

MongoDB 中的 Raft 一致性协议

什么是“复制” Replication

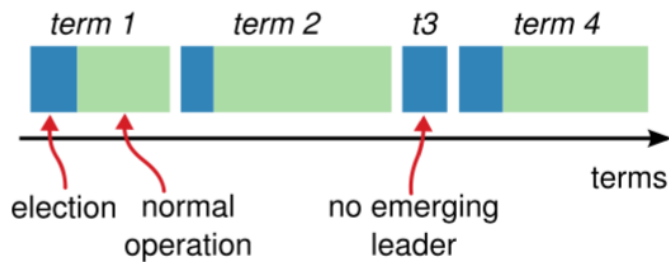
- 目的：容错，高可用性；而不是分布式读取
- 模型：复制状态机 Replicated State Machine
 - 相同的初始状态 + 相同的更新日志 = 相同的最终状态
- 保证：如果一个写操作被复制到大多数结点，那么之后只要任意大多数结点可用，这个写操作都不会受影响，称作“已提交”的写操作 - Committed writes

基于领导的复制 Leader-based Replication

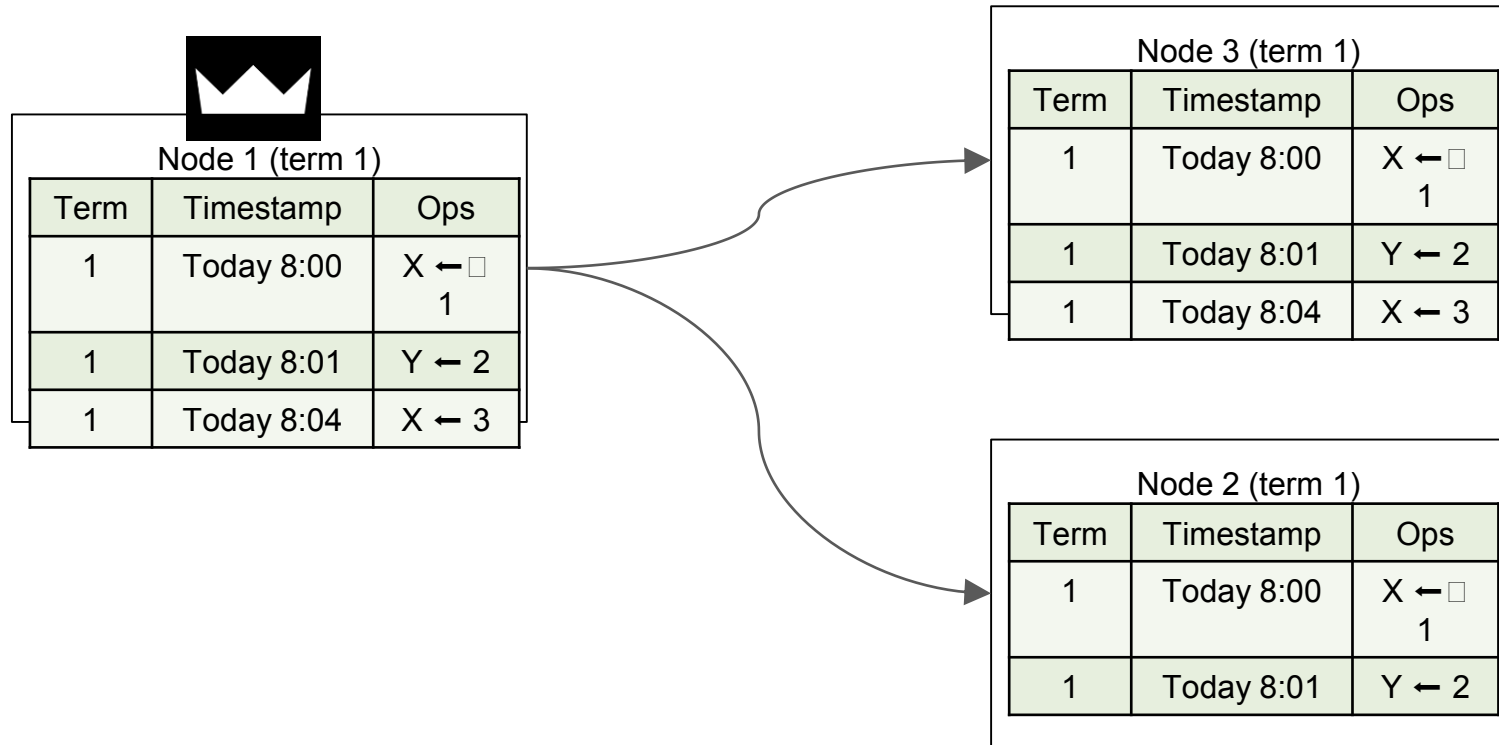
- 选举 Election
 - 从所有可用结点中选择一个**主结点**为客户提供读写服务
- 日志复制 Log Replication
 - 从 *主结点* 向 *从结点* 复制数据
 - 与 Raft 中的 *主结点* 向其他结点推送不同，*从结点* 从 *主结点* 拉取数据

