**RDT 实验报告**

**（GBN）**

**学 号 3020244294**

**姓 名 石子跃**

**学 院 智能与计算**

**专 业 网络安全**

**年 级 2020**

**任课教师 仇超**

**2022年 5月 1日**

### 1. 协议需求分析

#### Go-Back-N

**需求**:停等协议由于其特性导致网络利用率不高，于是需要回退N协议，在不需要确认数据包ACK的情况下仍可进行传输，增加传输效率。

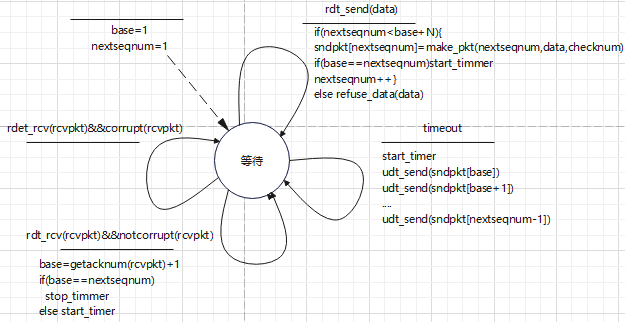
**所适用的场景**:网络质量良好丢包率偏低的网络中，因为单个分组端差错能够引起大量分组端重传，这将导致非常多的资源浪费。

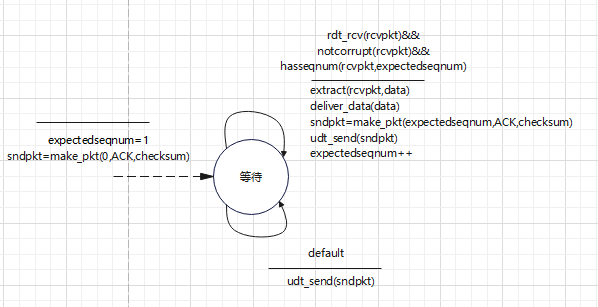
### 2. 协议的设计

#### Go-Back-N

**设计原理**：返回N协议(GO-Back -N)为了提高效率填满管道,在发送方等待确认时,应当有多个分组正在传送中.也就是说我们需要让多喝分组处于等待确认的状态,以便在发送方等待确认的同时,信道也能保持忙碌状态.返回N协议(GO-Back -N GBN)的关键是发送方能够在收到确认之前发送多个分组,但接收方只能缓存一个分组.发送方为发送出去的分组保留副本,直到来自接收方确认达到.

**FSM 图**：





### 3. 协议的实现

#### Go-Back-N

**主要数据结构**：

struct Sender

{

    int base;

    int nextseqnum;

    int N;

    float time\_out;

    struct pkt pkt\_buf[MAXBUFF];

} A;

struct Receiver

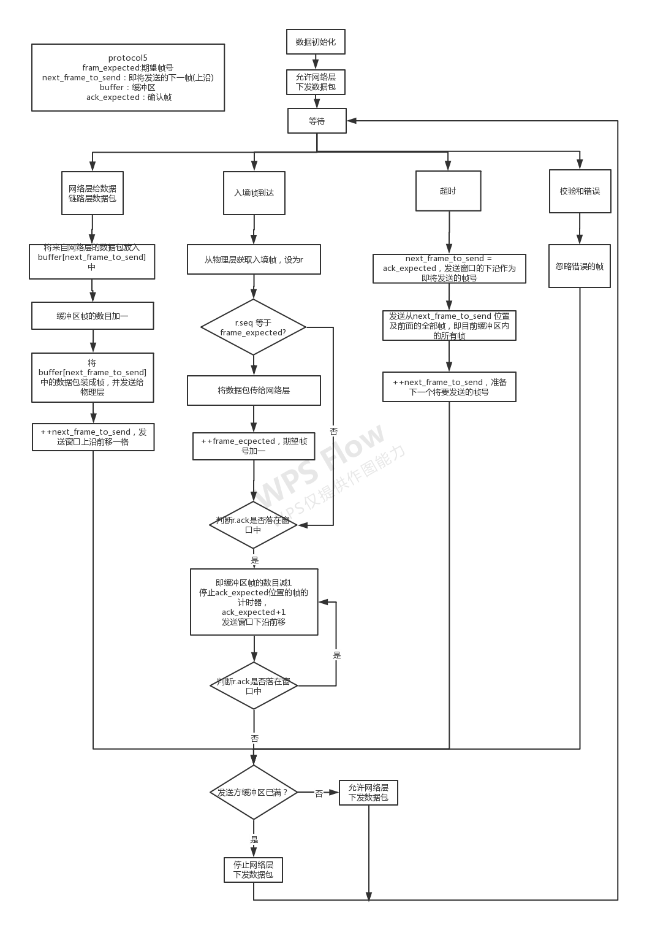
{

    int expectedseqnum;

    struct pkt last\_pkt;

