Destination Sequenced Distance Vector(DSDV)

Distance Vector Routing

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.

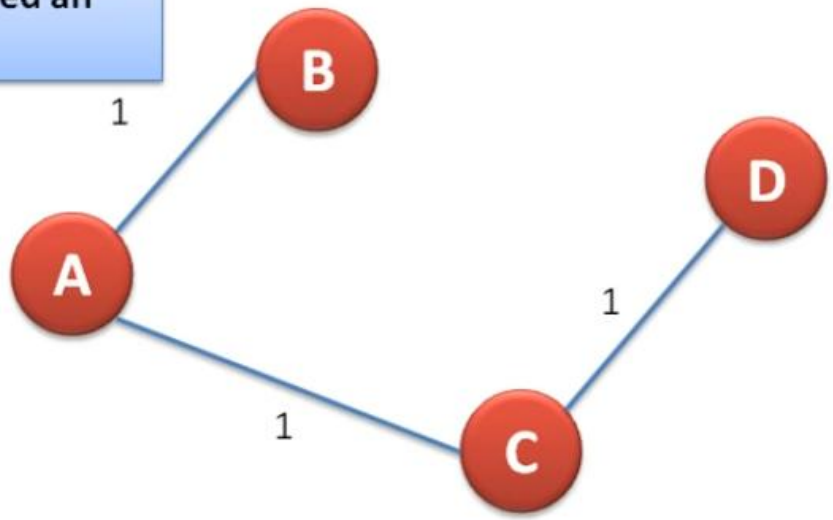
A link that is down/Not reachable is assigned an infinite cost.

Distance Vector Routing

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.

A link that is down/Not reachable is assigned an infinite cost.

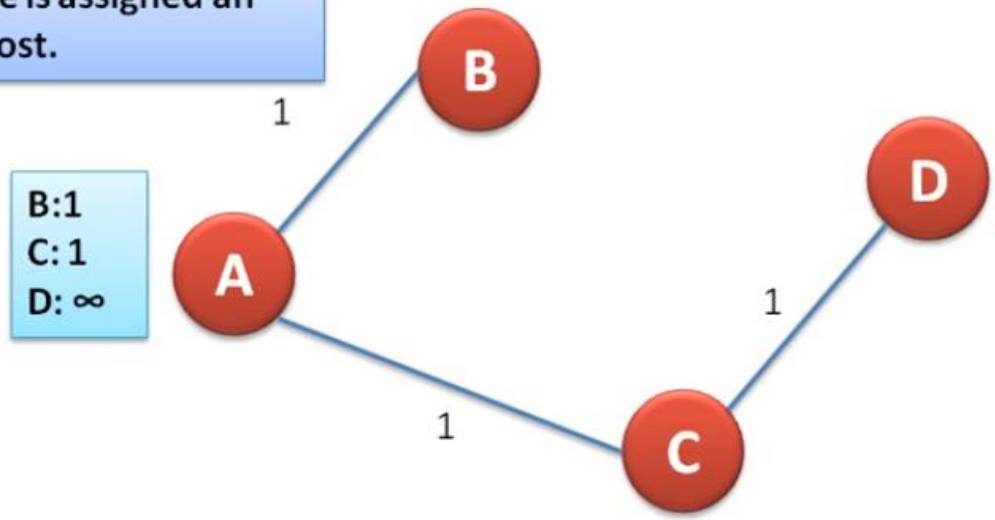


Distance Vector Routing

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.

A link that is down/Not reachable is assigned an infinite cost.

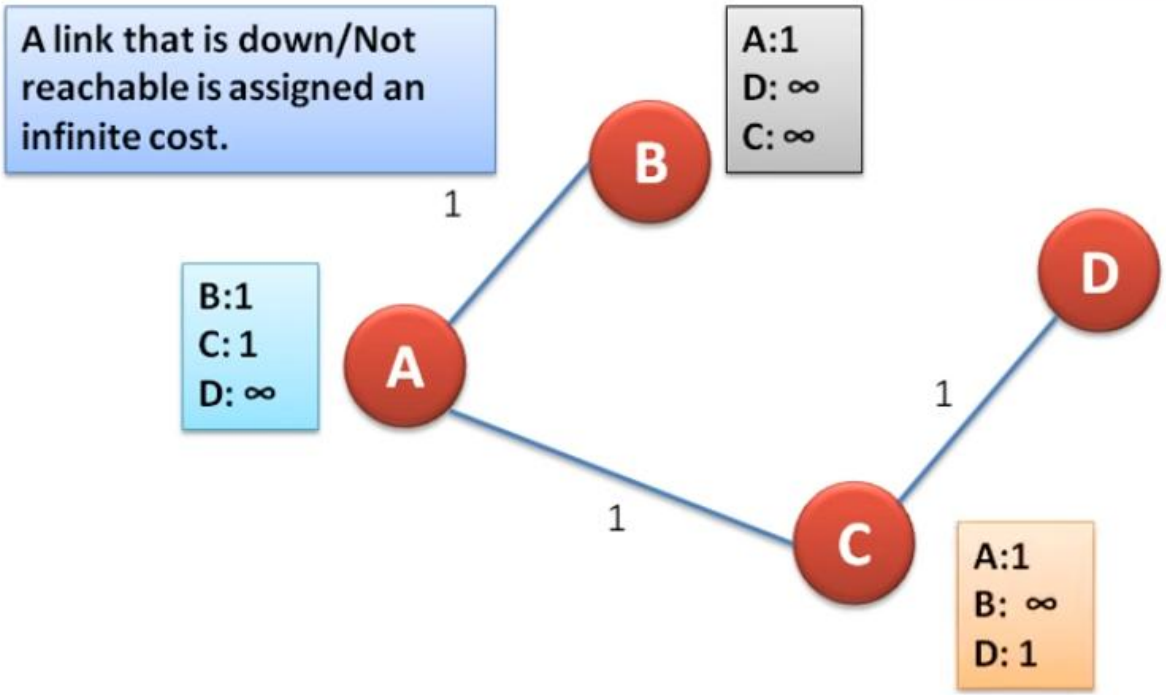


Distance Vector Routing

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.

A link that is down/Not reachable is assigned an infinite cost.

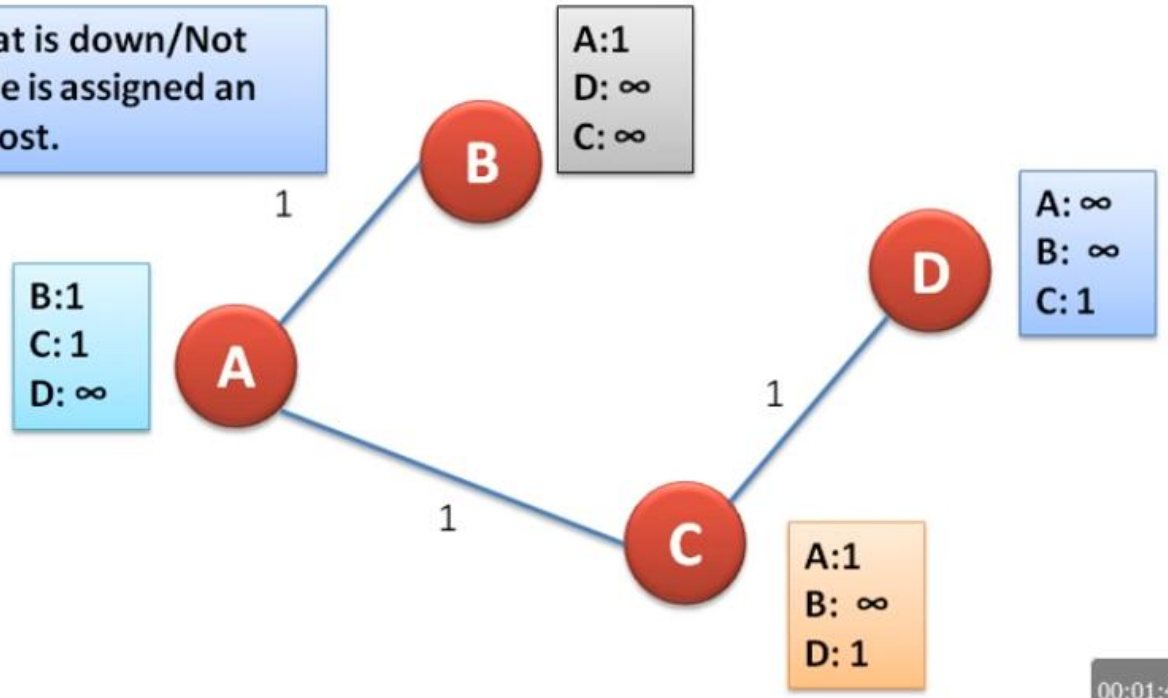


Distance Vector Routing

Each node constructs a one-dimensional array containing the "distances"(costs) to all other nodes and distributes that vector to its immediate neighbors.

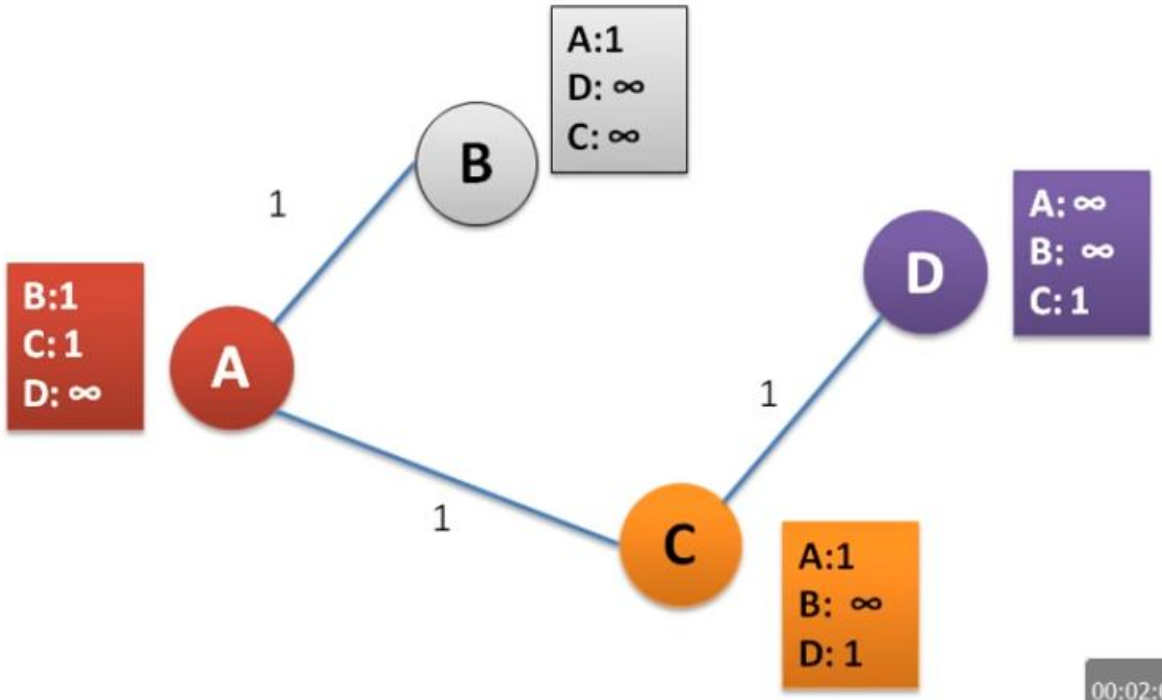
The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors.

A link that is down/Not reachable is assigned an infinite cost.

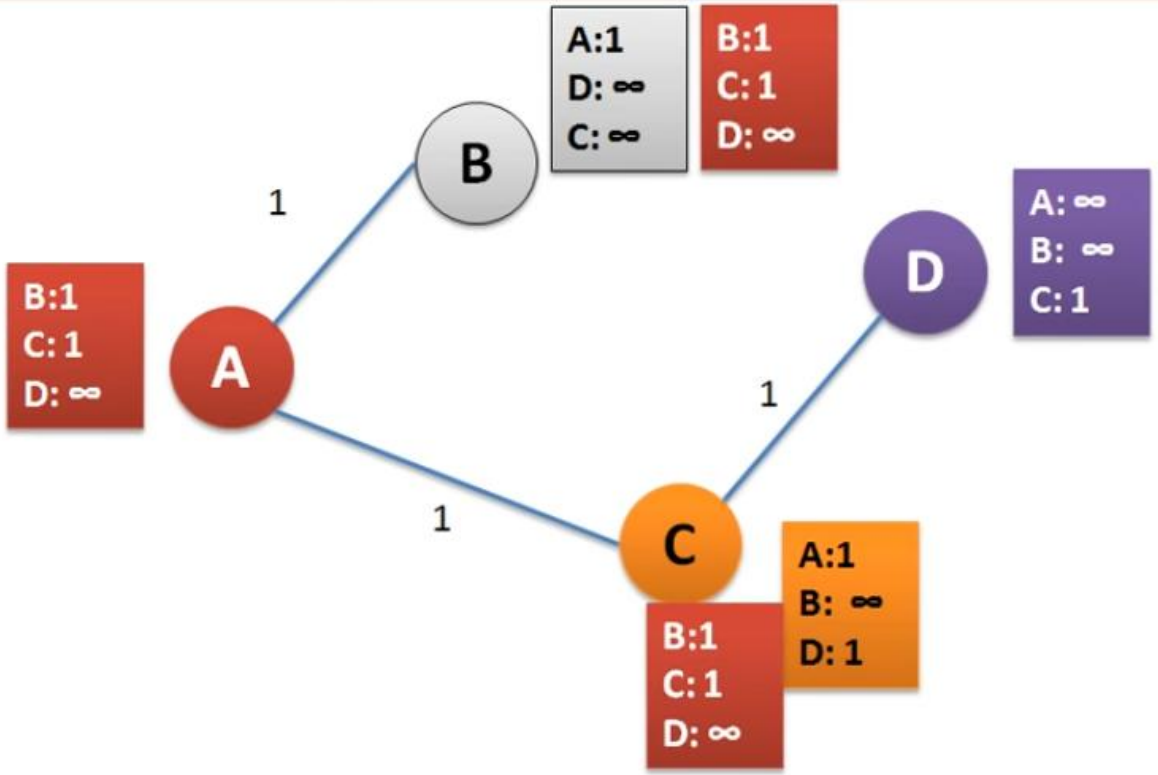


Distance Vector Routing

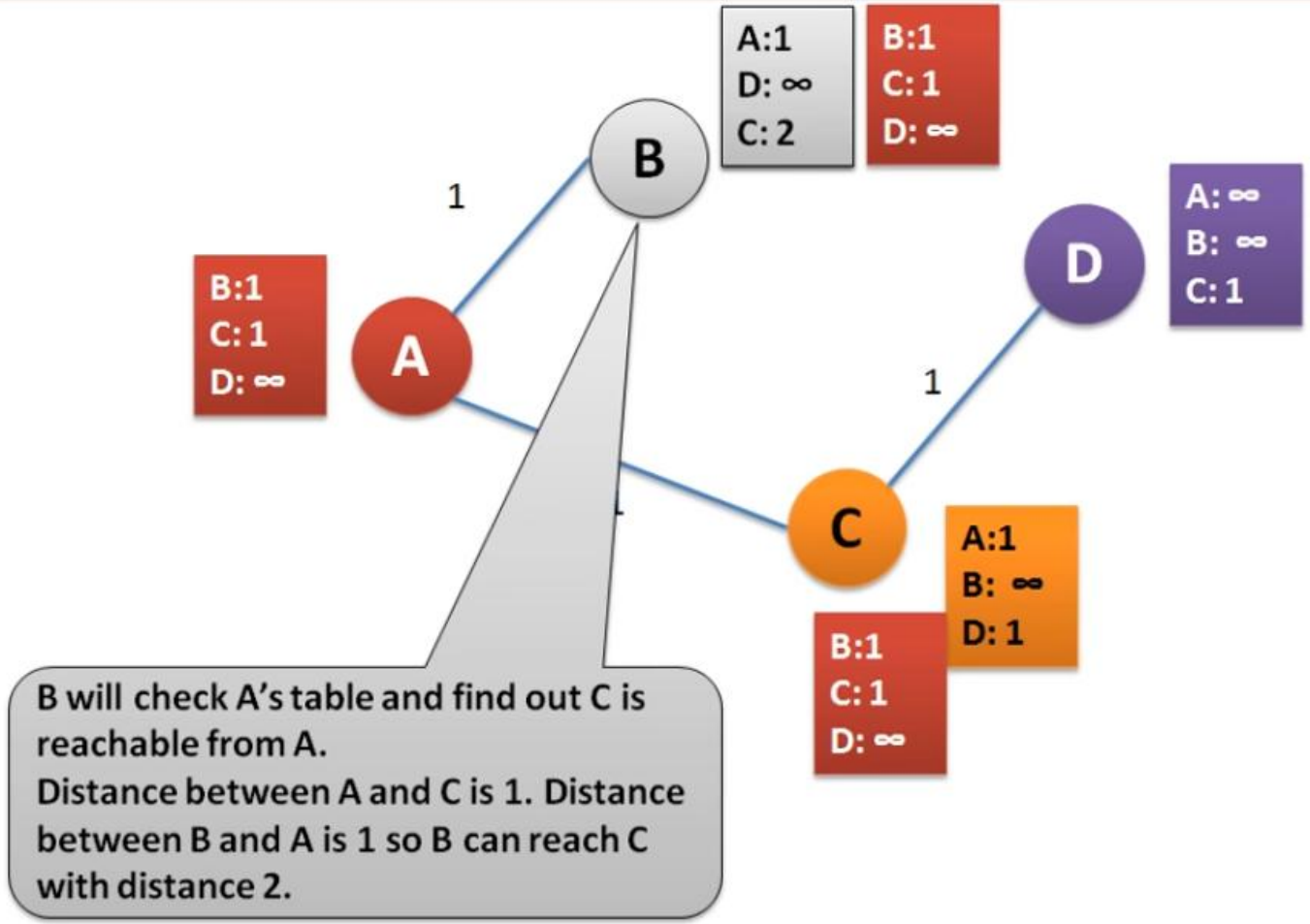
Each node broadcast their table to it's neighbours.



