## Implementation of :

1. Dvr using bellman ford
2. Link state algo using dijkstra’s

## Dvr using bellman ford

### Bellman ford:

* Given a graph and a source vertex src. Find shortest path from src to all other vertices
* The graph may contain negative weight edges
* Time complexity is O(VE)

**ALGO:**

**Input:** Graph and a source vertex src

**Output:** Shortest distance from src to all other vertices, report if there is a negative cycle

Make a list of edges

1. Initialise the dist as inf

* Create an array dist[V] -> all values to inf

1. Do the following |V-1| times

* Do the following for each edge u-v

1. If dist[v] > dist[u] + weight of uv, then update dist[v]

* Dist[v] = dist[u] + wright of uv

3. This step is to find a negative cycle:

- If dist[v] > dist[u] + weight of edge uv, then “Graph contains negative weight cycle”

The algo finds it in a bottom up manner

First calculate the shortest distance which have at most 1 edge in a path

Then 2

Then so on

Ith iteration - shortest path with atmost i edges - |V-1| are the max number of edges. Therefore go that many times

**Relaxation:**

If dist[v] > dist[u] + weight of uv, then update dist[v]

Dist[v] = dist[u] + wright of uv

## DVR - DISTANCE VECTOR ROUTING (DVR) - PROTOCOL

A distance-vector routing (DVR) protocol requires that a router inform its neighbors of topology changes periodically. Historically known as the old ARPANET routing algorithm (or known as Bellman-Ford algorithm).

Uses UDP packets

Bellman Ford Basics – Each router maintains a Distance Vector table containing the distance between itself and ALL possible destination nodes.

Distances,based on a chosen metric, are computed using information from the neighbors’ distance vectors.

Information kept by DV router -

1. Each router has an ID
2. Associated with each link connected to a router, there is a link cost (static or dynamic).
3. Intermediate ho**p**s

Basically - dest, dist, and next

Distance Vector Table Initialization -

1. Distance to itself = 0
2. Distance to ALL other routers = infinity number.

Assume:

1. Every router knows about the number of routers
2. And it knows about the neighbors too

What do you share?

1. Share only to neighbors
2. Only distance vectors. Not the whole table

**ALGORITHM:**

1. A router transmits its distance vector to each of its neighbors in a routing packet
2. Each router receives and saves the most recently received distance vector from each of its neighbors
3. A router recalculates its distance vector when
4. It receives a distance vector from a neighbor containing different information than before
5. It discovers that a link to a neighbor has gone down.

Dx(y) = Estimate of least cost from x to y

C(x,v) = Node x knows cost to each neighbor v

Dx = [Dx(y): y ∈ N ] = Node x maintains distance vector

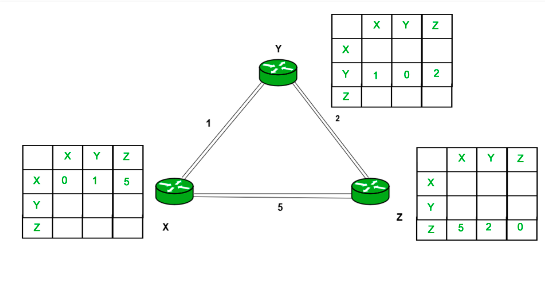
Node x also maintains its neighbors' distance vectors

– For each neighbor v, x maintains Dv = [Dv(y): y ∈ N ]

The DV calculation is based on minimizing the cost to each destination

1. From time-to-time, each node sends its own distance vector estimate to neighbors.
2. When a node x receives new DV estimate from any neighbor v, it saves v’s distance vector and it updates its own DV using B-F equation:

**Dx(y) = min { C(x,v) + Dv(y), Dx(y) } for each node y ∈ N**



1. Ex : At N1, i got N2

Don't use the old routing table when we get n2

N1->N3 = N1 -> N2 + N2 -> N3

Next: n2, n3

If there are >= 2 possibilities, take the best of them

The three keys to understand the Link State Routing algorithm:

1. Knowledge about the neighborhood: Instead of sending its routing table, a router sends the information about its neighborhood only. A router broadcast its identities and cost of the directly attached links to other routers.
2. Flooding: Each router sends the information to every other router on the internetwork except its neighbors. This process is known as Flooding. **Every router that receives the packet sends the copies to all its neighbors.** Finally, each and every router receives a copy of the same information.
3. Information sharing: A router sends the information to every other router only when the change occurs in the information.

