



哈尔滨工业大学
Harbin Institute of Technology

计算机网络 课程实验报告

实验名称	可靠数据传输协议-停等协议, GBN, SR					
姓名	陈一帆		院系	软件工程		
班级	1937102		学号	1191000606		
任课教师	李全龙		指导教师	李全龙		
实验地点	格物 213		实验时间	2021.11.06		
实验课表现	出勤、表现得分(10)		实验报告 得分(40)		实验总分	
	操作结果得分(50)					
教师评语						

实验目的：

理解可靠数据传输协议，掌握停等协议，GBN，SR协议的内容，并基于UDP编程实现

实验内容：

1. 基于UDP设计一个简单的停等协议，GBN协议，SR协议，实现单向的可靠数据传输协议，即服务器到客户端之间的传输。
2. 模拟引入数据包的丢失，验证所写实验的有效性。
3. 改进你写的协议，实现双向的数据传输。
4. 实现文件传输。

实验过程：

以文字描述、实验结果截图等形式阐述实验过程，必要时可附相应的代码截图或以附件形式提交。

1. GBN协议的编写

当GBN协议的发送端窗口缓存为1时，即为停等协议，因此本次实验仅实现GBN和SR。

a) 发送端

发送端运行前需要确认其监听/运行的IP地址加端口号，运行后首先完成相应的环境的配置，然后初始化SOCKET，绑定地址，然后便监听端口，此时以非阻塞的方式监听，即一段时间内接收不到用户发来的请求则返回-1，并不会一直等待，阻塞在此处。接收到用户发来的请求后，recvFrom函数会同时存储用户的地址。若为time，则返回服务器当前时间（来自于函数getCurTime），若为begin，表示此时需要给用户发送数据，在此过程中使用随机的方法判断发送的数据包是否丢失，如果判定为丢失，则不发送。发送数据的过程需要为数据包编号，不妨设发送端窗口大小为7，序号空间大小为20，即序号为0-19，一共32个数据包。为给数据包编号，构造的包的数据结构为

填充（一个字节）	序号（一个字节）	数据包大小（以1022字节分块）
----------	----------	------------------

由于序号可以为0，而在发送的过程中，第一个字节是不能为0的，因此在第一个字节处加一个填充字段，设为全一，然后便是序号字段，由于对位的操作繁琐，因此本实验中使用八个比特位来存储序号，其后则是数据包。

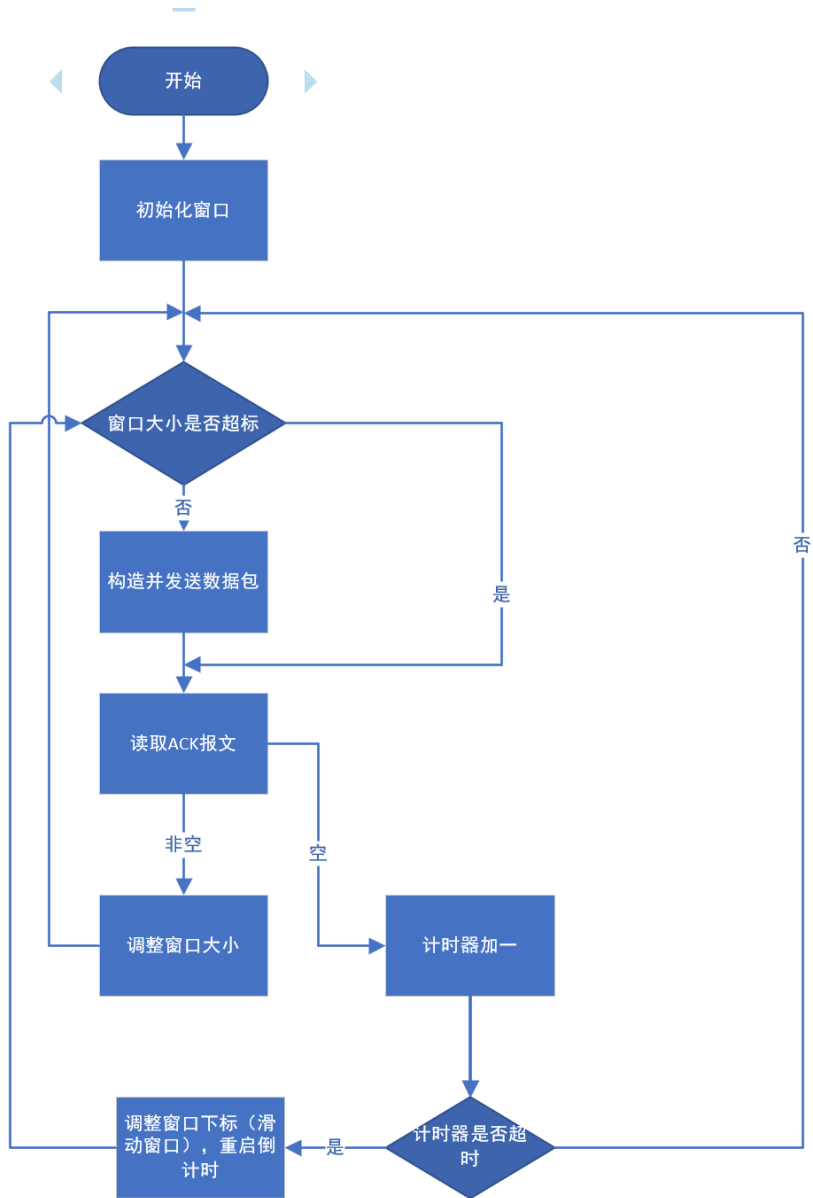
而返回的ACK数据结构中则如下所示。第一个字节为序号，第二个字节为结尾标识符。同样由于序号不能为0，因此在此序号加一。此处需要注意的是，当发送端发出的第一个数据包丢失（pkt0）的情况下，接收端并不知道返回多少，因此此时没有确认过的数据包，也就不能返回累计确认的ack0，因此本实验中设置第一个包不会丢失。

