# 对Paxos算法的理解

Paxos算法是一种用于解决分布式系统中一致性问题的算法，是分布式领域中非常重要的算法。因其复杂性被称为难理解和实现的算法。有着信息传递和高度容错两个特点。其诞生背景是为了解决机器宕机和网络异常这两个问题。Paxos算法协议中有三类节点角色，分别是proposer（提案者）、acceptor（批准者）和learners（学习者）。

Paxos算法执行有两个阶段，阶段一是由proposer选择一个编号N，然后半数以上acceptor接受到这个编号N的proposer提案请求，如果acceptor收到一个编号N的proposer请求，小于它已经响应过的请求就拒绝，大于则响应编号N的请求，同时acceptor将不再接受编号小于N的提案。阶段二是如果一个proposer收到半数以上acceptor对其发出的编号为N的提案请求的响应，他就会发送一个[N,V]提案的accept请求给半数以上的acceptor（V是acceptor收到响应中编号最大的提案Value）。Acceptor收到针对编号N的提案的accept请求，如果没有对编号大于N的请求做出响应就接受该提案，反之则拒绝。

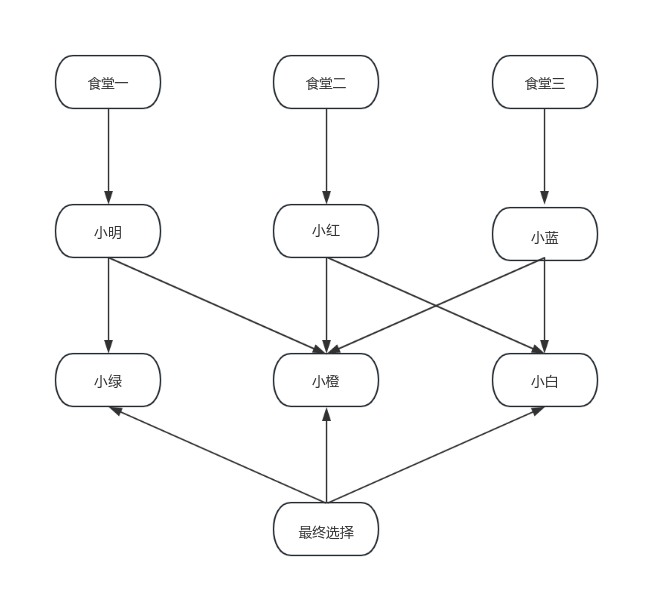
如下图寝室六人要去食堂吃饭，小明想去食堂一，小红想去食堂二，小蓝想去食堂三，需要从中选一个食堂，设小明、小红、小蓝他们分别将发消息给除他们三位的其他两名室友，且为了去心仪的食堂小明、小红、小蓝分别发了2块、1块、3块的红包给其他人。（如图，小明发送给小绿、小橙，小红发送给小橙、小白，小蓝发送给小橙、小白）。

小明的消息先传达给小绿、小橙，他们之前都没有收到消息所以同意了小明的请求，并且返回消息给小明告诉他不会再接受其他人的提议除非他们给的更多；然后小红的消息传达给了小橙、小白，因为小白没有接受过请求，所以同意了小红请求，且返回消息给小红告诉他不会再接受其他人的提议除非他们给的更多，而小橙已经答应过小明的请求，但是因为小红发的红包比小明的少，所以小橙拒绝小红的请求；最后小蓝的消息传达给小橙、小白，他们都接受过提议但是由于小蓝发的红包大于小橙接受的小明的红包和小白接受小红的红包，所以他们都接受小蓝的提议，且返回消息给小蓝告诉他不会再接受其他人的提议除非他们给的更多；此时小红知道自己的请求被拒绝了所以又发了三块给小橙、小白，小橙和小白就接受了小红的提案且返回消息给小红谢谢她。

小蓝收到了半数以上的回复，所以小蓝提交[3,食堂三]的提案，小明也收到了半数以上的回复，所以小明提交[2,食堂一]的提案，小红也收到了半数以上的回复，所以小红提交[4,食堂二]的提案。

小绿和小橙接收到小明的提案[2，食堂一]，小绿通过该请求,小橙承诺不再接受红包小于4块的提案，所以拒绝； 小橙和小白接收到小红的提案[4，食堂二]，都通过该提案； 小橙和小白接收到小蓝的提案[3，食堂三]，它们都承诺不再接受红包小于4块的提案，所以都拒绝。

此时小橙、小白都已经通过了提案，并达成了半数以上，所以小明、小红、小蓝将达成一致接受提案[4,食堂二]，他们在宿舍群知道提案通过，所以最终选择食堂二。



在以上列子里小明、小红、小蓝可以看作Paxos算法里的proposer，而小绿、小橙、小白则是acceptor，发送的红包就是proposer选择的提案编号。

Paxos算法还有一些安全性要求，一是只有被提出的value才可以被选定，二是只有一个value被选定，三是当某个进程已经认为某个value选定了，那么这个value必须是真实被选定。

Paxos算法的最终目标是为了保证最终只有一个被提出的value被选中，进程最终获取被选中的value。