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 2019

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.

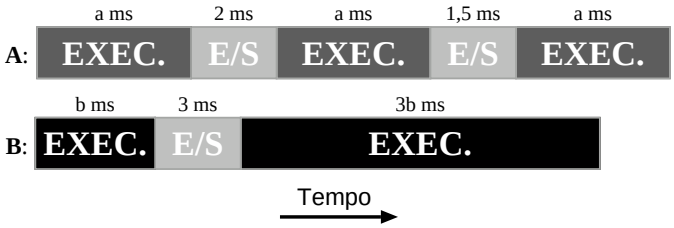
Nome -
Assinatura -

---

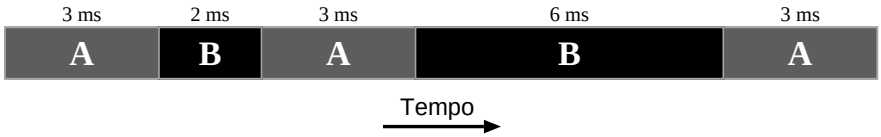
1. (1,5) Suponha que um programa A, que precisa executar no processador por $3a$ ms, faça duas operações de E/S com durações de, respectivamente, 2 ms e 1,5 ms, sendo que a primeira é feita após A executar por 1/3 do seu tempo de execução, e que a segunda é feita após A executar por mais 1/3 do seu tempo de execução. Suponha ainda que um programa B, que precisa executar no processador por $4b$ ms, faça uma operação de E/S, com duração de 3 ms, após executar por 1/4 do seu tempo de execução. Se a multiprogramação for usada somente para evitar a ociosidade do processador quando operações de E/S são feitas, e se um programa C, que não faz operações de E/S, executar por

c ms somente quando não for possível evitar a ociosidade do processador com A e B executando, quais serão os menores valores que a, b e c poderão assumir para evitar completamente a ociosidade? Justifique a sua resposta.

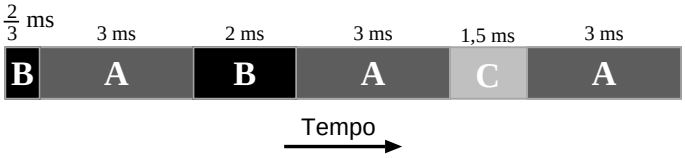
Resp.: Pelo enunciado, vemos que o programa A executará por a ms antes de fazer a primera operação de E/S, com duração de 2 ms, por mais a ms entre a primeira e a segunda operação de E/S, e finalmente por mais a ms após o término da segunda operação, com duração de 1,5 ms. Já o programa B executará por b ms antes de fazer a sua operação de E/S, com duração de 3 ms, e por mais $3b$ ms depois de fazer essa operação. A figura a seguir ilustra esse comportamento de A e de B.



Se A começar a executar antes de B então, para evitar completamente a ociosidade do processador, o tempo de b ms em que B executa antes de fazer a sua operação de E/S deverá ser pelo menos igual ao tempo da primeira operação de E/S feita por A, que é de 2 ms. Além disso, o tempo de a ms em que A executa após fazer a sua primeira operação de E/S deverá ser pelo menos igual ao tempo da operação de E/S feita por B, que é de 3 ms. Como o tempo da segunda operação de E/S de A é menor que o tempo da primeira operação de E/S, e como B executa novamente por $3b$ ms após fazer a sua operação de E/S, então não é necessário iniciar a execução de C para evitar a ociosidade do processador. Podemos concluir que $a \geq 3$ e $b \geq 2$ neste primeiro caso, conforme ilustrado abaixo para $a = 3$ e $b = 2$.

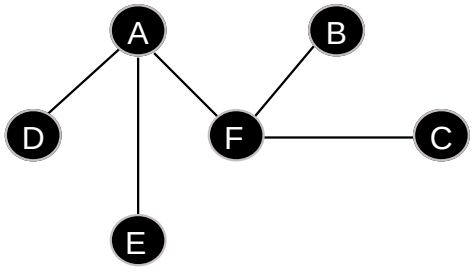


Agora, se B começar a executar antes de A então, para evitar completamente a ociosidade do processador, o tempo de a ms em que A executa antes de fazer a sua primeira operação de E/S deverá ser pelo menos igual ao tempo da operação de E/S feita por B, que é de 3 ms. Adicionalmente, o tempo de $3b$ ms em que B executa após fazer a sua operação de E/S deverá ser pelo menos igual ao tempo da primeira operação de E/S feita por A, que é de 2 ms. Além disso, como B já terá terminado sua execução quando A iniciar a sua segunda operação de E/S, então precisaremos executar C em paralelo com essa operação. Com isso, o tempo de c ms em que C executa no processador deverá ser pelo menos igual ao tempo da segunda operação de E/S feita por A, que é de 1,5 ms. Podemos concluir que $a \geq 3$, $b \geq \frac{2}{3}$ e $c \geq 1,5$ neste segundo caso, ilustrado abaixo para $a = 3$, $b = \frac{2}{3}$ e $c = 1,5$.