序号（一个字节）	“\0”（一个字节）
----------	------------

关于确认，丢失，窗口滑动的处理。

发送端有一个一定长度的发送窗口，只有序号在窗口内的数据包才会被发送，只有当收到在窗口内的ack时，窗口才会向前滑动，当窗口长期没有滑动的情况下，时钟会超时，此时窗口向后滑动，在程序中的表现则是窗口的前端下标向后滑到窗口的base处，然后再发送数据包，使得窗口前端下标离开base。

主要的流程图：



发送端主要的函数

seqIsAvailable()判断窗口大小是否超标

```
/*
Method:    seqIsAvailable
FullName:  seqIsAvailable
Access:    public
Returns:   bool
Qualifier: 当前序列号 curSeq 是否可用
*/
bool seqIsAvailable()
{
    int step;
    step = currentSeq - currentAck;
    step = step >= 0 ? step : step + SEQ_SIZE;
    //序列号是否在当前发送窗口之内
    if (step >= SEND_WIND_SIZE)
    {
        return false;
    }

    if (ack[currentSeq] == false)
    {
        return true;
    }

    return false;
}
```

TimeoutHandler()超时处理函数，主要是滑动窗口

```

/*
Method:    timeoutHandler
FullName:  timeoutHandler
Access:    public
Returns:   void
Qualifier: 超时重传处理函数，滑动窗口内的数据帧都要重传
*/
void timeoutHandler()
{
    printf("There is a time out\n");
    for (int i = 0; i < SEND_WIND_SIZE; ++i)
    {
        ack[(i + currentAck) % SEQ_SIZE] = false;
    }

    if (currentSeq > currentAck)
    {
        totalIndex -= (currentSeq - currentAck);
    }
    else if (currentSeq < currentAck)
    {
        totalIndex -= (currentSeq - currentAck + 20);
    }
    currentSeq = currentAck;
    printf("currentSeq->%d, currentAck->%d, totalIndex->%d\n", currentSeq, currentAck, totalIndex);
}

```

ackHandler()确认函数，同样是调整窗口的下标

```

void ackHandler(char c)
{
    unsigned char index = (unsigned char)c - 1; //序列号减一

    if (currentAck <= currentSeq && (currentAck <= index && index <= currentSeq))
    {
        printf("When window=[%d,%d) received ack%d\n", currentAck, currentSeq, index);
        for (int i = currentAck; i <= index; ++i)
        {
            ack[i] = TRUE;
        }
        ackNums += (index - currentAck + 1);
        currentAck = (index + 1) % SEQ_SIZE;
    }
    else if (currentSeq < currentAck && (index >= currentAck || index <= currentSeq))
    {
        printf("When window=[%d,%d) received ack%d\n", currentAck, currentSeq, index);
        if (index >= currentAck)
        {
            for (int i = currentAck; i <= index; ++i)
            {
                ack[i] = TRUE;
            }
            ackNums += (index - currentAck + 1);
            currentAck = (index + 1) % SEQ_SIZE;
        }

        if (index <= currentSeq)
        {
            for (int i = currentAck; i < SEQ_SIZE; ++i)

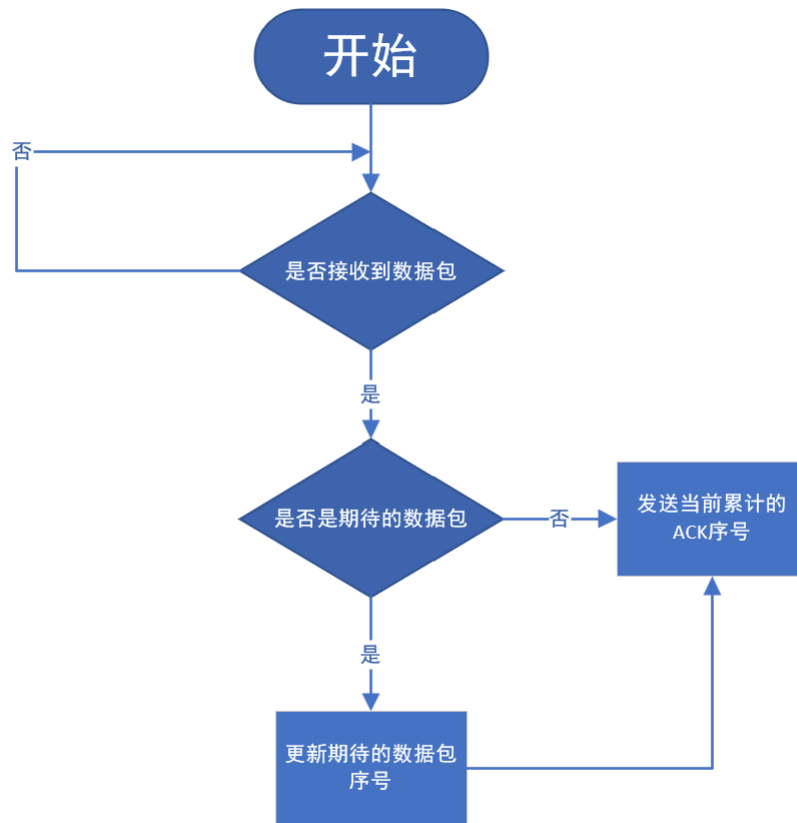
```

