一种易于理解的一致性算法 – Raft

习钟 常慧敏 刘政江
郑万啸 梁传栋 谈海波

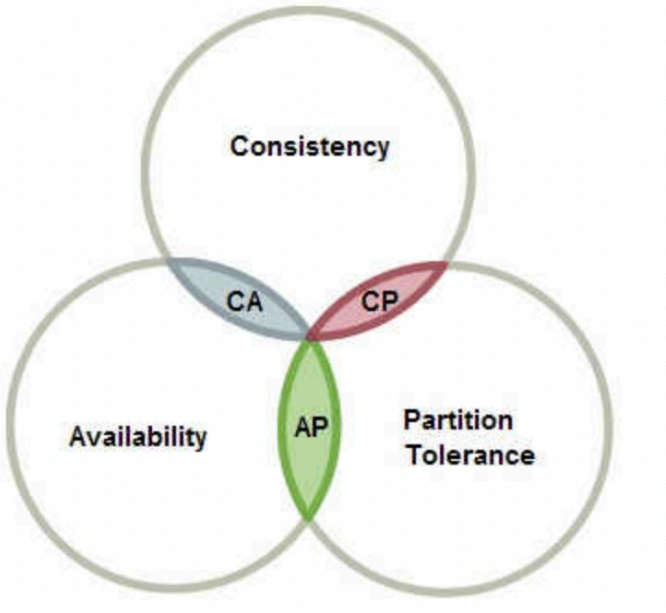
2021 年 11 月 25 日

分布式系统

大数据时代
Google “三驾马车”

- Google File System
- Google MapReduce
- Google BigTable

CAP 原理



- Consistency
- Availability
- Partition Tolerance

Why

Q：为什么需要一致性?

Why

Q：为什么需要一致性?

1. 数据不能存在单个节点（主机）上，否则可能出现单点故障
2. 多个节点（主机）需要保证具有相同的数据

“一致性就是数据保持一致，在分布式系统中，可以理解为多个节点中数据的值是一致的。”

“组织机器使其最终状态一致并允许局部失败的算法称为一致性算法。”

一致性

弱一致性

- DNS
- Gossip

强一致性

1. Paxos
2. Raft(ETCD)
3. ZAB(ZooKeeper)

易于理解（Understandable）

Before Raft,

“在过去的 10 年里，Leslie Lamport（2013 年图灵奖获得者）的 Paxos 算法几乎已经成为一致性的代名词”

易于理解（Understandable）

Before Raft,

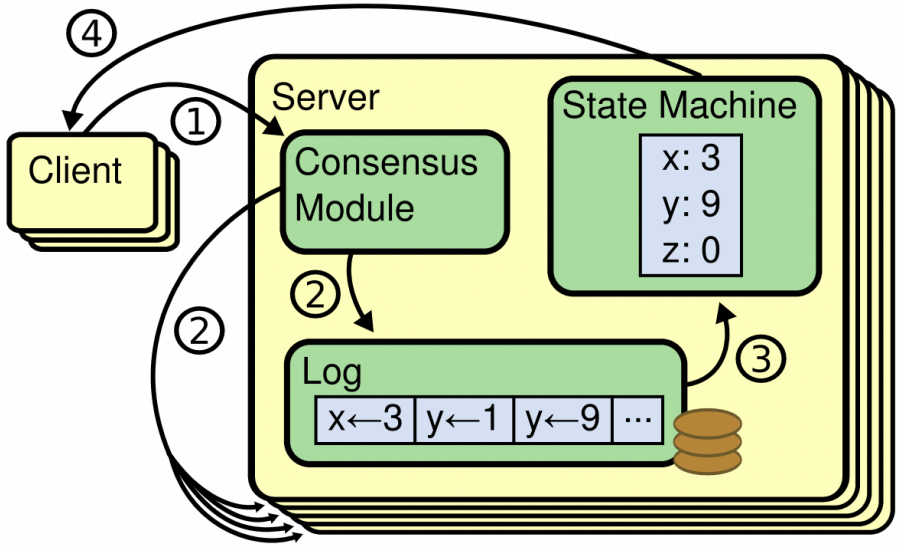
“在过去的 10 年里，Leslie Lamport（2013 年图灵奖获得者）的 Paxos 算法几乎已经成为一致性的代名词”