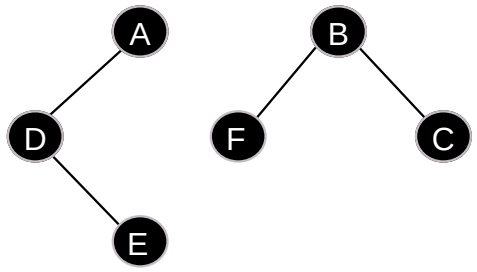


Finalmente, temos $a \geq 3$, $b \geq \max\{2, \frac{2}{3}\} = 2$ e $c \geq 1,5$. Como desejamos os menores valores possíveis para a, b e c, então $a = 3$, $b = 2$ e $c = 1,5$ (note que C somente precisa executar por 1,5 ms no segundo caso).

2. (1,0) Um aluno de sistemas operacionais disse que a hierarquia dada na figura a seguir relaciona corretamente os processos A, B, C, D, E e F em execução no sistema operacional. A hierarquia do aluno está correta? Justifique a sua resposta.



Resp.: A hierarquia do aluno não está correta porque existem três erros nela. O primeiro erro é que, estando no mesmo nível, C e F não podem estar relacionados um com o outro, o que implica em que não deveria existir uma aresta entre eles. O segundo erro é que a aresta entre A e E não faz sentido, já que o pai de E precisa estar em um nível imediatamente acima dele. Finalmente, o último erro é que F somente pode ter um pai (A ou B), porque cada processo filho é criado por um único pai. Uma possível alternativa à hierarquia dada é a seguinte:

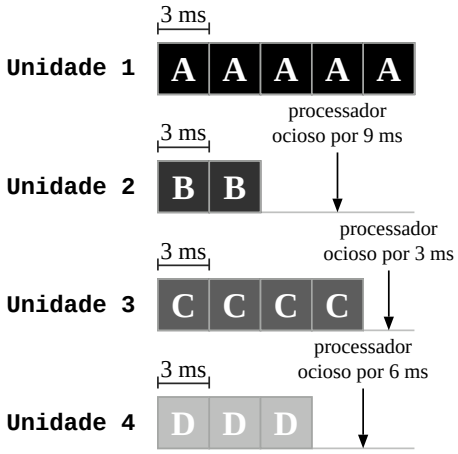


3. (2,0) Suponha que o sistema operacional esteja executando diretamente sobre o hardware de um computador onde cada operação de E/S demore x ms. Suponha ainda que um processo tenha executado por 13 000 ms e que, durante a sua execução, tenha feito 1 950 operações de E/S. Se o sistema operacional agora executar sobre uma máquina virtual que reduza a velocidade do processador em 35% e a velocidade das operações de E/S em 60%, e se além disso forem feitas 1 550 operações de E/S a mais do que sobre o hardware, para que valor de x o tempo de execução do processo na máquina virtual aumentará, em relação ao hardware, por 15 050 ms? Justifique a sua resposta.

Resp.: Como o tempo total de execução é de 13 000 ms, e como o processo faz 1 950 operações de E/S com duração de x ms cada, então $1\,950x$ ms dos 13 000 ms são gastos com operações de E/S quando a execução ocorre sobre o hardware do computador. Logo, o processo executa no processador desse hardware por $(13\,000 - 1\,950x)$ ms. Note que a velocidade do processador ser reduzida em 35% significa que a velocidade do processador virtual é 65% da velocidade do processador real, o que por sua vez significa que, durante os $(13\,000 - 1\,950x)$

ms, somente 65% das instruções podem ser executadas. O tempo necessário para executar 100% das instruções sobre a máquina virtual é de $\frac{13\,000-1\,950x}{0{,}65} = (20\,000 - 3000x)$ ms. Agora, como o processo faz $1\,950 + 1\,550 = 3\,500$ operações de E/S na máquina virtual, e como o novo tempo de cada operação de E/S é de $\frac{x}{0{,}4} = 2,5x$ ms (já que, similarmente à redução da velocidade 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 $3\,500 \times 2,5x = 8\,750x$ ms dos $13\,000 + 15\,050 = 28\,050$ ms de execução do processo na máquina virtual são gastos com E/S. Logo, o tempo de execução do processo no processador virtual é de $(28\,050 - 8\,750x)$ ms. Usando $20\,000 - 3000x = 28\,050 - 8\,750x$, temos $x = 1,4$. Logo, o tempo de cada operação de E/S no hardware é de 1,4 ms.

