北京邮电大学《计算机网络》课程实验报告

实验名称	数据链路层滑动窗口协议的设计与实现	学院	计算机	指导教师	王晓茹
班 级	班内序号	学 号	学生姓名	成绩	
2018211318	02	2018210074	熊宇		
实验内容	本次实验选用的滑动窗口协议为选择重传协议，利用所学数据链路层原理，自行设计一个滑动窗口协议，在仿真环境下编程实现有噪音信道环境下两站点之间无差错双工通信。信道模型为 8000bps 全双工卫星信道，信道传播时延 270 毫秒，信道误码率为 10^{-5}，信道提供帧传输服务，网络层分组长度固定为 256 字节。 本次实验选用的滑动窗口协议为选择重传协议，并且使用了 NAK 通知机制。				
学生实验报告	(详见“实验报告和源程序”册)				
课程设计成绩评定	评语: 成绩: 指导教师签名: 年 月 日				

注：评语要体现每个学生的工作情况，可以加页。

北京邮电大学《计算机网络》课程实验报告

<table>
<tr><td>实验名称</td><td colspan="2">数据链路层滑动窗口协议的设计与实现</td><td>学院</td><td>计算机</td><td>指导教师</td><td>王晓茹</td></tr>
<tr><td>班 级</td><td>班内序号</td><td colspan="2">学 号</td><td>学生姓名</td><td colspan="2">成绩</td></tr>
<tr><td>2018211318</td><td>02</td><td colspan="2">2018210074</td><td>熊宇</td><td colspan="2"></td></tr>
<tr><td></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td></tr>
<tr><td>实验内容</td><td colspan="6">本次实验选用的滑动窗口协议为：
1. 选择重传
是否选用了 NAK 机制：
1. 使用了 NAK 通知机制。</td></tr>
<tr><td>学生实验报告</td><td colspan="6">实验报告是否上交？
已上交。</td></tr>
<tr><td>课程设计成绩评定</td><td colspan="6">评语：
1. 程序是否存在 BUG？
编写过程中出现 BUG，修正后目前无 BUG。
2. 20 分钟后，发送效率性能情况？峰值？
详见性能测试表。
3. 性能表与实际测试性能是否相符？
相符。
4. 程序中是否使用了合理的数据结构？如数据帧的设计？
使用了。详见实验报告【实验设计】数据结构部分。</td></tr>
</table>

	5. 在数据帧头部中，是否使用了字节计数域？ 并未使用。 6. 如何设定最大窗口？（写出公式和具体值） NR_BUFS = ((MAX_SEQ + 1) / 2)，其中 SEQ 用 5 bit 表示，因此 MAX_SEQ = (2^5) - 1 = 31，因此选取 NR_BUFS=16。 7. 如何设定超时重传时间？（给出计算公式）是否合理？ ack_timer 和 data_timer 的选取要综合考虑,ack_timer 要远小于 data_timer，NR_BUFS 保持不变。 最终选取 ack_timer=280,data_timer=5000 8. 如何设定 ACK 超时时间？（给出公式） ack_timer 和 data_timer 的选取要综合考虑,ack_timer 要远小于 data_timer，NR_BUFS 保持不变。 最终选取 ack_timer=280,data_timer=5000 9. 程序的书写风格是否符合实验报告的要求？ 符合。 10. 实验分工是否合理？ 单人完成。

目录

北京郵電大學

计算机网络实验报告

题目：选择重传协议

班　　级：2018211318 班

学　　号：2018210074

姓　　名：熊宇

学　　院：计算机学院

2020 年 5 月 4 日

实验报告

一 实验内容和实验目的

利用所学数据链路层原理，自己设计一个滑动窗口协议，在仿真环境下编程实现有噪音信道环境下两站点之间无差错双工通信。信道模型为 8000bps 全双工卫星信道，信道传播时延 270 毫秒，信道误码率为 10-5，信道提供字节流传输服务，网络层分组长度固定为 256 字节。

通过该实验，进一步巩固和深刻理解数据链路层误码检测的 CRC 校验技术，以及滑动窗口的工作机理。滑动窗口机制的两个主要目标：(1) 实现有噪音信道环境下的无差错传输；(2)充分利用传输信道的带宽。在程序能够稳定运行并成功实现第一个目标之后，运行程序并检查在信道没有误码和存在误码两种情况下的信道利用率。为实现第二个目标，提高滑动窗口协议信道利用率，需要根据信道实际情况合理地为协议配置工作参数，包括滑动窗口的大小和重传定时器时限以及 ACK 搭载定时器的时限。这些参数的设计，需要充分理解滑动窗口协议的工作原理并利用所学的理论知识，经过认真的推算，计算出最优取值，并通过程序的运行进行验证。

通过该实验提高同学的编程能力和实践动手能力，体验协议软件在设计上各种问题和调试难度，设计在运行期可跟踪分析协议工作过程的协议软件，巩固和深刻理解理论知识并利用这些知识对系统进行优化，对实际系统中的协议分层和协议软件的设计与实现有基本的认识。

二 实验设备环境

Windows 环境 PC 机，Microsoft Visual Studio 2013+集成化开发环境。

三 实验设计

（1）数据结构：

① 帧

/* 定义帧类型 */

```
typedef struct FRAME
{
  unsigned char kind; /* 帧的种类 */
  unsigned char ack; /* ACK 号 */
  unsigned char seq; /* 序列号 */
  packet data; /* 需要被转发的数据 */
  unsigned int  padding;
}FRAME;

DATA Frame
    +=========+========+========+==============+========+
    | KIND(1) | SEQ(1) | ACK(1) | DATA(240~256) | CRC(4) |
    +=========+========+========+==============+========+
```

```
ACK Frame
    +=========+========+========+
    | KIND(1) | ACK(1) | CRC(4) |
    +=========+========+========+

NAK Frame
    +=========+========+========+
    | KIND(1) | ACK(1) | CRC(4) |
    +=========+========+========+
KIND：表示帧的类别 ACK：ACK 序列号 SEQ：帧序列号 CRC：校验和
```

② 全局变量解释

```
/* 发送窗口上限 */
static unsigned char next_frame_to_send;
/* 发送窗口下限 */
static unsigned char ack_expected;
/* 接收窗口上限 */
static unsigned char too_far;
/* 接收窗口下限 */
static unsigned char frame_expected;
/* 计数，发了多少个 */
static unsigned char nbuffered;

/* 数据帧超时机制 */
#define DATA_TIMER  5000
/* ACK 超时机制 */
#define ACK_TIMER 280
/* 帧的最大长度 */
#define MAX_SEQ 31
/* 窗口大小 */
#define NR_BUFS ((MAX_SEQ + 1) / 2)
/* 实现 k 的按模 +1 */
#define inc(k) if(k < MAX_SEQ)k++; else k=0

/* phl_ready = 0 意味着物理层还没有准备好 */
static int phl_ready = 0;
/* no_nak = true 意味着还没有发送过一个 NAK */
bool no_nak = true;
```