Raft in MongoDB - 选举

- “Term” 任期
 - 用来区分不同的选举**尝试**，选举可能成功或失败
- 什么时候决定发起选举？
 - 生存信息 Liveness information & 计时器 Timer
 - 选举超时包括一定随机性，减少平票的可能性




任期 Term 与操作日志 Oplog



任期 Term 与操作日志 Oplog

Node 1 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3
2	Today 8:10	Z ← 0



Node 3 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3
2	Today 8:10	Z ← 0

Node 2 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

选举规则 #1

- 同一任期 (Term), 只为一个候选人投票
- 一个任期内, 得到大多数投票, 则成为主结点

Node 2: 我想在 term 2 成为主结点。投我一票吧？

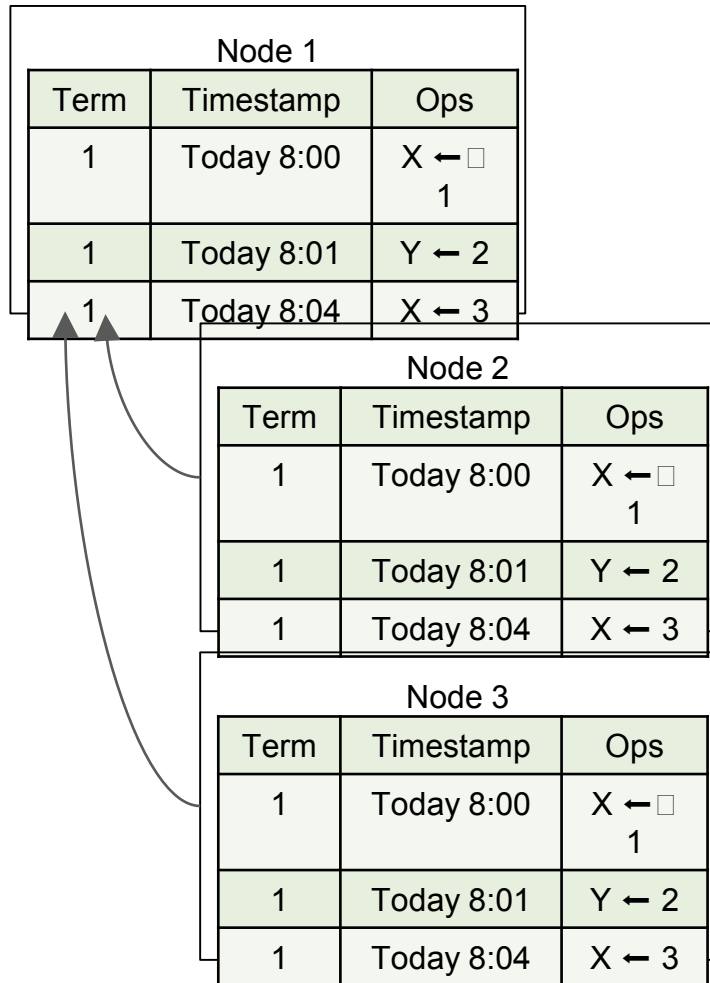
Node 1: 好, 因为我当前的 term 是 1, 我会把 term 更新到 2。

Node 3: 我想在 term 2 成为主结点。投我一票吧？

Node 1: 不行了。我已经在 term 2 投过票了。
等你 term 更高的时候再来吧。

几秒钟后...

