

Raft

Raft将一致性问题分解为三个子问题：

- (1). 领导选举：1.旧Leader宕机选举新Leader
- (2). 日志复制：Leader接受日志，复制到其它节点并保持一致
- (3). 安全性：状态机的安全，某一节点应用某个日志条目到状态机，其它节点不能在此条目应用不同分支。
此外还涉及额外选举机制上的限制

① 领导选举：

(1) Follower (追随者)：

1. 响应候选人或领导者的 RPC 请求

2. 选举超时，都没有收到现任 Leader 的 RPC 请求，也没有给候选人投票，自己则转变为 Candidates

(2). Candidates (候选人)

1. 自增当前任期号，给自己投票，重置超时计数器，并发 RequestVote RPC 给其他所有服务器

2. 收到大多数服务器投票，成为 Leader

3. 收到 Leader 的 AppendEntries RPCs，转变为 Follower

4. 选举超时，重新投票

(3). Leader：

1. 发送空的 AppendEntries RPC 给每个服务器，表示 Leader 已选举出

2. 跟据每个 Follower 的日志情况，发送 AppendEntries RPC 给 Follower 同步日志

(4). RequestVote RPC

传入参数:

term	候选人的任期号
candidateId	请求选票的候选人的 Id
lastLogIndex	候选人的最后日志条目的索引值
lastLogTerm	候选人最后日志条目的任期号

返回值:

term	当前任期号，以便于候选人去更新自己的任期号
voteGranted	候选人赢得了此张选票时为 true

若接收方的 current term > 传入的 term

或接收方的 term == 传入的 term，但候选人最后日志任期号及索引不比接收方新，则拒绝给候选人投票

否则给候选人投票

注意: Follower 只能投一票，且遵循先到先投票的规则

(5). 得到大多数Follower 投票的 Candidate 成为 leader，否则重新选举

② 日志复制

AppendEntries RPC

传入参数:

term	领导人的任期号
leaderId	领导人的 Id，以便于跟随者重定向请求
prevLogIndex	新的日志条目紧随之前的索引值
prevLogTerm	prevLogIndex 条目的任期号
entries[]	准备存储的日志条目（表示心跳时为空；一次性发送多个是为了提高效率）
leaderCommit	领导人已经提交的日志的索引值

返回值:

term	当前的任期号，用于领导人去更新自己
success	跟随者包含了匹配上 prevLogIndex 和 prevLogTerm 的日志时为真

Leader 的 AppendEntries RPC 会包含最新日志条目的任期和索引，若 Follower 与 leader 日志条目不相同，则 leader 后续的 RPC 请求会递减 nextIndex（通过 prevLogIndex 参数返回）直到与 follower 当前日志条目匹配。

