实验 3　Raft 协议的实现之一

1. 实验目的

通过实现 Raft 论文中提到的 Leader 选举以及追加日志条目的功能，加深对一致性协议的原理理解并编程实现。

2. 实验准备

a) 实验源地址：http://nil.csail.mit.edu/6.824/2022/labs/lab-raft.html

b) 对理解 Raft 可能有帮助的可视化动画：http://thesecretlivesofdata.com/raft/

3. 实验步骤

3.1 阅读原文 http://nil.csail.mit.edu/6.824/2022/labs/lab-raft.html 更好地理解实验内容。

3.2 完善 src/raft/raft.go 来实现 raft

该文件中包含了代码框架、以及如何发送、接收 RPC 的示例，src/raft 下的其他文件是为测试服务的。

其中，raft.go 中提供了一下几个接口供测试程序以及最终的 KV 存储服务使用（本课程实验暂不涉及）：

- rf := Make(peers, me, persister, applyCh)，通过 Make 来创建 Raft 节点，peer 参数表示该节点的网络标识符数组（具体参见代码）；me 代表该节点在 peers 数组中的下标。
- rf.Start(command interface{}) (index, term, isleader)，将命令追加到日志副本中。Start()不需要等待日志追加完成即可返回。
- rf.GetState() (term, isLeader)：供测试程序使用，获取当前节点的 term，以及是否为 Leader。
- type ApplyMsg，该服务希望你将每个新的日志条目，封装为 ApplyMsg，发送给 Make 函数中的 applyCh 参数（channel）。

3.2.1 Part 2A：实现 Leader 选举

2A 主要和 Leader 的选举机制有关，对应 Raft 论文的 Figure 2。

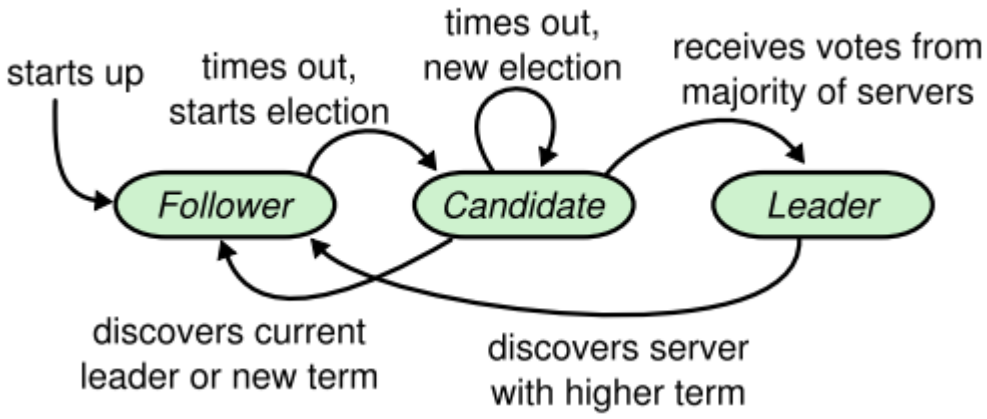
正常情况下，Leader 会周期性地向所有 Follower 发送心跳，告知自己的 Leader 身份。

触发选举：当 Follower 一段时间（election timeout）没有收到 Leader 心跳时，则认

为 Leader 挂掉，发起选举。

Follower 发起选举：将自己的 term + 1，表示进入下一个 term；将自己的状态转换为 Candidate；为自己投票，并同时向集群中的所有服务器发起 RequestVote 的 RPC。

选举结束条件：赢得选举；其他 Follower 成为 Leader；选举超时。



提示：

a) 按照论文的 Figure 2，只需要关心发送和接收 RequestVote RPC、与选举相关的服务器规则以及相关的状态。

b) 将 Leader 选举的 Figure 2 状态添加到 raft.go 上的 Raft 结构中，还需要定义一个结构体来保存每个日志条目的信息。

c) 完善 RequestVoteArgs 和 RequestVoteReply 结构。

修改 Make()以创建一个后台 go 协程，该协程将在一段时间内未从其他 peers 收到请求投票 RPC 时，发送 RequestVote RPC 来定期启动 Leader 选举。如果已经有一个 Leader，或者其成为 Leader，其他 peers 就会知道哪个节点是 Leader。

