Ad-Hoc wireless networks

- A network of devices without any centralized access point or existing infrastructure
  - A network of laptops, cellular phones, PDAs, printers
  - WSNs

Routing protocols for Ad Hoc wireless networks

- The properties of the ad-hoc network routing protocol
  - Simple
  - Less storage space
  - Loop free
  - Short control message (Low overhead)
  - Less power consumption
  - Multiple disjoint routes
  - Fast rerouting mechanism

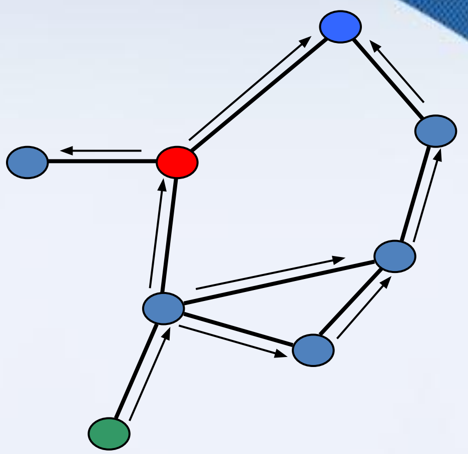
Routing protocols for Ad Hoc wireless networks

- Based on the routing information update mechanism, protocols can be classified into three:
  1) Proactive (table-driven) routing protocols
  2) Reactive (on-demand) routing protocol
  3) Hybrid routing protocols

Routing protocols for Ad Hoc wireless networks

Proactive or table-driven	**Reactive or on-demand**	**Hybrid**
Information about the network topology in each node • periodic exchanges • information usually flooded • path finding based on local information	Information about the network topology obtained when required • information acquired by connection processes • path finding based on acquired information	Merge the two schemes according to the topology • nodes within a region may use proactive routing • other nodes may use reactive routing

The Basics - Flooding

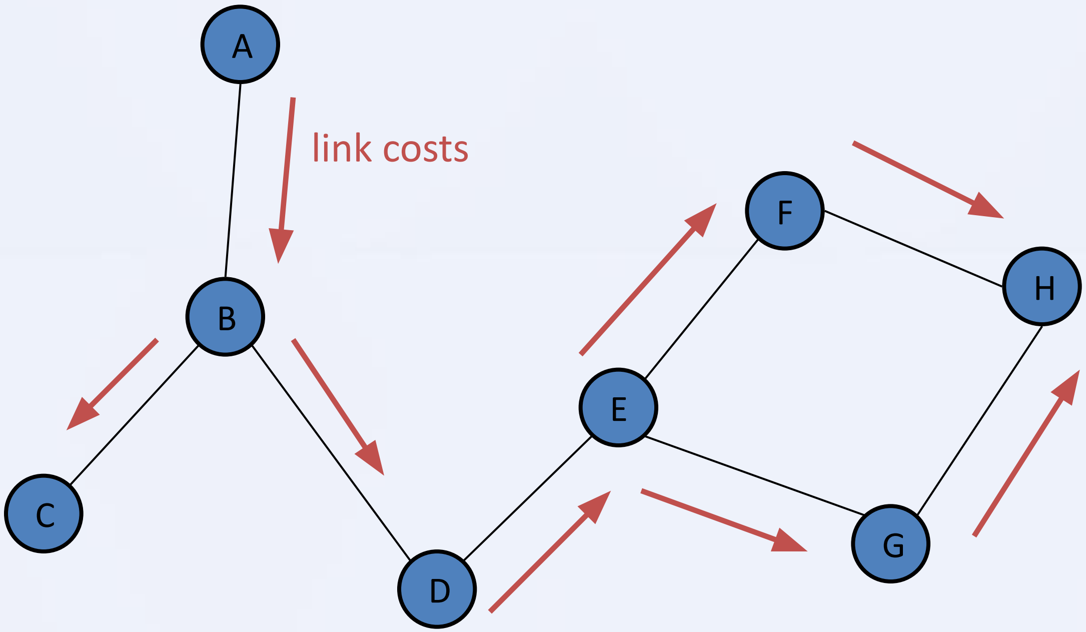


- Simplest of all routing protocols
- Send all info to everybody
  - If data not for you, send to all neighbors
- Robust
  - destination is *almost* guaranteed to receive data
- Resource Intensive
  - unnecessary traffic
  - load increases, network performance drops quickly

