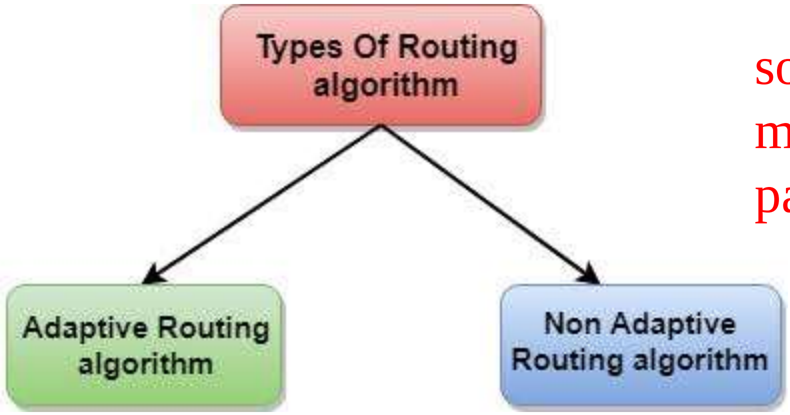
Routing algorithm



In order to transfer the packets from source to the destination, the network layer must determine the best route through which packets can be transmitted.

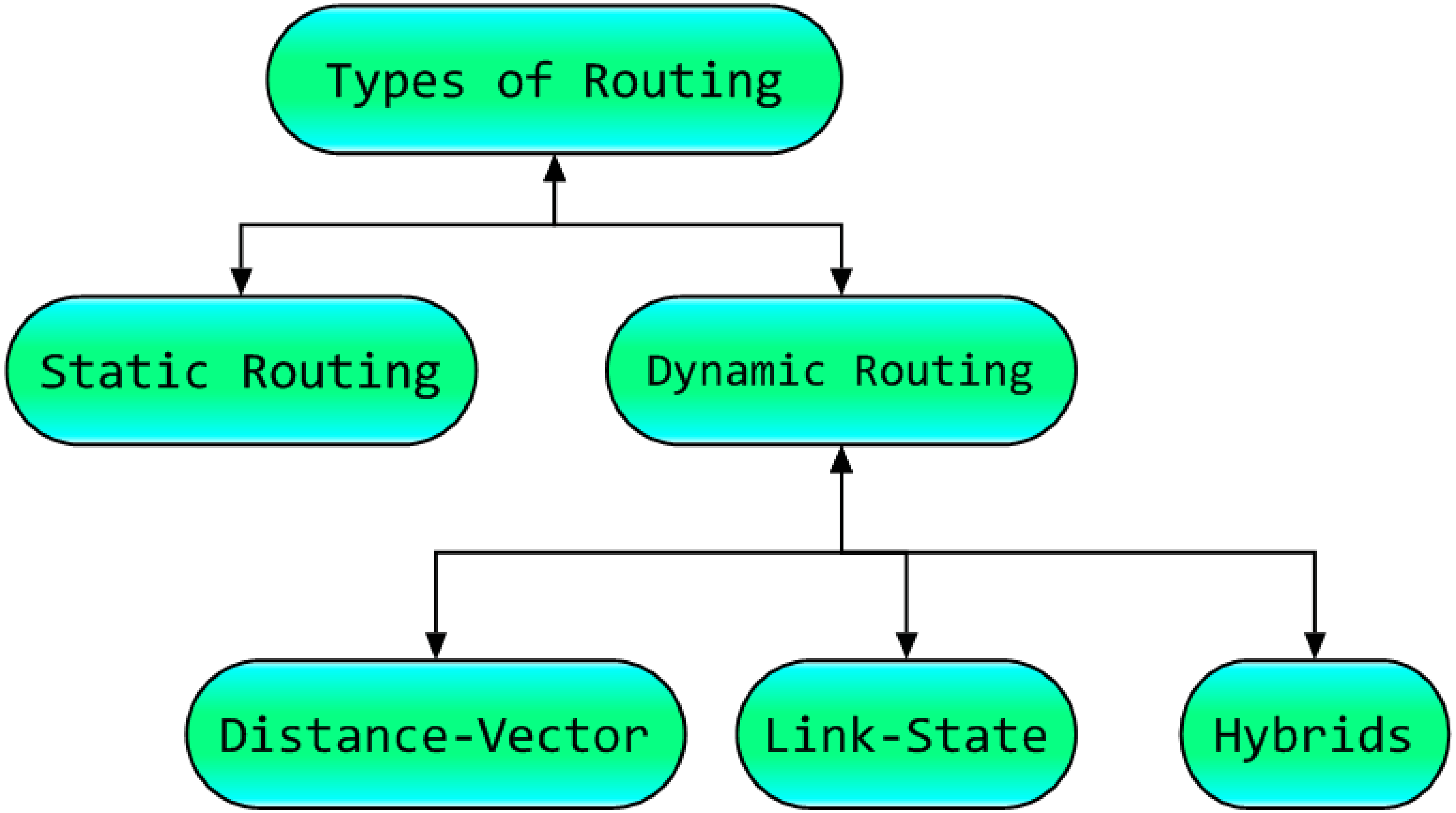
Adaptive Routing algorithm

1. Also known as dynamic routing algorithm.
2. Makes the routing decisions based on the topology and network traffic.
3. Parameters: hop count, distance and estimated transit time.

Non-Adaptive Routing algorithm

1. Also known as a static routing algorithm.
2. The routing information stores to the routers.
3. Do not take the routing decision based on the network topology or network traffic.

Types of Routing Algorithm



Distance Vector Routing Algorithm

The Distance vector algorithm is iterative, asynchronous and distributed.

Distributed: Each node receives information from one or more of its directly attached neighbors, performs calculation and then distributes the result back to its neighbors.

