滑动窗口协议 实验报告

信科 李康为 1900013086

1 实验目的与要求

本实验要求实现一个数据链路层协议的数据传送部分。在一个数据链路层的模拟实现环境中，用 C++ 语言实现三个数据链路层的协议：

- 1 比特滑动窗口协议
- 回退 N 帧滑动窗口协议
- 选择性重传协议

2 实验过程

根据实验的要求，首先可以对基本的结构和变量等进行定义。而根据滑动窗口协议，考虑到发送的消息和滑动窗口均为先进先出的结构，因此可以借助 queue 与 deque 来对它们进行存储。

```
typedef enum{data, ack, nak} frame_kind;

typedef struct frame_head
{
    frame_kind kind;            // 帧类型
    unsigned int seq;           // 序列号
    unsigned int ack;           // 确认号
    unsigned char data[100];    // 数据
};

typedef struct frame
{
    frame_head head;            // 帧头
    unsigned int size;          // 数据的大小
};
```

对于停等协议而言，可以视为回退 N 帧滑动窗口协议的特例，过程相似；而第三个函数选择性重传协议与回退 N 帧滑动窗口协议的区别则在于重传时的帧不同。因此三个函数的大体框架基本相同。

```
int Template(char *pBuffer, int bufferSize, UINT8 messageType)
```

```
{
    frame buffer;
    if (messageType == MSG_TYPE_SEND) {
        ...
        }
    } else if (messageType == MSG_TYPE_RECEIVE) {
        ...
        }
    } else if (messageType == MSG_TYPE_TIMEOUT) {
        ...
    } else {
        return -1;
    }
    return 0;
}
```

以回退 N 帧滑动窗口协议为例，若 `messageType` 为 `MSG_TYPE_SEND`，则将待发送的帧缓存，存入发送队列 `queue<frame> sendq` 中，接下来需要判断发送窗口是否达到规定限度，若未达到规定限度，则新打开一个窗口，调用给定的函数 `SendFRAMEPacket` 将该帧发送。

若 `messageType` 为 `MSG_TYPE_RECEIVE`，则测试函数检查 ACK 值后，将该 ACK 对应的窗口关闭，若发送队列中存在等待发送的帧，便将一个等待发送的帧发送并打开一个新的窗口。

若 `messageType` 为 `MSG_TYPE_TIMEOUT`，查看其返回的 ACK，若在窗口内部，则，测试函数将根据帧序号将该帧以及后面发送过的帧重新发送。

而对于第一个函数而言，所需要做出的改变为将算法的窗口设置为 1，而第三个函数，对于 RECEIVE 命令，之后在重发时遍历窗口找到对应的 seq 的数据包并重新发送即可。

3 实验中遇到的问题

这一次的上机实验是第一次接触 NetRiver 网络实验系统，开始时有些摸不着头绪，但是在认真听过助教的讲解与阅读实验手册后，也逐渐对 NetRiver 实验系统熟悉了起来。另一方面，在完成第一个任务即停等协议测试函数时，由于对于协议的原理还没有深入理解便匆忙开始写代码，在定义记录当前状态的布尔变量时没有规定为静态变量，导致测试错误，后来经过多次的 printf 调试，我终于找到了错误所在。

4 感想与建议

NetRiver 虽然支持断点调试与单步执行等，但是由于各种因素，实际情况中 debug 体验不佳。通过助教学长的建议，我主要借助 printf 来进行调试，最终顺利的完成了这一次的滑动窗口协议上机实验，期望后续的三次上机实验都能顺利完成。

总的来说，这一次的上机实验中我收获到了很多，对于滑动窗口协议本身的机制等有了更为深刻的理解，是一次很棒的经历。