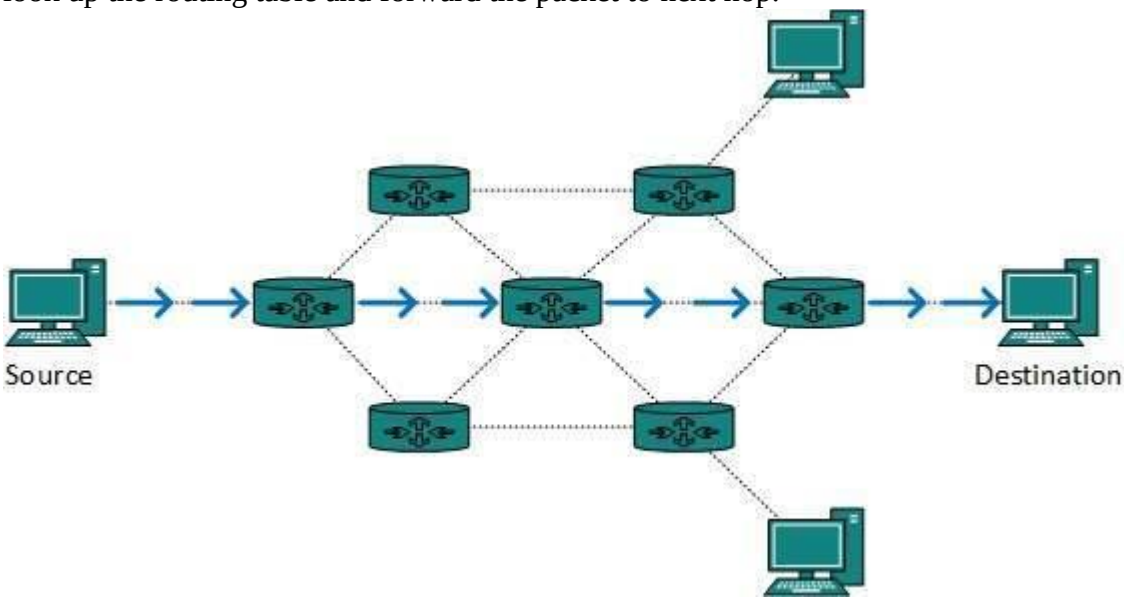
Unicast routing

Most of the traffic on the internet and intranets known as unicast data or unicast traffic is sent with specified destination. Routing unicast data over the internet is called unicast routing. It is the simplest form of routing because the destination is already known. Hence the router just has to look up the routing table and forward the packet to next hop.

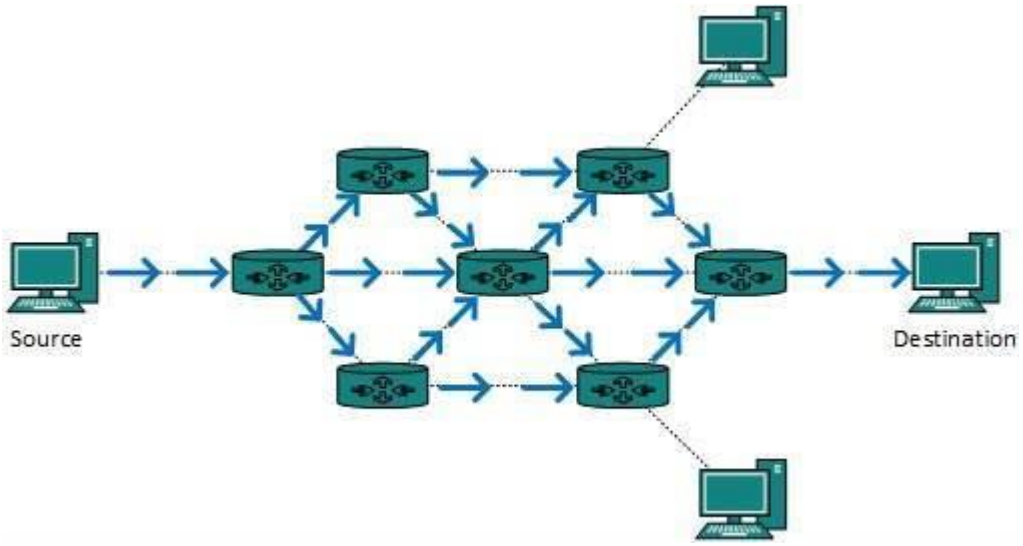


Broadcast routing

By default, the broadcast packets are not routed and forwarded by the routers on any network. Routers create broadcast domains. But it can be configured to forward broadcasts in some special cases. A broadcast message is destined to all network devices.

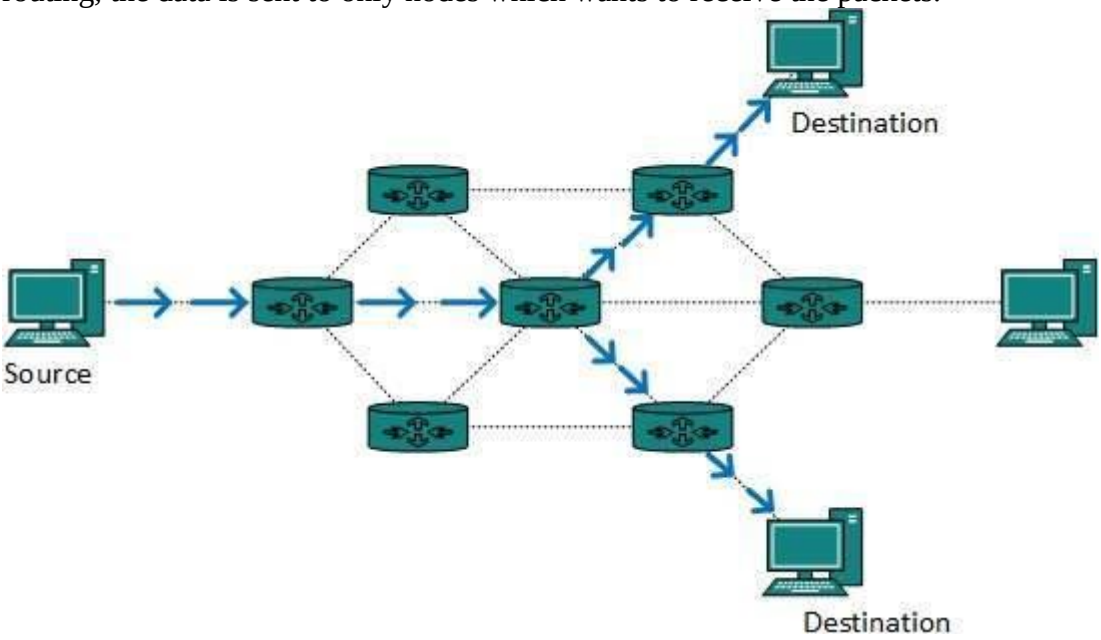
Broadcast routing can be done in two ways (algorithm):

- A router creates a data packet and then sends it to each host one by one. In this case, the router creates multiple copies of single data packet with different destination addresses. All packets are sent as unicast but because they are sent to all, it simulates as if router is broadcasting. This method consumes lots of bandwidth and router must destination address of each node.
- Secondly, when router receives a packet that is to be broadcasted, it simply floods those packets out of all interfaces. All routers are configured in the same way.
- This method is easy on router's CPU but may cause the problem of duplicate packets received from peer routers.
- Reverse path forwarding is a technique, in which router knows in advance about its predecessor from where it should receive broadcast. This technique is used to detect and discard duplicates.



Multicast Routing

Multicast routing is special case of broadcast routing with significance difference and challenges. In broadcast routing, packets are sent to all nodes even if they do not want it. But in Multicast routing, the data is sent to only nodes which wants to receive the packets.



The router must know that there are nodes, which wish to receive multicast packets (or stream) then only it should forward. Multicast routing works spanning tree protocol to avoid looping.

Multicast routing also uses reverse path Forwarding technique, to detect and discard duplicates and loops.

UNICAST ROUTING PROTOCOLS

There are two kinds of routing protocols available to route unicast packets:

Distance Vector Routing Protocol

Distance Vector is simple routing protocol which takes routing decision on the number of hops between source and destination. A route with less number of hops is considered as the best route. Every router advertises its set best routes to other routers. Ultimately, all routers build up their network topology based on the advertisements of their peer routers,

For example Routing Information Protocol (RIP).

Routing Information Protocol (RIP) is a dynamic routing protocol which uses hop count as a routing metric to find the best path between the source and the destination network. It is a distance vector routing protocol which works on the application layer of OSI model. RIP uses port number 520.

