Paxos认识和理解

Paxos算法是莱斯利·兰伯特于1990年提出的一种基于消息传递且具有高度容错特性的一致性算法。Paxos算法解决的问题正是分布式一致性问题，即一个分布式系统中的各个进程如何就某个值（决议）达成一致。

Paxos算法运行在允许宕机故障的异步系统中，不要求可靠的消息传递，可容忍消息丢失、延迟、乱序以及重复。它利用大多数机制保证了2F+1的容错能力，即2F+1个节点的系统最多允许F个节点同时出现故障。

一个或多个提议进程可以发起提案，Paxos算法使所有提案中的某一个提案，在所有进程中达成一致。系统中的多数派同时认可该提案，即达成了一致。最多只针对一个确定的提案达成一致。

Paxos将系统中的角色分为提议者 (Proposer)，决策者 (Acceptor)，和最终决策学习者 (Learner):Proposer: 提出提案 (Proposal)。Proposal信息包括提案编号 (Proposal ID) 和提议的值 (Value)。Acceptor：参与决策，回应Proposers的提案。收到Proposal后可以接受提案，若Proposal获得多数Acceptors的接受，则称该Proposal被批准。Learner：不参与决策，从Proposers/Acceptors学习最新达成一致的提案（Value）。在多副本状态机中，每个副本同时具有Proposer、Acceptor、Learner三种角色。

Paxos算法通过一个决议分为两个阶段：

1.prepare阶段：proposer选择一个提案编号n并将prepare请求发送给acceptors中的一个多数派；acceptor收到prepare消息后，如果提案的编号大于它已经回复的所有prepare消息(回复消息表示接受accept)，则acceptor将自己上次接受的提案回复给proposer，并承诺不再回复小于n的提案；

2.批准阶段：当一个proposer收到了多数acceptors对prepare的回复后，就进入批准阶段。它要向回复prepare请求的acceptors发送accept请求，包括编号n和根据P2c决定的value（如果根据P2c没有已经接受的value，那么它可以自由决定value）。在不违背自己向其他proposer的承诺的前提下，acceptor收到accept请求后即批准这个请求。