Sistemas Operacionais
Segunda Lista de Exercícios – Solução

Norton Trevisan Roman

14 de novembro de 2013

2. $n \cdot T$

4. (a) Com $Q = \infty$ cada processo roda até o fim. O ciclo então será rodar T e desperdiçar S no escalonamento. A eficiência será então $\frac{T}{T+S}$

   (b) Com $Q > T$, o comportamento é idêntico, uma vez que, para o processo, houve a interrupção do processamento de qualquer forma.

   (c) De um modo geral, com $Q < T$, o sistema fará $\lceil \frac{T}{Q} \rceil$ trocas, ao rodar até T (já incluindo a troca em T). Assim, para rodar T, o sistema desperdiçará $\lceil \frac{T}{Q} \rceil \cdot S$ de tempo da CPU. Sua eficiência será então $\frac{T}{T+\lceil \frac{T}{Q} \rceil \cdot S}$.

   No caso de $Q = S$, a eficiência será $\frac{T}{T+\lceil \frac{T}{S} \rceil \cdot S} = \frac{1}{2} = 0,5$

   No caso de $Q = T$, será $\frac{T}{T+\lceil \frac{T}{T} \rceil \cdot S} = \frac{T}{T+S}$ ou $\frac{Q}{Q+S}$

   Então a eficiência ficará entre $0,5$ e $\frac{Q}{Q+S}$

   (d) Foi calculada na (c). É $0,5 = 50\%$

   (e) Quando $Q \to 0$, teremos

$$\lim_{Q \to 0} \frac{T}{T + \lceil \frac{T}{Q} \rceil \cdot S} = 0$$

5. Minimizar o tempo médio de retorno envolve ordenar os serviços. Então a ordem será 3,5,6,9, com X inserido na posição correta.

6. (a) Supondo a fila do round robin como sendo A, B, C, D e E, com quantum de 1 min.:

Tempo Inicial	Situação *A*	*B*	*C*	*D*	*E*	*Tempo Final*	*Obs.*
	10	6	2	4	8	0’	Situação inicial
0’	9	5	1	3	7	5’	Situação após 5’
5’	8	4	0	2	6	10’	Após 8’, C termina
10’	7	3		1	5	14’	
14’	6	2		0	4	18’	Após 17’, D termina
18’	5	1			3	21’	
21’	4	0			2	24’	Após 23’, B termina
24’	3				1	26’	
26’	2				0	28’	Após 28’, E termina
28’	1					29’	
29’	0					30’	Após 30’, A termina

   Temos então

$$\frac{8 + 17 + 23 + 28 + 30}{5} = \frac{106}{5} = 21,2$$

(b) Por prioridade, a ordem de execução será B, E, A, C, D.

B	6	=	6
E	8+6	=	14
A	10+8+6	=	24
C	2+10+8+6	=	26
D	4+2+10+8+6	=	30

$$\frac{6+14+24+26+30}{5} = \frac{100}{5} = 20$$

(c) A ordem de execução será A, B, C, D, E.

A	10	=	10
B	6+10	=	16
C	2+6+10	=	18
D	4+2+6+10	=	22
E	8+4+2+6+10	=	30

$$\frac{10+16+18+22+30}{5} = \frac{96}{5} = 19,2$$

(c) A ordem de execução será C, D, B, E, A.

C	2	=	2
D	4+2	=	6
B	6+4+2	=	12
E	8+6+4+2	=	20
A	10+8+6+4+2	=	30

$$\frac{2+6+12+20+30}{5} = \frac{70}{5} = 14$$

7. Quando for posto para rodar, entra na primeira fila (quantum = 1). Rodado seu quantum, vai para a segunda fila (rodando 2 quanta). Findado seu tempo, vai para a terceira fila e, quando posto novamente para rodar, rodará 4 quanta, e assim por diante.

   Então, ele irá rodar 1 + 2 + 4 + 8 + 15 (dos 16 possíveis), sendo escalonado 5 vezes.

   Note que matematicamente, como a cada escalonamento ele ganha o dobro do tempo, ele será escalonado $\log_2 30 = 4,907 \approx 5$ vezes para rodar seus 30 quanta.