4. (1,5) Suponha que quatro processos, A, B, C e D, tenham sido executados como na figura a seguir, em um computador com quatro unidades de processamento. Suponha ainda que o escalonador coloque um processo para executar novamente no processador, se necessário, por mais 3 ms, somente após todos os outros processos alocados àquele processador terem tido a chance de executar por 3 ms. Responda, justificando a sua resposta:

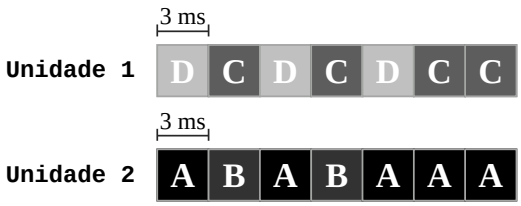


(a) (0,7) Como será a execução se agora existir somente uma unidade de processamento, e se os processos começarem na ordem A, D, B e C? Quais serão os tempos de término de cada processo?

Resp.: Pela figura do enunciado, vemos que os tempos de execução dos processos A, B, C e D são de, respectivamente, 15 ms, 6 ms, 12 ms e 9 ms. Na figura a seguir mostramos a ordem de execução obtida quando a ordem inicial é A, D, B e C. Pela ordem dada na figura, vemos que os tempos de término dos processos A, B, C e D serão de, respectivamente, 42 ms, 21 ms, 39 ms e 30 ms.

(b) (0,8) Como será a execução se agora existirem duas unidades de processamento, e se supusermos que os processos C e D sempre executem na unidade 1, com D começando a executar antes de C, e que os processos A e B sempre executem na unidade 2, com A começando a executar antes de B? Quais serão os tempos de término de cada processo?

Resp.: Usando novamente os tempos de execução dos processos A, B, C e D, iguais a, respectivamente, 15 ms, 6 ms, 12 ms e 9 ms, e lembrando que C e D sempre executam na unidade 1, que A e B sempre executam na unidade 2, que D começa antes de C, e que A começa antes de B, então teremos, para cada unidade, a ordem de execução dada na figura a seguir. Pelas ordens dadas na figura, vemos que os tempos de término dos processos A, B, C e D serão de, respectivamente, 21 ms, 12 ms, 21 ms e 15 ms.



5. (2,0) Suponha que n palavras de memória R_i, $1 \leq i \leq n$, sejam compartilhadas pelos processos A, B e C, sendo que todas são inicializadas com o valor 0. O processo A continuamente escolhe uma palavra R_a de modo aleatório, espera que seu valor se torne igual a 0, e depois disso salva nela um novo valor maior que 0. Já o processo B continuamente escolhe uma palavra R_b de modo aleatório, espera que seu valor se torne diferente de 0, e depois disso salva esse valor em um arquivo e altera o valor da palavra para 0. Finalmente, o processo C continuamente decrementa de 1 unidade o valor de cada palavra R_c, após esperar que seu valor se torne diferente de 0. Como $2n$ semáforos binários podem ser usados para garantir que os processos executem sem condições de corrida ou impasses? Justifique a sua resposta.