But,

表: 一致性算法比较

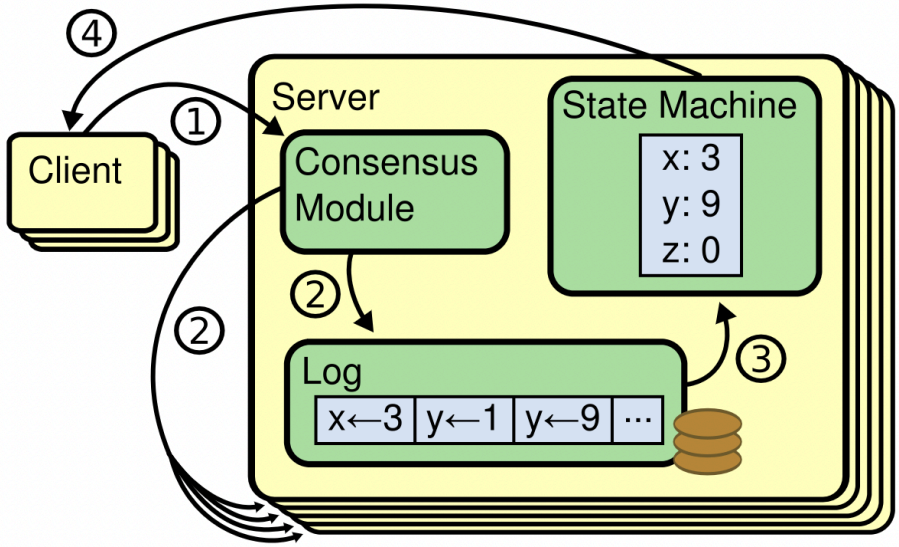
Paxos	Raft
难以理解	易于理解
只有思路	明确实现

复制状态机 (Replicated state machine)



- 集群指令（读或写）同步给所有节点
- 状态变更状态
- 网络延迟远远慢于内存操作
- 不要求立即被执行，只需维护一份指令顺序一致的日志

复制状态机 (Replicated state machine)



- 集群指令（读或写）同步给所有节点
- 状态变更状态
- 网络延迟远远慢于内存操作
- 不要求立即被执行，只需维护一份指令顺序一致的日志

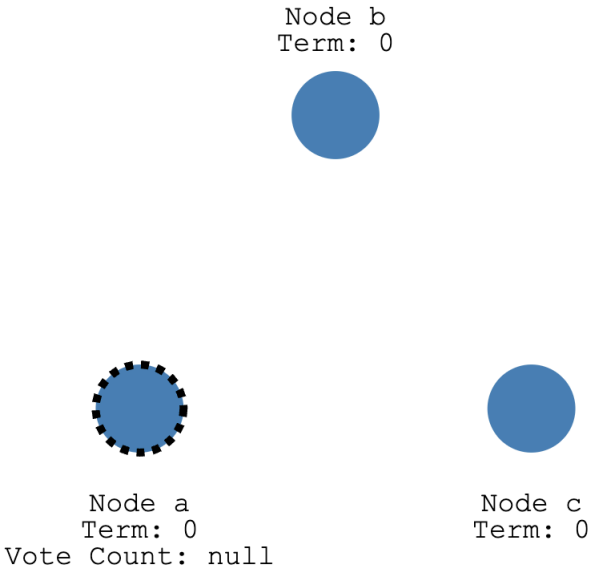
“指令日志的维护”

目标

输入：写入命令
输出：所有节点最终处于相同状态
约束条件：

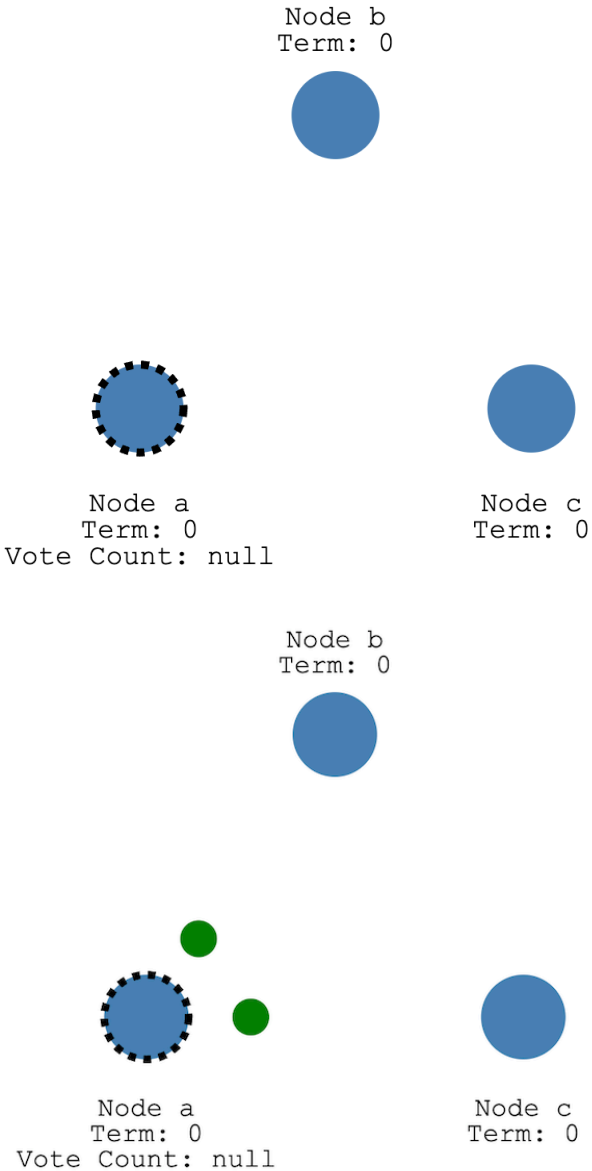
- 网络不确定性
- 基本可用性
- 不依赖物理时钟保持一致
- 快速响应（不依赖最慢节点）

