重庆科技学院

题 目： 对paxos算法的理解

学生姓名： 魏雨楠 学 号： 2021443910

学 院： 数理与大数据学院 专业班级： 大数据21-02

完成日期： 2023 年 12月 4日

成绩（百分制）：

授课教师：

Paxos是一种分布式一致性算法，简单说就是在多台机器之间通过一致性协商过程，让所有机器保存同样的数据。客户端访问任一台机器，得到的结果都会是一样的。

这种一致性的存储十分重要。大型系统常常需要存储一些被频繁修改和访问的数据，例如配置，机器地址等。这种存储常常成为一种关键单点，如果单点故障，会影响整个系统的服务。因此，必须将单点变成分布式系统，也就是说，将数据同时存储在多台机器上，即使部分机器故障，依然不影响整个系统的服务。将数据存储多台机器，就必然产生一致性的问题。多台机器同时的改动，如何达成一致，同步到所有的机器。Paxos就是解决这种一致性问题的基础算法。

从最简单的思路出发去理解一个复杂的算法。

如果现在有多台机器存储同一份配置{“name”: “apple"}，某台机器接受请求要将配置改为{“name”:”banana"}，需要在系统内同步这一项改动。最简单的办法是直接向其他机器广播，让其他机器也改为{“name”:”banana"}。

这样会面临几个问题，首先，因为所有机器都可能发出修改配置的提议，如果两个机器同时广播，其他机器应该选择哪一个？如果按照到达的先后顺序，这样两个广播到达不同的机器的先后可能是不一样的，会导致不同机器选择了不同的修改请求，系统出现不一致。因此，一定需要有某种规则来确定优先级。最简单直接的就是比大小，给每一个提议分配一个自增的ID，多个提议到来，选择最大的。

其次，由于网络的不稳定性，很可能ID较大的请求并未到达某些机器，而这些机器却接受了ID较小的提议，系统出现了不一致。可见，一次提议是不可靠的，我们需要一个协商的过程，先看看其他机器是否已经有通过的提议，如果有，应该按照先到先得的原则，放弃自己的提议，如果没有，可以放心地让其他机器接受自己的提议。

然而，在发送让其他机器接受自己的提议时候，也会有很多状况发生。例如，在获得其他机器的协商反馈后，其他机器又接受了新的提议，这时你还以为别人并没有通过提议，正常发送提议。其他机器接收到你的提议后应该怎么处理？又例如，你发送完提议，需要获得其他机器的确认反馈，如果只有部分确认了怎么办？这时系统可能不一致了。

paxos的总结：

Paxos将角色分为提议者（proposer），接受者（acceptor），学习者（learner）。

算法流程如下：1，提议者任意发起提议。所有的提议都要分配一个不重复的自增的ID。这里提议者应该理解为发起提议的机器，因为同一个机器，也可以是接受者。同时多个机器可以同时发起提议，但是提议ID必须是全局自增的。理由上面已经讲了，需要给接受者作为优先级的凭证。

2，接受者要保存接受过的最大的ID，并且只处理大于等于当前ID的提议。如果提议ID合法，则返回本机器已同意接受的提议，没有则返回提议为空。只处理较大的ID是很有必要的，否则协商过程可能会一直反复，难以达成一致。

3，提议者如果收到过半数的机器返回，并且返回值都为空，提议者认为提议可行，向所有机器广播提交提议。如果返回值里面有非空，提议者放弃自身的提议，选择非空里面ID最大的提交。如果返回不足半数，跳到第一步，重新提议。这里体现了两个协商原则，首先，机器需要有过半数返回才能协商成功。其次，提议是先到先得，只要有其他提议先被接受了，当前提议者只能服从。

4，接受者接收到提交的提议，同样地只处理ID较大的提议。如果提议合法，则接受提议。

5，提议者如果收到过半数的机器接受提议，则提议成功。否则，返回第一步重试。

3和5环节失败了，提议者都会回到1环节获得新的ID重试，重试是很有必要的，保证了再网络不稳定的情况下，系统也能最大程度的保证达成一致。