UNIVERSIDADE DE ÉVORA

Trabalho 1 – Simulador de SO com Modelo de 5 Estados

Ricardo Matono (44448), João Rouxinol (44451) e André Rato (45517)

Ano Letivo 2020/2021

Sistemas Operativos 1
Docente: Luís Rato

Índice

Introdução

É pretendida a implementação, utilizando a linguagem C, de um simulador de SO considerando um modelo de 5 estados (NEW, READY, RUNNING, BLOCKED e EXIT), utilizando dois algoritmos de escalonamento (RR – Round Robin Standard – e VRR – Virtual Round Robin).

Os processos são definidos por um conjunto de pseudo-instruções-máquina representadas por número, que representam alternadamente:

- o tempo que o processo fica no CPU (estado RUNNING);
- o tempo que o processo gasta no estado BLOCKED após chegar ao 1º lugar da fila.

Cada processo é iniciado no instante indicado na primeira posição do array correspondente ao mesmo, ou seja, o instante inicial de um processo é o valor em programas[i][0].

Para que um processo seja dado como terminado, é necessário que, no instante após o último valor diferente de zero, o processo termina.

Características do Simulador

A implementação do simulador de processos deve ter em consideração as seguintes características:

1. a mudança entre estados é infinitamente rápida (os processos ficam nos diversos estados 1 instante de tempo, seguindo-se outra mudança de estado, e assim sucessivamente);
2. quando um processo é criado, o estado do mesmo muda para NEW e permanece nesse estado durante 1 instante de tempo;
3. quando um processo sai do estado NEW muda para READY; se a fila READY estiver vazia e o CPU (estado RUNNING) não estiver ocupado por um processo, pode mudar diretamente para RUNNING;
4. os instantes da fila BLOCKED, para cada processo, só contam se este estiver no primeiro lugar da fila;
5. quando um processo termina, passa do CPU (estado RUNNING) para o estado EXIT, onde permanece 1 instante de tempo;
6. após 1 instante de tempo em EXIT, os processos desaparecem do sistema;
7. o escalonamento a implementar deve compreender 2 algoritmos, RR e VRR, ambos com Quantum = 3;
8. os processos que saem de BLOCKED, passam para READY, no caso de RR, e para READY-AUX, no caso de VRR;
9. o número máximo de programas a dar entrada no sistema é de 10, e cada programa tem a dimensão máxima de 10 (1 instante inicial e 9 instruções alternadas de CPU e BLOCK);
10. se no mesmo instante puderem entrar processos na fila de READY vindos de RUNNING, BLOCKED e NEW, o primeiro a entrar na fila é o processo da fila de BLOCKED, o segundo o processo que estava no CPU (estado RUNNING) e por último o processo NEW;

11. o programa deve ter como output o estado de cada processo em cada instante.

Implementação do Simulador

- ## Estruturas

É utilizada uma estrutura de nome ***program***, utilizada para guardar a informação correspondente a cada programa a ser executado. Tem como componentes:

- int now: processo na qual o programa está a executar;
- int state: estado na qual o programa se encontra;
- int start: instante em que o programa começa a ser iniciado;
- int cycle[11]: conjunto de processos do programa.

- ## Estados

Os estados são representados por inteiros, tais que:

0 » Exit;
1 » Ready;
2 » Run;
3 » Block;
4 » Processo não criado;
5 » New;
6 » Processo terminado;
7 » Auxiliar (uso exclusivo do algoritmo VRR).

- ## Filas

As filas são representadas por inteiros, tais que:

0 » Ready queue;
1 » Block queue;
2 » Auxiliar queue (uso exclusivo do algoritmo VRR).

Os métodos responsáveis por realizar operações nas filas são os mesmos nos três casos: o método ***front*** retorna o id do programa que está na frente da fila que foi passada como argumento, sem o retirar da mesma; o método ***enqueue*** tem como função colocar o id de um programa no final da fila desejada; por fim, mas não menos importante, o método ***dequeue*** é responsável por retirar o elemento mais antigo da fila, ou seja, aquele que está em primeiro lugar.

- ## Mudanças de Estado

As mudanças de estado são efetuadas na função ***state_change***, que recebe como argumentos o id do programa ***(program_id***) e uma variável que indica se o algoritmo escolhido é RR ou VRR (***is_vrr***). Todas as transições são feitas de acordo com as características do simulador e do algoritmo escolhido.