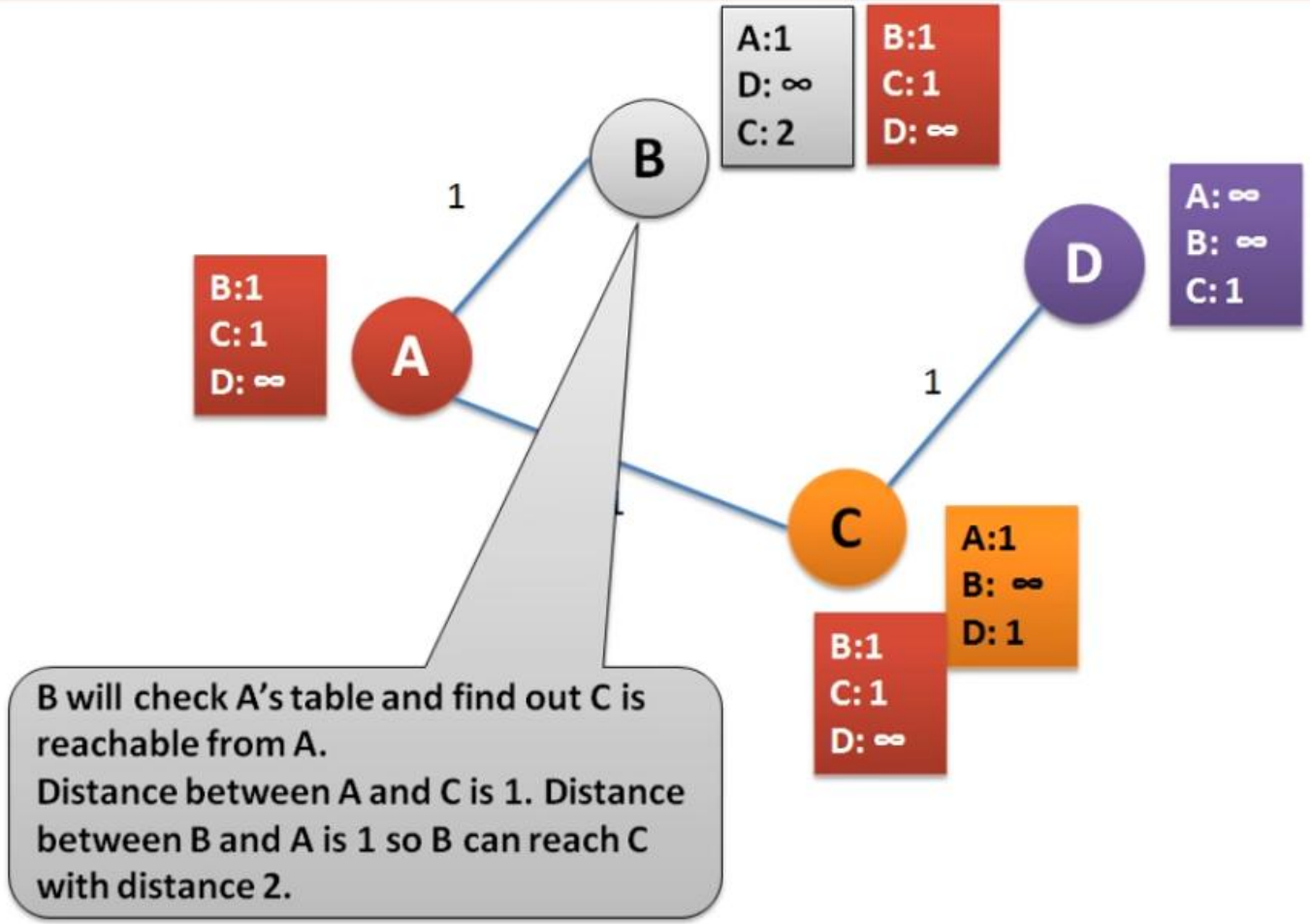
Distance Vector Routing



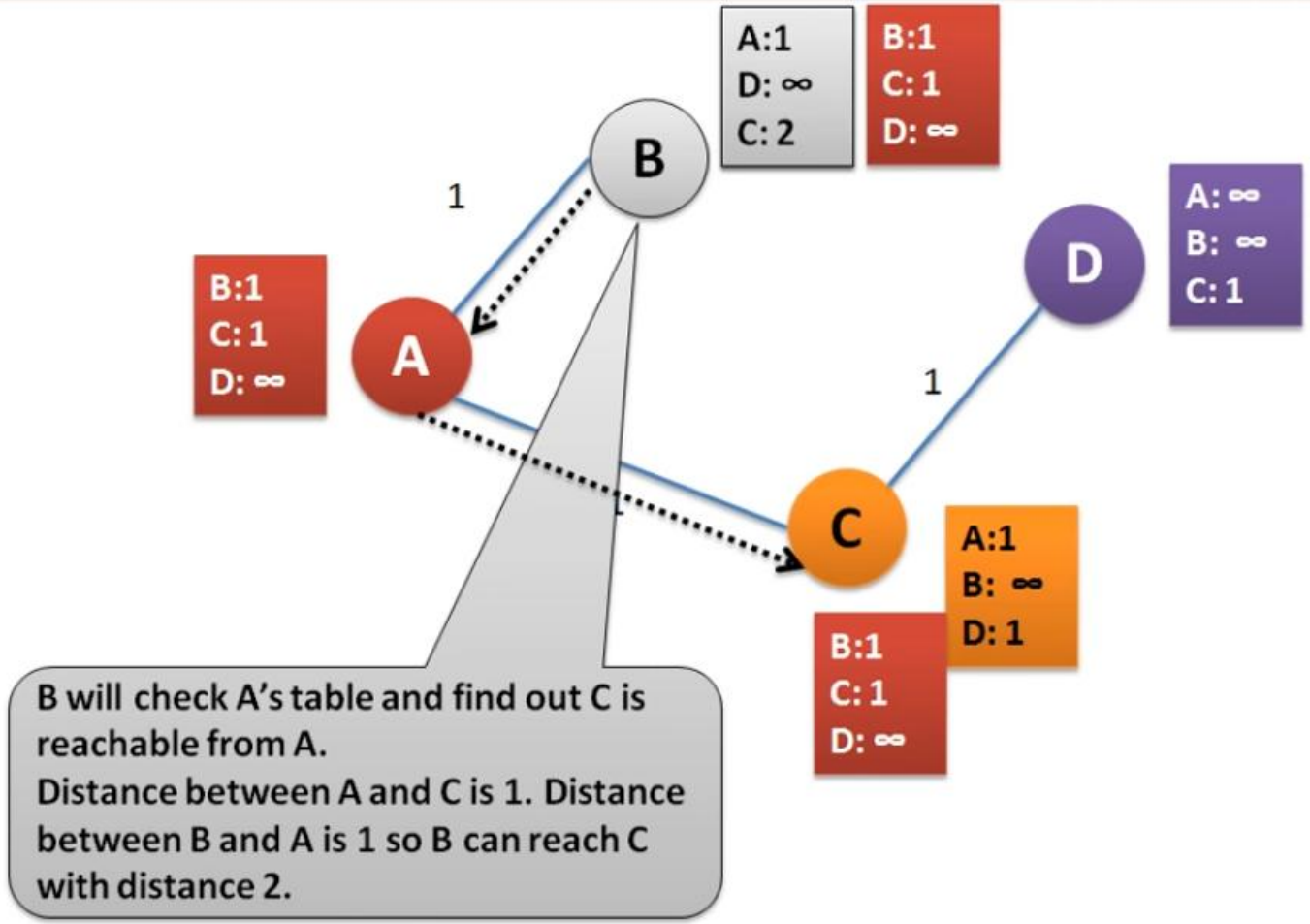
Distance Vector Routing



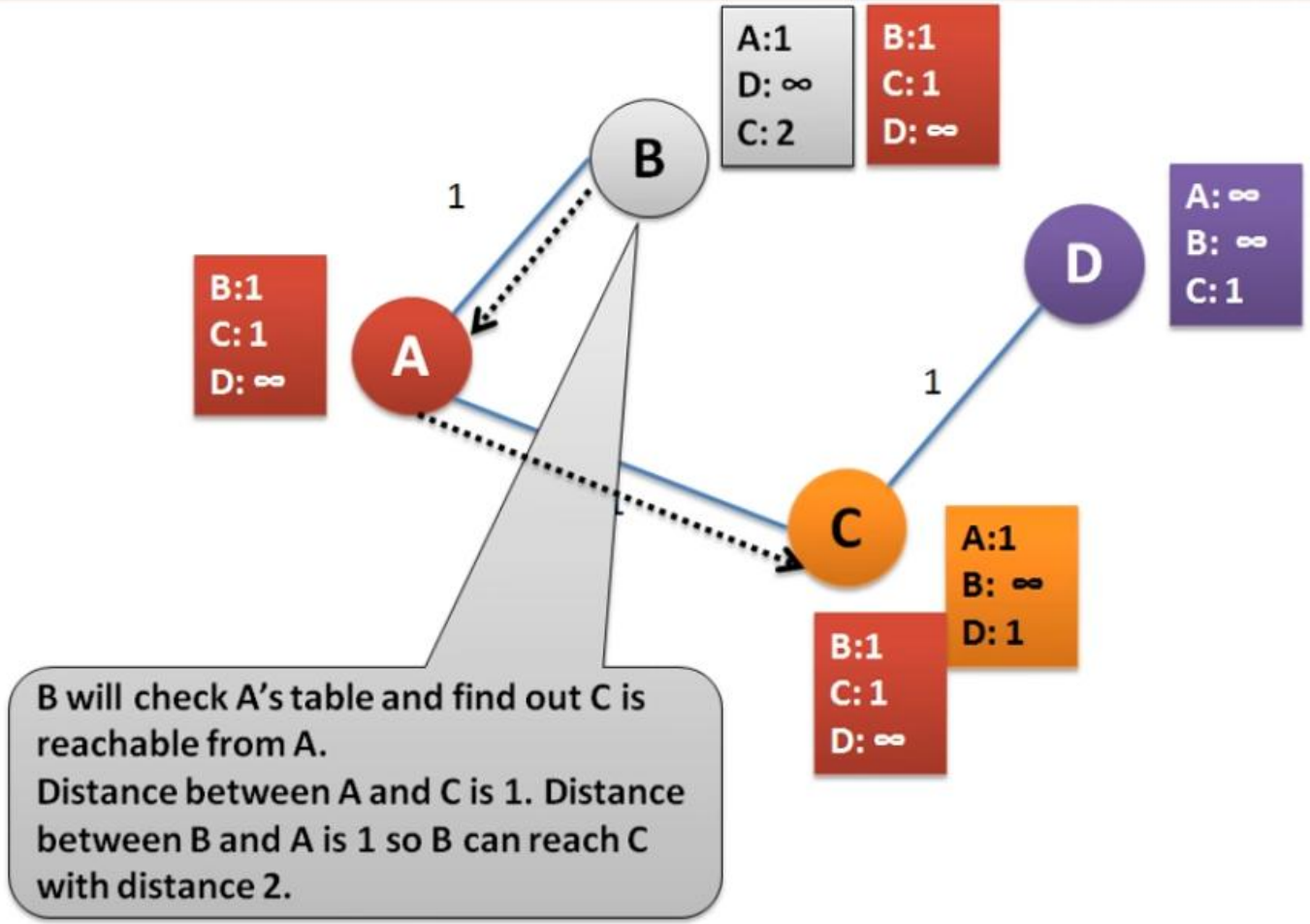
Distance Vector Routing



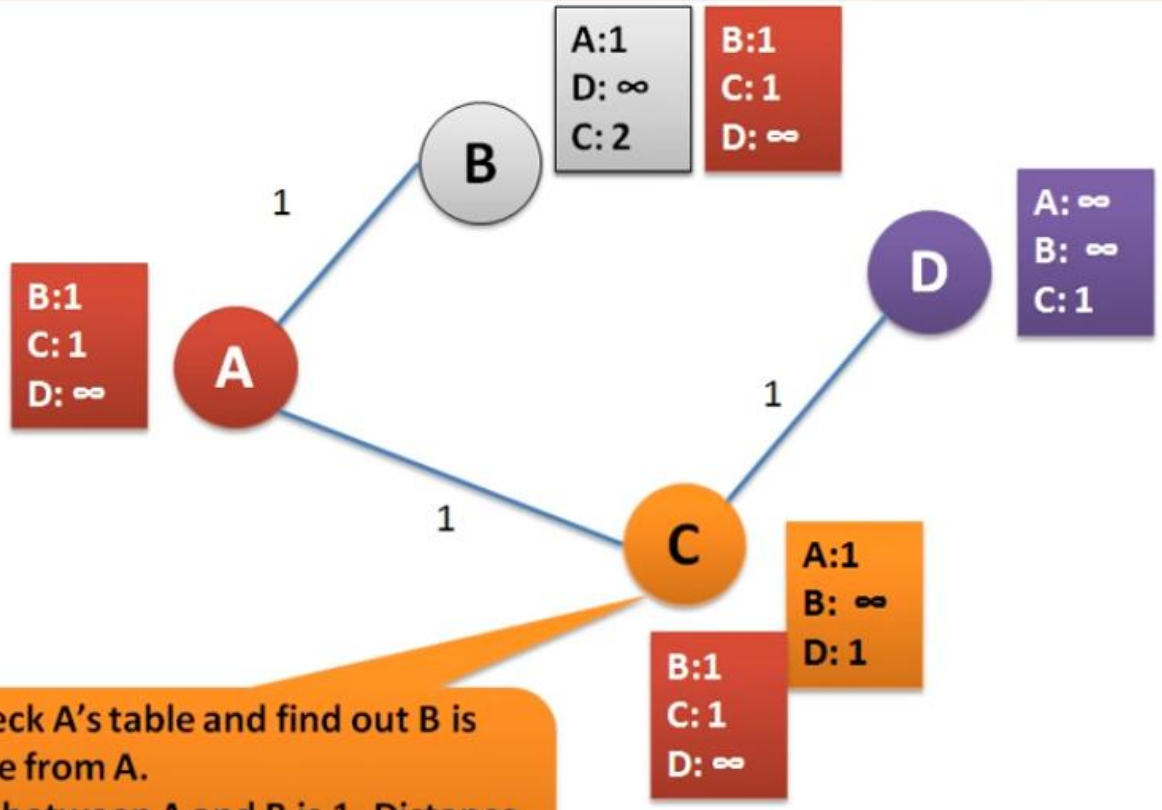
Distance Vector Routing



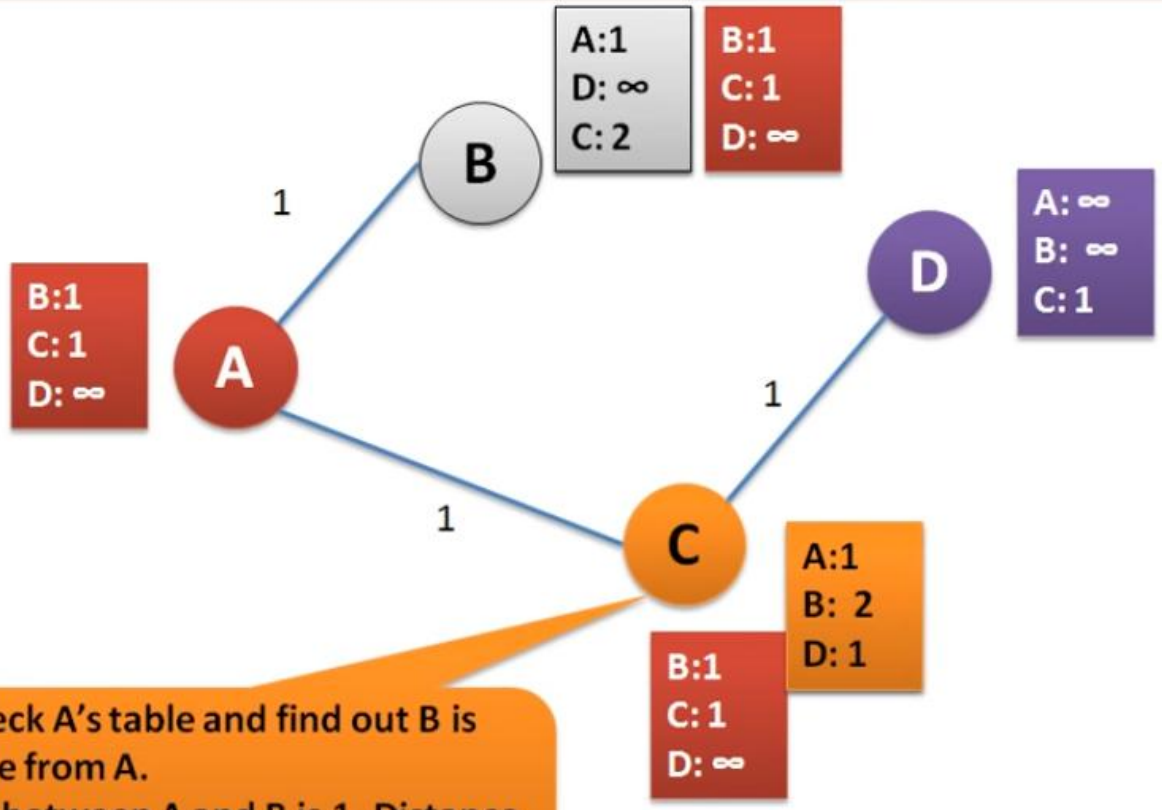
Distance Vector Routing



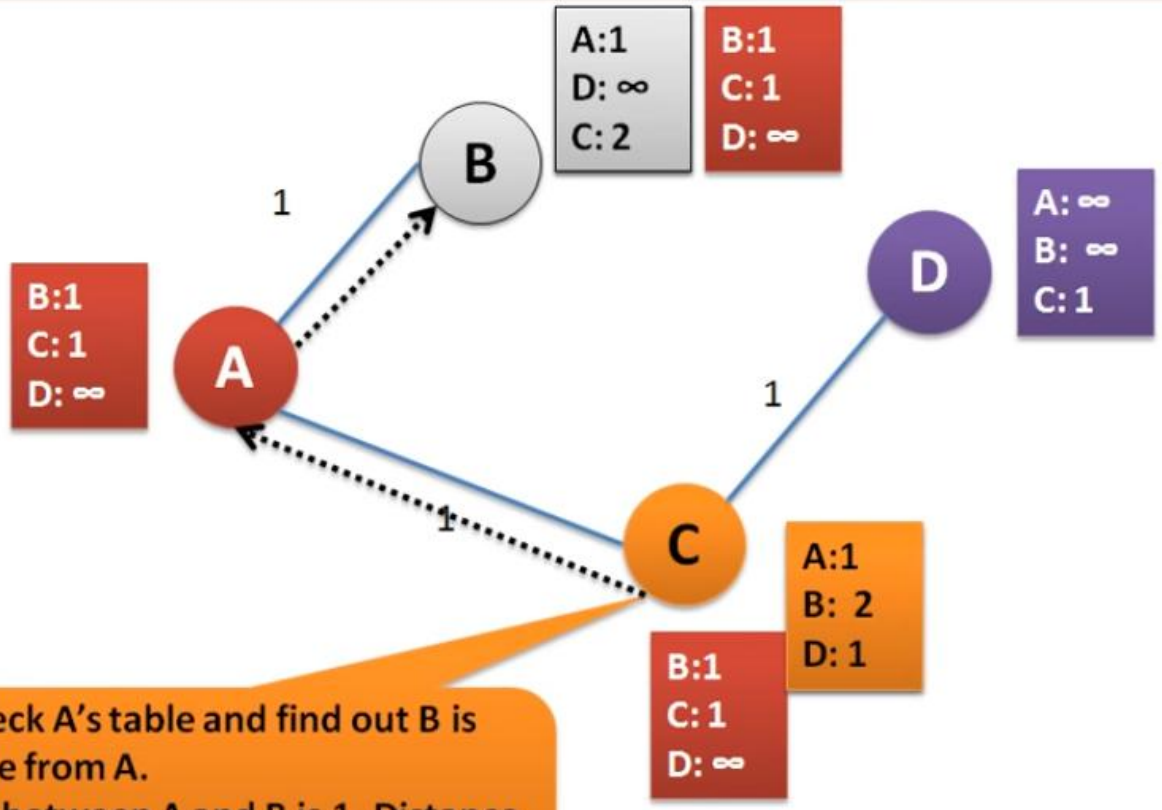
Distance Vector Routing



Distance Vector Routing

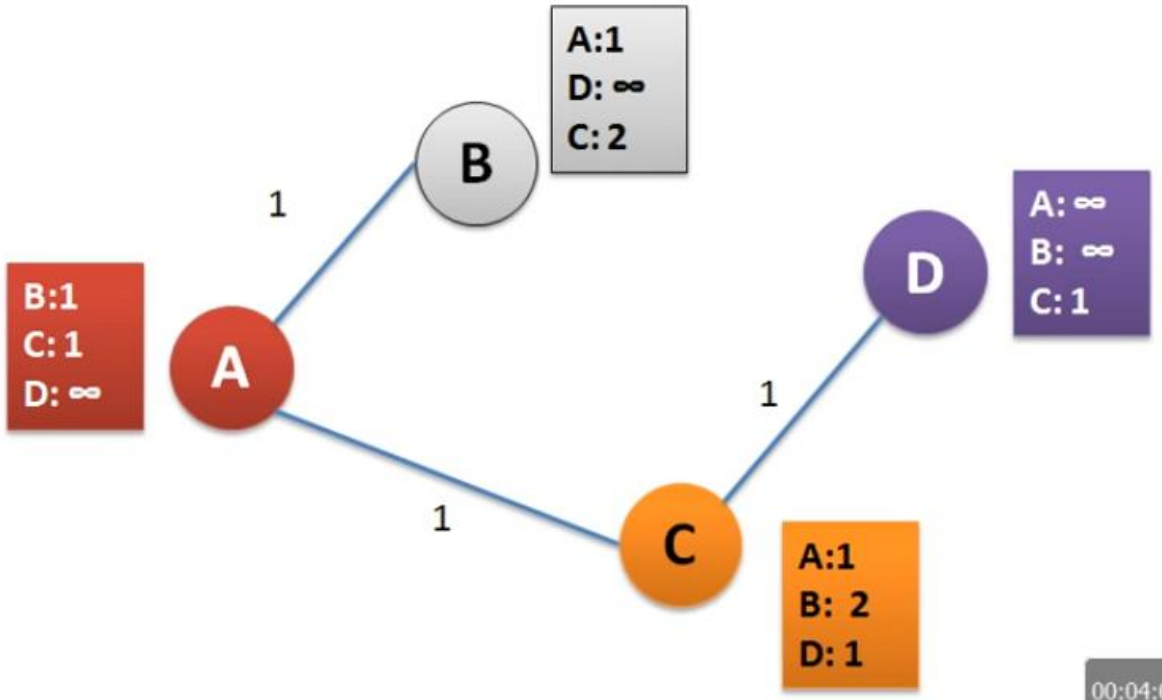


Distance Vector Routing



