Computer Network – Homework 3 Report

Name: Calvin Liu (劉益瑋)
Student ID: B06902100

I. How to execute the program

1. Run “make” in terminal to compile the programs.
2. Then, these are the arguments for each program (run these programs in separate terminals).

   agent.c

   ```
   ./agent <sender IP> <recv IP> <sender port> <agent port> <recv port> <loss_rate>
   example:  ./agent local local 5000 5001 5002 0.15
   ```

   sender.cpp

   ```
   ./sender  <agent IP> <sender_IP> <sender port> <agent port>
   example:  ./sender local local 5000
   ```

   receiver.cpp

   ```
   ./receiver <agent IP> <receiver IP> <agent port> <recv port> <filename_output> <mode>
   example: ./receiver local local 5001 5002 output 1
   ```

   *set mode to 1 if user wants to play the video file. If file received is not a video file, error(s) might occur, so only set mode to 1 if you are going to send a video file from sender. For any other value of mode, the program will not play the video.
   <filename_output> will be the name of the file.

3. After running above commands, in the terminal of ./sender, user is asked to give the input for command and filename. The command is either “send” or “play”, any other command is invalid. The filename is the name of the file (inside the sender_file) that you want to send or play.

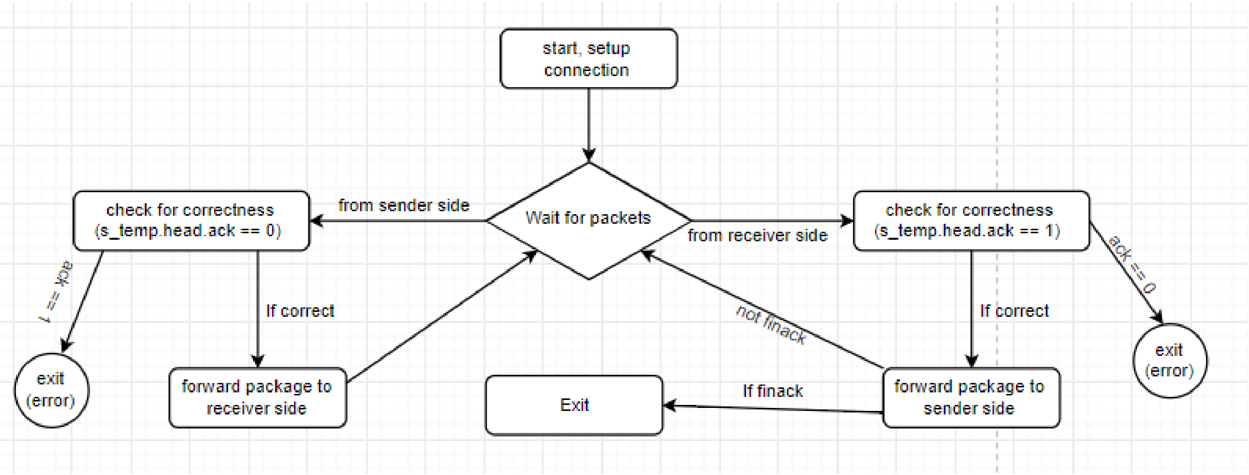
II. Notes regarding the program

For some reasons that I do not know of, my sender side often timeout (so the threshold and window size are perpetually low). However, file transmission works correctly (the content of the file received in the receiver_file is identical with the content of the file sent from the sender_file) and I think I have theoretically implemented Go-Back-N protocol and congestion control correctly.

