Raft 选举机制

节点角色

在介绍 Raft 的选举机制之前，需要先介绍一下 Raft 中规定的角色：

1. Leader，领导者，Raft 是强领导模型，一切以领导者为准
2. Candidate，候选者，向其他结点发出请求投票的 RPC 消息，如果其获取了集群中大多数的投票（超过半数），就可以正式成为领导者。
3. Follower，跟随者，默默处理着来自于领导者的心跳等信息，如果Leader 的心跳信息超时，则 Follower 会主动推荐自己为候选人。

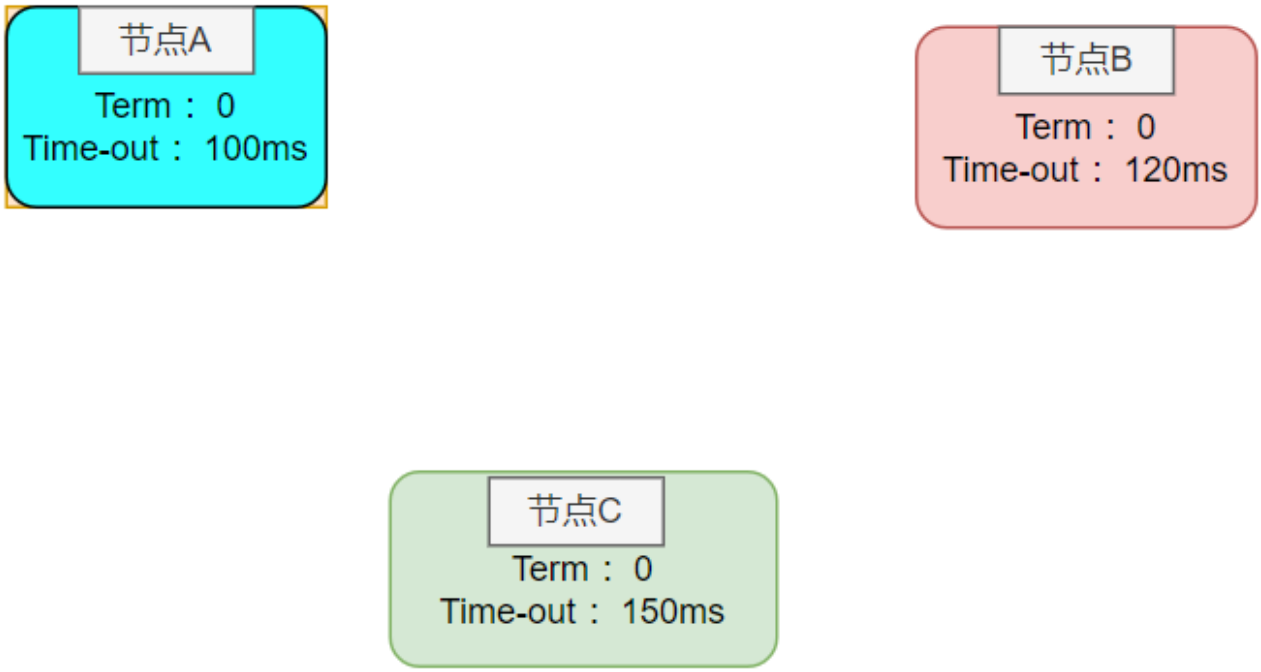
选举规则

对于选举，我们肯定不陌生，就好像班上选班长一样，都是根据一定的选举规则选举的，那么接下来我们就来看看 Raft 中的选举规则：

1. Raft 是强领导者模型，任何时候都只允许存在一个且唯一一个的 Leader，即选举名额只有一个；
2. Leader 会周期性地向所有跟随者发送心跳信息，阻止 Follower 发起新的选举；
3. Follower 在指定时间内没有收到 Leader· 的心跳信息，就认为集群中已经没有 Leader 了，则推荐自己为 Candidate， 发起新一轮的选举；
4. 在一次选举中，获得大多数选票的 Candidate 将成为 Leader
5. 在一次选举中，对于同一任期的投票，一个结点只有一张选票（面对多个同任期的请求投票信息，采用 FIFO 的方式处理。
6. 当任期编号相同时，日志完整性高的跟随者，拒绝投票给日志完整性低的 Candidate