Node 3: 不知道为什么他们都不投我票。哦, 我收到了 Node 2 发来的心跳包, 他成了 term 2 的主结点了。好吧, 我放弃了。



选举规则 #1

- 同一任期 (Term), 只为一个候选人投票
- 一个任期内, 得到大多数投票, 则成为主结点

❑ 可以为更高的 term 投票

❑ 但不能为更低或者相同的 term 投票

Node 2: 我想当 term 5 的主结点, 投一票吧?

Node 3: 能不能给我的 term 4 投一票?

Node 1: 呃... 我现在的 term 只有 1, 不过把 term 更新到 5 也无所谓。行, 投你一票!

Node 1: 我现在 term 是 5。对不起, 我不能给比我的 term 还低的 term 投票了。

另外, 我保证不给小于等于 5 的 term 投票了。

选举超时的例子

Node 1 在 term 2 从 Node 2 和他自己各拿到一票

Node 1 (votes: 2)	
Term	Timestamp
1	Today 8:00
1	Today 8:01
1	Today 8:04

Node 2	
Term	Timestamp
1	Today 8:00
1	Today 8:01
1	Today 8:04

Node 3	
Term	Timestamp
1	Today 8:00
1	Today 8:01
1	Today 8:04

Node 4 在 term 2 从 Node 5 和他自己各拿到一票

Node 4 (votes: 2)	
Term	Timestamp
1	Today 8:00
1	Today 8:01
1	Today 8:04

Node 5	
Term	Timestamp
1	Today 8:00
1	Today 8:01
1	Today 8:04

选举规则 #2

- 只有当候选人的操作日志至少跟投票人一样新时，才投赞同票。

Node 1 (term 1)		
Term	Timestamp	Ops
1	Today 8:00	X ← □ 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2: 为我的 term 2 投一票吧？
我的日志包含 term 1 中今天 8:01 的记录。

Node 1: 不行，还没我新呢。

Node 2 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← □ 1
1	Today 8:01	Y ← 2

Node 3 (term 1)		
Term	Timestamp	Ops
1	Today 8:00	X ← □ 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

给 Node 3 的请求也是一样，被否决。

选举规则 #2

- 只有当候选人的操作日志至少跟投票人一样新时，才投赞同票。

Node 1 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2

Node 3 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

选举规则 #2

- 只有当候选人的操作日志至少跟投票人一样新时，才投赞同票。

Node 1 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 3: 求在 term 3 投我一票！我知道 term 1 里发生在今天 8:04 的操作。

Node 1: OK，因为我也只知道这么多。

Node 3 (term 3)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2 (term 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2