② 主程序中变量解释

```
/* 事件，在 protocol.h 中已完成定义 */
#define NETWORK_LAYER_READY  0
#define PHYSICAL_LAYER_READY 1
```

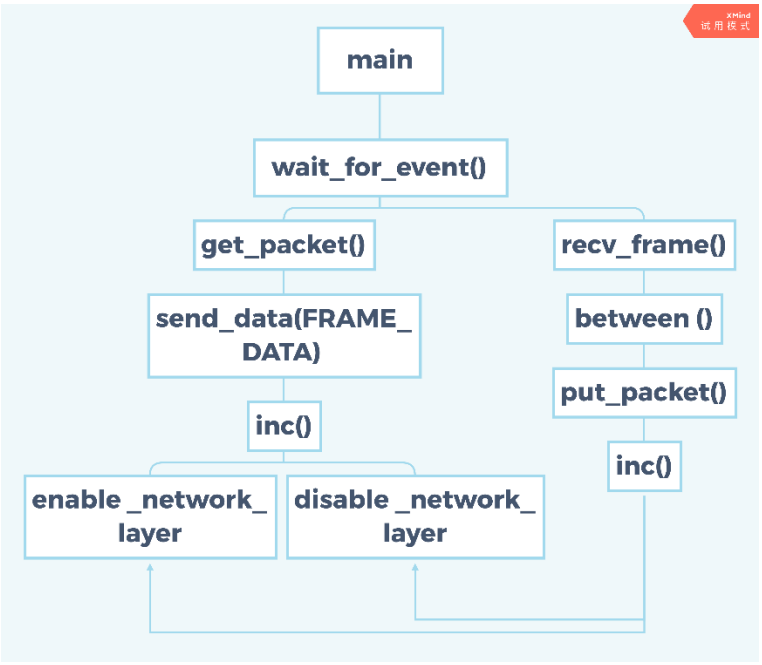
```
#define FRAME_RECEIVED      2
#define DATA_TIMEOUT        3
#define ACK_TIMEOUT         4

int event, arg;
FRAME f;
int len = 0;
int i;


/* 发送窗口的“椅子” */
packet out_buf[NR_BUFS];
/* 接受窗口的“椅子” */
packet in_buf[NR_BUFS];
/* 收到的数据帧该坐的“椅子”当前是否空闲 */
bool arrived[NR_BUFS];

/* 初始化 */
/* 接收端初始化时，enable_network_layer() 被抑制 */
enable_network_layer();
ack_expected = 0;
next_frame_to_send = 0;
frame_expected = 0;
too_far = NR_BUFS;
nbuffered = 0;
```

（2）模块结构



`static int between(unsigned char a, unsigned char b, unsigned char c)`
作用：判断数据帧是否位于接受窗口内，来判断是否暂存
参数：`a,b,c`，均为字节类型，其中两个分别为窗口的上、下界，一个为帧的编号。其中，发送窗口的上界和下界分别为 `next_frame_to_send` 和 `ack_expected`，接收窗口的上界和下界分别为 `too_far` 和 `frame_expected`，均定义在 `main` 函数中。

`static void put_frame(unsigned char* frame, int len)`
作用：向网络层上交数据帧
参数：`frame`，字节数组，由除 `padding` 域之外的帧内容转换而来；`len`，整型，为帧的当前长度。

`static void send_data(unsigned char fk, unsigned char frame_nr, unsigned char frame_expected, packet buffer[])`
作用：发出帧（此处按帧的类型用 `if` 语句进行不同的操作，所以我将三个函数合在一起）
参数：`fk`，字节类型，为帧的内容；`frame_nr`，字节类型，为帧的编号；`frame_expected`，字节类型，为希望收到的帧的编号；`buffer[]`，二维字节数组，为缓冲区。

`int main(int argc, char** argv)`
作用：主程式，包含选择重传协议的算法流程。
参数：`argc`，整型，表示命令行参数的个数；`argv`，二维字符数组，表示参数内容。

（3）算法流程

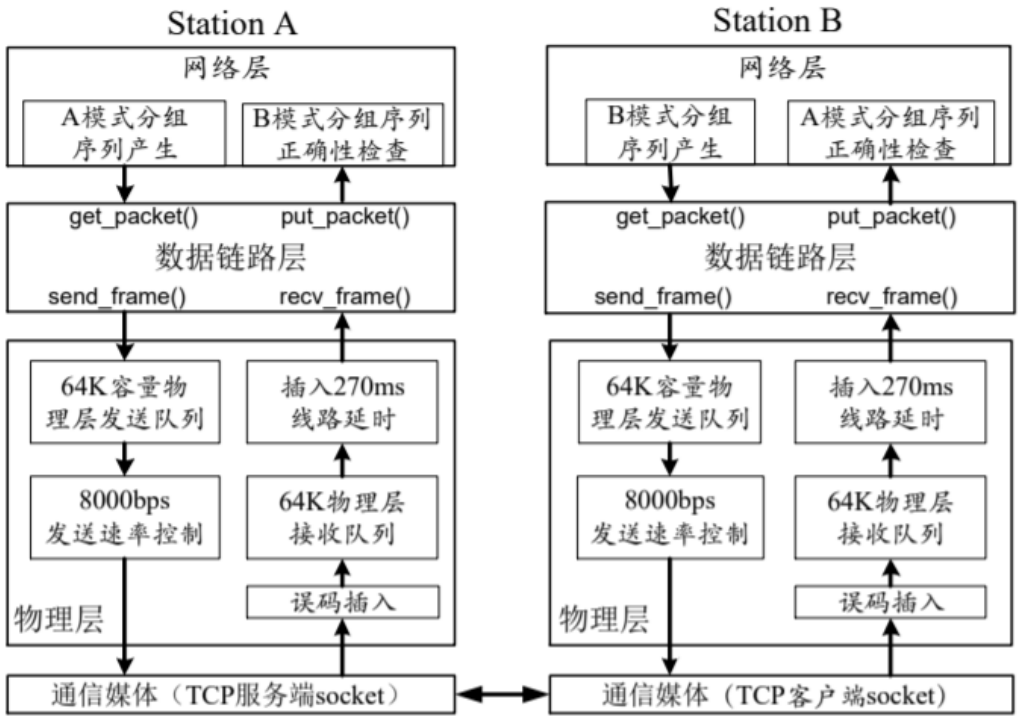


图 1　程序总体结构

物理层：为数据链路层提供的服务为8000bps，270ms传播延时10-5误码率的字节流传输通道。为

了仿真实现上述服务质量的信道,利用在同一台计算机上TCP socket完成两个站点之间的通信。由于同一台计算机上TCP通信传播时延短、传播速度快、没有误码，物理层仿真程序在发送端利用“令牌桶”算法限制发送速率以仿真8000bps线路；在接收端误码插入模块利用一个伪随机数“随机地”篡改从TCP收到的数据，使得所接收到的每个比特出现差错的概率为10-5；接收到的数据缓冲后延时270ms才提交给数据链路层程序，以仿真信道的传播时延特性。为了简化程序，省略了“成帧”功能，数据链路层利用接口函数send_data()和recv_frame()发送和接收一帧。

