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
AP1 - Segundo Semestre de 2007

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. (1.5) Qual é a principal desvantagem que temos ao usarmos um sistema de processamento em lote? Qual conceito foi desenvolvido para tentar evitar esta desvantagem?

   Resp.:-A principal desvantagem de um sistema de processamento em lote é a de o processador ser alocado a um programa até que ele termine a sua execução. Neste sistema, se o programa executar uma operação de E/S, o processador ficará ocioso até esta operação terminar. Se vários programas pudessem compartilhar o processador, quando um deles executasse uma operação de E/S, ele poderia ser bloqueado até a operação terminar, e um outro programa poderia executar no processador. Logo, o processador somente ficaria ocioso se nenhum programa no sistema pudesse ser executado (isto é, se todos os programas estivessem bloqueados).
   -O conceito criado para evitar a ociosidade do processador é o da multiprogramação, que permite a vários programas compartilharem o processador e, com isso, evita a ociosidade do processador quando um programa em execução executa uma operação de E/S.

2. (1.5) Como podemos acessar um sistema de arquivos diferente daquele em que reside o sistema operacional? Justifique a sua resposta.

   Resp.: Para podermos acessar os arquivos de um sistema de arquivos diferente daquele em que reside o sistema operacional, devemos usar o conceito da montagem de um sistema de arquivos. Inicialmente, devemos escolher um ponto de montagem, um diretório na hierarquia do sistema de arquivos com o sistema operacional, a partir do qual serão acessados os arquivos do sistema de arquivos a ser montado. Depois disso, executamos uma chamada ao sistema operacional para montar o sistema de arquivos desejado, usando o diretório escolhido como o ponto de montagem. Agora, os arquivos do sistema de arquivos montado poderão ser acessados como se fossem arquivos do sistema de arquivos com o sistema operacional, a partir do diretório escolhido como o ponto de montagem.

3. (1.5) Suponha que um processo executou uma chamada ao sistema ope-

racional. Qual seria a diferença essencial entre uma chamada feita em um sistema em camadas daquela feita em um sistema cliente-servidor? Justifique a sua resposta.

Resp.: Em um sistema baseado em camadas, o núcleo do sistema operacional é o responsável por executar as chamadas ao sistema operacional. Logo, sempre que um processo fizer uma chamada ao sistema operacional, uma instrução TRAP será executada para alternar o modo de execução do processador do modo usuário, em que os processos são executados, para o modo supervisor, em que o núcleo do sistema executa. Já em um sistema cliente-servidor, as chamadas ao sistema operacional são executadas através do envio de uma mensagem pelo processo que faz a chamada (o cliente) a um processo responsável por executar esta chamada (o servidor) . Ambos os processos executam no modo usuário e, com isso, não é necessária a execução de uma instrução TRAP. Note que o núcleo do sistema (chamado de micronúcleo) somente trata da troca de mensagens entre os processos executando no modo usuário, e do acesso aos dispositivos físicos através de mensagens especiais enviadas pelos processos responsáveis pelo gerenciamento dos dispositivos.

4. (1.5) Descreva o que é um modelo de processos, enfatizando como o processador é compartilhado por todos os processos do sistema, e as funções da tabela de processos e da troca de contexto.

Resp.: No modelo de processos, os processos em execução no sistema operacional são organizados como um conjunto de processos seqüenciais. Em geral, o próprio sistema operacional é dividido em processos que são também considerados pelo modelo. Neste modelo, cada processo é associado a um processador virtual, que possui o mesmo estado e os mesmos registradores do processador real do computador, incluindo o contador de programa, que aponta para a próxima instrução a ser executada do processo. O estado e os registradores de cada processador virtual são armazenados (junto com outros dados necessários à execução do processo) na entrada do processo, associado ao processador virtual, em uma tabela chamada de tabela de processos. Cada processo

em execução no sistema possui uma entrada nesta tabela. Para permitir que o processador seja compartilhado por todos os processos, de tempos em tempos, ou quando o processo em execução faz uma operação de E/S, o escalonador do sistema escolhe um novo processo para executar no processador. Para alternar o processador entre os processos e garantir a correta execução dos processos do sistema, o escalonador deverá executar uma troca de contexto, descrita a seguir. O escalonador primeiramente suspende o processo em execução, e salva o estado e os valores dos registradores do processador (além de outros dados necessários à execução do processo) na entrada da tabela de processos associada ao processo suspenso. Depois disso, o escalonador copia o estado e os valores dos registradores do processo escolhido para executar, da sua entrada na tabela de processos (junto com os outros dados necessários à execução do processo), para o estado e os registradores do processador e, em seguida, coloca o processo escolhido para executar no processador.

5. (2.0) Suponha que um conjunto de processos estejam cooperando entre si para executar uma tafera. Cada um dos processos acessa freqüentemente uma variável compartilhada R para obter um identificador. Este identificador deve ser único. Para garantir isso, cada processo lê o valor de R para obter o identificador, depois incrementa o valor obtido em 1 unidade, e depois armazena o novo valor em R. A tafera será executada corretamente somente se garantirmos que cada processo sempre obtenha um identificador único . Responda:

   (a) (1.0) Se permitirmos o acesso irrestriro a R, não poderemos garantir que a tarefa será executada corretamente, pois teremos condições de corrida. Quais são elas?