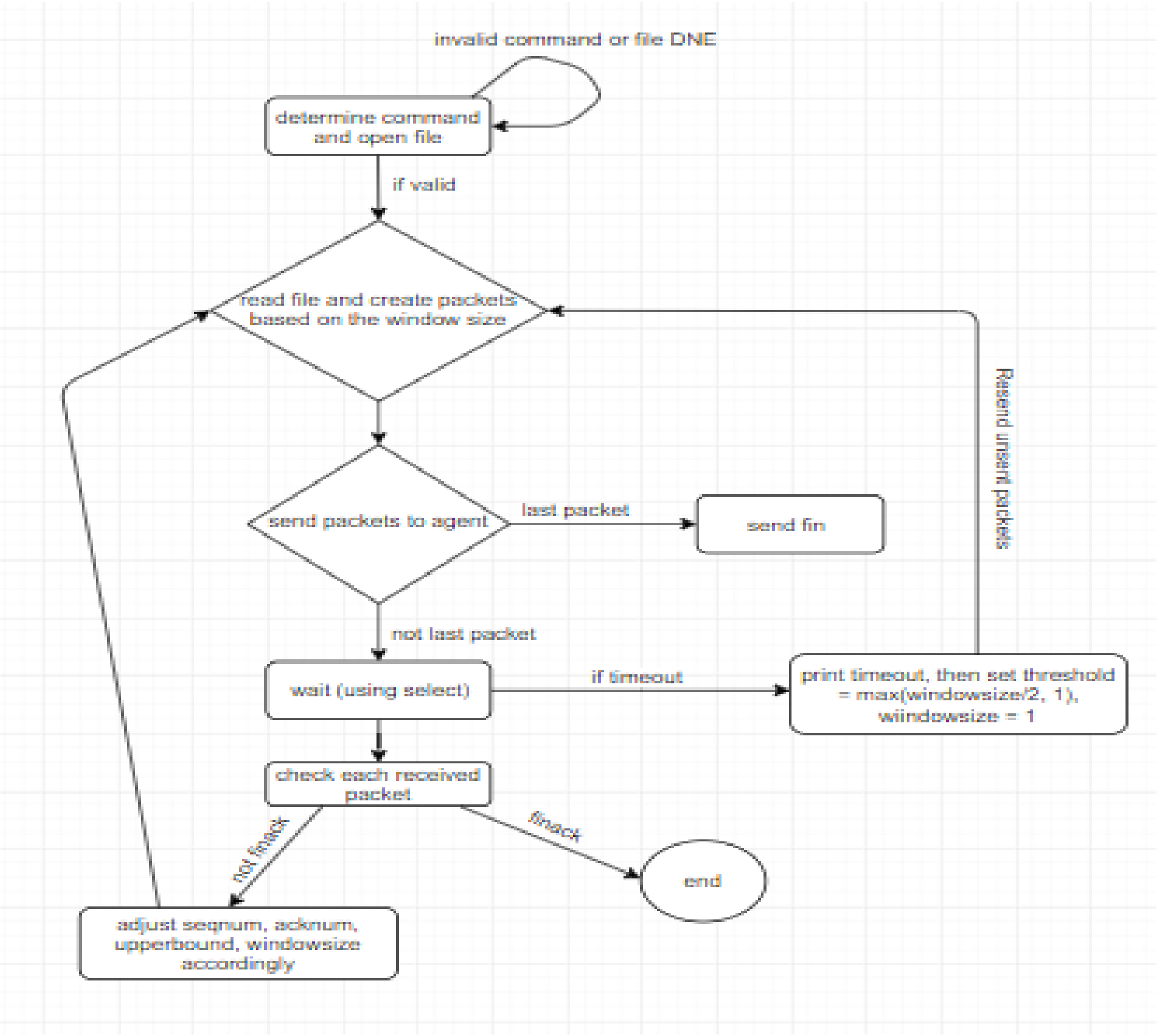
III. Flowcharts of sender, agent, and receiver

Some details of the code are omitted in these flowcharts, but the general idea of the code should still be understandable after looking at the flowcharts.

