Advanced Networks and Communications
Network Simulator

Velin Kerkov
2022691

12.03.2016

Overview

My system consists of

- Main - this is used to run the simulator. It accepts only one argument when ran - the simulator's configuration file name. Its only job is to instantiate the simulator class.
- Simulator - this is the main driver of the RIP simulation. It is configured through a configuration file. Its job is to set up the network and control when what is happening. It has the following configurations available:
    - -numOfNodes -> Integer parameter. Specifies how many nodes are there in the network. The indexing of the nodes always starts at 0
    - - maxExchanges -> Integer parameter with default 100. Specifies maximum number of "exchanges" in range [0, inf].
    - -untilStability -> String-boolean flag with default "false". If "true" the simulation will stop once stability is achieved in the network. If "false" the simulation will stop after -maxExchanges number of exchanges.
    - -manual -> String-boolean flag with default "false". If "false" system runs the simulation without any user interference. If "true" the system gives the following commands to the user:
        - "Enter" -> tells the system to simulate the next exchange
        - "split-horizon-on" -> turns split-horizon heuristic on for the next and any subsequent exchanges
        - "split-horizon-off" -> turns split-horizon heuristic off for the next and any subsequent exchanges
    - -splitHorizon -> String flag with default "off". If set to "on" split-horizon heuristic will be on when the simulation starts, otherwise split-horizon heuristic will be off when the simulation starts
    - -infinity -> Integer with default 16. Specifies the infinity cost for the current simulation.
- NetworkNode - representation of a network node. Identified by an ID, has reference to a RoutingTable and known neighbors and implements the RIP protocol.
- RoutingTable - representation of a routing table. It holds RouteTableEntries.
- NetworkLink - representation of a network link between two NetworkNodes with cost associated to the link.
- ScheduledEvent - an abstract class which represents something that should happen at some point. Currently available events are:
    - LinkCostChangeEvent - schedules a link cost change
    - ShowBestRoute - schedules the simulator to output the best route between two nodes
    - TraceRouteTableEvent - schedules the simulator to output the route table of a node

User manual

The program is configured using a .txt file in a specific format. Here are the specs for the configuration file:

```
// line 0 contains the configuration flags. -numOfNodes is the only one required
// but for clarity I have defined all of the other values as well
-numOfNodes 4 -maxExchanges 10 -untilStability true -manual true -splitHorizon on -infinity 64
// link descriptions
// linkNode1 linkNode2 linkCost
      0          1         3
      0          3         7
      1          2         6
      1          3         3
      2          3         2
// link descriptions end comment.
## links changes
// links cost changes scheduling
// linkNode1 linkNode2 changeAfterExchange newCost
       0         1               5            -1
// links cost changes scheduling end
## show best routes
// show best routes scheduling
// fromNode toNode showAfterExchange
       0      1            8
// show best routes scheduling end
## trace routing tables
// trace routing tables scheduling. The simulator will output the routing table for
// a node on every “exchange” with index between the start.. and endExchangeIndex
// node startExchangeIndex endExchangeIndex
     0            0                 5
```

This is the breakup of the first example network. <u>All the comments (lines starting with **//**) can not be in the file and all of the lines starting with **##** must be in the file in the specified order.</u> For example, this file will only schedule route tables tracing for node0 from the beginning to the 5th exchange:

```
-numOfNodes 3 -maxExchanges 10 -untilStability true -manual false -splitHorizon on
0 1 1
1 2 1
## links changes
## show best routes
## trace routing tables
0 0 5
```

Link descriptions - the line “0 1 1” is considered to represent the link in both directions - thus cost (1) is set to the link between node0 and node1 as well as node1 and node0. There is little input validation to check if node ids are in the range [0, numOfNodes] and if link cost is in the range [-1, inf), but nothing more. Same sort of validation is performed for all node ids. Warnings will be displayed if you try to schedule events after the program has terminated. I hope this is clear enough but if there are any problems please feel free to contact me.