Hop Count:

Hop count is the number of routers occurring in between the source and destination network. The path with the lowest hop count is considered as the best route to reach a network and therefore placed in the routing table. RIP prevents routing loops by limiting the number of hopes allowed in a path from source and destination. The maximum hop count allowed for RIP is 15 and hop count of 16 is considered as network unreachable.

Features of RIP:

1. Updates of the network are exchanged periodically.
2. Updates (routing information) are always broadcast
3. Full routing tables are sent in updates.
4. Routers always trust on routing information received from neighbor routers. This is also known as *Routing on rumours*.

Link State Routing Protocol

Link State protocol is slightly complicated protocol than Distance Vector. It takes into account the states of links of all the routers in a network. This technique helps routes build a common graph of the entire network. All routers then calculate their best path for routing purposes.for example,

Open Shortest Path First (OSPF) and Intermediate System to Intermediate System (ISIS).

OSPF stands for Open Shortest Path First which uses link-state routing algorithm. Using the link state information which is available in routers, it constructs the topology in which the topology determines the routing table for routing decisions. It supports both variable-length subnet masking and classless inter-domain routing addressing models. Since it uses Dijkstra's algorithm, it computes the shortest path tree for each route. The main advantages of the OSPF (Open Shortest Path first) is that it handles the error detection by itself and it uses multicast addressing for routing in a broadcast domain.

Advantages of Distance Vector routing –

- It is simpler to configure and maintain than link state routing.

Disadvantages of Distance Vector routing –

- It is slower to converge than link state
- It creates more traffic than link state since a hop count change must be propagated to all routers and processed on each router. Hop count updates take place on a periodic basis, even if there are no changes in the network topology, so bandwidth-wasting broadcasts still occur.
- For larger networks, distance vector routing results in larger routing tables than link state since each router must know about all other routers. This can also lead to congestion on WAN links.

Note – Distance Vector routing uses UDP (User datagram protocol) for transportation.

Difference between RIP and OSPF

SR.NO	RIP	OSPF
1	RIP Stands for Routing Information Protocol.	OSPF stands for Open Shortest Path First.
2	RIP works on Bellman Ford algorithm.	OSPF works on Dijkstra algorithm.
3	It is a Distance Vector protocol and it uses the distance or hops count to determine the transmission path.	It is a link state protocol and it analyzes different sources like the speed, cost and path congestion while identifying the shortest path.
4	It is basically use for smaller size organization.	It is basically use for larger size organization in the network.
5	It allows a maximum of 15 hops.	There is no such restriction on the hop count.
6	It is not a more intelligent dynamic routing protocol.	It is a more intelligent routing protocol than RIP.
7	The networks are classified as areas and tables here.	The networks are classified as areas, sub areas, autonomous systems and backbone areas here.
8	Its administrative distance is 120.	Its administrative distance is 110.
9	RIP uses UDP (User Datagram Protocol) Protocol.	OSPF works for IP (Internet Protocol) Protocol.
10	It calculates the metric in terms of Hop Count.	It calculates the metric in terms of bandwidth.

Chapter - 21

Two common methods for calculating the shortest path between the routers are —

1. Distance Vector Routing
2. Link State Routing

Distance Vector Routing — Each router periodically shares its knowledge about the entire network with its neighbours

The three keys to understand about how ~~network~~ this algorithm works —

1. Knowledge about the whole network :- Each router shares its knowledge about the entire network.
2. Routing only to neighbours — Each router periodically sends its knowledge about the network only to those routers to which it has direct links.

3. Information Sharing at regular intervals -

For example, every 30 seconds each router sends its information about the whole network to its neighbours.

Fig. below shows the 1st step in the algorithm. The text boxes indicate the relationships of the routers to their neighbours.

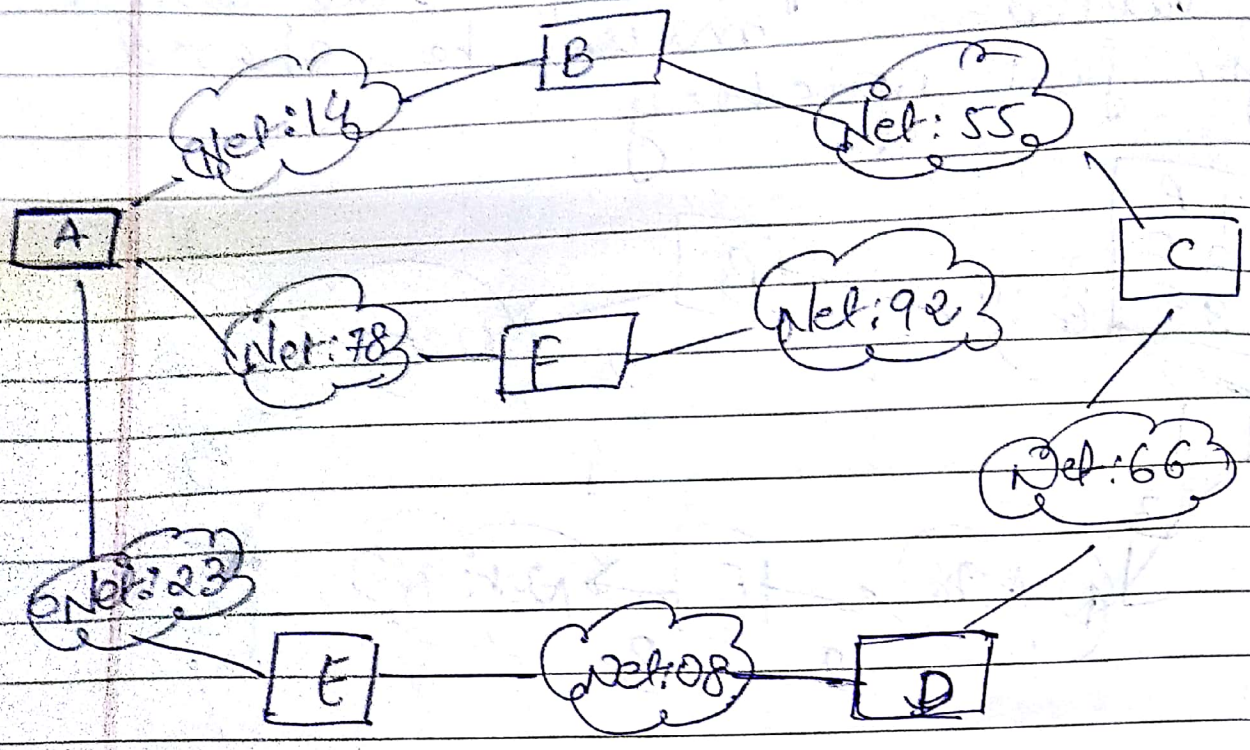
Each router sends its information about the internetwork only to its neighbours (immediate). How do then other routers learn about each other & share knowledge?

A router sends its knowledge to its neighbours. The neighbours add this knowledge to their own knowledge & send the whole table to their own neighbours. In this way, the 1st router gets its own information back plus new information about its neighbour's other neighbours

Each of these neighbours add its knowledge & sends the update table on to its neighbours! (to neighbours of neighbours of neighbours of the original router) & so on ...

