<u>3D3 Computer Networks Project 2</u>

Group 10 members:

Aniket Agarwal	17317437
Arnav Malhotra	17317424
Daniel Woodward	15324849
Grace McLaughlin	15301485

Running the Files

1. Open 7 terminals and navigate to the folder containing all the files in all of them.
2. Write the command make in any of terminal to compile the program
3. To run any of the routers write the command 'make runA' to run router A and 'make runB' to run router B and similarly for all the routers.
   NOTE: Only one router can be run in one terminal. We could have run all the routers in one terminal as well but that creates a confusion about what is happening in the terminal window and you have to open the output files again and again so to avoid that we made it like this.
4. Write the command ./H "hello there!" to send the data packet from router H to router D.

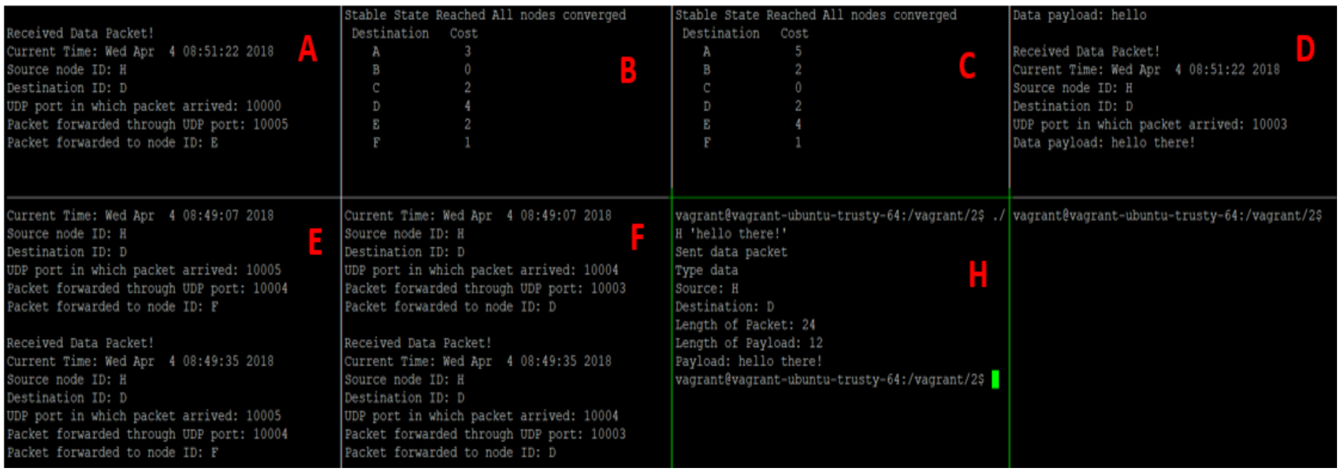
Implementation Description:

- **Files** -
  - H to send packet data.
  - Makefile to run the routers.
  - Topology contains the input data.
  - my-router.cpp file for the code.
- **Classes**-
  - **Router Class:** A router table is maintained by each router which is implemented as an array of router class. The cost of the shortest path to a particular destination and the next hop along the shortest path is contained by each router. The router class checks the validity of the next hop when sending packets by a Boolean invalid function that is used to check the next routers port, name, and cost.
  - **R_node Class:** Contains basic information about a single router. Each router maintains a vector of node class, representing the information for the router's neighbouring routers. Such information includes the ID of the neighbour, the port number to reach the neighbour, the timer (to determine if the router is still valid), and the neighbour's socket address.
  - **DistVector Class:** The wrapper class that contains all information that the current router holds. It includes its address, ID, routing table, and neighbour node classes. This class uses the bellman-ford algorithm to find the minimum distance between nodes. There is a list of the neighbouring nodes as well as a map of the nodes with the name of the node and the port number associated. Here, if the inputted router name is 'H', meaning it is at the starting router, the packet entered is sent through the most cost-effective route to router 'D' and prints out the convergence. If the start router is not

entered, the routers initiated continue to update themselves with the costs to each node.
    - **Header Class**: Includes the header for each port with the type of data inputted, the length of the payload, the destination router name, and the source router name.
- **Routing tables**-
    - Each router maintains an up-to-date routing table, as well as a backup routing table containing the initial routing information. The up-to-date routing table updates its entries based on the information contained in a received advertisement. It also keeps track of which routers in the network are invalid, making sure to route in a manner that avoids such invalid routers. The routing tables are printed out in the terminal as well as to a text file, such as 'routing-outputA.txt' for node A. The tables are shown below along with the convergence and the timing.
- **Packet types**-
    - **Data:** Contained by the packet which is needed to be forwarded to a particular destination.
    - **Advertisement:** Contains routers DV, Used to update neighbouring routers' routing tables. Sent periodically to concerned router's neighbours. Counter used to calculate convergence is set to 0 whenever the table is updated.
    - **Wakeup:** This packet is sent periodically from the child process to the parent process, parent is woken up to listen for the packets. Keeps a track on the counter for convergence and once it reaches 5 it prints the value.
    - **Reset:** if any of the routers become invalid, then this alerts all routers to reset their routing tables to initial configuration.
- **Difficulties**
    1. **Getting the Convergence:** After all the routers are connected to each other we have to show that it has reached a stable state and print a final table showing all the cost between all the routers from that node. So we made our update function Boolean type and while we are updating our dv table while it is advertising we are also setting a counter as 0 every time the table gets updated. When it reaches WakeUp case it checks if it takes more than 5 times a print function is called which indicates that no table is currently being updated and the router is being self-casted periodically. In this case we came face-to-face with another bug that this table was printed even if all the routers were not connected. So to further solve that bug we included a condition in that print function itself to check that if there are more than one routers that are invalid than it indicates stable state not reached otherwise it has reached the stable state and it should print the table.
    2. **Data Routing From A-D:** When a packet is sent from the starting router, H, to router D it is sent in the most cost-effective route. Below shows a packet 'hello there!' sent at router H. This then uses the bellman-ford method to reach the destination, hopping to router A, E, F, and then D. But according to our question the path it should take is A, B, F, C and D. Even though the path it takes is shortest-one but it is still not the desired one. We were unable to solve this bug but this is the only bug in our program.
- **Timing-** Each router broadcasts its distance vector every 5 second. Also, if a router does not get any advertisements from its neighbour for 5 seconds, that neighbour becomes invalid. The routing tables continue to update until they have converged on

the lowest cost to each route which takes a maximum of 30-45 seconds. A timer is also set for when a wakeup packet is received, it checks if the timer has expired, or the time between the current time and that from when the timer was started is greater than 5 seconds, the packet is self-cast with a packet type of reset.

- **Contributions-** Aniket wrote the whole code with the contribution of Vanshika Agarwal (17317504) from Group 14. Daniel and Grace edited the code and worked on solving the difficulties. Arnav, Grace, Daniel wrote the major sections of the report and Aniket wrote the Difficulties faced and Running the files section. Daniel also made the PowerPoint Presentation for the Project demo.
  Additional Contribution: Vanshika Agarwal (17317504) from Group 14 contributed in writing the code.
- **Sending packets**- When a packet is sent from the starting router, H, to router D it is sent in the most cost-effective route. Below shows a packet 'hello there!' sent at router H. This then uses the bellman-ford method to reach the destination, hopping to router A, E, F, and then D.



```
Received Data Packet!
Current Time: Wed Apr  4 08:51:22 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10000
Packet forwarded through UDP port: 10005
Packet forwarded to node ID: E
```

```
Stable State Reached All nodes converged
 Destination   Cost
    A           3
    B           0
    C           2
    D           4
    E           2
    F           1
```

```
Stable State Reached All nodes converged
 Destination   Cost
    A           5
    B           2
    C           0
    D           2
    E           4
    F           1
```

```
Data payload: hello

Received Data Packet!
Current Time: Wed Apr  4 08:51:22 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10003
Data payload: hello there!
```

```
Current Time: Wed Apr  4 08:49:07 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10005
Packet forwarded through UDP port: 10004
Packet forwarded to node ID: F

Received Data Packet!
Current Time: Wed Apr  4 08:49:35 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10005
Packet forwarded through UDP port: 10004
Packet forwarded to node ID: F
```

```
Current Time: Wed Apr  4 08:49:07 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10004
Packet forwarded through UDP port: 10003
Packet forwarded to node ID: D

Received Data Packet!
Current Time: Wed Apr  4 08:49:35 2018
Source node ID: H
Destination ID: D
UDP port in which packet arrived: 10004
Packet forwarded through UDP port: 10003
Packet forwarded to node ID: D
```

```
vagrant@vagrant-ubuntu-trusty-64:/vagrant/2$ ./
H 'hello there!'
Sent data packet
Type data
Source: H
Destination: D
Length of Packet: 24
Length of Payload: 12
Payload: hello there!
vagrant@vagrant-ubuntu-trusty-64:/vagrant/2$
```

```
vagrant@vagrant-ubuntu-trusty-64:/vagrant/2$
```

- **Routing Tables**
  Router A
  Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:06 2018
  Stable State Reached All nodes converged
   Destination: A B C D E F
   Cost: 0 3 5 7 1 4
  Router B
  Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:08 2018
  Stable State Reached All nodes converged
   Destination: A B C D E F
   Cost: 3 0 2 4 2 1
  Router C
  Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:05 2018
  Stable State Reached All nodes converged
   Destination: A B C D E F
   Cost: 5 2 0 2 4 1
  Router D

Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:08 2018

Stable State Reached All nodes converged

```
 Destination: A  B  C  D  E  F
 Cost:        7  4  2  0  6  3
```

Router E

Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:05 2018

Stable State Reached All nodes converged

```
 Destination: A  B  C  D  E  F
 Cost:        1  2  4  6  0  3
```

Router F

Stable State Reached All nodes converged. Time of convergence:Wed Apr 4 22:48:02 2018

Stable State Reached All nodes converged

```
 Destination: A  B  C  D  E  F
 Cost:        4  1  1  3  3  0
```