Chapter 8 Homework

陈文迪 519021910071

作业中的引用内容均已标出

8.3 考虑下列一个系统某一时刻的快照。利用银行家算法回答下列问题。

	Allocation	Max	Available
	A B C D	A B C D	A B C D
T0	0 0 1 2	0 0 1 2	1 5 2 0
T1	1 0 0 0	1 7 5 0	
T2	1 3 5 4	2 3 5 6	
T3	0 6 3 2	0 6 5 2	
T4	0 0 1 4	0 6 5 6	

1. **Need** 矩阵的内容是什么?
2. 该系统是否处于安全状态?
3. 如果T1有一个请求（0，4，2，0），该请求能否立即被满足?

问题解答:

1. **Need** 矩阵的内容如下。

	Need
	A B C D
T0	0 0 0 0
T1	0 7 5 0
T2	1 0 0 2
T3	0 0 2 0
T4	0 6 4 2

2. 该系统处于安全状态。可以把目前可用的资源全部分配给T3，此后T3释放出所用分配给它的资源，可用资源变为（1，11，5，2），此后所有剩下的线程只需依次运行即可。一个可行的顺序是：T3，T0，T1，T2，T4。
3. 可以被立刻满足。重新分配后的系统快照如下。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	0 0 1 2	0 0 1 2	0 0 0 0	1 1 0 0
T1	1 4 2 0	1 7 5 0	0 3 3 0	
T2	1 3 5 4	2 3 5 6	1 0 0 2	
T3	0 6 3 2	0 6 5 2	0 0 2 0	
T4	0 0 1 4	0 6 5 6	0 6 4 2	

系统依然处于安全状态。我们可以按照这样的顺序执行：T0，T2，T1，T3，T4。

8.9 考虑下列一个系统某一时刻的快照。使用银行家算法判断下列状态是否安全。如果安全，则展示一个线程执行的可行顺序；否则，则展示为什么该状态是不安全的。

	Allocation	Max
	A B C D	A B C D
T0	3 0 1 4	5 1 1 7
T1	2 2 1 0	3 2 1 1
T2	3 1 2 1	3 3 2 1
T3	0 5 1 0	4 6 1 2
T4	4 2 1 2	6 3 2 5

1. **Available** = (0，3，0，1)
2. **Available** = (1，0，0，2)

问题解答：

1. 我们先将 **Need** 矩阵计算出来。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 0 1 4	5 1 1 7	2 1 0 3	0 3 0 1
T1	2 2 1 0	3 2 1 1	1 0 0 1	
T2	3 1 2 1	3 3 2 1	0 2 0 0	
T3	0 5 1 0	4 6 1 2	4 1 0 2	
T4	4 2 1 2	6 3 2 5	2 1 1 3	

