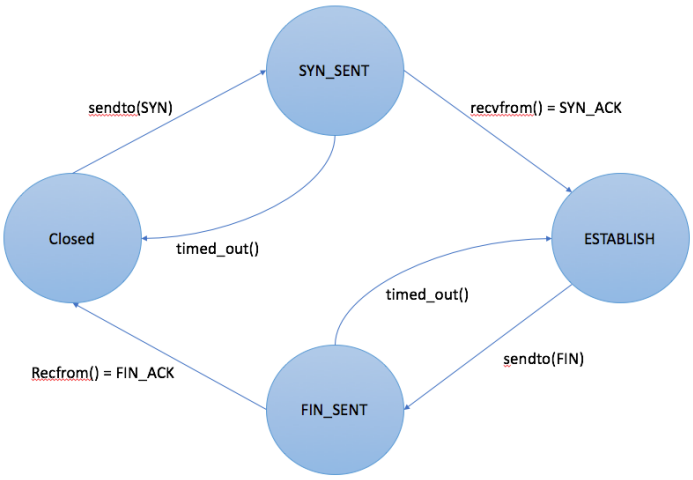
CS5450: Computer Networks Fundamentals Go-Back-N

I will explain the protocol from the view of both the client and server side and elaborate on each state since the logic may get convoluted visually.

Client Side FSM Diagram



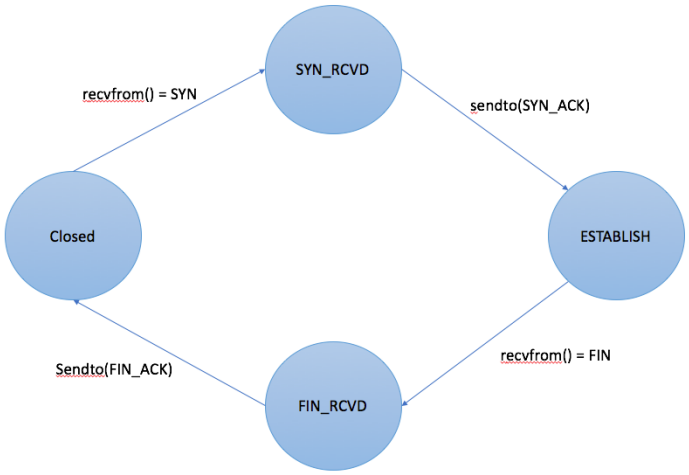
- **CLOSED:** This is where our TCP protocol will always be initiated as upon startup and end. After starting the program, the client will issue packet to the server with the **SYN** type set. Right after sending it, it will move the **SYN_SENT** state.

- **SYN_SENT:**
    - If the client is unable to receive a **SYN_ACK** packet type back, it will time out within 1 seconds and retrieve back to the **CLOSED** state.
    - If the client is able to receive a **SYN_ACK** packet type, it will proceed to become **ESTABLISH.** *In my protocol, I only use a two-way handshake due to the fact that the sender is the only one sending messages and the server is the only one that sends ACK messages.*

- **ESTABLISH:** The client side implementation is done following the book, "Computer Network, A Top-Down Approach." The client starts in slow mode to send one packet over to the server. If it is in fast mode, it will send two **DATA** type packets at once. After sending the packets, the client will set a single timer to wait for **DATAACK** results to come back. This timer is reset whenever a packet is received. The following logic is how these packets are processed when recvfrom() returns:
    - If it was interrupted by an alarm, the client will resend the **DATA** packets over to the server again. The protocol will retreat back to slow mode and send the first non-ACK'ed packet over.
    - If the packets received are outside of the window range it is discarded
    - If packets are in windows range, they are assumed to be a *cumulatively acknowledged.*

    - If all of the expected packets are not received in time, the window is updated to reflect the last ACK'ed packets.
- **FIN_SENT:** Once the client is finished reading the file, it will send a **FIN** type packet over to the server. After this, it will wait for the **FIN_ACK** package and return to the initial **CLOSED** state.

Server Side FSM Diagram



- **CLOSED:** In this state, the server is essentially waiting for any **SYN** packets from any client in order to transition to **SYN_RCVD.**
- **SYN_RCVD:** When the server transitions to this stage, it sends a **SYN_ACK** packet back to the server and goes to the **ESTABLISHED** state.
- **ESTABLISHED:** In the established state, the server is essentially processing any **DATA** packets it receives**.** The logic is as follows:
    - If the received packet is smaller or equal to the expected sequence number, it will ACK the sequence number back.
    - If the received packet is greater than the expected sequence number, it will ACK the number of the last received **DATA** packet.
    - If the received packet is a **FIN** packet, it will transition to **FIN_RCVD.**
- **FIN_RCVD:** In this state, the server simply sends a **FIN_ACK** packet and transitions to the **CLOSED** state.