Node 3: 求在 term 3 投我一票！我知道 term 1 里发生在今天 8:04 的操作。

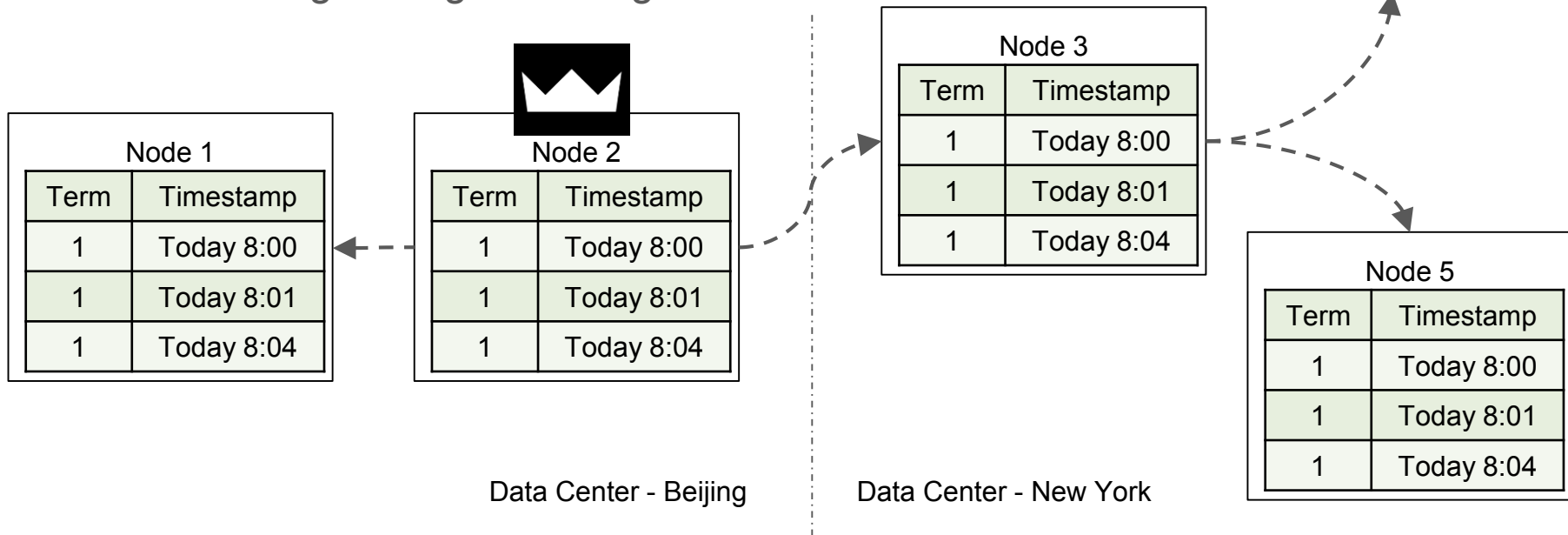
Node 2: OK，感觉你知道好像还多一点。

Raft 在 MongoDB 中的扩展

- Chaining 从结点链
- Priority / Preferred Primary 主结点优先级
- Primary Handover 主结点移交

扩展 - 从结点链 Chaining

- 使用 tailing query 拉取而不是推送
- 可配置 : `cfg.settings.chainingAllowed = false`



扩展 - 从结点链 Chaining

- 优点
 - 减少主结点负载
 - 支持不投票的从结点
 - 可投票的从结点只能从其他 可投票的从结点的复制数据
- 缺点
 - 叶子从结点报告进度时，延时会增加。

扩展 - 优先级

- 什么是优先级，为什么需要它？
- 建立在 Raft 协议之上
 - 老的选举协议把它作为选举协议的一部分。
 - 遵守相同的两个基本选举规则。
 - 只考虑自己和当前主结点的优先级，其他结点的优先级不决定选举与否。

主结点优先级的例子



Node 1 (priority: 1)

Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2 (priority: 2)

Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

主结点优先级比我低。
我决定 20 秒之后开始新一轮选举！

Node 3 (priority: 3)

Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

主结点优先级比我低。
我决定 10 秒之后开始新一轮选举！

主结点优先级的例子

Node 1 (priority: 1)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 3: term 2 投我一票？

Node 1: 啊...虽然我是主结点，但是有人term 超过我了，还是退下来吧。

Node 1: OK，Node 3，看好你哦！

Node 3 (priority: 3)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2 (priority: 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3


Node 3: term 2 投我一票？

Node 2: 行。

Node 2: 本来还想等 20 秒自己当主结点呢，有人比我快了，那我再等 20 秒好了。你行你就上。


扩展 - 避免不必要的回滚

- 正在进行的项目，目的是尽力减少老的主节点上不必要的回滚。
- 比如：高优先级节点赢得新一轮选举时，老主节点上可能有没有 commit 的数据



Node 1 (priority: 1)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2
1	Today 8:04	X ← 3

Node 2 (priority: 2)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2



Node 3 (priority: 3)		
Term	Timestamp	Ops
1	Today 8:00	X ← <input type="checkbox"/> 1
1	Today 8:01	Y ← 2

主节点优先级比我低。
我决定 10 秒之后开始新一轮选举！

总结

- MongoDB 选举和复制协议基于 Raft 协议，正确性得以保证。
- 在 Raft 基础上，我们扩展了选举和复制协议，方便使用和运维。

Queries & Results