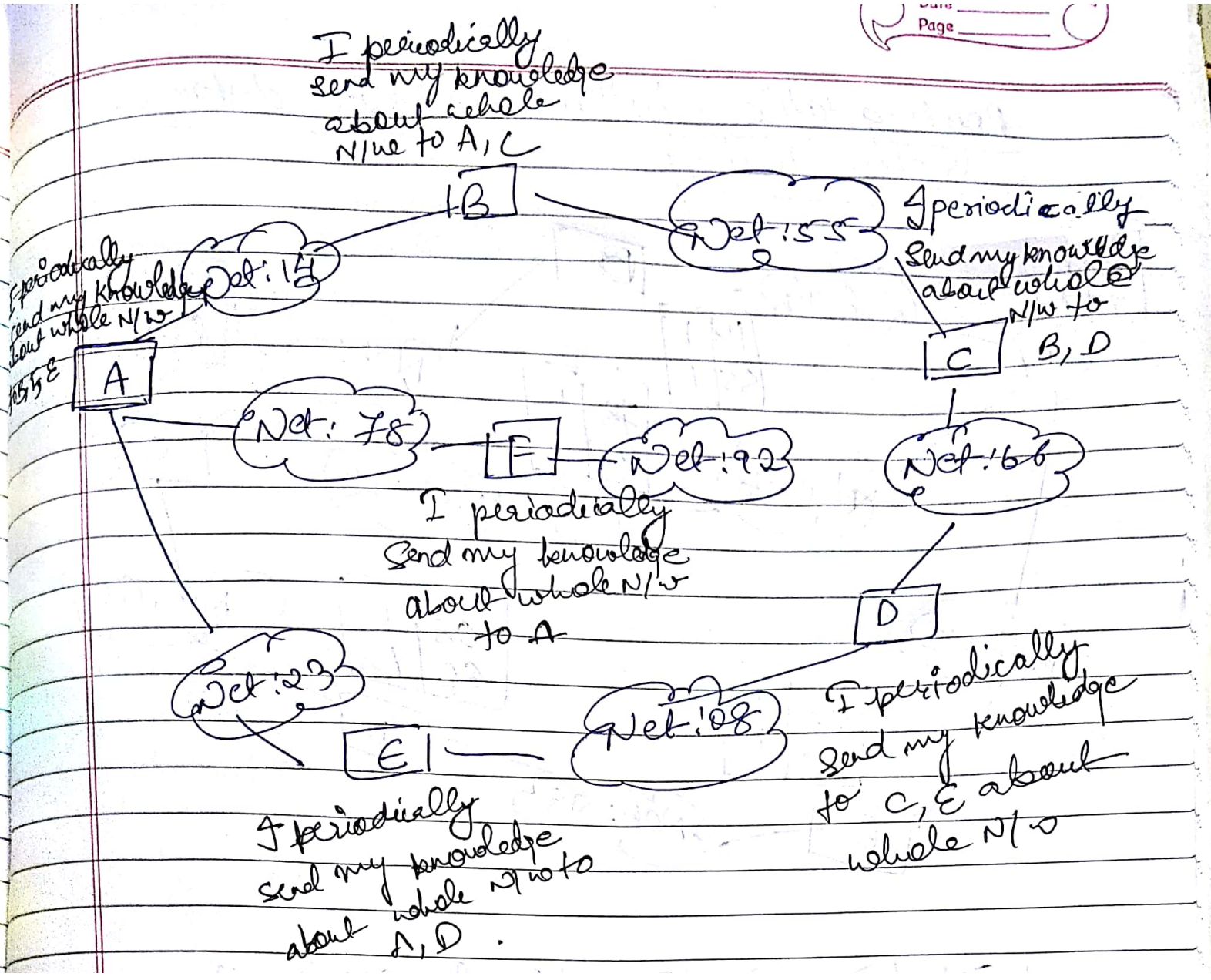
Eventually every router knows about every other router in the internetwork

Fig



In this example, clouds represents Local Area Networks (LANs). The number inside each cloud is that LAN's network ID.

These LANs can be of any type (Ethernet, Token Ring, FDDI, etc). The LANs ~~can be of any~~ are connected by Routers represented by boxes labeled A, B, C, D, E & F.


I periodically send my knowledge about whole N/w to A, C
B
Net: 55
Net: 15
I periodically Send my knowledge about whole N/w to A
A
C
I periodically Send my knowledge about whole N/w to B, D
Net: 78
F
Net: 92
Net: 66
I periodically Send my knowledge about whole N/w to A
D
Net: 23
Net: 08
E
I periodically send my knowledge about whole N/w to A, D.
I periodically Send my knowledge to C, E about whole N/w

Routing table –

N/W ID	Cost	Next hop

Updating the table –

When A receives a routing table from B, it uses the info to update its own table. It says to itself "B has sent me a table that shows how its packets can get to networks 55 & 14.

~~I know that my packets~~

I know that B is my neighbor, so my packet I can reach B in one hop.

So if I add one hop to all the costs shown in B's table, the sum will be my cost for reaching those other networks.

Updating Routing table for router A

A's old table

14	1	—
23	1	—
78	1	—

14	1
55	1

+ one hop

Received from B.

=

14	2	B
55	2	B

After adjustment

→

14	1	—
14	2	B
23	1	—
55	2	B
78	1	—

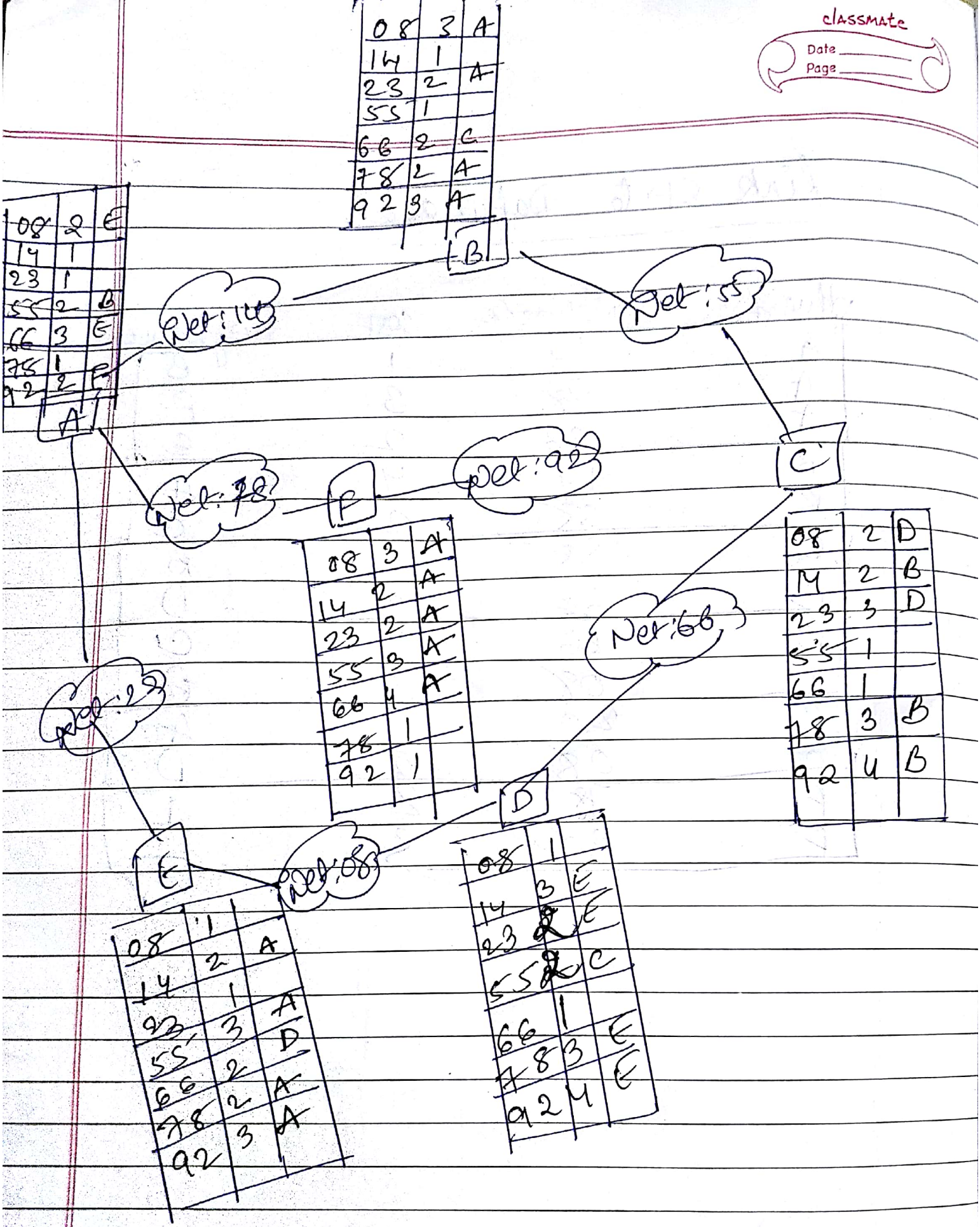
Combined

→

14	1	—
23	1	—
55	2	B
78	1	—

A's new table

This combined table may contain duplicate data for some n/w destinations. Router A : finds & purges any duplications & keeps advertised network shows the lowest cost.



