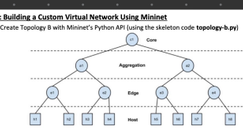
Part 2 Explanation:
With the host-to-host iperf approach used in topology-a, it is not possible to determine the bandwidth of the level 1 or 2 links in topology-b because the level 3 links form a bottleneck since every host-to-host path would have to include the level 3 links. This essentially prevents us from evaluating the bandwidth of the target switch-to-switch links because no matter how large their bandwidth is, our metrics will always read the bandwidth of the bottlenecking level 3 links (similar to how the transfer speed from a small pipe to a large pipe is constrained by the speed through the small pipe, regardless of how large the large pipe is)

Latency Explanation:
We can determine the latency of the targeted links by first calculating the latency of level 3 links: h1 to h2 only involves two level 3 links so to get the latency of one of them we simply divide by two. We are also depending upon the fact that the latencies are the same for each level, which is why dividing by 2 works, since both of the two links must have the same latency on account of them both being level 3 links. Now that we have the latency for level 3 links, we can use that in our calculation for level 2 link latency, and similarly divide by two to get the latency of a single level 2 link. Then we can use the calculated latencies for level 3 and 2 links in order to determine the latency for level 1 links, using the same approach. (exact calculations shown below)

Part 2: Building a Custom Virtual Network Using Mininet

Create Topology B with Mininet's Python API (using the skeleton code **topology-b.py**)



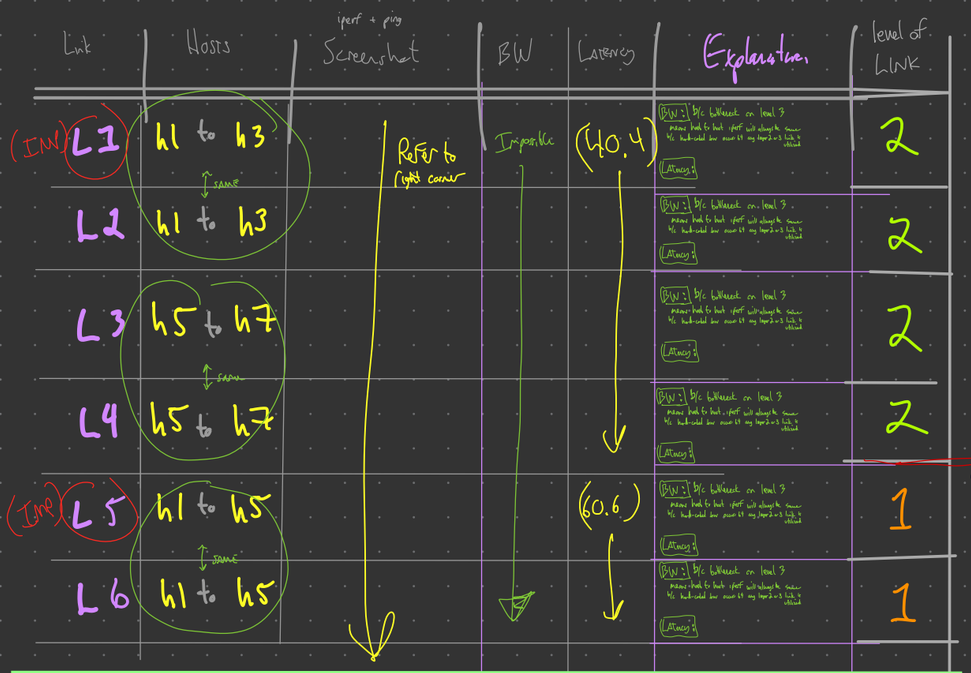
a. Note: Each level (i.e., core, aggregation, edge and host) is composed of a single layer of switches or hosts. (1: Core to Aggregation ,2: Aggregation to Edge ,3: Edge to Hosts)

b. Note: All links between hosts and edge switches will have the same predefined performance parameter. Here there are 3 types of links, the links between core and aggregation switches, the links between aggregation and edge switches, and the links between edge switches and hosts. You will see these specific parameters in the provided skeleton code (**topology-b.py**)

c. Create the topology in (**topology-b.py** or **topology-b.java**)

d. Evaluate Topology B similarly to Topology A (measurements of throughput and latency for links between switches (example: e1 to a1, a1 to c1, ... etc.). Write your answers in a file named (**topology-b-report.pdf**)

Latency Results:
Level 1 link(s): 60.6 ms
Level 2 link(s): 40.4 ms
Level 3 link(s): 21.6 ms