The Basic- Link State routing

- Like the shortest-path computation method
- Each node maintains a view of the network topology with a cost for each link
- Periodically broadcast link costs to its outgoing links to all other nodes such as flooding

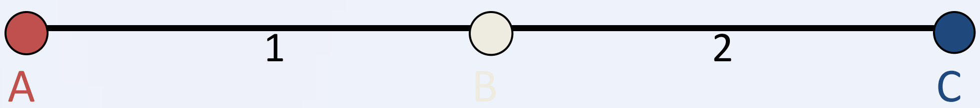
The Basic- Link State routing



The Basic- Distance Vector Routing

- Known also as Distributed Bellman-Ford or RIP (Routing Information Protocol)
- Every node maintains a routing table
  - all available destinations
  - the next node to reach to destination
  - the number of hops to reach the destination
- Periodically send table to all neighbors to maintain topology

The Basic- Distance Vector (Tables)



Dest.	Next	Metric	…
A	A	0	
B	B	1	
C	B	3	

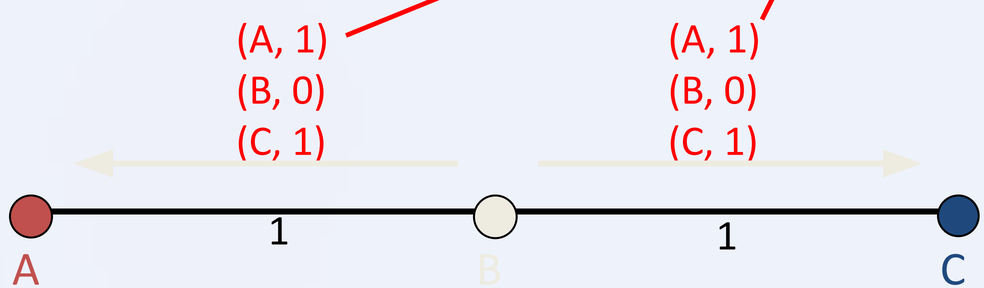
Dest.	Next	Metric	…
A	A	1	
B	B	0	
C	C	2	

Dest.	Next	Metric	…
A	B	3	
B	B	2	
C	C	0	

Distance Vector (Update)