Iterative: It is iterative in that its process continues until no more information is available to be exchanged between neighbors.

Asynchronous: It does not require that all of its nodes operate in the lock step with each other.

The Distance vector algorithm is a dynamic algorithm.

It is mainly used in ARPANET, and RIP.

Each router maintains a distance table known as **Vector**.

Distance Vector Routing Algorithm

Three Keys to understand the working of Distance Vector Routing Algorithm:

1. **Knowledge about the whole network:** Each router shares its knowledge through the entire network.
2. **Routing only to neighbors:** The router sends its knowledge about the network to only those routers which have direct links.
3. **Information sharing at regular intervals:** Within 30 seconds, the router sends the information to the neighboring routers.

Distance Vector Routing Algorithm

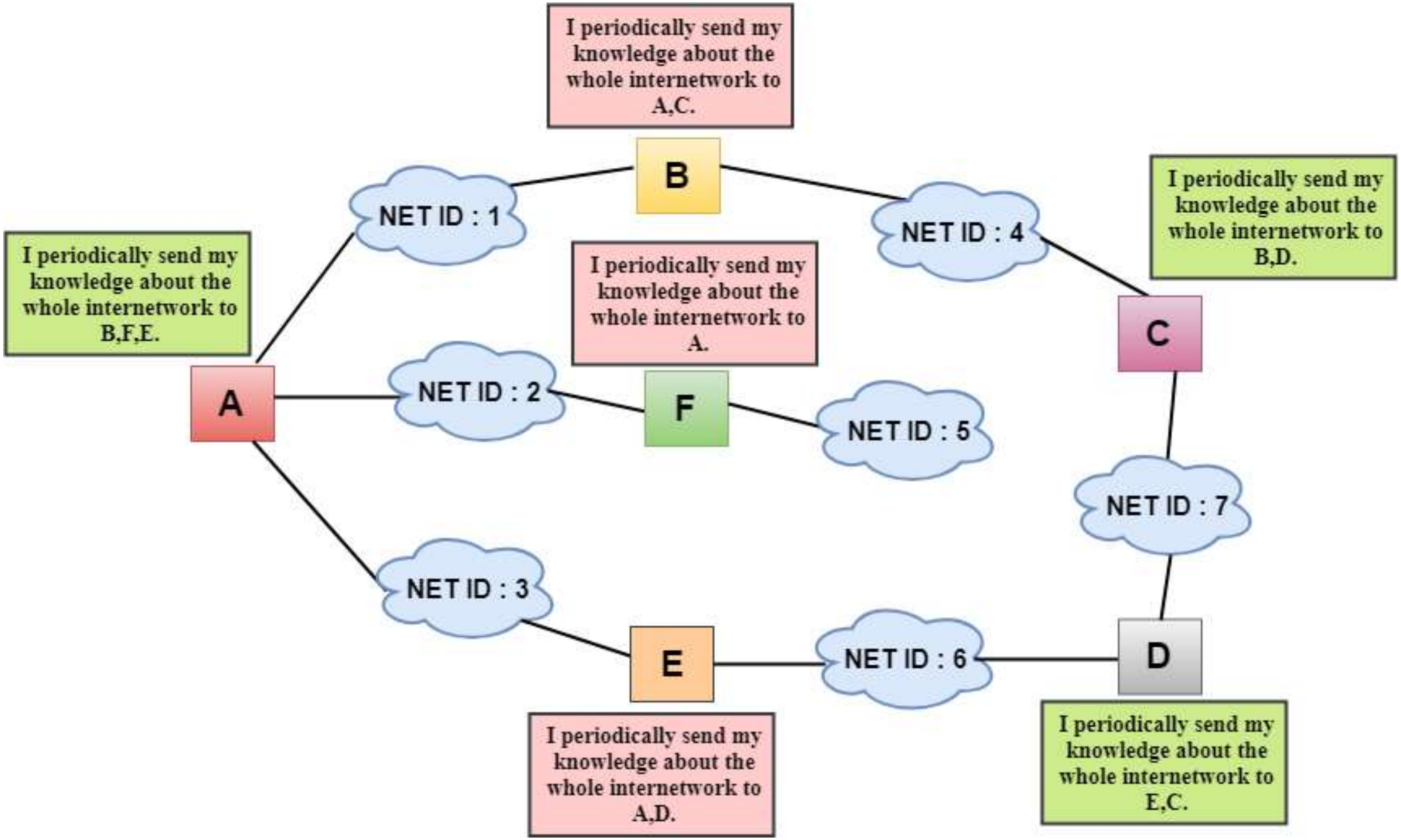
Let $d_x(y)$ be the cost of the least-cost path from node x to node y.

