do disco quando usamos a alocação contígua. Logo, somente poderemos armazenar o arquivo se o espaço de 1MB livre no disco for composto por blocos consecutivos deste disco.

   (b) (1.0) Se usarmos a alocação por lista encadeada, poderemos armazenar um arquivo com 4MB no disco, se ele estiver vazio? Justifique a sua resposta.

   Resp:. Não pois, como vimos na Aula 11, a alocação por lista encadeada usa a parte inicial de cada um dos blocos alocados ao arquivo para armazenar o ponteiro para o próximo bloco lógico deste arquivo. Com isso, teremos menos do que 4MB de espaço real disponível no disco para armazenar o arquivo.

   (c) (1.0) Quantos bits da memória principal ocuparia a tabela usada pelo algoritmo de alocação por lista encadeada utilizando um índice? Justifique a sua resposta.

   Resp:. Como o disco tem 4MB de tamanho, isto é, 4096KB, e como cada bloco do disco tem tamanho de 16KB, então o número de blocos no disco é igual a 256, pois $256 = 4096/16$. Além disso, como cada entrada da tabela precisa armazenar um número de bloco, e como temos 256 blocos, então cada entrada tem 8 bits

de tamanho. Finalmente, como o número de entradas na tabela é igual ao número de blocos no disco, então precisamos de 256 (o número de blocos) vezes 8 (o tamanho em bits de uma entrada) bits, ou seja, precisamos de 2048 bits para armazenar a tabela na memória.

6. (1.0) Suponha que um disco tem 256 blocos de 8KB. Se o sistema operacional alocar 64KB de memória para criar uma cache deste disco, qual será a fração de blocos do disco que poderá ser armazenada na cache? Justifique a sua resposta.

   Resp:. Como o tamanho da cache do disco é de 64KB, e como cada bloco tem 8KB de tamanho, então podemos armazenar até $64/8 = 8$ blocos na cache. Agora, como o número total de blocos no disco é de 256, então a fração de blocos que pode ser armazenada na cache é de $8/256 = 0,031250$, isto é, 3,125%.