b) 接收端

在GBN中接收端并没有接受窗口，或者说接收端窗口大小为1，他仅仅维护一个exceptedSeq，如果发来的包的序号不等于exceptedSeq则不做任何处理，注意此时序号在

发来的段中第二个字节处，若是相等，则返回ack。注意关于ack中的序号，需要加一以防止第一个字节为0。

主要的流程图：

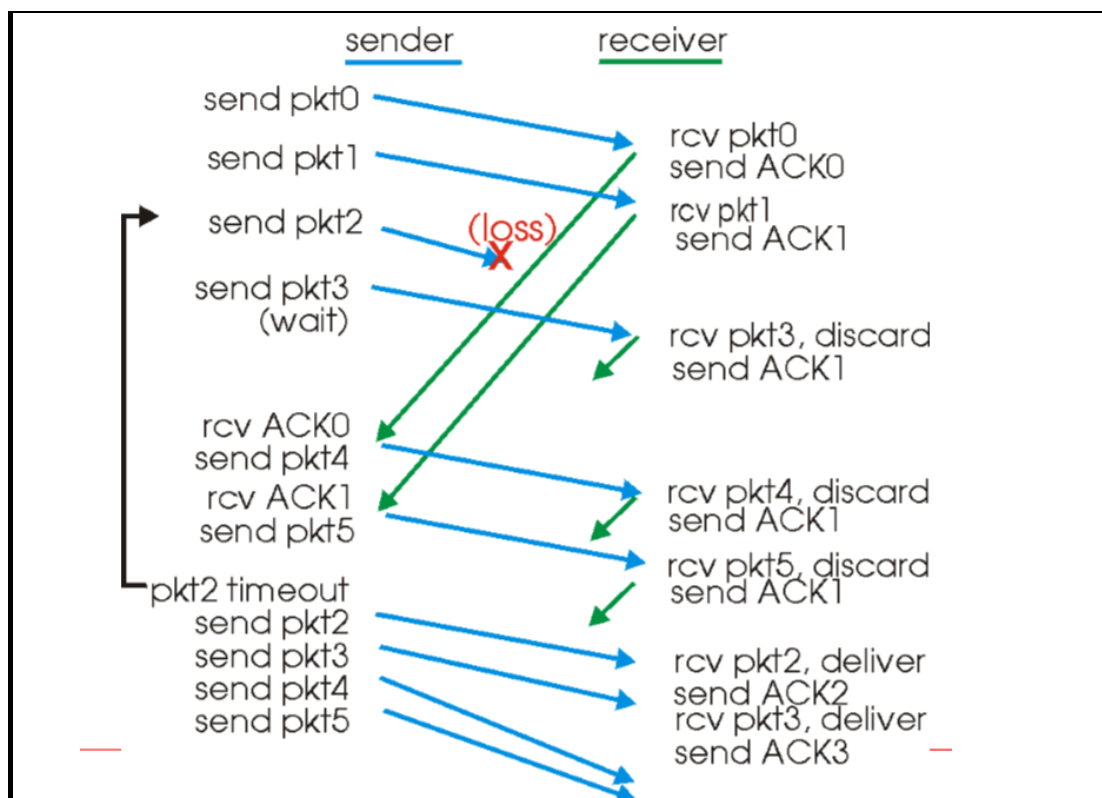


接收端用到的函数

getCurTime用户服务器端获取时间

```
/*
Method:      getCurTime
FullName:    getCurTime
Access:      public
Returns:     void
Qualifier:   获取当前系统时间，结果存入 ptime 中
Parameter:   char * ptime
*/
void getCurTime(char *ptime)
{
    char buffer[128];
    memset(buffer, 0, sizeof(buffer));
    time_t c_time;
    struct tm *p;
    time(&c_time);
    p = localtime(&c_time);
    sprintf(buffer, "%d/%d/%d %d:%d:%d",
            p->tm_year + 1900,
            p->tm_mon + 1,
            p->tm_mday,
            p->tm_hour,
            p->tm_min,
            p->tm_sec);
    strcpy(ptime, buffer);
}
```

