CS 118: Route Computation

George Varghese

Nov 2022

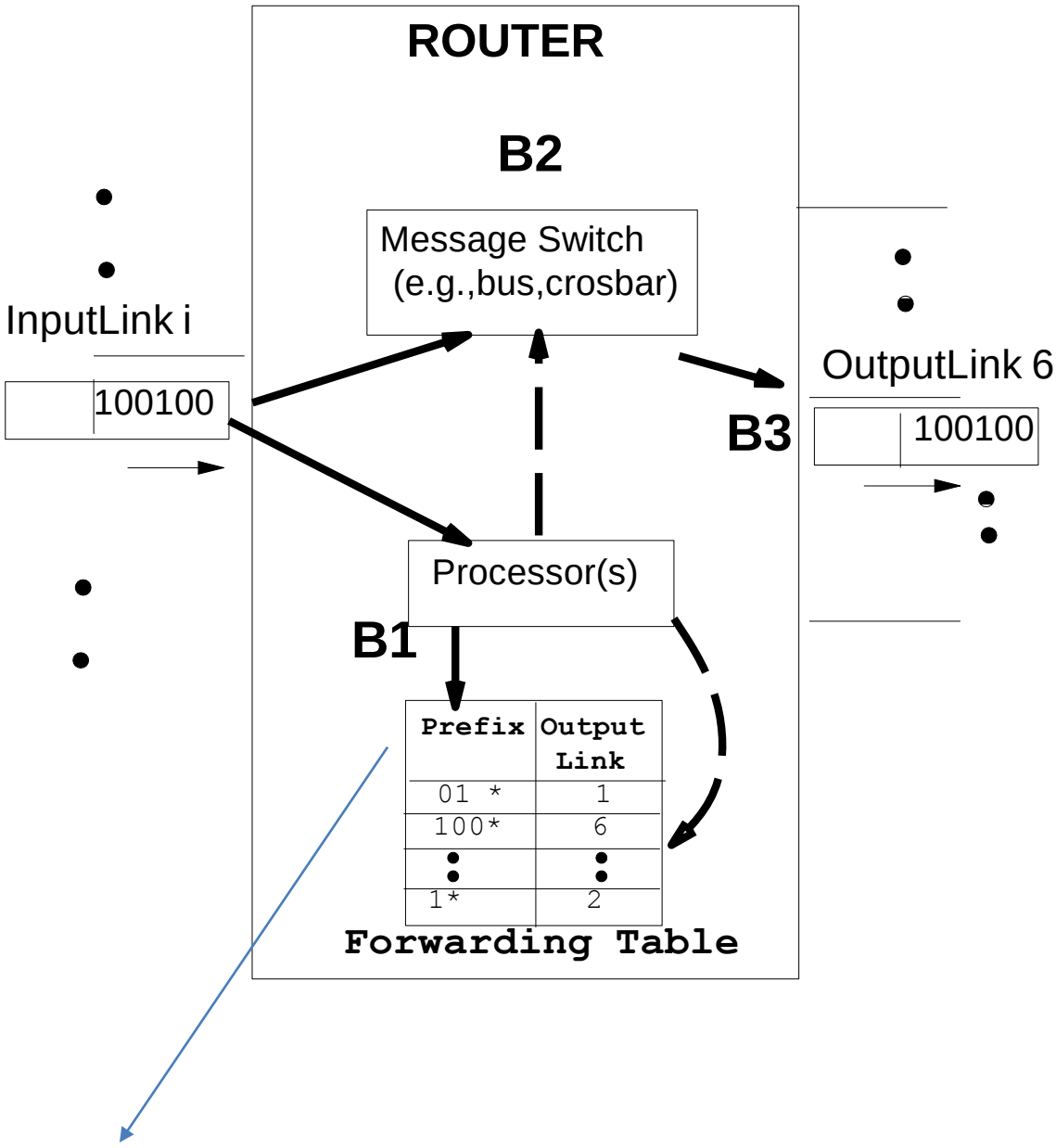
Inside Google's Software-Defined Network |

Imagine you are a kid out of UCLA and get hired in Google's NetInfra team where they are redoing their wide area network (B4) connecting their data centers. How could you rethink route computation?

Amin Vahdat, VP of Google Infrastructure on the Internet's success and problems

https://www.youtube.com/watch?v=DpO1Tfa4IZ4, 0 -3:17

ROUTE COMPUTATION?



Who builds this table? Route Computation

Implemented in a separate processor other than the forwarding ASICs

PART 1: THE BIG PICTURE

Only focus on routing within organizations. Next lecture on routing between organizations via BGP

Four parts to Routing

All work in parallel:

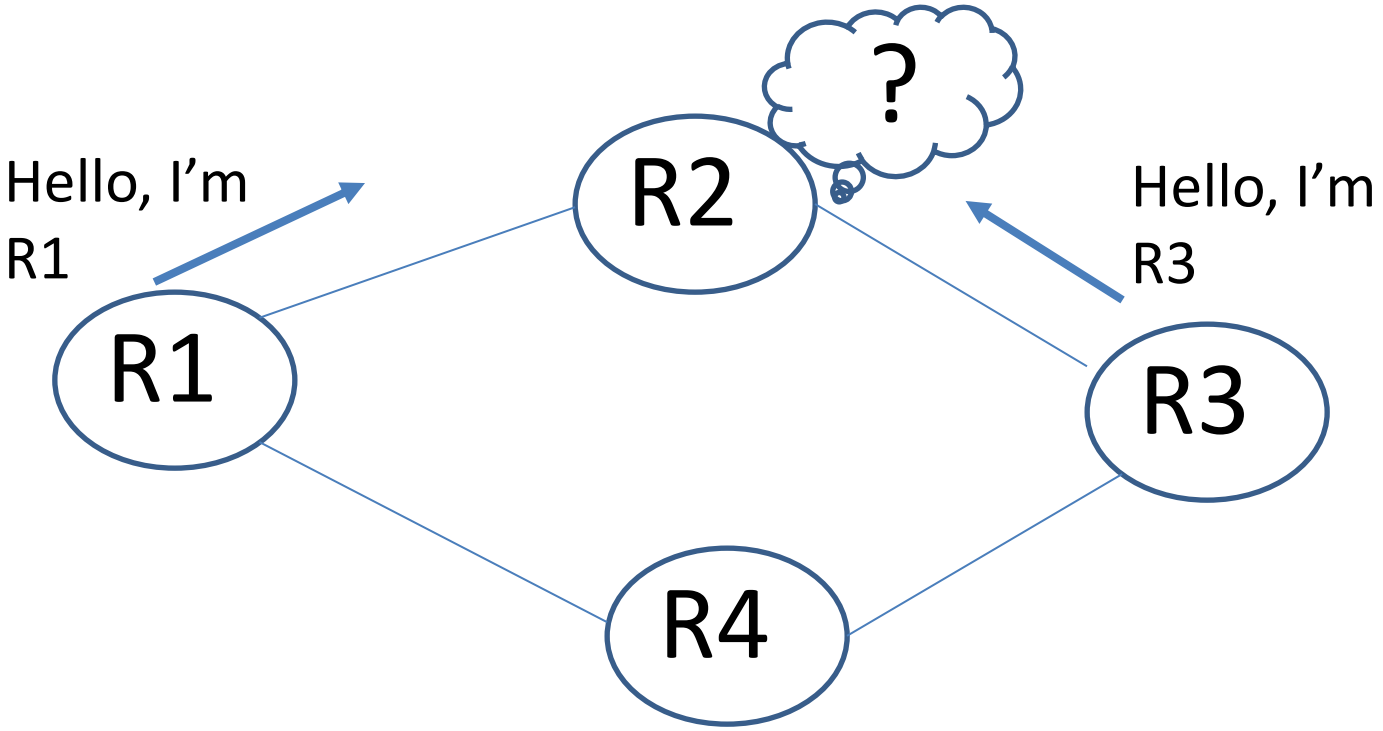
- *Set up Addresses and Topology* assign IP addresses, connect routers
- *Neighbor Determination* endnodes talk to routers (things like ARP), Routers to router neighbors
- *Compute Routes* most complex piece, this lecture but only for within organizations.
- *Forward packets* as we studied last lecture

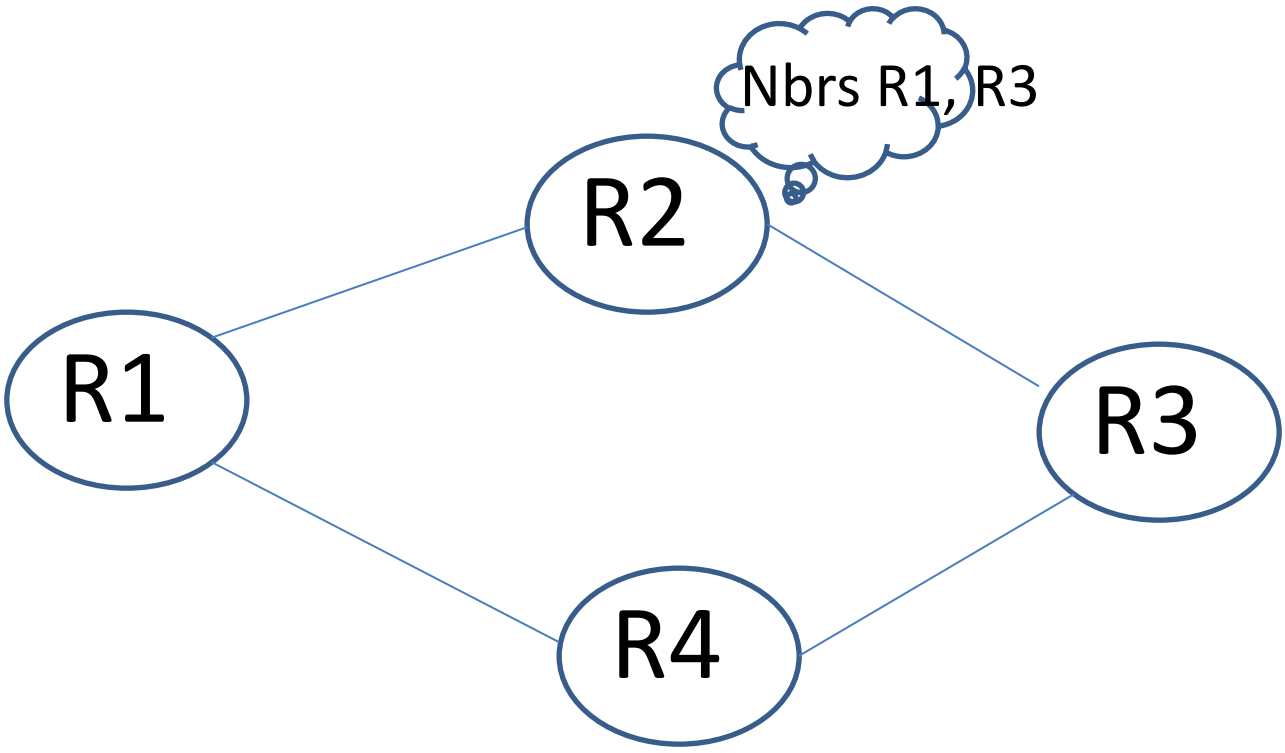
Flavors of Route Computation

First division:

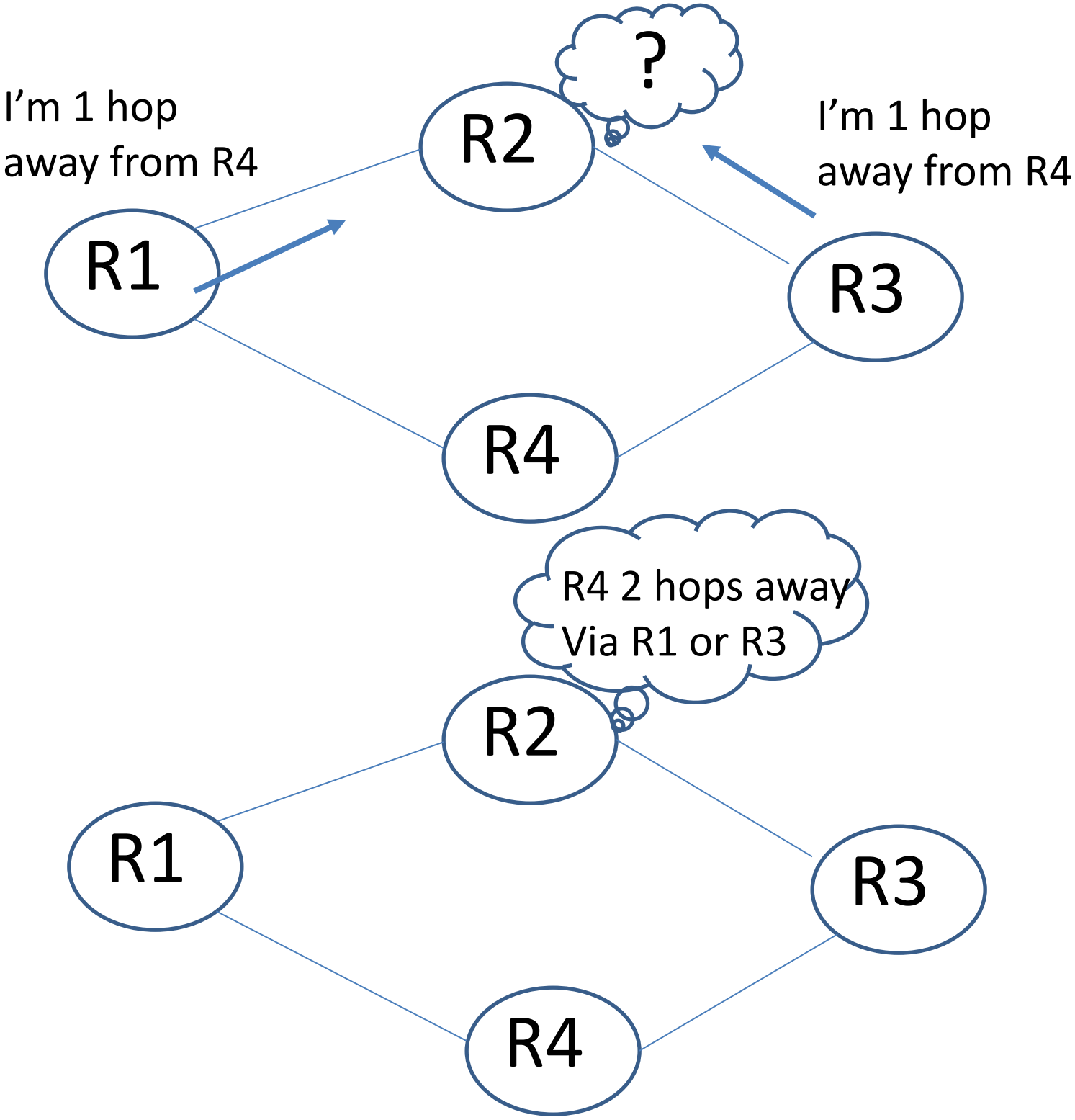
- *Intradomain routing* routing within an enterprise managed by one entity (AS, domain: e,g, UCLA, Level 3). Often shortest paths. Will study 2 flavors:
  - *Distance Vector* gossip protocol. Has problems with failures, so-called count to infinity
  - *Link State* more resilient to failures, Often used
- *Interdomain routing* routing between ISPs owned by different entities. Policy routing (next lecture)

START BY TALKING TO NEIGHBORS





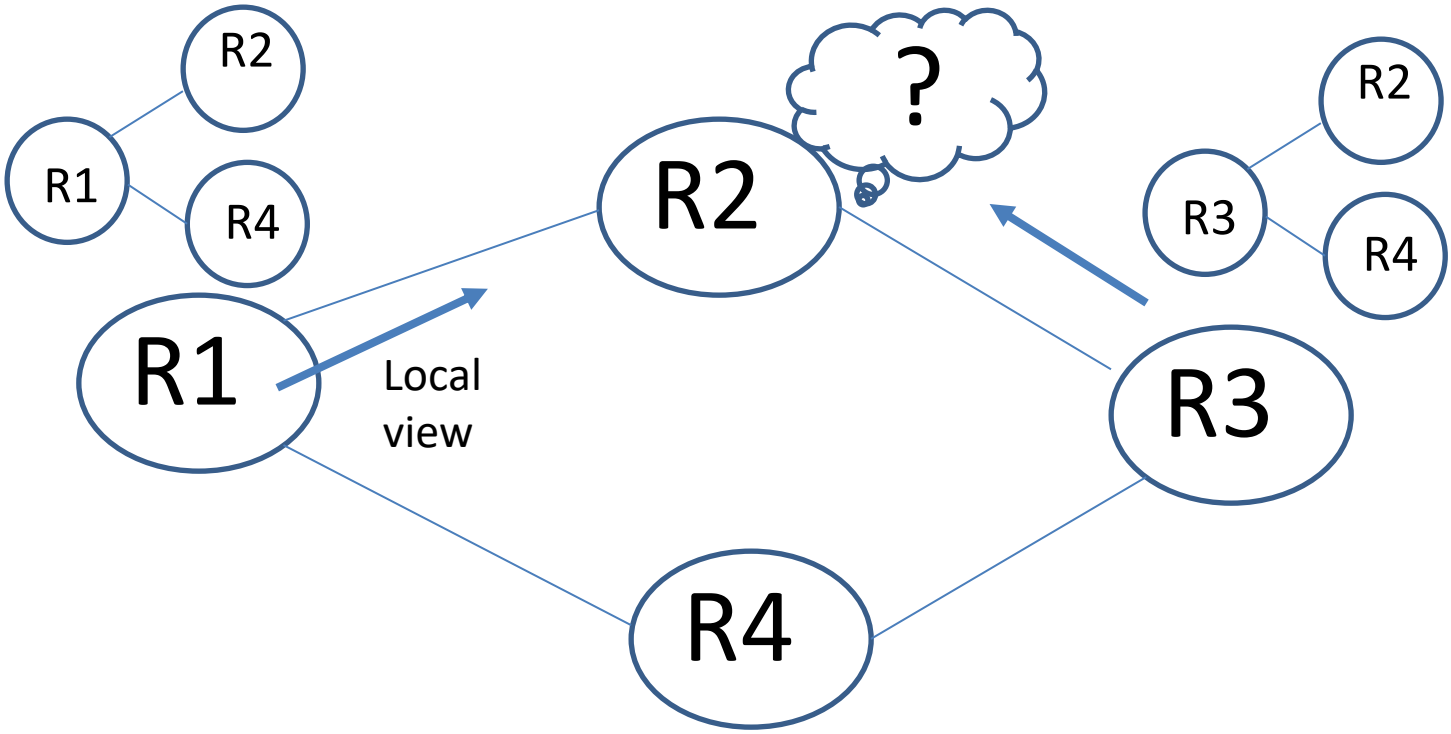
BEYOND 1 HOP:GOSSIP, DISTANCE VECTOR

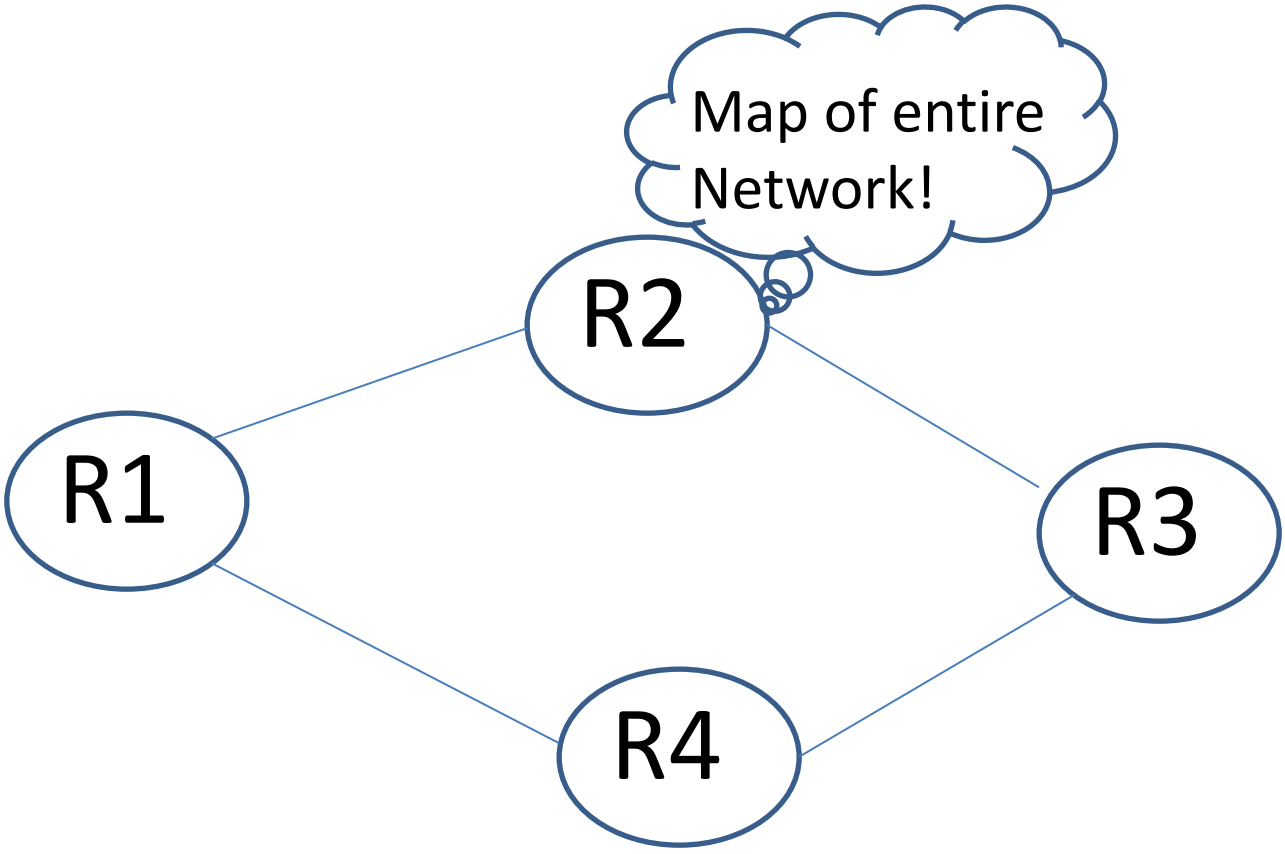


Like Spanning Tree but not just from a root node

PROBLEMS WITH GOSSIP: WHAT HAPPENS IF R4 CRASHES? WHAT COULD GO WRONG?

