2017013516-王艺霏-银行家算法

（1）T0时刻系统处于安全状态，安全序列为{P5,P4,P3,P2,P1}。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 进程  资源 | Work | Need | Allocation | Work+Allocation | Finish |
| A B C | A B C | A B C | A B C |
| P5 | 2 3 3 | 1 1 0 | 3 1 4 | 5 4 7 | true |
| P4 | 5 4 7 | 2 2 1 | 2 0 4 | 7 4 11 | true |
| P3 | 7 4 11 | 0 0 6 | 4 0 5 | 11 4 16 | true |
| P2 | 11 4 16 | 1 3 4 | 4 0 2 | 15 4 18 | true |
| P1 | 15 4 18 | 3 4 7 | 2 1 2 | 17 5 20 | true |

1. P2请求资源：P2:Request(0,3,4)

Request(0,3,4)<=Need(1,3,4);

Request(0,3,4)>Available(2,3,3)，让P2等待。

1. P4请求资源：P4:Request(2,0,1)

Request(2,0,1)<=Need(2,2,1);

Request(2,0,1)<=Available(2,3,3);

系统暂时先假定可为P4分配资源，并修改有关数据，再进行安全性检查：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 进程  进程  资源 | Work | Need | Allocation | Work+Allocation | Finish |
| A B C | A B C | A B C | A B C |
| P4 | 0 3 2 | 0 2 0 | 4 0 5 | 4 3 7 | true |
| P3 | 4 3 7 | 0 0 6 | 4 0 5 | 8 3 12 | true |
| P2 | 8 3 12 | 1 3 4 | 4 0 2 | 12 3 14 | true |
| P5 | 12 3 14 | 1 1 0 | 3 1 4 | 15 4 18 | true |
| P1 | 15 4 18 | 3 4 7 | 2 1 2 | 17 5 20 | true |

所以，可以找到一个安全序列{P4,P3,P2,P5,P1}，因此系统是安全的，可以立即将P1申请的资源分配给它。

1. P1请求资源：P1:Request(0,2,0)

Request(0,2,0)<=Need(3,4,7);

Request(0,2,0)<=Available(0,3,2);

系统暂时先假定可为P1分配资源，并修改有关数据：

|  |  |  |  |
| --- | --- | --- | --- |
| 进程  进程  资源 | Allocation | Need | Available |
| A B C | A B C | A B C |
| P1 | 2 3 2 | 3 2 7 | 0 1 2 |
| P2 | 4 0 2 | 1 3 4 |  |
| P3 | 4 0 5 | 0 0 6 |  |
| P4 | 4 0 5 | 0 2 0 |  |
| P5 | 3 1 4 | 1 1 0 |  |

因为此时Available（0,1,2）已不能满足任何进程的需求，故系统进入不安全状态，此时系统不分配资源。