Output

The output of the simulator is divided into sections for greater readability. It has the following format

```
================= Round N ===================
simulate network exchange start
// output if something has happened during the exchange (entry timeout for example )
simulate network exchange finish

// network status - STABLE or THERE_WERE_CHANGES.
------- !!! ------ network status after all exchanges have been completed ------- !!! -------

// optional. Shown only if there are scheduled network events to happen after round N
// exchange and the result of the events
####### Scheduled network events ######

================= End Round N ===================
```

Run instructions

You can either use the .jar file (but don't forget to give a configuration's filename as a parameter) or use maven to do whatever you like with it. The .jar file is in

`RIPNetworkSimulator/out/artifacts/RIPNetworkSimulator_jar`

Example:

```
$ cd project_dir/out/artifacts/RIPNetworkSimulator_jar
$ java -jar RIPNetworkSimulator.jar input2.txt
```

Status report

Currently my program implements all of the requirements for the assessed exercise as far as I understand them. I have documented the code appropriately and hope things are clear. My simulator simulates network exchanges between network nodes which use the a very simple RIP-based “protocol” for best route calculation and package navigation. The simulator has the ability to

- Compute routing tables for any preset number of exchanges or until stability is achieved
- Simulate network link cost change during the simulation
- Output the best available route from one node to another at specific times in the simulation
- Output the routing tables of specified nodes for a specified number of exchanges
- Engage, on request, split-horizon heuristic

Limitations

The current simulator implementation does not support direct usage of any other routing protocol except the simple RIP-based protocol. It also features a very ugly command-line interface which will make simulating and tracing big networks very difficult, however I hope it is enough to prove my understanding of the subject and that the simulator works.

Future work

If I had more time I would like to make the program much more modular so that different implementations of the unique components could be used. Also I would like to include a GUI and animations which will better illustrate what gets passed and when, or at least beautify the command-line output.

Network Example 1

Simulation for 15 exchanges WITHOUT split-horizon heuristic of a network with 3 nodes

In this example I will try to demonstrate how the simulator handles a link fail.
Interesting exchanges:

2 - network reaches initial stable state
5 - link fails
6 - node1 and node2 notice link fail
7,8,9 - count-to-infinity
10 - recovery from count-to-infinity => network stable
13 - route-table entry timeout and network fully recovers from link fail

Input file:

```
-numOfNodes 3 -maxExchanges 15 -untilStability false -manual false -splitHorizon off -infinity
10
0 1 1
1 2 1
## links changes
1 2 5 -1
## show best routes
0 2 0
0 2 1
0 2 2
0 2 3
0 2 4
0 2 5
0 2 6
0 2 7
0 2 8
0 2 9
## trace routing tables
0 0 15
1 0 15
2 0 15
```

Diagram of the network

```
 _______          ________        ________
|   0   | --------- 1 --------- |   1    | ----------- 1 ------------ |   2   |
|_______|                       |________|                            |_______|
```

Output and explanations:

Simulation start configuration:

Here you can how you have configured the simulator before the simulation actually starts. If you have defined "**-manual true**" the simulator will output this and start waiting for your command. From here you can the link cost matrix, as well as the initial states of all the nodes and their routing tables, as well as all the scheduled events and when will they happen.

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Starting simulation !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
links costs:
       0      1      -1
       1      0      1
       -1     1      0

State of nodes:

NetworkNode 0:
neighbours: [node1 : cost 1;]
RouteTable for node : 0
       dest cost next
          0  0   null

NetworkNode 1:
neighbours: [node2 : cost 1;node0 : cost 1;]
RouteTable for node : 1
       dest cost next
          1  0   null

NetworkNode 2:
neighbours: [node1 : cost 1;]
RouteTable for node : 2
       dest cost next
          2  0   null