BEYOND 1 HOP: GLOBAL VIEW, LINK STATE



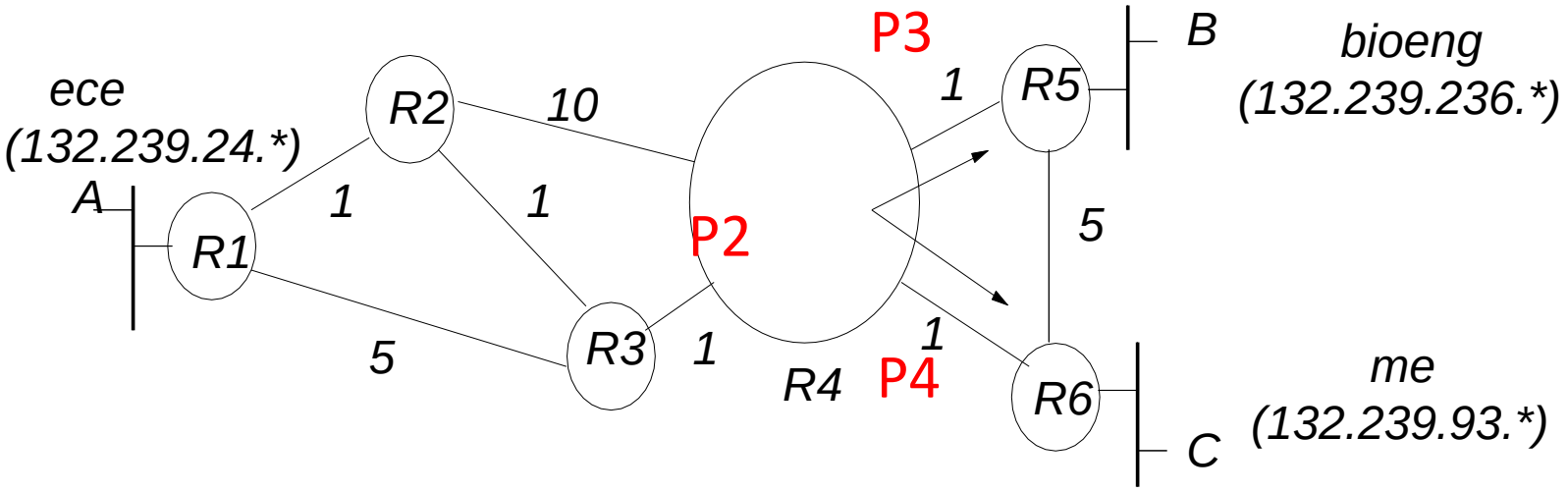


GLOBAL VIEW IS GREAT: CAN NOW USE SHORTEST PATH ALGORITHM FROM CS 180 LIKE DIJKSTRA'S ALGORITHM, BUT ALSO HARD BECAUSE?

PART 2: DISTANCE VECTOR

Routers gossip with neighbors to spread information but does badly with node failures

Mythical Topology

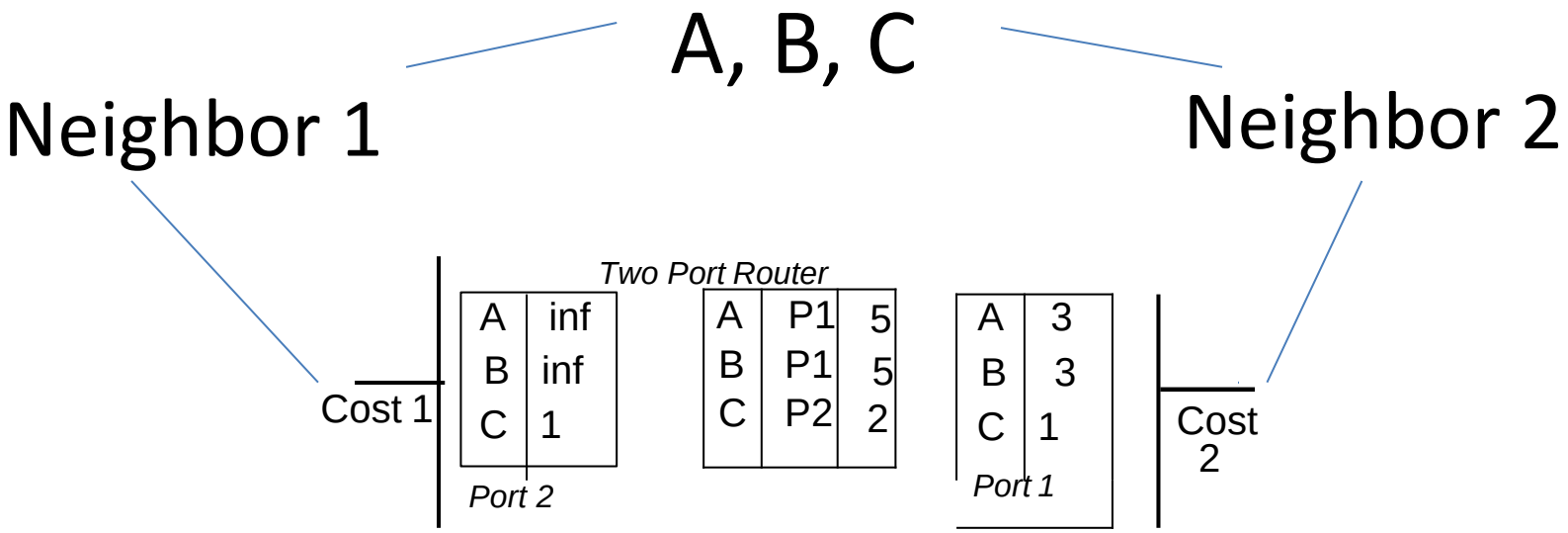


Prefix	Next hop interface
ece	P2
bioengineering	P3
mechanical	P4

Distance Vector

- Two principal methods for Route Computation: distance vector (IP) and link state (OSI). We will study both.
- Distance vector: how can we use spanning tree protocol idea. We found distances to min ID node by "gossip". Use same idea for updating distance to all nodes.
- Previously we kept (Root, distance, parent). Now we keep a vector of (ID, distance). Hence called distance vector.

Distance Vector Databases

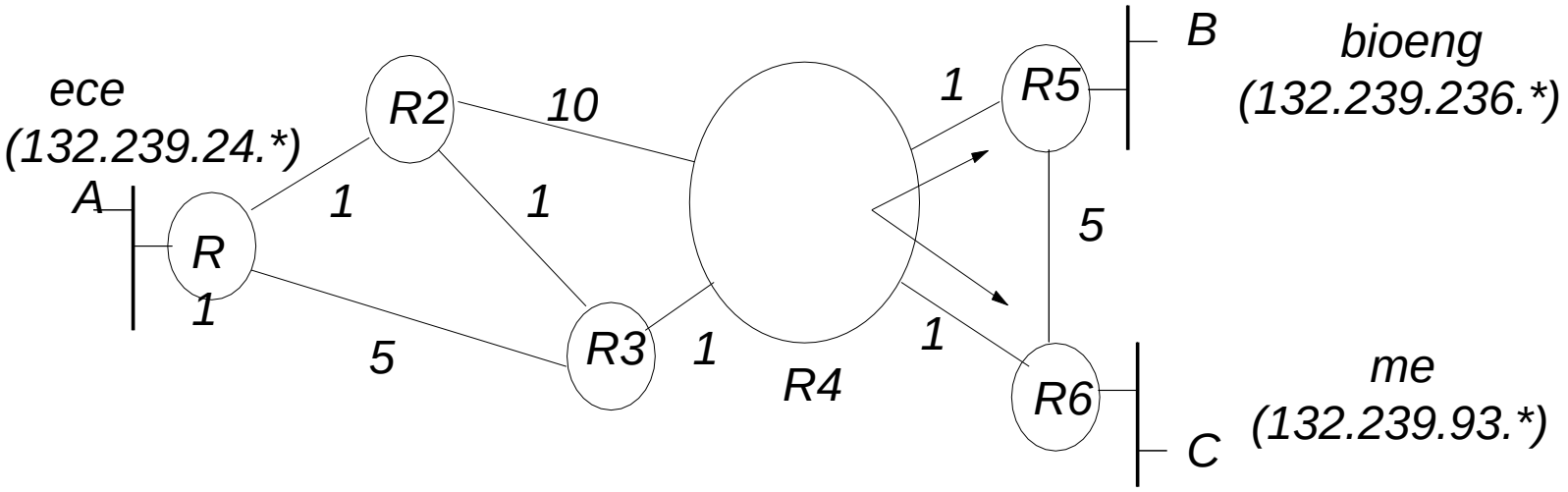


- As in Spanning Tree, we have port and central databases.
- Central is computed based on best port database.

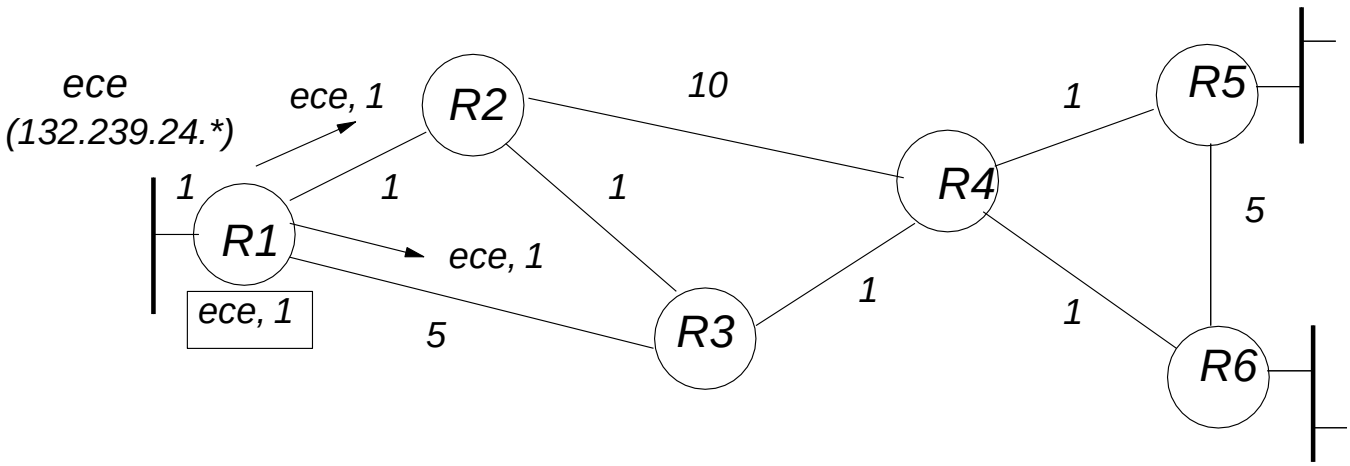
Link Failures and Distance Vector

- On link failure, delete stored distance vector for that port. Link failure is reported by neighbor discovery (because we haven't received a hello from that neighbor for a while).
- Different from spanning tree in that there is no aging of information except the local aging used to detect link failures. Instead must rely on something call count-to-infinity.
- Also unlike spanning tree, you send whenever information changes. Periodic sending is a good idea for robustness but not strictly necessary.

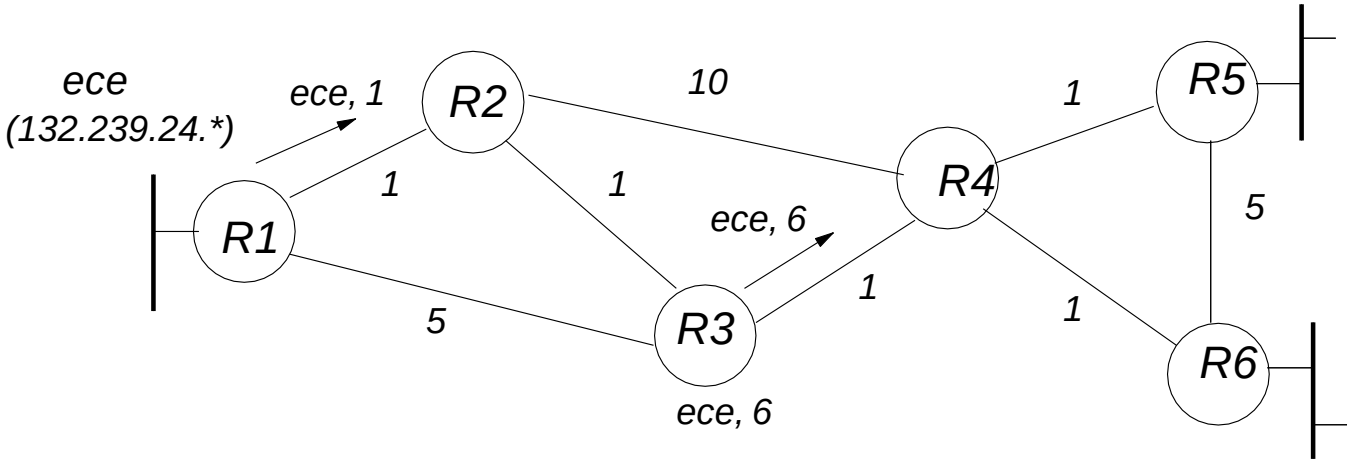
Mythical Topology



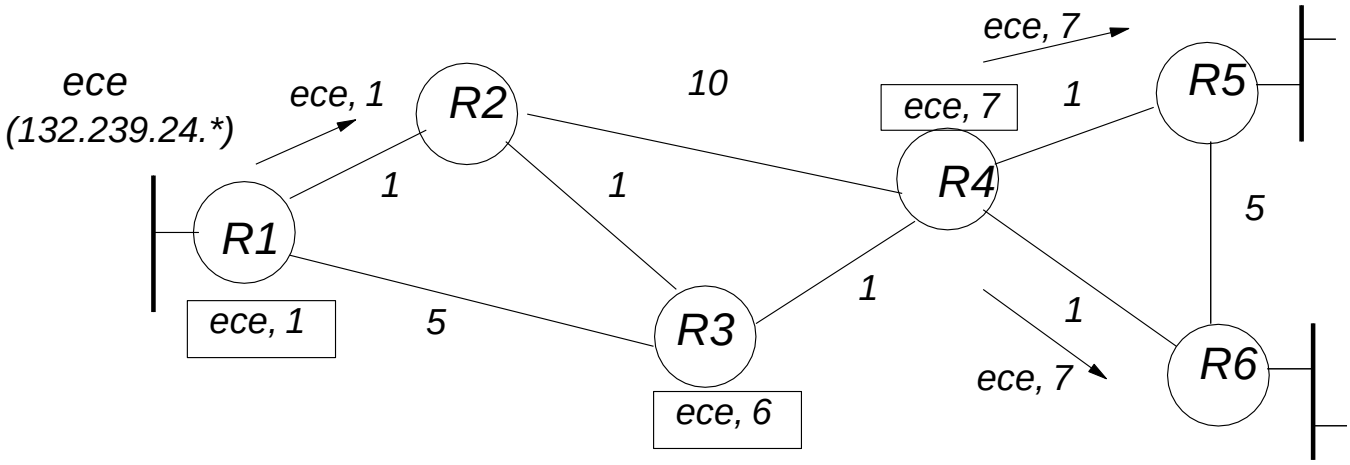
Snapshot 1



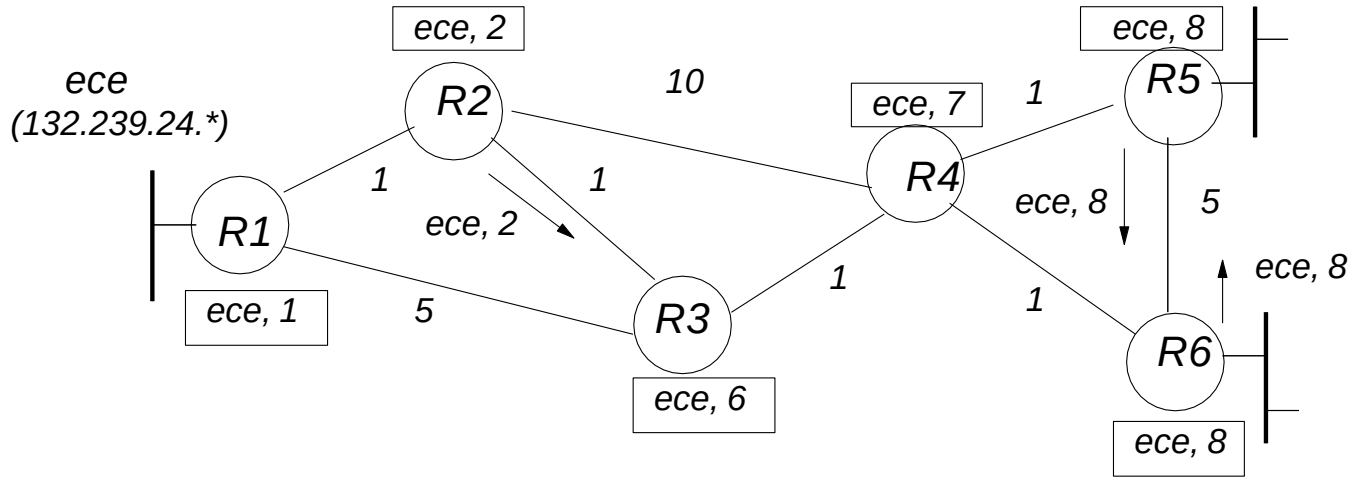
Snapshot 2



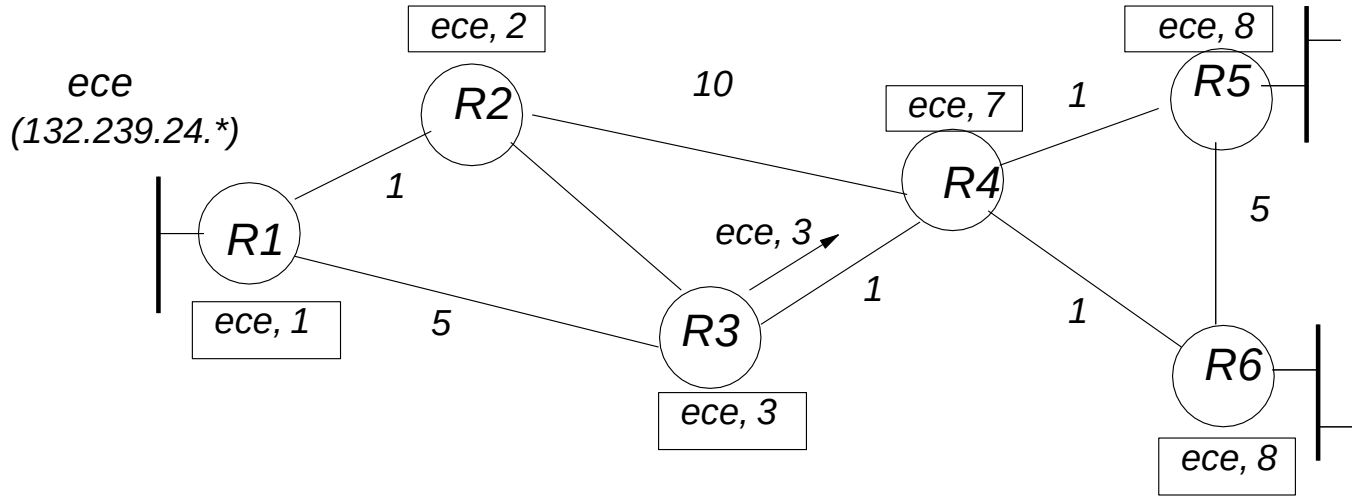
Snapshot 3


ece
(132.239.24.*)
ece, 1
R2
10
ece, 7
ece, 7
1
R5
R4
1
1
5
R1
1
ece, 1
5
R3
1
1
ece, 7
R6
ece, 6

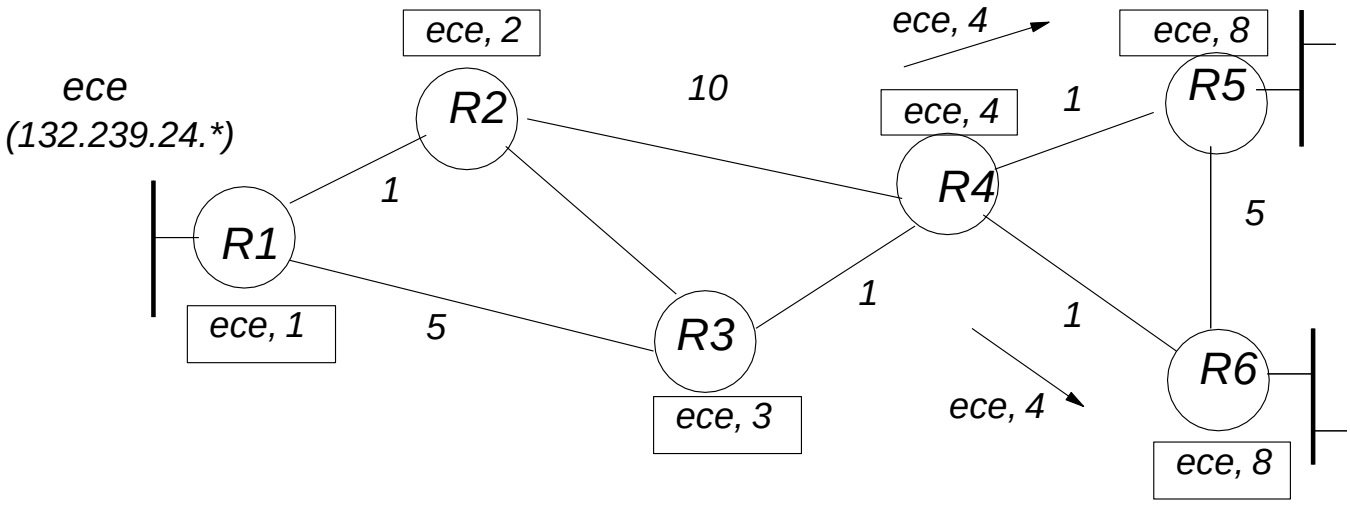
Snapshot 4


ece
(132.239.24.*)
ece, 2
R2
10
ece, 7
1
R5
ece, 8
1
1
R4
ece, 8
5
R1
ece, 2
ece, 8
ece, 1
5
R3
1
1
R6
ece, 6
ece, 8

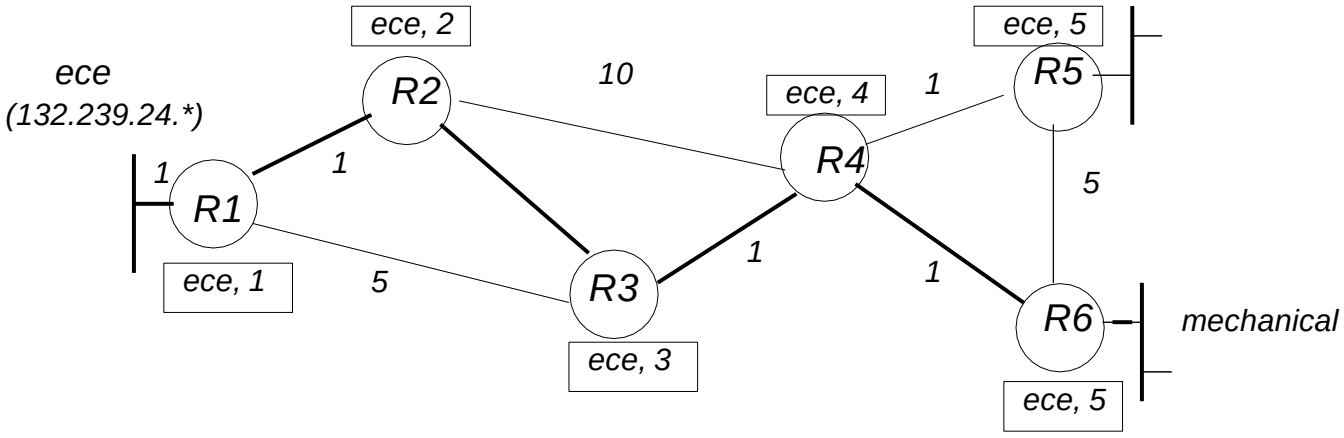
Snapshot 5


ece
(132.239.24.*)
ece, 2
R2
10
ece, 7
1
ece, 8
R5
R4
1
5
ece, 3
R1
ece, 1
5
1
1
R3
R6
ece, 3
ece, 8

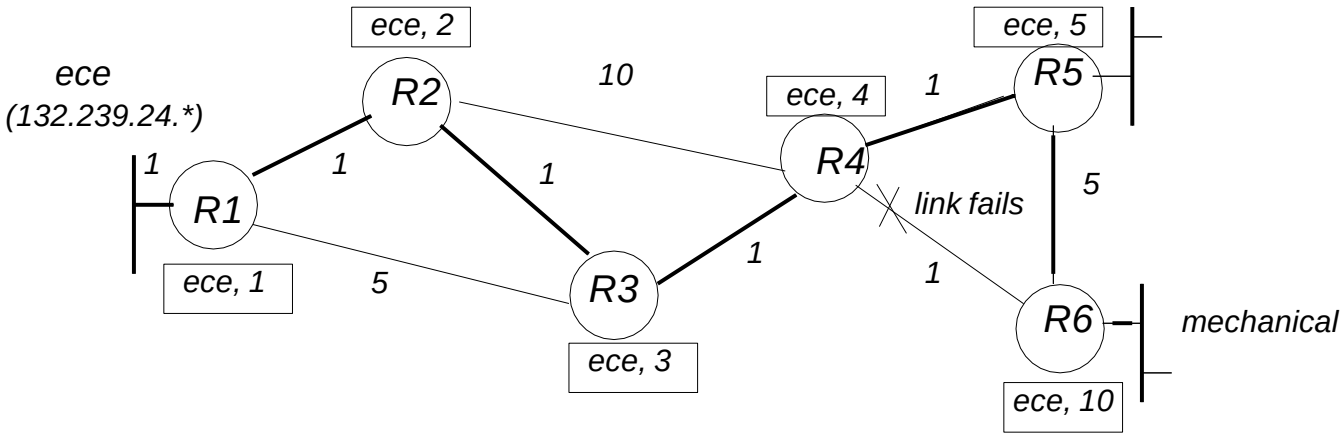
Snapshot 6


ece
(132.239.24.*)
R1
ece, 1
R2
ece, 2
1
10
5
R3
ece, 3
R4
ece, 4
1
1
1
ece, 4
R5
ece, 8
5
R6
ece, 4
ece, 8

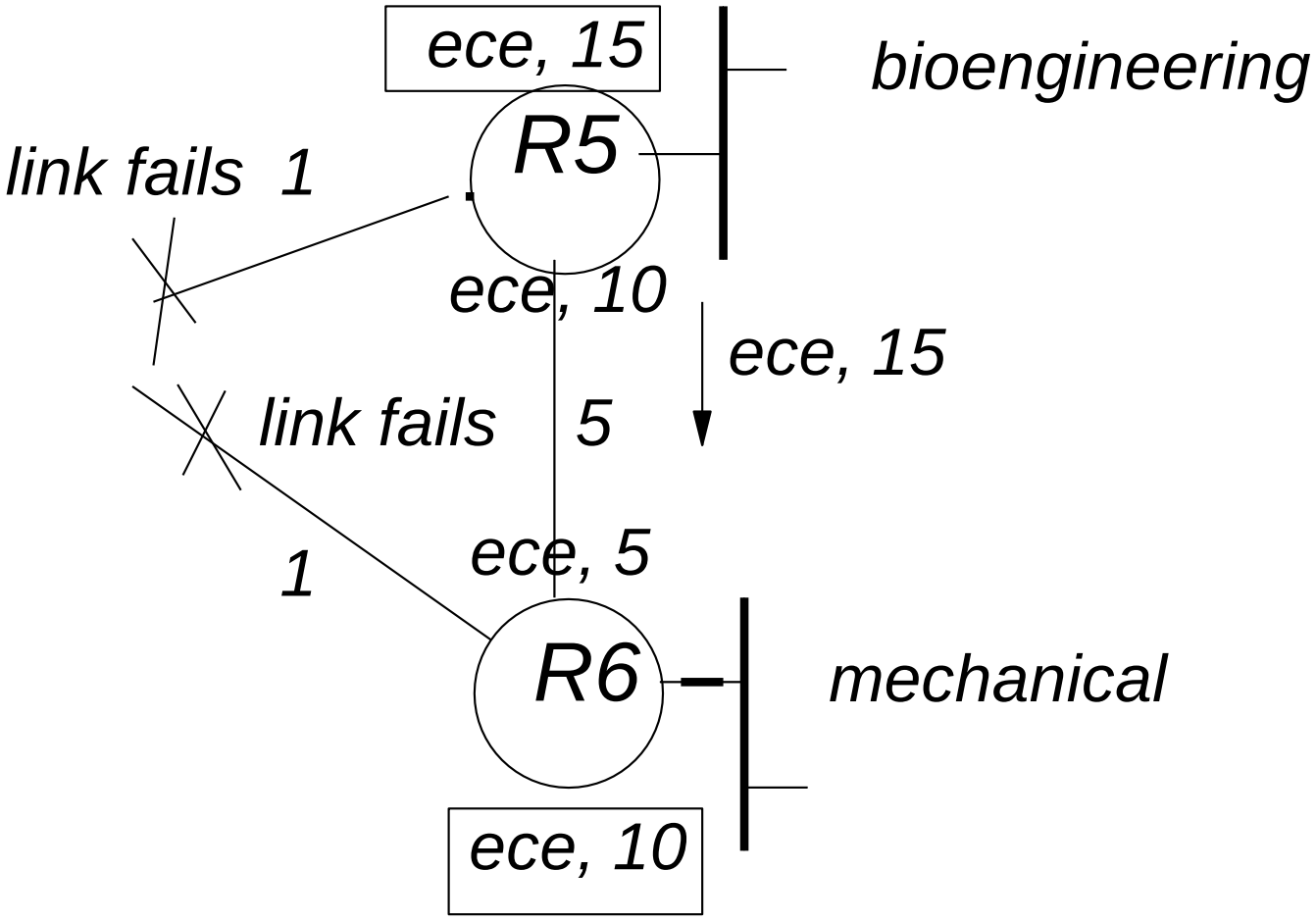
Snapshot 7


ece
(132.239.24.*)
1
R1
ece, 1
1
R2
ece, 2
10
5
R3
ece, 3
ece, 4
R4
1
1
1
R5
ece, 5
5
1
R6
ece, 5
mechanical

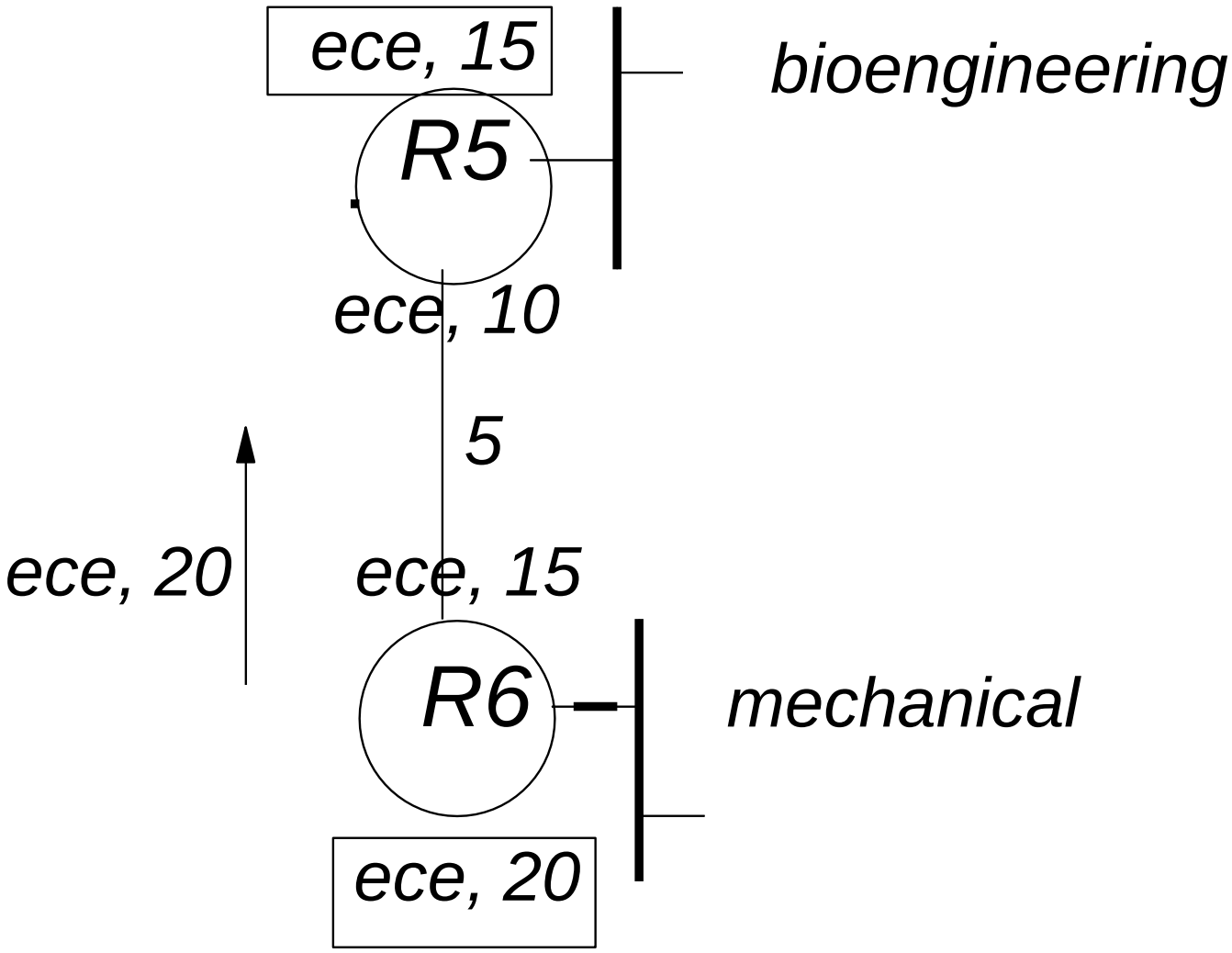
Snapshot 8: 1 Link Fails



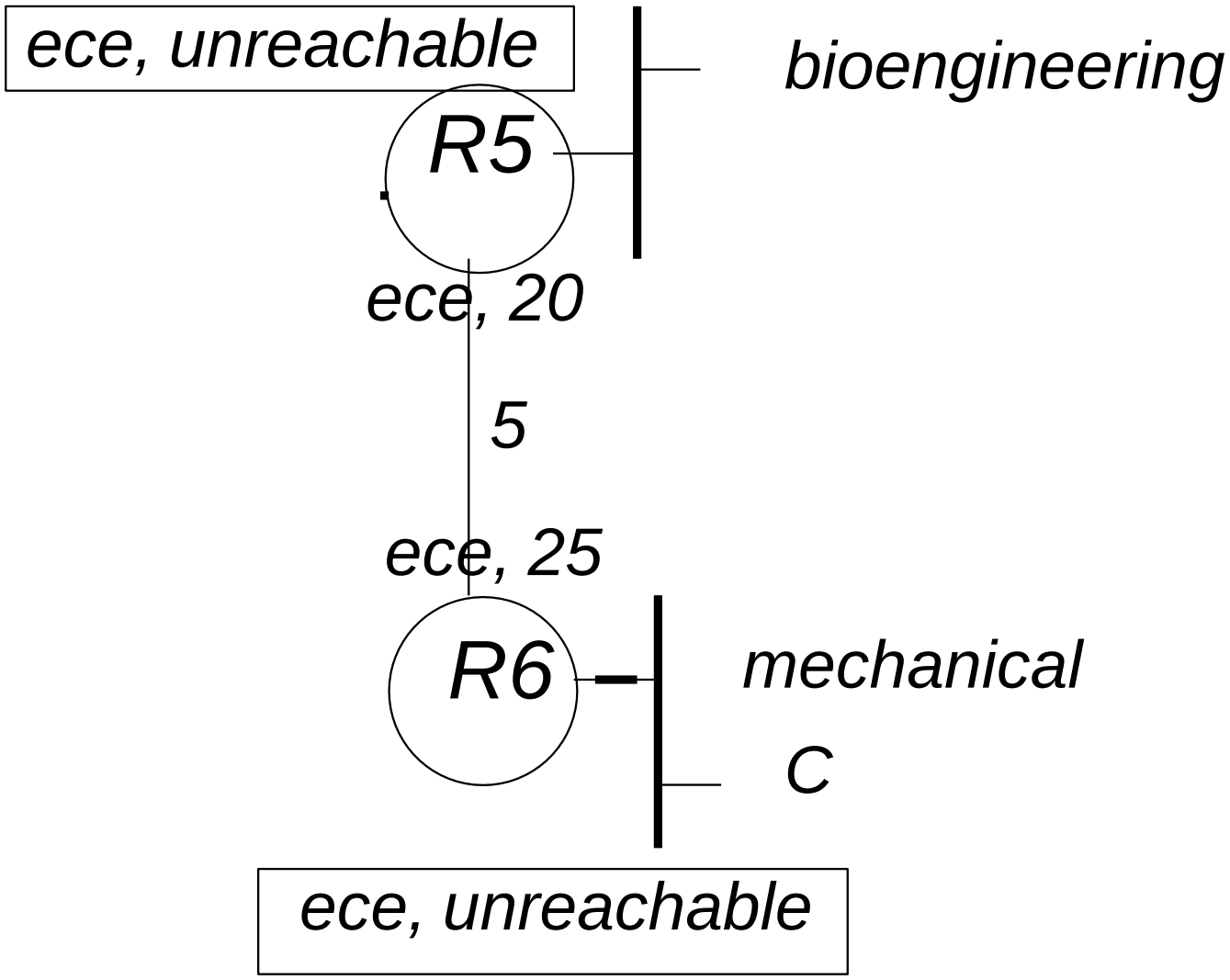
Snapshot 9: 2 Links fail



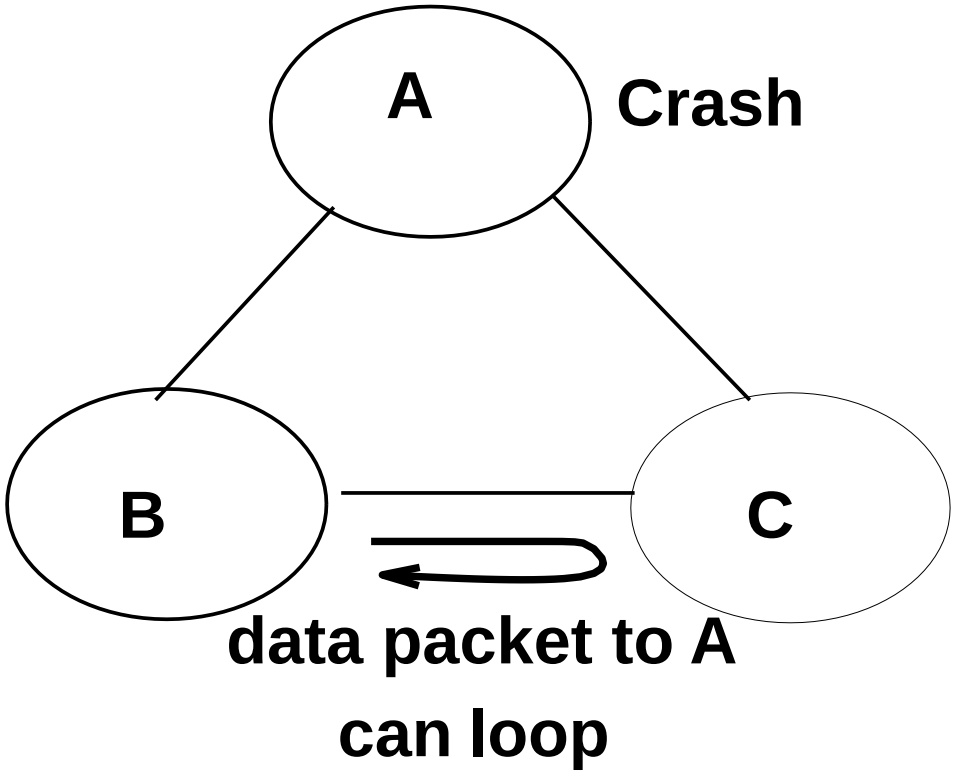
Snapshot 10: Count up



Snapshot 11: Convergence



Data Packet Looping



- After A crashes, B and C keep thinking the best way to get to A is through each other.
- Thus a data packet destined to A will keep looping until either the hop-count in the packet reaches its maximum value or B and C finally decide that A is down.

PART 3: LINK STATE

Each router broadcasts its local neighborhood to all other routers within its organization but must solve bootstrapping problem

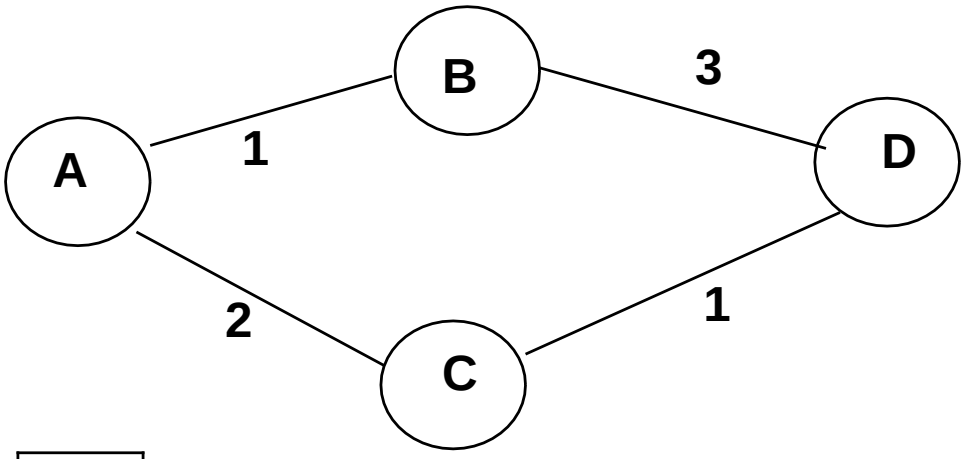
Link State History

- The ARPANET is a large national network that is part of the global Internet. (ARPANET has a net number). Classic network in a historic sense.
- Originally, ARPANET used distance vector. However, failure recovery times were very slow after node failures because of count-to-infinity problem. Also data packets kept looping during this period.
- New ARPANET moved to link state routing which has quicker response to failures and no count-up problem. Similar design used by OSPF inside most ISPs.

Link State: the basic idea

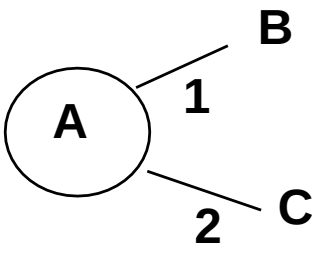
- Each node knows the default (or manager settable) cost of its outgoing links. Neighbor discovery is used to compile a list of neighbors that are UP. This information, along with link costs, is placed in a Link State Packet (LSP).
- Each source broadcasts its LSP to all other nodes using a primitive flooding mechanism called intelligent flooding.
- After the LSP propagation process stabilizes. each node has a complete and identical picture of the network graph. Then each node S uses any shortest path algorith (i.e., Dijkstra's) to compute the next node on the shortest path from S to every other node D.

LSP Generation

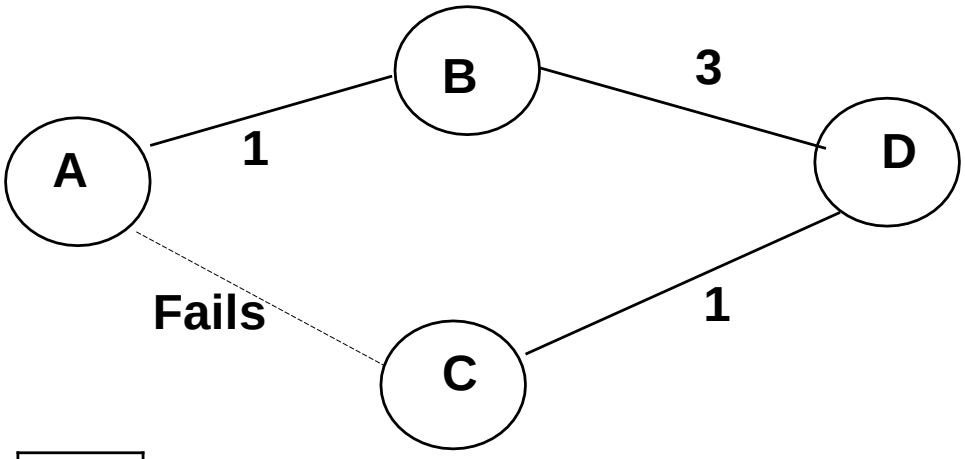


LSP
A
B
1
C
2

which means:

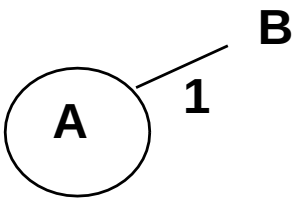


LSP Generation on Failure



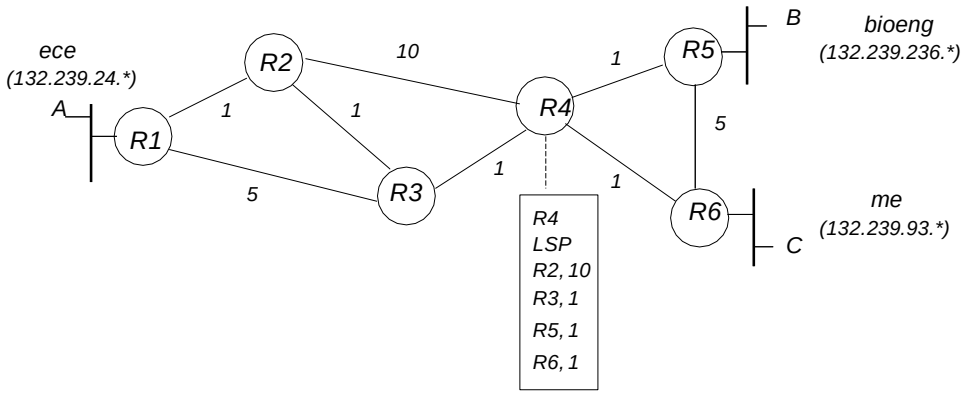
LSP
A
B
1

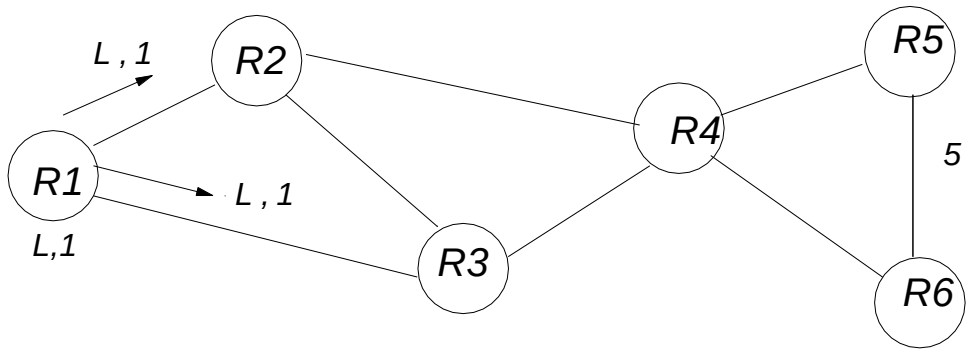
which means:

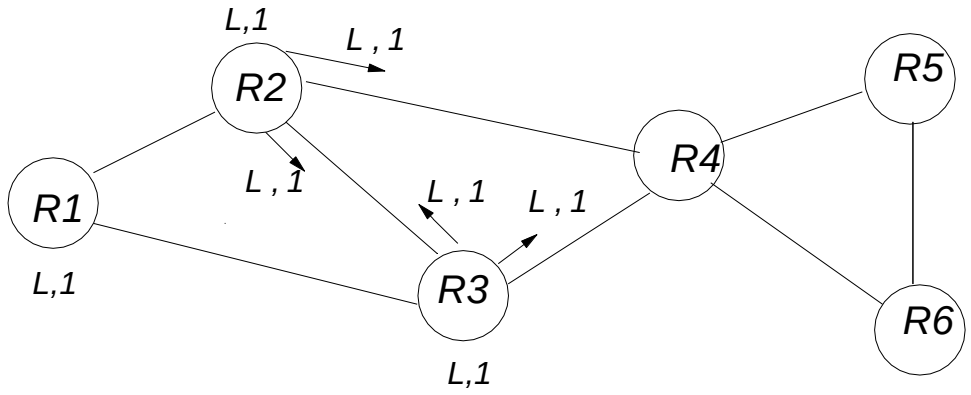


- If link AC fails, neighbor discovery in A and C will eventually detect failure.
- Only A and C recompute their LSP values and broadcast their LSPs again to all other nodes. Other nodes do not recompute or rebroadcast their LSPs.