The least costs are related by Bellman-Ford equation,

$$d_x(y) = \min_v\{c(x,v) + d_v(y)\}$$

1. For each neighbor v, the cost c(x,v) is the path cost from x to directly attached neighbor, v.
2. The distance vector x, i.e., D_x = [$D_x(y)$: y in N], containing its cost to all destinations, y, in N.
3. The distance vector of each of its neighbors, i.e., D_v = [$D_v(y)$: y in N] for each neighbor v of x.

Distance Vector Routing Algorithm



Distance Vector Routing Algorithm

Routing Table

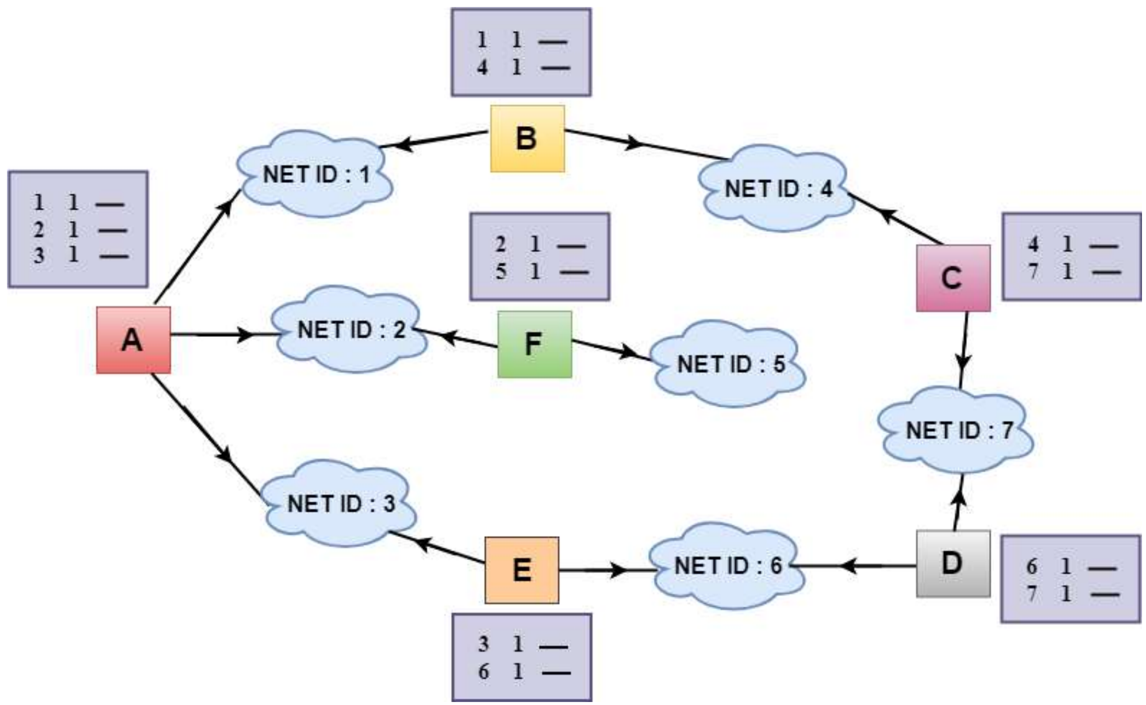
Two process occurs:

1. Creating the Table

NET ID	Cost	Next Hop
- - - - -	- - - -	- - - - -
- - - - -	- - - -	- - - - -
- - - - -	- - - -	- - - - -
- - - - -	- - - -	- - - - -
- - - - -	- - - -	- - - - -

2. Updating the Table

1. **NET ID:** The Network ID defines the final destination of the packet.
2. **Cost:** The cost is the number of hops that packet must take to get there.
3. **Next hop:** It is the router to which the packet must be delivered.

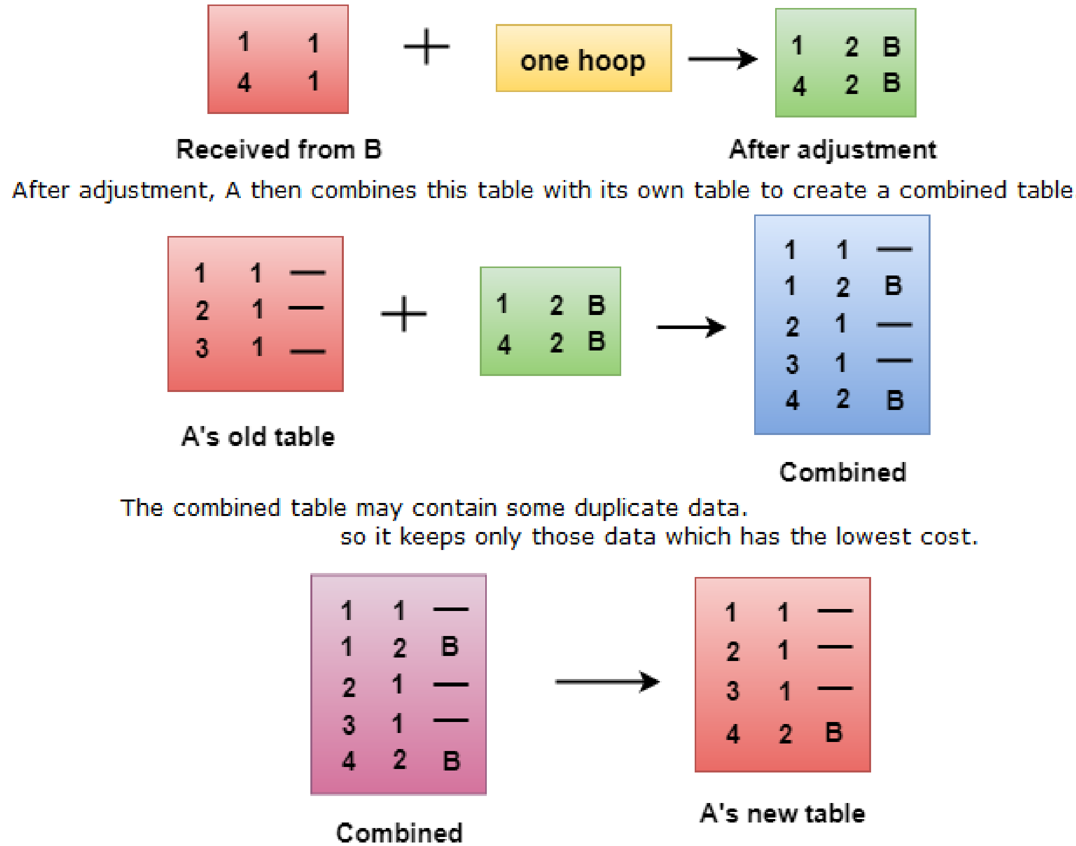


Distance Vector Routing Algorithm

Routing Table

Updating the Table

1. When A receives a routing table from B, then it uses its information to update the table.
2. The routing table of B shows how the packets can move to the networks 1 and 4.
3. The B is a neighbor to the A router, the packets from A to B can reach in one hop. So, 1 is added to all the costs given in the B's table and the sum will be the cost to reach a particular network.



Final routing tables of all the routers are given below:

Router A

6	2	E
1	1	–
3	1	–
4	2	B
7	3	E
2	1	–
5	2	F

Router B

6	3	E
1	1	–
3	2	A
4	1	–
7	2	C
2	2	A
5	3	A

Router C

6	2	D
1	2	B
3	3	D
4	1	–
7	1	–
2	3	B
5	4	B

Router D

6	1	–
1	3	E
3	2	E
4	2	C
7	1	–
2	3	E
5	4	E

Router E

6	1	–
1	2	A
3	1	–
4	3	A
7	2	D
2	2	A
5	3	A

Router F

6	3	A
1	2	A
3	2	A
4	3	A
7	4	A
2	1	–
5	1	–

Link State Routing Algorithm

Link state routing is a technique in which each router shares the knowledge of its neighborhood with every other router in the internetwork.

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.
3. **Information sharing:** A router sends the information to every other router only when the change occurs in the information.

Link State Routing Algorithm

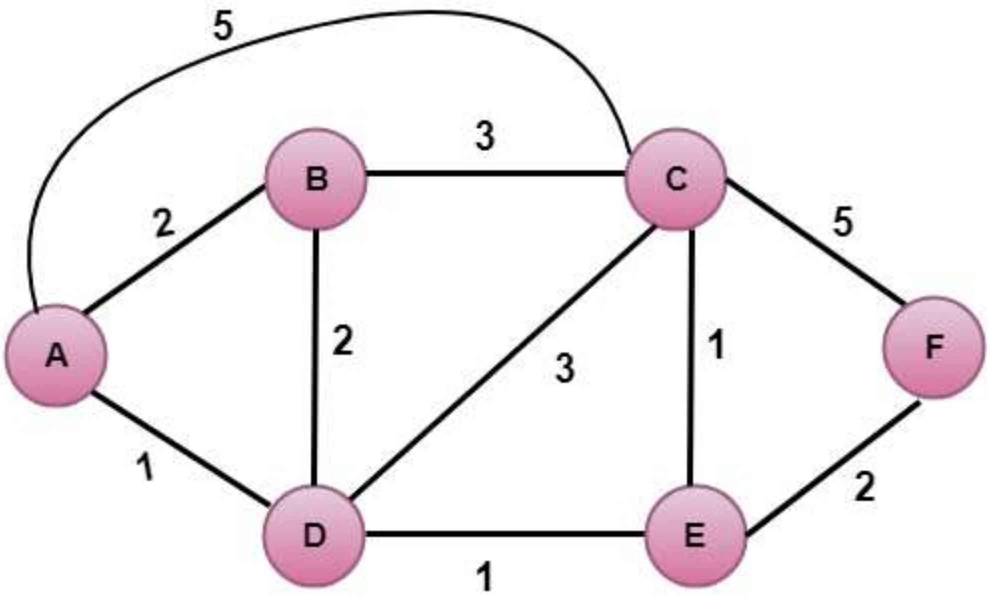
Link State Routing has two phases:

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

Link State Routing Algorithm: Notations

1. **c(i , j):** Link cost from node i to node j. If i and j nodes are not directly linked, then c(i , j) = ∞.
2. **D(v):** It defines the cost of the path from source code to destination v that has the least cost currently.
3. **P(v):** It defines the previous node (neighbor of v) along with current least cost path from source to v.
4. **N:** It is the total number of nodes available in the network.

Link State Routing Algorithm: Notations



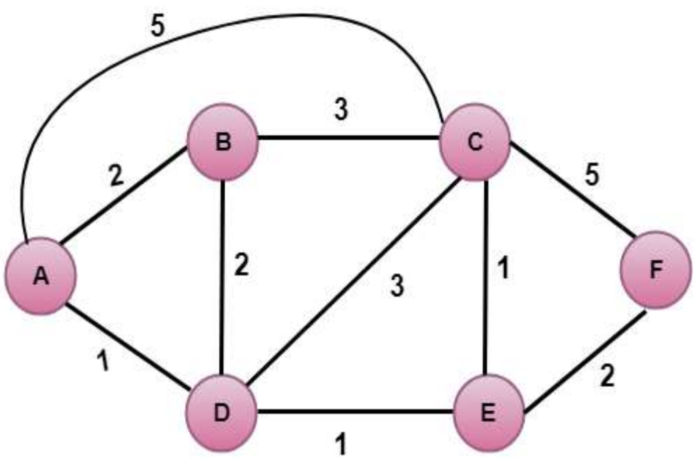
least cost path from A to its directly attached neighbors, B, C, D are 2,5,1

The cost from A to B is set to 2, from A to D is set to 1 and from A to C is set to 5.