Resp.: Para cada palavra R_i vamos precisar de 1 semáforo binário, chamado de $zero_i$, usado para detectar se o valor da palavra é igual a 0. Com isso, o semáforo associado à palavra escolhida por A é usado para bloqueá-lo se o valor dessa palavra é diferente de 0. Já cada um dos n semáforos binários restantes, chamado de $naozero_j$, é usado para detectar se a palavra R_j tem valor diferente de 0. Com isso, o semáforo associado à palavra escolhida por B é usado para bloqueá-lo se o valor dessa palavra é igual a 0, bem como para bloquear C nessa mesma situação. Como inicialmente todas as palavras têm valor igual a 0, então todos os semáforos $zero_i$, $1 \leq i \leq n$, são inicializados com 1 e todos os todos os semáforos $naozero_j$, $1 \leq j \leq n$, são inicializados com 0. A seguir mostramos os pseudocódigos para os processos A, B e C, usando os semáforos descritos anteriormente.

```
void ProcessoA(void)
{
  while(1)
  {
    // Código para escolher aleatoriamente uma palavra R_a, 1 ≤ a ≤ n.
    // Espera o valor da palavra R_a ser igual a 0.
    P(zero_a);
    // Código para escolher aleatoriamente um valor maior do que 0, para depois
    // armazená-lo na palavra R_a.
    // Usa a operação V sobre naozero_a para registrar que um valor diferente de 0
    // foi armazenado em R_a.
    V(naozero_a);
  }
}

void ProcessoB(void)
{
  while(1)
  {
    // Código para escolher aleatoriamente uma palavra R_b, 1 ≤ b ≤ n.
    // Espera o valor da palavra R_b ser diferente de 0.
    P(naozero_b);
    // Código para ler o valor da palavra R_b, e depois salvá-lo no arquivo.
    // Código para copiar o valor 0 na palavra R_b.
    // Usa a operação V sobre zero_b para registrar que um valor igual a 0 foi
    // armazenado em R_b.
    V(zero_b);
  }
}
```

```
void ProcessoC(void)
{
  while(1)
  {
    for(c = 1; c ≤ n; c++)
    {
      // Espera o valor da palavra R_c ser diferente de 0.
      P(naozero_c);
      // Código para armazenar o valor da palavra R_c em v.
      // Código para armazenar o valor v − 1 na palavra R_c.
      // Observe que agora o valor da palavra R_c pode ser igual ou diferente de 0,
      // sendo que em cada caso deveremos executar a operação V sobre o
      // semáforo correto.
      if(v − 1 == 0)
      {
        // Usa a operação V sobre zero_c porque agora o valor da palavra R_c é
        // igual a 0.
        V(zero_c);
      }
      else
      {
        // Usa a operação V sobre naozero_c porque o valor da palavra R_c ainda é
        // maior do que 0.
        V(naozero_c);
      }
    }
  }
}
```

6. (2,0) Suponha que três processos, A, B e C, que não fazem operações de E/S e que precisam executar por, respectivamente, 12 ms, 7 ms e 9 ms, sejam executados em dois sistemas operacionais. O primeiro sistema usa o algoritmo por *rounb robin*, com um **quantum** de 3 ms, ao escalonar os processos. Já o segundo sistema usa o algoritmo por prioridades, sendo que cada prioridade é incrementada em 1 unidade a cada 2 ms e que o processo com a menor prioridade executa no processador até existir um outro processo com uma prioridade menor do que a dele. Suponha ainda que, no primeiro sistema, os processos executem inicialmente na ordem B, C, A e que, no segundo sistema, as prioridades iniciais de A, B e C sejam de, respectivamente, 3, 6 e 0. Para cada sistema, dê a ordem de execução dos processos e, para cada processo, diga em qual sistema ele executará mais rapidamente. Justifique a sua

resposta.

Resp.: Usando os tempos dados no enunciado e o modo de executar o algoritmo por *round robin* descrito para o primeiro sistema, vemos que a ordem de execução dos processos nesse sistema será como a dada na primeira tabela a seguir. Nessa 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 e o processo correspondente. Pela tabela, vemos que a ordem de execução será BCABCABCAA e que os tempos de término de A, B e C serão de, respectivamente, 28 ms, 19 ms e 22 ms. Já a segunda tabela mostra a ordem de execução obtida para o segundo sistema, usando novamente os tempos dados no enunciado e o modo de executar o algoritmo por prioridades descrito. Para cada processo, o número dado na linha adicional dessa tabela mostra a prioridade cujo valor define que o processo deverá executar no processador. Pela tabela, vemos que a ordem de execução será CCCCAACAABBAABB e que os tempos de término de A, B e C serão de, respectivamente, 25 ms, 28 ms e 13 ms. Baseado nos tempos de término dos processos obtidos ao executá-los nos dois sistemas, vemos que A e C terminarão mais rapidamente no segundo sistema, e que B terminará mais rapidamente no primeiro sistema.

0	3	6	9	12	15	18	19	22	25	28
B	C	A	B	C	A	B	C	A	A	-

0	2	4	6	8	10	12	13	15	17	19	21	23	25	27	28
C	C	C	C	A	A	C	A	A	B	B	A	A	B	B	-
0	1	2	3	3	4	4	5	6	6	7	7	8	8	9	-