数据链路层：程序由同学自己设计完成，通过物理层提供的帧发送和接收函数，利用物理层提供的服务。通过get_packet()函数从网络层得到一个分组；当数据链路层成功接收到一个分组后，通过put_packet()函数提交给网络层。

网络层：利用数据链路层提供的“可靠的分组传输”服务，在站点A与站点B之间交换长度固定为256字节的数据分组。网络层把产生的分组交付数据链路层，并接受数据链路层提交来的数据分组。

四 实验分析

（1）实验结果简介：

此次实验，本小组采用的选择重传协议，能够较长时间地在有误码信道环境中无差错传输，同时，在传输过程中，如果出现错误，丢失的帧，则会发出警报，在接受方会得到提示，因此能够可靠地实现通信。

（2）协议参数的选取：

```
NR_BUFS 的选取
NR_BUFS = ((MAX_SEQ + 1) / 2)
因此选取 NR_BUFS=16

ack_timer 和 data_timer 的选取
要综合考虑，ack_timer 要远小于 data_timer，NR_BUFS 保持不变
最终选取 ack_timer=280,data_timer=5000
```

（3）理论分析：

根据所设计的滑动窗口工作机制(Go-Back-N 或者选择重传)，推导出在无差错信道环境下分组层能获得的最大信道利用率；推导出在有差错条件下分组层能获得的最大信道利用率。给出理论推导过程。理论推导的目的是得到信道利用率的极限数据。为了简化有差错条件下的最大利用率推导过程，可以对问题模型进行简化，比如：假定出现超时后一次重传可以 100%正确传输，但是简化问题分析的这些假设必须不会对整个结论产生较大的误差。

分析：由于需要携带帧讯息，因此最大的信息利用率为约为（256-3）/256≈98.8%。由于信道的最大比特率为 8000bps，可得出每传输一个字节耗时 1ms；每帧的附加信息固定为 10，耗时 10ms；若出现转义字符，则可能增加时间。现在假设信道上始终有数据需要传送，这样就可以简化模型。在有 10-5 的错误率的信道上，算下来在这 100000 个比特中可以传送 100000/（256*8）≈48 个数据包，则可得出每 48 个数据包会有一个出错，假设每出错一次，在限定时间内可以重

传该帧使其成为正确帧。则每传送 48 个数据包需要传送 48+1+1=50 次，此时信道利用率≈94.9%，而实际上由于程序设计的原因，当一个数据包超时后，往往要重复多次传输给数据包造成信道的浪费，若有 k 次重传，则信道利用率为（（256-3）/256）*（48/（48+1+k）），在此程式中，平均要重传 10 次，因此信道利用率降为 80.4%。在 ESC/FLAG 模式中传输的平均 250 个字符需要两倍的传输空间即极限值 500，此时的信道利用率的极限值是，而在 10-5 的错误率的信道上则利用率为 47.5%，若平均每个错帧重传 10 次，利用率将下降为 40.2%。

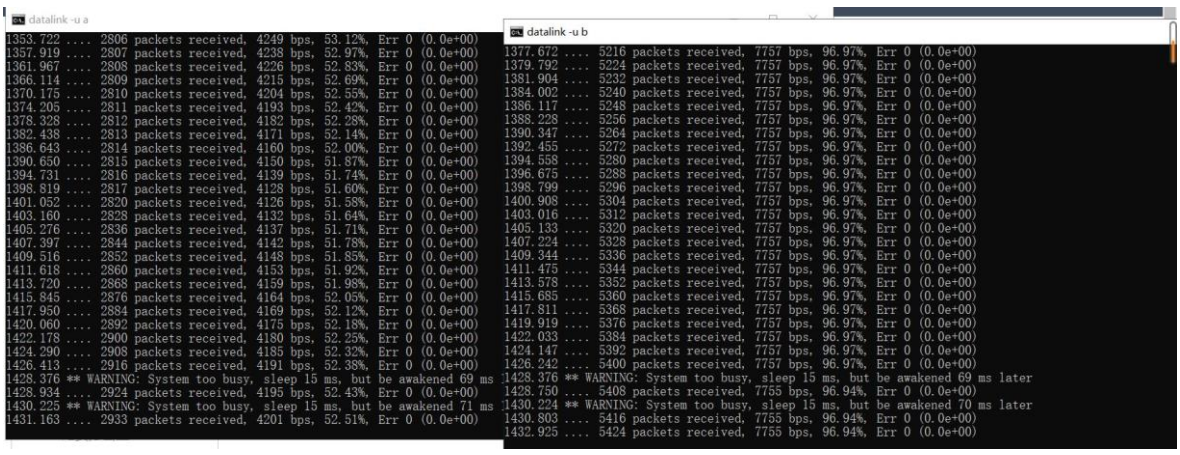
（4）实验测试：

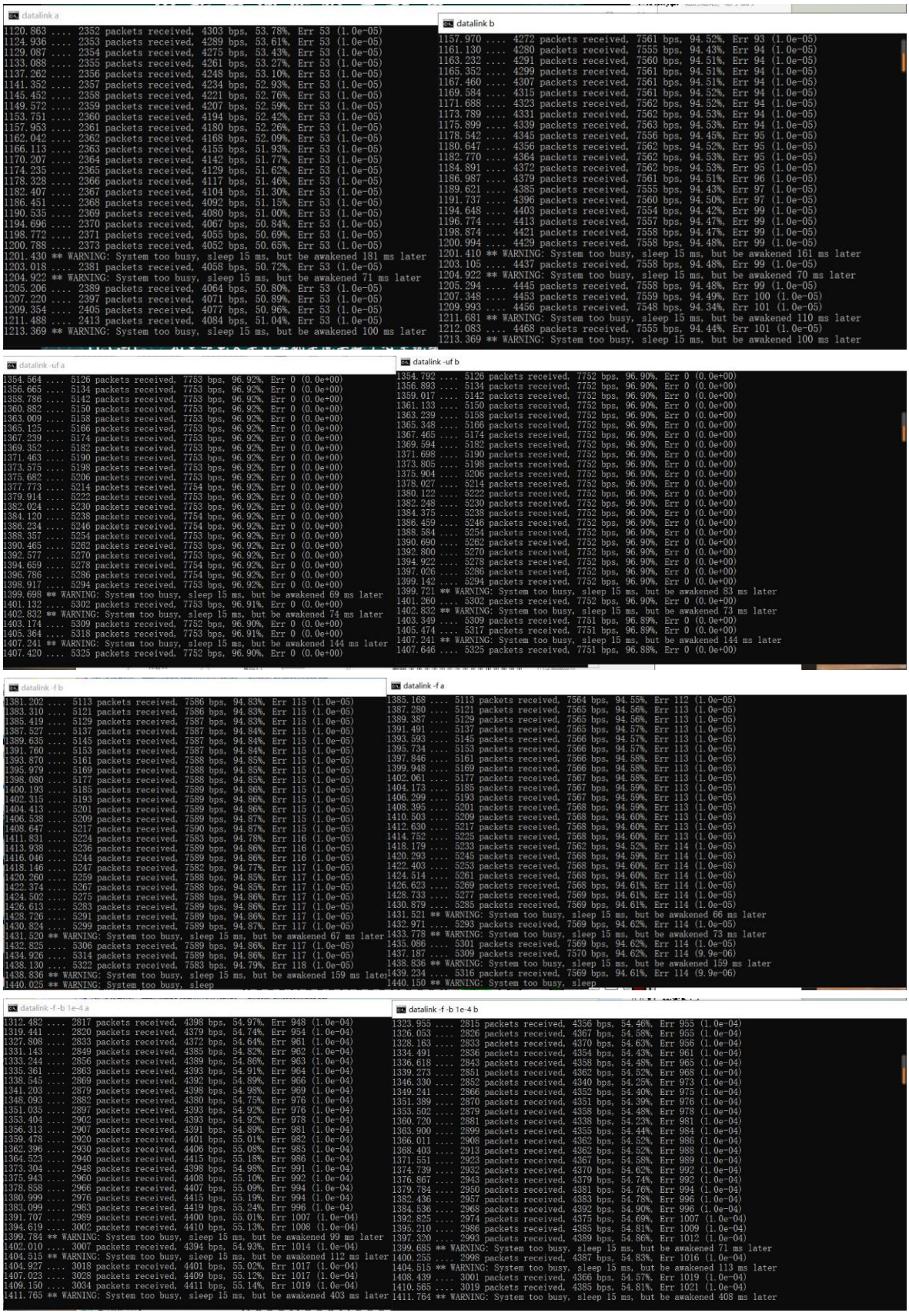
滑动窗口机制的两个主要目标：实现有误码信道环境下的无差错传输；充分利用传输信道的带宽。完成无差错传输只是达到了基本目标，追求信道实际利用率是关键。测试以下几种情况下你所设计的软件能否正常工作20分钟，记录线路利用率。在实验报告中分析这些数据，给出实际线路利用率离极限效率之间尚有多大差距，并解释产生这样结果的原因。

