

|  |
| --- |
| A B C |
| 3 4 7 |
| 1 3 4 |
| 0 0 6 |
| 2 2 1 |
| 1 1 0 |
|  |

由已知得，P1- P5 分别需要：

|  |
| --- |
| P1 |
| P2 |
| P3 |
| P4 |
| P5 |

一共有 2 3 3 个资源

（1）

1. 利用安全性算法对T0时刻的资源分配情况进行分析：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 最大  A B C | 需要  A B C | 已分配  A B C | 和  A B C |  |
| 2 3 3 | 0 0 6 | 4 0 5 | 6 3 8 | true |
| 6 3 8 | 2 2 1 | 2 0 4 | 8 3 12 | true |
| 8 3 12 | 1 1 0 | 3 1 4 | 11 4 16 | true |
| 11 4 16 | 3 4 7 | 2 1 2 | 13 5 18 | true |
| 13 5 18 | 1 3 4 | 4 0 2 | 17 5 20 | true |

所以 在T0时刻存在一个安全序列{P3,P4,P5,P1,P2}，所以系统是安全的。

（2）

P2发出请求向量Request2(0,3,4)，系统按银行家算法进行检查：

Request2(0,3,4)≤Need2(1,3,4)

Request2(0,3,4)＞Available2(2,3,3)

不予分配，让P2等待。

（3）

1. P4发出请求向量Request4(2,0,1)，系统按银行家算法进行检查：

Request4(2,0,1)≤Need4(2,2,1)

Request4(2,0,1)≤Available4(2,3,3)

系统先假定可为P4分配资源，并修改变量，由此形成的资源变化情况

|  |  |  |
| --- | --- | --- |
| 最大  A B C | 已分配  A B C | 需要  A B C |
| 5 5 9 | 2 1 2 | 3 4 7 |
| 5 3 6 | 4 0 2 | 1 3 4 |
| 4 0 11 | 4 0 5 | 0 0 6 |
| 4 2 5 | 4 0 5 | 0 2 0 |
| 4 2 4 | 3 1 4 | 1 1 0 |

：

可分配资源数： 0 3 2

再利用安全性算法检查此时系统，可以找到一个安全序列{P3,P4,P5,P1,P2}。因此系统是安全的，可以立即将P4所申请的资源分配给它。

（4）

P1发出请求向量Request1(0,2,0)，系统按银行家算法进行检查：

Request1(0,2,0)≤Need1(3,4,7)

Request1(0,2,0)≤Available1(0,3,2)

|  |  |  |
| --- | --- | --- |
| 最大  A B C | 已分配  A B C | 需要  A B C |
| 5 5 9 | 2 3 2 | 3 2 7 |
| 5 3 6 | 4 0 2 | 1 3 4 |
| 4 0 11 | 4 0 5 | 0 0 6 |
| 4 2 5 | 4 0 5 | 0 2 0 |
| 4 2 4 | 3 1 4 | 1 1 0 |

可知，可用资源Available(0,1,2)已不能满足任何进程的需要，所以系统进入不安全状态，此时系统不分配资源。

（word 打表格太死亡了TAT）