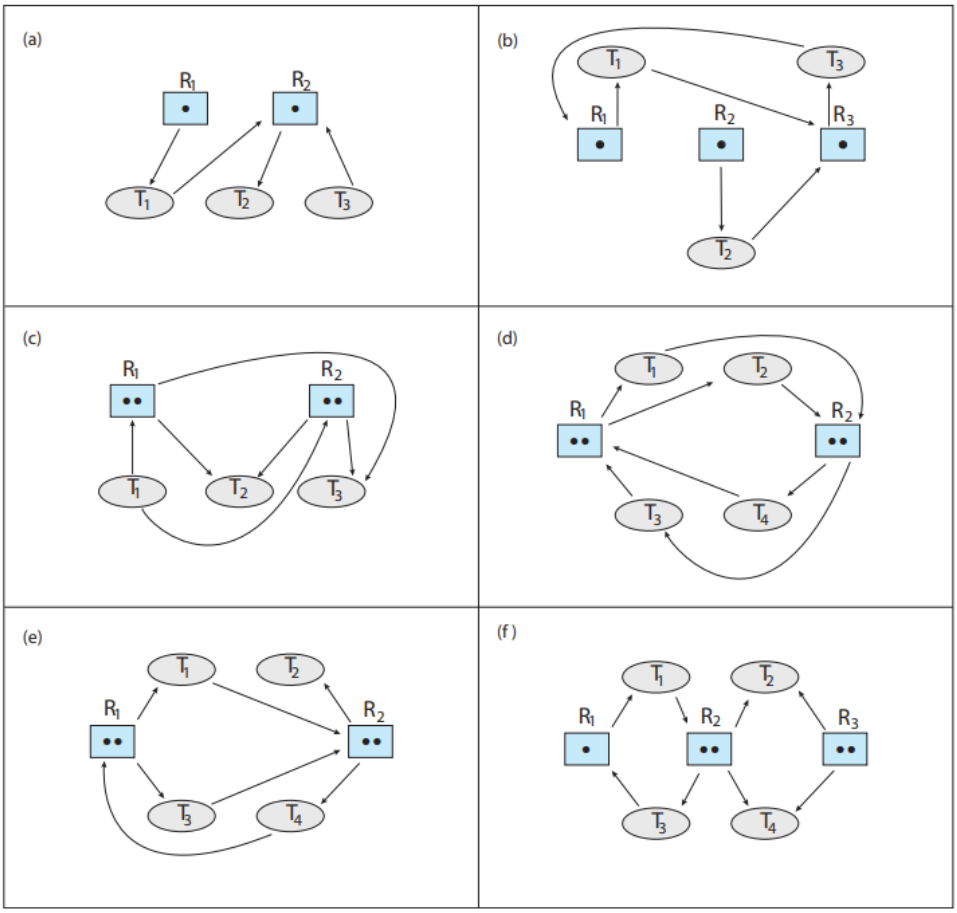
该状态是不安全的。当我们按照可用资源依次执行完T2，T1，T3之后D资源只用两个实例可用，此时T0和T4都无法完成。

2. 此时的系统快照如下。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 0 1 4	5 1 1 7	2 1 0 3	1 0 0 2
T1	2 2 1 0	3 2 1 1	1 0 0 1	
T2	3 1 2 1	3 3 2 1	0 2 0 0	
T3	0 5 1 0	4 6 1 2	4 1 0 2	
T4	4 2 1 2	6 3 2 5	2 1 1 3	

该状态是安全的。我们可以按照如下顺序执行完所有线程：T1，T2，T0，T3，T4。

8.18 下列六幅资源分配图中哪几幅存在死锁。对于死锁的情形，请指出线程和资源的回路；对于非死锁的情形，请指出在什么顺序下所有线程可以完成执行。



问题解答：

（a）图不存在死锁，可以按以下顺序执行完：T2，T3，T1。

（b）图存在死锁，因为有如下回路：R1->T1->R3->T3->R1。

（c）图不存在死锁，可以按以下顺序执行完：T2，T3，T1。

（d）图存在死锁，图中有两个回路：R1->T1->R2->T3->R1和R1->T2->R2->T4->R1。或者也可以看成R1->T1->R2->T4->R1和R1->T2->R2->T3->R1。

（e）图不存在死锁，可以按以下顺序执行完：T2，T1，T3，T4。

（f）图不存在死锁，可以按以下顺序执行完：T4，T2，T1，T3。

8.27 考虑下列一个系统某一时刻的快照。使用银行家算法判断下列状态是否安全。如果安全，则展示一个线程执行的可行顺序；否则，则展示为什么该状态是不安全的。

	Allocation	Max
	A B C D	A B C D
T0	1 2 0 2	4 3 1 6
T1	0 1 1 2	2 4 2 4
T2	1 2 4 0	3 6 5 1
T3	1 2 0 1	2 6 2 3
T4	1 0 0 1	3 1 1 2

1. **Available** = (2，2，2，3)
2. **Available** = (4，4，1，1)
3. **Available** = (3，0，1，4)
4. **Available** = (1，5，2，2)