性能测试记录表

序号	命令选项	说明	运行时间(秒）	Selective 算法线路利用率(%)	
				A	B
1	--utopia	无误码信道数据传输	1431	52.51	96.94
2	无	站点 A 分组层平缓方式发出数据，站点 B 周期性交替“发送 100 秒，停发 100 秒”	1213	51.04	94.44
3	--flood --utopia	无误码信道，站点 A 和站点 B 的分组层都洪水式产生分组	1401	96.90	96.88
4	--flood	站点 A/B 的分组层都洪水式产生分组	1460	94.61	94.79
5	--flood --ber=1e-4	站点 A/B 的分组层都洪水式产生分组，线路误码率设为 10^{-4}	1411	55.14	54.81

运行时间要求超过 10 分钟。（下面是实验截图）




datalink a
1120.863 .... 2352 packets received, 4303 bps, 53.78%, Err 53 (1.0e-05)
1124.936 .... 2353 packets received, 4289 bps, 53.61%, Err 53 (1.0e-05)
1129.087 .... 2354 packets received, 4275 bps, 53.43%, Err 53 (1.0e-05)
1133.088 .... 2355 packets received, 4261 bps, 53.27%, Err 53 (1.0e-05)
1137.262 .... 2356 packets received, 4248 bps, 53.10%, Err 53 (1.0e-05)
1141.352 .... 2357 packets received, 4234 bps, 52.93%, Err 53 (1.0e-05)
1145.452 .... 2358 packets received, 4221 bps, 52.76%, Err 53 (1.0e-05)
1149.572 .... 2359 packets received, 4207 bps, 52.59%, Err 53 (1.0e-05)
1153.751 .... 2360 packets received, 4194 bps, 52.42%, Err 53 (1.0e-05)
1157.953 .... 2361 packets received, 4180 bps, 52.26%, Err 53 (1.0e-05)
1162.042 .... 2362 packets received, 4168 bps, 52.09%, Err 53 (1.0e-05)
1166.113 .... 2363 packets received, 4155 bps, 51.93%, Err 53 (1.0e-05)
1170.207 .... 2364 packets received, 4142 bps, 51.77%, Err 53 (1.0e-05)
1174.235 .... 2365 packets received, 4129 bps, 51.62%, Err 53 (1.0e-05)
1178.328 .... 2366 packets received, 4117 bps, 51.46%, Err 53 (1.0e-05)
1182.407 .... 2367 packets received, 4104 bps, 51.30%, Err 53 (1.0e-05)
1186.451 .... 2368 packets received, 4092 bps, 51.15%, Err 53 (1.0e-05)
1190.535 .... 2369 packets received, 4080 bps, 51.00%, Err 53 (1.0e-05)
1194.696 .... 2370 packets received, 4067 bps, 50.84%, Err 53 (1.0e-05)
1198.772 .... 2371 packets received, 4055 bps, 50.69%, Err 53 (1.0e-05)
1200.788 .... 2373 packets received, 4052 bps, 50.65%, Err 53 (1.0e-05)
1201.430 ** WARNING: System too busy, sleep 15 ms, but be awakened 181 ms later
1203.018 .... 2381 packets received, 4058 bps, 50.72%, Err 53 (1.0e-05)
1204.922 ** WARNING: System too busy, sleep 15 ms, but be awakened 71 ms later
1205.206 .... 2389 packets received, 4064 bps, 50.80%, Err 53 (1.0e-05)
1207.220 .... 2397 packets received, 4071 bps, 50.89%, Err 53 (1.0e-05)
1209.354 .... 2405 packets received, 4077 bps, 50.96%, Err 53 (1.0e-05)
1211.488 .... 2413 packets received, 4084 bps, 51.04%, Err 53 (1.0e-05)
1213.369 ** WARNING: System too busy, sleep 15 ms, but be awakened 100 ms later
datalink b
1157.970 .... 4272 packets received, 7561 bps, 94.52%, Err 93 (1.0e-05)
1161.130 .... 4280 packets received, 7555 bps, 94.43%, Err 94 (1.0e-05)
1163.232 .... 4291 packets received, 7560 bps, 94.51%, Err 94 (1.0e-05)
1165.352 .... 4299 packets received, 7561 bps, 94.51%, Err 94 (1.0e-05)
1167.460 .... 4307 packets received, 7561 bps, 94.51%, Err 94 (1.0e-05)
1169.584 .... 4315 packets received, 7561 bps, 94.52%, Err 94 (1.0e-05)
1171.688 .... 4323 packets received, 7562 bps, 94.52%, Err 94 (1.0e-05)
1173.789 .... 4331 packets received, 7562 bps, 94.53%, Err 94 (1.0e-05)
1175.899 .... 4339 packets received, 7563 bps, 94.53%, Err 94 (1.0e-05)
1178.542 .... 4345 packets received, 7556 bps, 94.45%, Err 95 (1.0e-05)
1180.647 .... 4356 packets received, 7562 bps, 94.52%, Err 95 (1.0e-05)
1182.770 .... 4364 packets received, 7562 bps, 94.53%, Err 95 (1.0e-05)
1184.891 .... 4372 packets received, 7562 bps, 94.53%, Err 95 (1.0e-05)
1186.987 .... 4379 packets received, 7561 bps, 94.51%, Err 96 (1.0e-05)
1189.621 .... 4385 packets received, 7555 bps, 94.43%, Err 97 (1.0e-05)
1191.737 .... 4396 packets received, 7560 bps, 94.50%, Err 97 (1.0e-05)
1194.648 .... 4403 packets received, 7554 bps, 94.42%, Err 99 (1.0e-05)
1196.774 .... 4413 packets received, 7557 bps, 94.47%, Err 99 (1.0e-05)
1198.874 .... 4421 packets received, 7558 bps, 94.47%, Err 99 (1.0e-05)
1200.994 .... 4429 packets received, 7558 bps, 94.48%, Err 99 (1.0e-05)
1201.410 ** WARNING: System too busy, sleep 15 ms, but be awakened 161 ms later
1203.105 .... 4437 packets received, 7558 bps, 94.48%, Err 99 (1.0e-05)
1204.922 ** WARNING: System too busy, sleep 15 ms, but be awakened 70 ms later
1205.294 .... 4445 packets received, 7558 bps, 94.48%, Err 99 (1.0e-05)
1207.348 .... 4453 packets received, 7559 bps, 94.49%, Err 100 (1.0e-05)
1209.993 .... 4456 packets received, 7548 bps, 94.34%, Err 101 (1.0e-05)
1211.681 ** WARNING: System too busy, sleep 15 ms, but be awakened 110 ms later
1212.083 .... 4468 packets received, 7555 bps, 94.44%, Err 101 (1.0e-05)
1213.369 ** WARNING: System too busy, sleep 15 ms, but be awakened 100 ms later
datalink -uf a
1354.564 .... 5126 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1356.665 .... 5134 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1358.786 .... 5142 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1360.882 .... 5150 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1363.009 .... 5158 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1365.125 .... 5166 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1367.239 .... 5174 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1369.352 .... 5182 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1371.463 .... 5190 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1373.575 .... 5198 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1375.682 .... 5206 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1377.773 .... 5214 packets received, 7754 bps, 96.92%, Err 0 (0.0e+00)
1379.914 .... 5222 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1382.024 .... 5230 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1384.120 .... 5238 packets received, 7754 bps, 96.92%, Err 0 (0.0e+00)
1386.234 .... 5246 packets received, 7754 bps, 96.92%, Err 0 (0.0e+00)