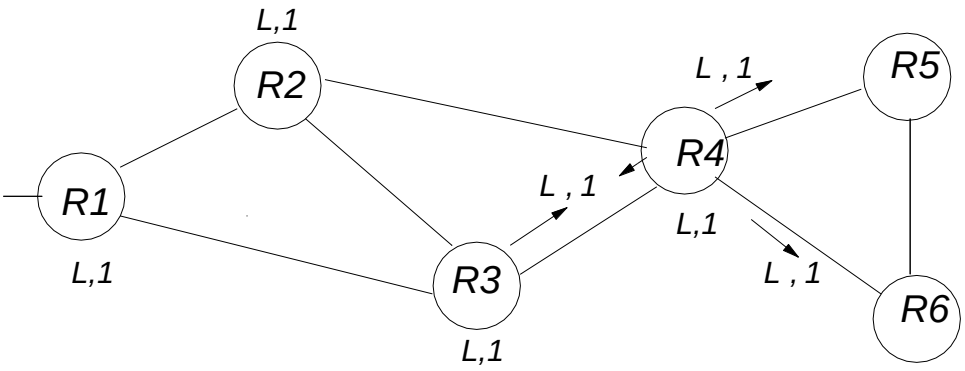
LSP Flooding

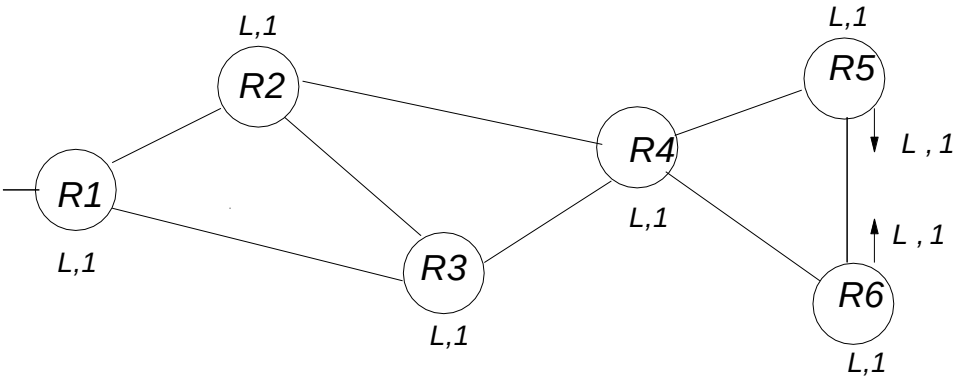


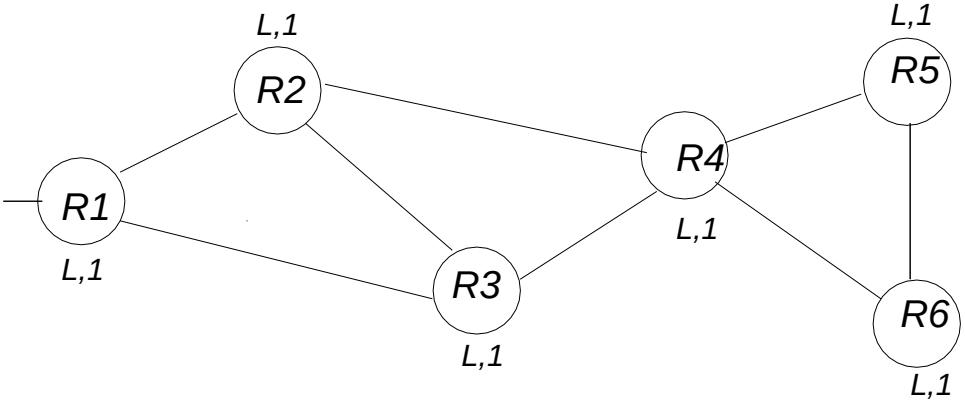




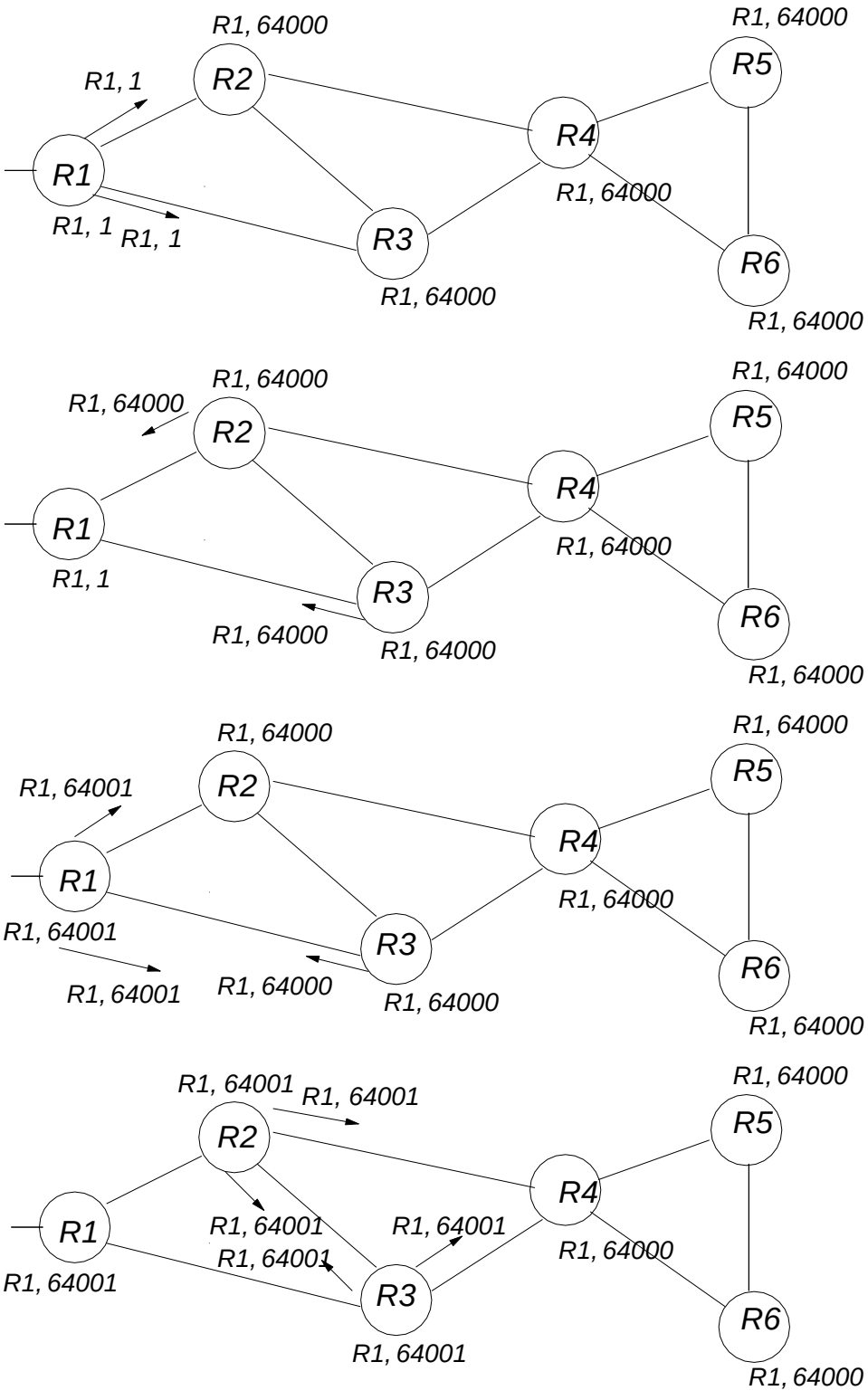
LSP Flooding Continued



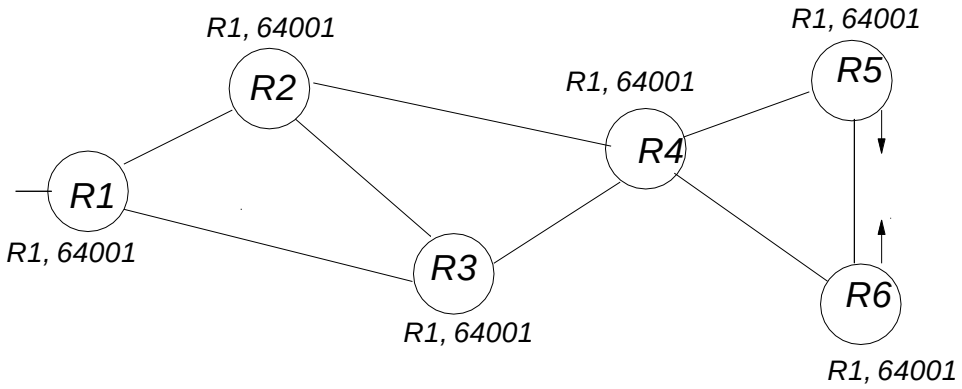




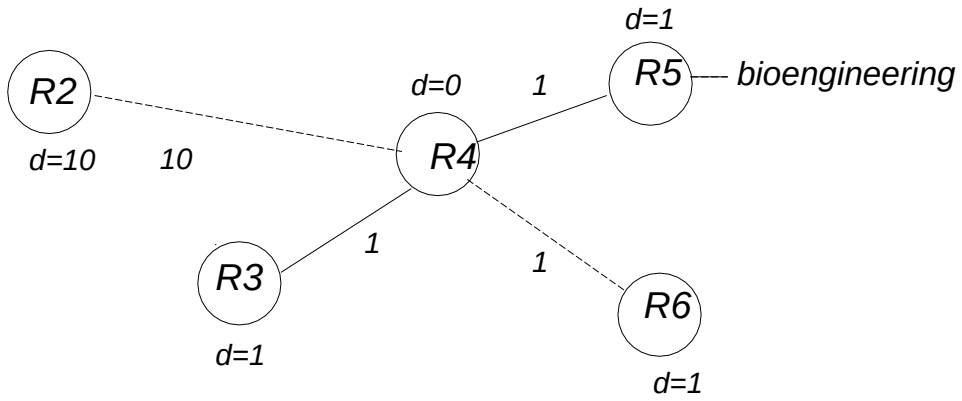
LSP Propagation after a crash

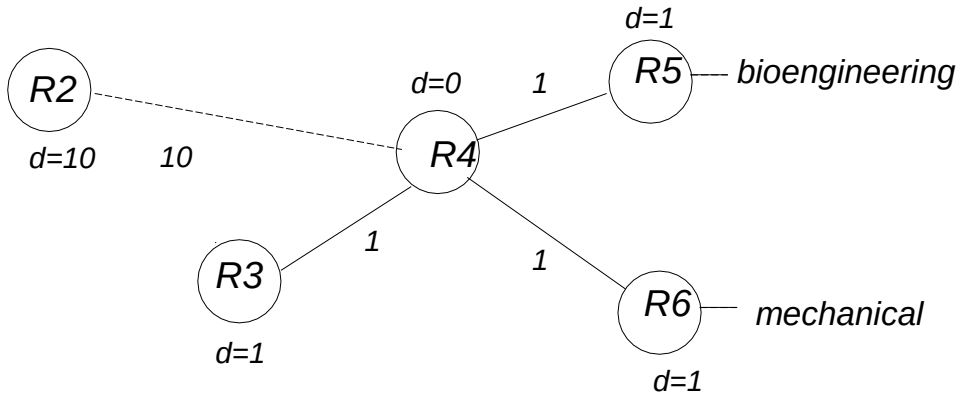


LSP Convergence after a crash

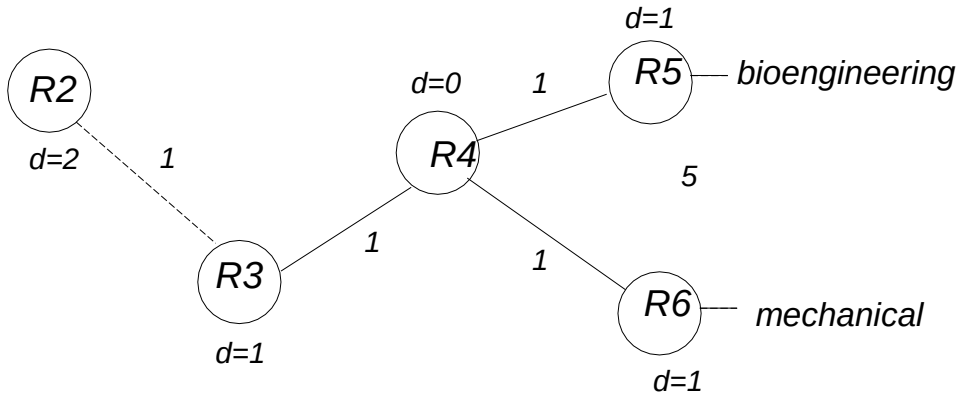


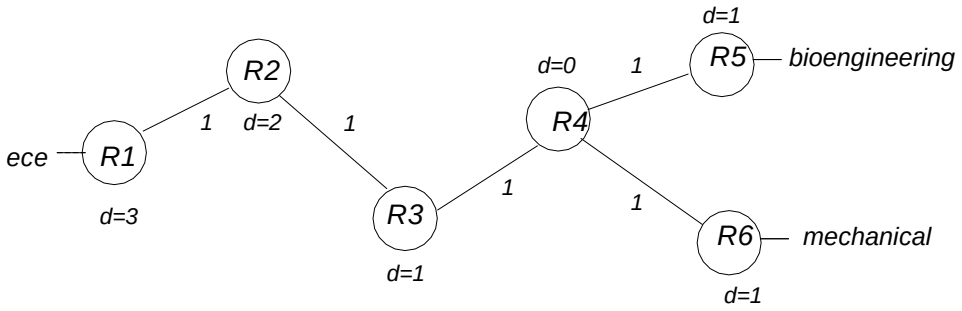
Computing Routes: Dijkstra's Algorithm

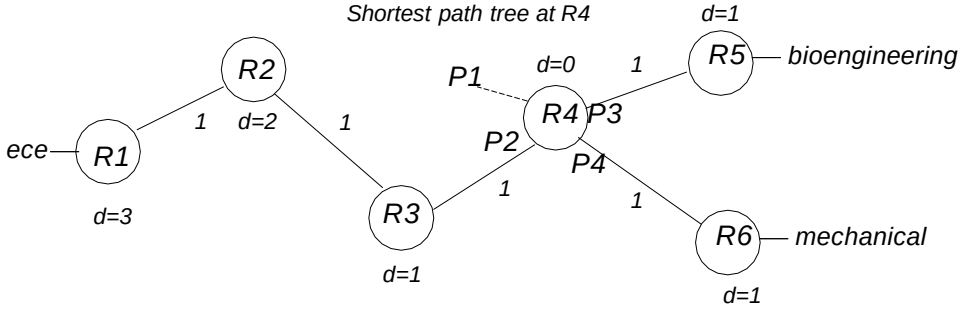




Dijkstra Continued







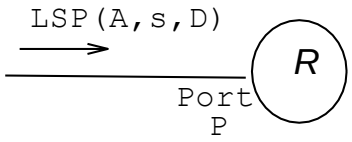
Prefix	Next hop interface
ece	P2
bioengineering	P3
mechanical	P4

Forwarding table at R4

Equal Cost Routes

- Today especially in data centers need a lot of parallelism (multiple 10 or 100 Gbps links to get more bandwidthj)
- To use all bandwidth its very easy to modify Dijkstra to keep track of all equal cost next hops.
- Because TCP does not like reordering within a connection, routers today do ECMP (Equal Cost Multpath) by splitting traffic between equal cost next hops based on a hash function that can has the Dest and Src IP address, and Dst and Src TCP Ports
- This hashing idea guarantees no TCP connection is reordered.

LINK STATE CODE



RECEIVE LSP(A,s,D) on PORT P

```
IF s < SEQ(A) THEN
    SEND STORED-LSP(A) ON PORT P
    ACK([A,P] = TRUE
ELSEIF s > SEQ (A) THEN
   IF A = ME THEN (*source must jump*)
     SEQ(ME) = s + 1;
     SEND STORED-LSP(A) ON ALL PORTS
     FOR ALL PORTS Q,ACK[ME,Q] = TRUE
   ELSE
     STORED-LSP(A) = LSP(A,s, D)
     SEND ACK(A,s) ON PORT P
     SEND STORED-LSP(A) ON ALL PORTS
     P<>Q  FOR ALL PORTS Q<>P,ACK[A,Q] =
     TRUE
ELSE
    SEQ(ME) = SEQ(ME) + 1;
    SEND ACK(A,s) ON PORT P
    ACK([A,P] = FALSE
```

PERIODICALLY

```
FOR ALL PORTS P and A with ACK[A,P]=
   TRUE  SEND STORED-LSP(A) ON PORT P
```

RECEIVE ACK (A, s)

```
IF s = SEQ(A) THEN
  ACK([A,P] = FALSE
```

LINK ON PORT P COMES UP

```
FOR ALL SOURCES A DO
   SET ACK[A,P]= TRUE (*send all LSPs on
   P*)
```

PART 4: LOOKING BACK

Other metrics, and why modern clouds are using SDN

General Principles

- Best effort: no guarantees (TCP will fix if needed)
- Soft state: Not like the hard state of a database that must be correct. We keep retransmitting route updates so if its wrong it will fix itself.
- Decentralized: no central person who knows all routes as in Google Maps. But things are changing

Generalizations

- Both flavors generalize nicely
- Can easily generalize distance vector to find min bandwidth paths or max reliability. Will explore in HW
- So does link state. Once you have the whole network its easy to compute other metrics. One famous example is Constrained Shortest Path for traffic engineering and another is equal cost load splitting. Homework and exam problems!

Amin Vahdat, VP of Google Infrastructure on Google's new approach based on SDN

https://www.youtube.com/watch?v=DpO1Tfa4IZ4, 14:25

Conclusions

- Distance vector still used as part of a protocol called RIP and Cisco’ s IGRP. Probably we use IGRP within UCLA.
- Link state or OSPF is used widely within ISPs because its more reliable after failure and can be easily modified to do traffic engineering based on say bandwidth.
- Next lecture we will see how ISPs use something called policy vector to compute routes between ISPs instead of merely shortest paths
- Note how private clouds like Google are doing their own routing based on centralized views so they can give their applications better guarantees