8. Temos os eventos

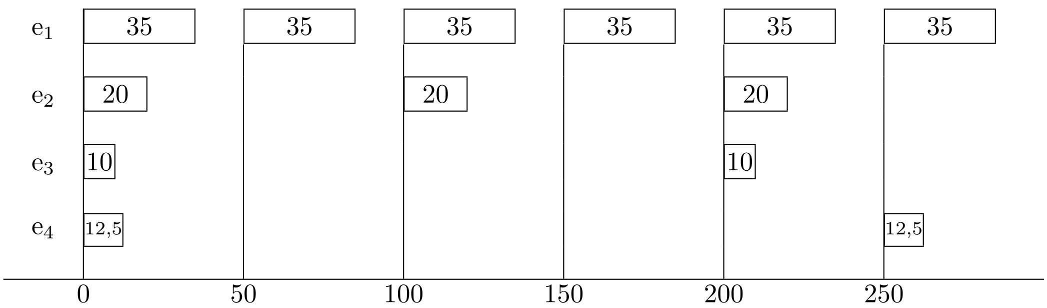
	e_1	e_2	e_3	e_4
surto	35	20	10	x
período	50	100	200	250

   Para que o sistema seja escalonável, suas frações da CPU não podem passar de 1, ou seja

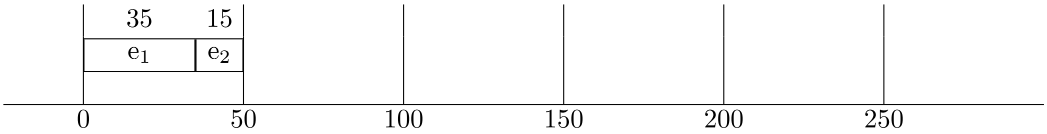
$$\frac{35}{50} + \frac{20}{100} + \frac{10}{200} + \frac{x}{250} \leq 1$$

$$\frac{190}{200} + \frac{x}{250} \leq 1 \Rightarrow \frac{x}{250} \leq \frac{10}{200} \Rightarrow x \leq 12,5$$

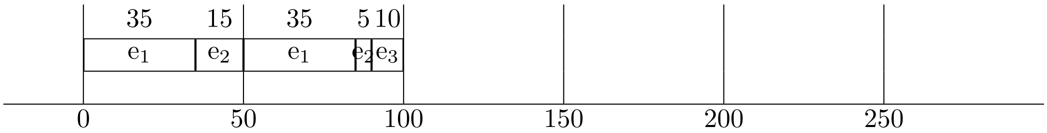
9. Requisições ao sistema aparecerão, ao longo do tempo, conforme abaixo:



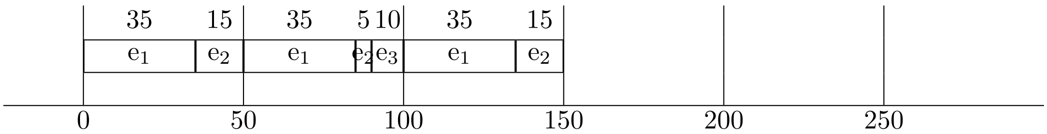
(a) RMS: por conta de suas frequências, as prioridades dos eventos serão e_1, e_2, e_3 e e_4. Inicialmente, todos competem, e_1 ganha, e o restante do tempo de CPU passa a e_2, que é interrompido por nova requisição de e_1:



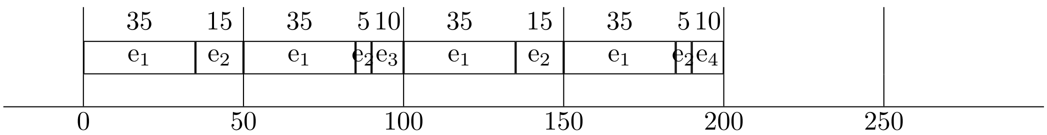
e_1 então roda. Ao finalizar, o restante de e_2 é finalizado. e_3 passa então a rodar (na fila ainda há e_4 não atendido).



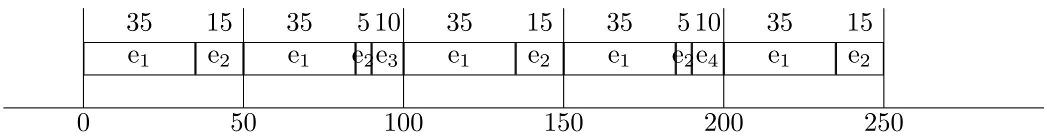
No instante 100, nova requisição de e_1 e e_2 surgem. e_1 roda, com o restante do tempo dado a e_2. Na fila ainda resta e_4



Em 150, surge nova requisição de e_1. e_2 é parado dando lugar a ele. No que resta do tempo, e_2 é finalizado, e e_4 começa a rodar.

