银行家算法作业：

1.初始：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | max | | | allocation | | | need | | | available | | |
|  | A | B | C | A | B | C | A | B | C | A | B | C |
| P1 | 5 | 5 | 9 | 2 | 1 | 2 | 3 | 4 | 7 | 2 | 3 | 3 |
| P2 | 5 | 3 | 6 | 4 | 0 | 2 | 1 | 3 | 4 |  |  |  |
| P3 | 4 | 0 | 11 | 4 | 0 | 5 | 0 | 0 | 6 |  |  |  |
| P4 | 4 | 2 | 5 | 2 | 0 | 4 | 2 | 2 | 1 |  |  |  |
| P5 | 4 | 2 | 4 | 3 | 1 | 4 | 1 | 1 | 0 |  |  |  |

所以安全序列：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | work | | | need | | | allocation | | | Work+allocation | | | finish |
|  | A | B | C | A | B | C | A | B | C | A | B | C |
| P4 | 2 | 3 | 3 | 2 | 2 | 1 | 2 | 0 | 4 | 4 | 3 | 7 | true |
| P2 | 4 | 3 | 7 | 1 | 3 | 4 | 4 | 0 | 2 | 8 | 3 | 9 | true |
| P3 | 8 | 3 | 9 | 0 | 0 | 6 | 4 | 0 | 5 | 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 |

最终安全序列为：p4->p2->p3->p5->p1,T0时刻是安全的。

2. T0时刻：Request2(0,3,4)<=need2(1,3,4)

Request2(0,3,4)>Available(2,3,3)

所以暂时不能分配，P2等待。

3. P4:Request4(2,0,1)

Request4(2,0,1)<=need4(2,2,1)

Request2(2,0,1)<=Available(2,3,3)

P4申请后，此时表格值：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | max | | | allocation | | | need | | | available | | |
|  | A | B | C | A | B | C | A | B | C | A | B | C |
| P1 | 5 | 5 | 9 | 2 | 1 | 2 | 3 | 4 | 7 | 2 | 3 | 3 |
|  |  |  |  |  |  |  |  |  |  | 0 | 3 | 2 |
| P2 | 5 | 3 | 6 | 4 | 0 | 2 | 1 | 3 | 4 |  |  |  |
| P3 | 4 | 0 | 11 | 4 | 0 | 5 | 0 | 0 | 6 |  |  |  |
| P4 | 4 | 2 | 5 | 2 | 0 | 4 | 2 | 2 | 1 |  |  |  |
|  |  |  |  | 4 | 0 | 5 | 0 | 2 | 0 |  |  |  |
| P5 | 4 | 2 | 4 | 3 | 1 | 4 | 1 | 1 | 0 |  |  |  |

此时安全性：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 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 |
| P2 | 4 | 3 | 7 | 1 | 3 | 4 | 4 | 0 | 2 | 8 | 3 | 9 | true |
| P3 | 8 | 3 | 9 | 0 | 0 | 6 | 4 | 0 | 5 | 12 | 3 | 14 | true |
| P5 | 12 | 3 | 14 | 2 | 2 | 1 | 3 | 1 | 4 | 15 | 4 | 18 | true |
| P1 | 15 | 4 | 18 | 3 | 4 | 7 | 2 | 1 | 2 | 17 | 5 | 20 | true |

所以安全序列为：P4-->P2-->P3-->P5-->P1

所以可以分配

4.P1:Request1(0,2,0)

Request1(0,2,0)<=need1(3,4,7)

Request1(0,2,0)<=Available(0,3,2)

P1申请后：表格变化为：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | max | | | allocation | | | need | | | available | | |
|  | A | B | C | A | B | C | A | B | C | A | B | C |
| P1 | 5 | 5 | 9 | 2 | 1 | 2 | 3 | 4 | 7 | 2 | 3 | 3 |
|  |  |  |  | 2 | 3 | 2 | 3 | 2 | 7 | 0 | 3 | 2 |
|  |  |  |  |  |  |  |  |  |  | 0 | 1 | 2 |
| P2 | 5 | 3 | 6 | 4 | 0 | 2 | 1 | 3 | 4 |  |  |  |
| P3 | 4 | 0 | 11 | 4 | 0 | 5 | 0 | 0 | 6 |  |  |  |
| P4 | 4 | 2 | 5 | 2 | 0 | 4 | 2 | 2 | 1 |  |  |  |
|  |  |  |  | 4 | 0 | 5 | 0 | 2 | 0 |  |  |  |
| P5 | 4 | 2 | 4 | 3 | 1 | 4 | 1 | 1 | 0 |  |  |  |

因为Available(0,1,2)不满足任何一个进程的需要，所以不能分配，没有安全序列。