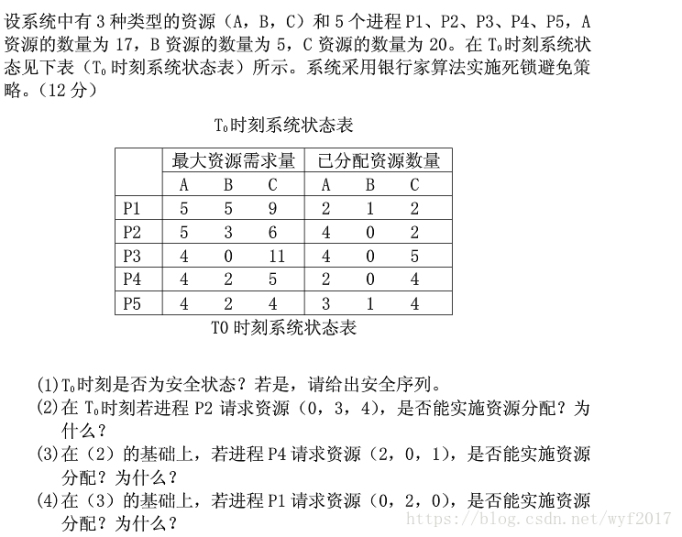
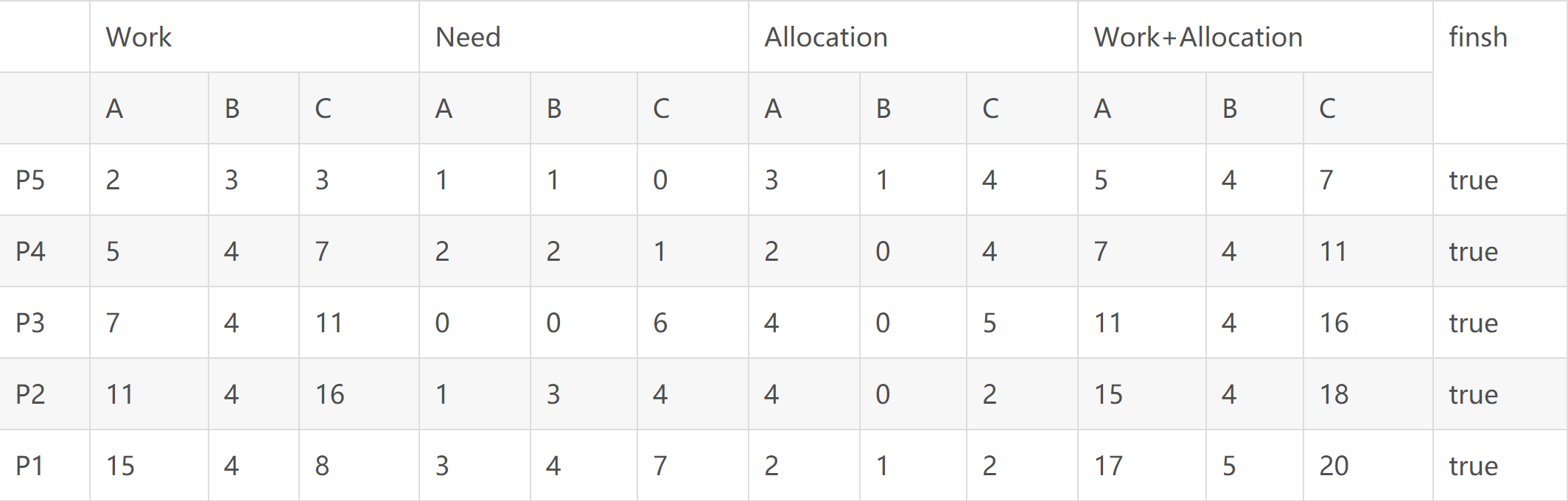
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | MAX | | | ALLOCATION | | | NEED | | | AVAILABLE | | |  |
|  | A | B | C | A | B | C | A | B | C | A | B | C |  |
|  | 5 | 5 | 9 | 2 | 1 | 2 | 3 | 4 | 7 | 2 | 2 | 3 |
|  | 5 | 3 | 6 | 4 | 0 | 2 | 1 | 3 | 4 |  |  |  |
|  | 4 | 0 | 11 | 4 | 0 | 5 | 0 | 0 | 6 |  |  |  |
|  | 4 | 2 | 5 | 2 | 0 | 4 | 2 | 2 | 1 |  |  |  |
|  | 4 | 2 | 4 | 3 | 1 | 4 | 1 | 1 | 0 |  |  |  |



因为系统资源R=（17,5,20）而系统分配给这几个线程的资源为Allocation=(15,2,17) 则可以求出Available=（2,3,3）

(1)在T0时刻，由于Availabel大于等于Need中 P5 所在行的向量，因此Availabel能满足 P5 的运行，在 P5 运行后，系统的状态变更为如下图所示:



因此，在T0时刻，存在安全序列：P5，P4，P3，P2，P1（并不唯一）

(2)P2请求资源，P2发出请求向量Request(i)(0,3,4),系统根据银行家算法进行检查;

 ① P2 申请资源Reuqest(i)（0,3,4）<=Need中 P2 所在行向量Need(i)（1,3,4）

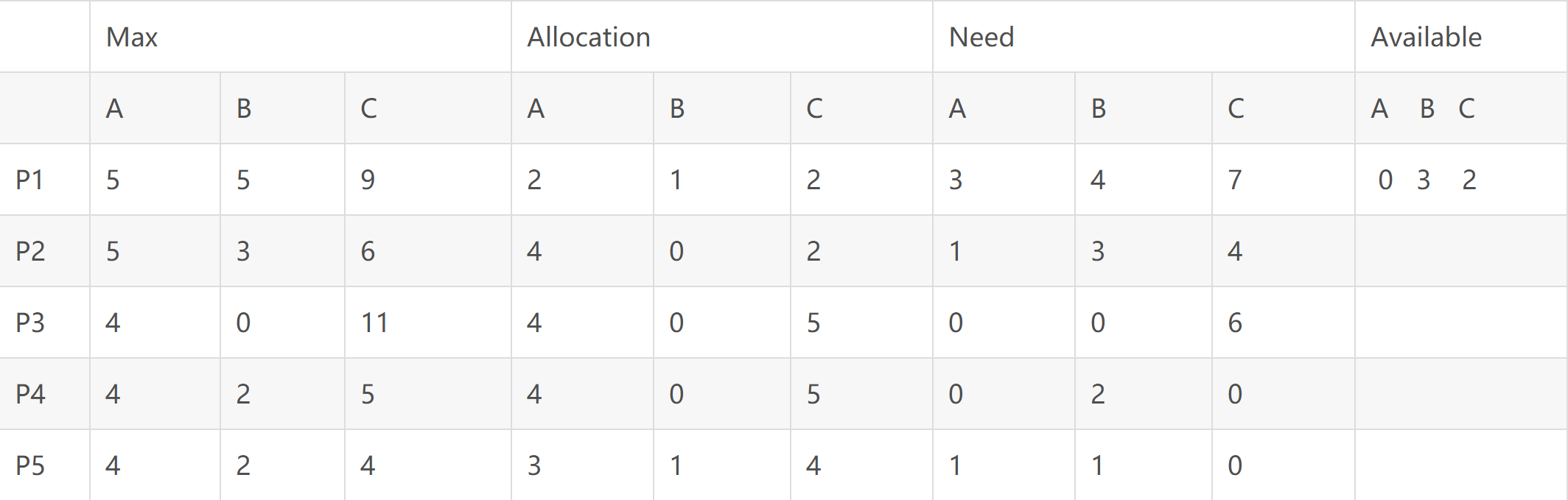
 ② P2 申请资源Reuqest(i)（0,3,4）>=可以利用资源向量Availabel（2,3,3），所以，该申请不给于分配

(3)P4请求资源，P4发出请求向量Request(i)(2,0,1),系统根据银行家算法进行检查;