Scheduled events:
       At exchange 0 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 1 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 2 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 3 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 4 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 5 : [LinkCostChangeEvent: 1, newCost: -1; ShowBestRouteEvent from 0 to 2;
TraceRouteTableEvent for node 0; TraceRouteTableEvent for node 1; TraceRouteTableEvent for node
2; ]
       At exchange 6 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 7 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 8 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
       At exchange 9 : [ShowBestRouteEvent from 0 to 2; TraceRouteTableEvent for node 0;
TraceRouteTableEvent for node 1; TraceRouteTableEvent for node 2; ]
```

Exchange 0

Exchange 0 is the first exchange in the network after it has been set-up. There are two types of events scheduled to happen at this exchange - show best route between node0 and node2 and traceRouteTable for all the nodes (0, 1, 2). This means that after the exchange is completed, the simulator will output the current best route from node0 to node2 as well as all the routing tables for all the nodes in the network.

From the output we can deduce that:

1. There ware changes in the routing tables of nodes 0, 1 and 2. Since this is the first exchange it means that initially all the nodes knew was a route to themselves. However, when the exchange happened new routes were included based on what the neighbors of a node have told the node. That is why we see changes in the route tables of all nodes.
2. After exchange 0 node0 knows a route to node1, but not to node2, node1 knows a route to both node1 and node2, and node2 knows a route to node1 and node0. This is because of the random, but sequential way the nodes exchange routing tables. What has happened is that node0 received node1 routing table before node2 has had a chance to tell node1 about itself. So when node0 receives node1 routing table it sees only a route to node1 and updates its own routing table to reflect that. Later node1 receives node0 routing table and appends a route to node0. After that node2 receives node1's routing table and sees routes both to node1 and node0.
3. There is no known route from node0 to node2. This is because from node0 point of view node2 does not exist in the network.

```
========================= Round 0 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0 1 2

########### ScheduledEvents after exchange 0 ###########
ShowBestRouteEvent from 0 to 2
       0 – >  next hop unknown yet

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null
          2    1    2

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0    2    1
          1    1    1
          2    0 null

========================= End round 0 =========================
```

Exchange 1

This is the second exchange. From the simulator's initial state output we know that the simulator will output the routing tables of all the nodes and the best route from node0 to node2.

From the output we can deduce:

1. There was something altered only in node0's routing table. This is expected. After exchange 0 we saw that only node0 does not know about node2, but node1 does. So when node0 receives node1's routing table it sees the entry for node2 and updates its own routing table. Because of the small size and lack of link cost changes node1 and node2 knew about node0 and knew the minimum route costs to all other nodes so they did not register any updates in their routing tables. If we look at the tables after exchange 0 and exchange 1 we will notice that only routing table 0 has changed by now including a route to node2.
2. The best route from node0 to node2 can now be traced since node0 now knows about node2. The best route is 0 -> 1 -> 2.

```
========================= Round 1 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0

########### ScheduledEvents after exchange 1 ###########
ShowBestRouteEvent from 0 to 2
       0 - > 1 -> 2

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1
          2    2    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null
          2    1    2

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0    2    1
          1    1    1
          2    0 null

========================= End round 1 =========================
```

Exchange 2,3,4

I group exchanges 2, 3 and 4 because nothing really exciting happens to the network. Again we have scheduled to track the routing tables for all nodes in the network as well as try to find the best route between node0 and node1.

From the output we can deduce:

1. The network was in a stable state. This means no node has changed its routing table during the exchange to acknowledge a better route or link cost change
2. The best route between node0 and node2 is still 0 -> 1 -> 2
3. Routing tables have indeed not changed.

```
========================= Round 2,3,4 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 2,3,4 ###########
ShowBestRouteEvent from 0 to 2
      0 – > 1 –> 2

TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1    1    1
         2    2    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0    1    0
         1    0 null
         2    1    2

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         0    2    1
         1    1    1
         2    0 null

========================= End round 2,3,4 ==========================
```

Exchange 5

This exchange yields the same results as exchanges 2,3 and 4. The only difference is that we have scheduled the link between node1 and node2 to change cost from 1 to -1. -1 reflects a failed link in the system so what we are basically saying is that there is now no connection between node1 and node2.