Step



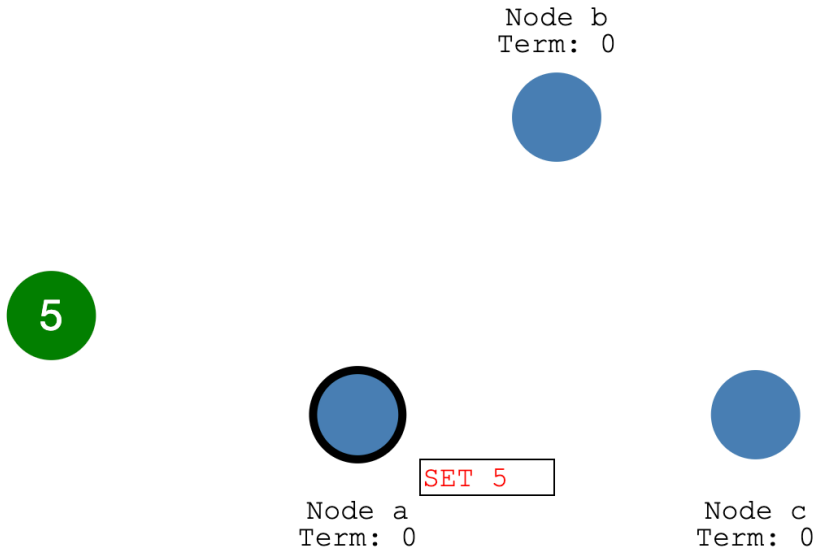
- Node
- term
- Vote count

Step

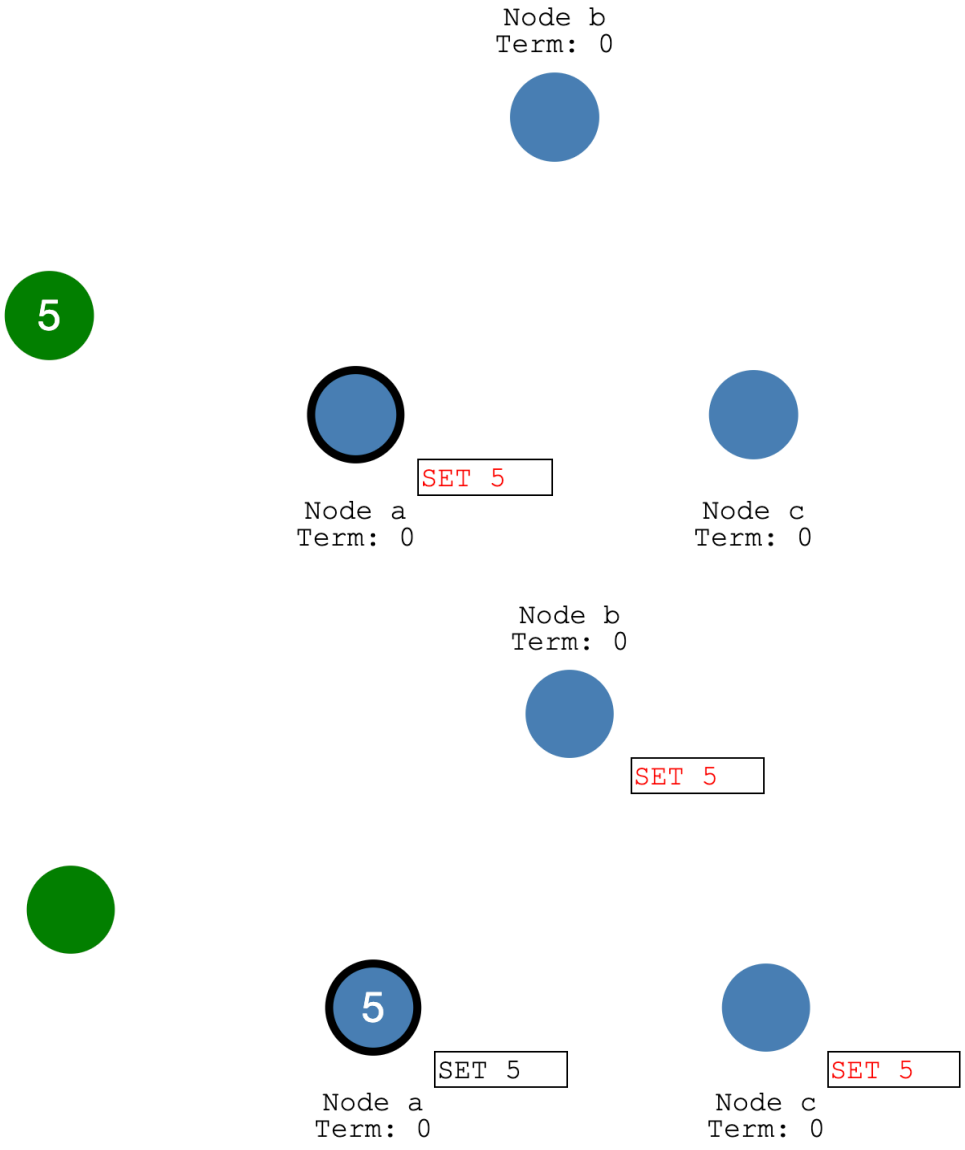


- Node
- term