1388.357 .... 5254 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1390.465 .... 5262 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1392.577 .... 5270 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1394.659 .... 5278 packets received, 7754 bps, 96.92%, Err 0 (0.0e+00)
1396.786 .... 5286 packets received, 7754 bps, 96.92%, Err 0 (0.0e+00)
1398.917 .... 5294 packets received, 7753 bps, 96.92%, Err 0 (0.0e+00)
1399.698 ** WARNING: System too busy, sleep 15 ms, but be awakened 69 ms later
1401.132 .... 5302 packets received, 7753 bps, 96.91%, Err 0 (0.0e+00)
1402.832 ** WARNING: System too busy, sleep 15 ms, but be awakened 74 ms later
1403.174 .... 5309 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1405.364 .... 5319 packets received, 7753 bps, 96.91%, Err 0 (0.0e+00)
1407.241 ** WARNING: System too busy, sleep 15 ms, but be awakened 144 ms later
1407.420 .... 5325 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
datalink -uf b
1354.792 .... 5126 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1356.893 .... 5134 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1359.017 .... 5142 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1361.133 .... 5150 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1363.239 .... 5158 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1365.348 .... 5166 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1367.465 .... 5174 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1369.594 .... 5182 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1371.698 .... 5190 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1373.805 .... 5198 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1375.904 .... 5206 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1378.027 .... 5214 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1380.122 .... 5222 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1382.248 .... 5230 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1384.375 .... 5238 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1386.459 .... 5246 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1388.584 .... 5254 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1390.690 .... 5262 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1392.800 .... 5270 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1394.922 .... 5278 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1397.026 .... 5286 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1399.142 .... 5294 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1399.721 ** WARNING: System too busy, sleep 15 ms, but be awakened 83 ms later
1401.260 .... 5302 packets received, 7752 bps, 96.90%, Err 0 (0.0e+00)
1402.832 ** WARNING: System too busy, sleep 15 ms, but be awakened 73 ms later
1403.349 .... 5309 packets received, 7751 bps, 96.89%, Err 0 (0.0e+00)
1405.474 .... 5317 packets received, 7751 bps, 96.89%, Err 0 (0.0e+00)
1407.241 ** WARNING: System too busy, sleep 15 ms, but be awakened 144 ms later
1407.646 .... 5325 packets received, 7751 bps, 96.88%, Err 0 (0.0e+00)
datalink -f b
1381.202 .... 5113 packets received, 7586 bps, 94.83%, Err 115 (1.0e-05)
1383.310 .... 5121 packets received, 7586 bps, 94.83%, Err 115 (1.0e-05)
1385.419 .... 5129 packets received, 7587 bps, 94.83%, Err 115 (1.0e-05)
1387.527 .... 5137 packets received, 7587 bps, 94.84%, Err 115 (1.0e-05)
1389.635 .... 5145 packets received, 7587 bps, 94.84%, Err 115 (1.0e-05)
1391.760 .... 5153 packets received, 7587 bps, 94.84%, Err 115 (1.0e-05)
1393.870 .... 5161 packets received, 7588 bps, 94.85%, Err 115 (1.0e-05)
1395.979 .... 5169 packets received, 7588 bps, 94.85%, Err 115 (1.0e-05)
1398.080 .... 5177 packets received, 7588 bps, 94.85%, Err 115 (1.0e-05)
1400.193 .... 5185 packets received, 7589 bps, 94.86%, Err 115 (1.0e-05)
1402.315 .... 5193 packets received, 7589 bps, 94.86%, Err 115 (1.0e-05)
1404.413 .... 5201 packets received, 7589 bps, 94.86%, Err 115 (1.0e-05)
1406.538 .... 5209 packets received, 7589 bps, 94.87%, Err 115 (1.0e-05)
1408.647 .... 5217 packets received, 7590 bps, 94.87%, Err 115 (1.0e-05)
1411.831 .... 5224 packets received, 7583 bps, 94.78%, Err 116 (1.0e-05)
1413.938 .... 5236 packets received, 7589 bps, 94.86%, Err 116 (1.0e-05)
1416.046 .... 5244 packets received, 7589 bps, 94.86%, Err 116 (1.0e-05)
1418.146 .... 5247 packets received, 7582 bps, 94.77%, Err 117 (1.0e-05)
1420.260 .... 5259 packets received, 7588 bps, 94.85%, Err 117 (1.0e-05)
1422.374 .... 5267 packets received, 7588 bps, 94.85%, Err 117 (1.0e-05)
1424.502 .... 5275 packets received, 7588 bps, 94.86%, Err 117 (1.0e-05)
1426.613 .... 5283 packets received, 7589 bps, 94.86%, Err 117 (1.0e-05)
1428.726 .... 5291 packets received, 7589 bps, 94.86%, Err 117 (1.0e-05)
1430.824 .... 5299 packets received, 7589 bps, 94.87%, Err 117 (1.0e-05)
1431.520 ** WARNING: System too busy, sleep 15 ms, but be awakened 67 ms later
1432.825 .... 5306 packets received, 7589 bps, 94.86%, Err 117 (1.0e-05)
1434.926 .... 5314 packets received, 7589 bps, 94.86%, Err 117 (1.0e-05)
1438.130 .... 5322 packets received, 7583 bps, 94.79%, Err 118 (1.0e-05)
1438.836 ** WARNING: System too busy, sleep 15 ms, but be awakened 159 ms later
1440.025 ** WARNING: System too busy, sleep
datalink -f a
1385.168 .... 5113 packets received, 7564 bps, 94.55%, Err 112 (1.0e-05)
1387.280 .... 5121 packets received, 7565 bps, 94.56%, Err 113 (1.0e-05)
1389.387 .... 5129 packets received, 7565 bps, 94.56%, Err 113 (1.0e-05)
1391.491 .... 5137 packets received, 7565 bps, 94.57%, Err 113 (1.0e-05)
1393.593 .... 5145 packets received, 7566 bps, 94.57%, Err 113 (1.0e-05)
1395.734 .... 5153 packets received, 7566 bps, 94.57%, Err 113 (1.0e-05)
1397.846 .... 5161 packets received, 7566 bps, 94.58%, Err 113 (1.0e-05)
1399.948 .... 5169 packets received, 7566 bps, 94.58%, Err 113 (1.0e-05)
1402.061 .... 5177 packets received, 7567 bps, 94.58%, Err 113 (1.0e-05)
