# Link State Routing Design

Every time a node discovers a change in its neighbors, it creates a LINK\_INFO\_PKT containing information about all its nodes, which has a separate sequence number, to prevent from wrapping around. It floods this packet to all other nodes. Recipients decrement the TTL and continue flooding to ensure everyone receives this packet. If the sequence number has been seen before it is ignored. When a valid LINK\_INFO\_PKT is received it is stored in a HashMap and the routing table must be recomputed. We can apply the Dijkstra's algorithms only if we have complete routing information - i.e. for every node on the network we must have link-state information. If it is not available we must wait until such information is available.

Once a routing table is computed, we can use this information to route information. For purposes of demonstration, we only transmit Ping / Ping reply packets once the routing table is complete. Alternatively, we could flood packets when routing information is unavailable to try to still deliver the packet, but it makes it a little harder to show that our routing implementation works. This is trivial, as we already demonstrated flooding for milestone one.

Each time we choose to send a packet, we look up the next hop in our routing table and send the packet to them. This recipient does the same until the packet arrives at its destination.

Occasionally a node may go offline. We need to ensure that all nodes are aware that he is offline and must remove his link state information. To do this we periodically make each node broadcast its linkstate information regardless of whether its neighbors have changed or not. If a node does not broadcast its linkstate information within a particular period of time, it is "timed out" and not considered for routing purposes until we see it again.

Nodes may name themselves with a string alias using the command “name <*alias>*”. The alias cannot consist entirely of numeric characters as it would cause some ambiguity on whether a user is trying to ping a particular node with node id "1234" or alias "1234" but node id = (something else) . We maintain two sets of HashMaps: from alias to nodeId and nodeId to alias.

The former is for identifying a node given an alias when a user wishes to ping using the alias. The latter is used for book-keeping, to update names, list the names of the nodes on the network.

Prevention of duplicate names is tiresome, as without a centralized authority it is hard to prevent users from both naming a node at the same time.