发送端与接收端的交互功能:



c) 双向传输

关于双向传输，可以通过更改控制流的方法来实现，如下图所示。

```

    文件(F)  编辑(E)  选择(S)  查看(V)  转到(G)  运行(R)  终端(T)  帮助(H)  myGBN.cpp - lab2 - Visual Studio Code

    mySR.cpp  myGBN.cpp x
    main(int, char *[])
    {
        209
        210
        211 while (true)
        212 {
        213     printf("Please choose to be sender or receiver or quit\n");
        214     printf("eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000\n");
        215
        216     char cmdBuffer[CMD_LENGTH];
        217     ZeroMemory(cmdBuffer, sizeof(cmdBuffer));
        218
        219     char mode[10];
        220     ZeroMemory(mode, 10);
        221     char ip[20];
        222     ZeroMemory(ip, 20);
        223     int port;
        224     gets(cmdBuffer);
        225     int ret = sscanf(cmdBuffer, "%s %s %d", &mode, &ip, &port);
        226
        227 > if (!strcmp(mode, "receiver"))...
        337 > else if (!strcmp(mode, "sender"))...
        487 > else if (!strcmp(mode, "quit"))...
        492     else
        493     {
        494         printf("Sorry, I can't understand this command\n");
        495     }
        496
        497
        498     WSACleanup();
        499     return 0;
        500 }
    
```

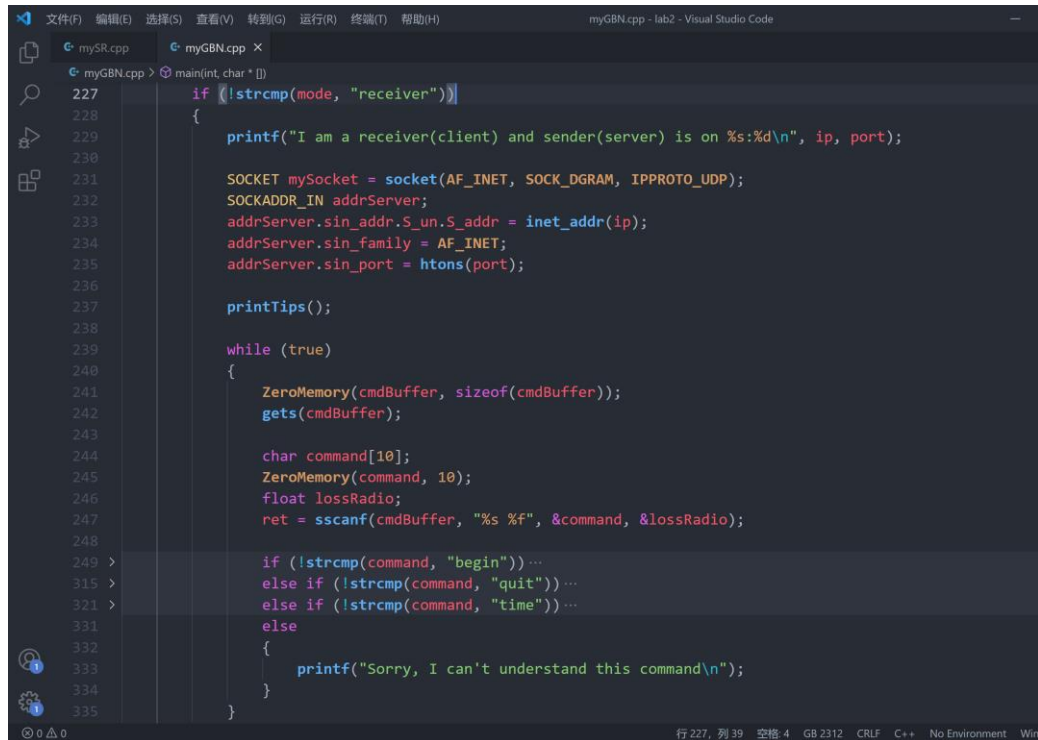
可以看到，程序会不断的读取命令行中的命令，命令可以是：

sender 127.0.0.1 8000表示进入发送模式，或者说此时程序是一个服务器，其绑定在127.0.0.1: 8000上，进入该模式后每处理完一条接收端发出的命令便询问是否要退出发送模式，然后便可以再次选择模式。

receiver 127.0.0.1 8000表示进入接收模式，此时程序是一个客服端，其会向127.0.0.1:8000处的服务器请求数据，为避免程序过于复杂，并不支持在选定地址后切换请求数据的地址，但是可以退出发送端模式后再次进入来选择地址。
quit退出，程序结束。