1404.173 .... 5185 packets received, 7567 bps, 94.59%, Err 113 (1.0e-05)
1406.299 .... 5193 packets received, 7567 bps, 94.59%, Err 113 (1.0e-05)
1408.395 .... 5201 packets received, 7568 bps, 94.59%, Err 113 (1.0e-05)
1410.503 .... 5209 packets received, 7568 bps, 94.60%, Err 113 (1.0e-05)
1412.630 .... 5217 packets received, 7568 bps, 94.60%, Err 113 (1.0e-05)
1414.752 .... 5225 packets received, 7568 bps, 94.60%, Err 113 (1.0e-05)
1418.179 .... 5233 packets received, 7562 bps, 94.52%, Err 114 (1.0e-05)
1420.293 .... 5245 packets received, 7568 bps, 94.59%, Err 114 (1.0e-05)
1422.403 .... 5253 packets received, 7568 bps, 94.60%, Err 114 (1.0e-05)
1424.514 .... 5261 packets received, 7568 bps, 94.60%, Err 114 (1.0e-05)
1426.623 .... 5269 packets received, 7568 bps, 94.61%, Err 114 (1.0e-05)
1428.733 .... 5277 packets received, 7569 bps, 94.61%, Err 114 (1.0e-05)
1430.879 .... 5285 packets received, 7569 bps, 94.61%, Err 114 (1.0e-05)
1431.521 ** WARNING: System too busy, sleep 15 ms, but be awakened 66 ms later
1432.971 .... 5293 packets received, 7569 bps, 94.62%, Err 114 (1.0e-05)
1433.778 ** WARNING: System too busy, sleep 15 ms, but be awakened 73 ms later
1435.086 .... 5301 packets received, 7569 bps, 94.62%, Err 114 (1.0e-05)
1437.187 .... 5309 packets received, 7570 bps, 94.62%, Err 114 (9.9e-06)
1438.836 ** WARNING: System too busy, sleep 15 ms, but be awakened 159 ms later
1439.234 .... 5316 packets received, 7569 bps, 94.61%, Err 114 (9.9e-06)
1440.150 ** WARNING: System too busy, sleep
datalink -f -b 1e-4 a
1312.482 .... 2817 packets received, 4398 bps, 54.97%, Err 948 (1.0e-04)
1319.441 .... 2820 packets received, 4379 bps, 54.74%, Err 954 (1.0e-04)
1327.808 .... 2833 packets received, 4372 bps, 54.64%, Err 961 (1.0e-04)
1331.143 .... 2849 packets received, 4385 bps, 54.82%, Err 962 (1.0e-04)
1333.244 .... 2856 packets received, 4389 bps, 54.86%, Err 963 (1.0e-04)
1335.361 .... 2863 packets received, 4393 bps, 54.91%, Err 964 (1.0e-04)
1338.545 .... 2869 packets received, 4392 bps, 54.89%, Err 966 (1.0e-04)
1341.203 .... 2879 packets received, 4398 bps, 54.98%, Err 969 (1.0e-04)
1348.093 .... 2882 packets received, 4380 bps, 54.75%, Err 976 (1.0e-04)
1351.035 .... 2897 packets received, 4393 bps, 54.92%, Err 976 (1.0e-04)
1353.404 .... 2902 packets received, 4393 bps, 54.92%, Err 978 (1.0e-04)
1356.313 .... 2907 packets received, 4391 bps, 54.89%, Err 981 (1.0e-04)
1359.478 .... 2920 packets received, 4401 bps, 55.01%, Err 982 (1.0e-04)
1362.396 .... 2930 packets received, 4406 bps, 55.08%, Err 985 (1.0e-04)
1364.523 .... 2940 packets received, 4415 bps, 55.18%, Err 986 (1.0e-04)
1373.304 .... 2948 packets received, 4398 bps, 54.98%, Err 991 (1.0e-04)
1375.943 .... 2960 packets received, 4408 bps, 55.10%, Err 992 (1.0e-04)
1378.858 .... 2966 packets received, 4407 bps, 55.09%, Err 994 (1.0e-04)
1380.999 .... 2976 packets received, 4415 bps, 55.19%, Err 994 (1.0e-04)
1383.099 .... 2983 packets received, 4419 bps, 55.24%, Err 996 (1.0e-04)
1391.707 .... 2989 packets received, 4400 bps, 55.01%, Err 1007 (1.0e-04)
1394.619 .... 3002 packets received, 4410 bps, 55.13%, Err 1008 (1.0e-04)
1399.784 ** WARNING: System too busy, sleep 15 ms, but be awakened 99 ms later
1402.010 .... 3007 packets received, 4394 bps, 54.93%, Err 1014 (1.0e-04)
1404.515 ** WARNING: System too busy, sleep 15 ms, but be awakened 112 ms later
1404.927 .... 3018 packets received, 4401 bps, 55.02%, Err 1017 (1.0e-04)
1407.023 .... 3028 packets received, 4409 bps, 55.12%, Err 1017 (1.0e-04)
1409.150 .... 3034 packets received, 4411 bps, 55.14%, Err 1019 (1.0e-04)
1411.765 ** WARNING: System too busy, sleep 15 ms, but be awakened 403 ms later
datalink -f -b 1e-4 b
1323.955 .... 2815 packets received, 4356 bps, 54.46%, Err 955 (1.0e-04)
1326.053 .... 2826 packets received, 4367 bps, 54.58%, Err 955 (1.0e-04)
1328.163 .... 2833 packets received, 4370 bps, 54.63%, Err 956 (1.0e-04)
1334.491 .... 2836 packets received, 4354 bps, 54.43%, Err 961 (1.0e-04)
1336.618 .... 2843 packets received, 4358 bps, 54.48%, Err 965 (1.0e-04)
1339.273 .... 2851 packets received, 4362 bps, 54.52%, Err 968 (1.0e-04)
1346.330 .... 2852 packets received, 4340 bps, 54.25%, Err 973 (1.0e-04)
1349.241 .... 2866 packets received, 4352 bps, 54.40%, Err 975 (1.0e-04)
1351.389 .... 2870 packets received, 4351 bps, 54.39%, Err 976 (1.0e-04)
1353.502 .... 2879 packets received, 4358 bps, 54.48%, Err 978 (1.0e-04)
1360.720 .... 2881 packets received, 4338 bps, 54.23%, Err 981 (1.0e-04)
1363.900 .... 2899 packets received, 4355 bps, 54.44%, Err 984 (1.0e-04)
1366.011 .... 2908 packets received, 4362 bps, 54.52%, Err 986 (1.0e-04)
1368.403 .... 2913 packets received, 4362 bps, 54.52%, Err 988 (1.0e-04)
1371.551 .... 2923 packets received, 4367 bps, 54.58%, Err 989 (1.0e-04)
1374.739 .... 2932 packets received, 4370 bps, 54.62%, Err 992 (1.0e-04)
1376.867 .... 2943 packets received, 4379 bps, 54.74%, Err 992 (1.0e-04)
1379.784 .... 2950 packets received, 4381 bps, 54.76%, Err 994 (1.0e-04)
1382.436 .... 2957 packets received, 4383 bps, 54.78%, Err 996 (1.0e-04)
1384.536 .... 2968 packets received, 4392 bps, 54.90%, Err 996 (1.0e-04)
1392.825 .... 2974 packets received, 4375 bps, 54.69%, Err 1007 (1.0e-04)
1395.210 .... 2986 packets received, 4385 bps, 54.81%, Err 1009 (1.0e-04)
1397.320 .... 2993 packets received, 4389 bps, 54.86%, Err 1012 (1.0e-04)
1399.685 ** WARNING: System too busy, sleep 15 ms, but be awakened 71 ms later
1400.255 .... 2998 packets received, 4387 bps, 54.83%, Err 1016 (1.0e-04)
1404.515 ** WARNING: System too busy, sleep 15 ms, but be awakened 113 ms later
1408.439 .... 3001 packets received, 4366 bps, 54.57%, Err 1019 (1.0e-04)
1410.565 .... 3019 packets received, 4385 bps, 54.81%, Err 1021 (1.0e-04)
1411.764 ** WARNING: System too busy, sleep 15 ms, but be awakened 408 ms later