问题解答：

我们先把 **Need** 矩阵计算出来。

	Allocation	Max	Need
	A B C D	A B C D	A B C D
T0	1 2 0 2	4 3 1 6	3 1 1 4
T1	0 1 1 2	2 4 2 4	2 3 1 2
T2	1 2 4 0	3 6 5 1	2 4 1 1
T3	1 2 0 1	2 6 2 3	1 4 2 2
T4	1 0 0 1	3 1 1 2	2 1 1 1

1. 系统处于安全状态，我们可以按如下次序执行完：T4，T0，T1，T2，T3。
2. 系统处于安全状态，我们可以按如下次序执行完：T2，T4，T1，T3，T0。
3. 系统处于不安全状态。当前可用的B资源实例是0个，所有线程都无法执行。
4. 系统处于安全状态，我们可以按如下次序执行完：T3，T1，T2，T4，T0。

8.28 考虑下列一个系统某一时刻的快照。利用银行家算法回答下列问题。

	Allocation	Max	Available
	A B C D	A B C D	A B C D
T0	3 1 4 1	6 4 7 3	2 2 2 4
T1	2 1 0 2	4 2 3 2	
T2	2 4 1 3	2 5 3 3	
T3	4 1 1 0	6 3 3 2	
T4	2 2 2 1	5 6 7 5	

1. 给出一个线程执行顺序来展示该系统处于安全状态
2. 如果T4有一个请求（2，2，2，4），该请求能否立即被满足?
3. 如果T2有一个请求（0，1，1，0），该请求能否立即被满足?
4. 如果T3有一个请求（2，2，1，2），该请求能否立即被满足?

问题解答:

1. 我们给出当前系统的**Need**矩阵。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 1 4 1	6 4 7 3	3 3 3 2	2 2 2 4
T1	2 1 0 2	4 2 3 2	2 1 3 0	
T2	2 4 1 3	2 5 3 3	0 1 2 0	
T3	4 1 1 0	6 3 3 2	2 2 2 2	
T4	2 2 2 1	5 6 7 5	3 4 5 4	

   系统处于安全状态，我们可以按如下次序执行完：T2，T3，T1，T0，T4。

2. 此时系统快照如下。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 1 4 1	6 4 7 3	3 3 3 2	0 0 0 0
T1	2 1 0 2	4 2 3 2	2 1 3 0	
T2	2 4 1 3	2 5 3 3	0 1 2 0	
T3	4 1 1 0	6 3 3 2	2 2 2 2	
T4	4 4 4 5	5 6 7 5	1 2 3 0	

   不能被立刻满足。从上表可以看出，此时所有资源的可用示例均为0，所有线程都无法完成。

3. 此时的系统快照如下。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 1 4 1	6 4 7 3	3 3 3 2	2 1 1 4
T1	2 1 0 2	4 2 3 2	2 1 3 0	
T2	2 5 2 3	2 5 3 3	0 0 1 0	
T3	4 1 1 0	6 3 3 2	2 2 2 2	
T4	2 2 2 1	5 6 7 5	3 4 5 4	

可以被立刻满足。因为系统仍然处于安全状态，我们可以按如下次序执行完：T2，T3，T1，T0，T4。

4. 此时的系统快照如下。

	Allocation	Max	Need	Available
	A B C D	A B C D	A B C D	A B C D
T0	3 1 4 1	6 4 7 3	3 3 3 2	0 0 1 2
T1	2 1 0 2	4 2 3 2	2 1 3 0	
T2	2 4 1 3	2 5 3 3	0 1 2 0	
T3	6 3 2 2	6 3 3 2	0 0 1 0	
T4	2 2 2 1	5 6 7 5	3 4 5 4	

可以被立刻满足。因为系统仍然处于安全状态，我们可以按如下次序执行完：T3，T2，T1，T0，T4。