

# Raft简介

分布式系统一致性协议

周敦飏

2016-07

# 讨论范围

- 分布式系统一致性
- 如何解决此类问题
- 通俗易懂Raft讲解
  - 动画介绍
  - 核心行为
- 要点总结

# —致性问题

- 分布式系统一致性
  - 安全性 (safty)
  - 活性 (liveness)
- Replication特性系统
  - 分布式系统 (CAP理论)
  - 日志或者数据的复制

# Raft特性

- 对象：Replication系统
- 条件：大多数Node存活即服务可用
- 容忍：失败模式，容许数据延迟和丢失
- 主观：算法设计易懂，便于生产实现

# Raft概念

- Follower (追随者 F)
- Candidate (候选人 C)
- Leader (领导者 L)
- Term (时期) 时间轴上阶段
- Node (节点) 泛指replication特性服务
- Repli-State-Machine (复制状态机)

# Raft概述

- Leader election
  - 选择节点集合中的某一个作为集群leader
  - 探测崩溃，选举出新的leader
- Log replication
  - 从clients接收commands，应用操作到系统
  - 写副本数据到其他的节点
- Safety
  - 只有最新的认可数据的server才能是leader

# Raft特点

- 强化Leader角色概念
- Leader选举
- 角色状态转换

# 选举动画

- <http://thesecretlivesofdata.com/raft/>



# 流程分析

- Leader选举

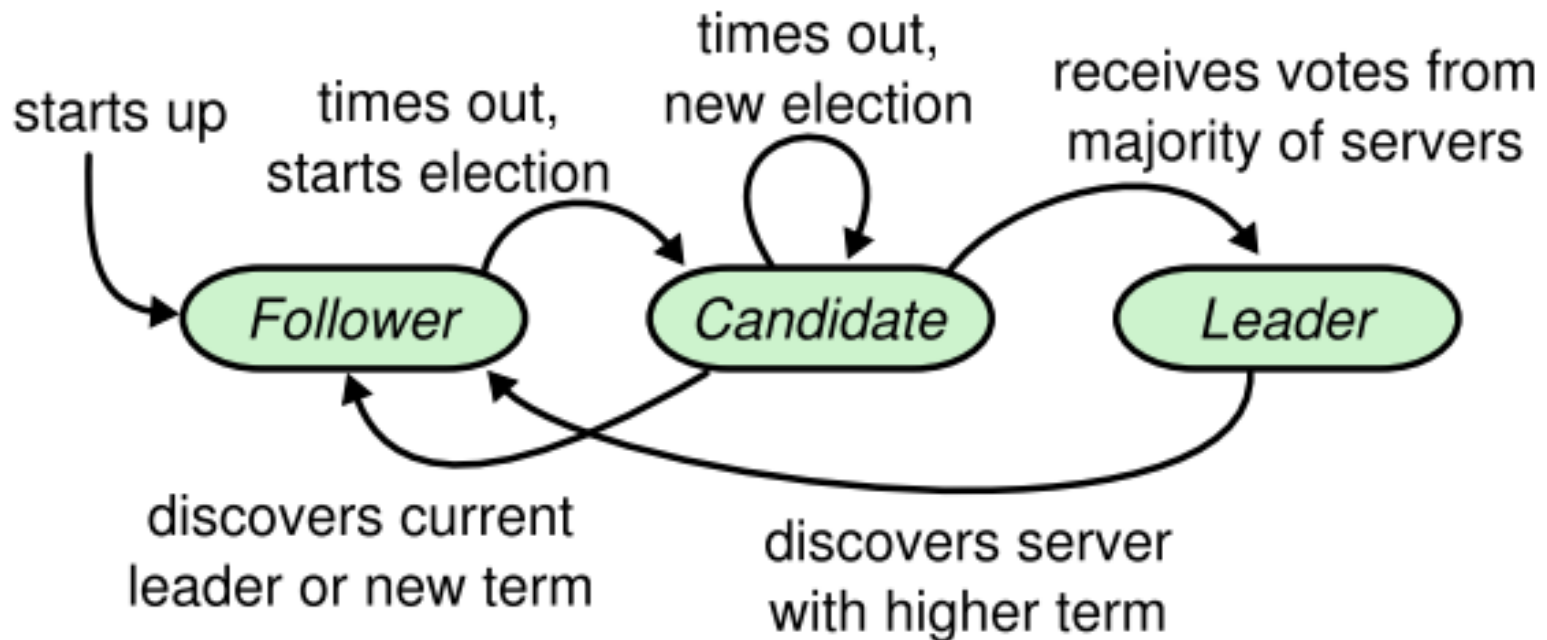
1. 初始都是follower
2. 收不到leader或者candidate的请求（选举超时）开始触发选举
3. F增加他的term num并且转变为候选者状态
4. F投票自己并且请求其他投票自己
5. F结果有三类：

