11210EECS302002 Introduction to Computer Networks Lab3 Bonus

111062613 蔡鎮宇

一、 server.c

主要的改變會在 server.c，因為我會用 pthread 來加速。selective_repeat()負責 multi-thread 管理，create 4 個 thread 操作 sendFile()。而 sendFile()中透過檢查 sliding window 來決定是否要送 packet (sendPacket() function)。Main thread 在 create 完後，負責 recvAck()接收傳回來的 Ack。Sliding window 的資訊會被記錄在 global struct 的 Cwnd 中，若要更改時，需要用到 mutex lock。Cwnd 中記錄了 send_base，還有 available，亦即現在該送出的 seq 號碼。

(i) selective_repeat()

裡面會 create 4 threads，接著就 call recvAck()，最後 join 其他 thread。

(ii) sendFile()

要用 check_available()檢查現在可以被送出的 seq 是否在 Cwnd 的範圍內，如果是的話，就啟用 sendPacket()，若非，則要繼續跑 while loop 等待 sliding window 的 send_base 增加。

(iii) sendPacket()

如同 lab3，要去檢查是不是最後一個要送出的封包了。接著因為要送出封包，所以要有 timer 紀錄。這邊用 clock_gettime()和 struct timespec。因為 timespec 中的 tv_nsec 代表 nanosecond，所以 100 ms 相當於 10^8 nanoseconds。這邊有一個 global boolean 的 array Ackfield 紀錄某個 seq 是否被 Acked 了，因此會有個 while loop 檢查，若在時限內 Ackfield[seq] 沒有變 true 就是 timeout 了，會重新開啟 timer，然後重送封包。

(iv) recvAck()

這邊用 ack_count 來記錄總共成功收了幾個 Ack，當作 while loop 條件。若收到的 Ack 在 sliding window 內，則會將 Ack_field 改成 true，讓 sendPacket()的 thread 知道成功傳送了。如果 Ack_num 是現在的 send_base，代表 sliding window 可以前進，所以會用 while loop 來推進。而若有其他無關 sliding window 的 Ack，我們都 ignore 掉。

二、 client.c

這邊只有用一個 thread 來接收封包、回送 Ack 和管理 sliding window，用 recvFile()。寫入封包則和 lab3 的 writeFile()相同。

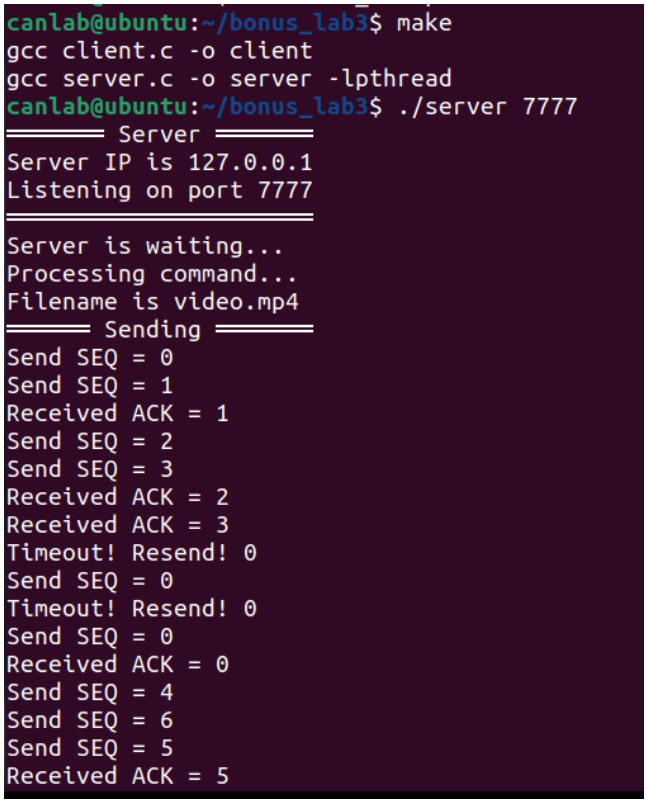
recvFile():