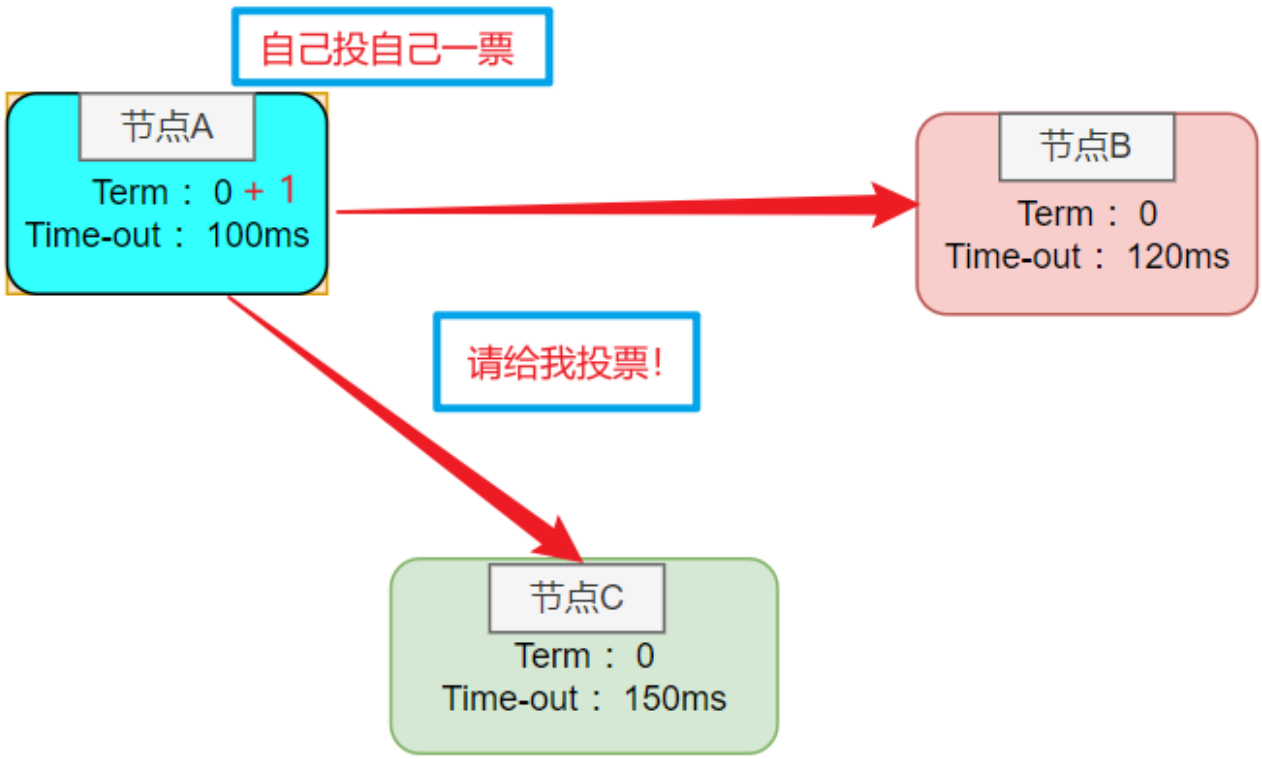
选举流程

有了这些前置知识之后，我们就可以来看看一次完整的选举流程了。



一开始，集群中所有节点都是 Follower，随机超时时间。

按照上图，由于没有 Leader 发送心跳信息，所以节点 A 将会是第一个超时的，这时候，节点 A 会增加自己的任期编号 （term++），并推举自己为候选人，**先给自己投上一张选票**，然后才向其他节点发送请求投票 RPC 消息，请求它们选举自己为 Leader。 `此时的结点 A 就是 Candidate~`