The cost from A to E and F are set to infinity as they are not directly linked to A.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞

Link State Routing Algorithm: Notations



a) Calculating shortest path from A to B

v = B, w = D

D(B) = min(D(B) , D(D) + c(D,B))

= min(2, 1+2)>

= min(2, 3)

The minimum value is 2. Therefore, the currently shortest path from A to B is 2.

b) Calculating shortest path from A to C

v = C, w = D

D(B) = min(D(C) , D(D) + c(D,C))

= min(5, 1+3)

= min(5, 4)

The minimum value is 4. Therefore, the currently shortest path from A to C is 4.</p>

c) Calculating shortest path from A to E

v = E, w = D

D(B) = min(D(E) , D(D) + c(D,E))

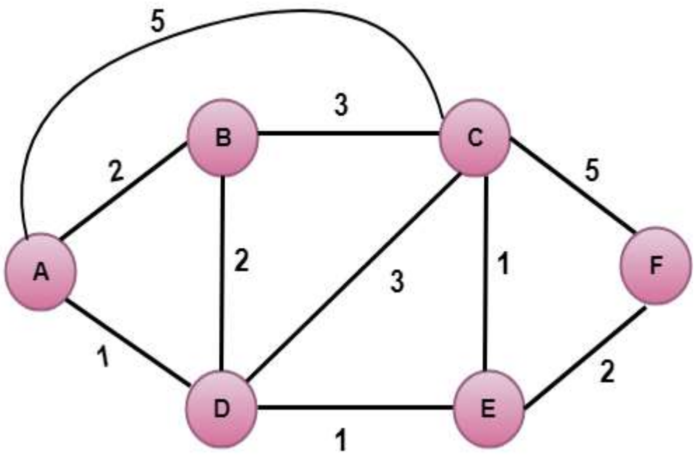
= min(∞, 1+1)

= min(∞, 2)

The minimum value is 2. Therefore, the currently shortest path from A to E is 2.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞

Link State Routing Algorithm: Notations



a) Calculating the shortest path from A to C.

v = C, w = B

D(B) = min(D(C) , D(B) + c(B,C))

= min(3 , 2+3)

= min(3,5)

The minimum value is 3. Therefore, the currently shortest path from A to C is 3.

b) Calculating the shortest path from A to F.

v = F, w = B

D(B) = min(D(F) , D(B) + c(B,F))

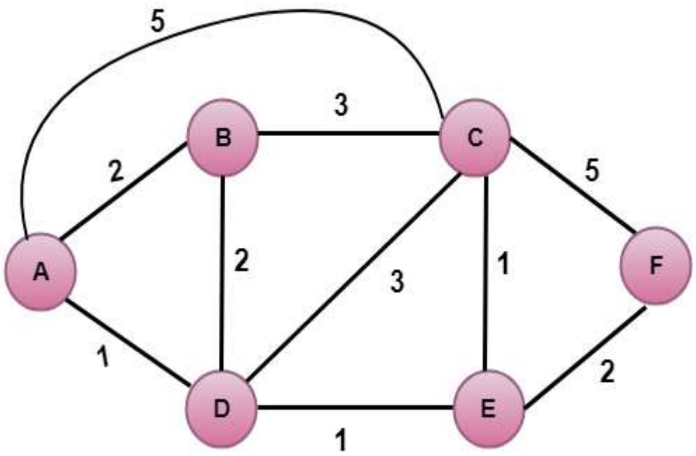
= min(4, ∞)

= min(4, ∞)

The minimum value is 4. Therefore, the currently shortest path from A to F is 4.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E

Link State Routing Algorithm: Notations



a) Calculating the shortest path from A to F.

v = F, w = C

D(B) = min(D(F) , D(C) + c(C,F))

= min(4, 3+5)

= min(4,8)

The minimum value is 4. Therefore, the currently shortest path from A to F is 4.