- Vote count

节点 a 请求节点 b, c 投票



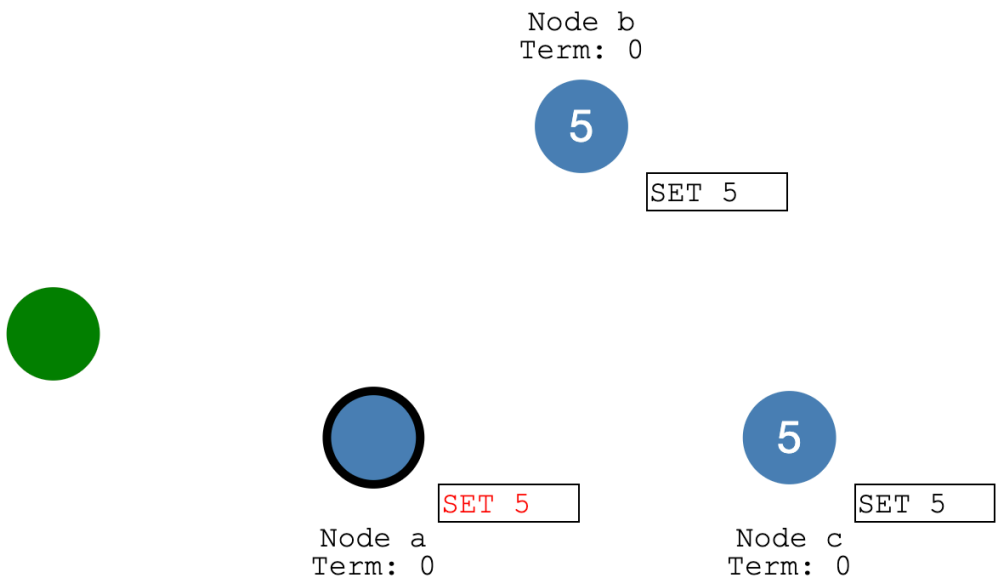
- 节点 a 成为了领导者，管理整个集群
- 每个更改都作为一个条目添加到节点日志中
- 日志并不会提交，因为还没有更改节点的值



- 节点 a 成为了领导者，管理整个集群
- 每个更改都作为一个条目添加到节点日志中
- 日志并不会提交，因为还没有更改节点的值

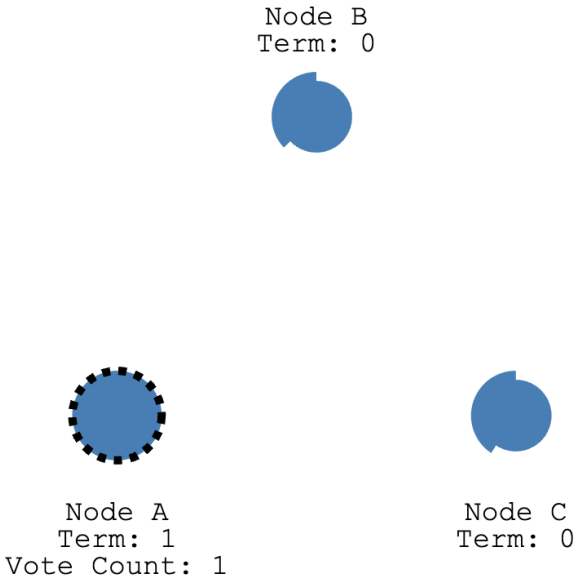
节点 a 复制日志到其他节点后并获得大多数确定后提交日志，改变自身值

日志复制（Log Replication）



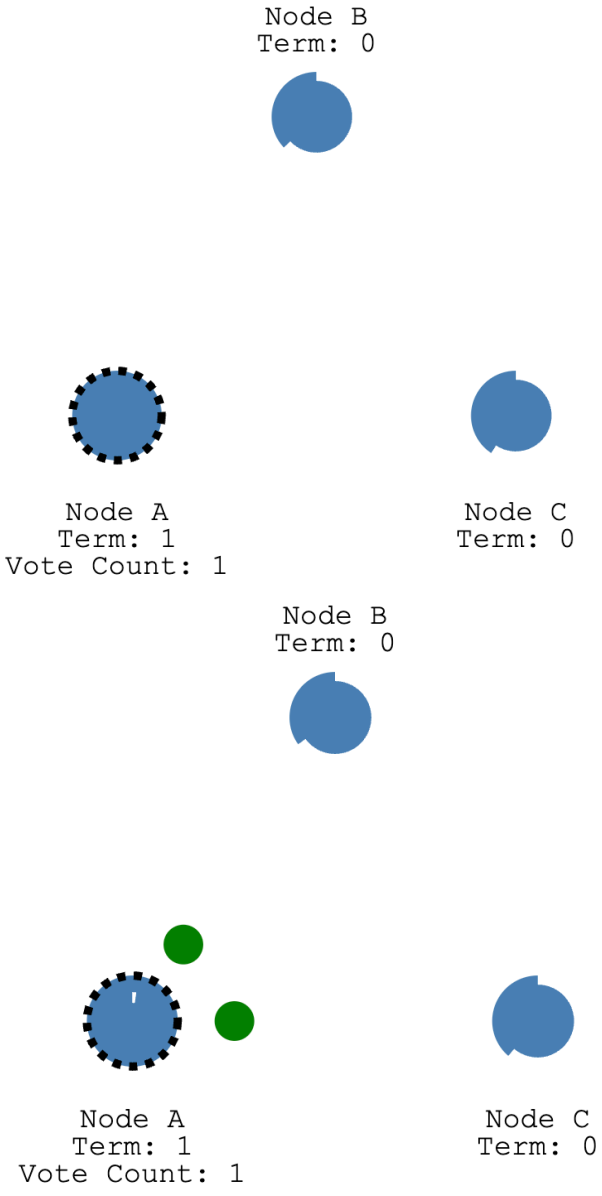
- 节点 a 通知其他节点，日志已经提交
- 其他节点自己可以修改自身的值了

Leader Selection



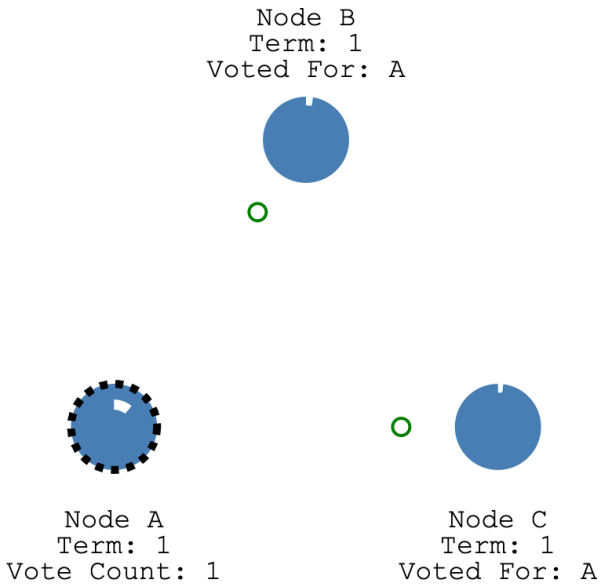
- timeset
- randon 100ms
  300ms

Leader Selection

