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
Gabarito da AD1 - Segundo Semestre de 2017

Atenção: 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.

Nome -
Assinatura -

---

1. (1,5) Suponha que um programa A precise executar no processador por 7s, e que ele faça duas operações de E/S com tempos de, respectivamente, 400ms e 750ms, após A executar por, respectivamente, 2s e 4s no processador. Responda, justificando a sua resposta, supondo que um programa B fique pronto para executar no processador ao mesmo tempo que A:

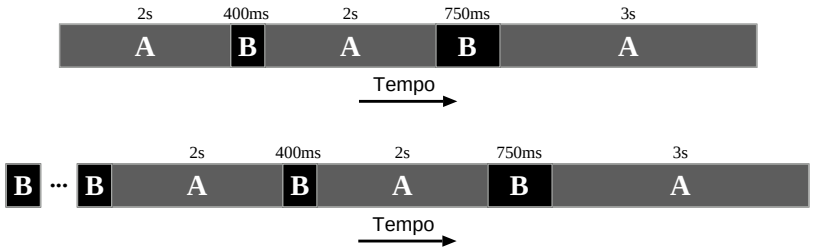
   (a) (0,5) Qual será o tempo de ociosidade do processador se a multiprogramação não for usada e B não fizer operações de E/S? Essa ociosidade dependerá da ordem de execução dos programas, A e

B, e do tempo de execução de B no processador?

Resp.: Se a multiprogramação não for usada, não teremos como evitar a ociosidade do processador durante uma das operações de E/S do programa A usando o programa B. Logo, o tempo de ociosidade do processador, que será igual à soma dos tempos das operações de E/S feitas por A, ou seja, $400 + 750 = 1150$ms, não dependerá da ordem de execução de A e B no processador e nem do tempo de execução de B no processador.

(b) (1,0) Como a ociosidade do processador ao executar as operações de E/S poderá ser evitada se a multiprogramação for usada somente para evitar essa ociosidade?

Resp.: Para evitar a ociosidade do processador usando a multiprogramação, o programa B precisa estar pronto para executar durante o tempo em que A faz a sua primeira operação de E/S. Nas figuras a seguir mostramos os dois possíveis modos de usar o programa B para evitar a ociosidade do processador, usando os menores tempos possíveis para B durante as operações de E/S feitas por A. Quando o programa A executar antes de B, como A faz duas operações de E/S, então B precisará fazer somente uma operação de E/S, cujo tempo não poderá ser maior do que o tempo de 2s entre as duas operações de E/S feitas por A. Além disso, B precisará execurar por pelo menos 400ms antes de fazer a sua operação de E/S e, depois de fazer essa operação, por pelo menos 750ms. Agora, se B executar antes de A, então B precisará fazer duas operações de E/S, sendo que a primeira não poderá ter tempo maior do que o tempo de 2s em que A executa antes de fazer a sua primeira operação de E/S, e a segunda não poderá ser maior do que o tempo de 2s em que A executa entre as suas duas operações de E/S. Além disso, o tempo de execução de B entre as duas operações de E/S deverá ser pelo menos igual ao tempo de 400ms da primeira operação de E/S de A e o tempo em que B executa após a sua segunda operação de E/S, até terminar a sua execução, deverá ser pelo menos igual ao tempo de 750ms da segunda operação de E/S de A.



2. (1,5) Suponha que um processador possa executar 5 400 000 instruções por segundo, e que 5/6 da capacidade do processador sejam usados exclusivamente para executar códigos do usuário. Se cada chamada ao sistema operacional requer 450 instruções (incluindo a instrução TRAP e todas as outras instruções necessárias à troca de contexto) para ser executada, quantas dessas chamadas por segundo podem ser feitas no máximo? Justifique a sua resposta.