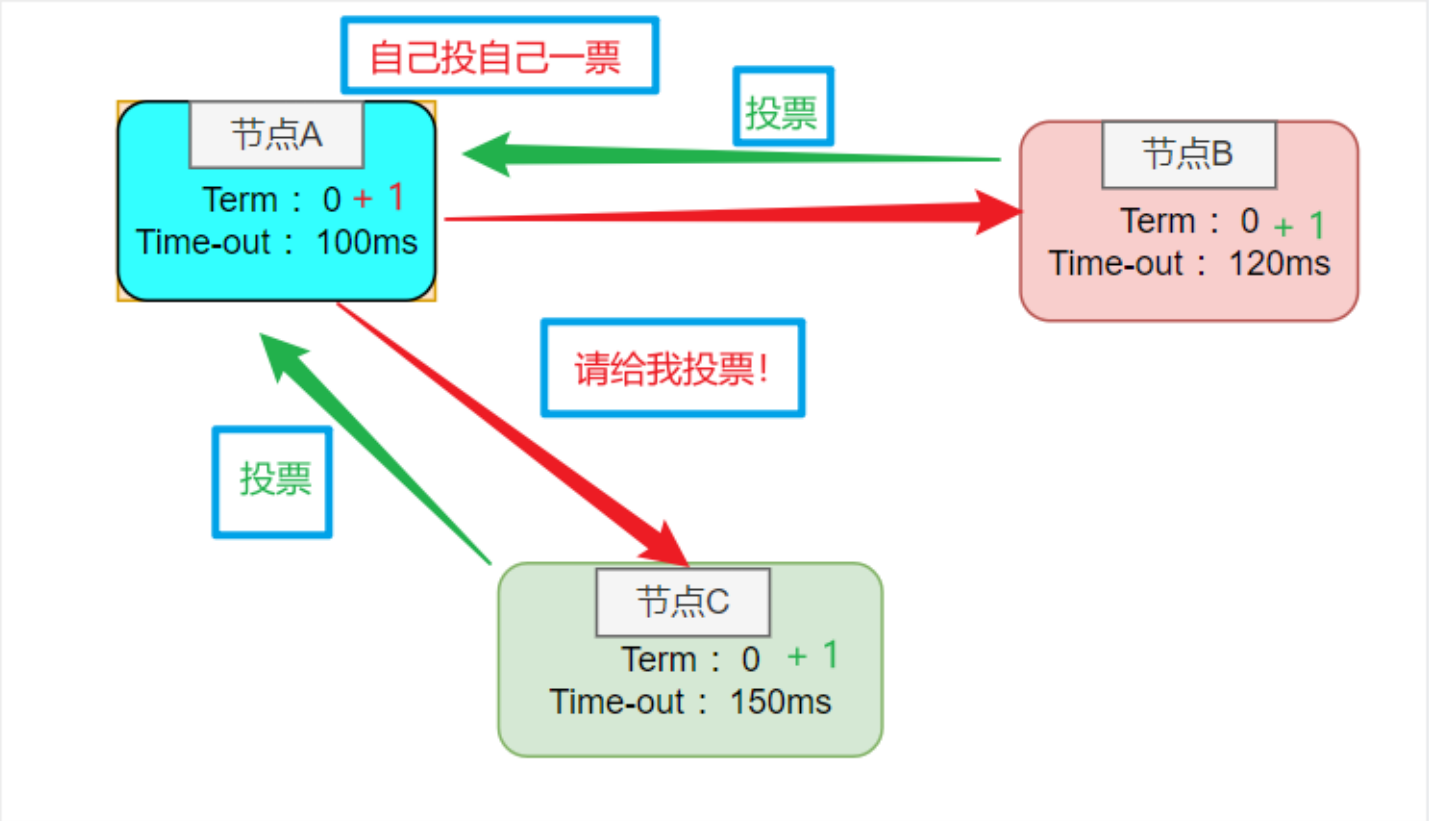


其他节点在收到结点 A 的消息之后，会根据投票规则来决定是否给 A 投票！

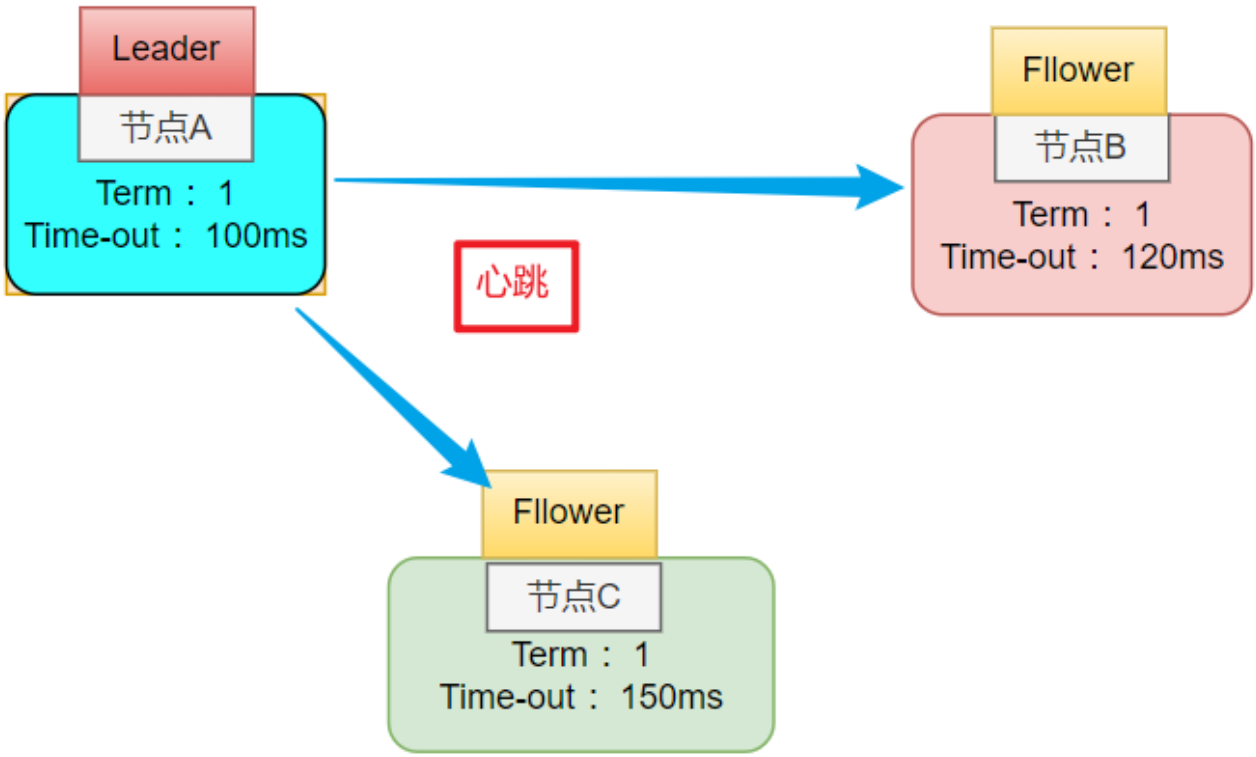
投票规则：

1. 在一次选举中，对于同一任期的投票，一个结点只有一张选票（面对多个同任期的请求投票信息，采用 FIFO 的方式处理。
2. 当任期编号相同时，日志完整性高的跟随者，拒绝投票给日志完整性低的 Candidate

其他节点的 term 为 0，而 A 的 term 为 1 ，而且在 term = 1 任期期间它们还没有投过票来，所以它们会把选票投给 A 结点，`并增加自己的 Term 编号`



此时结点 A 已经获得了集群中的大多数选票，根据选举规则，结点 A 将晋升为 Leader。



节点间的通讯方式

两类 RPC：

1. 请求投票（RequestVote）RPC，是**由候选人**在选举期间发起，通知各节点进行投票；
2. 日志复制（AppendEntries）RPC，是**由领导者发起**，用来复制日志和提供心跳消息。

说说你对 Term （任期）的理解