B broadcasts the new routing information to his neighbors

Routing table is updated

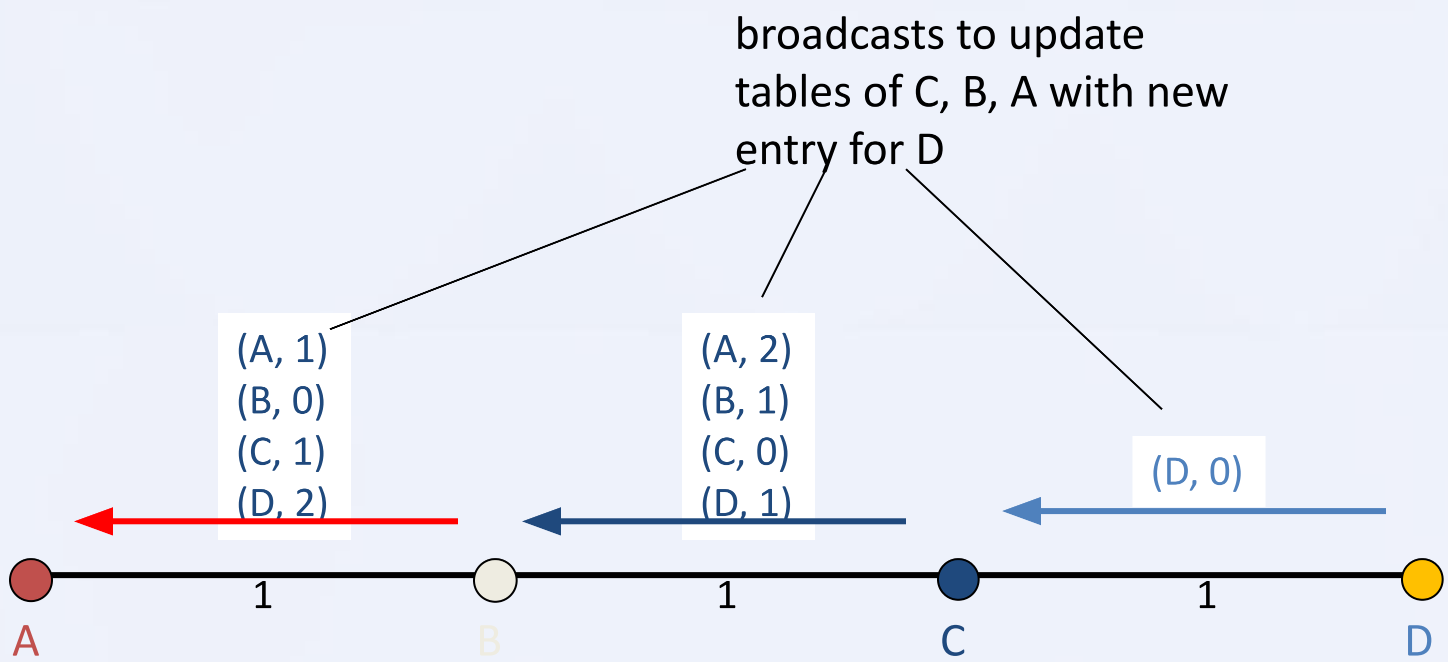


Dest.	Next	Metric	...
A	A	0	
B	B	1	
C	B	~~3~~ 2	

Dest.	Next	Metric	...
A	A	1	
B	B	0	
C	C	1	

Dest.	Next	Metric	...
A	B	~~3~~ 2	
B	B	1	
C	C	0	

Distance Vector (New Node)

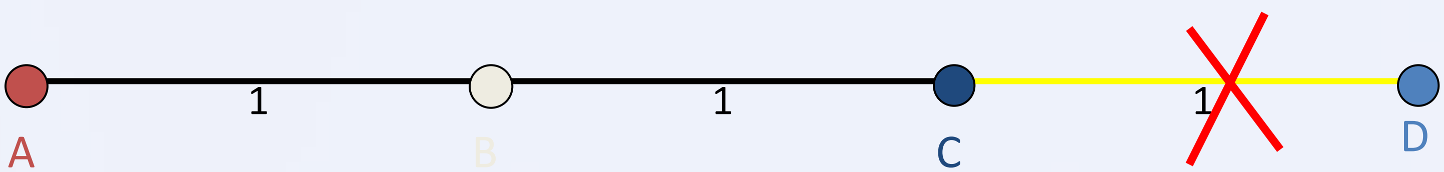


Dest.	Next	Metric	...
A	A	0	
B	B	1	
C	B	2	
D	B	3	

Dest.	Next	Metric	...
A	A	1	
B	B	0	
C	C	1	
D	C	2	

Dest.	Next	Metric	...
A	B	2	
B	B	1	
C	C	0	
D	D	1	

Distance Vector (Broken Link)