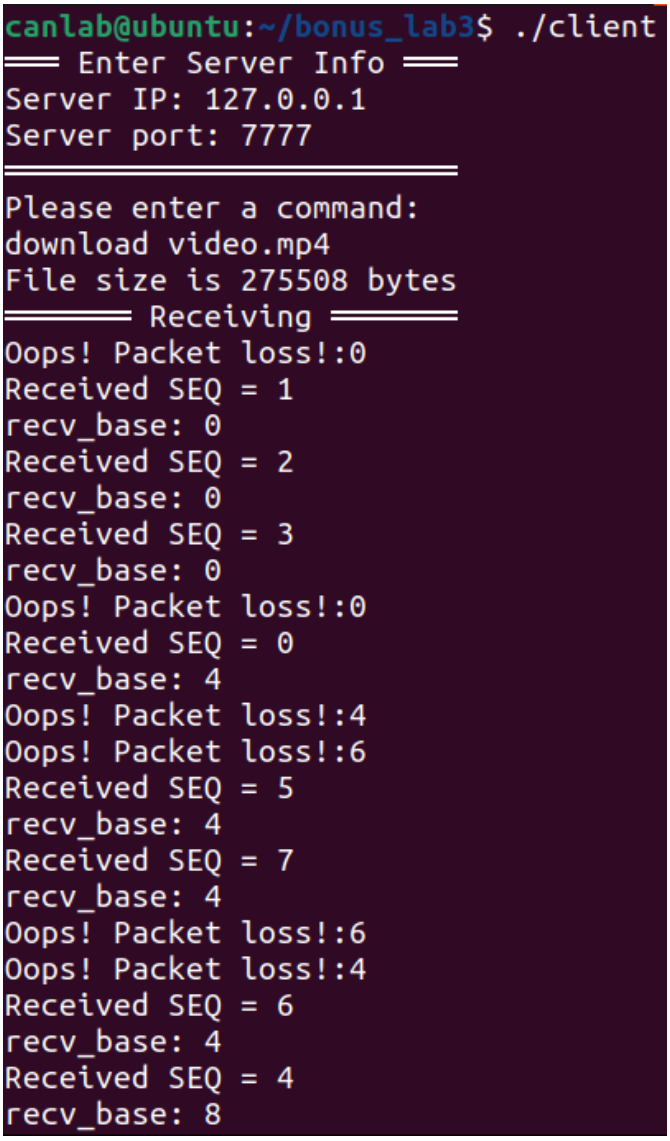
1. recvfrome() 從 server 端收封包，接著用 isLoss()模擬掉封包。
2. 封包的 seq 若為[recv_base-N, recv_base-1]，回傳 Ack。（N 為 window size）
3. seq 若在[recv_base, recv_base+N-1]，sendAck()，然後把把它設定為已 Acked，在這邊有個 global Boolean array，Recv_field[seq]設為 true，然後寫入 buffer。如果 seq == recv_base，則用 while loop 推進 recv_base。而對於其他的 sequence number，我們 ignore 掉即可。