(1). 若 Follower 与 leader 匹配的日志条目还有多条不同的时，则丢弃这些日志条目，并同步 Follower 日志。

(2). 若 (1) 中所述没有多余条目，则直接同步 Follower 日志。

注：leader Commit 表示，leader 已提交的最新日志。Follower 收到后即表示若有该日志且未提交，则可提交了（包括前面未提交的）。

③ 安全性

(1). 选举中的安全性：见①中(4)，眼看两个prelog与follower
目前最新日志比哪个较新，从而投票者拒绝投票

(2). 日志复制中的安全性：

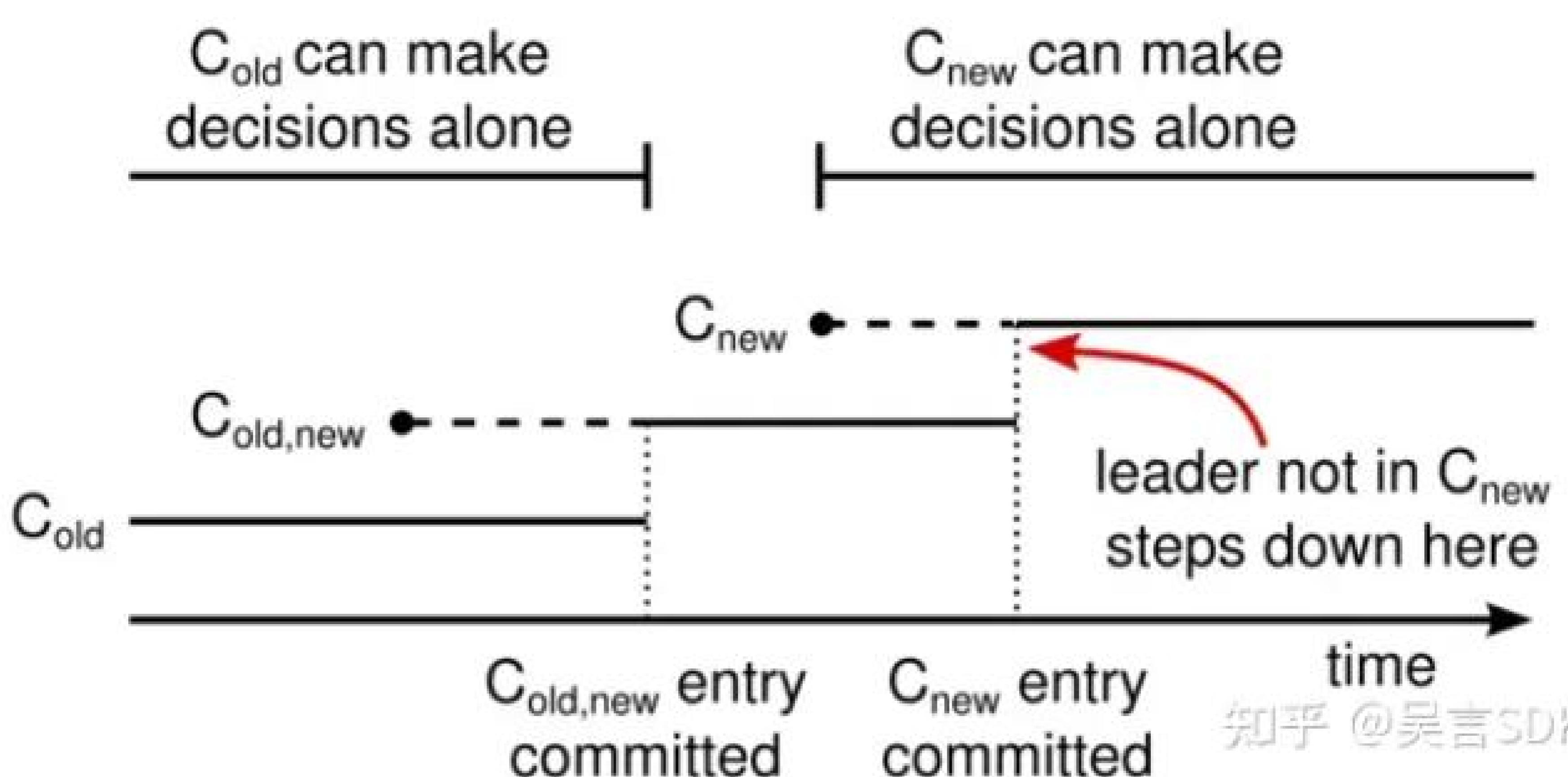
1. Leader 的 AppendEntries RPC 应包含 follower 所有
缺失的日志条目，而不是一个一个日志的发

2. 不在 leader 主前任期内的日志组不提交，而是等到
在 leader 主前任期内的日志条目提交时，才把旧的日志
组一起提交（即 leader term = 4，不选择提交