6 total = 2 required

all these 4 are layer-2 link so will have same metrics

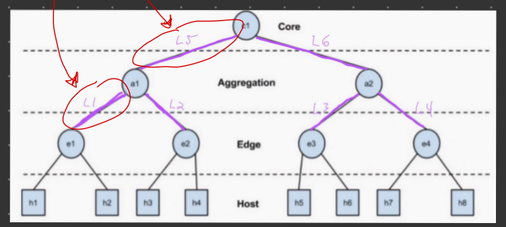
all these two are layer 1 links so will have same mtrc

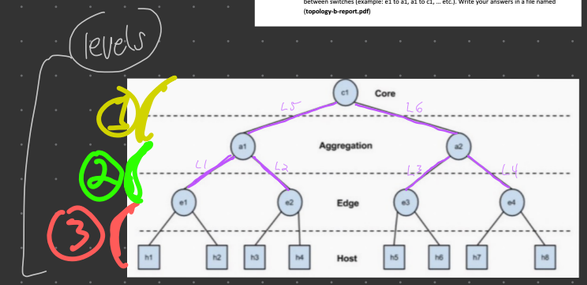
total level 3 links

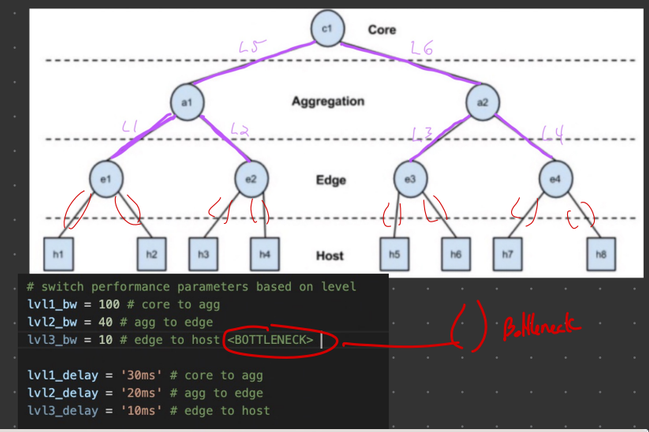
total level 2 links

total level 1 links

The links between layers will all have the same bandwidth and delay/latency
- so really only need to evaluate L1 and L5, since L1 will be same as L2,3,4 and L5 will be same as L6








```
mininet@mininet-vm: ~
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=59.9 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=41.3 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=40.9 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=40.6 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=40.5 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=40.3 ms
^C
--- 10.0.0.2 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7015ms
rtt min/avg/max/mdev = 40.288/43.147/59.901/6.338 ms
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=136 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=123 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=122 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=6 ttl=64 time=121 ms
^C
--- 10.0.0.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 121.112/124.186/136.488/5.537 ms
mininet> h1 ping h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=259 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=244 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=5 ttl=64 time=243 ms
64 bytes from 10.0.0.5: icmp_seq=6 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=7 ttl=64 time=242 ms
^C
--- 10.0.0.5 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6010ms
rtt min/avg/max/mdev = 241.834/244.868/258.751/5.703 ms
mininet>
```

LATENCY CALCULATIONS

$\left(\frac{43.15}{2}\right) = 21.6$ layer 3 links

$\frac{(124 - 2(21.6))}{2} = 40.4$ layer 2 links

$L5 = \frac{(244.7 - (2(21.6) + 2(40.4)))}{2} = 60.6$ layer 1 links

```
mininet@mininet-vm: ~
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=59.9 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=41.3 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=40.9 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=40.6 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=40.5 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=40.3 ms
^C
--- 10.0.0.2 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7015ms
rtt min/avg/max/mdev = 40.286/43.147/59.901/6.338 ms
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=136 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=123 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=122 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=6 ttl=64 time=121 ms
^C
--- 10.0.0.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 121.112/124.186/136.488/5.537 ms
mininet> h1 ping h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=259 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=244 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=5 ttl=64 time=243 ms
64 bytes from 10.0.0.5: icmp_seq=6 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=7 ttl=64 time=242 ms
^C
--- 10.0.0.5 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6010ms
rtt min/avg/max/mdev = 241.834/244.869/258.751/5.703 ms
mininet>
```

total = 244.7

total = 124

$(124 - 2(21.6)) = 2(\text{layer 2 links})$

$\frac{(124 - 2(21.6))}{2} = \text{layer 2 link} = 40.4$

$L5 = \frac{(total - (2z + 2y))}{2}$

$L5 = \frac{(244.7 - (2(21.6) + 2(40.4)))}{2} = \frac{(244.7 - 124)}{2} = 60.35$

(21.6) (40.4)

