Lab3 raft选举说明文档

补充相关结构体

raft

```go
// A Go object implementing a single Raft peer.
type Raft struct {
    mu        sync.Mutex          // Lock to protect shared access to this peer's state
    peers     []*labrpc.ClientEnd // RPC end points of all peers
    persister *Persister          // Object to hold this peer's persisted state
    me        int                 // this peer's index into peers[]
    dead      int32               // set by Kill()

    // Your data here (2A, 2B, 2C).
    // Look at the paper's Figure 2 for a description of what
    // state a Raft server must maintain.
    currentTerm int //当前term
    voteFor     int //投票对象
    log         []LogEntry

    commitIndex int
    lastApplied int

    nextIndex  []int
    matchIndex []int

    myStatus    int           //当前raft状态
    voteTimeout time.Duration //变成candidate的超时时间，不同raft不同
    timer       *time.Ticker
    applyChan   chan ApplyMsg
}
```

RequestVoteArgs与RequestVoteReply

```go

type RequestVoteArgs struct {
    // Your data here (2A, 2B).
    Term         int //candidate's term
    CandidateId  int //candidate requesting vote
    LastLogIndex int
    LastLogTerm  int
}

// example RequestVote RPC reply structure.
// field names must start with capital letters!
type RequestVoteReply struct {
    // Your data here (2A).
    Term        int  //currentTerm, for candidate to update itself
    VoteGranted bool //true means candidate received vote
}

```

AppendEntryArgs与AppendEntryReply

```go

// 心跳结构体
type AppendEntryArgs struct {
    Term         int //leader's term
    LeaderId     int
    PreLogIndex  int
    PreLogTerm   int
    LeaderCommit int
}

type AppendEntryReply struct {
    Term    int //current term, for leader to update itself
    Success bool
}

```

是根据论文中表2进行补充的

主要函数实现

ticker()函数

这个函数是主入口，根据rf的当前状态是follower、leader、candidate来做出对应的响应，若是follower，则等待electionTimeOut，变为candidate；若是candidate，则广播选举，选出新的leader；若是leader，则定时发送心跳。

具体的实现详见代码

RequestVote()函数

分两种情况讨论，请求term小于当前term，说明是过期的请求；请求term大于当前term，则投出自己一票；若请求term等于当前term，则判断当前是否已经投过票，若已经投过票，不做任何处理，若还未投票，则投出自己一票

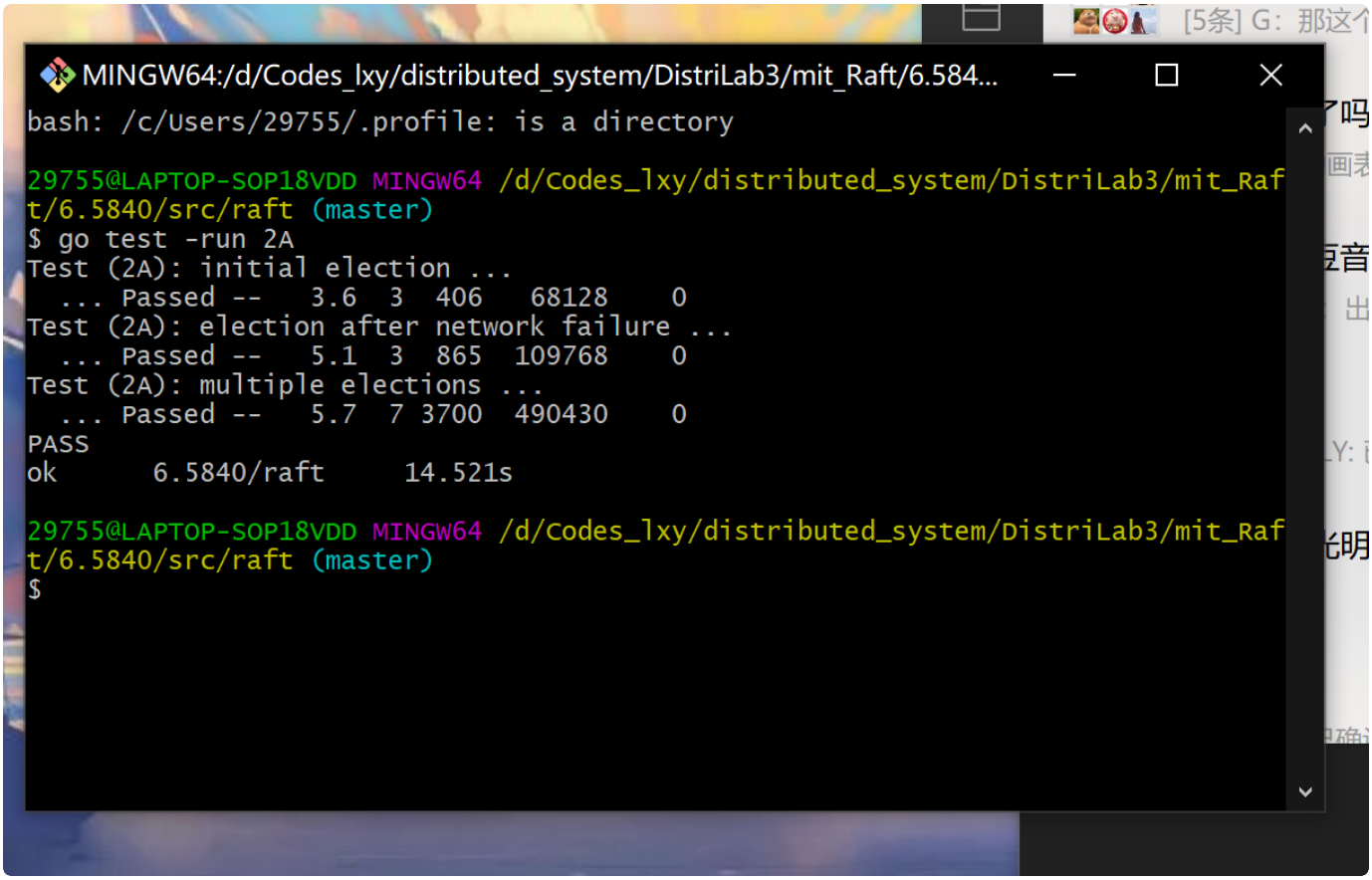
sendRequestVote()函数

若当前term为最新，则根据接收到的投票结果判断投票数是否已到达最大投票数的一半，若达到，成为新的leader，否则接收到新leader心跳，变成follower

sendAppendEntry()函数与AppendEntry()函数

逻辑比较简单，不再赘述

运行结果



测试全部通过