你了解一致性算法吗？你了解Paxos算法吗?

- 关于Paxos
  - Paxos 算法解决的问题是一个分布式系统如何就某个值(决议)达成一致。一个典型的场景是，在一个分布式数据库系统中，如果各节点的初始状态一致，每个节点执行相同的操作序列，那么 他们最后能得到一个一致的状态。为保证每个节点执行相同的命令序列，需要在每一条指令上执 行一个“一致性算法”以保证每个节点看到的指令一致。zookeeper 使用的 zab 算法是该算法的 一个实现。 在 Paxos 算法中，有三种角色:Proposer，Acceptor，Learners
- **Paxos** 三种角色:**Proposer**，**Acceptor**，**Learners**
  - ***Proposer***:

    只要 Proposer 发的提案被半数以上 Acceptor 接受，Proposer 就认为该提案里的 value 被选定 了。
  - ***Acceptor***: 只要 Acceptor 接受了某个提案，Acceptor 就认为该提案里的 value 被选定了。
  - ***Learner***: Acceptor 告诉 Learner 哪个 value 被选定，Learner 就认为那个 value 被选定。
- **Paxos** 算法分为哪两个阶段。具体是什么?
  - 阶段一(准 ***leader*** 确定):
    - Proposer 选择一个提案编号 N，然后向半数以上的 Acceptor 发送编号为 N 的 Prepare 请求
    - 如果一个 Acceptor 收到一个编号为 N 的 Prepare 请求，且 N 大于该 Acceptor 已经响应过的 所有 Prepare 请求的编号，那么它就会将它已经接受过的编号最大的提案(如果有的话)作为响 应反馈给 Proposer，同时该 Acceptor 承诺不再接受任何编号小于 N 的提案。
  - 阶段二(***leader*** 确认):
    - 如果 Proposer 收到半数以上 Acceptor 对其发出的编号为 N 的 Prepare 请求的响应，那么它 就会发送一个针对[N,V]提案的 Accept 请求给半数以上的 Acceptor。注意:V 就是收到的响应中 编号最大的提案的 value，如果响应中不包含任何提案，那么 V 就由 Proposer 自己决定
    - 如果 Acceptor 收到一个针对编号为 N 的提案的 Accept 请求，只要该 Acceptor 没有对编号 大于 N 的 Prepare 请求做出过响应，它就接受该提案。