FINAL ROUTING TABLES

Link State Routing :- The following are true of link state routing -

1. Knowledge about the neighbourhood

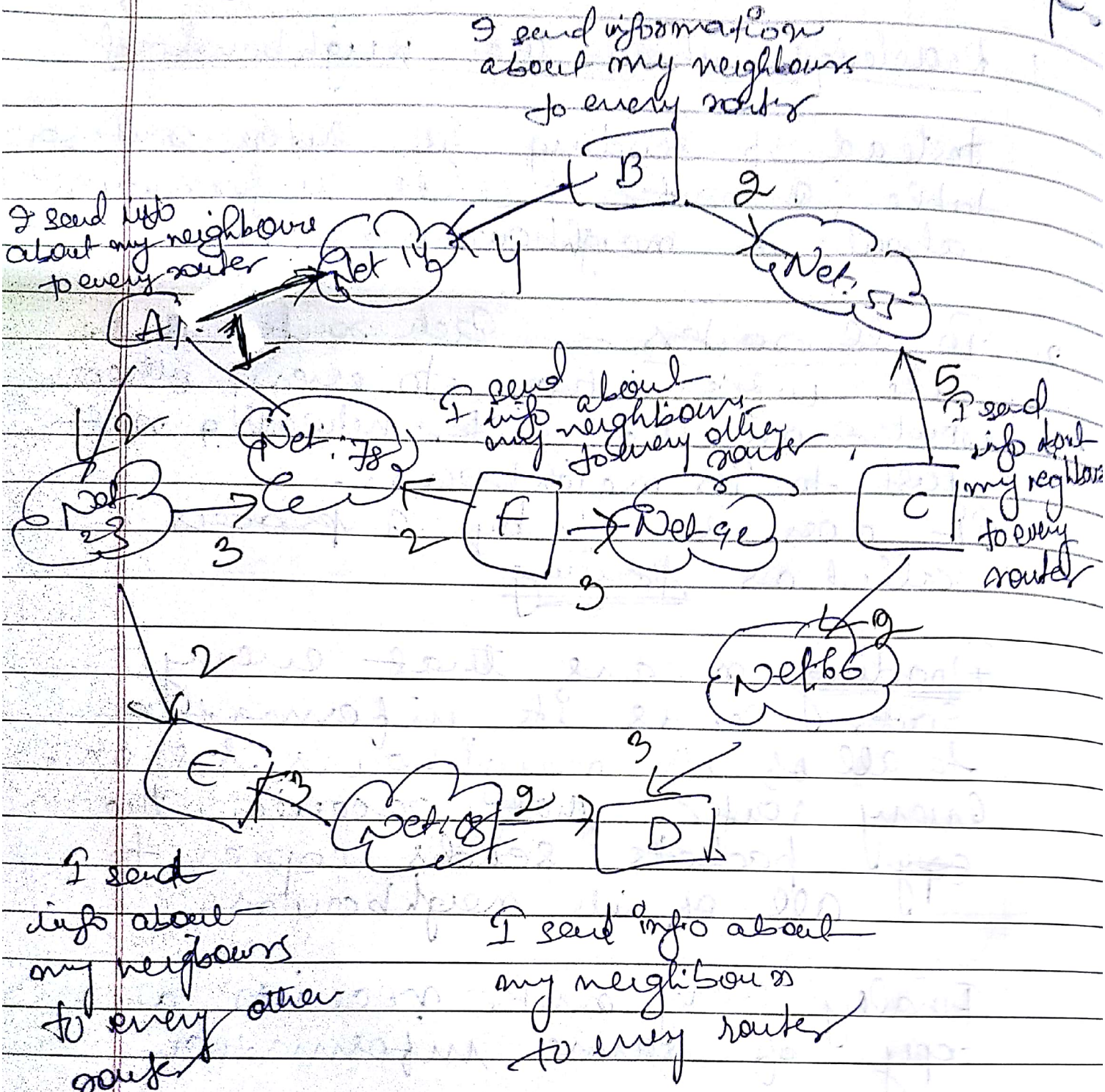
Instead of sending its entire routing table, a router sends information about its neighbourhood only.

2. To all routers - Each router sends its information to every other router on the internetwork, not just to its neighbours. It does so by a process called as flooding

Flooding means that every router sends its information to all of its neighbours. & so on. Every router that receives the ~~copy~~ packets sends copies to all of its neighbours.

Finally every router receives a copy of same information

2. Information Sharing when there is a change - Each router sends out information about the neighbours when there is a change.



Date
Page

The first step in the link state routing is information sharing as shown in fig. each G. router sends its knowledge about its neighbourhood to every other router

Packet Cost — On LSR, cost is a weighted value based on the variety of factors such as security, traffic, etc. — .

The cost of link from router A to network 14 might be different from cost from A to 23.

Link State Packet —

Advertiser	Network	cost	Neighbours

Getting information about neighbour

A router gets its information about its neighbours by periodically sending them a short greeting packet.

If the neighbour responds to the greeting packet, it is assumed to be alive & functioning.

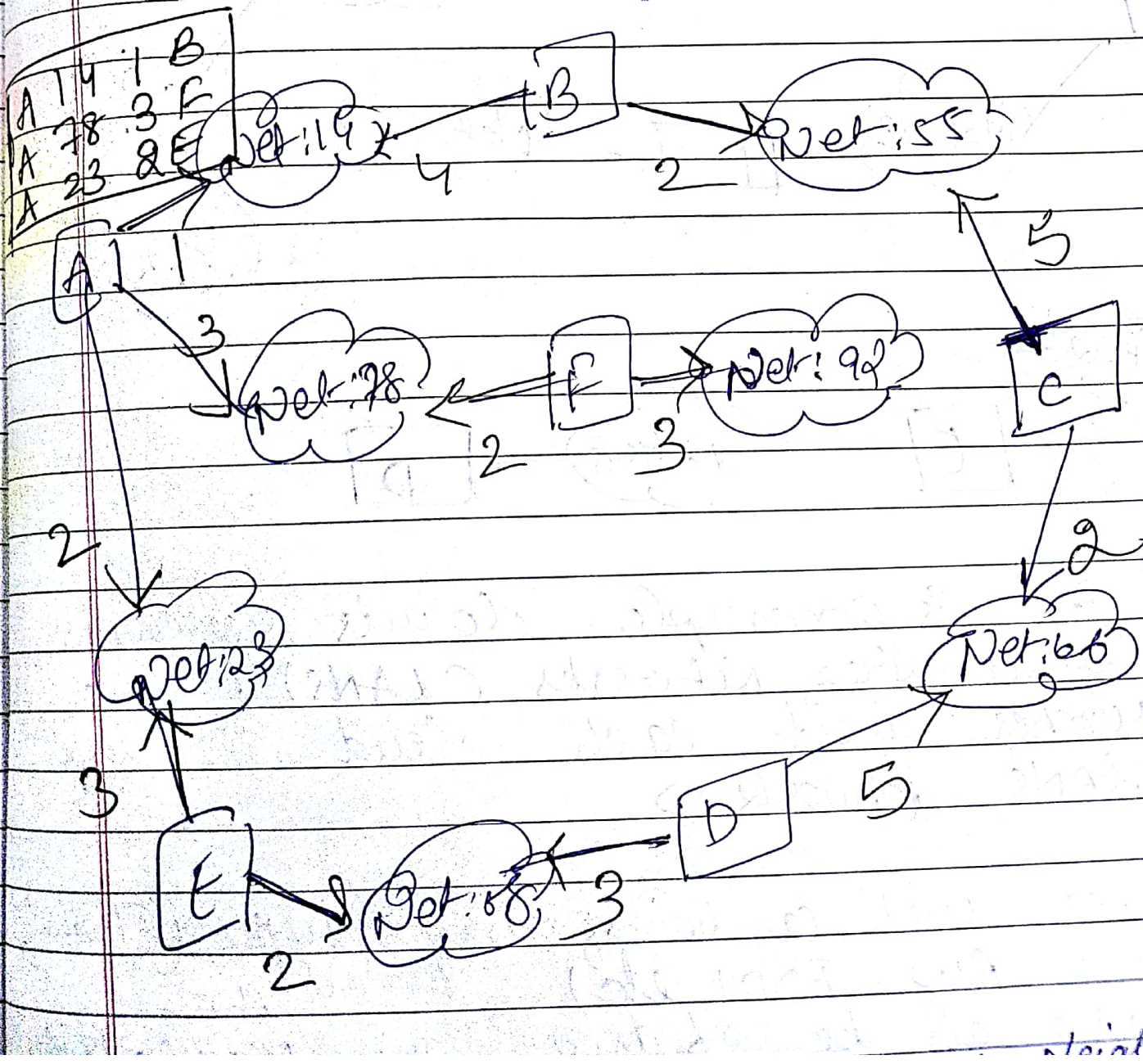
If it does not, a change is assumed to have occured & the sending router then alerts the rest of the network in the next LSP.

Link State Database - Every

Router receives every LSP & puts the information into LSD.