   Resp.: Somente uma condição de corrida ocorrerá, devido ao fato de cada processo primeiramente ler o valor do identificador de R, para depois armazenar o valor do próximo identificador em R. Como as operações de ler e de atualizar o valor de R não são atômicas, um processo pode ler o valor de R e, antes de ele salvar o novo valor em R, um ou mais processos podem ler este mesmo valor de R. Com isso, dois ou mais processos poderão obter o

mesmo identificador, e a tarefa não será mais corretamente executada.

(b) (1.0) Um aluno de Sistemas Operacionais propôs a seguinte solução para evitar as condições de corrida: usar um semáforo de contagem S_R, inicializado com o número de processos do conjunto. Cada processo, imediatamente antes de ler o valor de R, deve executar a operação P sobre S_R e, imediatamente após armazenar o novo valor em R, executar a operação V sobre S_R. A solução proposta pelo aluno está correta? Justifique a sua resposta.

Resp.: A solução proposta pelo aluno não está correta. O problema não está no uso das operações P e V, que foram corretamente posicionadas no código, mas sim no fato de a proposta usar um semáforo de contagem. Como queremos garantir o acesso exclusivo a R, S_R deverá ser um semáforo binário, para garantir a exclusão mútua. O semáforo deverá ser inicializado com o valor 1, antes de executarmos os processos cooperativos, pois inicialmente R não estará sendo acessado pelos processos.

6. (2.0) Suponha que quatro processos A, B, C e D estão para ser executados no processador, e serão os únicos processos em execução. Responda:

(a) (1.0) Suponha que o sistema operacional usa o escalonamento por *round robin*, e que os processos foram escolhidos pelo escalonador na seguinte ordem, até terminarem suas execuções: A, B, C, D, A, B, C, D, A, B, C, D, A, B, C, B e C. Quais seriam os tempos de execução dos processos, se cada quantum for igual a uma unidade de tempo?

Resp.: Como existem duas possíveis interpretações para o que seria o tempo de execução de um processo, o tempo decorrido até o processo terminar a sua execução, ou a quantidade total de tempo em que o processo executou no processador, existem duas respostas corretas:

- O tempo de execução é tempo decorrido até um processo terminar a sua execução. Neste caso, na tabela dada a seguir

mostramos, na primeira linha, as unidades de tempo e, na segunda linha, os processos. Cada coluna da tabela (com exceção da última coluna, que representa a unidade de tempo após a execução dos processos) representa um quantum, e portanto, uma unidade de tempo. Os processos são mostrados na tabela, da esquerda para a direita, na ordem em que foram escolhidos pelo escalonador. Pela tabela, vemos que o tempo de execução dos processos A, B, C e D são de, respectivamente, 13, 16, 17 e 12 unidades de tempo.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	B	C	-

- O tempo de execução é a quantidade total de tempo em que um processo executou no processador. Neste caso, como cada quantum equivale a uma unidade de tempo, para obtermos os tempos de execução de cada um dos processos, bastará contarmos o número de ocorrências de cada um dos processos na seqüência de execução dada no enunciado da questão. Pelo enunciado, vemos que os tempos de execução dos processos A, B, C e D são de, respectivamente, 4, 5, 5 e 3 unidades de tempo.

(b) (1.0) Suponha agora que o sistema operacional usa o escalonamento por prioridades, e que os processos foram escolhidos pelo escalonador na seguinte ordem, até terminarem suas execuções: D, D, D, A, A, A, C, C, A, B, C, C, B, B, C, B, B. Quais seriam as prioridades dos processos A, B e C, se cada processo executar até que a sua prioridade não seja a maior de todas, e se a prioridade de D for 12?

Resp.: O processo D executa por três vezes consecutivas, e depois termina a sua execução, pois não é mais escolhido pelo escalonador. Logo, a prioridade do processo A pode ser qualquer valor $a \leq 10$, pois a prioridade do processo D seria de 9 após ele terminar a sua execução, e um processo executa até a sua prioridade deixar de ser a maior. Na tabela dada seguir mostramos, na primeira linha, as prioridades dos processos antes de serem escolhidos

pelo escalonador e, na segunda linha, os processos, da esquerda para a direita, na ordem escolhida pelo escalonador. Note que somente precisamos considerar a seqüência de execução até a primeira execução do processo B, pois depois disso já teremos obtido todas as prioridades. Cada coluna da tabela representa uma escolha do escalonador, e portanto, um decréscimo na prioridade. Como podemos ver pela tabela, se a prioridade do processo A for a, então as prioridades dos processos B e C serão de, respectivamente, $a - 3$ e $a - 2$.

12	11	10	a	$a-1$	$a-2$	$a-2$	$a-3$	$a-3$	$a-3$
D	D	D	A	A	A	C	C	A	B