# 流程分析

- Leader选举-选举结果
  1. 得到大多数票赢得选举
  2. 其他F成为leader
  3. 没有获胜者

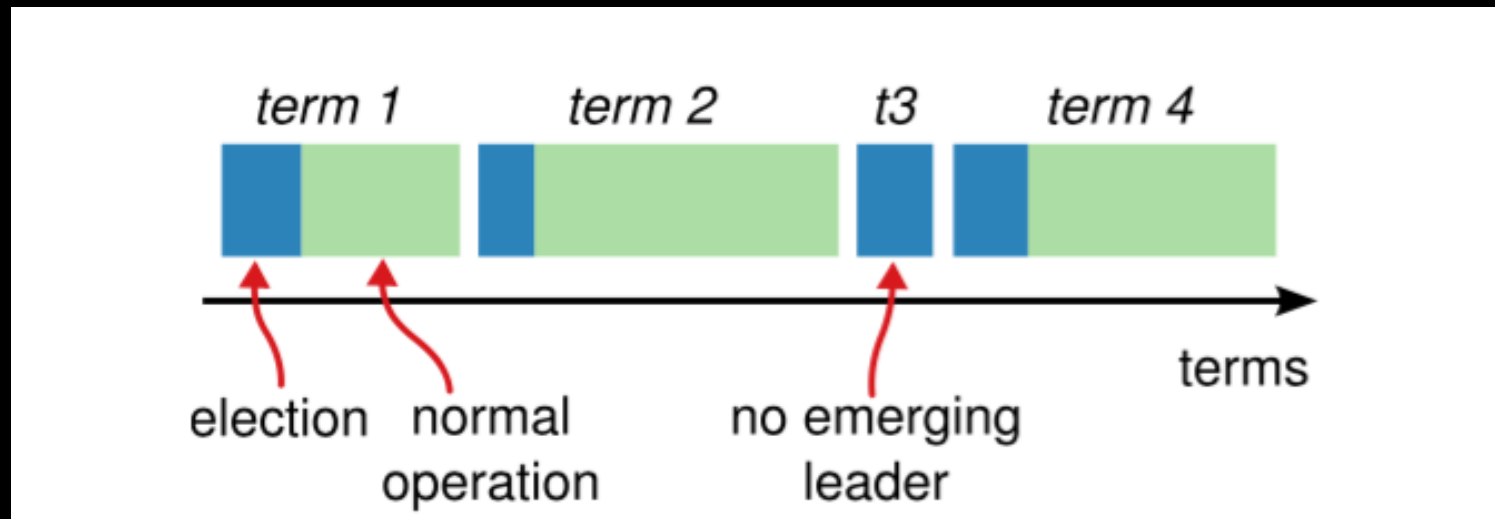
# 选举约定

- 所有的node在同一个时间阶段（term）只会是三种状态中的之一



# 选举约定

- Term是一个不定的时间长度，不会重叠，每个term最多一个leader，可能没有，成功选举后leader带领cluster直到term结束



# 数据复制动画

- <http://thesecretlivesofdata.com/raft/>

# 数据复制

- Log Replication
  1. Client发送数据请求
  2. Leader接收到请求
  3. L发出append请求（记录数据）
  4. L收到大多数回复，返回给client状态
  5. L下个心跳给定commit给follower
  6. 如果传递失败会leader会不断重试

# 流程约定

- Log entry (数据) 特性
  1. Log index
  2. Term num
  3. Command
- Log持久化存储
- Entry大多数存储成功则为committed

