Simulation of Distance-Vector Routing Protocol using PEERSIM

Introduction

Distance vector routing is a simple distributed routing protocol. This protocol allows routers to automatically discover destinations which are reachable inside the network, as well as calculating the shortest path available to each of these destinations found. The shortest path is calculated based on a cost associated with every link[1]. Nodes broadcast and receive information from their directly attached neighbours, perform a calculation and distribute results back to their neighbours in an iterative process. This is in contrast to the link state routing protocol, which broadcasts to the entire network, rather than just to its neighbours. Examples of distance vector routing protocols include RIP (routing information protocol), IGRP (interior gateway routing protocol) and EIGRP (enhanced IGRP)[2].

Difficulties

The main difficulty I found with this project considering the distance vector routing protocol, was with implementing the Bellman-Ford algorithm in PEERSIM. Furthermore the PEERSIM configuration files were difficult to implement correctly, considering if certain aspects were not included or initialised properly, the simulation would not run at all.

Results

```
CDSimulator: starting simulation
[0] drawing ...
done.
CDSimulator: cycle 0 done
[1] drawing ...
done.
ND#  0->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  1->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  2->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  3->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  4->|   |
    cost|   |
     val|   |
-----------------------------------------
CDSimulator: cycle 1 done
```

```
[2] drawing ...
done.
ND#  0->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  1->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  2->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  3->|   |
    cost|   |
     val|   |
-----------------------------------------
ND#  4->|   |
    cost|   |
     val|   |
-----------------------------------------
CDSimulator: cycle 2 done
```

```
[3] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  3, |
-----------------------------------------
CDSimulator: cycle 3 done
```

```
[4] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
```

```
[5] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
```

```
[6] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,CDSimulator: cycle 5 done
  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
CDSimulator: cycle 6 done
```

```
[7] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
CDSimulator: cycle 7 done
```

```
[8] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
CDSimulator: cycle 8 done
```

```
[9] drawing ...
done.
ND#  0->|   0,  1,  2,  3,  4, |
    cost|   0,  5,  4,  1,  3, |
     val|   0,  1,  3,  3,  3, |
-----------------------------------------
ND#  1->|   0,  1,  2,  3,  4, |
    cost|   5,  0,  7,  4,  2, |
     val|   0,  1,  4,  4,  4, |
-----------------------------------------
ND#  2->|   0,  1,  2,  3,  4, |
    cost|   4,  7,  0,  3,  5, |
     val|   3,  3,  2,  3,  3, |
-----------------------------------------
ND#  3->|   0,  1,  2,  3,  4, |
    cost|   1,  4,  3,  0,  2, |
     val|   0,  4,  2,  3,  4, |
-----------------------------------------
ND#  4->|   0,  1,  2,  3,  4, |
    cost|   3,  2,  5,  2,  0, |
     val|   3,  1,  3,  3,  4, |
-----------------------------------------
CDSimulator: cycle 9 done
```

[1] http://cnp3book.info.ucl.ac.be/principles/dv.html

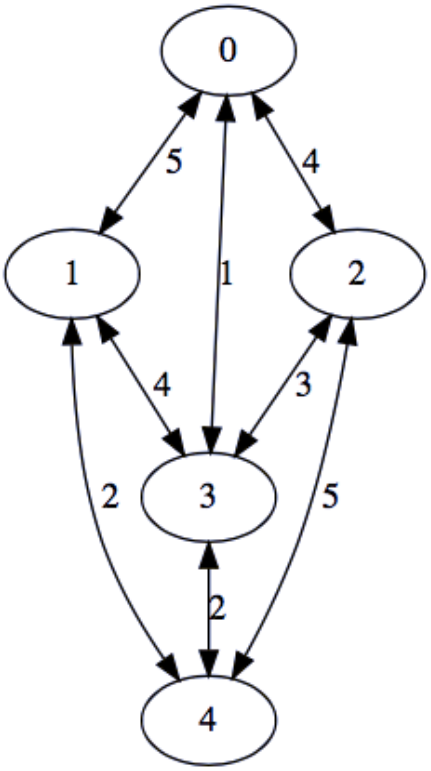
[2] http://www.linfo.org/distance_vector.html

Validation

In order to validate these results, they can be compared to the example given through the Link-State Routing Protocol given through the week 9 practical. Considering the 8th cycle shown above in the results is identical to the that of the 8th cycle in the link-state routing protocol, it can be assumed that the distance vector protocol has been implemented correctly.