Term 是由一个单调递增的数字来标识的，它所代表的任期并不是指任期时间，而是一个任期编码，会影响到 Leader 的选举和请求的处理。

1. 在 Raft 算法中约定，如果一个候选人或者领导者，发现自己的任期编号比其他节点小，那么它会立即恢复成跟随者状态。比如分区错误恢复后，任期编号为 3 的领导者节点 B，收到来自新领导者的，包含任期编号为 4 的心跳消息，那么节点 B 将立即恢复成跟随者状态。
2. 还约定如果一个节点接收到一个包含较小的任期编号值的请求，那么它会直接拒绝这个请求。比如节点 C 的任期编号为 4，收到包含任期编号为 3 的请求投票 RPC 消息，那么它将拒绝这个消息。

说说你对随机超时时间的理解

超时时间：指 Follower 节点需要在规定时间内收到 Leader 的心跳消息，如果没有收到，则超时；这里面所说的规定时间就是超时时间。

随机超时时间：每个 Follower 节点的超时时间不同，而且都是随机的；`这个随机时间不仅仅是指 Follower 等待 Leader 心跳信息超时的时间间隔 ，也是指当没有候选人赢得过半票数，选举无效了，这时需要等待一个随机时间间隔`

随机超时时间的作用：

> 可以使得超时时间都分散开来，在大多数情况下只有一个服务器节点先发起选举，而不是同时发起选举，这样就能减少因选票瓜分导致选举失败的情况