There are several challenges I faced in this project. The first was the connection and teardown mechanism as the program skeleton did not have an ACKing sequence number, so I confirmed with Yunhao that that a two-way handshake was acceptable in this case. The second challenge I had was error handling the sending and receiving mechanism with the protocol. When I sent my files to the regular sendto() function, it ran smoothly but when packet started dropping there was some situations when the sender and receiver became out of sync. It turned out that if my receiver needed to re-ACKed previous packets. Lastly, I had some issue as the timing out mechanism was taking too long. File transfers took too much time. I switch alarm for seitimer() to get a better resolution.

```
root@server:~/code# make clean
rm -f *.o sender receiver
root@server:~/code# make
gcc -Wall -ansi  -c sender.c
gcc -Wall -ansi  -c gbn.c
gcc -Wall -ansi  -c helper.c
gcc -Wall -ansi -o sender sender.o gbn.o helper.o
gcc -Wall -ansi  -c receiver.c
gcc -Wall -ansi -o receiver receiver.o gbn.o helper.o
root@server:~/code# ./test_files.sh
starting test
Tests/Giant1.test PASSED!
Source: 332867062313f4514828ef7ae69f5f66  Tests/Giant1.test = Dest:
332867062313f4514828ef7ae69f5f66  ./outfile
Took 12 seconds
Tests/Giant2.test PASSED!
Source: 8574c6aec5d2c227291b3e8470279117  Tests/Giant2.test = Dest:
8574c6aec5d2c227291b3e8470279117  ./outfile
Took 6 seconds
Tests/Giant3.test PASSED!
Source: d7de02f5fb313ed7c37380f160c0f2dd  Tests/Giant3.test = Dest:
d7de02f5fb313ed7c37380f160c0f2dd  ./outfile
Took 7 seconds
Tests/Giant4.test PASSED!
Source: 8e812e2f4ed95906841feea3922fcd8d  Tests/Giant4.test = Dest:
8e812e2f4ed95906841feea3922fcd8d  ./outfile
Took 47 seconds
Tests/Large1.test PASSED!
Source: 681e6062bd06d5e9633a0c69c86d8bb5  Tests/Large1.test = Dest:
681e6062bd06d5e9633a0c69c86d8bb5  ./outfile
Took 1 seconds
Tests/Large2.test PASSED!
Source: d641312886d971718fdb456f6a0f20f1  Tests/Large2.test = Dest:
d641312886d971718fdb456f6a0f20f1  ./outfile
Took 1 seconds
Tests/Large3.test PASSED!
Source: d641f0df52b8cc48d8e578e24cb63dc5  Tests/Large3.test = Dest:
d641f0df52b8cc48d8e578e24cb63dc5  ./outfile
Took 0 seconds
Tests/Large4.test PASSED!
Source: 3cdc0533308a6f3b34934e549c0a10ab  Tests/Large4.test = Dest:
3cdc0533308a6f3b34934e549c0a10ab  ./outfile
Took 5 seconds
Tests/Small1.test PASSED!
Source: 17a0380ddee52c736aaa5cf3746c09a9  Tests/Small1.test = Dest:
17a0380ddee52c736aaa5cf3746c09a9  ./outfile
Took 1 seconds
Tests/Small2.test PASSED!
Source: c4240372c55c20dd42be992fd9c7321f  Tests/Small2.test = Dest:
c4240372c55c20dd42be992fd9c7321f  ./outfile
Took 0 seconds
```

```
Tests/Small3.test PASSED!
Source: 0715daeeeaa041c07bf5297065ec8a87  Tests/Small3.test = Dest:
0715daeeeaa041c07bf5297065ec8a87  ./outfile
Took 1 seconds
Tests/Small4.test PASSED!
Source: 9cdadc2d3dd58494831538bb07e069a6  Tests/Small4.test = Dest:
9cdadc2d3dd58494831538bb07e069a6  ./outfile
Took 0 seconds
Tests/Tiny1.test PASSED!
Source: 0cc175b9c0f1b6a831c399e269772661  Tests/Tiny1.test = Dest:
0cc175b9c0f1b6a831c399e269772661  ./outfile
Took 0 seconds
Tests/Tiny2.test PASSED!
Source: 187ef4436122d1cc2f40dc2b92f0eba0  Tests/Tiny2.test = Dest:
187ef4436122d1cc2f40dc2b92f0eba0  ./outfile
Took 1 seconds
Tests/Tiny3.test PASSED!
Source: 36846677e3a8f4c0b16d8bdf8ef18608  Tests/Tiny3.test = Dest:
36846677e3a8f4c0b16d8bdf8ef18608  ./outfile
Took 0 seconds
Tests/Tiny4.test PASSED!
Source: c3fcd3d76192e4007dfb496cca67e13b  Tests/Tiny4.test = Dest:
c3fcd3d76192e4007dfb496cca67e13b  ./outfile
Took 0 seconds
```