而在发送模式和接收模式内部，同样有处理各种命令循环和判断。

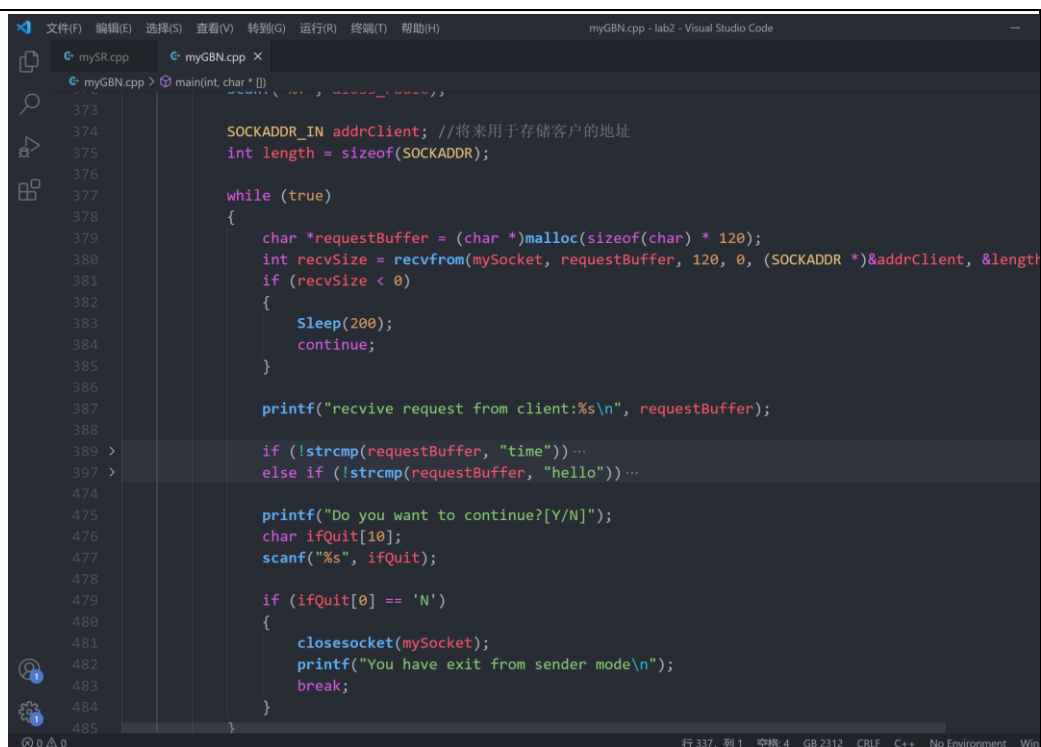
接收模式：



```
227 if (!strcmp(mode, "receiver"))
228 {
229     printf("I am a receiver(client) and sender(server) is on %s:%d\n", ip, port);
230
231     SOCKET mySocket = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
232     SOCKADDR_IN addrServer;
233     addrServer.sin_addr.S_un.S_addr = inet_addr(ip);
234     addrServer.sin_family = AF_INET;
235     addrServer.sin_port = htons(port);
236
237     printTips();
238
239     while (true)
240     {
241         ZeroMemory(cmdBuffer, sizeof(cmdBuffer));
242         gets(cmdBuffer);
243
244         char command[10];
245         ZeroMemory(command, 10);
246         float lossRadio;
247         ret = sscanf(cmdBuffer, "%s %f", &command, &lossRadio);
248
249         if (!strcmp(command, "begin"))...
315         else if (!strcmp(command, "quit"))...
321         else if (!strcmp(command, "time"))...
331         else
332         {
333             printf("Sorry, I can't understand this command\n");
334         }
335     }
```

其中begin为请求数据，quit为退出接收模式，time表示向服务器申请时间。

而在发送端，则为：



```
373
374
375     SOCKADDR_IN addrClient; //将来用于存储客户的地址
376     int length = sizeof(SOCKADDR);
377
378     while (true)
379     {
380         char *requestBuffer = (char *)malloc(sizeof(char) * 120);
381         int recvSize = recvfrom(mySocket, requestBuffer, 120, 0, (SOCKADDR *)&addrClient, &length);
382         if (recvSize < 0)
383         {
384             Sleep(200);
385             continue;
386         }
387
388         printf("recv request from client:%s\n", requestBuffer);
389
390         if (!strcmp(requestBuffer, "time"))...
391         else if (!strcmp(requestBuffer, "hello"))...
392
393         printf("Do you want to continue?[Y/N]");
394         char ifQuit[10];
395         scanf("%s", ifQuit);
396
397         if (ifQuit[0] == 'N')
398         {
399             closesocket(mySocket);
400             printf("You have exit from sender mode\n");
401             break;
402         }
403     }
404 }
```

如图所示,其中hello为对接收端请求数据的相应,而time是对发送端请求时间的响应,在处理完发送端的命令后,此时程序会询问是否要退出接收模式。

d) 文件传输

本实验中为了程序不过于复杂,仅准备了一个文件用于传输,发送端会在启动前便将文件读入内存之中,然后以字符数组的方法发给接收端,接收端会将提取收到的包中的数据,将之放入输出流中,待数据发送完之后便结束输出流。由此实现了文件的传输。

2. SR协议的编写

在SR协议中,发送端和接收端有了相同的窗口大小,双方都可以缓存数据。与GBN相比,仅ACK确认重传等机制发生了改变,其他数据结构等均未发生改变。

a) 发送端

发送端此时并不采用累计确认的方法,而是逐个确认的方法,超时重传仍然是通过滑动窗口的方式来实现,但是已经确认过的会做上标记不会再次重传。

b) 接收端

接收端由于接收窗口的大小不再为1,因此较为复杂。

首先是是否确认信号数组与缓存数组,因为缓存数组是一块内存,而内存是不支持滑动的,复制也较为复杂,因此使用空闲指针的方法,取两个指针分别指向空闲区域的头部,尾部,这样空闲区域便方便查找,同时接收端窗口确认数组里存储对应的包在缓存中的位置。