Because every router receives the same LSPs, every router builds the same database. It stores this database on its disk and uses it to calculate its routing table

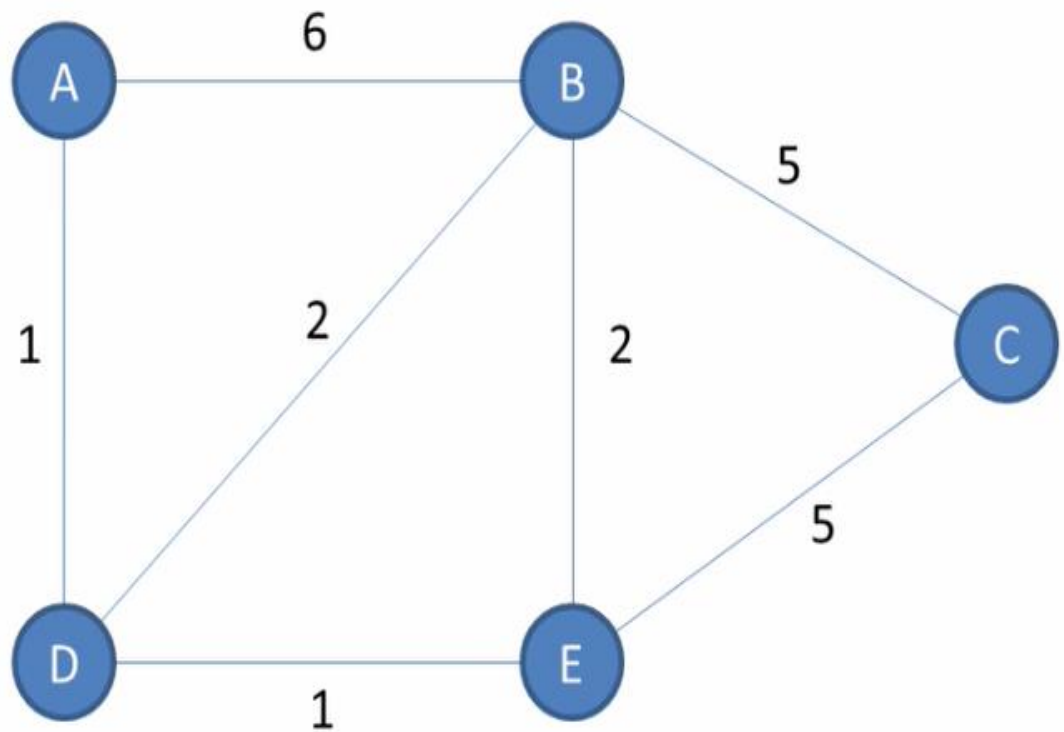
If the router is added or deleted from the system the whole database must be shared for fast updating.



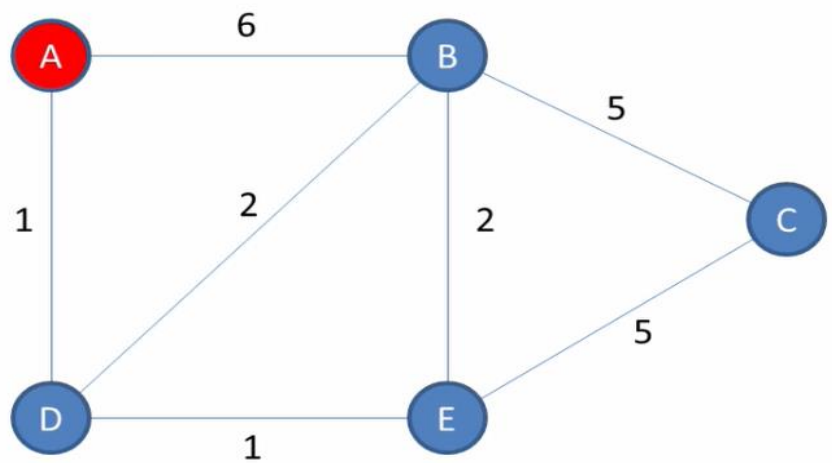
Link State Database

Advertiser	Network	Cost	Neighbour
A	14	1	B
A	28	3	F
A	23	2	E
B	14	4	A
B	55	2	C
C	55	5	B
C	66	2	D
D	66	5	C
D	08	3	E
E	23	3	A
E	08	2	D
F	78	2	A
F	92	3	—

Dijkstra Algorithm


A
B
C
D
E
6
5
1
2
2
5
1

Consider the start vertex, A



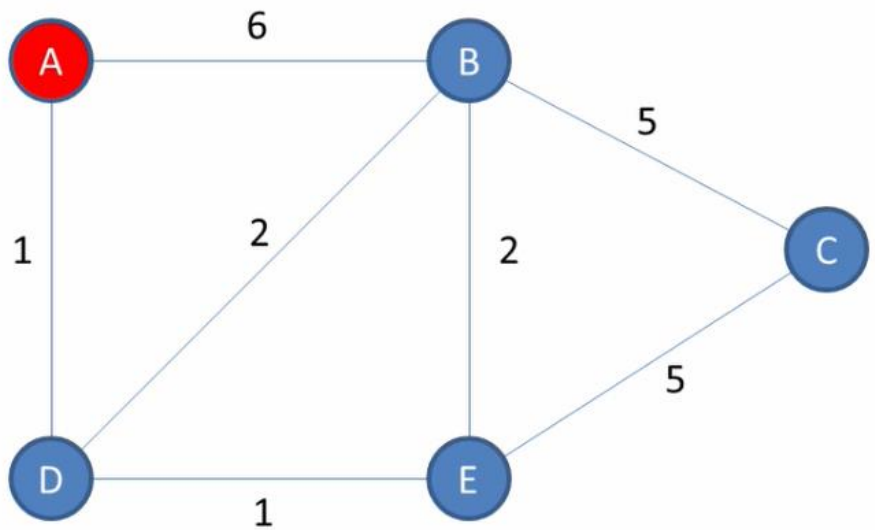
Vertex	Shortest distance from A	Previous vertex
A		
B		
C		
D		
E		

Visited = [ ]

Unvisited = [A, B, C, D, E]

Visit the unvisited vertex with the smallest known distance from the start vertex

First time around, this is the start vertex itself, A

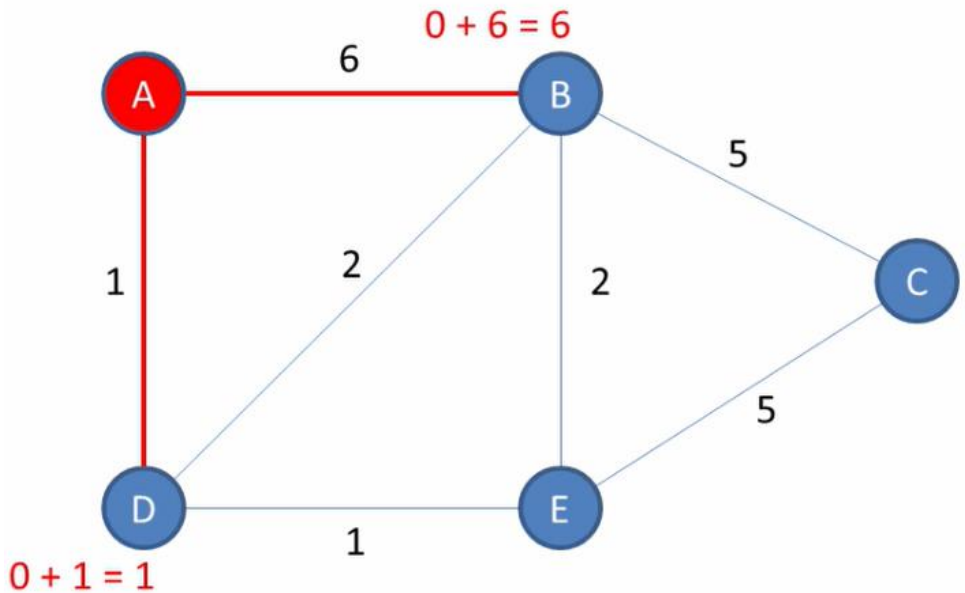


Vertex	Shortest distance from A	Previous vertex
A	0	
B	∞	
C	∞	
D	∞	
E	∞	

Visited = [ ]

Unvisited = [A, B, C, D, E]

For the current vertex, calculate the distance of each neighbour from the start vertex