```
========================= Round 5 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 5 ###########
LinkCostChangeEvent, link between 1 and 2 from 1 to -1

ShowBestRouteEvent from 0 to 2
       0 - > 1 -> 2

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1
          2    2    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null
          2    1    2

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0    2    1
          1    1    1
          2    0 null

========================= End round 5 ==========================
```

Exchange 6

During this exchange the network will have to propagate the changes in its topology.

From the output we can deduce:

1. During the exchange node1 and node2 notice that the link between them is gone. Both nodes update to reflect that they are not neighbors anymore. Each node also deletes any routes that have been learned from the other node for consistency.
2. The best route from node0 to node2 is now incomplete. There is no info what to do after node1 in the path.

```
========================= Round 6 ==========================
simulate network exchange start
Node 1 remove neighbour 2
Node 2 remove neighbour 1
Node 2 will remove entry because it came from neightbor 1 : RouteEntry{ dest: 0, cost: 2,
next:1}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
1 2

########### ScheduledEvents after exchange 6 ###########
ShowBestRouteEvent from 0 to 2
       0 – > 1 ->  next hop unknown yet

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1
          2    2    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          2    0 null

========================= End round 6 ==========================
```

Exchange 7, 8, 9

During those exchange we can see the counting to infinity problem. Node0 and node1 start to feed each other information about a route to node2. However because they send only destination -> minRouteLenght pairs there is no way for one of the nodes to notice that the advertised route from the other node includes the first node and thus there is no way to notice the cycle that has arose.

From the output we can deduce:

1. Routing tables for node1 and node0 are the only ones that get updated in during the exchanges. This is because they keep telling each other lies. The cost to node2 starts rising in both route tables. Node0 goes from 4 to 6 to 8 while node1 goes from 3 to 5 to 7.
2. The show-best-route can not find a route from node0 to node2. It also shows a cycle in the network.

```
========================= Round 7,8,9 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0 1

########### ScheduledEvents after exchange 7,8,9 ###########
ShowBestRouteEvent from 0 to 2
There is a cycle in the routes meaning the end node (2) has become unreachable!
      0 - > 1 -> 0

TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost     next
         0    0     null
         1    1        1
         2    4,6,8    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost     next
         0    1        0
         1    0     null
         2    3,5,7    0

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost     next
         2    0     null

========================= End round 7,8,9 =========================
```

Exchange 10

During this exchange we will be expecting to see the cost from node0 to node2 in node0's routing table to reach infinity defined as 10 in the configuration file.

From the output we can deduce:

1. Node0 notices that node2 is no longer reachable. This happens because the cost for reaching node2 from node0 becomes equal to the defined infinity cost 10. This causes node0 to drop the route from its routing table and not advertise it to its neighbors.

```
========================= Round 10 =========================
simulate network exchange start
Node 0 dropping route for dest (2) because infinity (10) was reached.
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0 1

########### ScheduledEvents after exchange 10 ###########
TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null
          2    9    0

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          2    0 null

========================= End round 10 =========================
```

Exchange 11,12

They are not interesting at all.

From the output we can deduce:

1. The network is stable.
2. Node1's cost to node2 does not go up.

```
========================= Round 11,12 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 11,12 ###########
TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null
          2    9    0

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          2    0 null

========================= End round 11,12 ==========================
```

Exchange 13

During this round we get to see a timeout in the routing table of node1. After this exchange the network has finally recovered from the link fail during exchange 5.

From the output we can deduce:

1. The entry for node2 in node1's routing table expires. This happens because node1 has not heard anything about node2 from anywhere for the past 3 exchanges - 11,12,13.
2. The network has finally fully recovered from the link fail during exchange 5. The routing tables of all nodes represent the current topography of the network without any redundant/outdated info.

