HW6: Routing

COM-208: Computer Networks

Link-state routing

Basic

Consider the network in Figure 1. Execute the link-state (Dijkstra's) algorithm we saw in class to compute the least-cost path from each of x, v, and t to all the other routers.

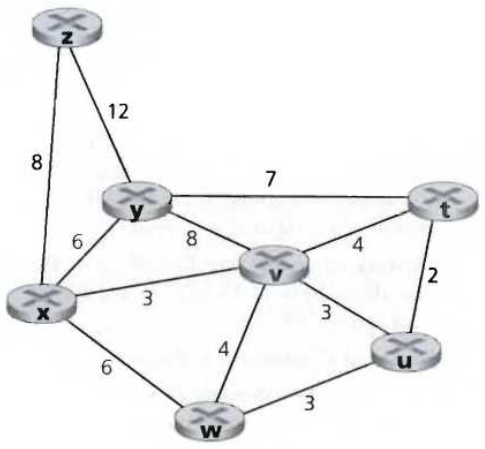


Figure 1: Network topology.

Distance-vector routing

Basic

Consider the network in Figure 2. Execute the distance-vector (Bellman-Ford) algorithm we saw in class and show the information that router z knows after each iteration.

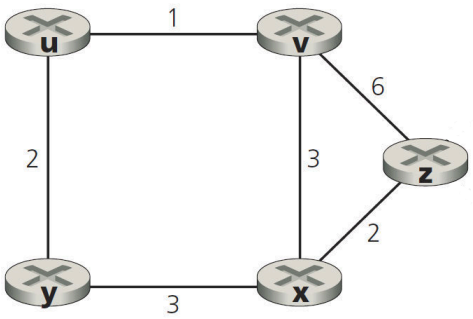


Figure 2: Network topology.

Convergence

Intermediate

What is the maximum number of iterations required for the distance-vector (Bellman-Ford) algorithm that we saw in class to converge (i.e., to finish, assuming no change occurs in the network graph and link costs)? Justify your answer.

Poisoned reverse

Advanced

Consider the network in Figure 3. The routers in the network run the distance-vector (Bellman-Ford) algorithm we saw in class, with poisoned reverse enabled. Suppose the algorithm has run for some time and has converged to the correct least-cost paths.

Table 4 contains the reachability information from each router to router A (e.g., from router D we can reach router A through router C with cost 3).

Now suppose that link $A - B$ goes down.

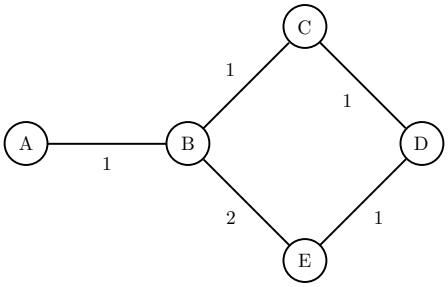


Figure 3: Network topology.

- Describe the next 6 steps of the algorithm. For each step, show the reachability information from each router to router A (cost of the least-cost path and next hop).

		from router B	from router C	from router D	from router E
route to router A	step 0	1 via A	2 via B	3 via C	3 via B
	step 1				
	step 2				
	step 3				
	step 4				
	step 5				
	step 6				

Figure 4: Reachability information from each router to router A.

- Will the algorithm converge to the correct least-cost path values? If yes, in how many steps?
- Propose a simple way to make the algorithm converge faster.

Put forwarding and routing together

Consider the network in Figure 6, consisting of:

- End-systems A, B and X, DNS server C, and web server D.
- IP routers R_1, R_2, R_3, and R_4.
- The link costs are noted in Figure 6.
- End-systems A, B and X use C as their local DNS server.

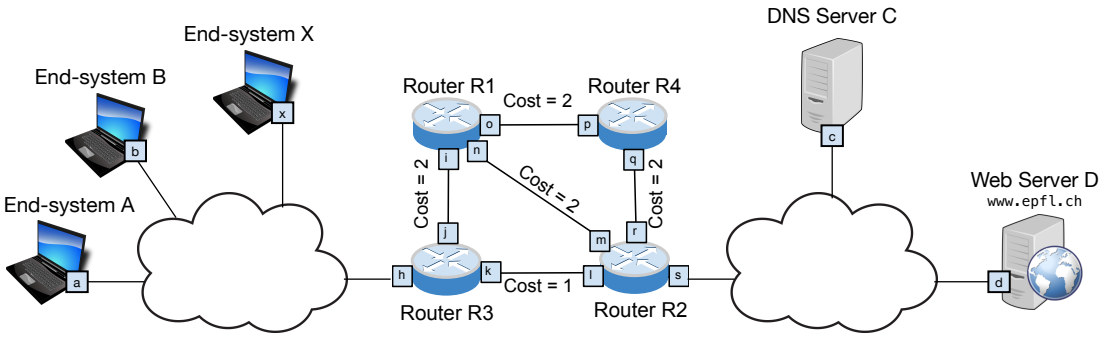


Figure 6: Network topology.

IP prefix/address allocation

Basic

Allocate an IP prefix to each IP subnet and an IP address to each network interface that needs one, following these rules:

- All IP addresses must be allocated from `8.8.8.0/24`.
- Each IP subnet must be allocated the smallest possible IP prefix and must have one broadcast IP address.
- IP router interfaces have IP addresses.

Explain how you compute each IP prefix and fill in Table 7.

Subnet number	IP prefix	Interfaces and IP addresses	Broadcast IP address
Example: 1	10.1.1.0/24	x: 10.1.1.0 y: 10.1.1.1 z: 10.1.1.2	10.1.1.255

Figure 7: Allocation of IP prefixes and IP addresses for the network in Figure 6.

Forwarding tables

Basic

IP routers R_1, R_2, R_3, and R_4 participate in a least-cost path routing algorithm. All the links between the routers are in good condition (no link has failed or is failing), and the algorithm has converged.

Show the forwarding tables of routers R_3 and R_2.

a) Router R_3:

destination IP prefix	output link

b) Router R_2:

destination IP prefix	output link

Routing = populating the forwarding tables

Basic

Routers R_1, R_2, R_3, and R_4 run the Bellman-Ford algorithm.

Show the final state of the Bellman-Ford tables for all the routers, **once the algorithm has converged**.

Answer by filling in the tables below. Here is an example table:

Router X:

from \ to	X	Y	Z
X	0	$5(Y)$	$10(Y)$
Y	5	0	5

$10(Y)$ means that the path from router X to router Z has cost 10 and goes through X's neighbor Y.

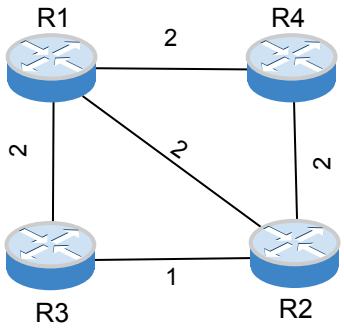


Figure 8: Network topology used in this question.

Router R_1:

from \ to	R_1	R_2	R_3	R_4
R_1				
R_2				
R_3				
R_4				

Router R_2:

from \ to	R_1	R_2	R_3	R_4
R_2				
R_1				
R_3				
R_4				

Router R_3:

from \ to	R_1	R_2	R_3	R_4
R_3				
R_1				
R_2				
R_4				

Router R_4:

from \ to	R_1	R_2	R_3	R_4
R_4				
R_1				
R_2				
R_3				

Routing in the presence of link failures

Advanced

The link between routers R_1 and R_2 fails.

Show the Bellman-Ford tables for all the routers **from the moment the link fails and until the algorithm has re-converged**. Write clearly after how many interactions (neighbor exchanges) the algorithm re-converges.

Answer by filling in the tables below. We have provided tables for two iterations, but the algorithm may reconverge faster (in which case you just leave some tables empty).

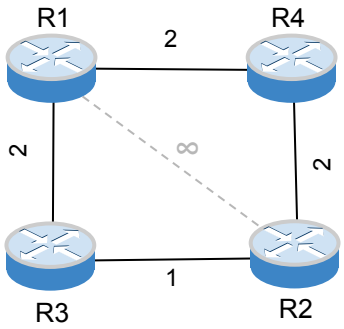


Figure 9: Network topology after the link failure.

State after the link failure:

Router R_1:

from \ to	R_1	R_2	R_3	R_4
R_1				
R_2				
R_3				
R_4				

Router R_2:

from \ to	R_1	R_2	R_3	R_4
R_2				
R_1				
R_3				
R_4				

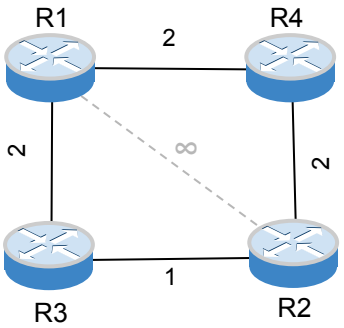
After the first exchange:

Router R_1:

from \ to	R_1	R_2	R_3	R_4
R_1				
R_2				
R_3				
R_4				

Router R_2:

from \ to	R_1	R_2	R_3	R_4
R_2				
R_1				
R_3				
R_4				



Router R_3:

from \ to	R_1	R_2	R_3	R_4
R_3				
R_1				
R_2				
R_4				

Router R_4:

from \ to	R_1	R_2	R_3	R_4
R_4				
R_1				
R_2				
R_3				

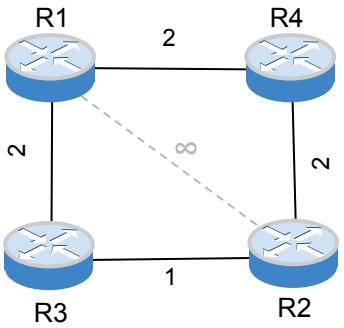
After the second exchange:

Router R_1:

from \ to	R_1	R_2	R_3	R_4
R_1				
R_2				
R_3				
R_4				

Router R_2:

from \ to	R_1	R_2	R_3	R_4
R_2				
R_1				
R_3				
R_4				



Router R_3:

from \ to	R_1	R_2	R_3	R_4
R_3				
R_1				
R_2				
R_4				

Router R_4:

from \ to	R_1	R_2	R_3	R_4
R_4				
R_1				
R_2				
R_3				