Link State Routing Algorithm: Notations

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E

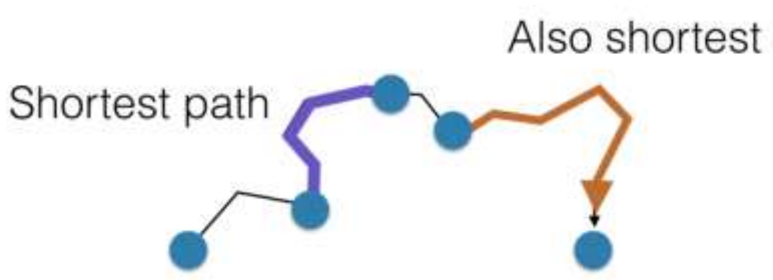
Final table:

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E
6	ADEBCF					

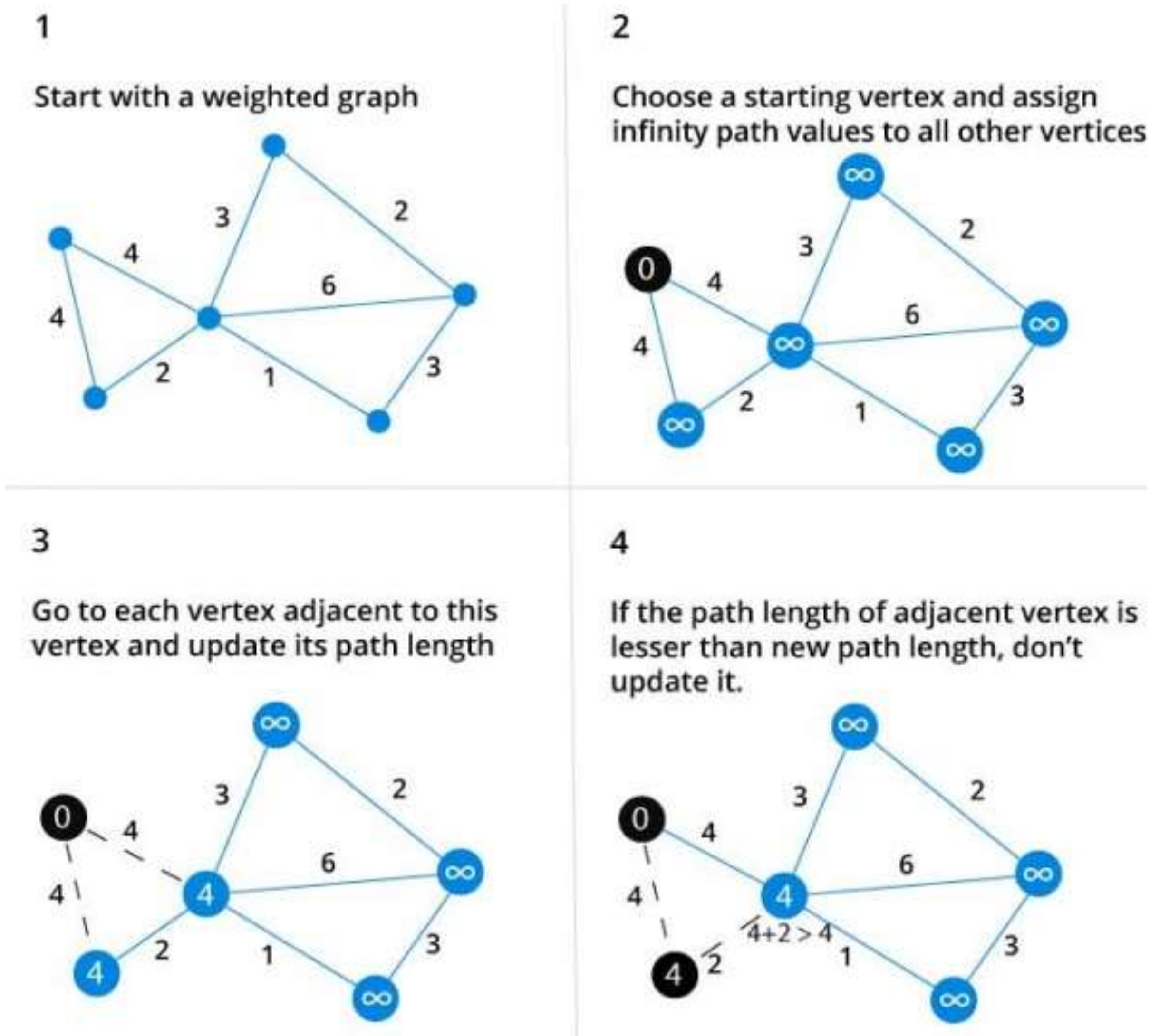
Dijkstra Algorithm

Dijkstra's algorithm allows us to find the shortest path between any two vertices of a graph.