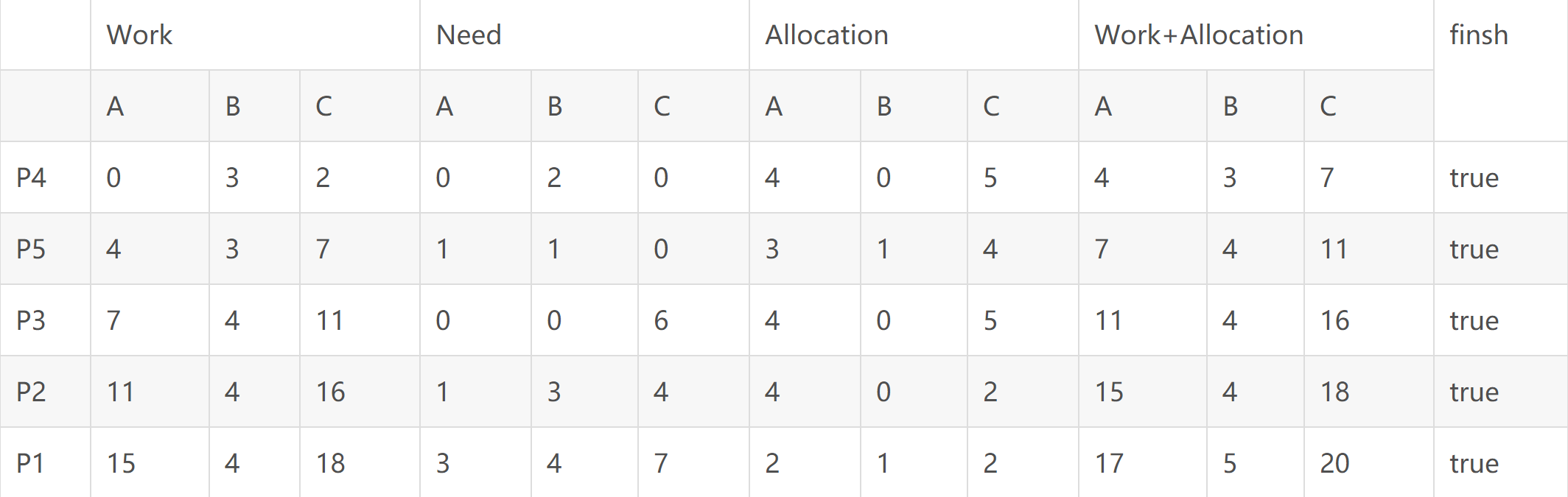
 ①Reuqest(i)（2,0,1）<= Need(i)（2,2,1）

 ② Reuqest(i)（2,0,1 <= Availabel（2,3,3）

 ③对 P4 的申请（2,0,1）进行预分配后，系统的状态为：



可利用资源向量Availabel=（0,3,2），大于Need中 P4 所在行的向量（0,2,0），因此可以满足 P4 的运行。P4 运行结束后，系统的状态变为：



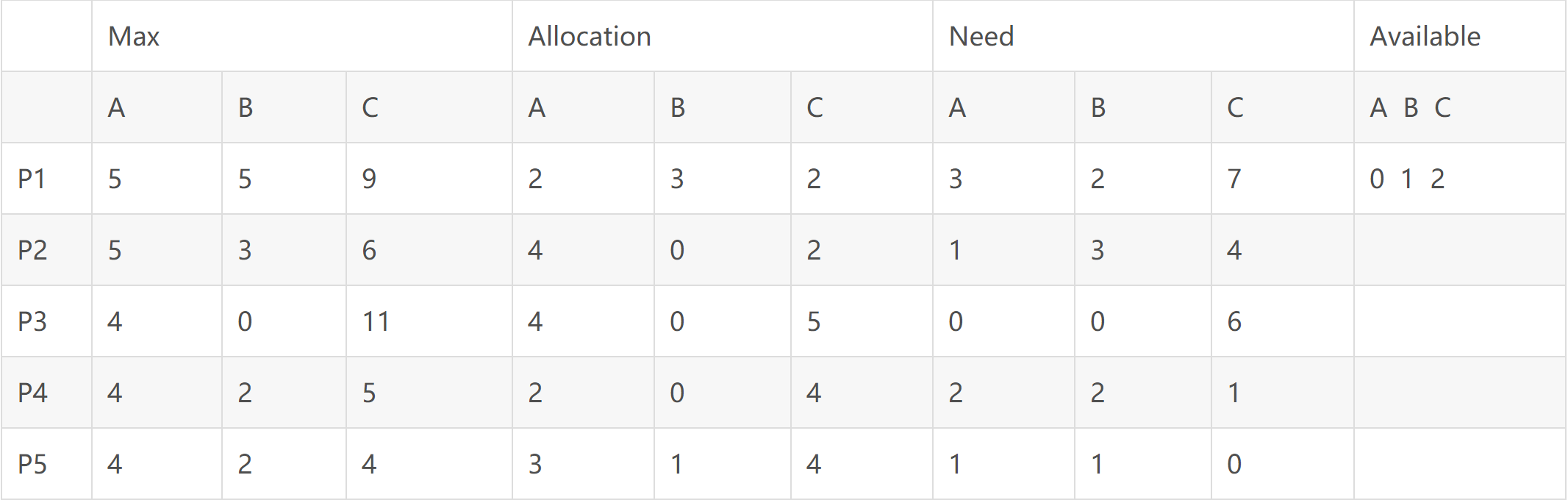
同理依次推导，可计算出存在安全序列P4，P5，P3，P2，P1(并不唯一）

(4)P1请求资源，P1发出请求向量Request(i)(0,2,0),系统根据银行家算法进行检查;

 ①Request(i)(0,2,0)<= Need(i)（3,4,7）

 ② Request(i)(0,2,0)<= Availabel（2,3,3）

 ③对 P1 的申请（0,2,0）进行预分配后，系统的状态为：



由于Availabel不大于等于 P1 到 P5 任一进程在Need中的需求向量，因此系统进行预分配后

处于不安全状态，所以对于 P1 申请资源（0,2,0）不给予分配。

注意:因为(4)是建立在第(3)问的基础上的所以Available=（0,3,2）-（0,2,0）=（0,1,2）