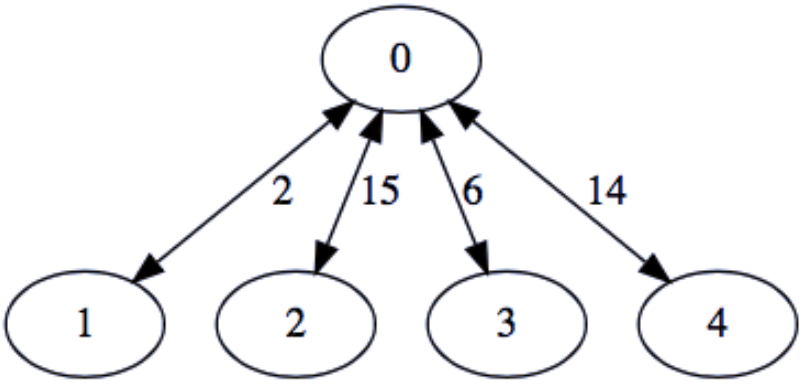
Graphs

In order to further validate the results, different graphs were constructed. Initially utilising the peersim.dynamics method WireKOut, the following graph can be visualised.

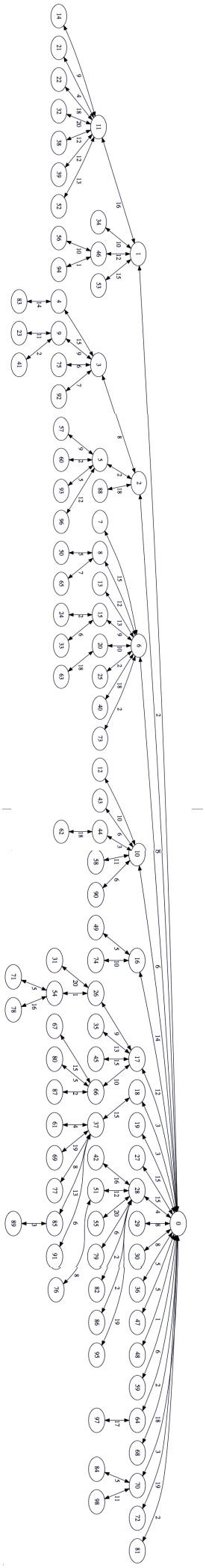


Furthermore, in order to contrast various topologies, a number of other graphs were produced by utilising the WireFromFile class in peersim.dynamics.WireGraph.

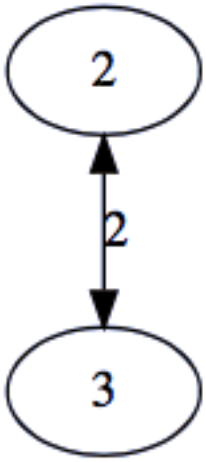
Star Graph



Internet Graph



Random Graph



Findings

In comparison to the Link State Routing Protocol, the Distance Vector Protocol is not as scalable. The distance vector routing protocol requires little management and can be considered efficient for a small network. However when considering a larger network, distance vector protocols tend to have poor convergence properties. As routers are not requires to maintain the state of all links in a network, distance vector protocols tend to consume less overhead at the expense of limited visibility. As routers will only have a limited view of the network, certain tools such as the split horizon technique and the poison reverse technique are needed in order to be able to prevent certain routing loops[3]. Contrasting this to the link state protocol, highlights that link state routing protocols will typically require more overhead than simply processing broadcasts from its direct neighbours single this protocol requires maintaining link state information for the entire area. Although higher overhead is a draw-back in the link state protocol, this provides much more robust operations and scalability[4]. Distance vector protocols may also create more traffic than link state protocols, since a hop count change must be propagated and processed on each router. These updates take place on a periodic basis, therefore if there are no changes in network topology, bandwidth-wasting broadcasts will occur. Furthermore for larger networks, distance vector routing results in a larger routing table than a link state protocol would create, since each router must know about all other routers. This can also tend to lead to congestion in WAN links[5].

Overall, this distance vector routing protocol is simpler to configure and easier to maintain than the link state routing protocol, however it suffers from a number of different drawbacks which do not make it suitable for use with large networks.

[3] http://www.infroref.org/i3230CE_Poison%20Reverse.htm

[4] http://www.ciscopress.com/articles/article.asp?p=24090&seqNum=3

[5] https://www.geeksforgeeks.org/computer-network-routing-protocols-set-1-distance-vector-routing/