Uses ospf

Reliable Flooding

1. Initial state: Each node knows the cost of its neighbors.
2. Final state: Each node knows the entire graph.

Each node uses Dijkstra's algorithm on the graph to calculate the optimal routes to all nodes.

The Link state routing algorithm is also known as Dijkstra's algorithm which is used to find the shortest path from one node to every other node in the network.

The Dijkstra's algorithm is iterative, and it has the property that after kth iteration of the algorithm, the least cost paths are well known for k destination nodes.

**Initialization**

N = {A} // **A is a root node**.

for all nodes v

if v adjacent to A

then D(v) = c(A,v)

else D(v) = infinity

**loop**

find w not in N such that D(w) is a minimum.

Add w to N

Update D(v) for all v adjacent to w and not in N:

D(v) = min(D(v) , D(w) + c(w,v))

Until all nodes in N

Bellman Ford’s algorithm

Like other Dynamic Programming Problems, the algorithm calculates shortest paths in a bottom-up manner. It first calculates the shortest distances which have at-most one edge in the path. Then, it calculates the shortest paths with at-most 2 edges, and so on. After the i-th iteration of outer loop, the shortest paths with at most i edges are calculated. There can be maximum |V| – 1 edge in any simple path, that is why the outer loop runs |v| – 1 time. The idea is, assuming that there is no negative weight cycle if we have calculated shortest paths with at most i edges, then an iteration over all edges guarantees to give the shortest path with at-most (i+1) edges

Dijkstra’s algorithm

Dijkstra’s algorithm is very similar to Prim’s algorithm for minimum spanning tree. Like Prim’s MST, we generate an SPT (shortest path tree) with a given source as root. We maintain two sets, one set contains vertices included in the shortest-path tree, the other set includes vertices not yet included in the shortest-path tree. At every step of the algorithm, we find a vertex which is in the other set (set of not yet included) and has a minimum distance from the source.

A routing loop is a serious network problem which happens when a data packet is continually routed through the same routers over and over. The data packets continue to be routed within the network in an endless circle. A routing loop can have a catastrophic impact on a network, and in some cases, completely disabling the network. Normally Routing Loop is a problem associated with Distance Vector Protocols.

How routing loops affect network performance?

• A major portion of the precious bandwidth which is available for normal user traffic of the affected routers will be consumed by looping IP datagram packets.

• The major portion of the processing power of the affected routers is used to process the looping IP datagram packets.

Routing Loop can happen in large internetworks when a second topology change emerges before the network is able to converge on the first change. Convergence is the term used to describe the condition when all routers in an internetwork have agreed on a common topology. Link state protocols tend to converge very quickly, while distance vector protocols tend to converge slowly.

The following methods are used to avoid Routing Loops.

Maximum hop Count

Maximum hop count mechanism can be used to prevent Routing Loops. Distance Vector protocols use the TTL (Time-to-Live) value in the IP datagram header to avoid Routing Loops. When an IP datagram move from router to router, a router keeps track of the hops in the TTL field in the IP datagram header. For each hop a packet goes through, the packet’s TTL field is decremented by one. If this value reaches 0, the packet is dropped by the router that decremented the value from 1 to 0.

Split Horizon

A split horizon is a routing configuration that stops a route from being advertised back in the direction from which it came. Split Horizon mechanism states that if a neighbouring router sends a route to a router, the receiving router will not propagate this route back to the advertising router on the same interface.

Route Poisoning

Route Poisoning is another method for avoiding routing loops. When a router detects that one of its connected routes has failed, the router will poison the route by assigning an infinite metric to it.

Hold-down Timers

Hold-down timer is another mechanism used to prevent bad routes from being restored and propagated by mistake. When a route is placed in a hold-down state, routers will neither advertise the route nor accept advertisements about it for a specific interval called the hold-down period.