UNIT3 ROUTING

3.1 LEAST COST ROUTING

In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help of forwarding tables. The source host needs no forwarding table because it delivers its packet to the default router in its local network.

Routing a packet from its source to its destination means routing the packet from a source router to a destination router.

Least-Cost Routing

When an internet is modeled as a weighted graph, one of the ways to interpret the best route from the source router to the destination router is to find the least cost between the two.

That is, the source router chooses a route to the destination router in such a way that the total cost for the route is the least cost among all possible routes.

In Figure 3.1.1, the best route between A and E is A-B-E, with the cost of 6.

This means that each router needs to find the least-cost route between itself and all the other routers to be able to route a packet towards the destination.

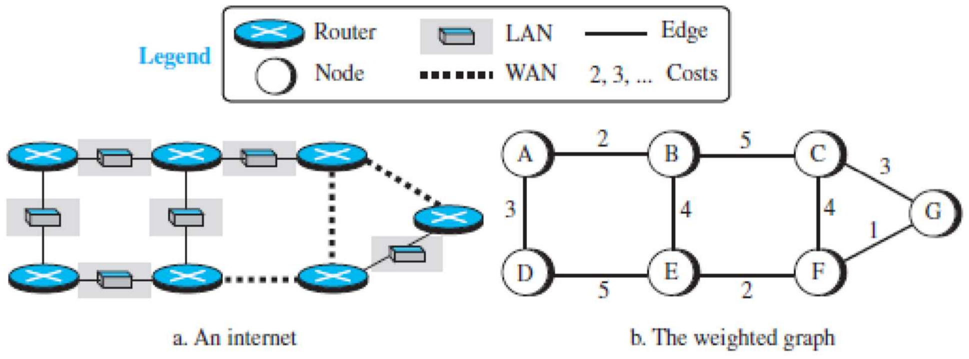


Fig3.1.1:Internet with graph.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-597]

Least-Cost Trees

If there are N routers in an internet, there are (N -1) least-cost paths from each router to any other router.This means we need N (N -1) least-cost paths for the whole internet. For example, If we have only 10 routers in an internet, we need 90 least-cost paths. A least-cost tree is a tree with the source router as the root that spans the whole graph (visits all other nodes) and in which the path between the root and any other node is the shortest.

In this way, we can have only one shortest-path tree for each node; we have N least cost trees for the whole internet. Figure 3.1.2 shows the seven least-cost trees for the internet.

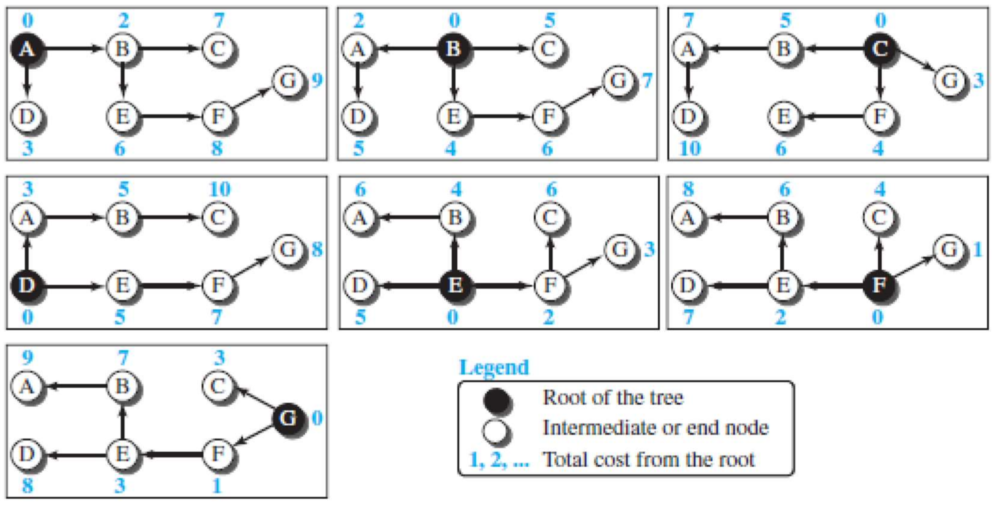


Fig 3.1.2: Seven least-cost trees for the internet.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-597]

1. The least-cost route from X to Y in X's tree is the inverse of the least-cost route from Y to X in Y's tree; the cost in both directions is the same.