60.6
40.4
21.6

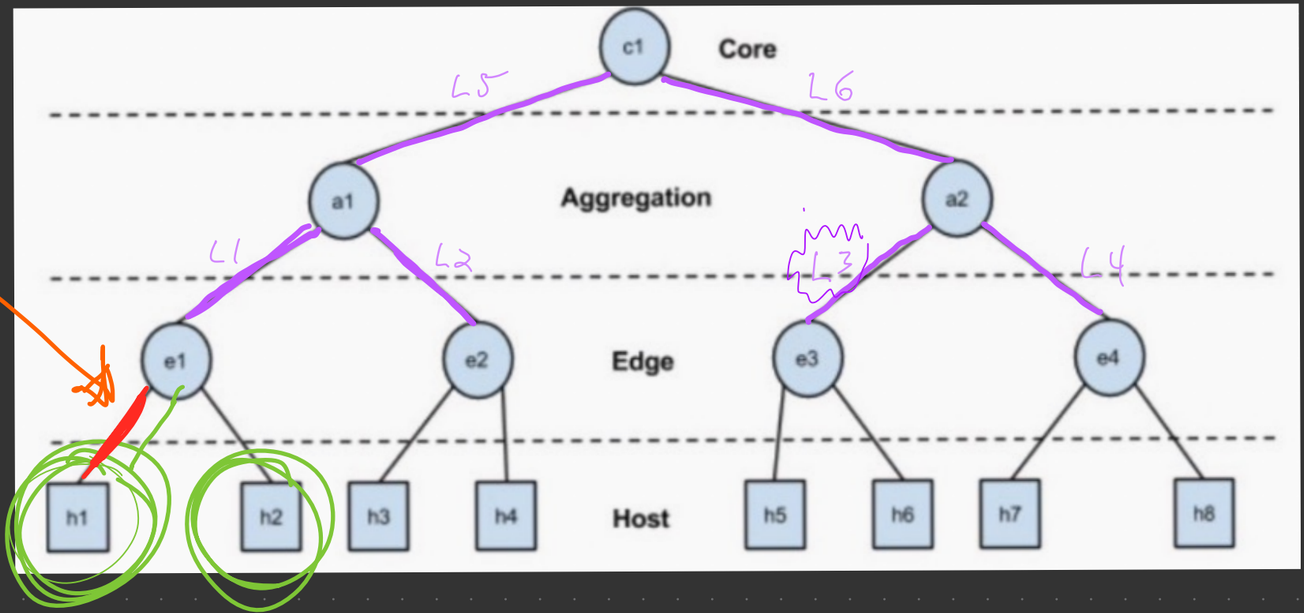
```
lvl1_delay = '30ms' # core to agg
lvl2_delay = '20ms' # agg to edge
lvl3_delay = '10ms' # edge to host
```

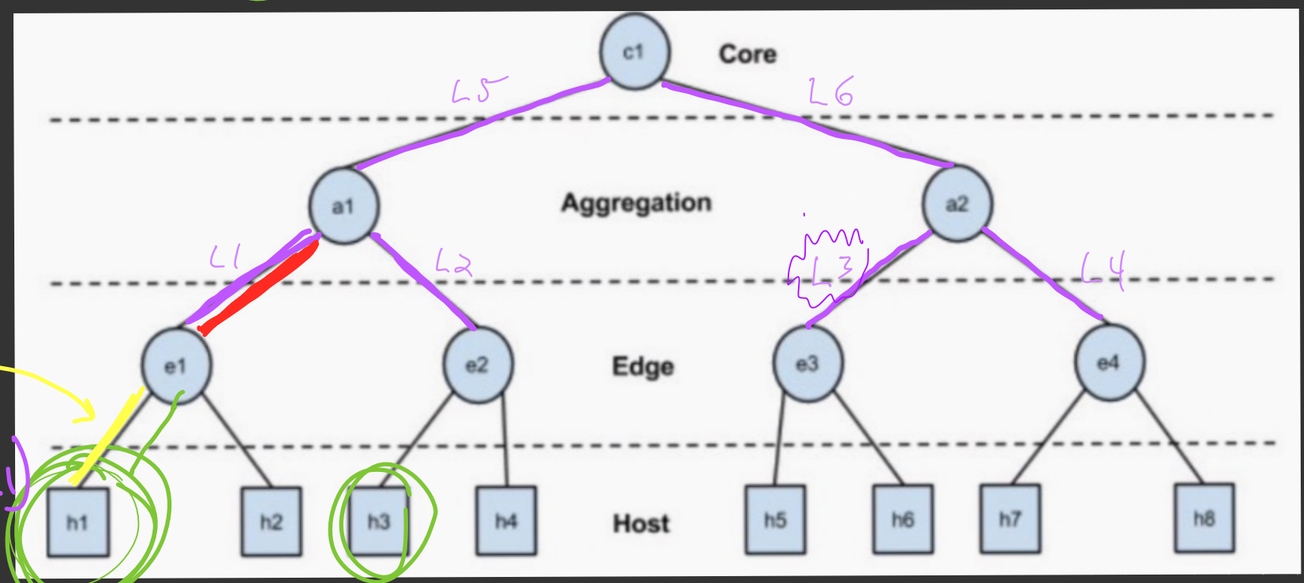
✓ SAME layer-to-layer PROPORTIONS ☆

PING

h1 to h2, then ÷ by 2 to get z

z = latency of layer 3 links, like h1 to e1





h1 to h3, to get total, and total = 2(z) + 2(L1)

total − 2(layer3 link) = cost of L1 & L2

$\left(\frac{\text{total} - 2(\text{layer3 link})}{2}\right)$ = cost of L1 = cost of layer2 link

so to get L1 alone, we do:

$L1 = \frac{(\text{tot} - 2(z))}{2}$

y = latency of layer 2 links, like e1 to a1

to Get (L5) ...

(h1) to (h5), to get Total,

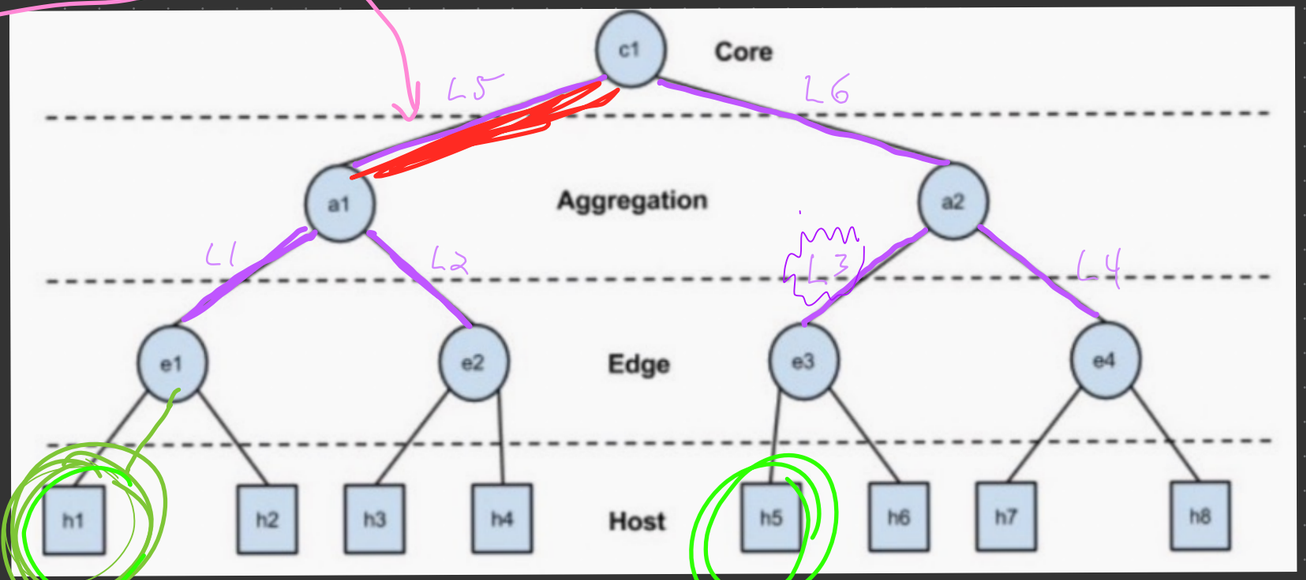
Total = 2(z) + 2(y) + 2(x)

2(L5) = 2(x), (L5) = x

$$2X = (t_{total} - (2z + 2y))$$

$$L5 = \frac{(t_{total} - (2z + 2y))}{2}$$

X = latency of layer 1 links, like a1 to c1 ≡ L5




```
mininet@mininet-vm: ~
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=59.9 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=41.3 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=40.9 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=40.6 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=40.5 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=40.8 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=40.3 ms
^C
--- 10.0.0.2 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7015ms
rtt min/avg/max/mdev = 40.286/43.147/59.901/6.338 ms
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=136 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=123 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=122 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=121 ms
64 bytes from 10.0.0.3: icmp_seq=6 ttl=64 time=121 ms
^C
--- 10.0.0.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 121.112/124.186/136.488/5.537 ms
mininet> h1 ping h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=259 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=244 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=5 ttl=64 time=243 ms
64 bytes from 10.0.0.5: icmp_seq=6 ttl=64 time=242 ms
64 bytes from 10.0.0.5: icmp_seq=7 ttl=64 time=242 ms
^C
--- 10.0.0.5 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6010ms
rtt min/avg/max/mdev = 241.834/244.868/258.751/5.703 ms
mininet>
```