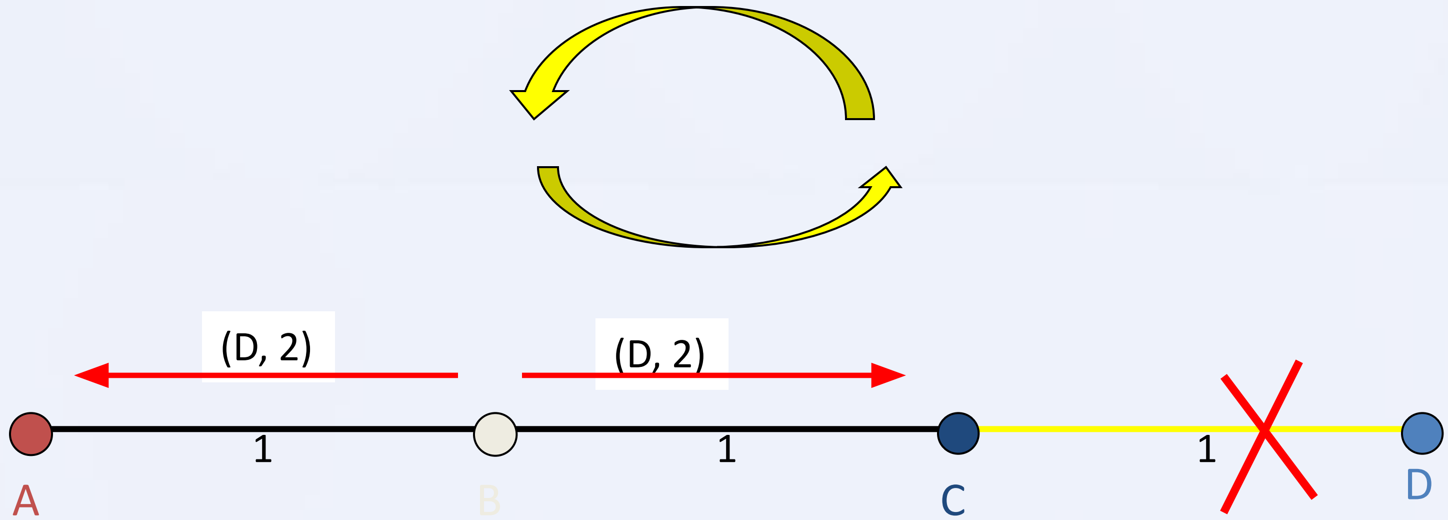


Dest.	Next	Metric	…
…	…	…	
D	B	3	

Dest.	Next	Metric	…
…	…	…	
D	C	2	

Dest.	Next	Metric	…
…	…	…	
D	D	∞	

Distance Vector (Loops)