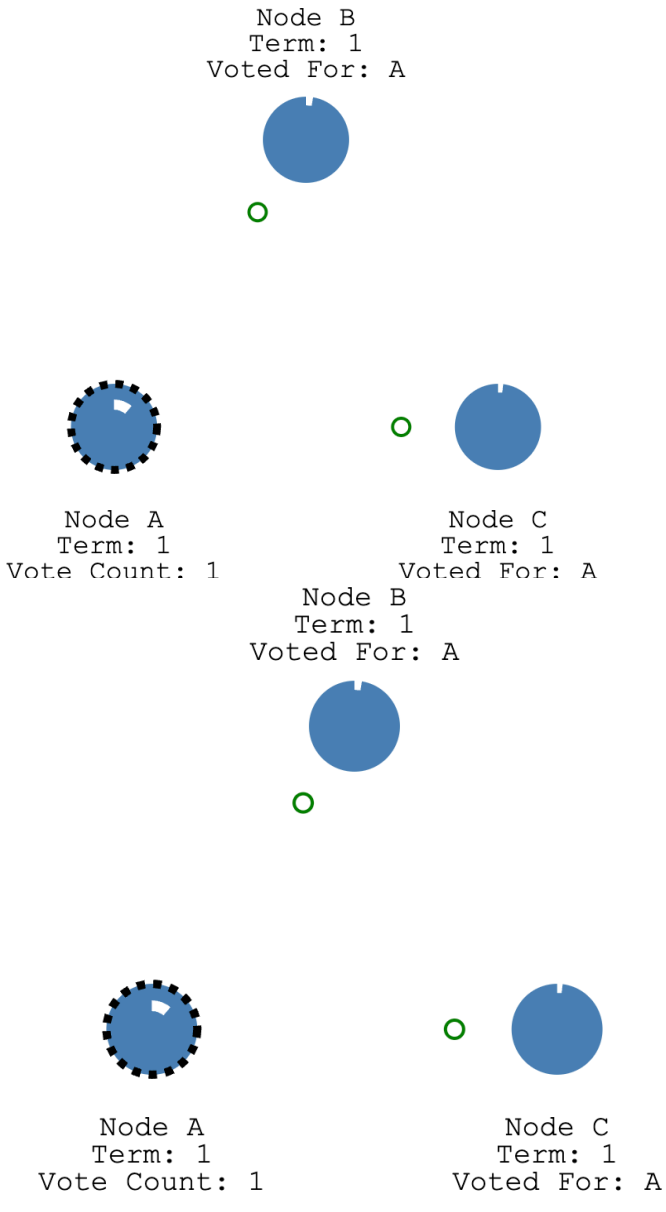


- timeset
- randon 100ms 300ms

节点 a 请求节点 b, c 投票 (Request Vote messages to other nodes.)



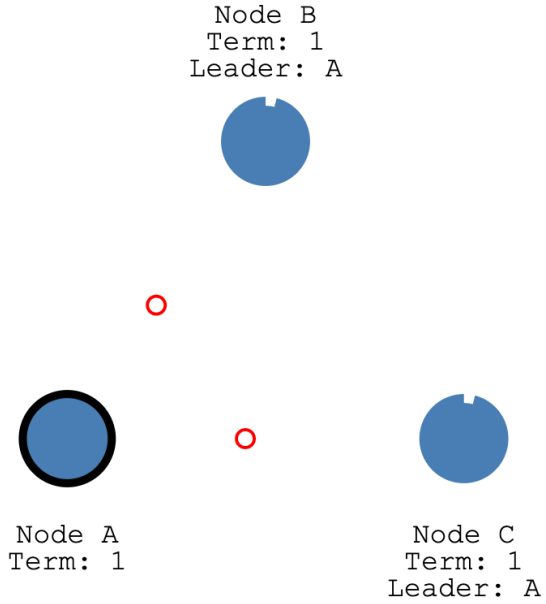
- If the receiving node hasn't voted yet in this term then it votes for the candidate...



- If the receiving node hasn't voted yet in this term then it votes for the candidate...