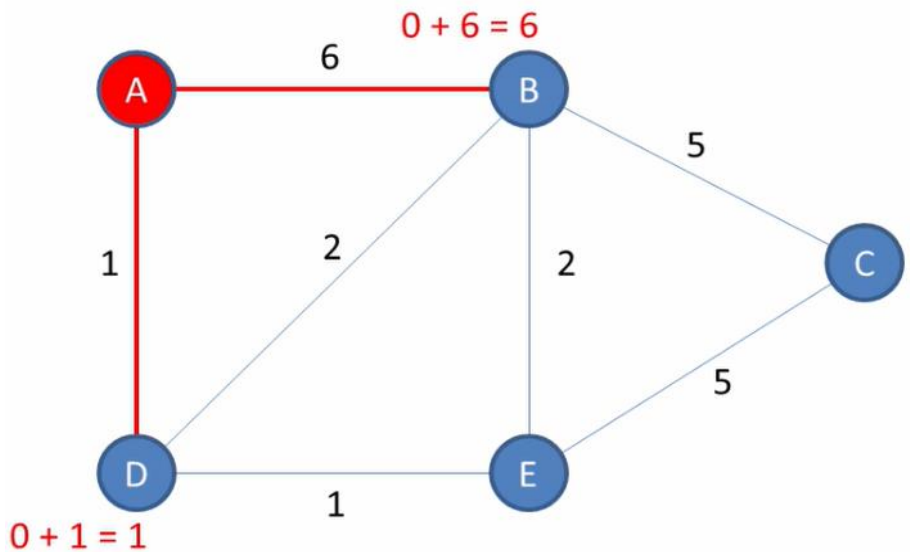
Vertex	Shortest distance from A	Previous vertex
A	0	
B	∞	
C	∞	
D	∞	
E	∞	

Visited = [ ]

Unvisited = [A, B, C, D, E]

Update the previous vertex for each of the updated distances

In this case we visited B and D via A

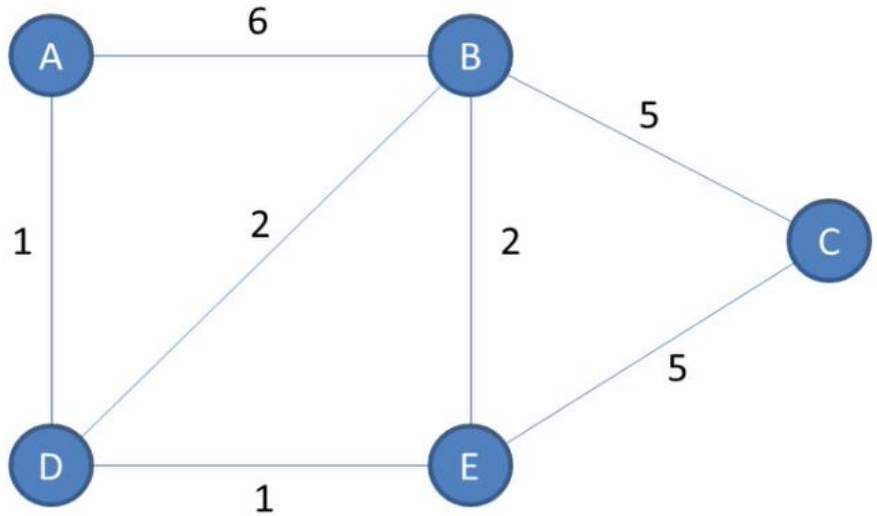


Vertex	Shortest distance from A	Previous vertex
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Visited = [ ]

Unvisited = [A, B, C, D, E]

Add the current vertex to the list of visited vertices



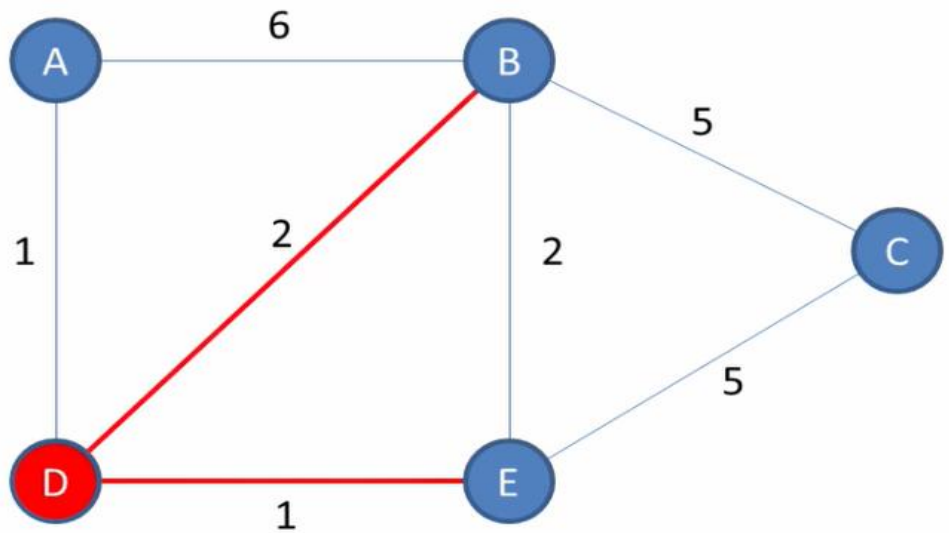
Vertex	Shortest distance from A	Previous vertex
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Visited = [A]

Unvisited = [B, C, D, E]

For the current vertex, examine its unvisited neighbours

We are currently visiting D and its unvisited neighbours are B and E

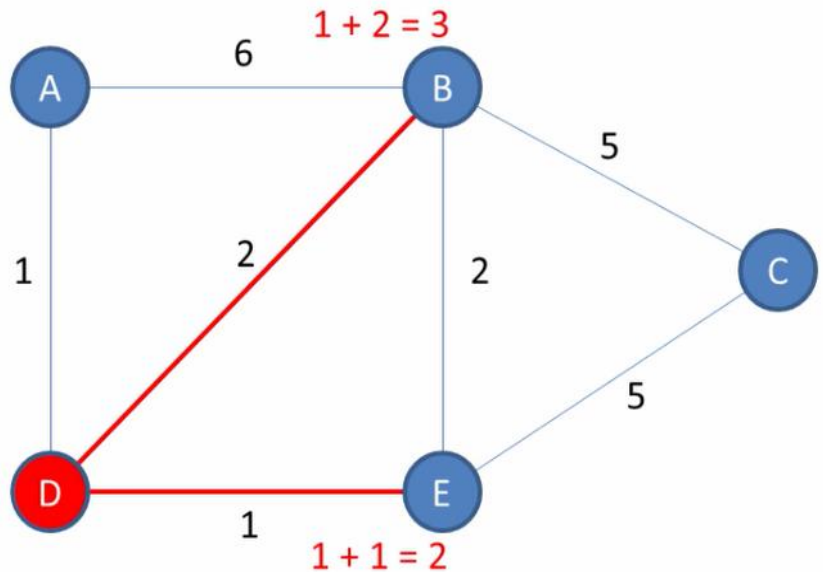


Vertex	Shortest distance from A	Previous vertex
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Visited = [A]

Unvisited = [B, C, D, E]

For the current vertex, calculate the distance of each neighbour from the start vertex



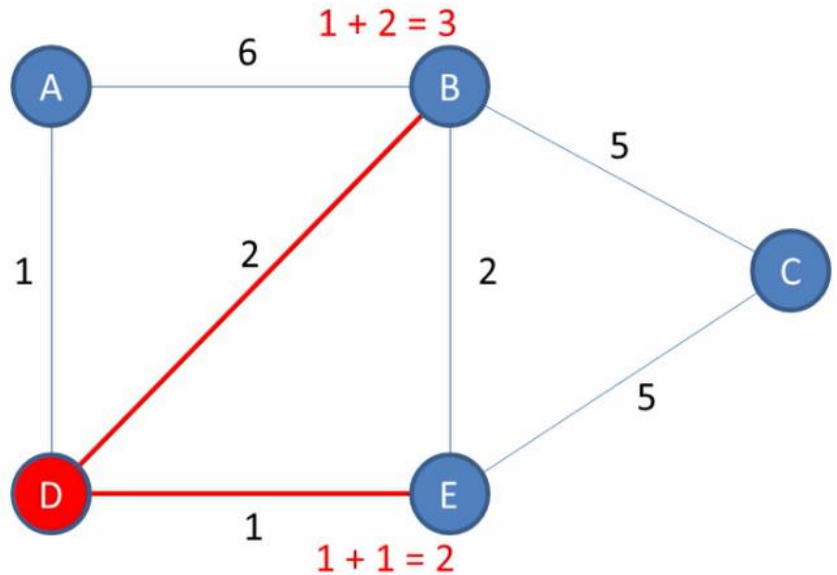
Vertex	Shortest distance from A	Previous vertex
A	0	
B	6	A
C	∞	
D	1	A
E	∞	

Visited = [A]