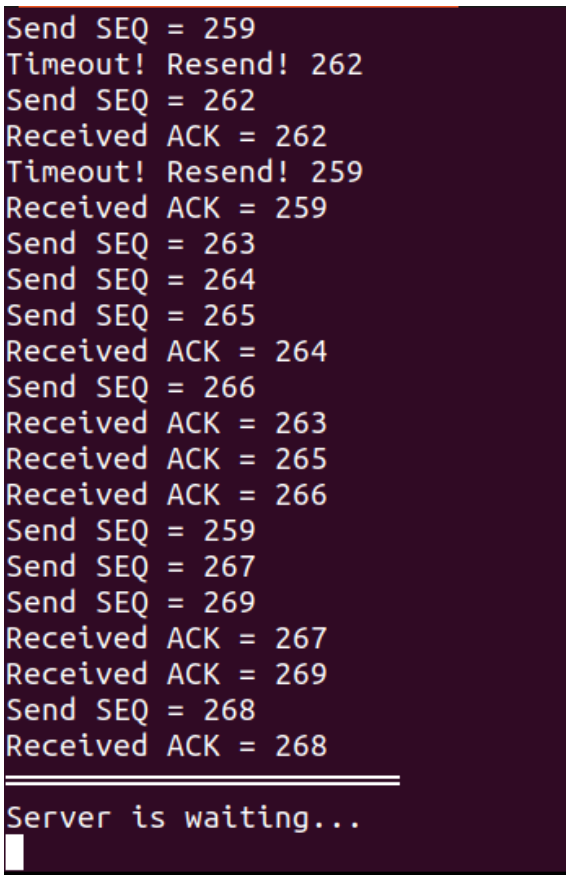
三、what I learned

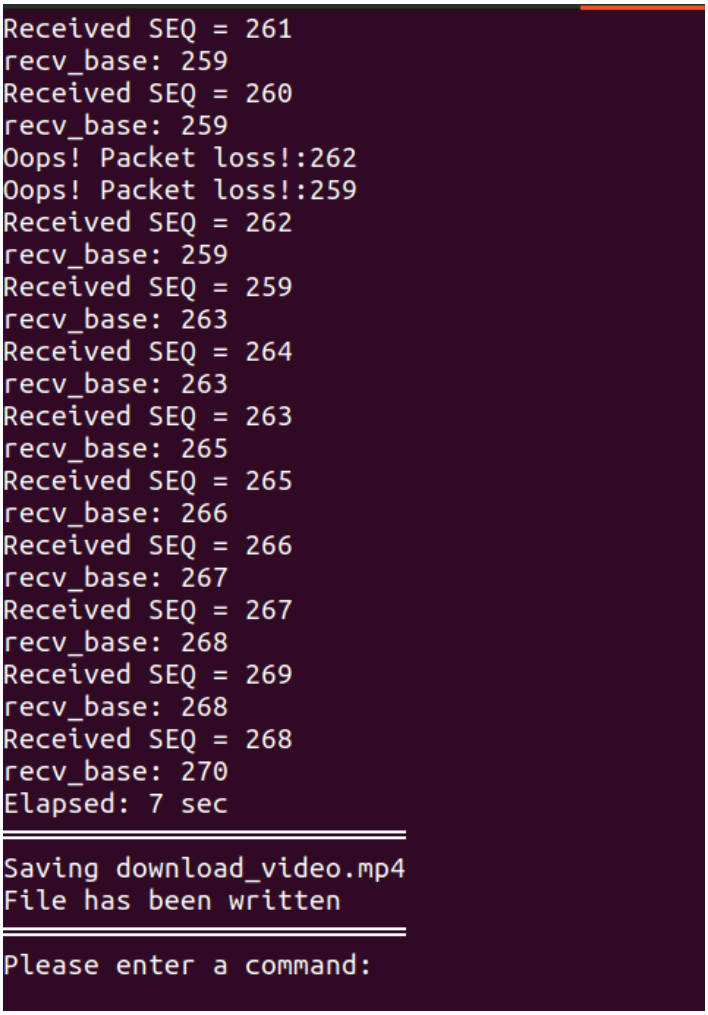
相較於 lab3，selective repeat protocol 真的比較難設計，在這邊我用了 4 個 thread，卻只有得到將近 2x 的 speedup，顯示還有地方可以改善。原本我想要在 client.c 也使用 multi-thread 來做，但是卡在 recvfrom()不知道要怎麼終止，而且測下去好像頂多快個一秒，所以就放棄了。 或許可以用一個 thread 做 recvfrom()，然後再找其他 thread 去做 sendAck()和寫入 buffer 是個更可行的方法。

四、 Screenshot






Send SEQ = 259
Timeout! Resend! 262
Send SEQ = 262
Received ACK = 262
Timeout! Resend! 259
Received ACK = 259
Send SEQ = 263
Send SEQ = 264
Send SEQ = 265
Received ACK = 264
Send SEQ = 266
Received ACK = 263
Received ACK = 265
Received ACK = 266
Send SEQ = 259
Send SEQ = 267
Send SEQ = 269
Received ACK = 267
Received ACK = 269
Send SEQ = 268
Received ACK = 268
Server is waiting...


Received SEQ = 261
recv_base: 259
Received SEQ = 260
recv_base: 259
Oops! Packet loss!:262
Oops! Packet loss!:259
Received SEQ = 262
recv_base: 259
Received SEQ = 259
recv_base: 263
Received SEQ = 264
recv_base: 263
Received SEQ = 263
recv_base: 265
Received SEQ = 265
recv_base: 266
Received SEQ = 266
recv_base: 267
Received SEQ = 267
recv_base: 268
Received SEQ = 269
recv_base: 268
Received SEQ = 268
recv_base: 270
Elapsed: 7 sec
Saving download_video.mp4
File has been written
Please enter a command: