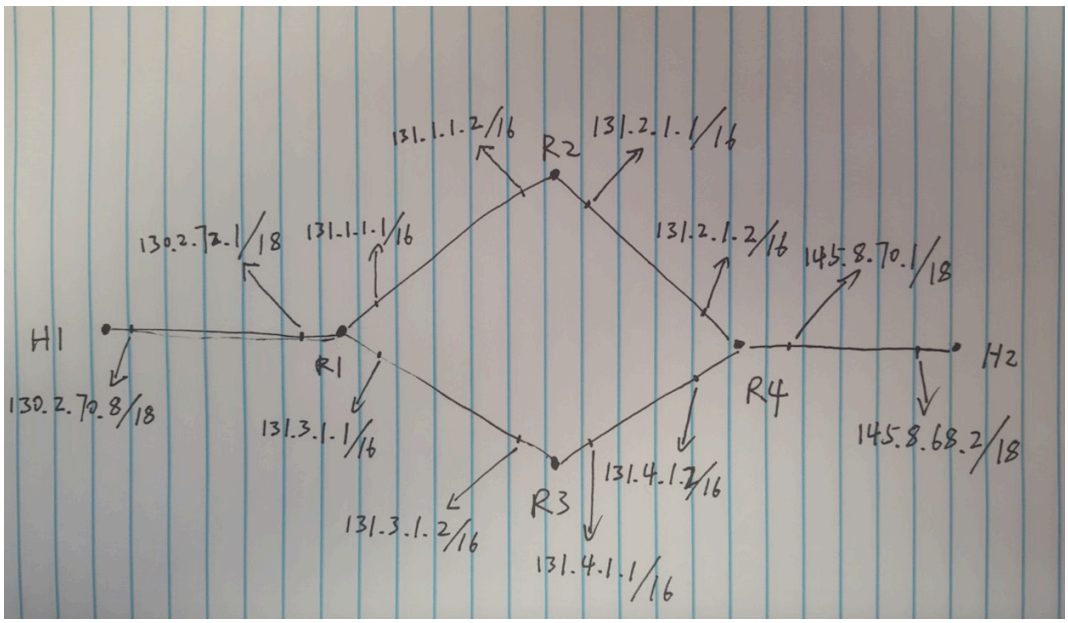
Note: the entire files for codes&configs&outputs are in the root/attachment/quagga-ixp folder, including topo.py, start.py, quagga config files, PartC codes, PartC outputs, daemon logs, etc.

PART A:

A1:

(a) topo.py file: in root/PartA/topo.py

(b) network topology figure, as follows:



130.2.64.0/18 is the subnet between H1 and R1
145.8.64.0/18 is the subnet between H2 and R4

mininext> net
H1 H1-eth0:R1-eth0
H2 H2-eth0:R4-eth2
R1 R1-eth0:H1-eth0 R1-eth1:R2-eth0 R1-eth2:R3-eth0
R2 R2-eth0:R1-eth1 R2-eth1:R4-eth0
R3 R3-eth0:R1-eth2 R3-eth1:R4-eth1
R4 R4-eth0:R2-eth1 R4-eth1:R3-eth1 R4-eth2:H2-eth0
c0

more screenshots of 'mininext> node.ifconfig' result in the root/PartA/A1_screenshots/ folder

A2:

(a) all routing tables in root/PartA/A2_screenshots/ folder

what I did: after A1 assign ip to all interfaces using 'host ifconfig' command, for A2 I used 'host ip route add' command to add route from this host to all accessible hosts

(b) mininext> H1 traceroute H2

traceroute to 145.8.68.2 (145.8.68.2), 30 hops max, 60 byte packets
1 130.2.72.1 (130.2.72.1) 0.028 ms 0.007 ms 0.005 ms
2 131.1.1.2 (131.1.1.2) 0.014 ms 0.007 ms 0.007 ms
3 131.2.1.2 (131.2.1.2) 0.016 ms 0.010 ms 0.009 ms
4 145.8.68.2 (145.8.68.2) 0.015 ms 0.010 ms 0.011 ms

mininext> H2 traceroute H1
traceroute to 130.2.70.8 (130.2.70.8), 30 hops max, 60 byte packets
1 145.8.70.1 (145.8.70.1) 0.024 ms 0.004 ms 0.004 ms
2 131.2.1.1 (131.2.1.1) 0.013 ms 0.005 ms 0.005 ms
3 131.1.1.1 (131.1.1.1) 0.012 ms 0.007 ms 0.006 ms
4 130.2.70.8 (130.2.70.8) 0.020 ms 0.006 ms 0007 ms

PART B:

B1:

(a) & (b) Commands in order:(I didn't continue PART B after A2...I restarted the topo, so I needed to re-assign ips)

```
# set ip_forward to 1
H1 sudo sysctl -w net.ipv4.ip_forward=1
H2 sudo sysctl -w net.ipv4.ip_forward=1
R1 sudo sysctl -w net.ipv4.ip_forward=1
R2 sudo sysctl -w net.ipv4.ip_forward=1
R3 sudo sysctl -w net.ipv4.ip_forward=1
R4 sudo sysctl -w net.ipv4.ip_forward=1
# assign ips
H1 ifconfig H1-eth0 130.2.70.8/18
R1 ifconfig R1-eth0 130.2.72.1/18
R1 ifconfig R1-eth1 131.1.1.1/16
R1 ifconfig R1-eth2 131.3.1.1/16
R2 ifconfig R2-eth0 131.1.1.2/16
R2 ifconfig R2-eth1 131.2.1.1/16
```

R3 ifconfig R3-eth0 131.3.1.2/16
R3 ifconfig R3-eth1 131.4.1.1/16
R4 ifconfig R4-eth0 131.2.1.2/16
R4 ifconfig R4-eth1 131.4.1.2/16
R4 ifconfig R4-eth2 145.8.70.1/18
H2 ifconfig H2-eth0 145.8.68.2/18
write .conf files, file details in root/PartB/B1_config/ folder
H1 vi /etc/quagga/daemons
H1 vi /etc/quagga/zebra.conf
H1 vi /etc/quagga/ripd.conf
H1 /etc/init.d/quagga restart #start daemon
H1 telnet localhost 2602 #login to ripd
(in ripd cli, type following cmd: enable; conf t; routing rip; network XXX; exit)
...
...
...(for other hosts, same commands)

Finally, all 6 nodes are running zebra and ripd daemons

B2:

(a) routing table screenshots in root/PartB/B2_screenshots/ folder

(b) mininext> H1 traceroute H2

traceroute to 145.8.68.2 (145.8.68.2), 30 hops max, 60 byte packets
1 130.2.72.1 (130.2.72.1) 0.024 ms 0.005 ms 0.004 ms
2 131.1.1.2 (131.1.1.2) 0.012 ms 0.006 ms 0.005 ms
3 131.2.1.2 (131.2.1.2) 0.013 ms 0.007 ms 0.006 ms
4 145.8.68.2 (145.8.68.2) 0.011 ms 0.008 ms 0.007 ms

mininext> H2 traceroute H1
traceroute to 130.2.70.8 (130.2.70.8), 30 hops max, 60 byte packets
1 145.8.70.1 (145.8.70.1) 0.030 ms 0.007 ms 0.005 ms
2 131.2.1.1 (131.2.1.1) 0.015 ms 0.008 ms 0.009 ms
3 131.1.1.1 (131.1.1.1) 0.018 ms 0.010 ms 0.009 ms
4 130.2.70.8 (130.2.70.8) 0.017 ms 0.012 ms 0.011 ms

(c) I finished configuring the last node (H2) and immediately ran the 'ping', but it already worked, which means the converge time is very small. One can use a script to precisely record the two time points and compare them.

B3:

(a) As R1-R2 is on the current path, I brought down R1-R2 by the command 'link R1 R2 down'

(b) I used a time counter in my mobile phone, and the time for connectivity establishment is roughly 35 seconds.

(c) mininext> H1 traceroute H2

traceroute to 145.8.68.2 (145.8.68.2), 30 hops max, 60 byte packets
1 130.2.72.1 (130.2.72.1) 0.022 ms 0.004 ms 0.003 ms
2 131.3.1.2 (131.3.1.2) 0.012 ms 0.005 ms 0.004 ms
3 131.4.1.2 (131.4.1.2) 0.017 ms 0.010 ms 0.010 ms
4 130.2.70.8 (130.2.70.8) 0.016 ms 0.011 ms 0.010 ms

mininext> H2 traceroute H1
traceroute to 130.2.70.8 (130.2.70.8), 30 hops max, 60 byte packets
1 145.8.70.1 (145.8.70.1) 0.029 ms 0.006 ms 0.005 ms
2 131.4.1.1 (131.4.1.1) 0.016 ms 0.008 ms 0.006 ms
3 131.3.1.1 (131.3.1.1) 0.017 ms 0.010 ms 0.010 ms
4 130.2.70.8 (130.2.70.8) 0.016 ms 0.011 ms 0.010 ms

PART C:

C1:

(a) routing protocol codes (each node has a .py daemon file) are in root/PartC/C1_codes/ folder: H1_Daemon.py, R1_Daemon.py, R2_Daemon.py, R3_Daemon.py, R4_Daemon.py (I didn't run protocol daemon for H2 node, because H2 is in the end of the directed route, and cannot route to any other node)

more about these codes:

1. weights of each edge stored in the weights.conf file
2. the protocol algorithm is implemented in the rt_update() function, which can be seen in every Node_Daemon.py file
3. I output the routing table (with timestamp) of each node to corresponding files (logNode.txt)
4. the daemon log file and pid file are output to the root/PartC/C1_codes/logs/ folder, though no useful info...
5. I referenced a module 'daemon.py' from the internet

How to run:

this may be challenge because I didn't consider transplanting my code to other machines. To run it: fisrt copy 'quagga-ixp' folder in the root/attachment/ folder to your current directory; in each Node_Daemon.py file, change the 'currentPath' variable to your

current directory, and change the content of the 'setting' dictionary to match your directory.

Then, run 'sudo python start.py' which will yield the topology with static routing table (PART A result), then in the order 'H1->R1->R2->R3->R4->H2', run 'mininext> Node python ./part_c/Node_Daemon.py restart'; if everything goes well, you'll get the correct logNode.txt

Any problem running, can contact me at yunke.tian@stonybrook.edu. Thanks!

(b) time taken:

According to my code, and my daemon starting order, the first time stamp of R4's routing table should be the moment of system start; and, the time stamp when H1's routing table start to be stable, should be the moment of convergence.

So, according to my logR4.txt, the first time stamp is:

04:42:25

while in logH1.txt the time stamp for H1's routing table to be stable is:

04:42:28

So, the time taken is 3 seconds, which is highly relavant to the check period I set (6s)

(c) application later routing table at each node:

please open root/PartC/output/logNode.txt to see the routing tables (just check the 10th routing table...it should be converged)

C2:

(a) time taken:

This time, just check H1's routing table to see when the converged table began to change, and when the change is again converged.

According to logH1.txt, time stamp when table start to change is:

04:44:10

while the time stamp when the change become stable again is:

04:44:16

So, the time taken is 6 seconds, just one period I set.

(b) the application layer routing table at each node:

Again, please open root/PartC/output/logNode.txt to see the routing tables (just check the last table)