   For example, in Figure 3.1.2, the route from A to F in A's tree is (A -B-E-F), but the route from F to A in F's tree is (F -E-B-A), which is the inverse of the first route.The cost is 8 in each case.

2. Instead of travelling from X to Z using X's tree, we can travel from X to Y using X's tree and continue from Y to Z using Y's tree.

   For example, in Figure 3.1.2 , we can go from A to G in A's tree using the route (A B-E -F -G). We can also go from A to E in A's tree (A -B-E) and then continue in E's tree using the route (E -F-G).The combination of the two routes in the second case is the same route as in the first case. The cost in the first case is 9; the cost in the second case is also 9 (6 3).

ROUTING ALGORITHMS

Distance-Vector Routing

In distance-vector routing, a router continuously tells all of its neighbors what it knows about the whole internet.

Bellman-Ford Equation

In distance-vector routing Bellman-Ford equation is used to find the least cost (shortest distance) between a source node, x, and a destination node, y, through some intermediary nodes (a, b, c, …) when the costs between the source and the intermediary nodes and the least costs between the intermediary nodes and the destination are given.

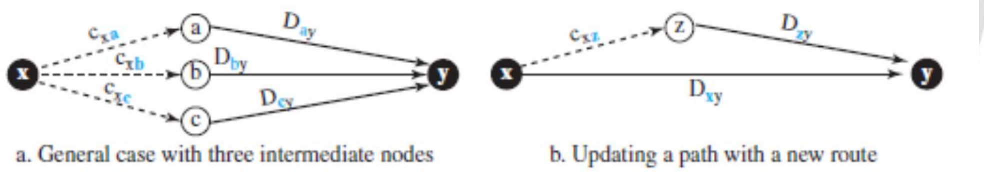
The following shows the general case in which Dij is the shortest distance and cij is the cost between nodes i and j.

$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}), \dots\}$$

In distance-vector routing, we want to update an existing least cost with a least cost through an intermediary node, such as *z*, ie,if the intermediate node is shorter.In this case,the equation can be written as:

$$D_{xy} = \min\{D_{xy}, (c_{xz} + D_{zy})\}$$

Graphical idea behind Bellman-Ford equation



c_{xa}
c_{xb}
c_{xc}
D_{ay}
D_{by}
D_{cy}
x
a
b
c
y
a. General case with three intermediate nodes
c_{xz}
D_{zy}
D_{xy}
z
b. Updating a path with a new route

Fig3.1.3: Graphical idea.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-599]

Bellman-Ford equation help us to build a new least-cost path from previously established least-cost paths.In the Figure 3.1.3, we can think of (*a-y*),(*b-y*), and (*c-y*) as previously established least-cost paths and (*x-y*) as the new least-cost path.

Distance Vectors

The concept of a distance vector is the reason for the name distance-vector routing.A least-cost tree is a combination of least-cost paths from the root of the tree to all destinations.

Figure 3.1.4 shows the tree for node A in the internet and the corresponding distance vector.

A distance vector does not give the path to the destinations as the least-cost tree does; it gives only the least costs to the destinations.

Note that the name of the distance vector defines the root, the indexes define the destinations, and the value of each cell defines the least cost from the root to the destination.

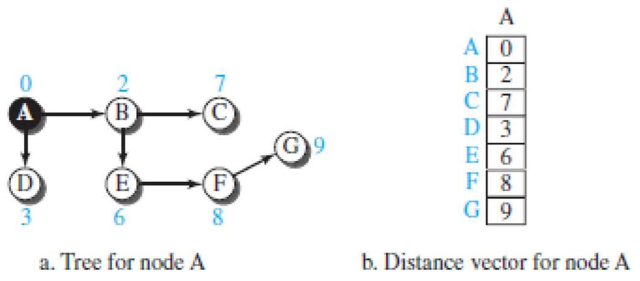


Fig3.1.4: Seven least-cost trees for the internet.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-600]

Each node in an internet, when it starts its function, creates a very basic distance vector with the minimum information the node can obtain from its neighborhood. The node sends some greeting messages out of its interfaces and discovers the identity of the immediate neighbors and the distance between itself and each neighbor.

It then makes a simple distance vector by inserting the discovered distances in the corresponding cells and leaves the value of other cells as infinity.

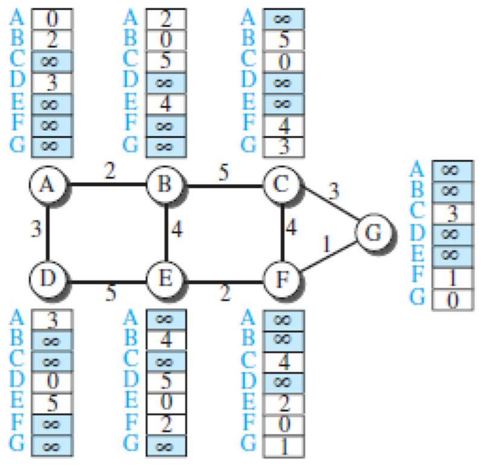


Fig3.1.5: First distance vector for internet.
[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-600]

Description of above diagram

Consider(For example), Node A in figure 3.1.5, thinks that it is not connected to node G because the corresponding cell shows the least cost of infinity.To improve these vectors, the nodes in the internet need to help each other by exchanging information. After each node has created its vector, it sends a copy of the vector to all its immediate neighbors. After a node receives a distance vector from a neighbor, it updates its distance vector using the Bellman-Ford equation (second case).

The figure 3.1.6, shows two asynchronous events, happening one after another with some time in between.In the first event, node A has sent its vector to node B. Node B updates its vector using the cost cBA=2. In the second event, node E has sent its vector to node B.Node B updates its vector using the cost cEA= 4.

After the first event, node B has one improvement in its vector: its least cost to node D has changed from infinity to 5 (via node A). After the second event, node B has one more improvement in its vector; its least cost to node F has changed from infinity to 6 (via node E).

By exchanging the vectors,we can stabilize the system and allows all nodes to find the ultimate least cost between themselves and any other node.After updating a node, it immediately sends its updated vector to all neighbors.

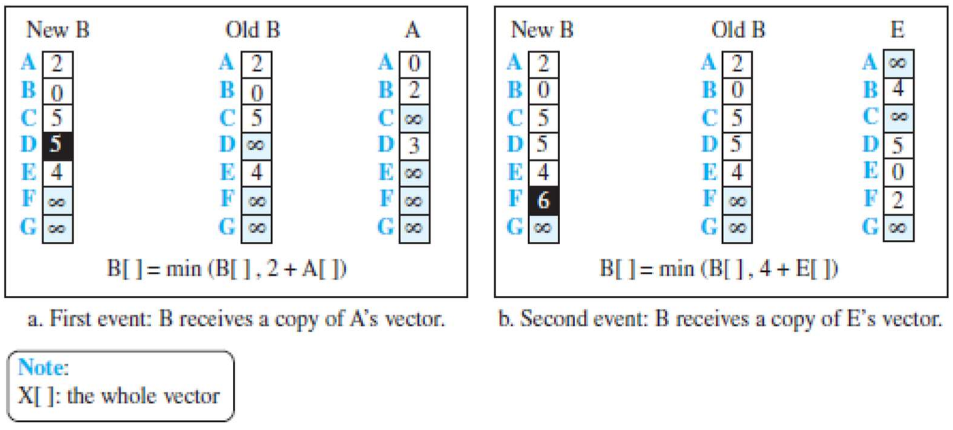

New B
Old B
A
A 2 B 0 C 5 D 5 E 4 F ∞ G ∞
A 2 B 0 C 5 D ∞ E 4 F ∞ G ∞
A 0 B 2 C ∞ D 3 E ∞ F ∞ G ∞
B[ ] = min (B[ ], 2 + A[ ])
a. First event: B receives a copy of A's vector.
New B
Old B
E
A 2 B 0 C 5 D 5 E 4 F 6 G ∞
A 2 B 0 C 5 D 5 E 4 F ∞ G ∞
A ∞ B 4 C ∞ D 5 E 0 F 2 G ∞
B[ ] = min (B[ ], 4 + E[ ])
b. Second event: B receives a copy of E's vector.
Note:
X[ ]: the whole vector

Fig3.1.6:Updating distance vector.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-601]

Count to Infinity

For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance-vector routing, this takes some time. The problem is called count to infinity.

Two-Node Loop

Example of count to infinity is the two-node loop problem.To understand the problem, consider the Figure 3.1.7. The figure shows a system with three nodes. Initially both nodes A and B know how to reach node X. But suddenly, the link between A and X fails.

Node A changes its table. If A can send its table to B immediately, everything is fine. However, the system becomes unstable if B sends its forwarding table to A before receiving A's forwarding table.

Node A receives the update and, assuming that B has found a way to reach X, immediately updates its forwarding table.

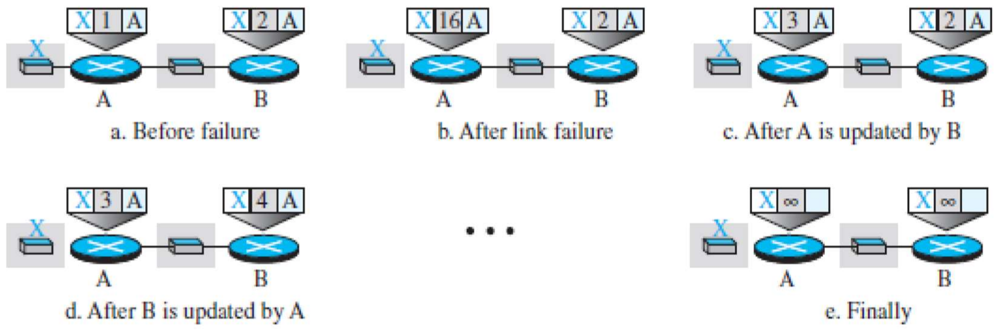


Fig3.1.7:Two node instability.

[Source : "Data Communications and Networking" by Behrouz A. Forouzan,Page-603]

Now A sends its new update to B.Now B thinks that something has been changed around A and updates its forwarding table. The cost of reaching X increases gradually until it reaches infinity. At this moment, both A and B know that X cannot be reached. During this time the system is not stable. Node A thinks that the route to X is via B; node B thinks that the route to X is via A.

If A receives a packet destined for X, the packet goes to B and then comes back to A. Similarly, if B receives a packet destined for X, it goes to A and comes back to B. Packets bounce between A and B, creating a two-node loop problem.

A few solutions have been proposed for instability of this kind.

Split Horizon

One solution to instability is called split horizon. In this method, instead of flooding the table through each interface, each node sends only part of its table through each interface.

If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows). Taking information from node A, modifying it, and sending it back to node A is what creates the confusion. In this method, node B eliminates the last line of its forwarding table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X.

Later, when node A sends its forwarding table to B, node B also corrects its forwarding table. The system becomes stable after the first update: both node A and node B know that X is not reachable.