GSI018 – SISTEMAS OPERACIONAIS

Operating Systems – William Stallings – 7th Edition
Chapter 06 – Deadlock and Starvation

Pedro Henrique Silva Santana – 12011BSI218 – pedro.santana@ufu.br
Victor Hugo Martins Alves – 12011BSI217 – victor.alves1@ufu.br

REVIEW QUESTIONS

6.2 What are the three conditions that must be present for deadlock to be possible?

As três condições para que seja possível o deadlock são:

- Exclusão Mútua;
- Manter e Esperar;
- Nenhuma Preempção.

6.3 What are the four conditions that create deadlock?

As quatro condições para criar um deadlock são:

- Exclusão Mútua;
- Manter e Esperar;
- Nenhuma Preempção;
- Espera Circular.

6.4 How can the hold-and-wait condition be prevented?

A condição de Manter e Esperar pode der prevenida ao exigir que uma solicitação de processo reúna todos os recursos de uma única vez, assim bloqueando o processo até que as condições sejam concedidas ao mesmo tempo.

6.5 List two ways in which the no-preemption condition can be prevented.

- Caso um processo que possua certos recursos tiver uma solicitação adicional negada, esse processo deve liberar seus recursos originais e, se necessário, solicitá-los novamente em conjunto com o recurso adicional.
- Se um processo solicita um recurso atualmente detido por outro processo, o sistema operacional pode antecipar o segundo processo e exigir que ele libere seus recursos.

PROBLEMS

6.2 Show how each of the techniques of prevention, avoidance, and detection can be applied to Figure 6.1.

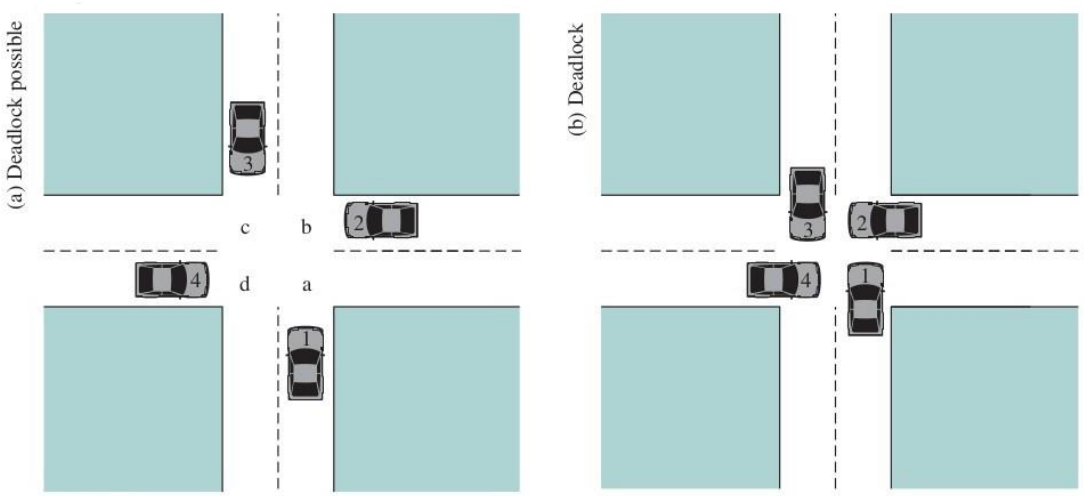

Figure 6.1 Illustration of Deadlock

Prevenção

Indireto - Exclusão mútua dos processos;

Indireto - Exigir os processos requisitassem todos os recursos ao mesmo tempo;

Direto - Ordenação dos tipos de recursos disponíveis.

Evitação

Recusa de início de um próximo processo para evitar o deadlock caso este possua exigências que levem ao tal.

Detecção

Abandono dos processos e recupera para um estado anterior;

Abandono sucessivo com base na ordem dos critérios de cada;

Preempção sucessiva até que o deadlock não exista.

6.5 Given the following state for the Banker's Algorithm.

i) 6 processes P0 through P5;

ii) 4 resource types: A (15instances); B (6 instances);

iii) C (9 instances); D (10 instances);

Resource V

A	B	C	D
15	6	9	10

iv) Snapshot at time T0:

Available

A	B	C	D
6	3	5	4

Current allocation

	A	B	C	D
P0	2	0	2	1
P1	0	1	1	1
P2	4	1	0	2
P3	1	0	0	1
P4	1	1	0	0
P5	1	0	1	1

Maximum demand

	A	B	C	D
P0	9	5	5	5
P1	2	2	3	3
P2	7	5	4	4
P3	3	3	3	2
P4	5	2	2	1
P5	4	4	4	4

a. Verify that the Available array has been calculated correctly.

Alocados

A	B	C	D
9	3	4	6

Available = Resource – Aloc

Available

A	B	C	D
6	3	5	4

Vetor Available está correto.

b. Calculate the Need matrix.

	A	B	C	D
P0	7	5	3	4
P1	2	1	2	2
P2	3	4	4	2
P3	2	3	3	1
P4	4	1	2	1
P5	3	4	3	3

c. Show that the current state is safe, that is, show a safe sequence of processes. In addition, to the sequence show how the Available (working array) changes as each process terminates.

Até o processo P2 ocorreria sem problemas pois o valor alocado ainda é menor que o disponível. Após isso, os demais processos não teriam dificuldade para execução pois seria disponibilizado recursos dos processos anteriores, garantindo ainda sim uma diferença entre o alocado e o disponível.

d. Given the request (3,2,3,3) from Process P5. Should this request be granted? Why or why not?

Alocados

A	B	C	D
11	5	6	8

Available

A	B	C	D
4	1	3	2

Sim, essa requisição pode ser garantida pois ainda teria disponível recursos.