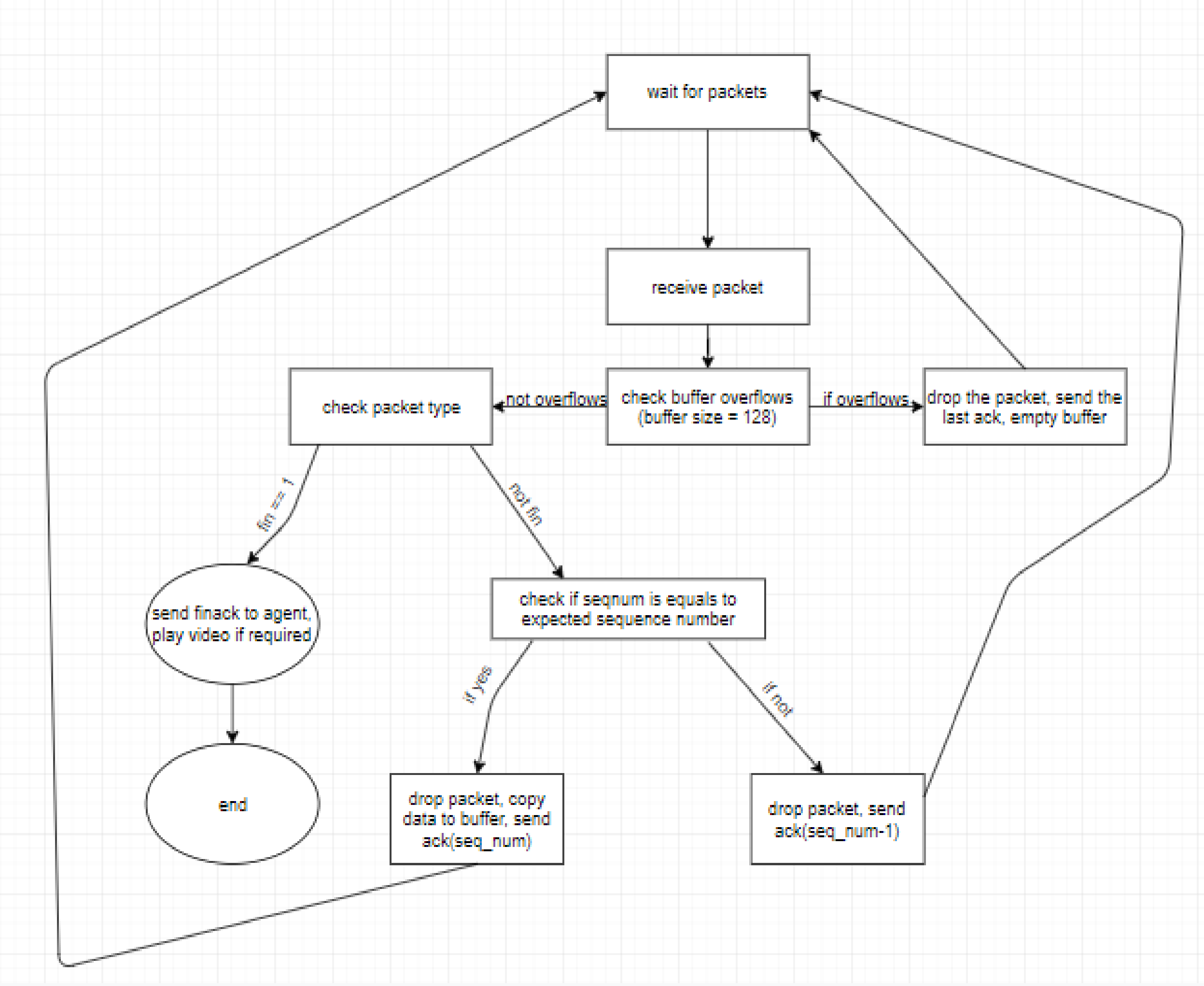
1. agent.c



2. sender.cpp


invalid command or file DNE
determine command and open file
if valid
read file and create packets based on the window size
Resend unsent packets
send packets to agent
last packet
send fin
not last packet
wait (using select)
if timeout
print timeout, then set threshold = max(windowsize/2, 1), wiindowsize = 1
check each received packet
finack
end
not finack
adjust seqnum, acknum, upperbound, windowsize accordingly

3. receiver.cpp



wait for packets
receive packet
check packet type
not overflows
check buffer overflows (buffer size = 128)
if overflows
drop the packet, send the last ack, empty buffer
fin == 1
not fin
send finack to agent, play video if required
check if seqnum is equals to expected sequence number
end
if yes
if not
drop packet, copy data to buffer, send ack(seq_num)
drop packet, send ack(seq_num-1)