- ## Time Quantum

O time quantum e as suas mudanças de estado são geridos pela função ***quantum_change***, que recebe como argumentos o id do programa (***program_id***) e uma

variável que indica se o algoritmo escolhido é RR ou VRR (***is_vrr***). O quantum limita o número de ciclos que cada processo pode estar no CPU continuamente. Após atingido o limite, o processo volta para a READY queue, entrando um novo processo para o CPU e o quantum time é restabelecido.

- **Criação de Programas**

Utilizando a função ***create***, os programas são criados tendo em conta o array bidimensional declarado na ***main***, o time quantum recebido como argumento na função e o algoritmo escolhido (***is_vrr***). É a partir daqui que o programa começa a ser executado, ou seja, é aqui que são tomadas as decisões iniciais relacionadas com o início dos programas.

- **Execução**

A execução dos programas e correspondentes processos é feita utilizando a função ***run***. Esta função recebe como argumento o time quantum e uma variável que indica qual o algoritmo de escalonamento escolhido (***is_vrr***). É nesta função que se encontram todas as condições para cada mudança de estado poder ocorrer.

- **Main**

É na função ***main*** que o array bidimensional de programas é definido, assim como o time quantum e o algoritmo de escalonamento. Para a escolha do algoritmo de escalonamento e do valor de time quantum, faz-se o seguinte:

- para o algoritmo de escalonamento, alterar o valor da variável ***run_mode*** para VRR, se o método pretendido for Virtual Round Robin, ou para RR, caso o algoritmo pretendido for Round Robin Standard;

- para o time quantum, basta alterar o valor da variável ***quantum_time*** para o valor pretendido.

Exemplo de Intput/Output

- Input:

```
int programas[5][10] = {
     {0, 3, 1, 2, 2, 4, 1, 1, 1, 1 } ,
     {1, 2, 4, 2, 4, 2, 0, 0, 0, 0 } ,
     {3, 1, 6, 1, 6, 1, 6, 1, 0, 0 } ,
     {3, 6, 1, 6, 1, 6, 1, 6, 0, 0 } ,
     {5, 9, 1, 9, 0, 0, 0, 0, 0, 0 }
};
```

- Output:

RR

Inst	P01	P02	P03	P04	P05
1	NEW				
2	RUN	NEW			
3	RUN	RDY			
4	RUN	RDY	NEW	NEW	
5	BLK	RUN	RDY	RDY	
6	RDY	RUN	RDY	RDY	NEW
7	RDY	BLK	RUN	RDY	RDY
8	RDY	BLK	BLK	RUN	RDY
9	RDY	BLK	BLK	RUN	RDY
10	RDY	BLK	BLK	RUN	RDY
11	RUN	RDY	BLK	RDY	RDY
12	RUN	RDY	BLK	RDY	RDY
13	BLK	RDY	BLK	RDY	RUN
14	BLK	RDY	BLK	RDY	RUN
15	BLK	RDY	BLK	RDY	RUN
16	BLK	RUN	BLK	RDY	RDY
17	BLK	RUN	RDY	RDY	RDY
18	BLK	BLK	RDY	RUN	RDY
19	RDY	BLK	RDY	RUN	RDY
20	RDY	BLK	RDY	RUN	RDY
21	RDY	BLK	RDY	BLK	RUN
22	RDY	BLK	RDY	BLK	RUN
23	RDY	RDY	RDY	BLK	RUN
24	RDY	RDY	RUN	RDY	RDY
25	RUN	RDY	BLK	RDY	RDY
26	RUN	RDY	BLK	RDY	RDY
27	RUN	RDY	BLK	RDY	RDY
28	RDY	RUN	BLK	RDY	RDY
29	RDY	RUN	BLK	RDY	RDY
30	RDY	EXT	BLK	RUN	RDY
31	RDY	---	RDY	RUN	RDY
32	RDY	---	RDY	RUN	RDY
33	RDY	---	RDY	RDY	RUN
34	RDY	---	RDY	RDY	RUN
35	RDY	---	RDY	RDY	RUN
36	RUN	---	RDY	RDY	BLK
37	BLK	---	RUN	RDY	RDY
38	RDY	---	BLK	RUN	RDY
39	RDY	---	BLK	RUN	RDY
40	RDY	---	BLK	RUN	RDY
41	RDY	---	BLK	BLK	RUN
42	RDY	---	BLK	BLK	RUN
43	RDY	---	BLK	BLK	RUN
44	RUN	---	RDY	BLK	RDY
45	BLK	---	RUN	RDY	RDY
46	RDY	---	EXT	RDY	RUN
47	RDY	---	---	RDY	RUN
48	RDY	---	---	RDY	RUN
49	RDY	---	---	RUN	RDY
50	RDY	---	---	RUN	RDY
51	RDY	---	---	RUN	RDY
52	RUN	---	---	RDY	RDY
53	EXT	---	---	RDY	RUN
54	---	---	---	RDY	RUN
55	---	---	---	RDY	RUN
56	---	---	---	RUN	EXT
57	---	---	---	RUN	---
58	---	---	---	RUN	---
59	---	---	---	BLK	---
60	---	---	---	RUN	---
61	---	---	---	RUN	---
62	---	---	---	RUN	---
63	---	---	---	RUN	---
64	---	---	---	RUN	---
65	---	---	---	RUN	---
66	---	---	---	EXT	---
67	---	---	---	---	---

