

## 作业八

姓名: 刘涵之 学号: 519021910102

Practice Exercise: 8.3, 8.9, 8.18, 8.27, 8.28

Consider the following snapshot of a system:

		<u>Allocation</u>	<u>Max</u>	<u>Available</u>
		<i>A B C D</i>	<i>A B C D</i>	<i>A B C D</i>
8.3	$T_0$	0 0 1 2	0 0 1 2	1 5 2 0
	$T_1$	1 0 0 0	1 7 5 0	
	$T_2$	1 3 5 4	2 3 5 6	
	$T_3$	0 6 3 2	0 6 5 2	
	$T_4$	0 0 1 4	0 6 5 6	

Answer the following questions using the banker's algorithm:

- What is the content of the matrix *Need*?
- Is the system in a safe state?
- If a request from thread  $T_1$  arrives for (0,4,2,0), can the request be granted immediately?

a. Need: A B C D

$T_0$  0 0 0 0

$T_1$  0 7 5 0

$T_2$  1 0 0 2

$T_3$  0 0 2 0

$T_4$  0 6 4 2

b. Yes. 当前状态下T3可以执行, 执行后释放资源, 然后就可以让任意其他线程执行。

c. Yes. 当前可用资源为 (1 5 2 0), 分配掉(0 4 2 0)后还有(1 1 0 0), 然后可以以T0 -> T2 -> T3 -> T4 -> T1的顺序执行

Consider the following snapshot of a system:

	<u>Allocation</u>				<u>Max</u>			
	A	B	C	D	A	B	C	D
$T_0$	3	0	1	4	5	1	1	7
$T_1$	2	2	1	0	3	2	1	1
$T_2$	3	1	2	1	3	3	2	1
$T_3$	0	5	1	0	4	6	1	2
$T_4$	4	2	1	2	6	3	2	5

8.9

Using the banker's algorithm, determine whether or not each of the following states is unsafe. If the state is safe, illustrate the order in which the threads may complete. Otherwise, illustrate why the state is unsafe.

- $Available = (0, 3, 0, 1)$
- $Available = (1, 0, 0, 2)$

Need: A B C D

$T_0$  2 1 0 3

$T_1$  1 0 0 1

$T_2$  0 2 0 0

$T_3$  4 1 0 2

$T_4$  2 1 1 3

- 不安全  $T_2 \rightarrow T_1 \rightarrow T_3$ , 然后因为D资源不足, 剩下的线程无法执行
- 安全  $T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_0 \rightarrow T_4$

8.18

Which of the six resource-allocation graphs shown in Figure 8.12 illustrate deadlock? For those situations that are deadlocked, provide the cycle of threads and resources. Where there is not a deadlock situation, illustrate the order in which the threads may complete execution.

- (a) 非死锁,  $T_2 \rightarrow T_3 \rightarrow T_1$
- (b) 死锁,  $\rightarrow T_1 \rightarrow R_3 \rightarrow T_3 \rightarrow R_1 \rightarrow$
- (c) 非死锁,  $T_2 \rightarrow T_3 \rightarrow T_1$
- (d) 死锁,  $\rightarrow T_1, T_2 \rightarrow R_2 \rightarrow T_3, T_4 \rightarrow R_1 \rightarrow$
- (e) 非死锁,  $T_2 \rightarrow T_1 \rightarrow T_3 \rightarrow T_4$
- (f) 非死锁,  $T_2 \rightarrow T_4 \rightarrow T_1 \rightarrow T_3$

8.27

Consider the following snapshot of a system:

	<u>Allocation</u>	<u>Max</u>
	<u>A B C D</u>	<u>A B C D</u>
$T_0$	1 2 0 2	4 3 1 6
$T_1$	0 1 1 2	2 4 2 4
$T_2$	1 2 4 0	3 6 5 1
$T_3$	1 2 0 1	2 6 2 3
$T_4$	1 0 0 1	3 1 1 2

Using the banker's algorithm, determine whether or not each of the following states is unsafe. If the state is safe, illustrate the order in which the threads may complete. Otherwise, illustrate why the state is unsafe.

- $Available = (2, 2, 2, 3)$
- $Available = (4, 4, 1, 1)$
- $Available = (3, 0, 1, 4)$
- $Available = (1, 5, 2, 2)$

Need: A B C D

$T_0$  3 1 1 4

$T_1$  2 3 1 2

$T_2$  2 4 1 1

$T_3$  1 4 2 2

$T_4$  2 1 1 1

- 安全  $T_4 \rightarrow T_0 \rightarrow T_1 \rightarrow T_2 \rightarrow T_3$
- 安全  $T_4 \rightarrow T_1 \rightarrow T_0 \rightarrow T_2 \rightarrow T_3$
- 不安全, 所有线程都由于没有B资源而无法执行
- 安全,  $T_3 \rightarrow T_4 \rightarrow T_1 \rightarrow T_2 \rightarrow T_0$

8.28

Consider the following snapshot of a system:

	<u>Allocation</u>	<u>Max</u>	<u>Available</u>
	<i>A B C D</i>	<i>A B C D</i>	<i>A B C D</i>
$T_0$	3 1 4 1	6 4 7 3	2 2 2 4
$T_1$	2 1 0 2	4 2 3 2	
$T_2$	2 4 1 3	2 5 3 3	
$T_3$	4 1 1 0	6 3 3 2	
$T_4$	2 2 2 1	5 6 7 5	

Answer the following questions using the banker's algorithm:

- Illustrate that the system is in a safe state by demonstrating an order in which the threads may complete.
- If a request from thread  $T_4$  arrives for (2, 2, 2, 4), can the request be granted immediately?
- If a request from thread  $T_2$  arrives for (0, 1, 1, 0), can the request be granted immediately?
- If a request from thread  $T_3$  arrives for (2, 2, 1, 2), can the request be granted immediately?

Need: A B C D

$T_0$  3 3 3 2

$T_1$  2 1 3 0

$T_2$  0 1 2 0

$T_3$  2 2 2 2

$T_4$  3 4 5 4

- $T_2 \rightarrow T_3 \rightarrow T_0 \rightarrow T_1 \rightarrow T_4$
- 不能，分配完后系统资源耗尽，没有线程可以执行，陷入死锁
- 不能，C资源不够，陷入死锁
- 不能，BC资源不够，陷入死锁