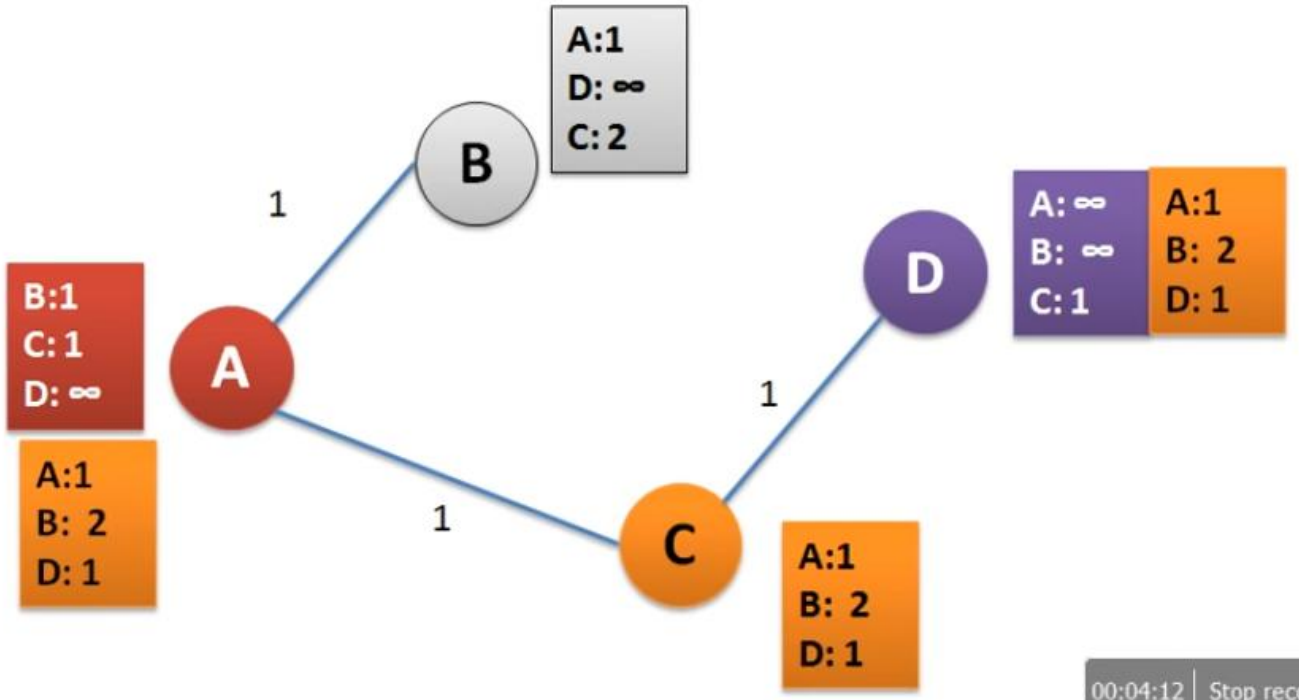
Distance Vector Routing

After A's broadcasting status of routing table. Now we will see Node C broadcasting.



Distance Vector Routing

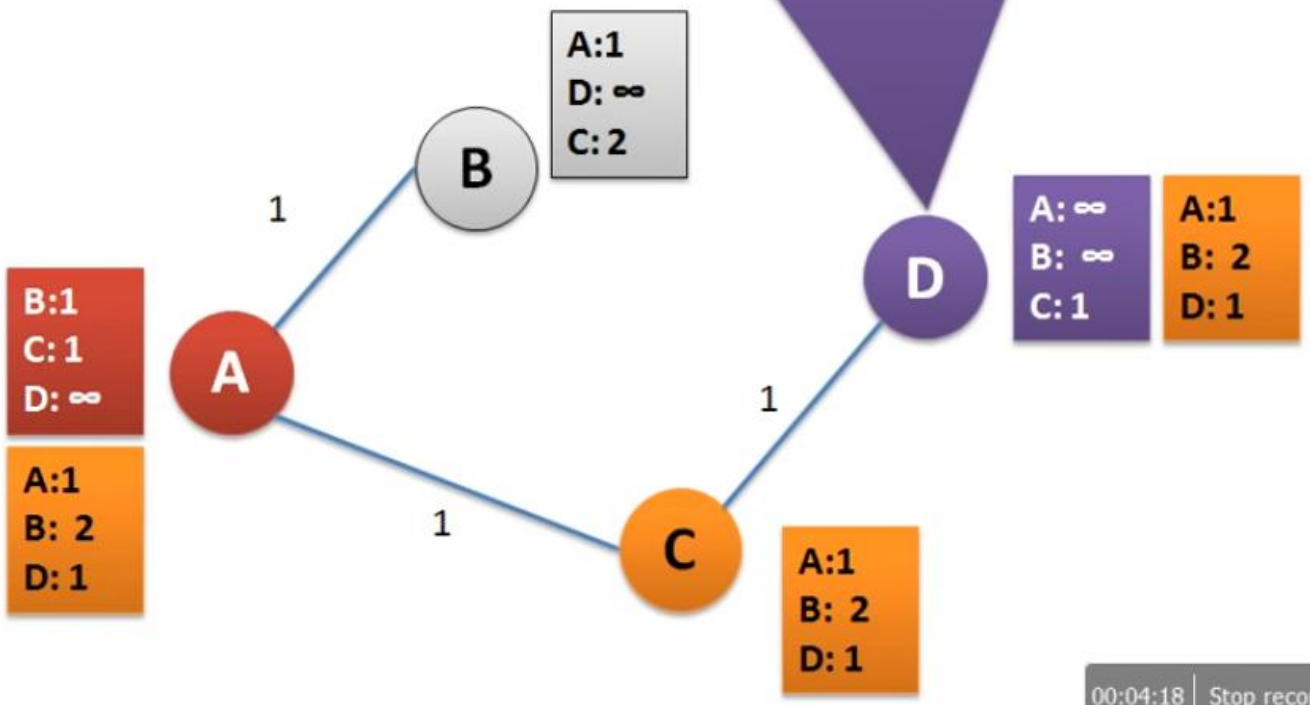
After A's broadcasting status of routing table. Now we will see Node C broadcasting.



Distance Vector Routing

D will check C's table. C find out B is reachable from C with distance 2. Distance between C and D is 1. So D can reach node B via C with distance 3.