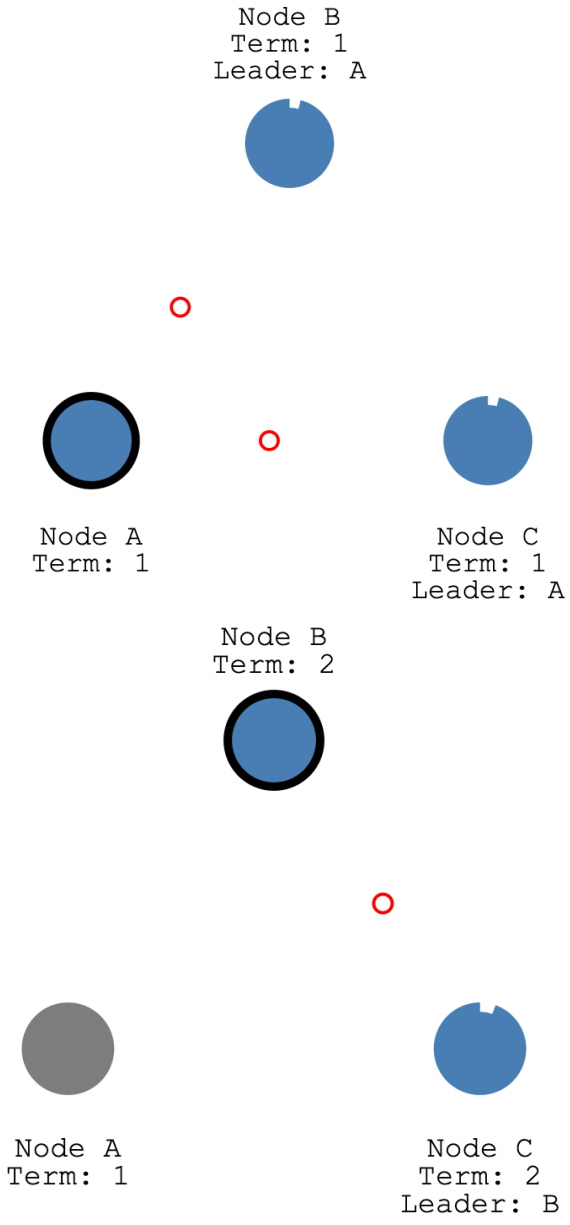
开始新一轮的计时（node resets its election timeout.）