```
========================= Round 13 ==========================
simulate network exchange start
Node 1 removing timeout entry: RouteEntry{ dest: 2, cost: 9, next:0}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
1

########### ScheduledEvents after exchange 13 ###########
TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1    1    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0    1    0
         1    0 null

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         2    0 null

========================= End round 13 ==========================
```

Exchange 14

Final exchange. Nothing interesting.

```
========================= Round 14 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 14 ###########
 . . .
========================= End round 14 ==========================
```

Simulation for 15 exchanges WITH split-horizon heuristic of a network with 3 nodes

This is the same network as the above but with split-horizon heuristic turned on.

Interesting exchanges:

- 2 - network reaches initial stable state
- 5 - link fails
- 6 - node1 and node2 notice link fail
- 7 - network back in stable state
- 9 - route-table entry timeout and network fully recovers from link fail

This breakdown shows how much of a difference can split-horizon have on network convergence. It takes 4 exchanges less for the network to fully recover from the link fail. And the difference will be even greater if we used the RIP default infinity cost of 16 instead of 10.

Input:

```
-numOfNodes 3 -maxExchanges 15 -untilStability false -manual false -splitHorizon on -infinity 10
0 1 1
1 2 1
## links changes
1 2 5 -1
## show best routes
0 2 0
0 2 1
0 2 2
0 2 3
0 2 4
0 2 5
0 2 6
0 2 7
0 2 8
0 2 9
## trace routing tables
0 0 15
1 0 15
2 0 15
```

Output:

Exchange 0,1,2,3,4,5,6

During exchanges 0, 1, 2, 3, 4 and 5 the same things happen as before.
At exchange 2 the network becomes stable.
At exchange 5 the link between node1 and node2 fails.
At exchange 6 node1 and node2 register that they are no longer neighbors and update accordingly.

Exchange 7

This is the exchange which demonstrates very clearly how split-horizon saves the day.

From the output we can deduce:

1. Network is in stable state. This is because split-horizon is turned on. This means that while node0 and node1 still exchange routing tables, those routing tables do not include routes learned from the other node. This helps because it breaks the count-to-infinity and the network is stable only 1 exchange after the link has failed.

```
========================= Round 7 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 7 ###########
ShowBestRouteEvent from 0 to 2
      0 – > 1 ->  next hop unknown yet

TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1    1    1
         2    2    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0    1    0
         1    0 null

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         2    0 null

========================= End round 7 =========================
```

Exchange 8

Network stable. Nothing interesting.

```
========================= Round 8 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 8 ###########
. . .
========================= End round 8 ==========================
```

Exchange 9

Entry in node0’s routing table timeouts after not hearing anything about node2 for 4 cycles. Network has finally fully recovered from link loss.

From the output we can deduce:

1. Node0’s entry to node2 timeouts. This happens because at exchange 5 the link between node1 and node2 fails. Because of this node1 removes its entry to node2 from its routing table. At exchange 6,7,8 node1 and node0 exchange routing tables . Node1’s routing table does not contain a route for node2 and because of the split-horizon when node0 does not include the route to node2 when it sends its routing table to node1 because node0 has learned the route to node2 from node1. This means that node0 stops receiving any updates for node2 so after the default 4 cycles the entry timeouts.
2. Network has finally fully recovered from link loss since there is no more redundant/outdated info.

```
========================= Round 9 ==========================
simulate network exchange start
Node 0 removing timeout entry: RouteEntry{ dest: 2, cost: 2, next:1}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0

########### ScheduledEvents after exchange 9 ###########
ShowBestRouteEvent from 0 to 2
       0 – >  next hop unknown yet

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    1    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    1    0
          1    0 null

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          2    0 null

========================= End round 9 ==========================
```

Exchange 10,11,12,13,14

Nothing interesting.

Network Example 2

Simulation for 10 exchanges WITH split-horizon heuristic of a network with 4 nodes

Input:

```
-numOfNodes 4 -maxExchanges 10 -untilStability false -manual false -splitHorizon on -infinity 60
0 1 3
0 3 7
1 2 6
1 3 3
2 3 2
## links changes
0 1 5 -1
## show best routes
0 1 4
1 0 4
0 1 6
1 0 6
3 0 6
0 1 7
1 0 7
3 0 7
0 1 8
1 0 8
3 0 8
0 1 9
1 0 9
## trace routing tables
0 0 10
1 0 10
2 0 10
3 0 10
```

Diagram of the network

```
          _______
          |__3__|
         /   |   \
        7    |    2
   ____/     |     \_______
   |_0_|     3      |__2__|
       \     |     /
        3    |    6
         \__|__  /
          |_1_|
```

Output:

Simulator configuration:

Again, the first few lines just present the initial state of the simulator as well as all the events that have been scheduled to happen. I did not include it because I don't feel it is particularly interesting.

Exchange 0:

As before during the first exchange ever all nodes learn about their immediate neighbors. The lucky ones even learn about some of the neighbors of their neighbors.

From the output we can deduce:

1. All nodes have learned about their neighbors reflected in all routing tables having 4 entries = number of nodes in the network.
2. Some of the nodes even learn a better route to a destination from the initial. Node3 initially learns a route to node0 with cost 7 but then it learns a better route through node1 with cost 6 (blue entries). Same thing for node1 which learns in 1 round a route to node2 and a better route to node2 (yellow entries). This is because of the sequential way the simulator works. It first made node0 send its routing table to node1 and node3 thus node1 and node3 learn about node0. Some time later node1 sends its routing table to node3 which includes a better route to node0 so node3 updates its routing table to reflect that.
3. Node2 does the most progress by learning figuring out better routes for two destinations (red entries).

```
========================= Round 0 ==========================
simulate network exchange start
Node 1 logging RouteEntry{ dest: 0, cost: 3, next:0}
Node 3 logging RouteEntry{ dest: 0, cost: 7, next:0}
Node 2 logging RouteEntry{ dest: 0, cost: 9, next:1}
Node 2 logging RouteEntry{ dest: 1, cost: 6, next:1}
Node 0 logging RouteEntry{ dest: 1, cost: 3, next:1}
Node 3 logging RouteEntry{ dest: 0, cost: 6, next:1}
Node 3 logging RouteEntry{ dest: 1, cost: 3, next:1}
Node 1 logging RouteEntry{ dest: 2, cost: 6, next:2}
Node 3 logging RouteEntry{ dest: 2, cost: 2, next:2}
Node 2 logging RouteEntry{ dest: 0, cost: 8, next:3}
Node 2 logging RouteEntry{ dest: 1, cost: 5, next:3}
Node 2 logging RouteEntry{ dest: 3, cost: 2, next:3}
Node 0 logging RouteEntry{ dest: 2, cost: 9, next:3}
Node 0 logging RouteEntry{ dest: 3, cost: 7, next:3}
Node 1 logging RouteEntry{ dest: 2, cost: 5, next:3}
Node 1 logging RouteEntry{ dest: 3, cost: 3, next:3}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0 1 2 3

########### ScheduledEvents after exchange 0 ###########
TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1    3    1
          2    9    3
          3    7    3

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0    3    0
          1    0 null
          2    5    3
          3    3    3

TraceRouteTableEvent for node 2
```

```
RouteTable for node : 2
      dest cost next
         0    8    3
         1    5    3
         2    0 null
         3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
      dest cost next
         0    6    1
         1    3    1
         2    2    2
         3    0 null

========================= End round 0 ==========================
```

Exchange 1

From the output we can deduce:

1. During this exchange node0 found out better routes to both node2 and node3.
2. Non of the other nodes found out better routes. This is expected since in exchange 0 node1, node2 and node3 not only acknowledged the presence of the other nodes on the network but also managed to find out the best routes to them.