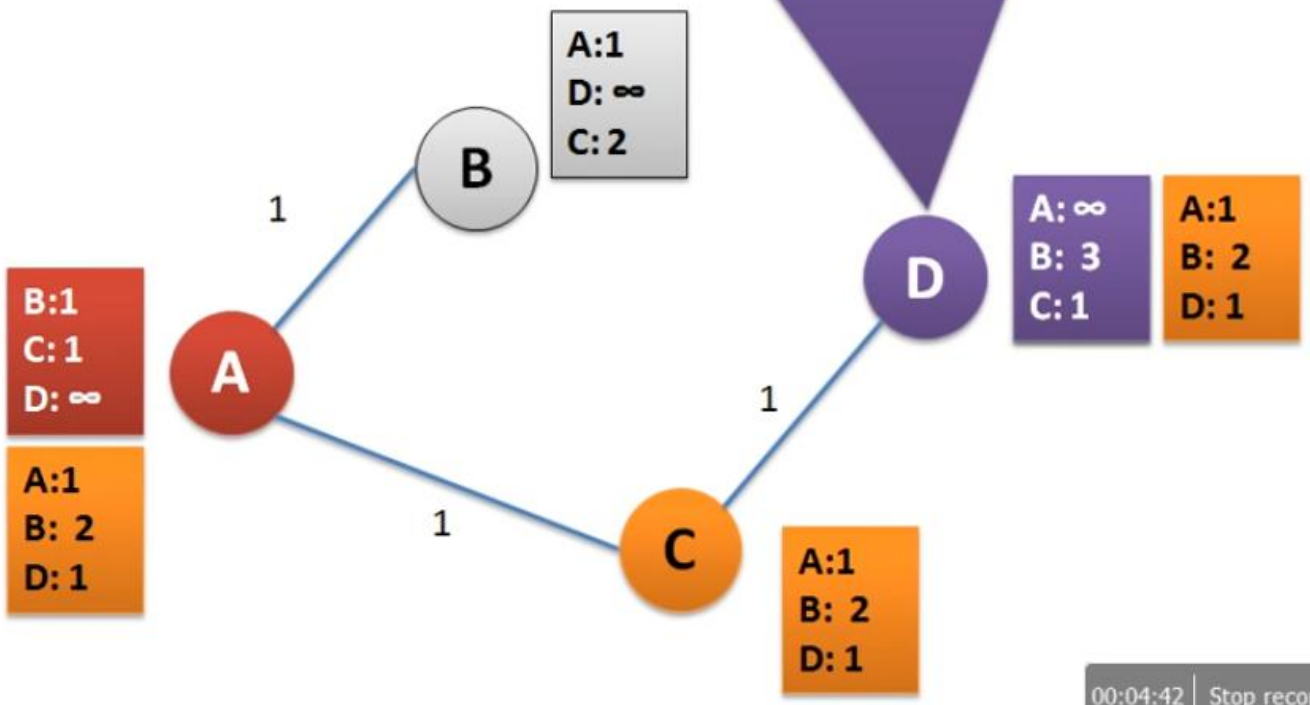
D also find out that it can reach A also via C with distance 2.



Distance Vector Routing

D will check C's table. C find out B is reachable from C with distance 2. Distance between C and D is 1. So D can reach node B via C with distance 3.

D also find out that it can reach A also via C with distance 2.

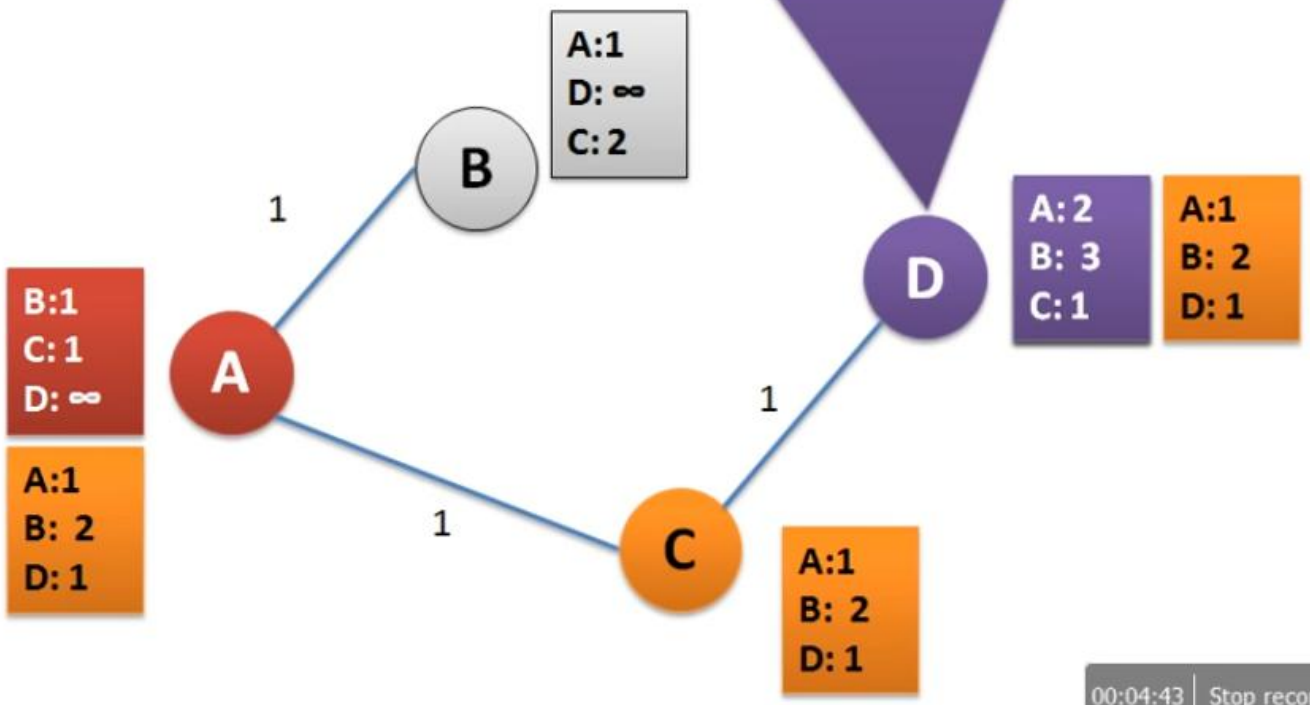


00:04:42 | Stop recording

Distance Vector Routing

D will check C's table. C find out B is reachable from C with distance 2. Distance between C and D is 1. So D can reach node B via C with distance 3.

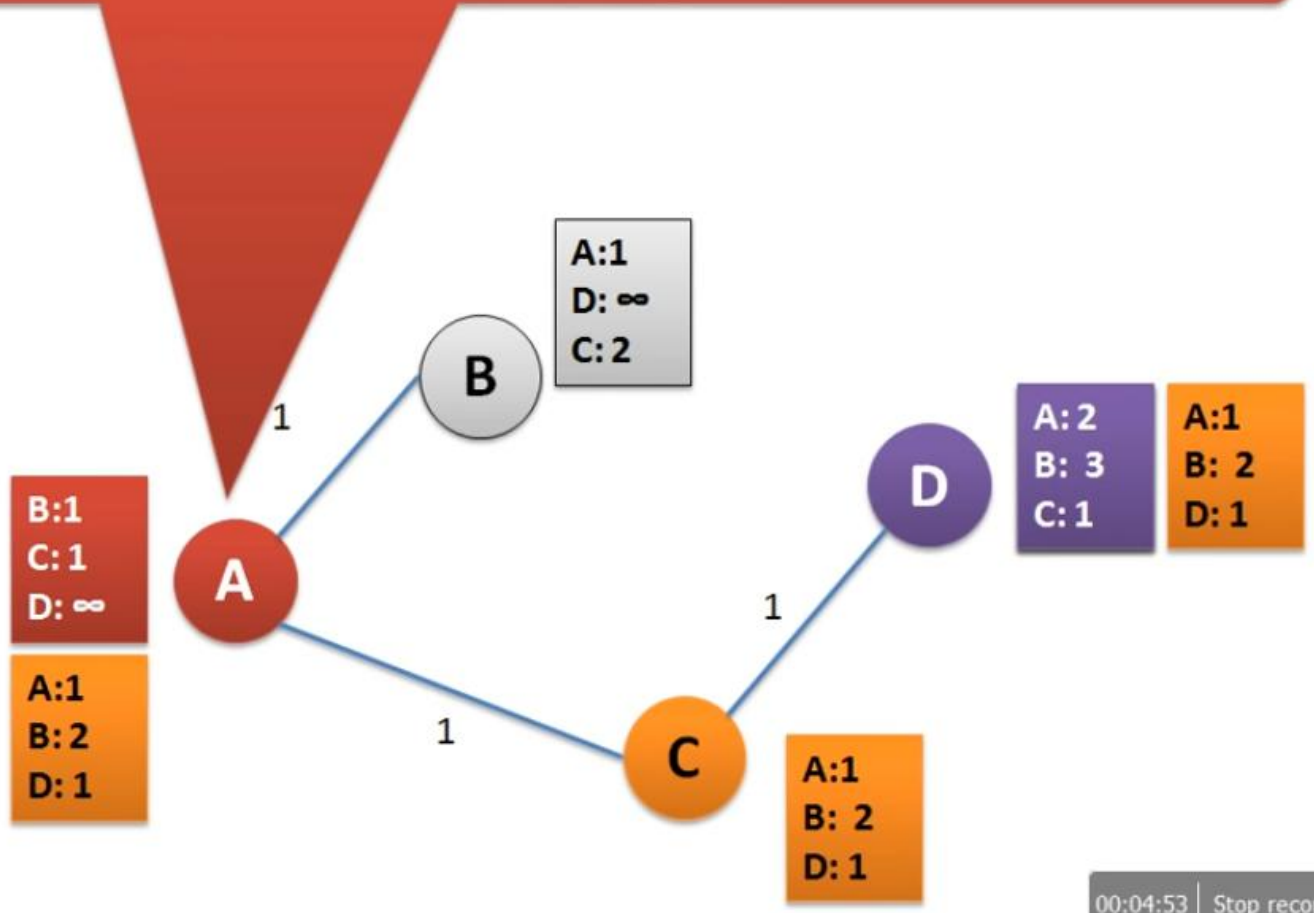
D also find out that it can reach A also via C with distance 2.



00:04:43 | Stop recording

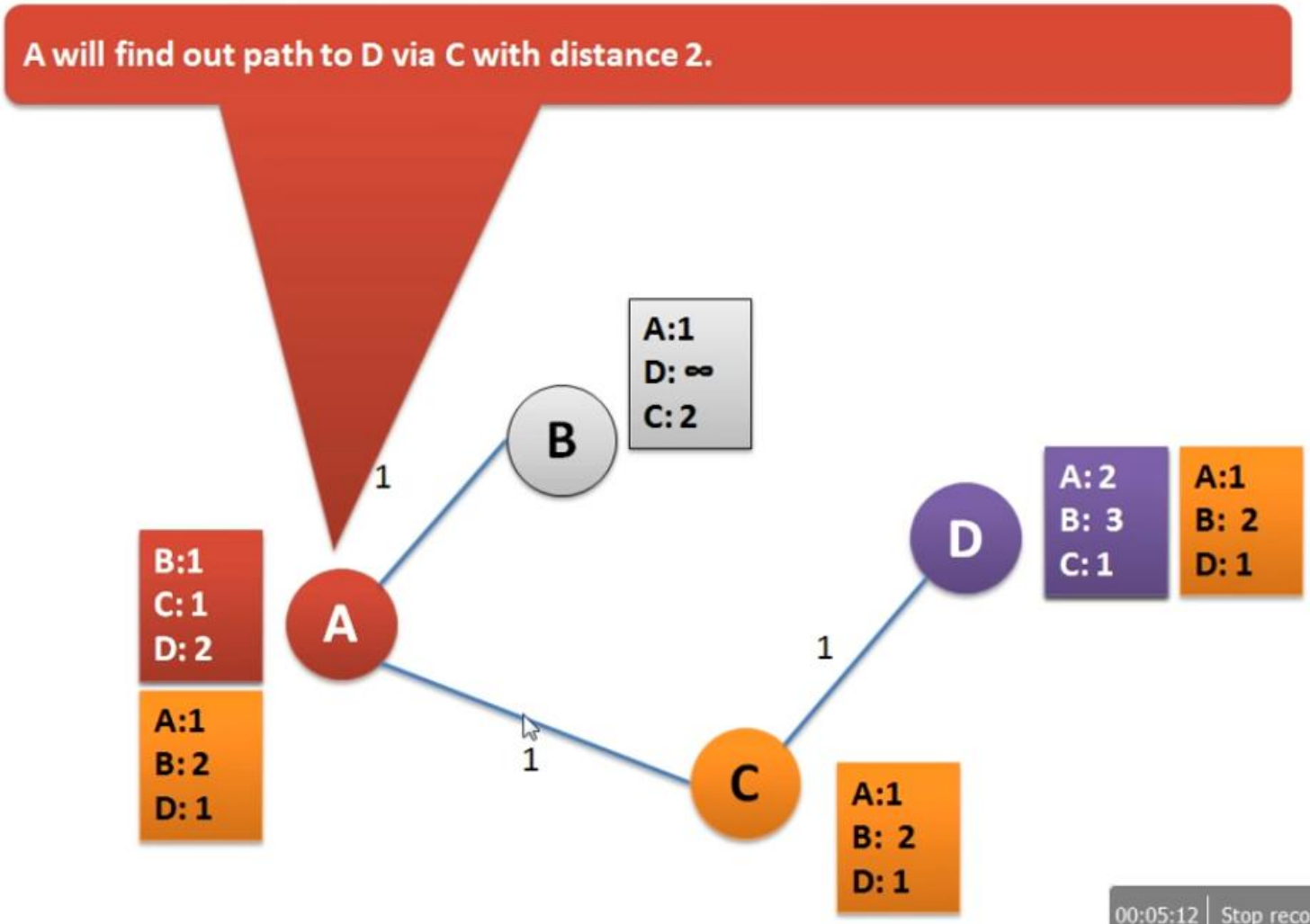
Distance Vector Routing

A will find out path to D via C with distance 2.



