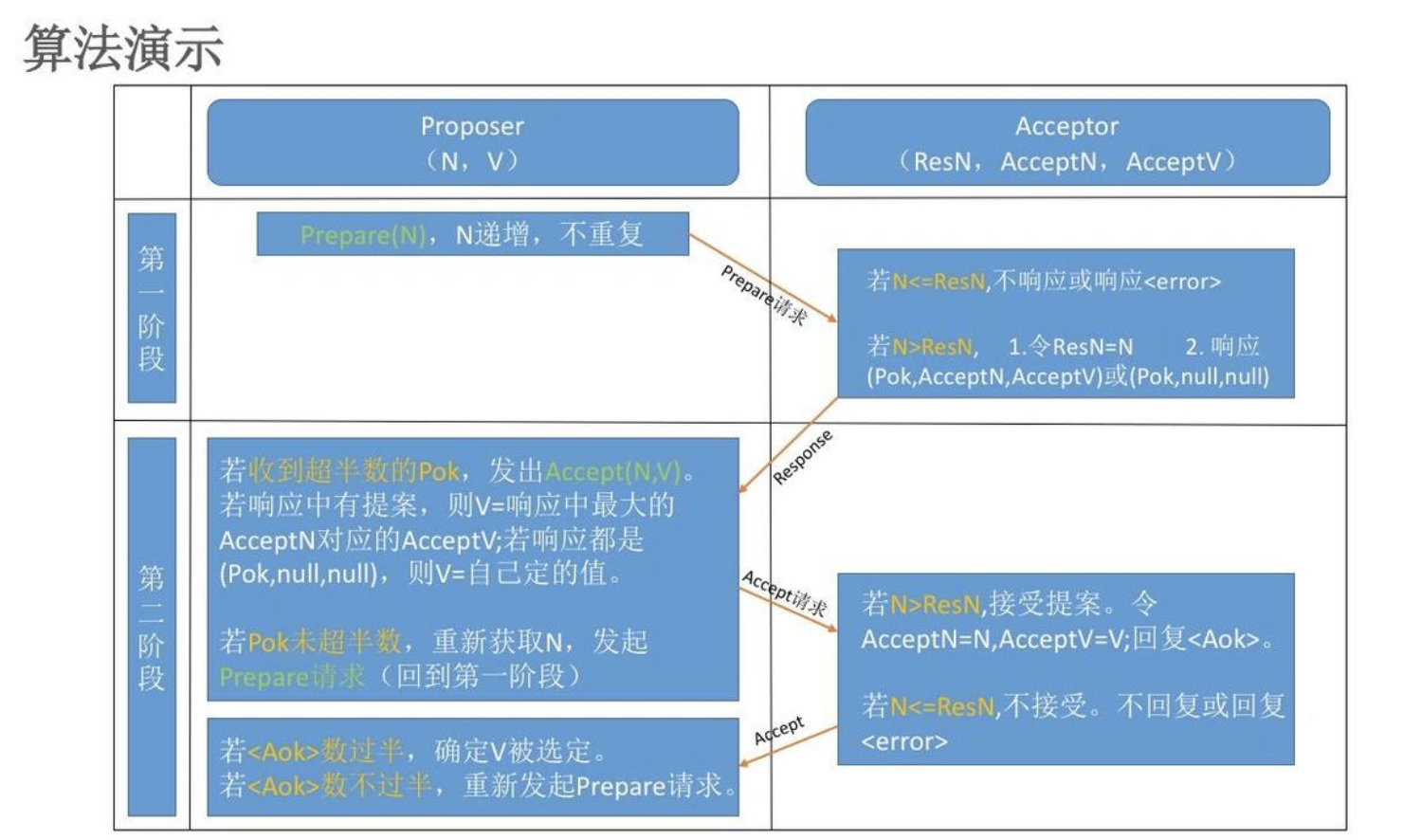
# 1.01 Paxos算法

* 最简单的方案——只有一个Acceptor

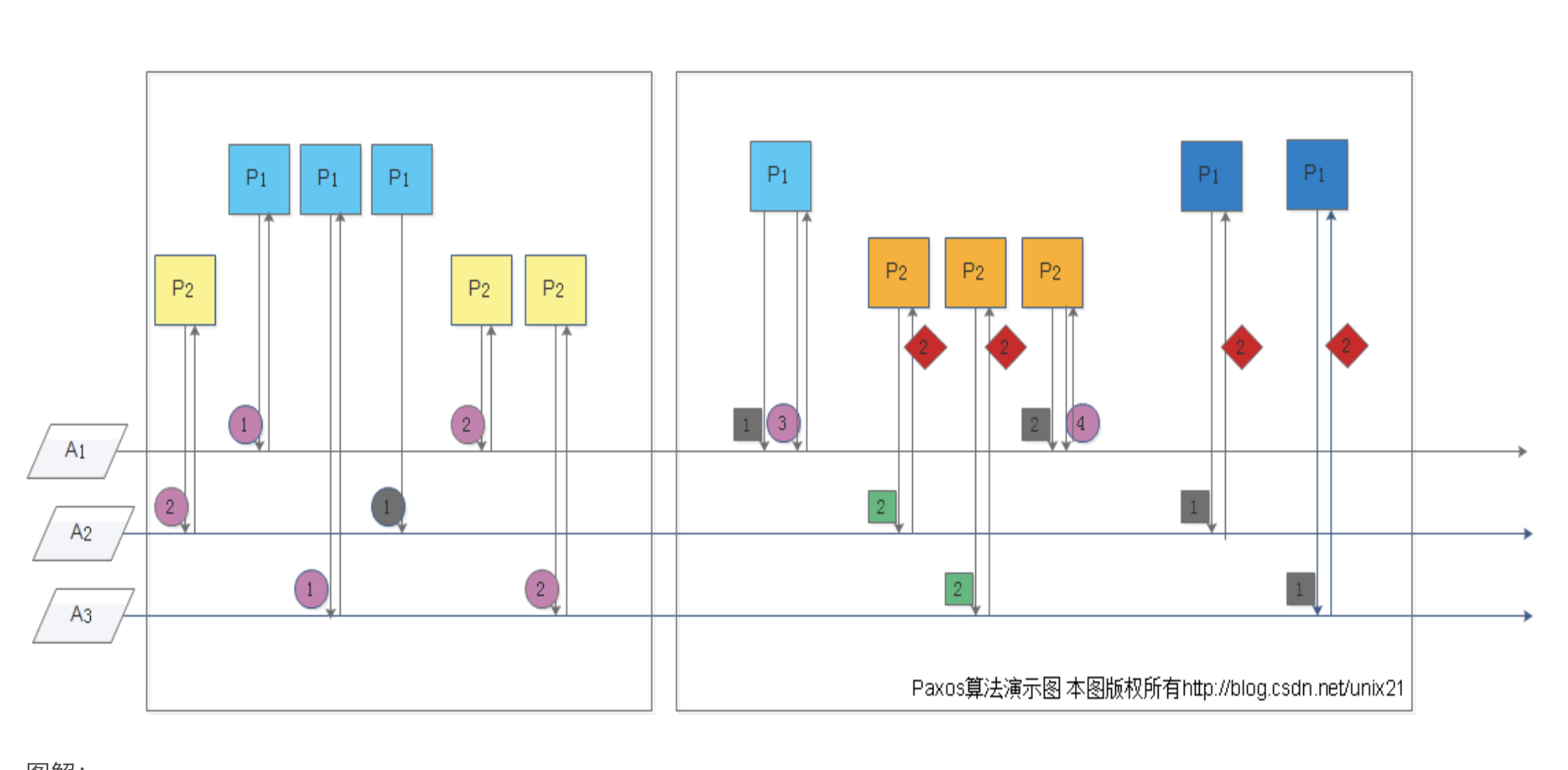
假设只有一个Acceptor（可以有多个Proposer），只要Acceptor接受它收到的第一个提案，则该提案被选定，该提案里的value就是被选定的value。这样就保证只有一个value会被选定。

但是，如果这个唯一的Acceptor宕机了，那么整个系统就**无法工作**了！

因此，必须要有**多个Acceptor**！



## 1.011 Message flow: Basic Paxos



图解：

A1，,A2和A3就是Acceptor。

P1，p2和p3就是Proposer。浅色的P1和P2说明是进行提议，深色的P1和P2说明是拿到表决。

圆圈123表明是每次提议序号，递增即可。黑色的图表示被黑了，也就是否决。方块表示投票结果，绿方块表示投票通过，红色菱形表示最终的投票结果。

整个事件是按照时间线从左到右发展。

事件发展：

第一个框代表第一阶段--提议

1. p2最先找到A2，P2提议序号是2，A2记录下，因为之前没有其他的序号所以成功了，然后返回标志给p2;

2. p1找到A1，P1提议序号是1，A1记录下，因为之前没有其他的序号所以成功了，然后返回标志给p1;

3. p1找到A3，P1提议序号是1，A3记录下，因为之前没有其他的序号所以成功了，然后返回标志给p1;

问题来了

4. p1找到A2，P1提议序号是1，A2已经记录下提议序号2，2>1，所以不成功;

5. p2找到A1，P2提议序号是2，A1已经记录下提议序号1，1<2，所以成功;，然后返回标志给p2;