实现 RequestVote() RPC 函数，以便服务器投票给其它节点。

d) 提前定义 AppendEntries RPC 结构（尽管可能还不需要所有参数）以实现心跳检测，并让 Leader 定期发送。AppendEntries RPC 函数需要重置选举超时时间，以便其它服务器当选时，该服务器不会以 Leader 的身份继续运行。

e) 确保不同 Peers 不会在同一时间选举超时，否则所有 Peers 将只为自己投票，没有人会成为 Leader。

f) 测试程序要求 Leader 发送心跳检测 RPC 的频率不超过 10 次/ 秒。

g) 测试程序要求 Raft 在旧 Leader 失败后五秒内选出新 Leader。在发生分裂投票的情况下，领导人选举可能需要多轮投票，那么则需要选择足够短的选举超时（心跳

间隔也是如此），确保即使选举需要多次轮断，也能在五秒内完成。

h) 论文第 5.2 节提到选举超时应该在 150 到 300 毫秒范围内。只有当 Leader 发送一次心跳包的远小于 150 毫秒，这种范围才有意义。由于测试将发送心跳包的频率限制在 10 次/秒内（也就是大于 100 毫秒），因此必须使用比论文 150 到 300 毫秒更大的选举超时时间，但请不要太大，因为那可能导致无法在 5 秒内选出 Leader。

i) 需要编写定期或延迟执行的代码。最简单的方法是新起一个 goroutine，在协程的循环中调用 time.Sleep()。

j) http://nil.csail.mit.edu/6.824/2022/labs/guidance.html 中对于如何开发和调试代码给出了一些建议。

k) 如果代码在通过测试时遇到问题，请再次阅读论文的 Figure 2 。

l) 不要忘了实现 GetState()。

m) 当要停止一个实例时，测试程序会调用 Raft 的 rf.Kill()。您可能希望在所有循环中执行此功能，以避免已经挂掉的 Raft 实例打印令人困惑的信息。

n) Go RPC 仅发送以大写字母为首的结构体字段。子结构体还必须具有大写字段名称（例如数组中的日志记录字段）。labgob 包会警告这一点。

o) 测试。通过 go test -run 2A 将会对 2A 部分进行测试。

```
zhuchangzhen@zhucz:~/6.824/src/raft$ go test -run 2A
Test (2A): initial election ...
  ... Passed --   3.0  3   58   14372    0
Test (2A): election after network failure ...
  ... Passed --   5.5  3  134   26666    0
Test (2A): multiple elections ...
  ... Passed --   5.6  7  486  100572    0
PASS
ok      6.824/raft      14.144s
```

3.2.2 Part 2B 实现日志追加

提示：

a) 第一个目标应该是通过 TestBasicAgree2B()。首先实现 Start()，然后按照 Figure 2，实现 RPC 函数 AppendEntries 来收发新的日志条目。

b) 实现选举限制（论文第 5.4.1 节）。

c) 在早期的 2B 实验测试中，一个无法达成一致的方法是：即使领导人还活着，也举行多次的选举。在选举计时器中找到并修复这个 bug ，或在赢得选举后不要立

即发送心跳包。

d) 代码中可能需要循环检测某些事件。不要让这些循环不间断连续执行，不然会使得服务运行变慢，最终导致测试失败。

e) 测试。通过 go test -run 2B 将会对 2B 部分进行测试，同时需要保证 2A 部分正确。

```
zhuchangzhen@zhucz:~/6.824/src/raft$ go test -run 2B
Test (2B): basic agreement ...
  ... Passed --   0.6  3   16    4546    3
Test (2B): RPC byte count ...
  ... Passed --   1.5  3   48  114350   11
Test (2B): agreement after follower reconnects ...
  ... Passed --   5.6  3  127   34391    7
Test (2B): no agreement if too many followers disconnect ...
  ... Passed --   3.4  5  216   44042    3
Test (2B): concurrent Start()s ...
  ... Passed --   0.7  3   12    3390    6
Test (2B): rejoin of partitioned leader ...
  ... Passed --   6.0  3  183   46172    4
Test (2B): leader backs up quickly over incorrect follower logs ...
  ... Passed --  18.3  5 1844 1357102  102
Test (2B): RPC counts aren't too high ...
  ... Passed --   2.1  3   40   11902   12
PASS
ok      6.824/raft      38.289s
```

4. 实验要求

基于项目提供的框架实现 Raft 2A 以及 2B 部分，并通过所有测试用例。

5. 实验报告

通过图片的形式，结合自身代码实现，分析 Raft 中 Leader 选举以及日志追加的具体流程。

6. 提交方式

将实验代码与实验报告一同打包，并以学号_姓名__DSC_Lab3 格式命名，提交到研究生信息系统的课程平台。