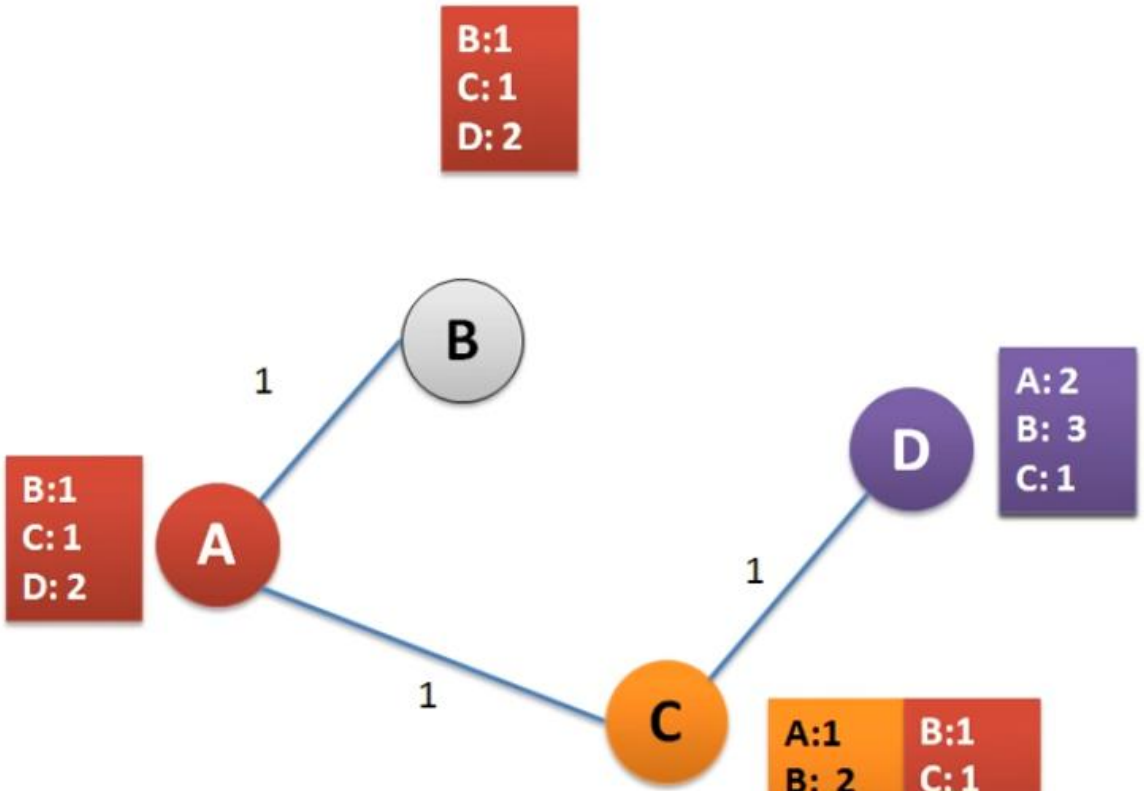
00:04:53 | Stop recording

Distance Vector Routing

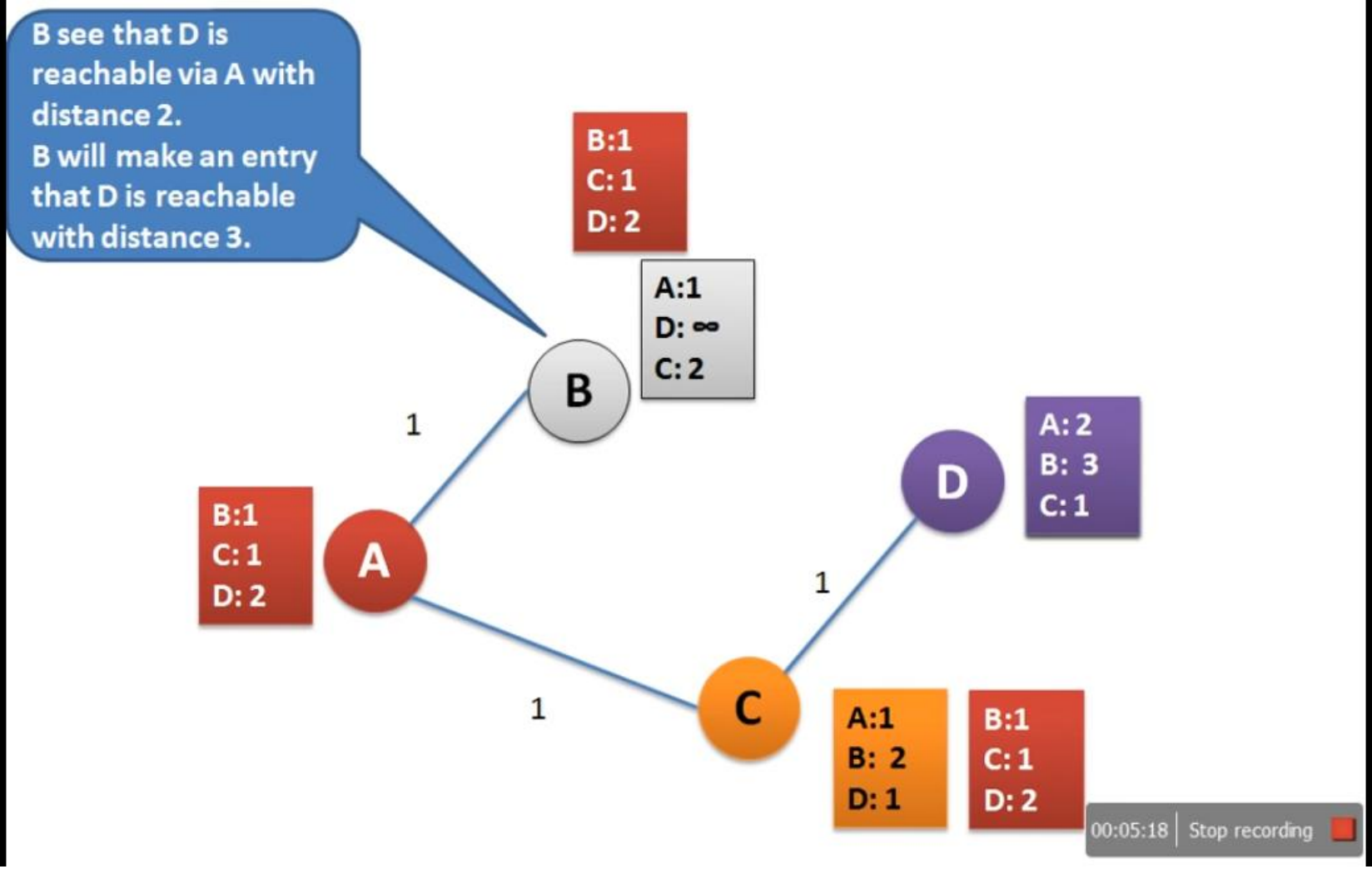


Distance Vector Routing

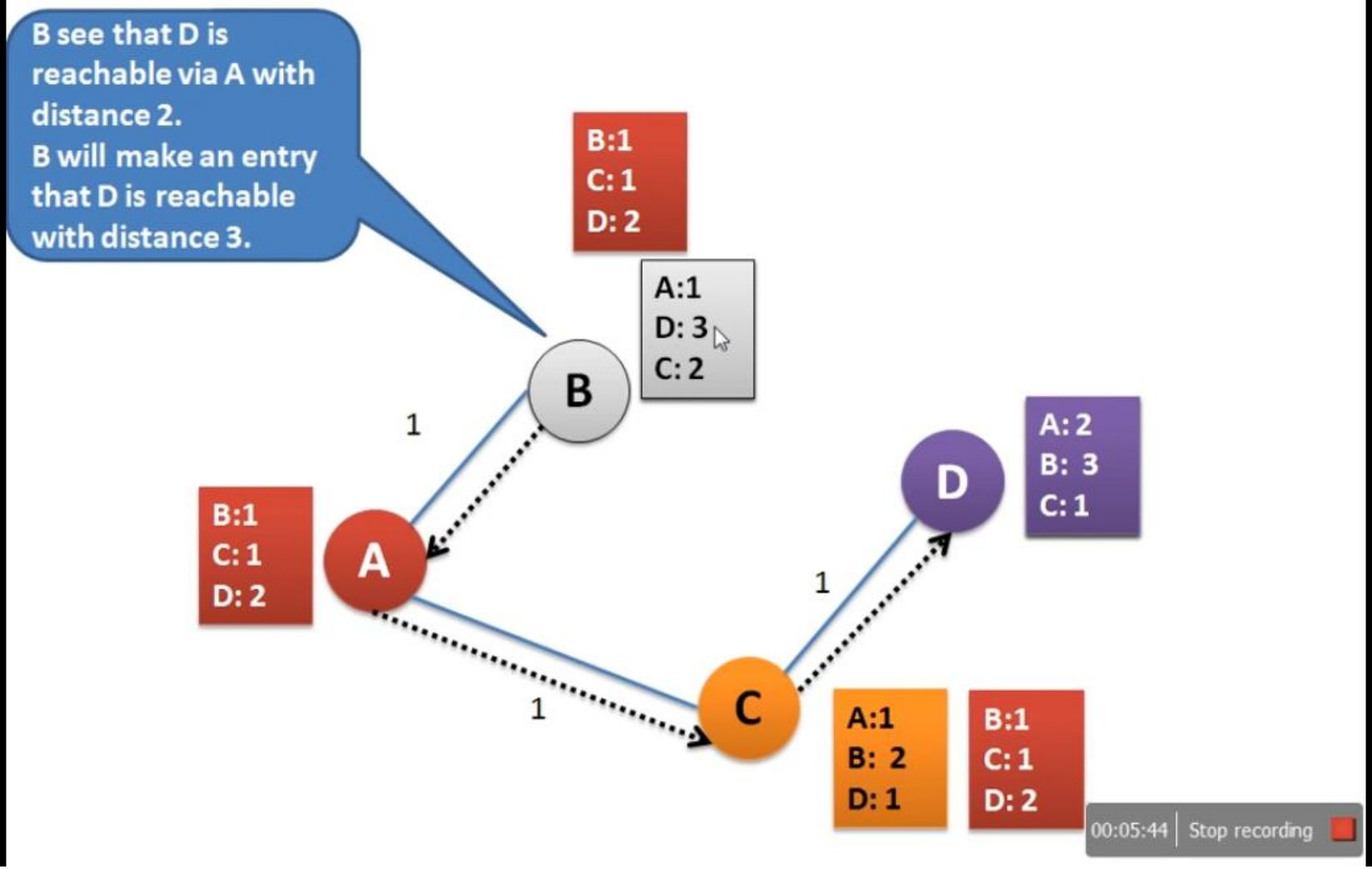
Each node periadically broadcast it's routing table to it's neighbour.



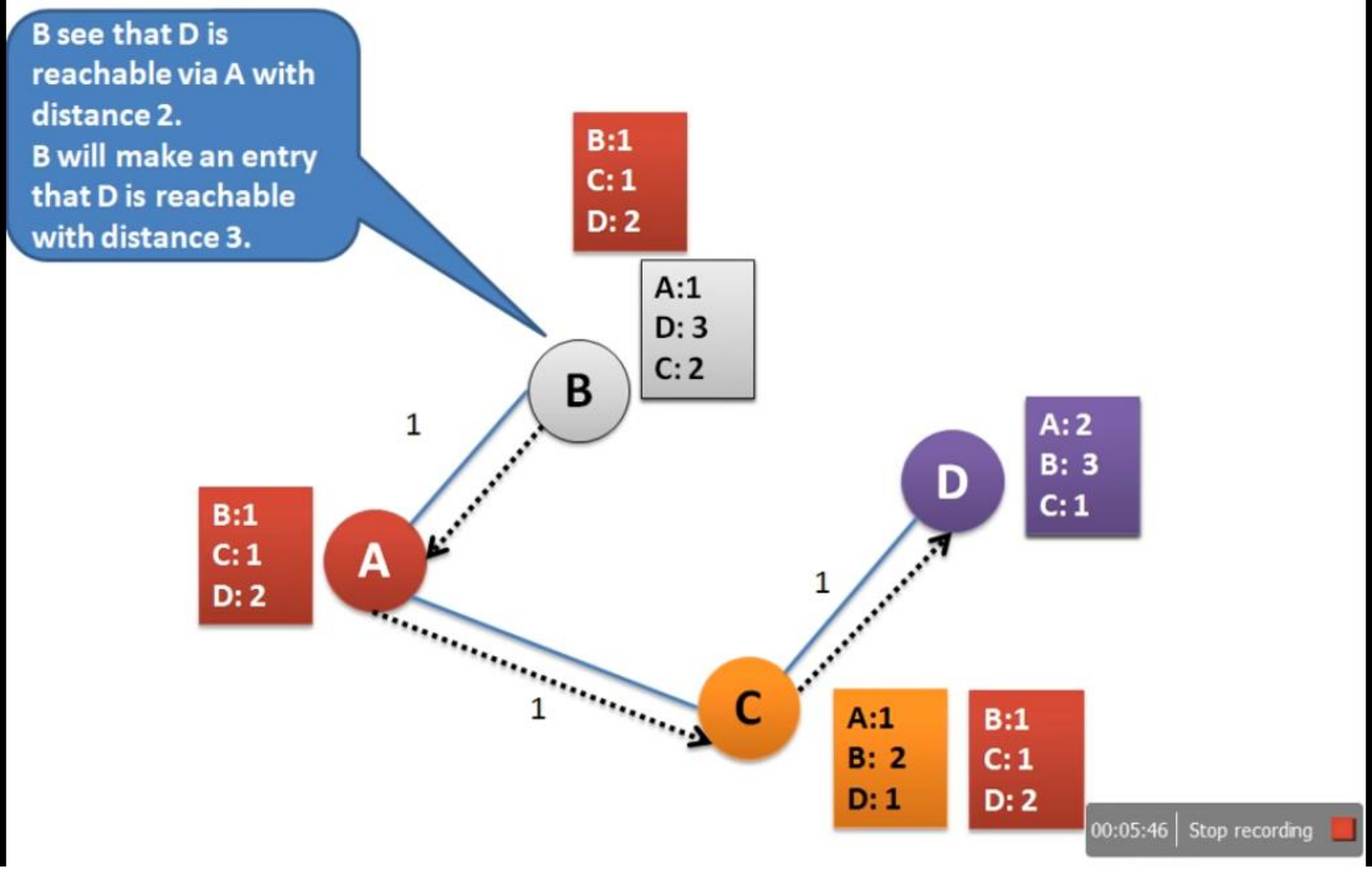
Distance Vector Routing



Distance Vector Routing

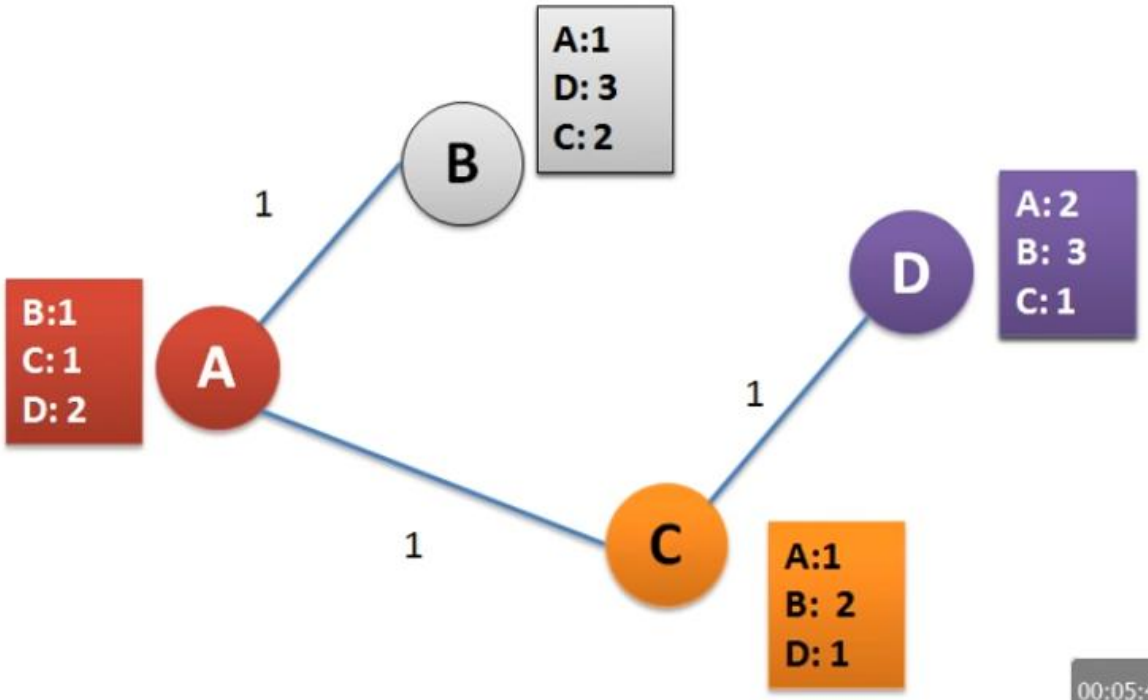


Distance Vector Routing



Distance Vector Routing

Each node also need to store direction. i.e. Node B can reach Node D via A.
Node B has to store this information.



Distance Vector Routing-Routing

Every node sends a message to its directly connected neighbors containing its personal list of distance. (for example, A sends its information to its neighbors B and C.)

If any of the recipients of the information from A find that A is advertising a path shorter than the one they currently know about, they update their list to give the new path length and note that they should send packets for that destination through A. (node B learns from A that node D can be reached at a cost of 1; B also knows it can reach A at a cost of 1, so it adds these to get the cost of reaching D by means of A. B records that it can reach D at a cost of 3 by going through A.)

After every node has exchanged a few updates with its directly connected neighbors, all nodes will know the least-cost path to all the other nodes.

In addition to updating their list of distances when they receive updates, the nodes need to keep track of which node told them about the path that they used to calculate the cost, so that they can create their forwarding table.

Distance Vector Routing-Routing

Every node sends a message to its directly connected neighbors containing its personal list of distance. (for example, A sends its information to its neighbors B and C.)

If any of the recipients of the information from A find that A is advertising a path shorter than the one they currently know about, they update their list to give the new path length and note that they should send packets for that destination through A. (node B learns from A that node D can be reached at a cost of 1; B also knows it can reach A at a cost of 1, so it adds these to get the cost of reaching D by means of A. B records that it can reach D at a cost of 3 by going through A.)

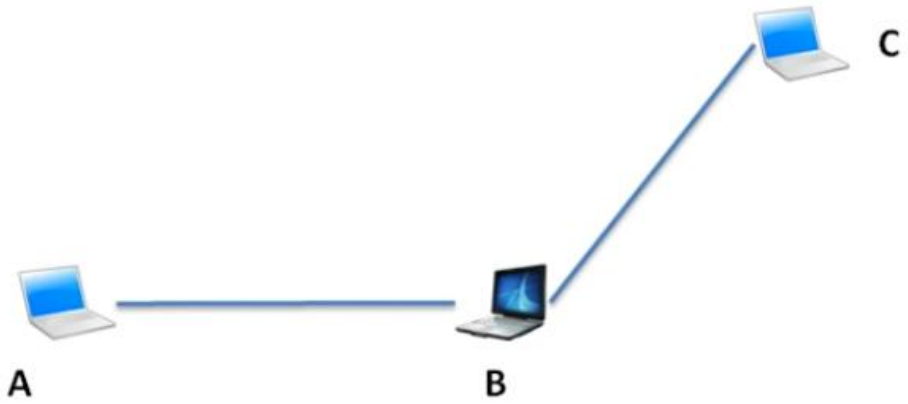
After every node has exchanged a few updates with its directly connected neighbors, all nodes will know the least-cost path to all the other nodes.

In addition to updating their list of distances when they receive updates, the nodes need to keep track of which node told them about the path that they used to calculate the cost, so that they can create their forwarding table.

Periodic Update

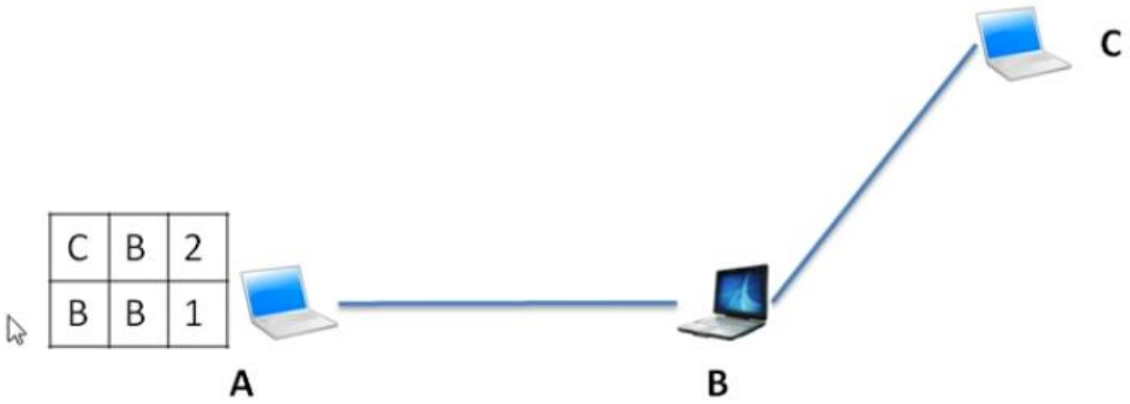
Distance Vector Routing-Routing table

Destination	Next Hop	Distance(hopcount)



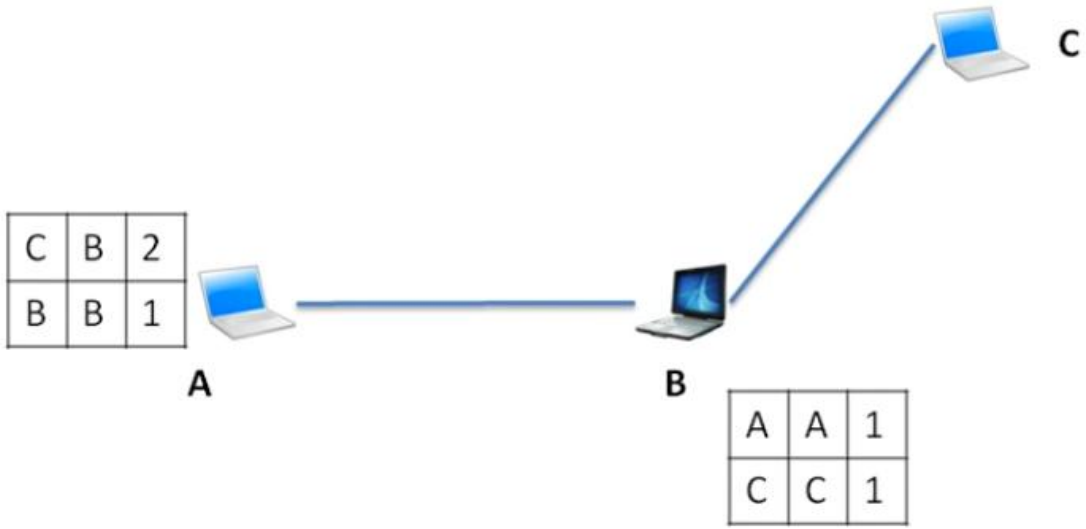
Distance Vector Routing-Routing table

Destination	Next Hop	Distance(hopcount)



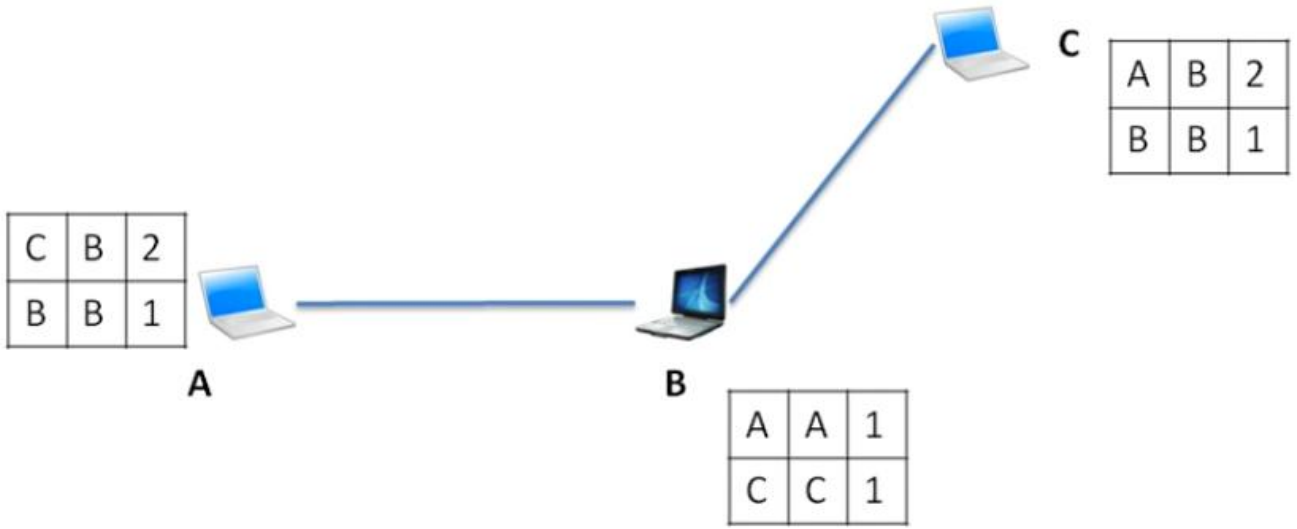
Distance Vector Routing-Routing table

Destination	Next Hop	Distance(hopcount)

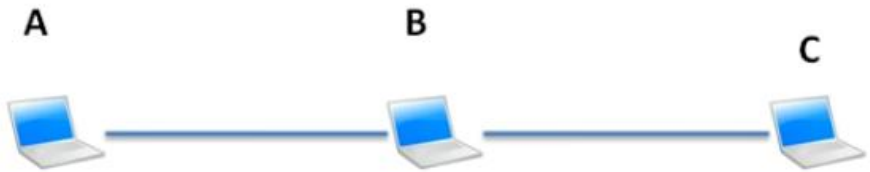


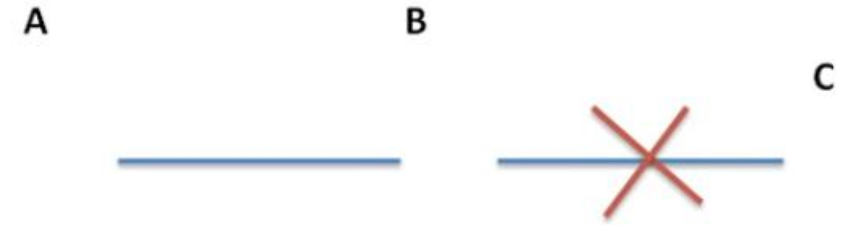
Distance Vector Routing-Routing table

Destination	Next Hop	Distance(hopcount)

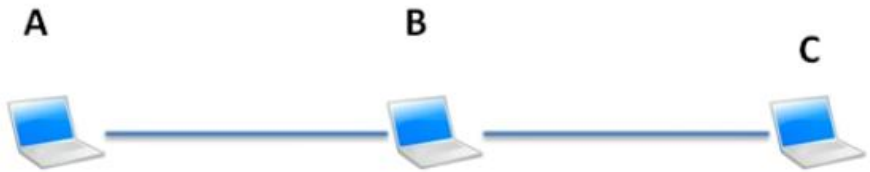


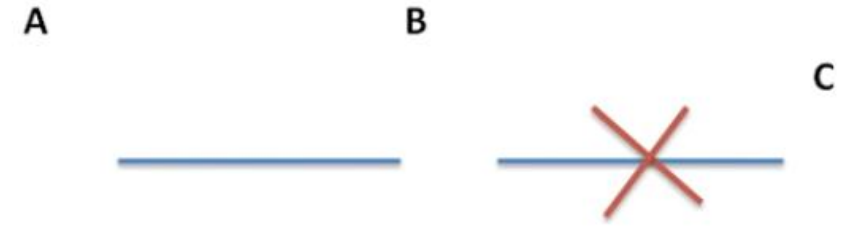
Distance Vector Routing-Count to infinity problem



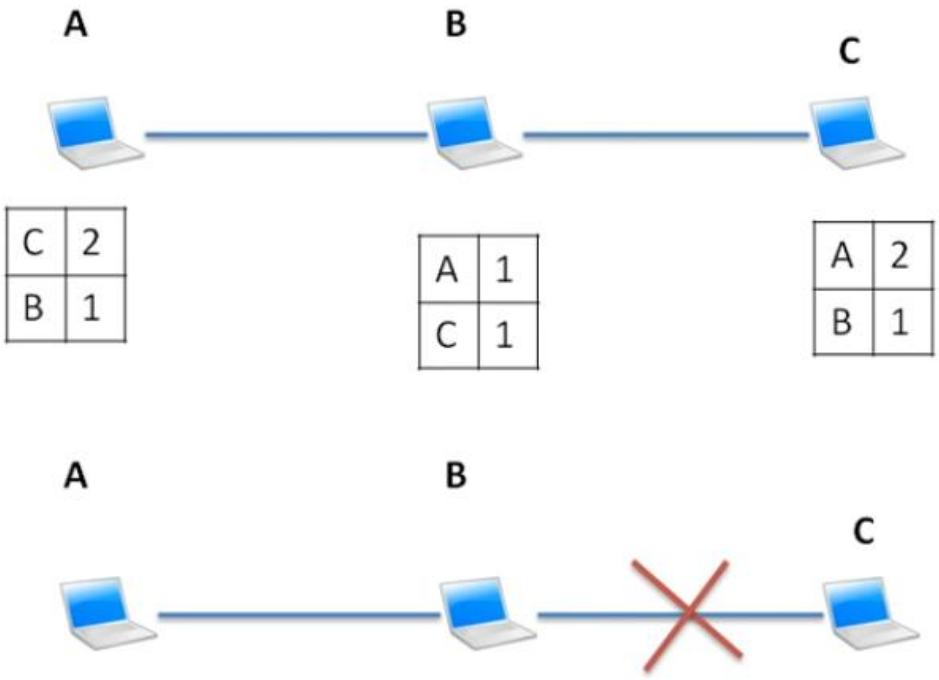


Distance Vector Routing-Count to infinity problem

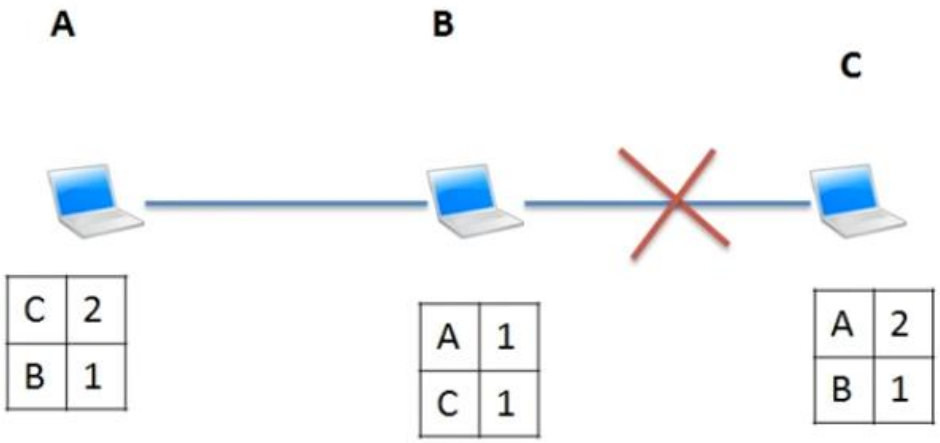




Distance Vector Routing-Count to infinity problem

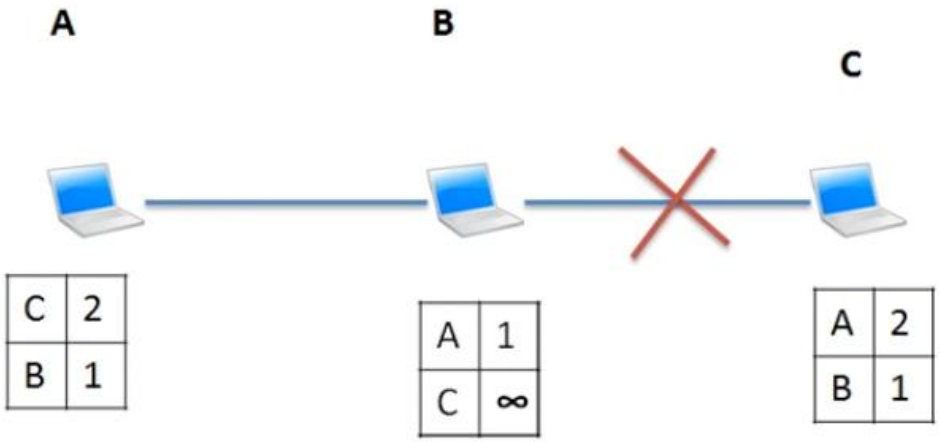


Distance Vector Routing-Count to infinity problem



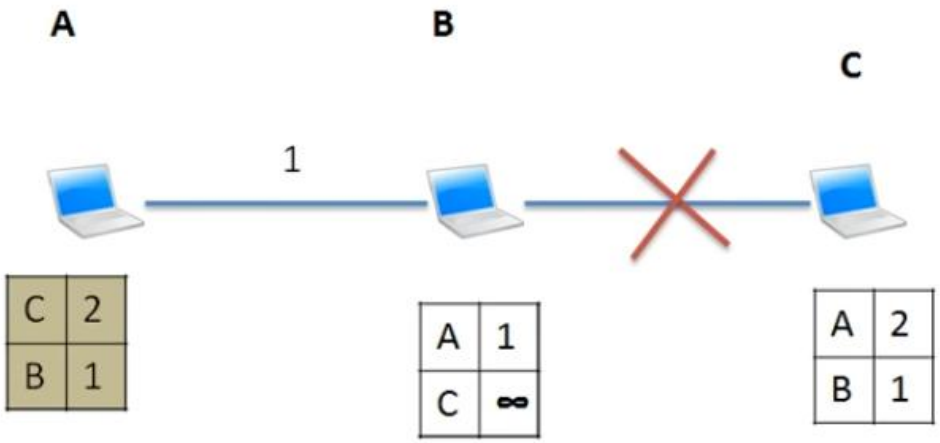
When B don't receive any update from C then B would know that C is disconnected.
B will set it's distance for C as Infinity.

Distance Vector Routing-Count to infinity problem



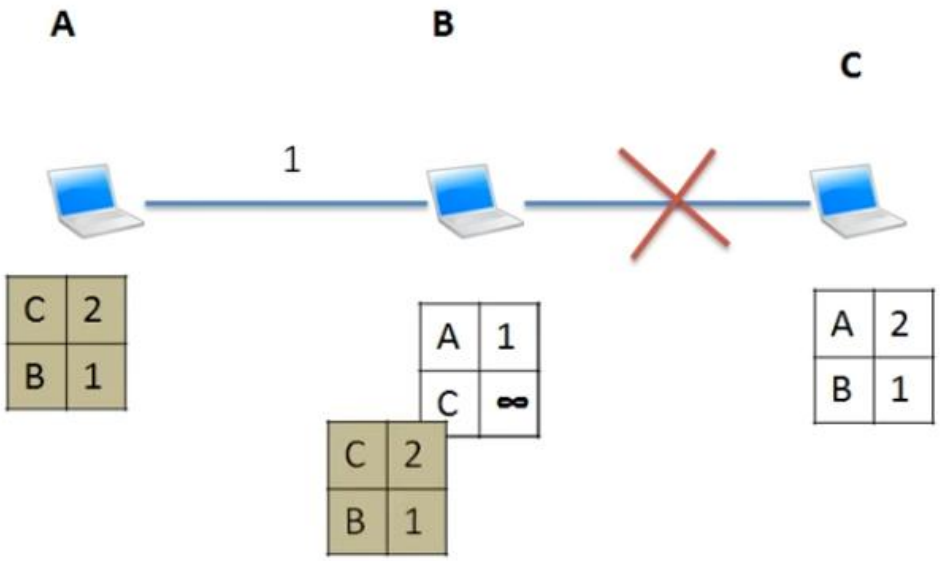
When B don't receive any update from C then B would know that C is disconnected.
B will set it's distance for C as Infinity.