```
========================= Round 1 ==========================
simulate network exchange start
Node 0 logging RouteEntry{ dest: 2, cost: 8, next:1}
Node 0 logging RouteEntry{ dest: 3, cost: 6, next:1}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0

########### ScheduledEvents after exchange 1 ###########
TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1    3    1
         2    8    1
         3    6    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0    3    0
         1    0 null
         2    5    3
         3    3    3

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         0    8    3
         1    5    3
         2    0 null
         3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
      dest cost next
         0    6    1
         1    3    1
         2    2    2
         3    0 null

========================= End round 1 =========================
```

Exchange 2,3,4,5

Nothing interesting happens during those exchanges.

From the output we can deduce that:

1. Network is stable since round 2
2. Nothing happens in the network
3. At round 4 we see the current best route from node0 to node1.
4. At round 5 link between route node0 and node1 fails.

```
========================= Round 2,3,4,5 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 4 ###########
ShowBestRouteEvent from 0 to 1
      0 - > 1

ShowBestRouteEvent from 1 to 0
      1 - > 0

########### ScheduledEvents after exchange 5 ###########
LinkCostChangeEvent
      link between 0 and 1 from 3 to -1

TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1    3    1
         2    8    1
         3    6    1

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0    3    0
         1    0 null
         2    5    3
         3    3    3

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         0    8    3
         1    5    3
         2    0 null
         3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
      dest cost next
         0    6    1
         1    3    1
         2    2    2
         3    0 null

========================= End round 2,3,4,5 ==========================
```

Exchange 6

In the previous exchange the link between node0 and node1 has failed.

From the output we can deduce:

1. Node0 and node1 notice the failed link and recover by removing all the entries in the table learned from the node on the other end of the failed link. They later fill in the missing routes.
2. Node1 shows signs of the count-to-infinity problem. A cycle is formed between node1, node2 and node3 about a path to node0. Because node1 has noticed the link failure it has deleted entries learned from node0 as well as through node0. When it later receives node2's routing table it "registers" a route to an unknown node0 so it just accepts it and thus closes the cycle between node1, node2 and node3 which now depend on each other.

```
========================= Round 6 ==========================
simulate network exchange start
Node 0 remove neighbour 1
Node 0 will remove entry because it came from neightbor 1 : RouteEntry{ dest: 3, cost: 6,
next:1}
Node 0 will remove entry because it came from neightbor 1 : RouteEntry{ dest: 2, cost: 8,
next:1}
Node 1 remove neighbour 0
Node 1 logging RouteEntry{ dest: 0, cost: 14, next:2}
Node 0 logging RouteEntry{ dest: 1, cost: 10, next:3}
Node 0 logging RouteEntry{ dest: 2, cost: 9, next:3}
Node 0 logging RouteEntry{ dest: 3, cost: 7, next:3}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
0 1
########### ScheduledEvents after exchange 6 ###########
ShowBestRouteEvent from 0 to 1
       0 - > 3 -> 1
ShowBestRouteEvent from 1 to 0
There is a cycle in the routes meaning the end node (0) has become unreachable!
       1 - > 2 -> 3 -> 1
ShowBestRouteEvent from 3 to 0
There is a cycle in the routes meaning the end node (0) has become unreachable!
       3 - > 1 -> 2 -> 3
TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1   10    3
          2    9    3
          3    7    3
TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0   14    2
          1    0 null
          2    5    3
          3    3    3
TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0    8    3
          1    5    3
          2    0 null
          3    2    3
TraceRouteTableEvent for node 3
RouteTable for node : 3
       dest cost next
          0    6    1
          1    3    1
          2    2    2
          3    0 null
========================= End round 6 ==========================
```

Exchange 7

This is very interesting exchange.

From the output we can deduce that:

1. We can very clearly see the counting-to-infinity happening between nodes 1, 2 and 3 which all feed the others wrong info about a route to node0. Even though split-horizon is turned on it will not help in this situation because it can only prevent two nodes from feed each other information.
2. Node3 does not register that node0 can be reached through the direct link connecting them with cost 7. This is again because of the sequential way the simulator works. What has happened is that node0 has advertised its routing table to node3 before node3 has had the chance to know about the link fail. This means that node3 still holds a route entry for node0 through node1 for cost 6 and thus it disregards node0's route.