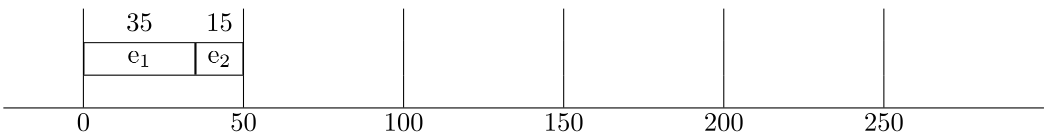


Em 200, novas requisições de e_1, e_2 e e_3. e_4 é interrompido (a 2,5 de seu fim) para dar lugar a e_1. Findado e_1, e_2 roda. A fila contém ainda e_4 e e_3.

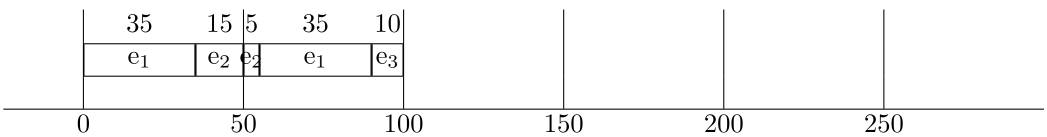


Em 250 nova requisição de e_1 e e_4 chegam. Na fila estão e_2 (parcial), e_3 e e_4 (parcial). Estourou o tempo para e_4.

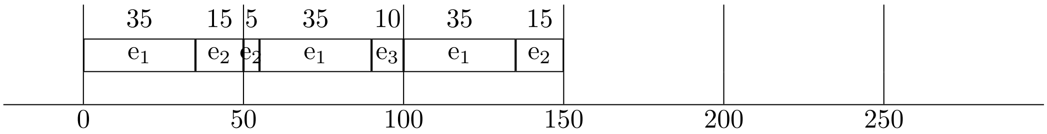
(b) Inicialmente, todos os eventos competem pela CPU. Ganha aquele cujo prazo de vencimento estiver mais próximo: e_1. Na fila ficam e_2, e_3 e e_4. Terminado e_1, o próximo a vencer é colocado para rodar: e_2.



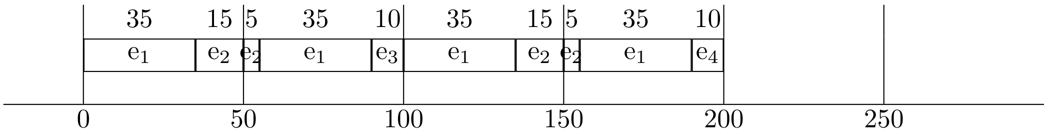
Em 50, surge um novo evento: e_1. Como seu vencimento é idêntico a e_2, este continua rodando. Após terminado, e_1 volta a rodar. Terminado, é a vez de e_3, por ter um prazo mais curto. A fila ainda contém e_4



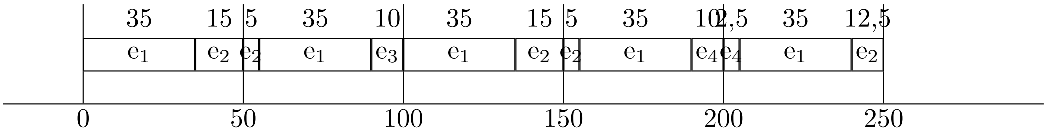
No instante 100, nova requisição de e_1 e e_2 surgem. e_1 roda, com o restante do tempo dado a e_2. Na fila ainda restam e_2 e e_4



Em 150, surge nova requisição de e_1. Por ter mesmo prazo de e_2, e_2 continua a rodar, sendo seguido por e_1. No que resta do tempo e_4 começa a rodar.



Em 200, surgem e_1, e_2 e e_3. Como e_1 e e_4 empatam, e_4 continua a rodar, seguido por e_1 e e_2:



etc...