Distance Vector Routing-Count to infinity problem



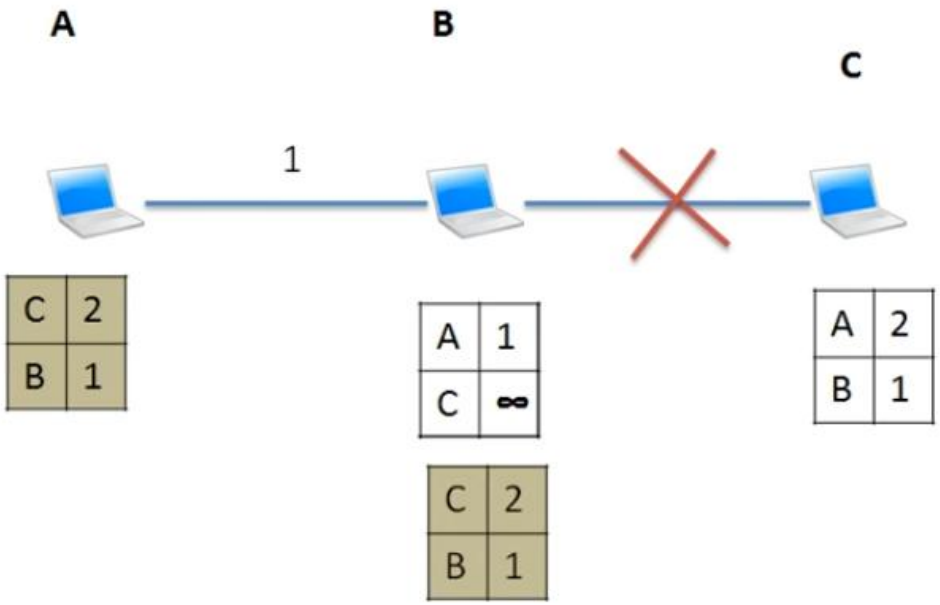
B from A's table will find out that C can be reachable from A with distance 3.
B can reach A with distance 1 and from A, C is reachable with distance 2.

Distance Vector Routing-Count to infinity problem



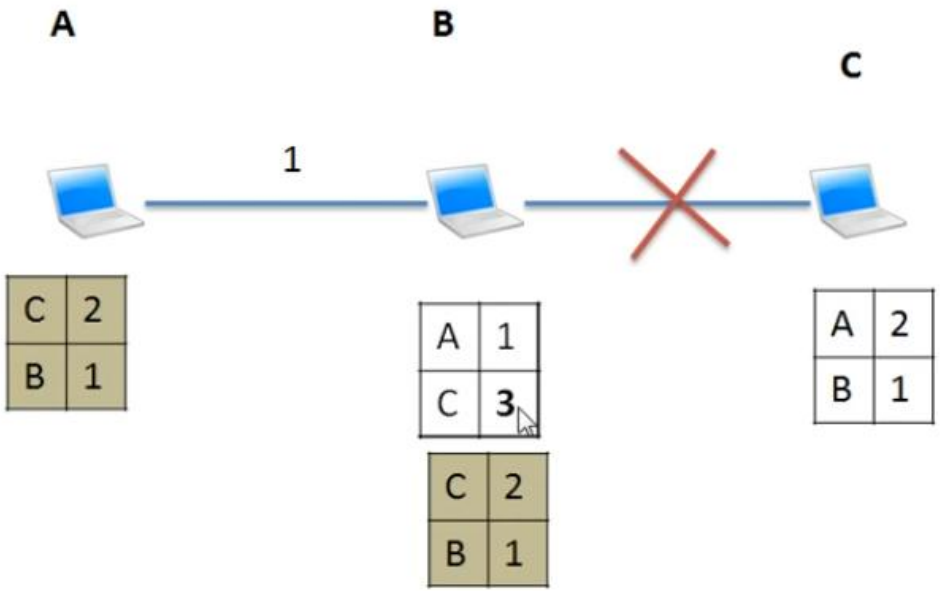
B from A's table will find out that C can be reachable from A with distance 3.
B can reach A with distance 1 and from A, C is reachable with distance 2.

Distance Vector Routing-Count to infinity problem



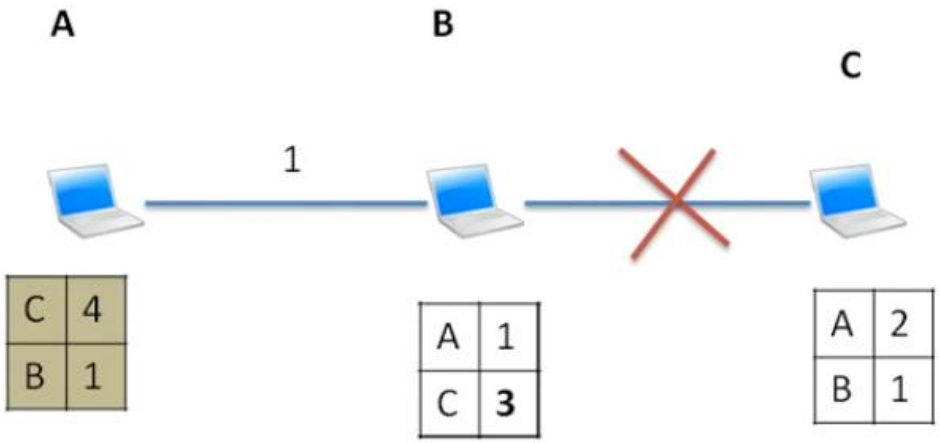
B from A's table will find out that C can be reachable from A with distance 3.
B can reach A with distance 1 and from A, C is reachable with distance 2.

Distance Vector Routing-Count to infinity problem



B from A's table will find out that C can be reachable from A with distance 3.
B can reach A with distance 1 and from A, C is reachable with distance 2.

Distance Vector Routing-Count to infinity problem



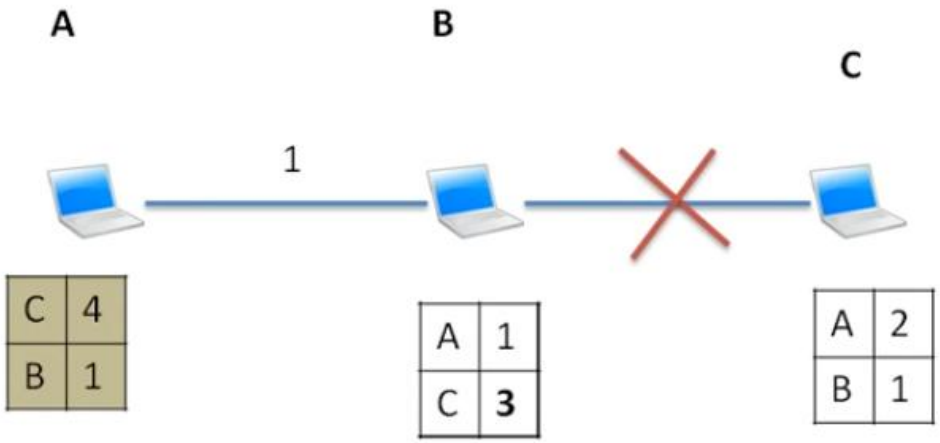
When B send it's table to A. A will update it's distance for Node C. Because A is reaching C via B.

A to C distance = A to B distance + B to C distance

= 1 + 3

= 4

Distance Vector Routing-Count to infinity problem



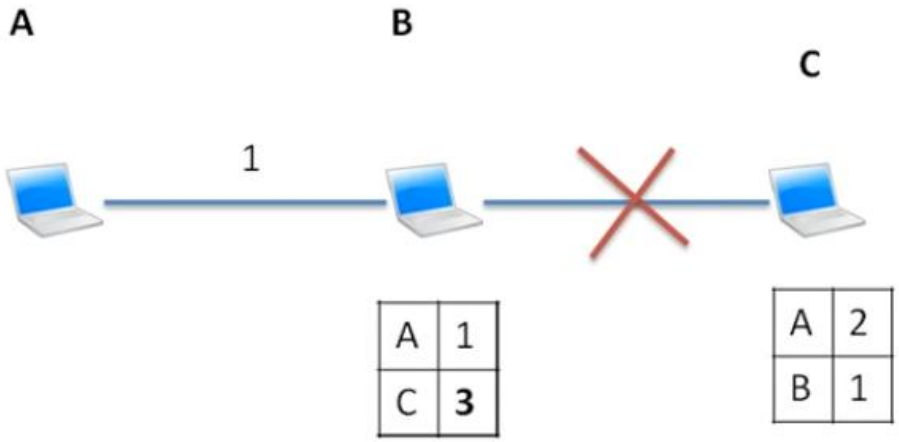
When B send it's table to A. A will update it's distance for Node C. Because A is reaching C via B.

A to C distance = A to B distance + B to C distance

= 1 + 3

= 4

Distance Vector Routing-Count to infinity problem



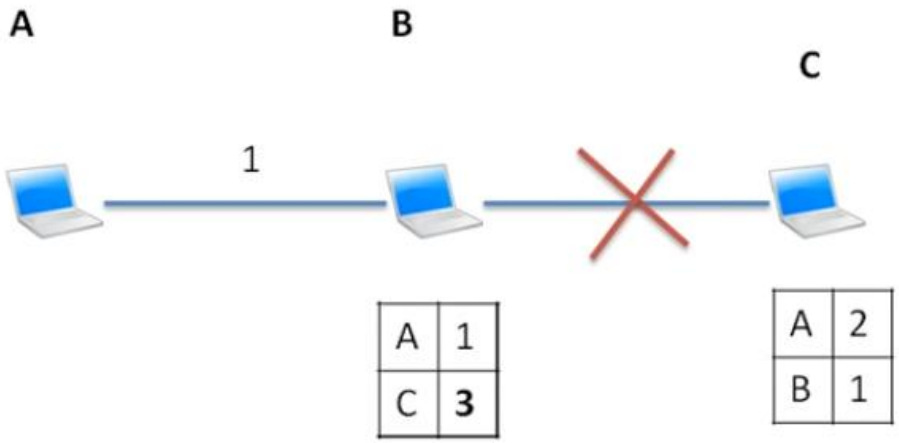
When B send it's table to A. A will update it's distance for Node C. Because A is reaching C via B.

A to C distance = A to B distance + B to C distance

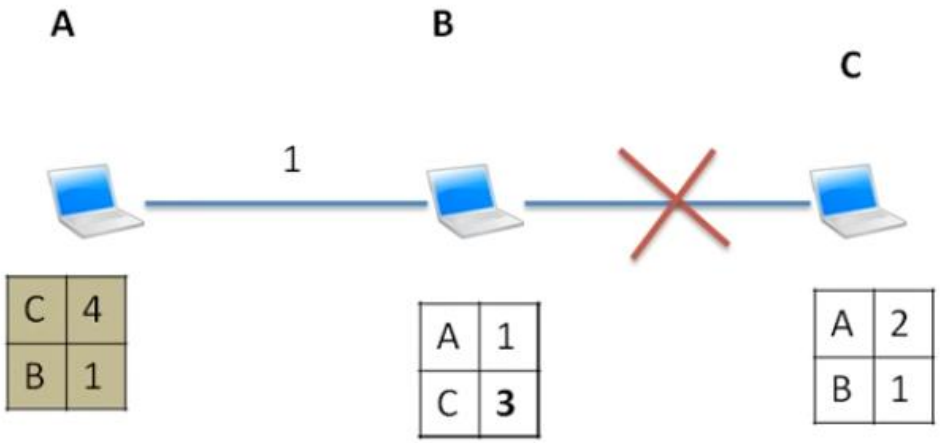
= 1 + 3

= 4

Distance Vector Routing-Count to infinity problem



Distance Vector Routing-Count to infinity problem



When B send it's table to A. A will update it's distance for Node C. Because A is reaching C via B.

A to C distance = A to B distance + B to C distance

= 1 + 3

= 4

Destination Sequenced Distance Vector Routing

To solve Distance vector problems, Destination sequence number added with every routing entry.

Destination Sequenced Distance Vector Routing

To solve Distance vector problems, Destination sequence number added with every routing entry.

A node will update it's table if it receive an updated route to destination. [If a node receive an route with higher sequence number]

Destination Sequenced Distance Vector Routing

To solve Distance vector problems, Destination sequence number added with every routing entry.

A node will update it's table if it receive an updated route to destination. [If a node receive an route with higher sequence number]

In DSDV routing table entry include
<destination, next hop, distance, sequence number>

Destination Sequenced Distance Vector Routing

To solve Distance vector problems, Destination sequence number added with every routing entry.

A node will update it's table if it receive an updated route to destination. [If a node receive an route with higher sequence number]

In DSDV routing table entry include
<destination, next hop, distance, sequence number>

Destination	Next Hop(Via)	Distance	Sequence number	Install Time

Destination Sequenced Distance Vector Routing

To solve Distance vector problems, Destination sequence number added with every routing entry.

A node will update it's table if it receive an updated route to destination. [If a node receive an route with higher sequence number]

In DSDV routing table entry include
<destination, next hop, distance, sequence number>

Destination	Next Hop(Via)	Distance	Sequence number	Install Time

Sequence number originated from destination. Ensures loop freeness.

Install Time when entry was made (used to delete stale entries from table)

Destination Sequenced Distance Vector Routing

- DSDV is Proactive (Table Driven)
  - Each node maintains routing information for all known destinations

Destination Sequenced Distance Vector Routing

- DSDV is Proactive (Table Driven)
  - Each node maintains routing information for all known destinations
  - Routing information must be updated periodically
  - Traffic overhead even if there is no change in network topology

DSDV-Route Advertisement

- Advertise to each neighbor own routing information
  - Destination Address
  - Metric = Number of Hops to Destination
  - Destination Sequence Number
- Rules to set sequence number information
  - On each advertisement increase own destination sequence number (use only even numbers)
  - If a node is no more reachable (timeout) increase sequence number of this node by 1 (odd sequence number) and set metric = ∞

DSDV-Route Tables

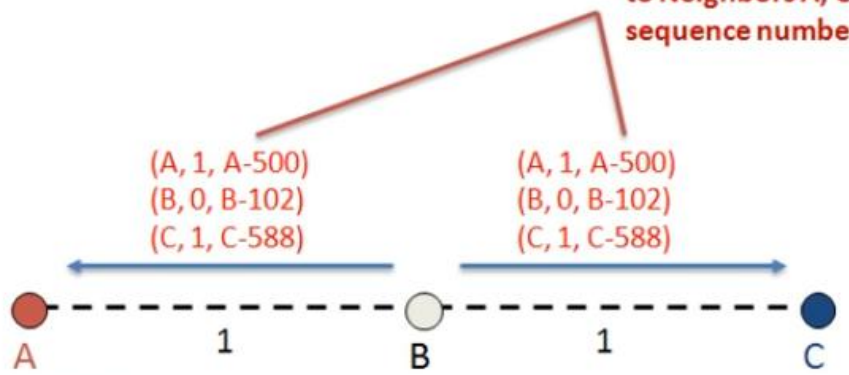
Dest.	Next	Metric	Seq
A	A	0	A-550
B	B	1	B-100
C	B	3	C-588

Dest.	Next	Metric	Seq
A	A	1	A-550
B	B	0	B-100
C	C	2	C-588

Dest.	Next	Metric	Seq.
A	B	1	A-550
B	B	2	B-100
C	C	0	C-588

DSDV-Route Advertisement

B increases Seq.Nr from 100 -> 102
B broadcasts routing information
to Neighbors A, C including destination
sequence numbers



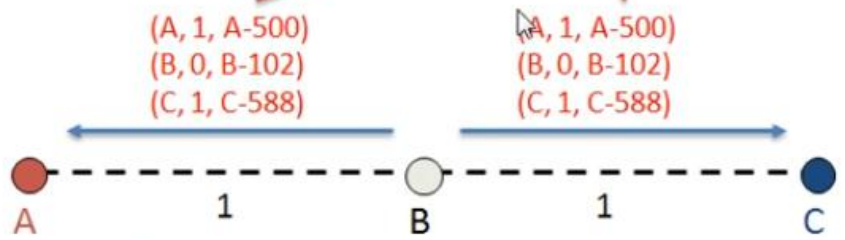
Dest.	Next	Metric	Seq
A	A	0	A-550
B	B	1	B-102
C	B	2	C-588