heartbeats



- 领导者节点不断通过心跳机制来维护自己的统治

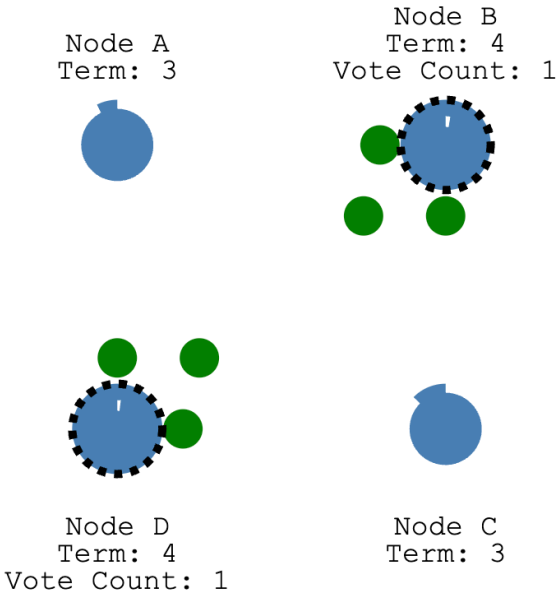
heartbeats



- 领导者节点不断通过心跳机制来维护自己的统治

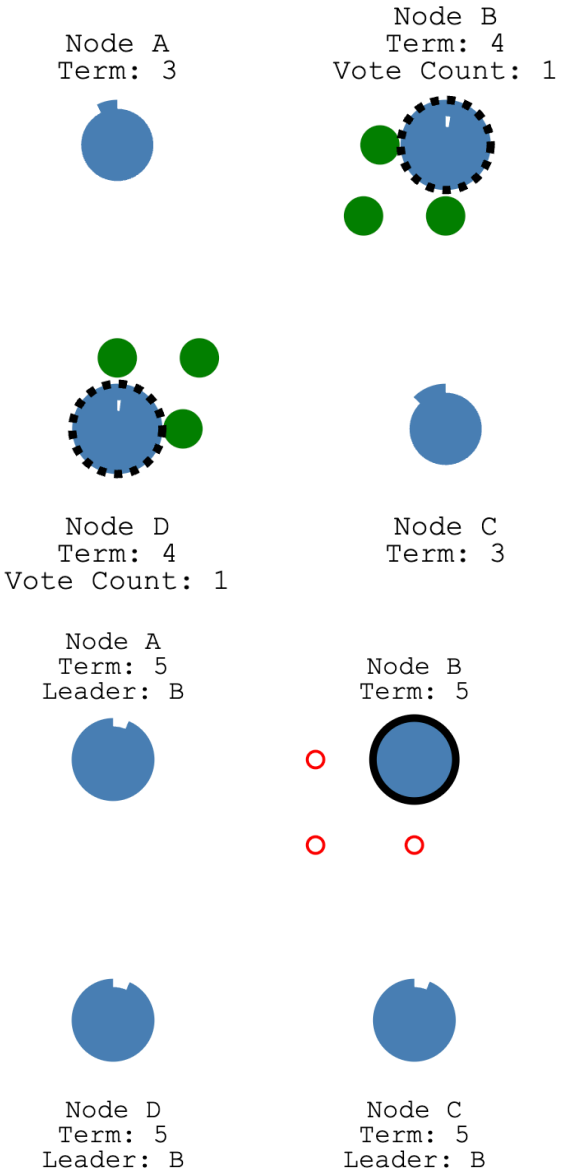
在一个 term 内收不到发过来的心跳机制就开始新一轮选举

heartbeats



- 多个节点的时候，票数冲突

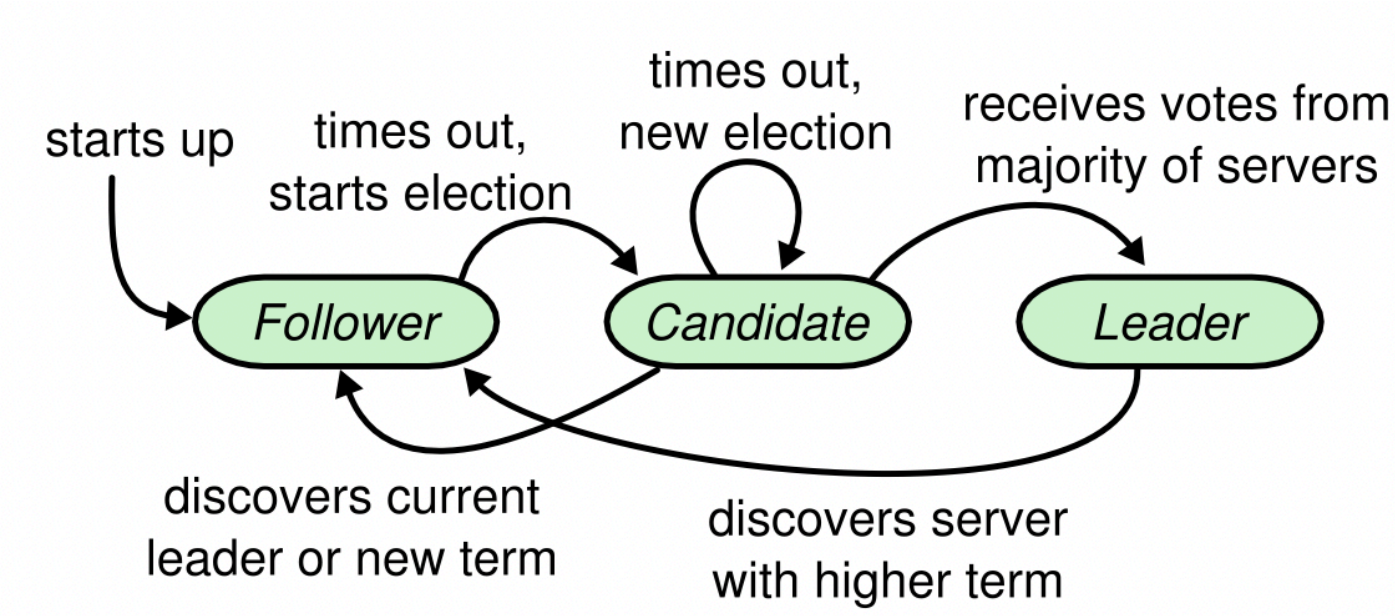
heartbeats



- 多个节点的时候，票数冲突

新一轮选举，term 大的获得了领导权

Summary



RPC(Remote Procedure Call)

- 候选人发起选举投票到跟随者或者候选人
- 领导者发起 RPC 到跟随者：1. 日志追加 2. 心跳通知

Reference

1. https://raft.github.io/
2. http://thesecretlivesofdata.com/raft/

- https://github.com/Viaxiz/homework

谢谢大家！