10. Uso da CPU pelo áudio: $2 \times \frac{1}{5}$.

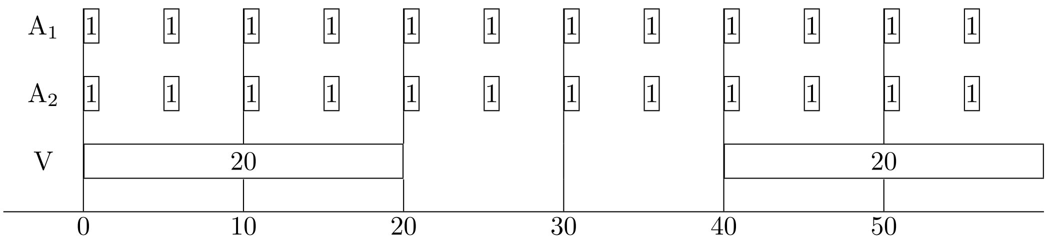
    Uso da CPU pelo vídeo: são 25 quadros a cada 1000ms, o que dá 1 quadro a cada 40ms $\Rightarrow \frac{1}{40}$. Como caa quadro usa 20ms, o uso da CPU é de $\frac{20}{40} = \frac{1}{2}$

    Uso total da CPU:

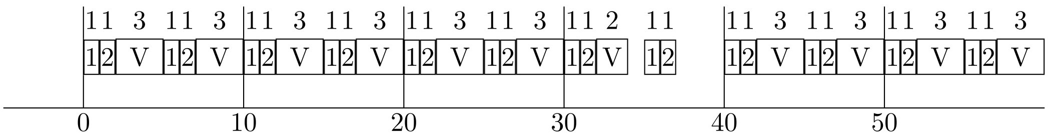
$$\frac{2}{5} + \frac{1}{2} = \frac{4+5}{10} = \frac{9}{10} \leq 1$$

    Portanto, é escalonável

11. Requisições ao sistema aparecerão, ao longo do tempo, conforme abaixo:



    (a) A_1 e A_2 têm a mesma prioridade. Qualquer um deles pode ser escolhido. Escolhamos A_1. Ao terminar, A_2 rodará. Ao fim deste, o único que sobra é V. Este rodará até que surjam novas requisições de A_1 e A_2 que, por terem maior prioridade, interrompem V. As trocas ficam, então:



    (b) A_1 e A_2 têm o mesmo prazo. Eles rodam. Findados, roda V. Quando Surgem novamente, o prazo de A_1 e A_2 continua sendo o mais curto. Portanto interrompem V. O resultado é idêntico ao do item (a).

12. (a) Supondo a fila P_1, P_2, P_3, P_4, P_5, temos:

P_1	10	=	10
P_2	14+10	=	24
P_3	5+14+10	=	29
P_4	7+5+14+10	=	36
P_5	20+7+5+14+10	=	56

$$\frac{10+24+29+36+56}{5} = \frac{155}{5} = 31$$

(b) No shortest job first, a fila fica P_3, P_4, P_1, P_2, P_5. Temos:

P_3	5	=	5
P_4	7+5	=	12
P_1	10+7+5	=	22
P_2	14+10+7+5	=	36
P_5	20+14+10+7+5	=	56

$$\frac{5+12+22+36+56}{5} = \frac{131}{5} = 26,2$$

(c) Com P_3, P_4, P_1, P_2, P_5, temos o mesmo tempo do item (b)

(d) Supondo a fila P_1, P_2, P_3, P_4, P_5, temos (com quantum de 2 u.t., e pressupondo que nada é gasto no escalonamento):

Situação					*Tempo*	*Obs.*
P_1	P_2	P_3	P_4	P_5	*Final*	
10	14	5	7	20	0	Situação inicial
8	12	3	5	18	10	Situação após 10 u.t.
6	10	1	3	16	20	Situação após 20 u.t.
4	8	0	1	14	29	Aos 25, P_3 termina
2	6		0	12	36	Aos 34, P_4 termina
0	4			10	42	Aos 38, P_1 termina
	2			8	46	Situação após 46 u.t.
	0			6	50	Aos 48, P_2 termina
				4	52	
				2	54	
				0	56	Aos 56, P_5 termina

$$\frac{25+34+38+48+56}{5} = \frac{201}{5} = 40,2$$

13. (a) Aos 8 u.t., estamos ainda no primeiro quantum, então P_1 = executando e $P_2 = P_3$ = Pronto.

    (b) Aos 11 u.t., o primeiro quantum terminou e o segundo se inicia, então P_2 = executando e $P_1 = P_3$ = Pronto.

    (c) Aos 33 u.t., processos podem ter terminado. Supondo que o escalonador consiga dar um novo quantum inteiro ao próximo processo (após outro processo terminar no meio de seu quantum), teremos:

Situação			*Tempo*	*Obs.*
P_1	P_2	P_3	*Final*	
18	4	13	0	Situação inicial
8	0	3	24	Às 14 u.t., P_2 termina
0		-	32	às 32 u.t. P_1 termina

    na 33, P_3 estará rodando (1 u.t. passada). Então P_3 = executando e $P_1 = P_2$ = Terminado.