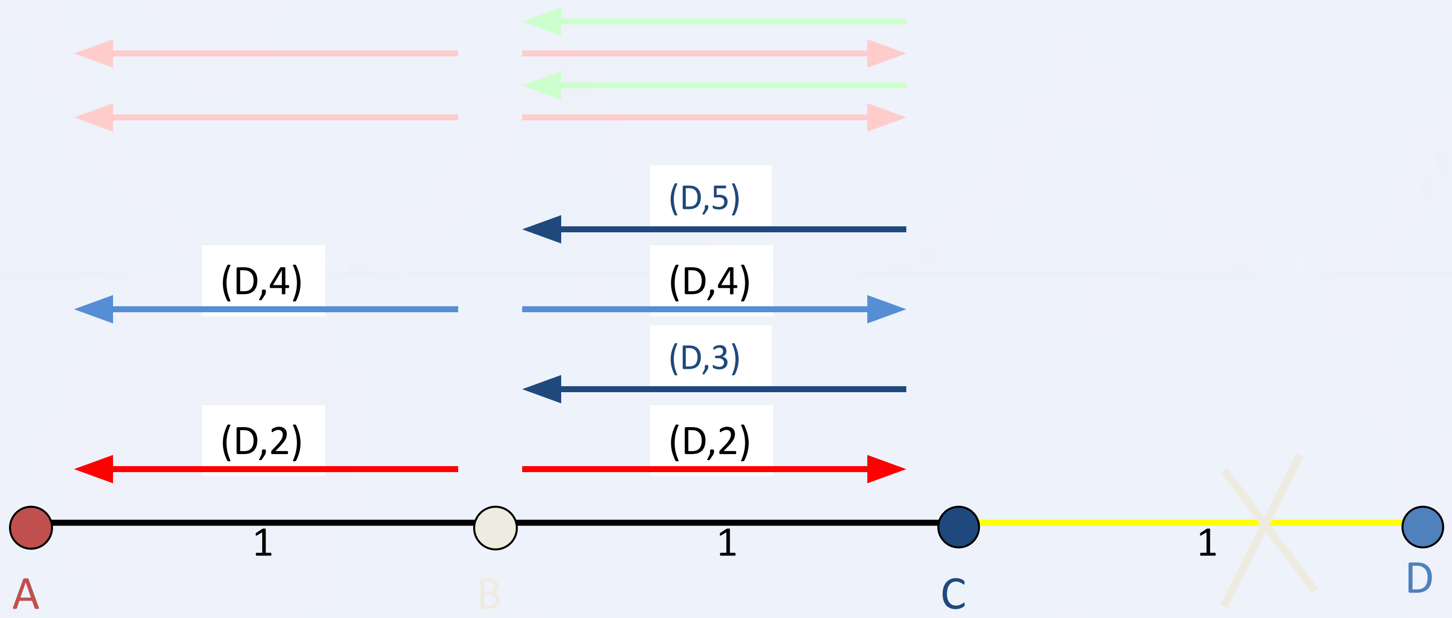


Dest.	Next	Metric	...
...	...	...	
D	B	3	

Dest.	Next	Metric	...
...	...	...	
D	C	2	

Dest.	Next	Metric	...
...	...	...	
D	B	3	

Distance Vector (Count to Infinity)



Dest.	Next	Metric	...
...	...	...	
D	B	3, 5, ...	

Dest.c	Next	Metric	...
...	...	...	
D	C	2, 4, 6...	

Dest.	Next	Metric	...
...	...	...	
D	B	3, 5, ...	

Distance Vector

- DV not suited for ad-hoc networks!
  - Loops
  - Count to Infinity
- New Solution -> DSDV Protocol

DSDV Protocol

- 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
  - Maintains routes which are never used

DSDV Protocol

- Keep the simplicity of Distance Vector
- *Guarantee Loop Freeness*
  - New Table Entry for Destination Sequence Number
- Allow fast reaction to topology changes
  - Make immediate route advertisement on significant changes in routing table

DSDV (Table Entries)

Destination	Next	Metric	Seq. Nr	Install Time	Stable Data
A	A	0	A-550	001000	Ptr_A
B	B	1	B-102	001200	Ptr_B
C	B	3	C-588	001200	Ptr_C
D	B	4	D-312	001200	Ptr_D

- **Sequence number** originated from destination. Ensures loop freeness.
- **Install Time** when entry was made (used to delete stale entries from table)
- **Stable Data** Pointer to a table holding information on how stable a route is. Used to damp fluctuations in network.

DSDV (Route Advertisements)

- 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 = ∞

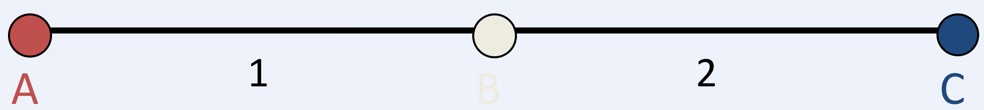
DSDV (Route Selection)

- 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 (Tables)



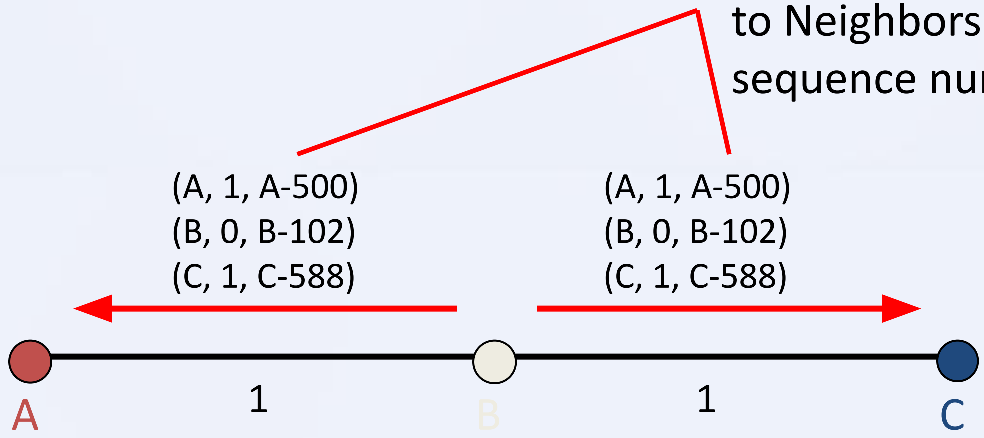
Dest.	Next	Metric	Seq
A	A	0	A-550
B	B	1	B-100
C	B	3	C-586