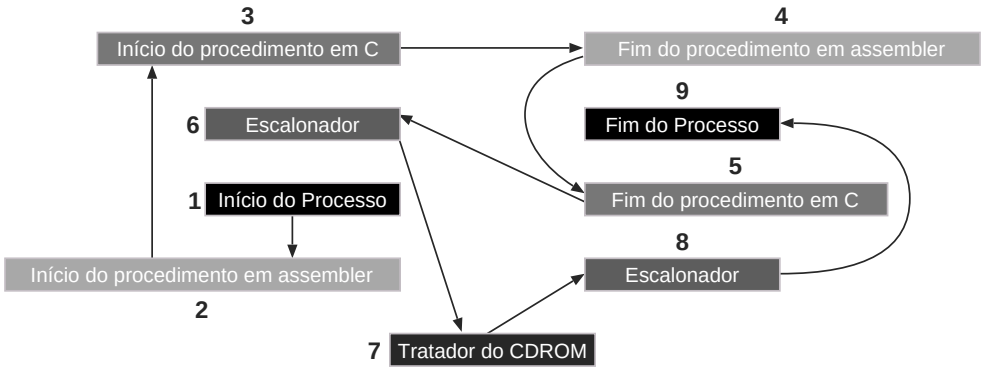
   Resp.: Se denotarmos por x o número máximo de chamadas por segundo que podem ser feitas, então o processador vai executar $450x$ instruções por segundo ao executar x chamadas ao sistema por segundo. Agora, como 5/6 da capacidade do processador estão disponíveis para executar códigos do usuário, então 1/6 da capacidade está disponível para executar as chamadas ao sistema, ou seja, o processador pode executar $5\,400\,000/6 = 900\,000$ instruções por segundo para tratar as chamadas. Logo, temos que $450x = 900\,000$ e, portanto, $x = 2000$, significando que podemos executar no máximo 2000 chamadas por segundo.

3. (1,5) Suponha que o sistema operacional esteja executando diretamente sobre o hardware de um computador cujas operações de E/S demoram 0,2ms. Suponha ainda que um processo tenha executado por 8 350ms e que, durante a sua execução, tenha feito 5 000 operações de E/S. Se o sistema operacional agora executar sobre uma máquina virtual que reduza a velocidade do processador em $x\%$ e a velocidade das operações de E/S em 60%, qual deverá ser esse fator de redução x, supondo que o processo tenha executado 4 000 operações de E/S na máquina virtual e que o tempo total de execução na máquina virtual tenha sido de

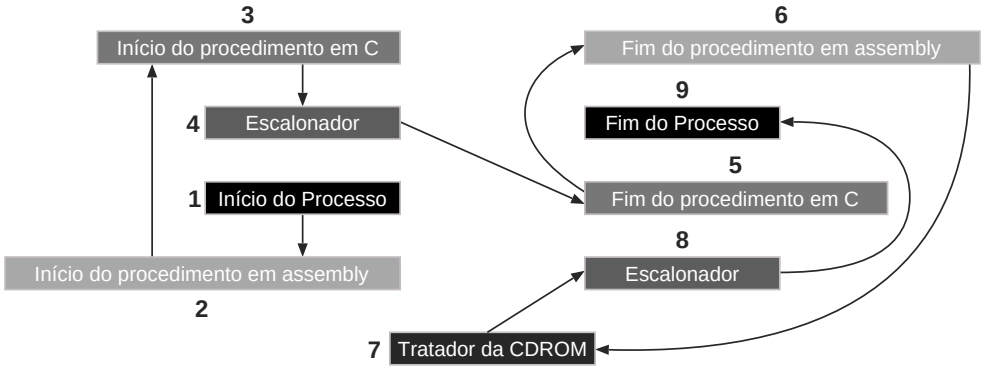
12 500ms? Justifique a sua resposta.

Resp.: Como o tempo total de execução é de 8 350 ms, e como o processo faz 5 000 operações de E/S com duração de 0,2ms, então 1 000 ms dos 8 350 ms são gastos com operações de E/S, quando a execução ocorre sobre o hardware do computador. Logo, o processo executa no processador do hardware por $8\,350 - 1\,000 = 7\,350$ ms. Note que a velocidade do processador ser reduzida em $x\%$ significa que a velocidade do processador virtual é $(100 - x)\%$ da velocidade do processador real, o que por sua vez significa que, durante os 7 350ms, somente $(100-x)\%$ das instruções são executadas. Com isso, quando o processo executa sobre a máquina virtual, seu tempo de execução no processador virtual é de $\frac{7\,350}{\frac{100-x}{100}} = \frac{735\,000}{100-x}$ ms. Agora, como o processo executa 4 000 operações de E/S na máquina virtual, e como o novo tempo de cada operação de E/S é de $\frac{0,2}{0,4} = 0,5$ ms (já que, similarmente à redução do tempo do processador, a redução da velocidade de cada operação de E/S em 60% significa que no mesmo tempo podemos, na máquina virtual, executar somente 40% das operações de E/S originais), então $4\,000 \times 0,5 = 2\,000$ ms dos 12 500 ms do tempo de execução do processo na máquina virtual são gastos com E/S. Logo, o tempo de execução do processo no processador virtual é de $12\,500 - 2\,000 = 10\,500$ ms e, com isso, podemos concluir que $\frac{735\,000}{100-x} = 10\,500$, ou seja, $x = 30$, implicando que a velocidade do processador virtual é reduzida em 30%.