五 研究探索

1. CRC 校验能力
CRC校验码具有以下检错能力：
检查出全部单个错；检查出全部离散的二位错；检查出全部奇数个错；检查出全部长度小于或等于K位的突发错。

2.由于本次试验过程的误码信道是一个比较固定的误码率，而在实际生活当中的误码率不是稳定的，可能会因为传输环境的不同，使得他的误码率波动比较大的，例如，下雨天和晴天，高噪声和低噪声的情况，传输的距离也是影响因素。对于这种动态的误码率的通信过程，可能需要其他的一些参数来控制基本参数值（窗口大小，重传时间等等）来完成。

3.对等协议实体之间的流量控制
在我们设计的协议当中，流量的控制主要通过接收窗口，发送窗口还有确认机制来实现。因为有窗口大小的限制，发送方不会一次性发送过多信息导致接收方被信息洪流所淹没，导致信息丢失。这样可能会导致信道的利用率降低，但是如果合理的设计窗口大小，依然可以达到较高的信号利用率。

六 实验总结

（1）问题解决

完成本次实验代码的编写花了大约8个小时左右，选择的时间是在周三下午和晚上，其中有很大一部分时间用于调试Debug。在编程语言上遇到了对结构类型的调用问题（还C++没好好学的债）。

（2）心得体会

通过这次实验，我对完整的通信过程有了更深刻的理解。在必须动手的实验中，为了实现通信而更加细致地了解了其完整而详尽的过程；此外，在编程的过程中，对代码有了详细的分析，不管是对各个变量，还是函数中的各个参数，都能够清楚知道它所代表的含义及所起的作用。因为是单人完成，本次实验也锻炼了我统筹兼顾和编写代码的能力，使我受益匪浅。

