银行家算法

银行家算法之例

资源情况 / 进程	Max A	Max B	Max C	Allocation A	Allocation B	Allocation C	Need A	Need B	Need C	Available A	Available B	Available C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

1. 先假设work=available，即本例的work=available=[3,3,2]在进程中找到能满足下述条件的：

```
Finish[i]=false;
Need[i,j]<=work[j];
```

从P0-P4的所需资源Need中可以依次排除P0

```
[7,4,3]>[3,3,2]
```

又P1的Need为[1,2,2]

```
1. [1,2,2]<[3,3,2]
```

所以第一个为P1.P1进程结束之后：
现available=work+allocation,即现在可用资源是前一进程的可用资源加上前一进程的已分配的资源。于是，现在的可用资源[3,3,2]+[2,0,0]=[5,3,2].
依次地，按照之前地方法做比较：

```
1. [0,1,1]<[5,3,2]
```

即第二个是P3。P3结束，现可用资源work=[5,3,2]+[2,1,1]=[7,4,3]

```
1. [4,3,1]<[7,4,3]
```

即第三个是P4.P4结束，现可用资源是[7,4,5]=[7,4,3]+[0,0,2]

```
1. [6,0,0]<[7,4,5]
```

即第四个是P2，现可用资源是[10,4,7]=[7,4,5]+[3,0,2]

```
1. [7,4,3]<[10,4,7]
```

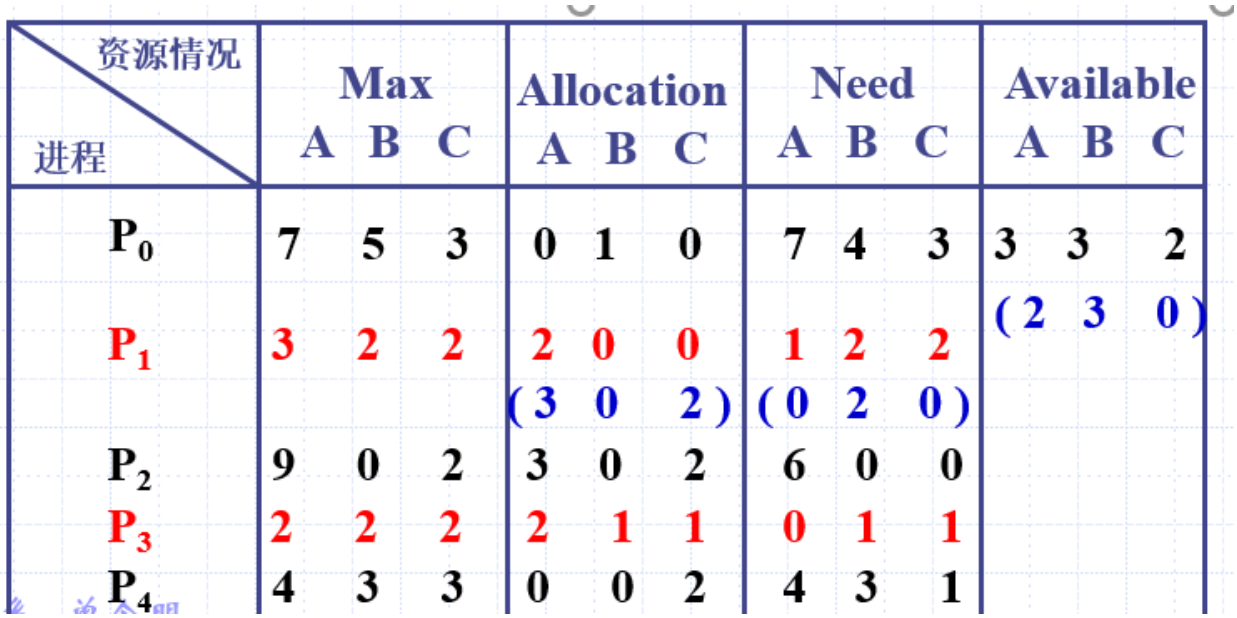
avaliable=work=[10,4,7]+[0,1,0]且现有资源数和题目所描述地A、B、C三类资源地数量10、5、7匹配.
综上所述：
存在一个安全序列{P1,P3,P4,P2,P0}。

资源情况 / 进程	Work A	Work B	Work C	Need A	Need B	Need C	Allocation A	Allocation B	Allocation C	W+A A	W+A B	W+A C	Finish
P_1	3	3	2	1	2	2	2	0	0	5	3	2	true
P_3	5	3	2	0	1	1	2	1	1	7	4	3	true
P_4	7	4	3	4	3	1	0	0	2	7	4	5	true
P_2	7	4	5	6	0	0	3	0	2	10	4	7	true
P_0	10	4	7	7	4	3	0	1	0	10	5	7	true

2. P1发出Request1(1，0，2)，系统检查：

```
Request1(1, 0, 2)≤Need1(1, 2, 2);
Request1(1, 0, 2)≤Available1(3, 3, 2)。
```

系统尝试分配资源：



资源情况 / 进程	Max A	Max B	Max C	Allocation A	Allocation B	Allocation C	Need A	Need B	Need C	Available A	Available B	Available C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2	(2	3	0)
				(3	0	2)	(0	2	0)			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

由安全性可以得到一个安全序列：{P1,P3,P4,P2,P0}。因此，系统是安全地，可以立即将P1申请地资源分配给它。

3. P4发出请求向量Request4(3，3，0)，系统按银行家算法进行检查：

```
Request4(3, 3, 0)≤Need4(4, 3, 1);
Request4(3, 3, 0)  >Available(2, 3, 0),
```

让P4等待。

4. P0发出请求向量Requst0(0，2，0)，系统按银行家算法进行检查
   ① Request0(0, 2, 0)≤Need0(7, 4, 3);
   ② Request0(0, 2, 0)≤Available(2, 3, 0);
   ③ 系统暂时先假定可为P0分配资源，并修改有关数据

资源情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	3	0	7	2	3	2	1	0
P_1	3	2	2	3	0	2	0	2	0			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

进行安全性检查，可用资源不能满足任何进程，系统不分配资源给P0