OLSR Functionality Report

Written by Matthew Cook and Stefan Stukelja

Optimized Link State Routing (OLSR) is a link-state based routing protocol, as opposed to the typical distance vector based routing protocols, whose initialization can be broken up into three sections: broadcasting, topology control, and building the routing tables. Following this initialization of all the information the protocol requires, packets are able to be sent from any source node to any valid destination node through the information obtained throughout the initialization process.

The purpose of the broadcast is to populate the two-hop neighbor tables of each node to expand their view of the network configuration. This is accomplished by having each node send its one-hop neighbor table to each other neighboring node and constructing a two-hop neighbor table for each neighboring node based off of the available one-hop neighbor tables. Once this is done, each node will have a larger view of the configuration of the network and the next step is to go through the topology control.

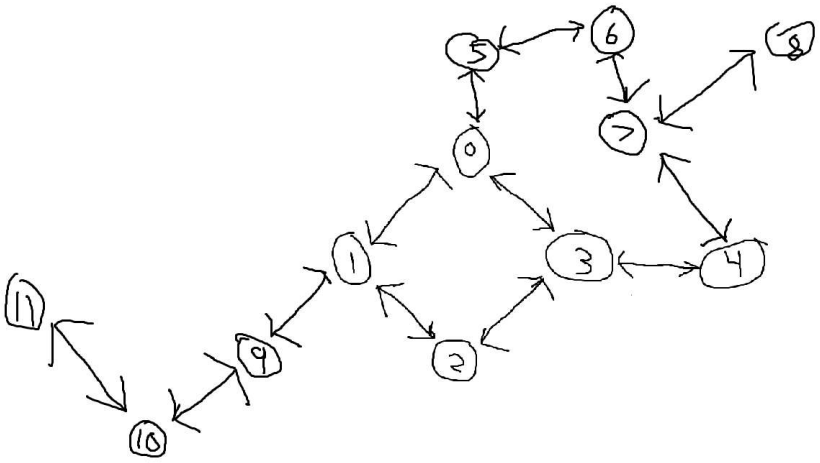
The topology control determines which nodes should be designated as Multi-Point Relays (MPR) which are the only nodes allowed to rebroadcast packets as they are being sent to their destination. This step is crucial to achieving a stable network as without MPRs a network would have to have each node repeatedly flood all neighboring nodes to transmit packets which would result in heavy power consumption by the network. As such, MPRs are chosen so there are as few as possible while still covering the whole network. The process for determining which nodes should be MPRs is divided into two steps which involve isolated nodes and nodes with many neighbors, respectively. Isolated nodes have only one one-hop neighbor and must have their sole neighbor set as a MPR so there is a possible route to the isolated node. Following this, nodes are considered in descending order by the amount of one-hop neighbors they have and if any one-hop or two-hop neighbors don’t have a neighboring MPR it is set as a MPR.

Routing tables can then be constructed by recursively traversing through MPRs to find each possible route with their sequences and pushing it into the routing table. To prevent duplicate or wasteful routes from being inside the table, each route must be checked against all current routes and be handled accordingly. Packets can then be sent throughout the network by building routes with the routing table. Packets being being sent to a destination which is not able to be reached can be checked as they will return false.

Deviation from Existing Research

Network systems that use Optimized Link State Routing Protocol, by default, have a shortest-hop algorithm to send packets from one node to another. This means that the nodes in the network will select the route in their routing table that will take the least amount of hops to get from the source to the destination. As a team, we modified the shortest-hop algorithm used to send packets so that it will take the remaining energy of the destination node's Multi-Point Relays into consideration. Our design takes the path with the shortest amount of hops unless a certain threshold of remaining energy is met by a destination node's nearest MPR. If the threshold is met, meaning that the node has lost a certain amount of its total energy, the source node attempts to search for the next closest route with more remaining energy. If such a route exists, the node uses the alternative route to get to its destination. The design that we have created increases overall lifespan of the network and allows it to function for a greater amount of time before the first node runs out of energy.