14. (a) Com 8 u.t., P_1 roda 5 u.t. e bloqueia para E/S. Então P_2 roda 3 u.t. (fechando as 8). Assim, P_1 = bloqueado, P_2 = rodando e P_3 = pronto.

    (b) Com 18 u.t., P_1 roda 5 u.t. e bloqueia para E/S. Então P_2 roda até o fim (4 u.t.). As 9 u.t. restantes são rodadas de P_3 (dentro de seu quantum de 10 u.t.). Enquanto rodava seu quantum, contudo, venceu o prazo de 10 u.t. em que P_1 esperava pela E/S, o que fez com que ele desbloqueasse e ficasse pronto para rodar. Assim, P_1 = pronto, P_2 = terminado e P_3 = rodando.

(c) Com 28 u.t., temos o seguinte:

Situação			*Tempo*	*Obs.*
P_1	P_2	P_3	*Final*	
14	4	12	0	Situação inicial
9	-	-	5	P_1 faz uma E/S e bloqueia
9	0	-	9	P_2 roda até o fim
$\underline{9}$		2	19	P_3 roda. Em 15, P_1 é desbloqueado
4		-	24	P_1 faz uma E/S e bloqueia
$\underline{4}$		0	26	P_3 termina

aos 28 P_1 ainda não desbloqueou, e os demais terminaram. Então P_1 = bloqueado, P_2 = P_3 = terminado.

15. Na mesma ordem do shortest job first: P_3, P_4, P_2 e P_1.

16. (a) Supondo a ordem P_1, P_2, P_3 e P_4:

Troca = 0:

$$\begin{array}{lrcl} P_1 & 40 & = & 40 \\ P_2 & 20+40 & = & 60 \\ P_3 & 50+20+40 & = & 110 \\ P_4 & 30+50+20+40 & = & 140 \end{array}$$

$$\frac{40+60+110+140}{4} = \frac{350}{4} = 87,5$$

Troca = 5 u.t.

$$\begin{array}{lrcl} P_1 & 40 & = & 40 \\ P_2 & 20+5+40 & = & 65 \\ P_3 & 50+5+20+5+40 & = & 120 \\ P_4 & 30+5+50+5+20+5+40 & = & 155 \end{array}$$

$$\frac{40+65+120+155}{4} = \frac{380}{4} = 95$$

(b) A ordem fica P_2, P_4, P_1, P_3.

Troca = 0:

$$\begin{array}{lrcl} P_2 & 20 & = & 20 \\ P_4 & 30+20 & = & 50 \\ P_1 & 40+30+20 & = & 90 \\ P_3 & 50+40+30+20 & = & 140 \end{array}$$

$$\frac{20+50+90+140}{4} = \frac{300}{4} = 75$$

Troca = 5:

$$\begin{array}{lrcl} P_2 & 20 & = & 20 \\ P_4 & 30+5+20 & = & 55 \\ P_1 & 40+5+30+5+20 & = & 100 \\ P_3 & 50+5+40+5+30+5+20 & = & 155 \end{array}$$

$$\frac{20+55+100+155}{4} = \frac{330}{4} = 82,5$$

(c) Com quantum = 20 u.t., e supondo a ordem P_1, P_2, P_3 e P_4:

Troca = 0:

Situação				*Tempo*	*Obs.*
P_1	P_2	P_3	P_4	*Final*	
40	20	50	30	0	Situação inicial
20	0	-	-	40	P_2 termina
-		30	10	80	
0		-	-	100	P_1 termina
		10	0	130	P_4 termina
		0		140	P_3 termina

$$\frac{40+100+130+140}{4} = \frac{410}{4} = 102,5$$

Troca = 5:

Situação				*Tempo*	*Obs.*
P_1	P_2	P_3	P_4	*Final*	
40	20	50	30	0	Situação inicial
20	0	-	-	45	P_2 termina
-		30	10	95	
0		-	-	120	P_1 termina
		10	0	160	P_4 termina
		0		175	P_3 termina

$$\frac{45+120+160+175}{4} = \frac{500}{4} = 125$$

17. Supondo a distribuição de bilhetes na forma consecutiva, teremos que P_1 recebe os bilhetes de 1–4, P_2 de 5–6, P_3 o 7 e P_4 de 8–10. Com a sequência de sordeios dada, os processos escalonados são P_4, P_2, P_1, P_4, P_1, P_1, P_2, P_4, P_1 e P_3.