实验结果:

GBN文件传输：左边是发送端（服务器），右边是接收端（客户），两者是同一个程序编译而成。

Windows PowerShell	Windows PowerShell
<pre> Windows PowerShell 版权所有 (C) Microsoft Corporation。保留所有权利。 安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2> .\myGBN.exe Please choose to be sender or receiver or quit eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000 sender 127.0.0.1 8000 I am a sender(server) and run on 127.0.0.1:8000 Success to read data in 32 packages Please set the loss radio of sent packet:0.1 recvie request from client:time Do you want to continue?[Y/N]Y recvie request from client:hello Begin to send data send a packet with seq 0 When window=[0,1] received ack0 send a packet with seq 1 When window=[1,2] received ack1 send a packet with seq 2 send a packet with seq 3 When window=[2,4] received ack3 send a packet with seq 4 send a packet with seq 5 When window=[4,6] received ack5 send a packet with seq 6 send a packet with seq 7 When window=[6,8] received ack7 send a packet with seq 8 send a packet with seq 9 send a packet with seq 10 When window=[8,11] received ack10 send a packet with seq 11 When window=[11,12] received ack11 send a packet with seq 12 When window=[12,13] received ack12 send a packet with seq 13 Packet with seq 13 loss send a packet with seq 14 send a packet with seq 15 send a packet with seq 16 Packet with seq 16 loss send a packet with seq 17 send a packet with seq 18 send a packet with seq 19 There is a time out currentSeq->13, currentAck->13, totalIndex->13 </pre>	<pre> Windows PowerShell 版权所有 (C) Microsoft Corporation。保留所有权利。 安装最新的 PowerShell，了解新功能和改进！https://aka.ms/PSWindows PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2> .\myGBN.exe Please choose to be sender or receiver or quit eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000 receiver 127.0.0.1 8000 I am a receiver(client) and sender(server) is on 127.0.0.1:8000 You can input listed commands time to get current time quit to exit client begin [X] to begin and set loss radio time time from sender(server):2021/11/11 16:13:33 begin 0.1 return ack0 to sender(server) return ack1 to sender(server) Ack with seq 2 loss return ack3 to sender(server) Ack with seq 4 loss return ack5 to sender(server) Ack with seq 6 loss return ack7 to sender(server) Ack with seq 8 loss Ack with seq 9 loss return ack10 to sender(server) return ack11 to sender(server) return ack12 to sender(server) return ack12 to sender(server) return ack12 to sender(server) Ack with seq 12 loss return ack12 to sender(server) return ack12 to sender(server) return ack13 to sender(server) return ack13 to sender(server) return ack13 to sender(server) Ack with seq 13 loss return ack13 to sender(server) return ack13 to sender(server) return ack14 to sender(server) return ack15 to sender(server) return ack16 to sender(server) return ack16 to sender(server) return ack16 to sender(server) </pre>