Unvisited = [B, C, D, E]

Update the previous vertex for each of the updated distances

In this case we visited B and E via D

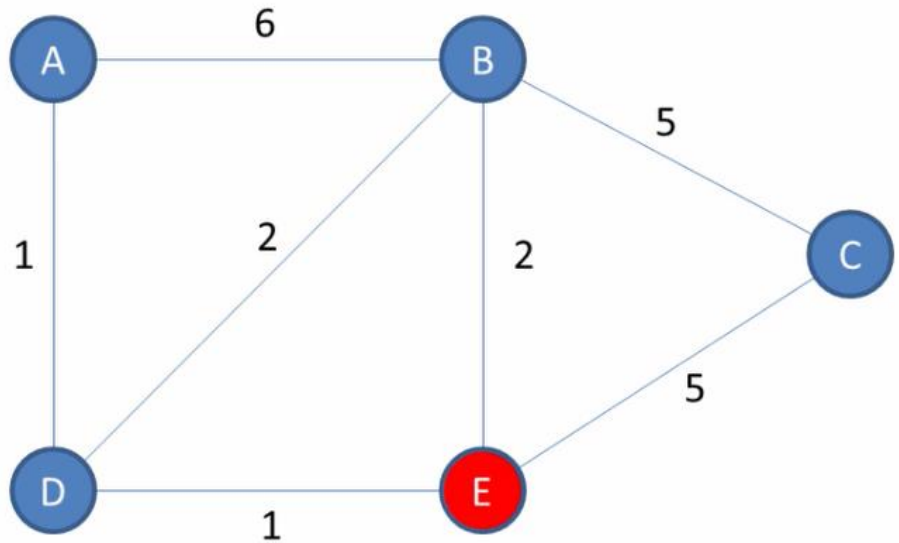


Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	A
C	∞	
D	1	A
E	2	

Visited = [A]

Unvisited = [B, C, D, E]

Visit the unvisited vertex with the smallest known distance from the start vertex
This time around, it is vertex E

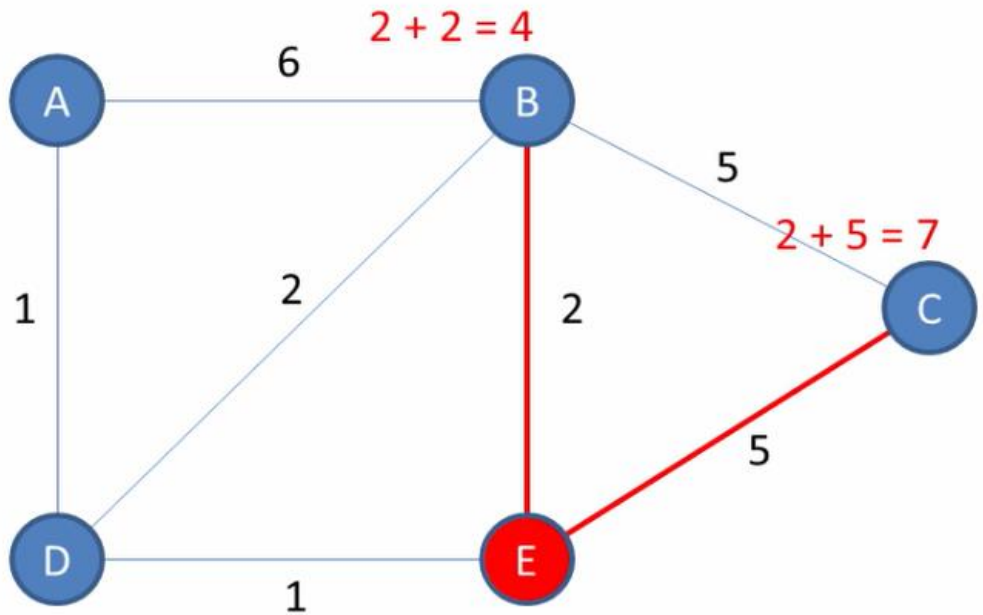


Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	∞	
D	1	A
E	2	D

Visited = [A, D]

Unvisited = [B, C, E]

For the current vertex, calculate the distance of each neighbour from the start vertex



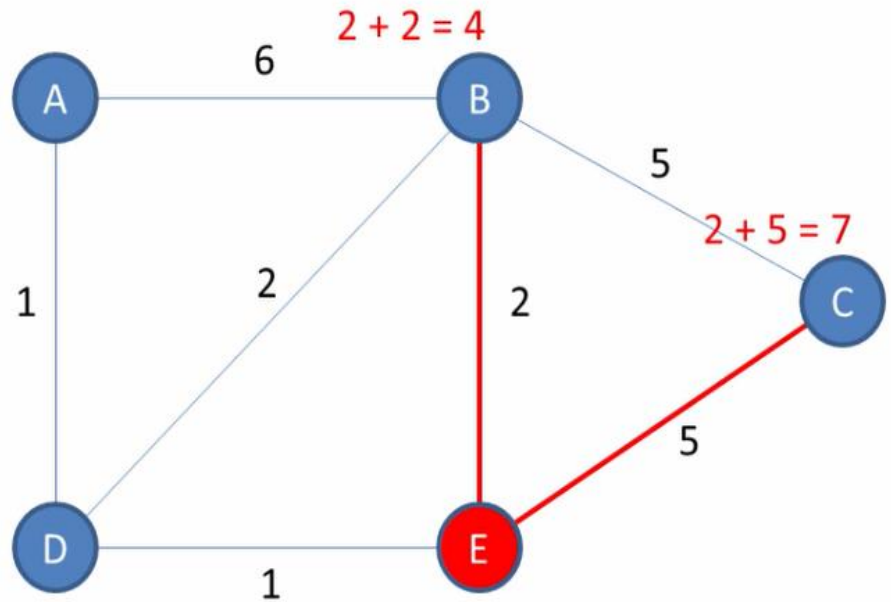
Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	∞	
D	1	A
E	2	D

Visited = [A, D]

Unvisited = [B, C, E]

Update the previous vertex for each of the updated distances

In this case we visited C via E

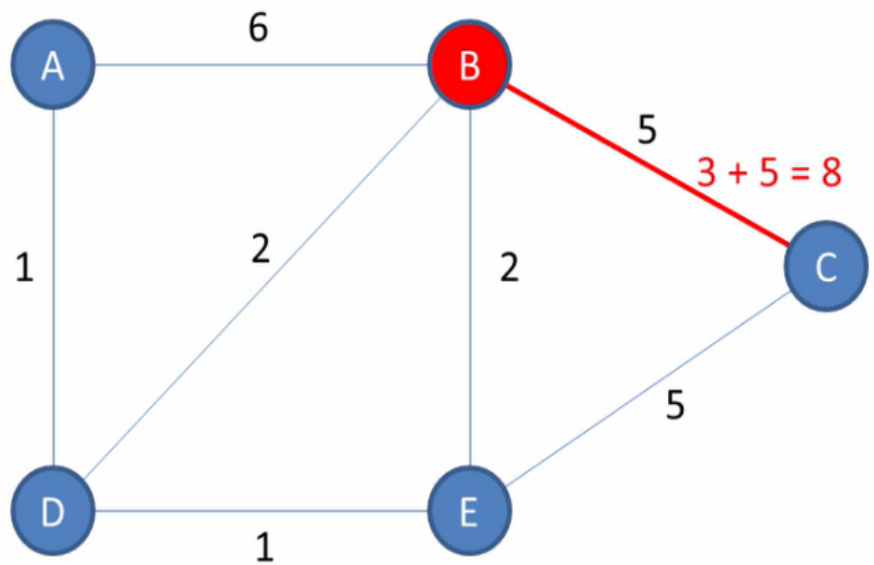


Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Visited = [A, D]

Unvisited = [B, C, E]

For the current vertex, calculate the distance of each neighbour from the start vertex



