CS 460: Routing

Kevin Haroldsen

1 Introduction

Routing is necessary for any large network of interconnected computers to function efficiently. There are many protocols that can be used to coordinate costs of sending from one node to the next; however, this lab will focus on the distance-vector algorithm.

2 Distance-Vector Algorithm

The distance-vector algorithm is a decentralized algorithm for computing best-cost paths from each node to each other node. Each node starts by knowing the costs to its own neighbors. Then, at determined intervals, each node sends a *distance vector* to each of its neighbors. This distance vector is the currently known best-cost paths to each other node from that node.

Upon receiving another distance vector, a node will compare its current best-cost path with the best-cost path of its neighbor, plus the cost to get to the neighbor. If the best-cost path of its neighbor, plus the cost to get to the neighbor, is less than its current best-cost path to a destination, it will set its distance vector to this value and add an entry in the forwarding table to use that link to forward to the destination.

2.1 Implementation Details

3 Experimentation

3.1 Five Nodes

3.2 Ring of Five Nodes

Here is a snippet of the output for five nodes that highlights its effectiveness.

```
0.002 dvr n3 -> n2 ({9: 1, 4: 0, 6: 1, 7: 0})
0.002 dvr n3 -> n4 ({9: 1, 4: 0, 6: 1, 7: 0})

== Sending packet n2 -> n3 ==
11.0 hop Link<(4) n2 -> n3>

== Sending packet n2 -> n4 ==
12.0 hop Link<(4) n2 -> n3>
12.001 hop Link<(6) n3 -> n4>

== Sending packet n2 -> n5 ==
13.0 hop Link<(3) n2 -> n1>
13.001 hop Link<(2) n1 -> n5>
```

```
== Sending packet n3 -> n1 ==
14.0 hop Link<(5) n3 -> n2>
14.001 hop Link<(3) n2 -> n1>

== Sending packet n1 -> n2 ==
96.0 hop Link<(1) n1 -> n2>

== Sending packet n1 -> n3 ==
97.0 hop Link<(1) n1 -> n2>
97.001 hop Link<(4) n2 -> n3>

== Sending packet n1 -> n4 ==
98.0 hop None

== Sending packet n1 -> n5 ==
99.0 hop None

== Sending packet n2 -> n1 ==
100.0 hop Link<(3) n2 -> n1>

...

== Sending packet n5 -> n1 ==
112.0 hop None

== Sending packet n5 -> n2 ==
113.0 hop None

== Sending packet n5 -> n3 ==
114.0 hop Link<(9) n5 -> n4>
114.001 hop Link<(7) n4 -> n3>

== Sending packet n5 -> n4 ==
115.0 hop Link<(9) n5 -> n4>
```

3.3 Fifteen Nodes