Project_ 1

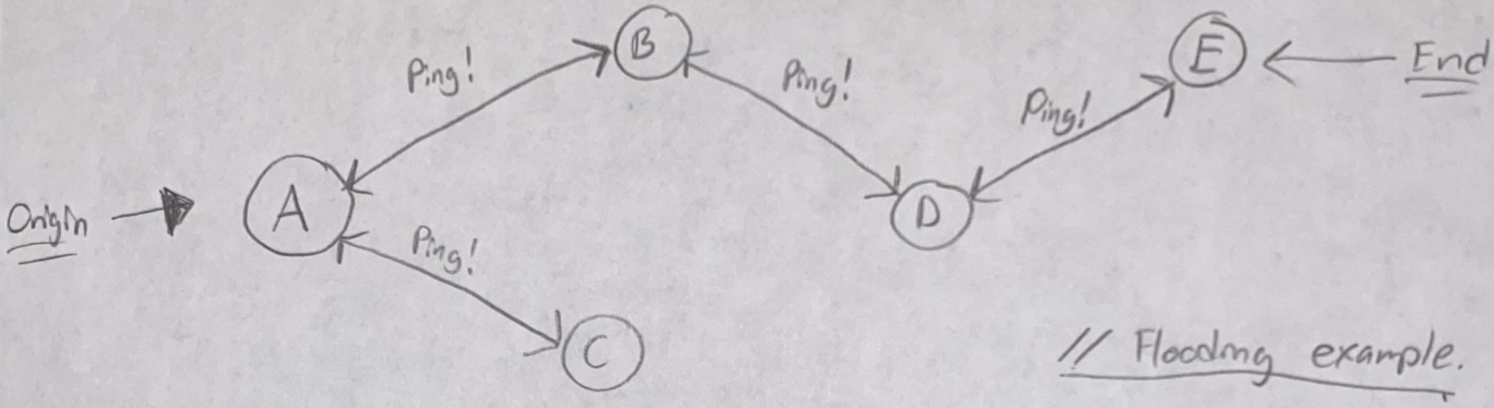
CSE 160, L2

Dale Alabastro & Shrithik Sareddy

Problem: We must send a packet to a recipient which has an unknown address. In order to get the packet there, we are going to need to use neighbor discovery, as well as flooding.

Flooding in this term means we will be sending a message out to every node that is connected to our origin node. After that, those nodes will do the same until it reaches its designated recipient.

For this project, we will be using TinyOS in the language of NesC, a sublanguage of C.



Some notes to keep track of is that neighbor discovery is built on flooding.

Our solution: To start this project, we first had to look at the skeleton code given to us in depth. In which, we devised a flow chart of sorts to show what was connected. This flow chart can be found in the .zip folder that was sent called "CSE160PR1DIAGRAM_PNG". In this chart, anything entering the left side of the files means that that specific file is calling another file. Anything leaving on the rightside means that that specific file is being called. Meanwhile, the python files technically encapsulates the whole thing in order to function properly. Of course, they are connected linearly. Also keep in mind, we need to install a header file called TOSSIM in order for this whole program to function appropriately.

In the code, we added into the interface "neighbor_discovery.nc" and in modules we added "neighbor_discoveryC.nc" and "neighbor_discoveryP.nc". From there, we wired all three files together. Our plan was to call flooding first. At the same time, while flooding runs, it calls neighbor discovery. What that means is that we start with flooding which will flood nodes with the assistance of neighbor discovery to get the neighboring nodes as well as checking if the target node has recieved the package.

Discussion Question

1A.) The pros of event driven programming is that it is easilly flexible, able to be changed quite well. It also allows hardware and sensors to easily interact with the software. Some downsides are that event driven programming is Complex at times as well as confusing. Even with simple programming the event driven programming is often complex and combursome.

2A.) With both mechanisms in place, the benefit of having both is that in long-terms, TTL will make sure that looping will cease and in short-term, the flooding mechanism will stop when it reaches its target destination. If we only had flooding checks, then when a loop occurs, there is nothing dropping the packets, thus the packet will go on infinitely. If we only had TTL checks, then the program may go on longer than necessary which is a waste of memory and time.

3A.) Best case situation in a flooding protocol is that we send /recieve 1 packet. Worst case situation is that we get n-1 packets sent and recieved. For best case, the source node and the target node are right next to each other. Therefore, it takes 1 packet. Worst case is when target node is the furthest possible node. Therefore, it requires to go through every node to get to the target.

4A.) In terms of efficiency, the better method could be finding the shortest path between the sorce node and the target node versus using neighbor discovery which floods all nodes. Thus we would free up more memory.

5A.) We originally came up with the idea of using a linked list idea. In terms of pros, time consumption would be lower than what we have now. For cons, we think memory would become an issue due to it dynamically allocating memory.