4. (1,5) Suponha que tenha ocorrido uma interrupção da CDROM enquanto um processo estava em execução. Um aluno de sistemas operacionais disse que a figura a seguir mostra a ordem correta das ações realizadas quando essa interrupção foi tratada pelo sistema operacional. Se você acha que a figura do aluno está correta basta responder que sim mas, se você acha que ela está errada, então indique quais são os erros.



Resp.: A figura do aluno está errada porque, apesar de a ordem das ações de 1 até 3 e de 7 até 9 estar correta, a ordem das ações de 3 até 7 está errada. Após a ação 3, o escalonador é chamado para escolher o "Tradador da CDROM"para ser executado e tratar da interrupção. Logo, a próxima ação "Escalonador"dada na parte esquerda da figura é a executada. Após isso, as ações 5, "Fim do procedimento em C", e 6, "Fim do procedimento em Assembly", são executadas, em ordem, para fazer as finalizações necessárias para executar o "Tratador da CDROM". Finalmente, o tratador é executado pela ação 7 e, após isso, a ordem das ações é a mesma dada pelo aluno. A seguir está a figura com as ações executadas na ordem correta:



5. (2,0) Suponha que dois processos, A e B, compartilhem uma região de memória R que pode armazenar um número, e que B também compartilhe, com um outro processo C, uma fila F que pode armazenar

até $n \geq 2$ números. O processo A continuamente escreve um número em R se o número anterior ainda não foi lido. Já o processo B continuamente lê um número ainda não lido por ele de R e depois copia esse número para o final de F se F não está cheia. Finalmente, o processo C continuamente remove, se existirem, os dois números iniciais de F, e imprime a soma deles na tela. Explique como dois semáforos de contagem e três semáforos binários podem ser usados para evitar possíveis condições de corrida e garantir o correto funcionamento de A, B e C, supondo que inicialmente não existe um número armazenado em R, que F está vazia, e que as seguintes funções estão disponíveis para serem usadas pelos processos: leR lê o número de R; $escreveR(v)$ copia para R o número v; $removeF$ remove o número do início de F; e $insereF(v)$ insere o número v no final de F. Justifique a sua resposta.

Resp.: A seguir mostramos como os cinco semáforos, três binários e dois de contagem, podem ser usados para implementar os códigos dos processos. O primeiro semáforo binário, chamado $escrever_R$, é usado para garantir que A possa escrever um novo número em R, ou seja, para garantir que um novo número seja escrito somente se B já leu o número anteriormente armazenado em R. Já o segundo semáforo binário, chamado ler_R, é usado para garantir que B somente possa ler novo um número de R após A tê-lo armazenado. Finalmente, o terceiro semáforo binário, chamado $acesso_F$, é usado para garantir o acesso exclusivo à fila F. Agora, o primeiro semáforo de contagem, chamado $vazia_F$, conta o número de entradas não usadas por elementos na fila, e é usado para bloquear o processo B quando a fila está cheia. Finalmente, o segundo semáforo de contagem, chamado $cheia_F$, conta o número de entradas com elementos na fila, e é usado para bloquear o processo C quando a fila está vazia. Como inicialmente a região de memória R e a fila F estão vazias e não estão sendo usadas, então os semáforos $escrever_R$, ler_R, $acesso_F$, $vazia_F$ e $cheia_F$ serão inicializados, respectivamente, com 1, 0, 1, n e 0. A seguir mostramos os códigos para os processos A, B e C:

```
void ProcessoA(void)
{
  while(1)
  {
    // Código para gerar um número e salvá-lo em v.
    // Usa a operação P sobre escrever_R para garantir que podemos escrever
    // um número em R.
    P(escrever_R);
    escreveR(v);
    // Usa a operação V sobre ler_R para indicar que um novo número foi
    // armazenado em R.
    V(ler_R);
  }
}

void ProcessoB(void)
{
  while(1)
  {
    // Usa a operação P sobre ler_R para garantir que podemos ler um
    // número de R.
    P(ler_R);
    // Lê o número v de R.
    v = leR();
    // Usa a operação V sobre escrever_R para indicar que um novo número
    // pode ser armazenado em R.
    V(escrever_R);
    // Garante que podemos armazenar pelo menos um número em F.
    P(vazia_F);
    // Garante o acesso exclusivo a F.
    P(acesso_F);
    // Insere o número v no final de F.
    insereF(v);
    // Libera o acesso exclusivo a F.
    V(acesso_F);
    // Usa a operação V sobre cheia_F para registrar que um número foi
    // inserido em F.
    V(cheia_F);
  }
}
```

```
void ProcessoC(void)
{
  while(1)
  {
    // Usa a operação P sobre cheia_F para garantir que pelo menos dois
    // números existam em F.
    P(cheia_F);
    P(cheia_F);
    // Garante o acesso exclusivo a F.
    P(acesso_F);
    // Remove dois números, v e u, do início de F.
    v = removeF();
    u = removeF();
    // Libera o acesso exclusivo a F.
    V(acesso_F);
    // Usa a operação V sobre vazia_F para registrar que dois número foram
    // removidos de F.
    V(vazia_F);
    V(vazia_F);
    // Código para imprimir a soma de v e u.
  }
}
```