18. (a) Basta rodar um processo de cada usuário: A_1, B_1, A_2, B_2, A_3, B_1, A_4, B_2, A_5, B_1

    (b) Basta rodar 2 processos de A e 1 de B: A_1, A_2, B_1, A_3, A_4, B_2, A_5, A_1, B_1

    (c) Basta rodar 3 processos de B e 1 de A: B_1, B_2, B_1, A_1, B_2, B_1, B_2, A_2, ...

19.

A			B			*Tempo*	*Obs.*
P_1	P_2	P_3	P_4	P_5	P_6	*Final*	
6	5	7	3	8	4	0	Situação inicial. Começará em A
4	3	5	-	-	-	6	Somente rodam em A
3	-	-	-	-	-	7	Em 7, troca de fila
-	-	-	1	6	-	11	Em 11 troca de fila. Recomeça em P_1 (que tinha sido interrompido)
1	1	3	-	-	-	17	
0	-	-	-	-	-	18	Em 18, P_1 termina, e troca de fila
	-	-	-	-	2	20	
	-	-	0	-	-	21	Em 21 P_4 termina
	-	-		5	-	22	Em 22 troca de fila
	0	-		-	-	23	Em 23 P_2 termina
		1		-	-	25	
		0		-	-	26	Em 26, P_3 termina. Não há mais nada em A
				3	0	30	Em 30, P_6 termina
				1		32	
				0		33	Em 33, P_5 termina

20. Algoritmos por prioridades em geral são piores que alternância circular quando as prioridades são idênticas. Isso porque gasta-se tempo atualizando-se as prioridades e mantendo a fila ordenada. Por outro lado, a alternância circular é incapaz de tratar prioridades, a menos que se incluam clones de cada processo na fila de prontos, em número correspondente à sua

prioridade, o que complica consideravelmente o gerenciamento do sistema. Então, para uma comparação mais justa, ignoraremos o tempo gasto com essas operações (decremento dos créditos e interrupção para análise da prioridade).

Round robin (quantum = 3):

Processos				*Trocas de*	*Tempo*	*Obs.*
A	*B*	*C*	*D*	*Contexto*	*Final*	
6	6	14	10	0	0	Situação inicial. Começará em A
4	-	-	-	1	2	Em 2, A bloqueia
B	3	-	-	2	5	Em 5, B bloqueia
-	B	11	7	4	11	Em 7, A desbloqueia
2	B	-	-	5	13	Em 13, A bloqueia
-	-	8	4	7	19	Em 15, B desbloqueou. Em 18, A desbloqueou
0	-	-	-	8	21	Em 21, A termina
	0	-	-	9	24	Em 24, B termina
		5	1	11	30	
		2	0	13	34	Em 34, D termina
		0		14	36	Em 36, C termina

Tempo médio de resposta: $\frac{21+24+34+36}{4} = \frac{115}{4} = 28,75$

Prioridades (entre parênteses está o número de créditos restantes):

Processos				*Trocas de*	*Tempo*	*Obs.*
A	*B*	*C*	*D*	*Contexto*	*Final*	
6(3)	6(3)	14(3)	10(3)	0	0	Situação inicial. Começará em A
4(1)	-	-	-	1	2	Em 2, A bloqueia
B	3(0)	-	-	2	5	Em 5, B bloqueia
-	B	11(0)	7(0)	4	11	Em 7, A desbloqueia
3(0)	B(0)	11(0)	7(0)	5	12	Acabaram os créditos
3(3)	B(3)	11(3)	7(3)	5	12	Redistribuição de créditos
2(2)	-	-	-	5	12	Em 13, A bloqueia
-	-	8(0)	4(0)	7	19	Em 15, B desbloqueou. Em 18, A desbloqueou
0(0)	-	-	-	8	21	Em 21, A termina
	0(0)	-	-	9	24	Em 24, B termina
		8(3)	4(3)	9	24	Redistribuição de créditos
		5(0)	1(0)	11	30	
		5(3)	1(3)	11	30	Redistribuição de créditos
		2(0)	0(0)	13	34	Em 34, D termina
		2(3)		13	34	Redistribuição de créditos
		0(1)		14	36	Em 36, C termina

E o tempo médio de retorno deu idêntico. Contudo, apenas porque ignoramos o tempo gasto no gerenciamento. A distribuição de créditos certamente deixaria o algoritmo mais lento, especialmente em um exemplo em que o quantum casa com a prioridade de cada processo.