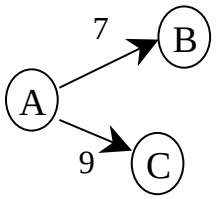
ECE 4380/6380: Computer Communications **Spring 2017**
Problem Set 8 **Due: in class, Wednesday, April 5**

Assigned reading:

- Peterson and Davie, Chapter 3. Section 3.4 is not required.
- We will cover Peterson and Davie, Chapter 4 after the next exam.

1. **Distance-vector routing**. A network has routers A, B, C, D, E, and F. Router A is booted and its routing table is empty. Initially router A is configured with two interfaces but it does not yet know the identities of the routers attached to the interfaces nor the weight that it should assign to the links to the neighbor routers. Note that the weight for a link is not necessarily the same in both directions.
   (a) Router A receives distance vector β from router B, and assigns a weight of 7 to the link from itself to router B. Give A's new routing table. Specify both the out going link to use and the distance from A, for each of the six routers.
   (b) Next, router A receives distance vector χ from router C, and assigns a weight of 9 to the link from itself to router C. Give A's new routing table.



Routing vector β	
Destination	Distance
A	3
B	0
C	7
D	8
E	5
F	7

Routing vector χ	
Destination	Distance
A	8
B	3
C	0
D	8
E	2
F	3

2. **Distance-vector routing.** Consider a network that has nodes designated as A, B, C, D, E, and F. There is a positive integer cost assigned to each link in the network, and do not assume that the cost is the same in each direction on a link. You are able to observe the current state of the forwarding tables at three of the nodes: nodes A, B, and C. The forwarding tables for these nodes are shown below

A's table			B's table			C's table		
Destination	Next Node	Cost	Destination	Next Node	Cost	Destination	Next Node	Cost
A	A	0	A	A	1	A	A	2
B	B	3	B	B	0	B	A	5
C	C	2	C	A	3	C	C	0
D	C	8	D	A	9	D	F	6
E	B	7	E	E	4	E	A	9
F	C	6	F	A	7	F	F	4

Costs for one or more links change in the network and C's forwarding table is updated to the following entries. Nodes A and B have not yet been notified of any of the changes.

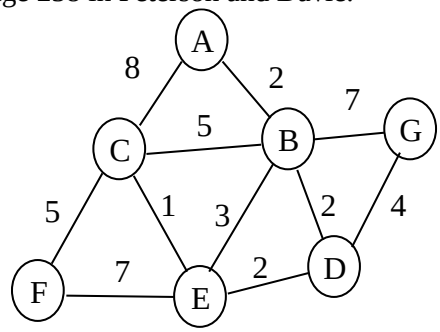
C's table		
Destination	Next Node	Cost
A	A	2
B	B	1
C	C	0
D	B	11
E	B	5
F	F	4

Assume the nodes are using a distributed version of the Bellman-Ford algorithm with split horizon with poison reverse. Nodes A and B have the routing table given at the start of the problem and node C has the updated routing table. Assume that node C is the next node to send distance vectors to its neighbors.

(a) List the distance vectors that C sends to neighbors A and B (use the format shown in problem 1).

(b) Immediately after A and B update their routing tables and before any other actions are taken, is the looping problem to destination D fixed (briefly explain your answer)?

3. **Link-state routing.** Show how the link-state algorithm builds the routing table for node *A* in the following network. Assume that the link weights are the same in both directions on a link. Use the same format as Table 3.14, page 258 in Peterson and Davie.



4. **Link-state routing.** As discussed in class, the link-state routing algorithm does not have the looping problems found with distance-vector routing. However, link-state routing does depend on all nodes having an up-to-date database of link-state packets (LSP's). If there is inconsistent information at different nodes then routing errors may occur. Consider a network that has nodes A, B, C, D, E, F, and G. Each node generates a LSP that contains the following information:
   - The ID of the node that created the LSP
   - A list of directly connected neighbors of that node, with the cost of the link to each one
   - Additional information such as sequence number and time to live

We will ignore the sequence number and time to live information and just focus on the ID and list of neighbors. Assume that each node generates its own LSP as shown below and that each node has an up-to-date copy of all LSP's

Source A	
Nbr	Cost
D	6
F	1

Source B	
Nbr	Cost
D	4
E	3
F	3

Source C	
Nbr	Cost
E	1

F	5
G	1

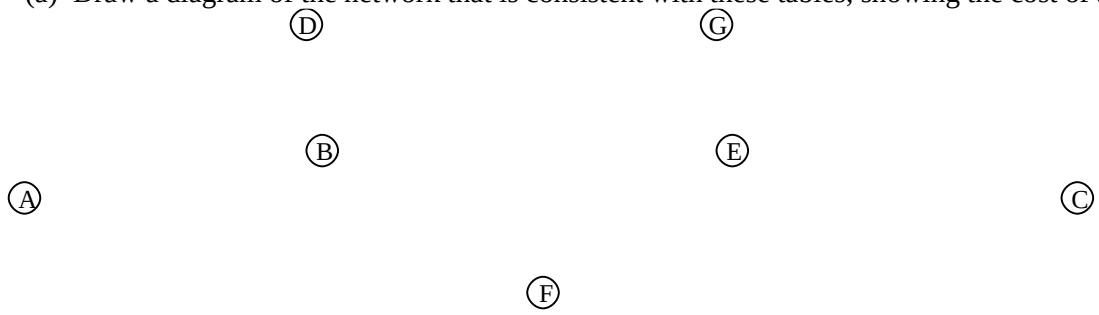
Source D	
Nbr	Cost
A	1
B	5
G	1

Source E	
Nbr	Cost
B	4
C	1
G	1

Source F	
Nbr	Cost
A	2
B	1
C	5

Source G	
Nbr	Cost
C	1
D	6
E	1

(a) Draw a diagram of the network that is consistent with these tables, showing the cost of all links.

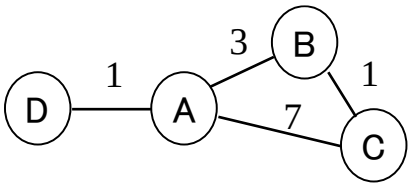


(b) For a packet that is generated at node A with node G as its destination, give the path that the packet follows.

(c) Assume that node B updates the cost of its link to D with a new value of 1. Node B runs Dijkstra's algorithm and updates its forwarding table, but it does not notify any of the other nodes in the network of the change (say for example, due to an error). Does this change the path that a packet from A to G follows? If so, give the new path.

(d) Next, assume that node D updates the cost of its link to G with a new value of 8. Node D updates its forwarding table and forwards its LSP packet to nodes A, F, and G, but because of another error, the LSP packet does not reach nodes B, C, or E. In addition, the change that node B made in part (c) still has not reached any of the other nodes in the network. Does this change the path that a packet from A to G follows? If so, give the new path.

5. Distance-Vector Routing



Consider a network with nodes A, B, C, and D and each node executes a distributed distance-vector protocol that includes triggered updates. That is, if a node calculates a new route to a destination or updates a new cost for an existing route to a destination, then it generates a triggered update. A node stores the most recent distance vector (DV) it has received from each of its neighbors. Assume that neither split horizon nor poison reverse has been implemented. The initial link costs are shown in the figure. Consider an example in which the link cost for the link between A and B decreases from 3 to 1. Below is the sequence of events that occur in response to this change *with respect to routes to node D only*. (We will ignore all other routes for this problem.)

Event number	Action	Response by neighbor
1.	link cost decreases from 3 to 1	Node B had these DVs for destination D *before* change Via A (dest=D, cost = 1) Via C (dest=D, cost = 5) *After* decrease in link cost: Node B's best route to D: next=A, cost =2 (cost reduced from 4 to 2)
2.	Node B generates triggered update with DV (dest=D, cost=2)	Node C had these DVs for D *before* receiving triggered update Via A (dest=D, cost = 1) Via B (dest=D, cost = 4) *After* change to C's DV table: Via B (dest=D, cost = 2) Node C's best route to D: next=B, cost=3 (cost reduced from 5 to 3)
3	Node C generates triggered update with DV (dest=D, cost=3)	Node B had these DVs for D *before* receiving triggered update Via A (dest=D, cost = 1) Via C (dest=D, cost = 5) *After* change to B's DV table: Via C (dest=D, cost = 3). Node B does not change route to D so no triggered update and sequence done

Note that we can ignore any actions at node A and D for this problem. Only the actions taken by nodes B and C are important, and only the triggered updates exchanged between nodes B and C need to be considered.

Next assume that the cost increases on the link between A and B from 1 to 10. Here are the first few steps. Continue filling out the table starting with the response in event 4 and continue until there are no more changes.

Event number	Action	Response by neighbor
1.	link cost increase from 1 to 10	Node B had these DVs for destination D *before* change Via A (dest=D, cost = 1) Via C (dest=D, cost = 3) *After* increase in link cost: Node B's best route to D: next=C, cost =4 (cost increases from 2 to 4)
2.	Node B generates triggered update with DV (dest=D, cost=4)	Node C had these DVs for D *before* receiving triggered update Via A (dest=D, cost = 1) Via B (dest=D, cost = 2) *After* change to C's DV table: Via B (dest=D, cost = 4) Node C's best route to D: next=B, cost=5 (cost increase from 3 to 5)
3	Node C generates triggered update with DV (dest=D, cost=5)	Node B had these DVs for D *before* receiving triggered update Via A (dest=D, cost = 1) Via C (dest=D, cost = 3)

		After change to B's DV table: Via C (dest=D, cost = 5). Node B's best route to D: next=C, cost =6 (cost increase from 4 to 6)
4	Node B generates triggered update with DV (dest=D, cost=6)	Node C had these DVs for D *before* receiving triggered update Via A (dest=D, cost = 1) Via B (dest=D, cost = 4)

Split horizon with poisoned reverse. The routing protocol is updated at all nodes to use the split horizon with poisoned reverse improvement. The same scenario starts from the beginning but now with the new routing protocol. That is, the link between A and B is initially 3 and it decreases to 1. After the change in link cost the network stabilizes with the new DV tables. Next, assume that the cost increases on the link between A and B from 1 to 10. Show the sequence of events that occur.

Event number	Action	Response by neighbor
1.	link cost increase from 1 to 10	Node B had these DVs for destination D *before* change Via A (dest=D, cost = 1) Via C (dest=D, cost = ←*Hint: what goes here?* *After* increase in link cost: Node B's best route to D: next= , cost =

EXTRA CREDIT. This problem is optional, and is worth 10 points of extra credit for this homework set.

6. **Computer program for minimum distance calculation**. Suppose a network has 100 nodes, numbered 0 through 99. Assume that the cost of the one-way link from node *i* to node *j* is

$$d_{i,j} = |i - j| + (i - j + 3)^2 + 5j$$

Write a computer program (in the language of your choice) to find the least cost path from node 0 to node 99 and the cost of the path. You can use either a distance-vector or link-state algorithm to find the solution. The output of your program should give the cost of the minimum distance route and the nodes in the path. For example, here is the output from my program (using a different link cost function):

```
dijkstra% cc linkstate.c -lm
dijkstra% a.out

link-state algorithm:
min cost 3454.000000
Reverse route 99  74  53  36  22  11  5  1  0

distance-vector algorithm:
min cost 3454.000000
```

```
Route   0  1  5  11  22  36  54  75  99
```

(Notice the routes have the same cost, but do not have the same sequence of nodes. There is more than one route with the minimum cost.) I have implemented both a distance-vector and link-state algorithm and you need only do one. Also, note if you implement the link-state algorithm you may find it easier to print the route in the reverse order. Either ordering for the route is fine.

(My former ECE 223 students cannot just turn in the Dijkstra's machine problem from 223. Either implement the Bellman-Ford algorithm (and compare its run time to Dijkstra's), or implement Dijkstra's in a new language.)

Hand in (a) a well-commented copy of your code and (b) a capture of the output of compiling and running your code.