} B;

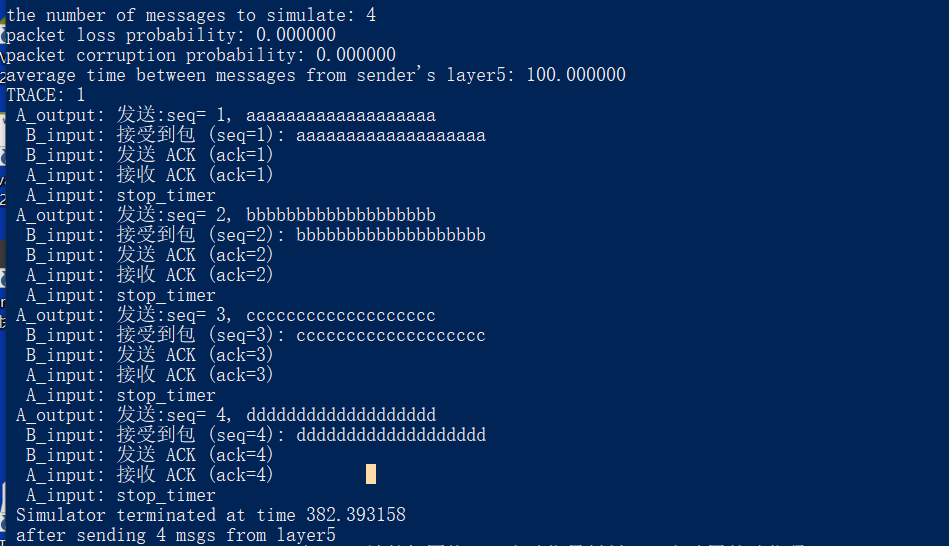
**流程图**：



### 4. 协议功能测试

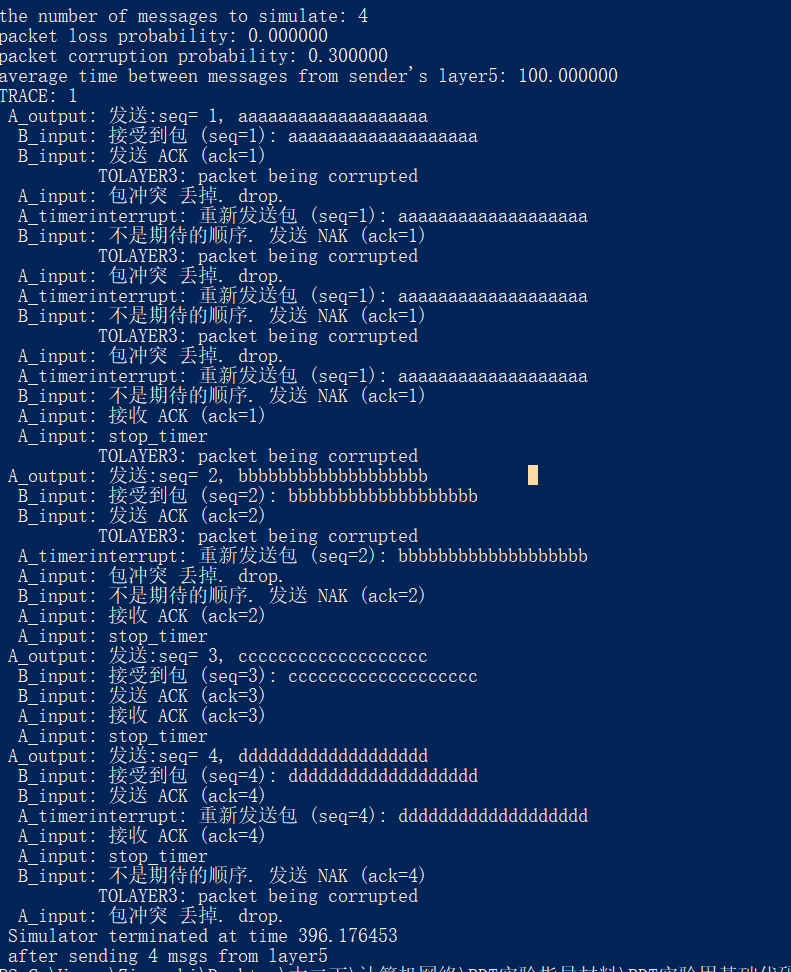
#### Go-Back-N

1) 无 error 无 loss；

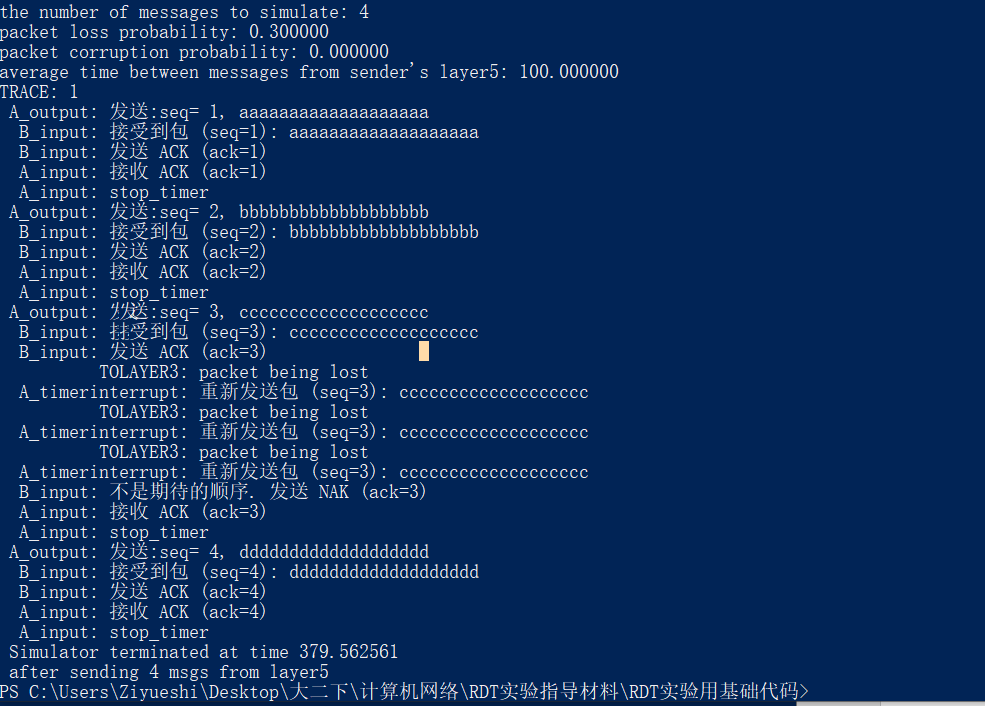


发现顺序很正确，证明程序和预期一致。

2) 数据包、ACK 包出现 error；

  
首先B的ACK包发生错误，导致A丢掉此包。等待，结果超时，A将重新发送所有未ACK的包，这时B发送了他最近接受的一个包，结果又包错误，重复上述步骤，直到A接受ACK。开始发送b，结果又是返回ACK时错误，重复上述。结果发送d的时候都正确，却引发了超时，只得重新发送d。

3) 数据包、ACK 包有 loss；



b的传输正确无loss，当B返回c的ACK时发生loss，连续三次，触发三次A的超时。最后B发送的NCK到A结束。继续D。

### 总结

1. 深入理解可靠数据传输RDT的基本原理。

2. 掌握Go-Back-N协议的设计方法，能够使用FSM图进行设计。

3. 掌握Go-Back-N协议的实现方法。

**本周任务完成表**

在“完成”“没完成”列对应打“√”

|  |  |  |  |
| --- | --- | --- | --- |
| 本周任务  要求 | 完成 | 没完成 | 备注 |
| 协议需求分析。  分别阐述两种协议的需求和所适用的网络场景。 | √ |  |  |
| 协议设计。  使用FSM工具分别设计两种协议的发送端和接收端的协议流程，并给出相应FSM图。 | √ |  |  |
| 协议实现。  根据FSM图在仿真框架下实现Go-Back-N协议。说明协议实现的具体方法、主要数据结构，给出流程图或者伪代码。 | √ |  |  |
| 实验结果及分析。 | √ |  |  |