七 源程序

见附页

```c
#include <stdio.h>
#include <string.h>
#include "protocol.h"
#include "datalink.h"

typedef enum
{
  false, true
}bool;


/* 定义包类型 */
typedef struct
{
  unsigned char info[PKT_LEN];
}packet;

/* 定义帧类型 */
typedef struct FRAME
{
  unsigned char kind; /* 帧的种类 */
  unsigned char ack; /* ACK 号 */
  unsigned char seq; /* 序列号 */
  packet data; /* 需要被转发的数据 */
  unsigned int  padding;
}FRAME;

/* 发送窗口上限 */
static unsigned char next_frame_to_send;
/* 发送窗口下限 */
static unsigned char ack_expected;
/* 接收窗口上限 */
static unsigned char too_far;
/* 接收窗口下限 */
static unsigned char frame_expected;
/* 计数，发了多少个 */
static unsigned char nbuffered;

/* 数据帧超时机制 */
#define DATA_TIMER  5000
/* ACK 超时机制 */
#define ACK_TIMER 280
/* 帧的最大长度 */
#define MAX_SEQ 31
/* 窗口大小 */
#define NR_BUFS ((MAX_SEQ + 1) / 2)
/* 实现 k 的按模 +1 */
#define inc(k) if(k < MAX_SEQ)k++; else k=0
```

```
/* phl_ready = 0 意味着物理层还没有准备好 */
static int phl_ready = 0;
/* no_nak = true 意味着还没有发送过一个 NAK */
bool no_nak = true;



/* 判断数据帧是否位于接受窗口内，来判断是否暂存 */
static int between(unsigned char a, unsigned char b, unsigned char c)
{
  return ((a <= b && b < c) || (c < a&& a <= b) || (b < c&& c < a));
}

/* 向网络层上交数据帧 */
static void put_frame(unsigned char* frame, int len)
{
  *(unsigned int*)(frame + len) = crc32(frame, len);
  send_frame(frame, len + 4);
  phl_ready = 0;
}

/* 发出帧，此处按帧的类型用 if 语句进行不同的操作，所以我将三个函数合在一起 */
static void send_data(unsigned char fk, unsigned char frame_nr, unsigned
char frame_expected, packet buffer[])
{
  /* 创建一个空白新帧备用 */
  FRAME s;
  s.kind = fk;
  /* s.seq 只对数据帧有意义 */
  s.seq = frame_nr;
  /* 累积确认，表示最后一个正确收到的 */
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);

  /* 如果是数据帧 */
  if (fk == FRAME_DATA)
  {
      /* 将数据帧复制到 s 中去 */
      memcpy(s.data.info, buffer[frame_nr % NR_BUFS].info, PKT_LEN);
      /* 打印信息 */
      dbg_frame("Send DATA %d %d,ID %d\n", s.seq, s.ack,
*(short*)&(s.data).info);
      /* 将数据帧下发给物理层 */
      put_frame((unsigned char*)&s, 3 + PKT_LEN);
      /* 发送数据帧时，启动数据帧计时器 */
      start_timer(frame_nr % NR_BUFS, DATA_TIMER);
  }
```

```
    /* 如果是 NAK 帧 */
    if (fk == FRAME_NAK)
    {
        /* 已经发了一个 NAK，no_nak 置 false */
        no_nak = false;
        /* 将 NAK 帧下发给物理层 */
        put_frame((unsigned char*)&s, 2);
    }

    /* 如果是 ACK 帧 */
    if (fk == FRAME_ACK)
    {
        /* 打印 */
        dbg_frame("Send ACK %d\n", s.ack);
        /* 将 ACK 帧下发给物理层 */
        put_frame((unsigned char*)&s, 2);
    }

    /* 发送帧时停止 */
    stop_ack_timer();
}



int main(int argc, char** argv)
{
    /*
        事件，在 protocol.h 中已完成定义
        #define NETWORK_LAYER_READY  0
        #define PHYSICAL_LAYER_READY 1
        #define FRAME_RECEIVED       2
        #define DATA_TIMEOUT         3
        #define ACK_TIMEOUT          4
    */
    int event, arg;
    FRAME f;
    int len = 0;
    int i;


    /* 发送窗口的“椅子” */
    packet out_buf[NR_BUFS];
    /* 接受窗口的“椅子” */
    packet in_buf[NR_BUFS];
    /* 收到的数据帧该坐的“椅子”当前是否空闲 */
    bool arrived[NR_BUFS];

    /* 初始化 */
```

```
    /* 接收端初始化时，enable_network_layer() 被抑制 */
    enable_network_layer();
    ack_expected = 0;
    next_frame_to_send = 0;
    frame_expected = 0;
    too_far = NR_BUFS;
    nbuffered = 0;

    for (i = 0; i < NR_BUFS; i++)
    {
        arrived[i] = false;
    }

    /* Create Communication Sockets */
    protocol_init(argc, argv);
    /* 打印作者信息、时间 */
    lprintf("Designed by 2018210074 - Xiong Yu, build: " __DATE__"
"__TIME__"\n");



    for (; ; )
    {
        event = wait_for_event(&arg);

        switch (event)
        {
            /* 对于发送端，网络层有信息要发送 */
        case NETWORK_LAYER_READY:
            /* 为发新帧做准备 */
            nbuffered++;
            /* 取数据 */
            get_packet(out_buf[next_frame_to_send % NR_BUFS].info);
            /* 发送 DATA 型数据帧 */
            send_data(FRAME_DATA, next_frame_to_send, frame_expected,
out_buf);
            /* 发送窗口上限 +1 */
            inc(next_frame_to_send);
            break;

            /* 物理层有信息来到 */
        case PHYSICAL_LAYER_READY:
            phl_ready = 1;
            break;

            /* 接收到帧 */
        case FRAME_RECEIVED:
            len = recv_frame((unsigned char*)&f, sizeof f);
            /* 检查帧是否正常 */
```

```
            if (len < 5 || crc32((unsigned char*)&f, len) != 0)
            {
                dbg_event("**** Receiver Error, Bad CRC Checksum\n");
                /* 如果之前没发过 NAK，那么就要发一个 NAK */
                if (no_nak)
                {
                    send_data(FRAME_NAK, 0, frame_expected, out_buf);
                }
                break;
            }
            /*if (f.kind == FRAME_ACK)
                dbg_frame("Recv ACK  %d\n", f.ack);*/

            /*
               如果是数据帧，则检查帧序号，落在接收窗口内接收，否则丢弃
               如果不等于接收窗口下限，且没发过 NAK，还得发一个 NAK
            */
            if (f.kind == FRAME_DATA)
            {

                if ((f.seq != frame_expected) && no_nak)
                    send_data(FRAME_NAK, 0, frame_expected, out_buf);
                else
                    /* 无法捎带，则单发 ACK */
                    start_ack_timer(ACK_TIMER);

                if (between(frame_expected, f.seq, too_far) &&
(arrived[f.seq % NR_BUFS] == false))
                {
                    dbg_frame("Recv DATA %d %d, ID %d\n", f.seq, f.ack,
*(short*)&(f.data).info);
                    /* 这个“椅子”有人坐了，不再空闲 */
                    arrived[f.seq % NR_BUFS] = true;
                    /* 信息写入该椅子 */
                    in_buf[f.seq % NR_BUFS] = f.data;
                    /* 确保按次序依次上交，当 frame_expected 这把椅子为 true 时才上
交 */
                    while (arrived[frame_expected % NR_BUFS])
                    {
                        put_packet(in_buf[frame_expected % NR_BUFS].info, len
- 7);
                        no_nak = true;
                        /* 让接收端的该椅子仍然可以接受数据 */
                        arrived[frame_expected % NR_BUFS] = false;
                        /* 接收窗口上限、下限 +1 */
                        inc(too_far);
                        inc(frame_expected);

                        start_ack_timer(ACK_TIMER);
```

```
                }
            }
        }

        /* 如果是 NAK 帧，则选择重传 */
        if ((f.kind == FRAME_NAK) && between(ack_expected, (f.ack + 1) %
(MAX_SEQ + 1), next_frame_to_send))
        {
            dbg_frame("Recv NAK DATA %d\n", f.ack);
            send_data(FRAME_DATA, (f.ack + 1) % (MAX_SEQ + 1),
frame_expected, out_buf);
        }


        /* 检查确认序号，落在发送窗口内则移动发送窗口，否则不做处理 */
        while (between(ack_expected, f.ack, next_frame_to_send))
        {
            nbuffered--;
            /* 收到正确确认，停止计时器 */
            stop_timer(ack_expected % NR_BUFS);
            inc(ack_expected);
        }
        break;

        /* ACK 超时，就单发一个 ACK */
    case ACK_TIMEOUT:
        dbg_event("******** DATA %d timeout *********\n", arg);
        send_data(FRAME_ACK, 0, frame_expected, out_buf);
        break;

        /* 超时 选择重传*/
    case DATA_TIMEOUT:
        dbg_event("---- DATA %d timeout\n", arg);
        if (!between(ack_expected, arg, next_frame_to_send))
            arg = arg + NR_BUFS;
        send_data(FRAME_DATA, arg, frame_expected, out_buf);
        break;
    }

    /* 内存没满，继续进行 */
    if (nbuffered < NR_BUFS && phl_ready)
        enable_network_layer();
    /* 内存满了，停止 */
    else
        disable_network_layer();
  }
}
```