term < 4 且未提交的日志，而是等到 term = 4 的日志提
交时，才选择将 term < 4 的未提交日志一起提交）

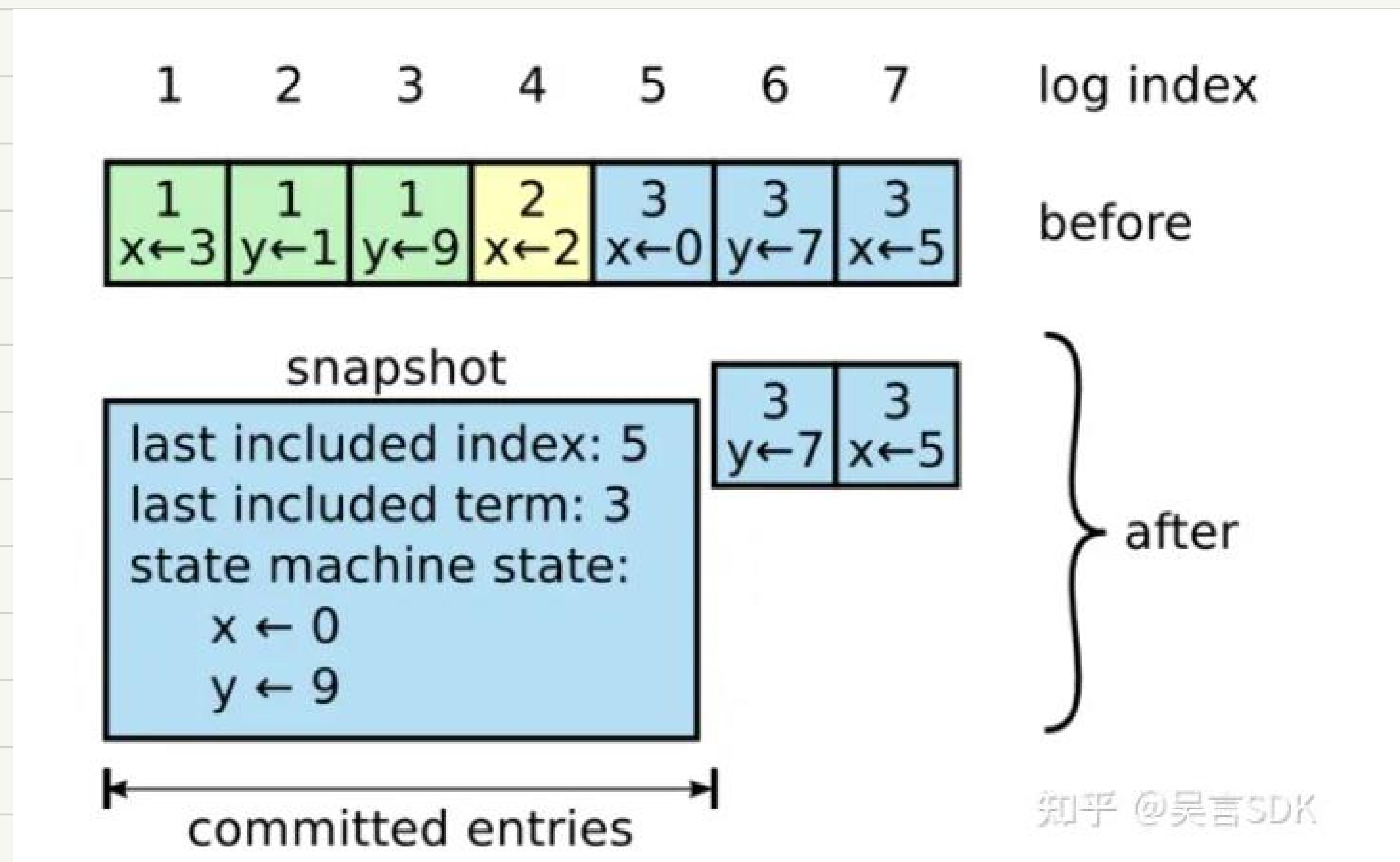
④ 集群成员变化：采用两阶段提交

- (1). 当有新节点加入集群时，必须同步告知新加入的节点
- (2). 由旧集群的 leader 发 $C_{old,new}$ entry 给集群所有节点，
若此时 $C_{old,new}$ 未提交，leader 客机了，则 old 端点只要超过
半数 old 节点投票则当选 leader，而 new 节点只需 old 节点，
若 new 节点投票才能当选 leader (若无客机，需 old 及 new 大多数派向树
大数 投票) (若无客机，需 old 及 new 大多数派向树 投票)
- (3). 集群 leader 提交 $C_{old,new}$ entry，并发 C_{new} entry
给集群所有节点 (若无客机，只需 new 节点半数复制到 new，即可提交)
若此时 C_{new} entry 未提交，leader 客机了，则 new 端点只要
超过半数 new 端点投票，则当选 leader，而 old 端点需要
集群半数端点投票才当选 leader
- (4). 集群 leader 提交 C_{new} entry，完成集群成员变化



⑤ 日志压缩:采用Snapshotting(快照)压缩

根据日志，又保留状态机中组件最新的状态，而把日志中间更新的一系列日志丢弃，只保留最新的日志



leader通过快照压缩后，直接将快照发给follower
而不需要 follower 重复整个过程