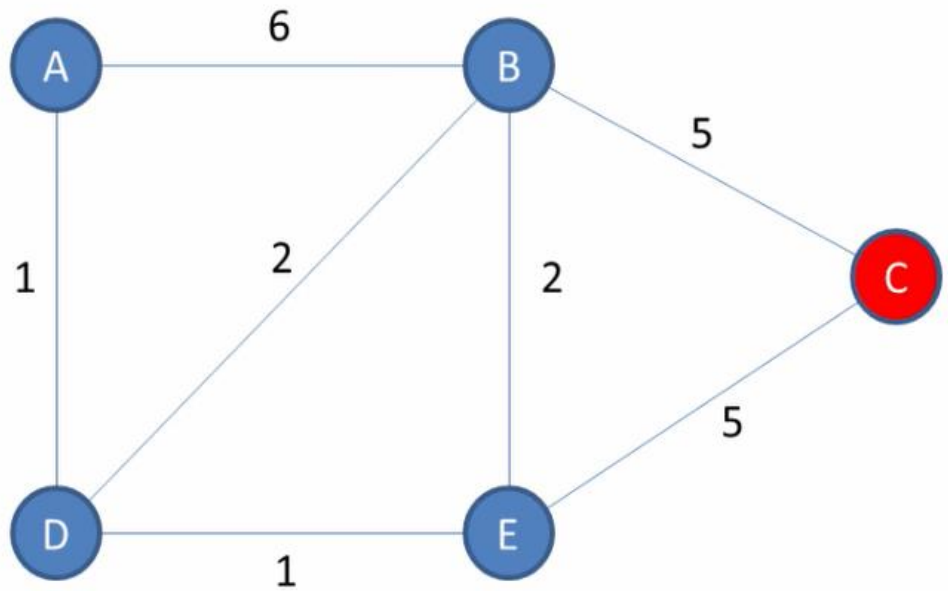
Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Visited = [A, D, E]

Unvisited = [B, C]

For the current vertex, examine its unvisited neighbours

We are currently visiting C and it has no unvisited neighbours



Vertex	Shortest distance from A	Previous vertex
A	0	
B	3	D
C	7	E
D	1	A
E	2	D

Visited = [A, D, E, B] Unvisited = [C]

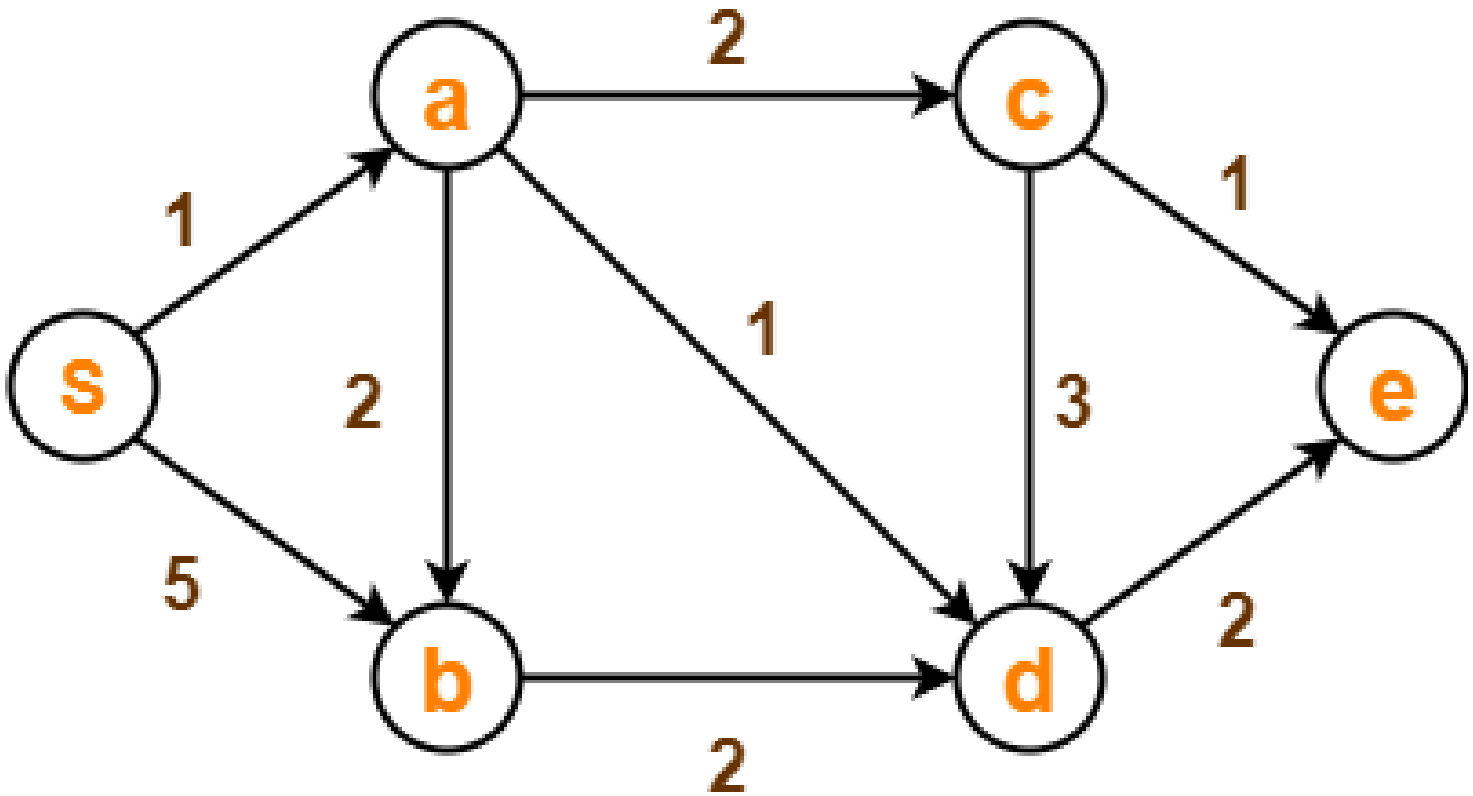
Algorithm

Let distance of start vertex from start vertex = 0
Let distance of all other vertices from start = ∞ (infinity)

Repeat
- Visit the unvisited vertex with the smallest known distance from the start vertex
- For the current vertex, examine its unvisited neighbours
- For the current vertex, calculate distance of each neighbour from start vertex
- If the calculated distance of a vertex is less than the known distance, update the shortest distance
- Update the previous vertex for each of the updated distances
- Add the current vertex to the list of visited vertices

Until all vertices visited

HOME WORK



Bellman-ford Algorithm

→ It is used to find all shortest path in a graph from source to all other nodes.

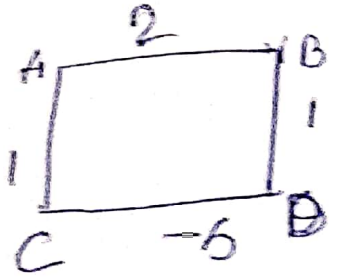
→ The algorithm requires that the graph does not contain any cycles of negative weight. but if it does, the algorithm is able to detect it.

Or

If there is a negative weight cycle, then shortest distance is not calculated, negative weight cycle is reported.

What is negative Cycle?

A cycle whose total weight is negative called negative



Cycle - A+B-D-C

weight - 2+1+(-5)+1

= (-1)

negative value means a negative cycle in graph.

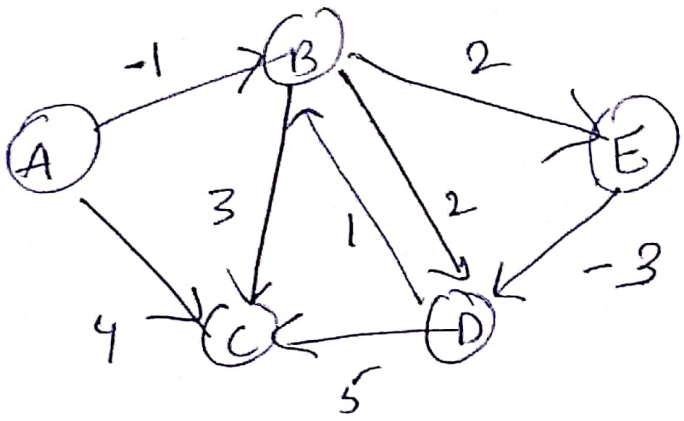
To detect negative cycle =

No. of iterations = No of vertices in graph.

To detect shortest path -

No of iterations = No. of vertices - 1.

Eg1.



No of vertices = 5

Max. no. of iteration

= 5 - 1

= 4

Initially

0	∞	∞	∞	∞
A	B	C	D	E

Source vertex A

1st Iteration

	A	B	C	D	E
A	0	-1	4	∞	∞
B	0	-1	2	~~∞~~	~~1~~
C	0	-1	2	1	1
D	0	-1	2	1	1
E	0	-1	2	-2	1

4th Iteration

	A	B	C	D	E	
A	0	-1	2	-2	1	- no change
B	1	-1	+2	-2	1	- No change
C	0	-1	2	-2	1	- No change
D	0	-1	2	-2	1	- No change
E	0	-1	2	-2	1	- No change

As distances are minimized after the second iteration, so third & forth iteration done update the distances.

But in computer loop will go upto (n-1) iterations i.e 4 iteration.