VRR

Inst	P01	P02	P03	P04	P05
1	NEW				
2	RUN	NEW			
3	RUN	RDY			
4	RUN	RDY	NEW	NEW	
5	BLK	RUN	RDY	RDY	
6	AUX	RUN	RDY	RDY	NEW
7	RUN	BLK	RDY	RDY	RDY
8	RUN	BLK	RDY	RDY	RDY
9	BLK	BLK	RUN	RDY	RDY
10	BLK	BLK	BLK	RUN	RDY
11	BLK	AUX	BLK	RUN	RDY
12	BLK	AUX	BLK	RUN	RDY
13	AUX	RUN	BLK	RDY	RDY
14	AUX	RUN	BLK	RDY	RDY
15	RUN	BLK	BLK	RDY	RDY
16	RUN	BLK	BLK	RDY	RDY
17	RUN	BLK	BLK	RDY	RDY
18	RDY	BLK	BLK	RDY	RUN
19	RDY	BLK	AUX	RDY	RUN
20	RDY	BLK	AUX	RDY	RUN
21	RDY	BLK	RUN	RDY	RDY
22	RDY	BLK	BLK	RUN	RDY
23	RDY	AUX	BLK	RUN	RDY
24	RDY	AUX	BLK	RUN	RDY
25	RDY	RUN	BLK	BLK	RDY
26	RDY	RUN	BLK	BLK	RDY
27	RUN	EXT	BLK	BLK	RDY
28	BLK	---	BLK	BLK	RUN
29	BLK	---	AUX	BLK	RUN
30	BLK	---	AUX	AUX	RUN
31	AUX	---	RUN	AUX	RDY
32	AUX	---	BLK	RUN	RDY
33	AUX	---	BLK	RUN	RDY
34	AUX	---	BLK	RUN	RDY
35	RUN	---	BLK	RDY	RDY
36	BLK	---	BLK	RDY	RUN
37	BLK	---	BLK	RDY	RUN
38	BLK	---	AUX	RDY	RUN
39	AUX	---	RUN	RDY	BLK
40	RUN	---	EXT	RDY	AUX
41	EXT	---	---	RDY	RUN
42	---	---	---	RDY	RUN
43	---	---	---	RDY	RUN
44	---	---	---	RUN	RDY
45	---	---	---	RUN	RDY
46	---	---	---	RUN	RDY
47	---	---	---	BLK	RUN
48	---	---	---	AUX	RUN
49	---	---	---	AUX	RUN
50	---	---	---	RUN	RDY
51	---	---	---	RUN	RDY
52	---	---	---	RUN	RDY
53	---	---	---	RDY	RUN
54	---	---	---	RDY	RUN
55	---	---	---	RDY	RUN
56	---	---	---	RUN	EXT
57	---	---	---	RUN	---
58	---	---	---	RUN	---
59	---	---	---	BLK	---
60	---	---	---	RUN	---
61	---	---	---	RUN	---
62	---	---	---	RUN	---
63	---	---	---	RUN	---
64	---	---	---	RUN	---
65	---	---	---	RUN	---
66	---	---	---	EXT	---
67	---	---	---	---	---