# 流程约定

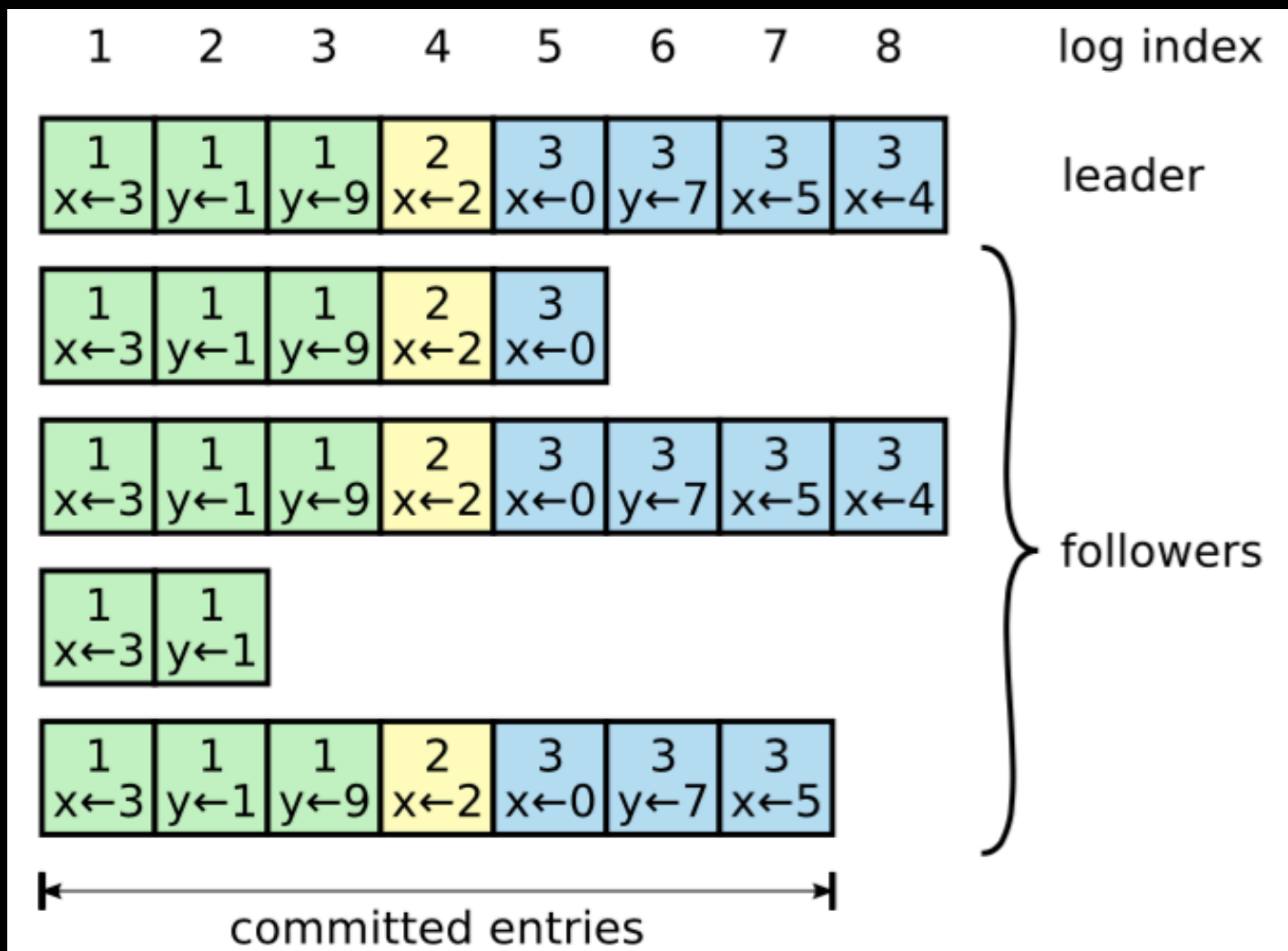
- Index和term相同那么之前的entry都是相同的
- 某个entry committed 那么之前的都是committed