Dest.	Next	Metric	Seq
A	A	1	A-550
B	B	0	B-102
C	C	1	C-588

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-102
C	C	0	C-588

DSDV-Route Advertisement

B increases Seq.Nr from 100 -> 102
B broadcasts routing information
to Neighbors A, C including destination
sequence numbers



Dest.	Next	Metric	Seq
A	A	0	A-550
B	B	1	B-102
C	B	2	C-588

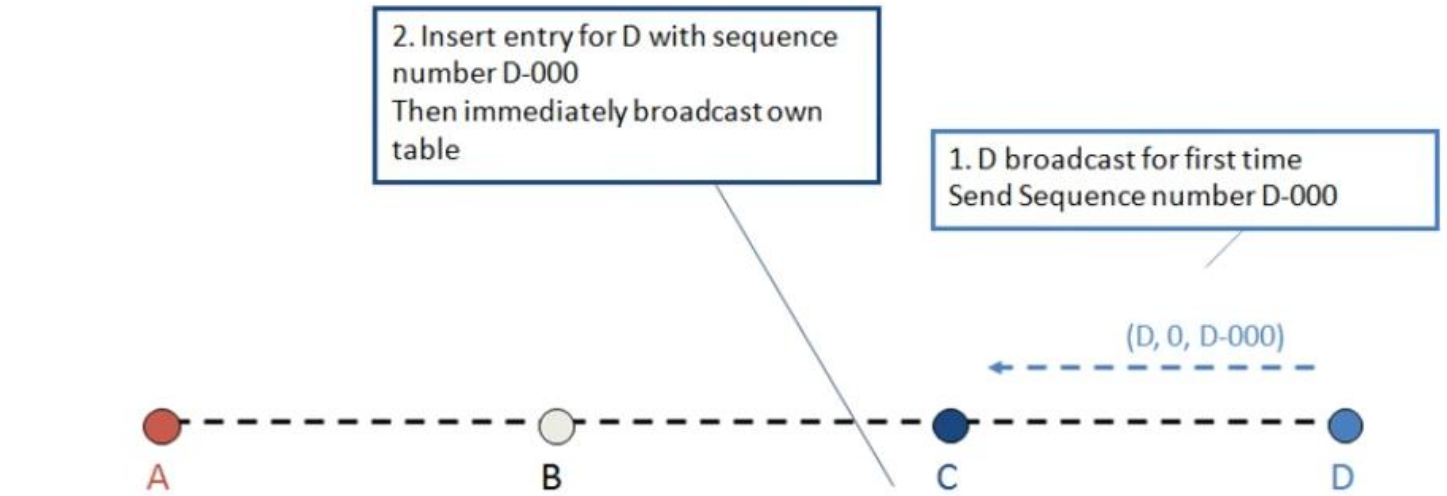
Dest.	Next	Metric	Seq
A	A	1	A-550
B	B	0	B-102
C	C	1	C-588

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-102
C	C	0	C-588

DSDV-Route Selection

- Update information is compared to own routing table
  - 1. Select route with higher destination sequence number (This ensure to use always newest information from destination)
  - 2. Select the route with better metric when sequence numbers are equal.

DSDV-New Node

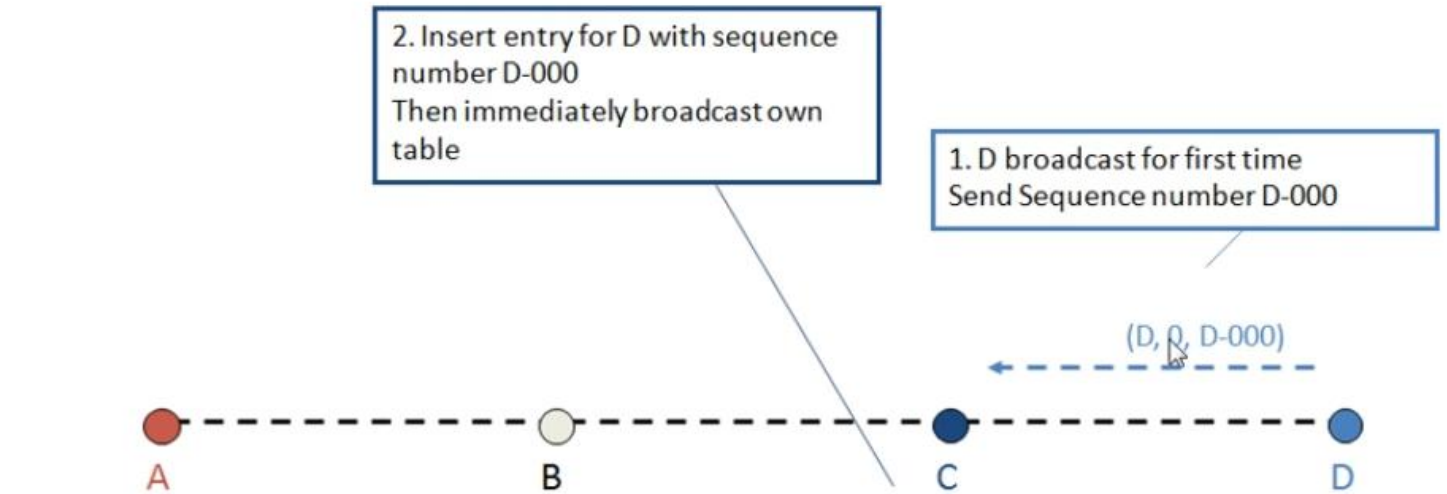


Dest.	Next	Metric	Seq.
A	A	0	A-550
B	B	1	B-104
C	B	2	C-590

Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-104
C	C	1	C-590

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-104
C	C	0	C-590
D	D	1	D-000

DSDV-New Node

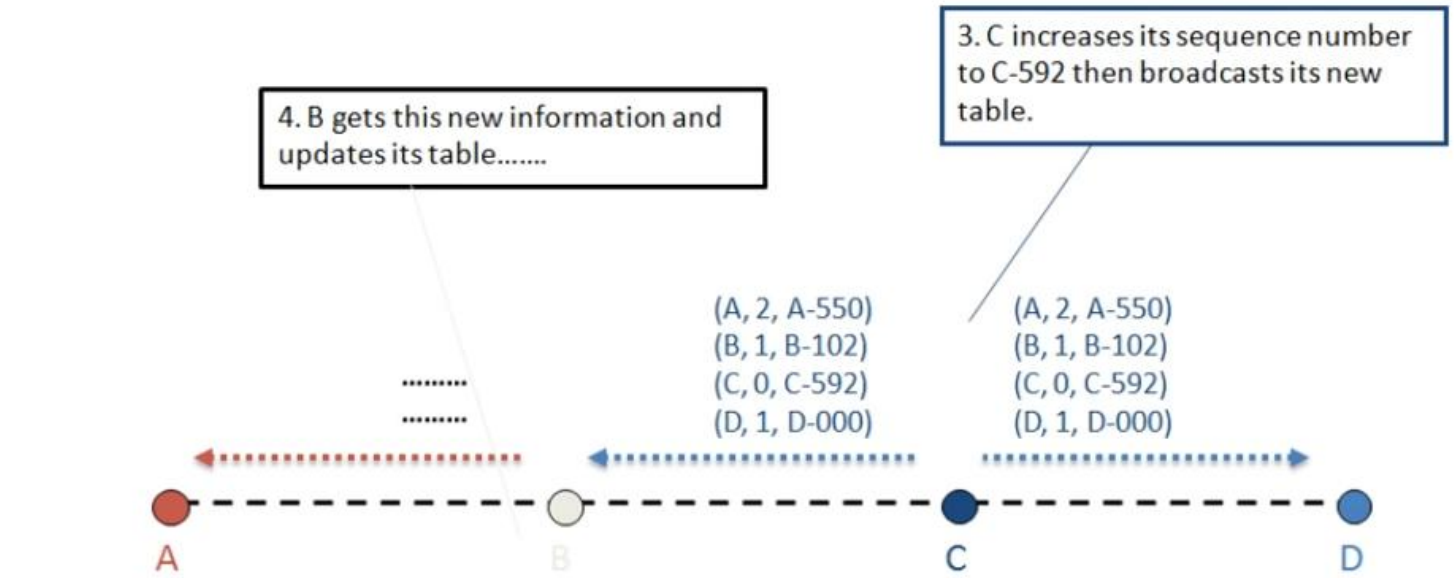


Dest.	Next	Metric	Seq.
A	A	0	A-550
B	B	1	B-104
C	B	2	C-590

Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-104
C	C	1	C-590

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-104
C	C	0	C-590
D	D	1	D-000

DSDV-New Node



Dest.	Next	Metric	Seq.
A	A	0	A-550
B	B	1	B-104
C	B	2	C-590

Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-102
C	C	1	C-592
D	C	2	D-000

Dest.	Next	Metric	Seq.
A	B	2	A-550
B	B	1	B-102
C	C	0	C-592
D	D	1	D-000

DSDV-No loop, No Count to Infinity Problem

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)

Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

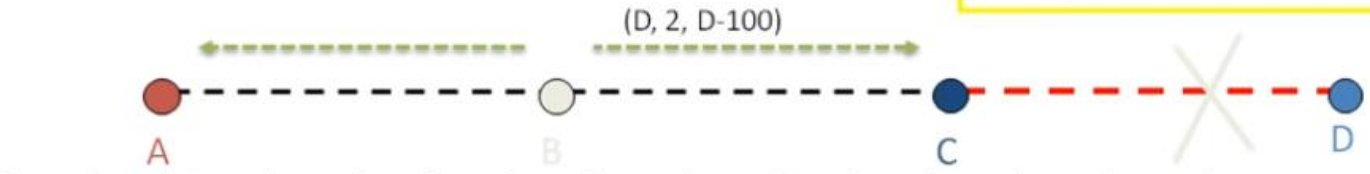
Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

DSDV-No loop, No Count to Infinity Problem

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)



Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

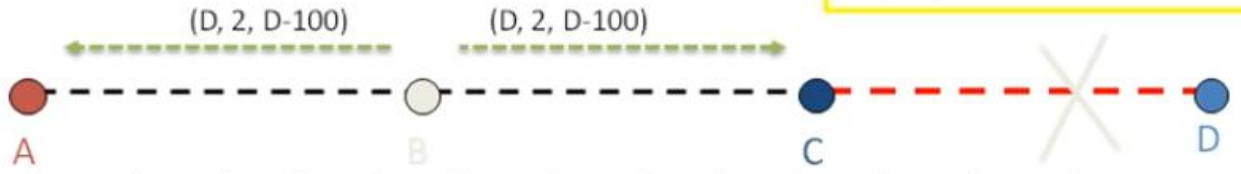
Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

DSDV-No loop, No Count to Infinity Problem

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)



Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

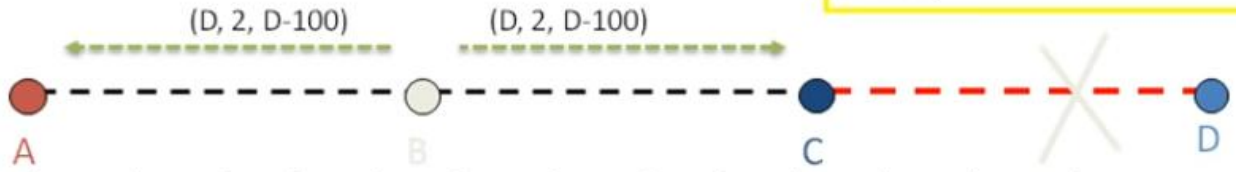
Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

DSDV-No loop, No Count to Infinity Problem

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)



Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

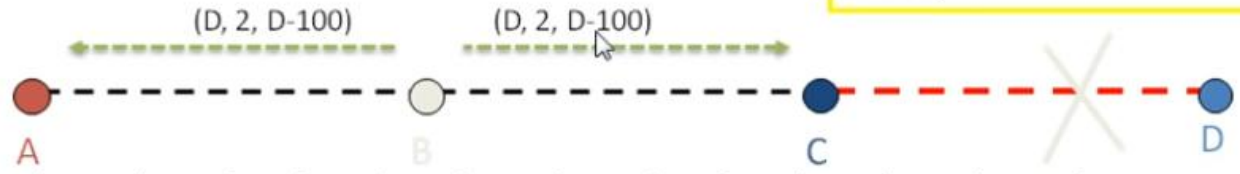
Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

DSDV-No loop, No Count to Infinity Problem

2. B does its broadcast
-> no affect on C (C knows that B has stale information because C has higher seq. number for destination D)
-> no loop -> no count to infinity

1. Node C detects broken Link:
-> Increase Seq. Nr. by 1
(only case where not the destination sets the sequence number -> odd number)

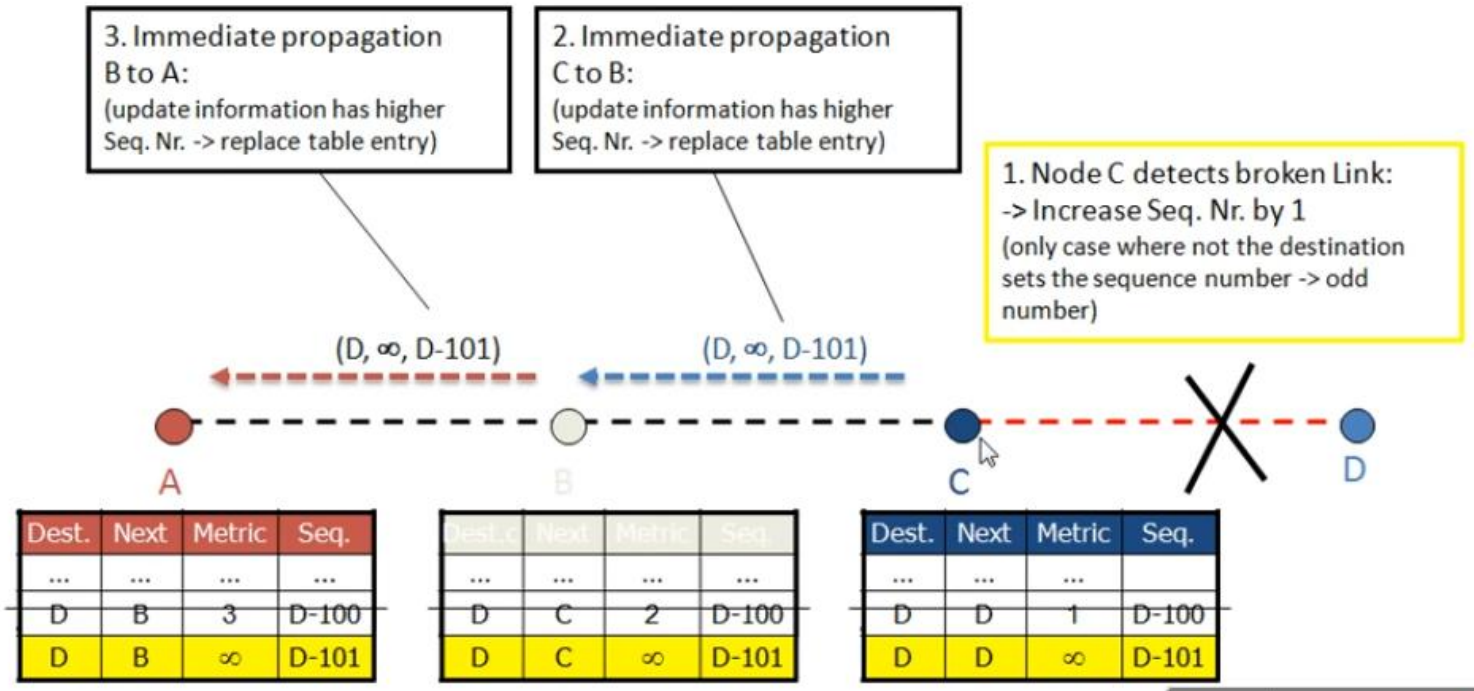


Dest.	Next	Metric	Seq.
...	...	...	
D	B	3	D-100

Dest.c	Next	Metric	Seq.
...	...	...	
D	C	2	D-100

Dest.	Next	Metric	Seq.
...	...	...	
D	D	∞	D-101

DSDV-Immediate Advertisement



DSDV-Immediate Advertisement