Results and Analysis



Pictured above is a physical representation of the layout of the small network used for testing purposes. The following test sets were used to show differences between the OLSR algorithm and OLSR with energy constraints:

Test

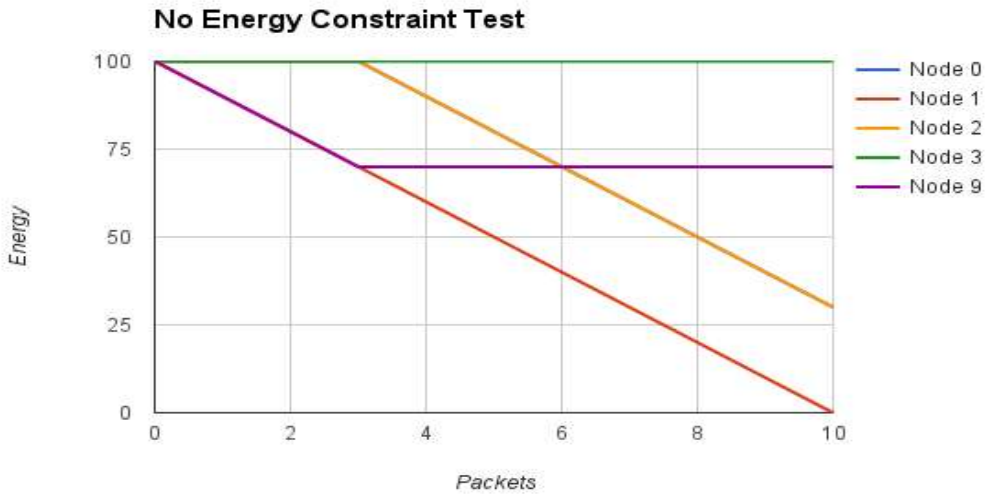
```
for(int i = 0; i < 3; i++)
{
   network->sendPacket(1, 9);
   network->checkNetworkPower();
}
for(int i = 0; i < 7; i++)
{
   network->sendPacket(0, 2);
   network->checkNetworkPower();
}
```

TestEnergy

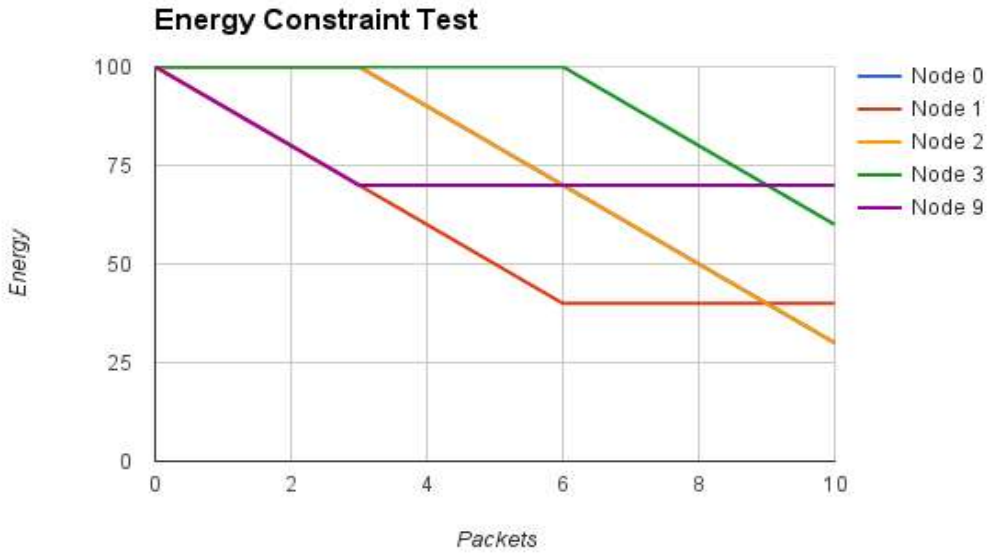
```
for(int i = 0; i < 3; i++)
{
   network->sendPacketEnergy(1, 9);
   network->checkNetworkPower();
}
for(int i = 0; i < 7; i++)
{
   network->sendPacketEnergy(0, 2);
```

```
    network->checkNetworkPower();
}
```

Note: sendPacket(x, y) sends a packet from node x to node y and prints out the intermediary nodes traversed to arrive to the destination. The current energy of each node is printed after each packet is sent with nodes that have died being removed from the network. Also, the third option in the program “Print Routing Table” demonstrates that all nodes have routes to build paths to each other non-neighboring node in the network. Sending packets to a neighboring node does not require to build a route and thus are not included in the routing table.



No Energy Constraint Test demonstrates how the OLSR algorithm will simply send packets through the shortest-hop path until failure.



Energy Constraint Test takes the energy of nodes into consideration and once the threshold of fifty energy is reached the protocol attempts to find routes with more energy to arrive at the destination. In this situation, the packets are sent through Node 3 instead of Node 1 to conserve the energy of Node 1.