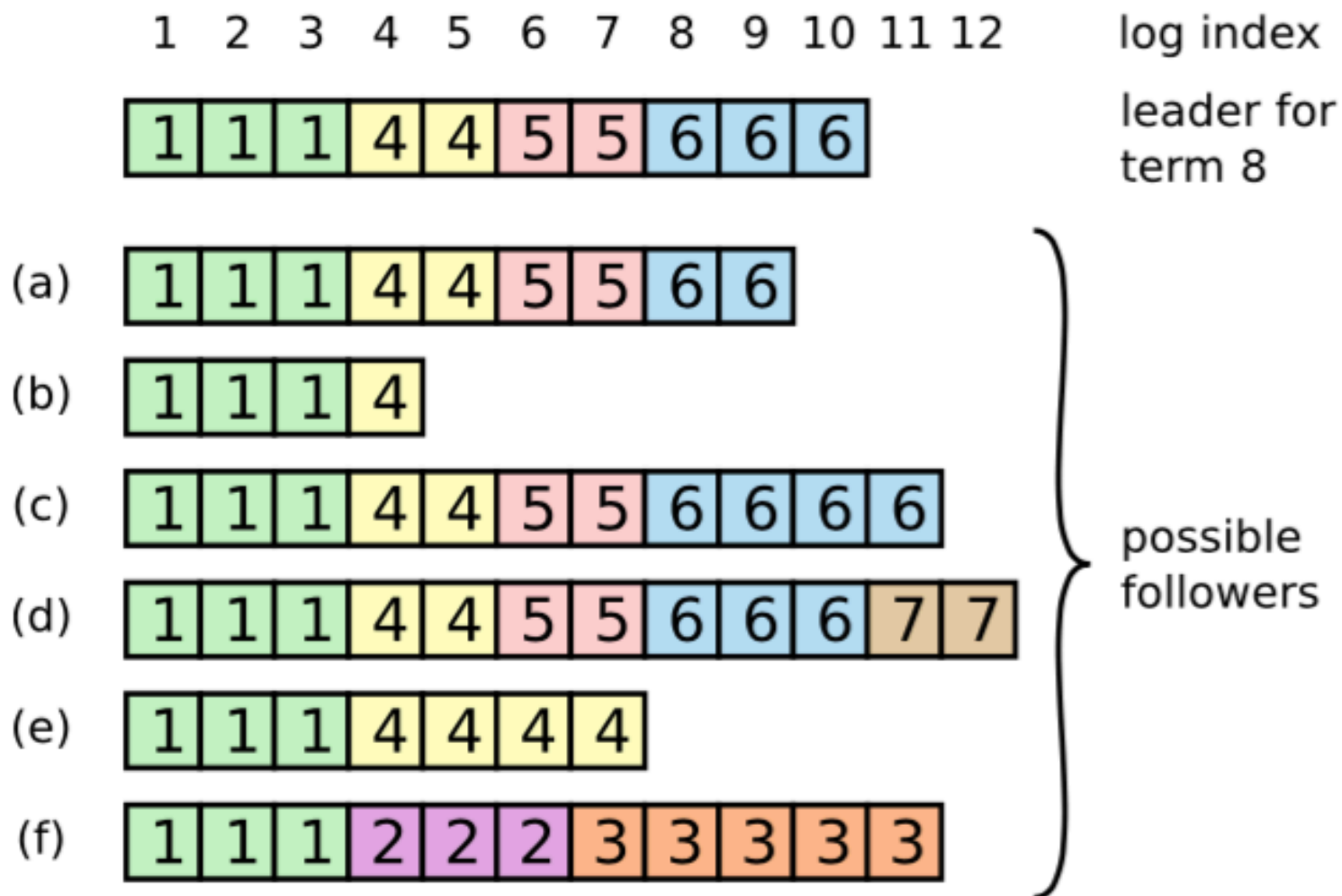
# 一致性检查

- 匹配检查：Append操作带上上一个entry的index和term
- 迭代重试：如果不匹配，重试前一条，直到成功

# 选举示例



# 同步数据实例



# 核心总结

- Leader election
  - 心跳和超时来探测崩溃
  - 随机超时时间简化选举投票
  - 每个时期最多一个被选举
- Log replication
  - 从clients接收commands, 应用到系统
  - 写副本数据到其他的节点, 覆盖不一致信息
  - 内建的一致性检查, 简化数据不同的处理方式
- Safety
  - 只有选举出来的leaders才有所有的提交数据记录
  - 只有最新提交数据的node才能是leader

# Q&A

- 演示: <http://thesecretlivesofdata.com/raft/>
- Paper: <https://raft.github.io/raft.pdf>