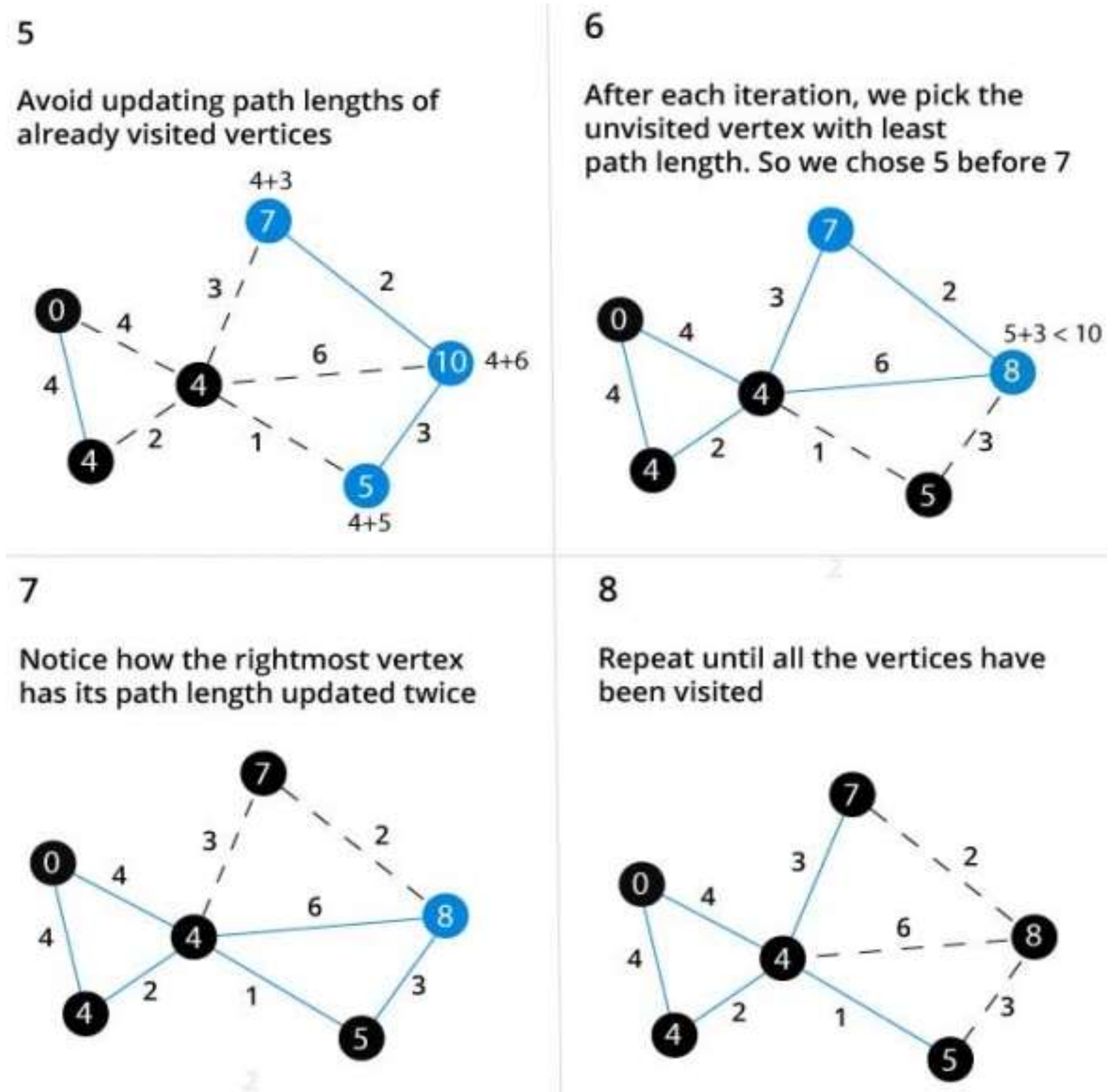
It differs from minimum spanning tree because the shortest distance between two vertices might not include all the vertices of the graph.



Dijkstra Algorithm



Dijkstra Algorithm



Dijkstra Algorithm