6. p2找到A3，P2提议序号是2，A3已经记录下提议序号1，1<2，所以成功;，然后返回标志给p2;

第二个框代表第二阶段--确认提议（投票）

7.p1找到A1，P1确认序号是1，A1已经记录下提议序号2，1<2，所以不确认，然后p1继续提议序号是3，周而复始...;

8.p2找到A2，P2确认序号是2，A2已经记录下提议序号2，2=2，所以确认成功;，然后返回投票标志给p2;

9.p2找到A3，P2确认序号是2，A3已经记录下提议序号2，2=2，所以确认成功;，然后返回投票标志给p2;

10.p2找到A1，P2确认序号是2，A1已经记录下提议序号3，2<3，所以不确认，;然后p2继续提议序号是4，周而复始...;  
问题来了

11.p1找到A2，P1确认序号是1，A1已经记录下确认序号2，1<2，所以不确认，然后返回确认序号2;

12.p1找到A3，P1确认序号是1，A3已经记录下确认序号2，1<2，所以不确认，然后返回确认序号2;

13.p1和p2都得到确认也就是投票结果是2。

14.所有的Learner最终学习的目标是2。

Paxos过程结束了，这样，一致性得到了保证，算法运行到最后所有的proposer都投“2”所有的acceptor都接受这个议题，也就是说在最初的第二阶段，议题是先入为主的，谁先占了先机，后面的proposer在第一阶段就会学习到这个议题而修改自己本身的议题，才能让一致性得到保证，这就是paxos算法的一个过程。该算法就是为了追求结果的一致性。

注：

<https://www.cnblogs.com/linbingdong/p/6253479.html>

<https://blog.csdn.net/ma_jiang/article/details/79022284>