6. (2,0) Quatro processos, A, B, C e D, foram inicializados e têm tempos de execução no processador de, respectivamente, 11, 7, 4 e 9 segundos. Para cada um dos seguintes algoritmos de escalonamento, determine a média dos tempos de término dos processos. Ignore o acréscimo (*overhead*) da comutação de processos e suponha que nenhum processo faça operações de E/S. Justifique a sua resposta.

   (a) (0,5) *Round robin* com um quantum de 3 segundos de duração, e com os processos inicialmente executando na ordem B, C, A e D.

   Resp.: Pelo enunciado, vemos que a ordem de execução dos processos é como dada na tabela a seguir. Nesta tabela mostramos como os processos são escolhidos pelo algoritmo, sendo que cada coluna refere-se à execução de um processo dando o tempo de início de cada quantum e o processo correspondente. Devido a os tempos dos processos A, B e C não serem múltiplos do tamanho do quantum de 3 segundos, A somente usa 2 segundos e B e C somente usam 1 segundo dos seus últimos quanta. Pela tabela,

vemos que os tempos de término dos processos A, B, C e D, são de, respectivamente, 31, 23, 16 e 29 segundos, o que nos dá um tempo médio de 24,75 segundos.

0	3	6	9	12	15	16	19	22	23	26	29
B	C	A	D	B	C	A	D	B	A	D	A

(b) (1,0) Escalonamento por prioridades, supondo que a prioridade do processo em execução seja reduzida por 2 unidades a cada 2 segundos, que um processo em execução somente seja suspenso quando um outro processo passa a ter a maior prioridade, e que as prioridades iniciais dos processos A, B, C e D sejam de, respectivamente, 3, 8, 0, 12.

Resp.: As tabelas a seguir são similares à tabela do item anterior e possuem somente mais uma linha, a terceira, que mostra a prioridade de cada processo antes de ele executar no segundo dado na mesma coluna. Novamente, devido a os tempos dos processos A, B e D serem ímpares e, consequentemente, não serem múltiplos do tamanho do tempo de 2 segundos usado ao reduzir as prioridades, eles usarão somente 1 segundo antes de terminarem. Agora, pelas tabelas, vemos que os tempos de término dos processos A, B, C e D, são de, respectivamente, 31, 18, 26 e 13 segundos, o que nos dá um tempo médio de 22 segundos.

0	2	4	6	8	10	12	13	15
D	D	D	B	B	D	D	B	A
12	10	8	8	6	6	4	4	3

17	18	20	22	24	26	28	30
B	A	C	A	C	A	A	A
2	1	0	-1	-2	-3	-5	-7

(c) (0,5) Trabalho mais curto primeiro.

Resp.: Como vimos na aula 6, no algoritmo do trabalho mais curto primeiro, os processos são executados, em ordem crescente, de acordo com os seus tempos de execução. Além disso, quando um processo começa a executar, ele executa exclusivamente no processador até terminar. Então a única possível ordem de execução é C, B, D e A. O processo C então executa do tempo 0 até o 4, o processo B do tempo 4 até o 11, o processo D do tempo 11 até o 20 e, finalmente, o processo A do tempo 20 até o 31. Logo, os tempos de término dos processos A, B, C e D são de, respectivamente, 31, 11, 4, e 20 segundos, o que nos dá um tempo médio de 16,5 segundos.