CS456 Assignment 3

Arwin Tio(astio), 20393326

Design

For this assignment I decided to use Java for the supposedly easier to use socket library as well as memory management (or the lack thereof).

I created two base classes: Receiver and Sender, which have the basic functionality that is common to the Go-Back-N and Selective-Repeat protocols. In particular, I implemented things like argument parsing, reading and writing to files (recvInfo, channelInfo), and other reusable functions. The gbn and recv classes inherit from these. I also implemented a class called CS456Packet, which deals with interpreting the byte array contents of a packet, the inverse - serializing a byte array payload into the packet format as specified in the assignment. The CS456 class also holds some utility functions for printing logs (as specified in the assignment).

Another notable design made in this assignment was the need for multithreading in the Sender to be able to send packets and listen for ACKs at the same time. This caused some (thankfully minor) race conditions so some effort also had to be put into synchronization with locks.

Other than that - the protocol/algorithms follow the assignment specifications quite closely.

One problem I had due to time constraints is that the protocols won't be able to handle over 131 kilobytes (with SEQUENCE_MODULO 256 and 512 bytes per packet) as I was unable to handle the modulo rotations. I will try to finish it if time permits.

Testing

To test, I sent my assignment source code (can that be considered self-hosting? self replicating? haha) through channel.jar using both Go-Back-N and Selective-Repeat using varying degrees of channel delay and discard probability. The most severe conditions I put it through was 1000 milliseconds delay and and 0.80 discard probability. The makefile and code was tested on ubuntu1204-002.