GBN互传：

Windows PowerShell	Windows PowerShell
<pre> send a packet with seq 11 When window=[11,12] received ack11 All data is OK Do you want to continue?[Y/N]N You have exit from sender mode Please choose to be sender or receiver or quit eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000 Sorry, I can't understand this command Please choose to be sender or receiver or quit eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000 receiver 127.0.0.1 8000 I am a receiver(client) and sender(server) is on 127.0.0.1:8000 You can input listed commands time to get current time quit to exit client begin [X] to begin and set loss radio time time from sender(server):2021/11/11 16:15:18 begin 0.1 return ack0 to sender(server) return ack1 to sender(server) return ack2 to sender(server) return ack3 to sender(server) return ack4 to sender(server) return ack5 to sender(server) Ack with seq 6 loss Ack with seq 7 loss Ack with seq 7 loss return ack7 to sender(server) return ack7 to sender(server) return ack7 to sender(server) Ack with seq 7 loss return ack9 to sender(server) return ack8 to sender(server) return ack9 to sender(server) return ack10 to sender(server) return ack11 to sender(server) Ack with seq 12 loss return ack12 to sender(server) return ack12 to sender(server) return ack12 to sender(server) return ack12 to sender(server) return ack13 to sender(server) return ack14 to sender(server) return ack15 to sender(server) return ack16 to sender(server) Ack with seq 17 loss return ack18 to sender(server) </pre>	<pre> return ack10 to sender(server) return ack11 to sender(server) All data is received quit You have exit from sender mode Please choose to be sender or receiver or quit eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000 sender 127.0.0.1 8000 I am a sender(server) and run on 127.0.0.1:8000 Success to read data in 32 packages Please set the loss radio of sent packet:0.2 recvie request from client:time Do you want to continue?[Y/N]Y recvie request from client:hello Begin to send data send a packet with seq 0 When window=[0,1] received ack0 send a packet with seq 1 When window=[1,2] received ack1 send a packet with seq 2 When window=[2,3] received ack2 send a packet with seq 3 When window=[3,4] received ack3 send a packet with seq 4 When window=[4,5] received ack4 send a packet with seq 5 When window=[5,6] received ack5 send a packet with seq 6 send a packet with seq 7 send a packet with seq 8 Packet with seq 8 loss send a packet with seq 9 send a packet with seq 10 Packet with seq 10 loss send a packet with seq 11 When window=[6,12] received ack7 send a packet with seq 12 send a packet with seq 13 send a packet with seq 14 Packet with seq 14 loss send a packet with seq 15 send a packet with seq 16 Packet with seq 16 loss send a packet with seq 17 There is a time out currentSeq->8, currentAck->8, totalIndex->8 send a packet with seq 8 When window=[8,9] received ack8 </pre>

```

Windows PowerShell

Ack with seq 7 loss
return ack7 to sender(server)
return ack8 to sender(server)
return ack9 to sender(server)
return ack10 to sender(server)
return ack11 to sender(server)
Ack with seq 12 loss
return ack12 to sender(server)
return ack12 to sender(server)
return ack12 to sender(server)
return ack13 to sender(server)
return ack14 to sender(server)
return ack15 to sender(server)
return ack16 to sender(server)
Ack with seq 17 loss
return ack18 to sender(server)
return ack18 to sender(server)
return ack18 to sender(server)
return ack18 to sender(server)
return ack19 to sender(server)
return ack19 to sender(server)
Ack with seq 19 loss
return ack19 to sender(server)
return ack19 to sender(server)
return ack19 to sender(server)
return ack0 to sender(server)
return ack1 to sender(server)
return ack2 to sender(server)
Ack with seq 3 loss
return ack4 to sender(server)
Ack with seq 5 loss
return ack6 to sender(server)
return ack7 to sender(server)
return ack7 to sender(server)
Ack with seq 8 loss
return ack9 to sender(server)
return ack10 to sender(server)
Ack with seq 11 loss
return ack11 to sender(server)
All data is received
quit
You have exit from sender mode
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
quit
This program is over
PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2> |

send a packet with seq 5
send a packet with seq 6
send a packet with seq 7
Packet with seq 7 loss
send a packet with seq 8
Packet with seq 8 loss
There is a time out
currentSeq->0, currentAck->0, totalIndex->20
send a packet with seq 0
When window=[0,1] received ack0
send a packet with seq 1
When window=[1,2] received ack1
send a packet with seq 2
When window=[2,3] received ack2
send a packet with seq 3
send a packet with seq 4
When window=[3,5] received ack4
send a packet with seq 5
send a packet with seq 6
When window=[5,7] received ack6
send a packet with seq 7
When window=[7,8] received ack7
send a packet with seq 8
Packet with seq 8 loss
send a packet with seq 9
There is a time out
currentSeq->8, currentAck->8, totalIndex->28
send a packet with seq 8
send a packet with seq 9
When window=[8,10] received ack9
send a packet with seq 10
When window=[10,11] received ack10
send a packet with seq 11
There is a time out
currentSeq->11, currentAck->11, totalIndex->31
send a packet with seq 11
When window=[11,12] received ack11
All data is OK
Do you want to continue?[Y/N]N
You have exit from sender mode
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
Sorry, I can't understand this command
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
quit
This program is over
PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2>

```

SR传输文件:

```

Windows PowerShell

PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2> .\mySR.exe
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
sender 127.0.0.1 8000
I am a sender(server) and run on 127.0.0.1:8000
Success to read data in 32 packages
Please set the loss radio of sent packet:0.1
recvive request from client:time
Do you want to continue?[Y/N]Y
recvive request from client:hello
Begin to send data
Send packet0
When window=[0,4] received ack0
Send packet2
When window=[1,5] received ack2
Send packet3
When window=[1,5] received ack3
Send packet4
When window=[1,5] received ack4
Send packet5
When window=[1,5] received ack5
There is a time out
currentSeq->1, currentAck->1, totalIndex->1
Send packet1
There is a time out
currentSeq->1, currentAck->1, totalIndex->1
Send packet1
When window=[1,5] received ack1
Send packet7
When window=[6,10] received ack7
Send packet8
When window=[6,10] received ack8
Send packet9
When window=[6,10] received ack9
Send packet10
Packet10 loss
There is a time out
currentSeq->6, currentAck->6, totalIndex->6
Send packet6
When window=[6,10] received ack6
Send packet11
When window=[10,14] received ack11
Send packet12
Send packet13
When window=[10,14] received ack13
Send packet14
When window=[10,14] received ack14
There is a time out

PS C:\Code\ComputerNetwork\Computer-Networks-Lab-master\lab2> .\mySR.exe
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
receiver 127.0.0.1 8000
I am a receiver(client) and sender(server) is on 127.0.0.1:8000
You can input listed commands
| time to get current time |
| quit to exit client |
| begin [X] to begin and set loss radio |
time
time from sender(server):2021/11/11 16:20:51
begin 0.1
Recv packet0,currentRecv=0
return ack0 to sender(server)
Recv packet2,currentRecv=1
return ack2 to sender(server)
Recv packet3,currentRecv=1
return ack3 to sender(server)
Recv packet4,currentRecv=1
return ack4 to sender(server)
Recv packet5,currentRecv=1
return ack5 to sender(server)
Recv packet1,currentRecv=1
Ack1 loss
Recv packet1,currentRecv=6,has delievered
return ack1 to sender(server)
Recv packet7,currentRecv=6
return ack7 to sender(server)
Recv packet8,currentRecv=6
return ack8 to sender(server)
Recv packet9,currentRecv=6
return ack9 to sender(server)
Recv packet6,currentRecv=6
return ack6 to sender(server)
Recv packet11,currentRecv=10
return ack11 to sender(server)
Recv packet12,currentRecv=10
Ack12 loss
Recv packet13,currentRecv=10
return ack13 to sender(server)
Recv packet10,currentRecv=10
return ack14 to sender(server)
Recv packet10,currentRecv=10
return ack10 to sender(server)
Recv packet12,currentRecv=15,has delievered
return ack12 to sender(server)
Recv packet16,currentRecv=15
return ack16 to sender(server)

```

SR互传成功:

```
Windows PowerShell
Send packet11
When window=[11,15] received ack11
All data is OK
Do you want to continue?[Y/N]Y
recv request from client:time
Do you want to continue?[Y/N]N
You have exit from sender mode
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
Sorry, I can't understand this command
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
receiver 127.0.0.1 8000
I am a receiver(client) and sender(server) is on 127.0.0.1:8000
You can input listed commands
| time to get current time |
| quit to exit client |
| begin [X] to begin and set loss radio |
begin 0
Recv packet0,currentRecv=0
return ack0 to sender(server)
Recv packet2,currentRecv=1
return ack2 to sender(server)
Recv packet3,currentRecv=1
return ack3 to sender(server)
Recv packet4,currentRecv=1
return ack4 to sender(server)
Recv packet5,currentRecv=1
return ack5 to sender(server)
Recv packet6,currentRecv=6
return ack6 to sender(server)
Recv packet7,currentRecv=6
return ack7 to sender(server)
Recv packet8,currentRecv=6
return ack8 to sender(server)
Recv packet9,currentRecv=6
return ack9 to sender(server)
Recv packet10,currentRecv=6
return ack10 to sender(server)
Recv packet11,currentRecv=6
return ack11 to sender(server)
Recv packet12,currentRecv=11
return ack12 to sender(server)
Recv packet13,currentRecv=11
return ack13 to sender(server)
Recv packet14,currentRecv=11
return ack14 to sender(server)
Recv packet15,currentRecv=11

Recv packet7,currentRecv=11,has delivered
return ack7 to sender(server)
Recv packet11,currentRecv=11
return ack11 to sender(server)
All data is received
time
time from sender(server):2021/11/11 16:25:6
quit
You have exit from sender mode
Please choose to be sender or receiver or quit
eg.sender 127.0.0.1 8000 or receiver 127.0.0.1 8000
sender 127.0.0.1 8000
I am a sender(server) and run on 127.0.0.1:8000
Success to read data in 32 packages
Please set the loss radio of sent packet:0.1
recv request from client:hello
Begin to send data
Send packet0
When window=[0,4] received ack0
Send packet2
When window=[1,5] received ack2
Send packet3
When window=[1,5] received ack3
Send packet4
When window=[1,5] received ack4
Send packet5
When window=[1,5] received ack5
There is a time out
currentSeq->1, currentAck->1, totalIndex->1
Send packet1
When window=[1,5] received ack1
Send packet7
Packet7 loss
Send packet8
When window=[6,10] received ack8
Send packet9
When window=[6,10] received ack9
Send packet10
When window=[6,10] received ack10
There is a time out
currentSeq->6, currentAck->6, totalIndex->6
Send packet6
When window=[6,10] received ack6
There is a time out
currentSeq->7, currentAck->7, totalIndex->7
Send packet7
When window=[7,11] received ack7
Send packet12
```

问题讨论:

本次实验中由于继续采用C++, 因此遇到了很多困难, 但好在顺利解决, 由于代码报告因时间原因相隔距离较久, 因此实验过程过程了遇到的困难可能记载的不是很完全。

1. SR实现中前期认为接收端, 发送端窗口可以不相等, 后来才发现相等。
2. GBN实现过程中遇到ACK序号, 数据包序号编号的问题, 经常混淆从0开始与从1开始, 这也是数据包实现中, 在第一个字节头部没有将序号加一, 而是加入了填充的原因, 是为了避免加1导致的混淆。
3. 在窗口滑动的构成了遇到了窗口不滑动的问题, 多次观察输出序列, debug, 最好发现是变量命名过程中出现了问题。
4. 在SR窗口实现的过程中缓存是不能滑动的, 因此将确认窗口与缓存空间中并不是简单的对应关系。
5. 在SR中对于已经delievred的包收到后, 仍需要返回ACK
6. 实验指导书问题讨论:
 - a) UDP编程的特点: 较自由, 接收函数可以设置组设与非阻塞, 不需要调用特定的函数连接, 不需要握手, 可以通过recvFrom函数获取接收端的地址, 可以针对接收端的地址来进行操作

心得体会:

此次实验比第一次实验难度更大, 尤其是C语言编程的情况下, 并且由于这个程序运行需要同时有接收端和发送端, 因此debug比较困难, 由于编程过于复杂繁琐, 最后已经完全脱离GBN, SR的传输知识了, 更多的精力是放在索引下标的选取, 内存分布的处理上。