```
========================= Round 7 ==========================
simulate network exchange start
Node 3 logging RouteEntry{ dest: 0, cost: 17, next:1}
Node 2 logging RouteEntry{ dest: 0, cost: 19, next:3}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
2 3

########### ScheduledEvents after exchange 7 ###########
ShowBestRouteEvent from 0 to 1
       0 - > 3 -> 1

ShowBestRouteEvent from 1 to 0
There is a cycle in the routes meaning the end node (0) has become unreachable!
       1 - > 2 -> 3 -> 1

ShowBestRouteEvent from 3 to 0
There is a cycle in the routes meaning the end node (0) has become unreachable!
       3 - > 1 -> 2 -> 3

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1   10    3
          2    9    3
          3    7    3

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0   14    2
          1    0 null
          2    5    3
          3    3    3

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0   19    3
          1    5    3
          2    0 null
          3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
       dest cost next
          0   17    1
          1    3    1
          2    2    2
          3    0 null

========================= End round 7 ==========================
```

Exchange 8

Network recovers.

From the output we can deduce:

1. Node3 figures out that the best route to node0 because this time the advertised route has a lower cost than the known route.
2. Node3 shares the new info with the other nodes which also promptly update their routes to node0.
3. Network recovers fully

```
========================= Round 8 ==========================
simulate network exchange start
Node 3 logging RouteEntry{ dest: 0, cost: 7, next:0}
Node 1 logging RouteEntry{ dest: 0, cost: 25, next:2}
Node 2 logging RouteEntry{ dest: 0, cost: 9, next:3}
Node 1 logging RouteEntry{ dest: 0, cost: 10, next:3}
simulate network exchange finish

------- !!! ------- There were a changes in node routing tables ------- !!! -------
Nodes with changed routing tables:
1 2 3

########### ScheduledEvents after exchange 8 ###########
ShowBestRouteEvent from 0 to 1
      0 - > 3 -> 1

ShowBestRouteEvent from 1 to 0
      1 - > 3 -> 0

ShowBestRouteEvent from 3 to 0
      3 - > 0

TraceRouteTableEvent for node 0
RouteTable for node : 0
      dest cost next
         0    0 null
         1   10    3
         2    9    3
         3    7    3

TraceRouteTableEvent for node 1
RouteTable for node : 1
      dest cost next
         0   10    3
         1    0 null
         2    5    3
         3    3    3

TraceRouteTableEvent for node 2
RouteTable for node : 2
      dest cost next
         0    9    3
         1    5    3
         2    0 null
         3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
      dest cost next
         0    7    0
         1    3    1
         2    2    2
         3    0 null

========================= End round 8 ==========================
```

Exchange 9

Nothing exciting.

From the output we can deduce:

1. Network is stable

```
========================= Round 9 ==========================
simulate network exchange start
simulate network exchange finish

------- !!! ------- Network stable for this round ------- !!! ------

########### ScheduledEvents after exchange 9 ###########
ShowBestRouteEvent from 0 to 1
       0 - > 3 -> 1

TraceRouteTableEvent for node 0
RouteTable for node : 0
       dest cost next
          0    0 null
          1   10    3
          2    9    3
          3    7    3

TraceRouteTableEvent for node 1
RouteTable for node : 1
       dest cost next
          0   10    3
          1    0 null
          2    5    3
          3    3    3

TraceRouteTableEvent for node 2
RouteTable for node : 2
       dest cost next
          0    9    3
          1    5    3
          2    0 null
          3    2    3

TraceRouteTableEvent for node 3
RouteTable for node : 3
       dest cost next
          0    7    0
          1    3    1
          2    2    2
          3    0 null

========================= End round 9 ==========================
```