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 (Respond to Topology Changes)

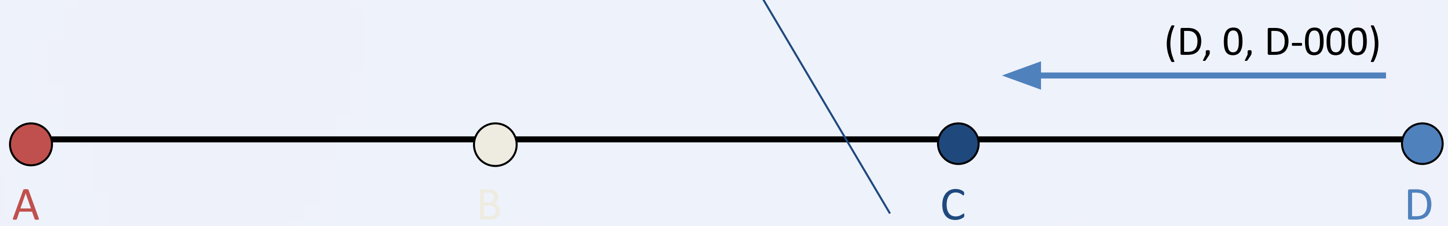
- Immediate advertisements
  - Information on new Routes, broken Links, metric change is immediately propagated to neighbors.

- Full/Incremental Update:
  - Full Update: Send all routing information from own table.
  - Incremental Update: Send only entries that has changed. (Make it fit into one single packet)

DSDV (New Node)

2. Insert entry for D with
sequence number D-000
Then immediately broadcast
own table

1. D broadcast for first time
Send Sequence number D-000



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

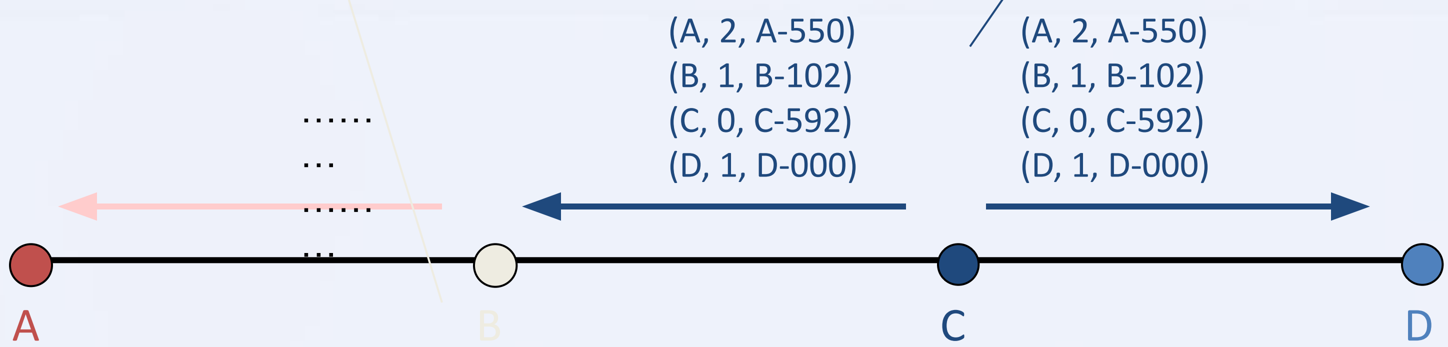
Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-104
C	C	1	C-590

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 cont.)

3. C increases its sequence number to C-592 then broadcasts its new table.

4. B gets this new information and updates its table…….



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

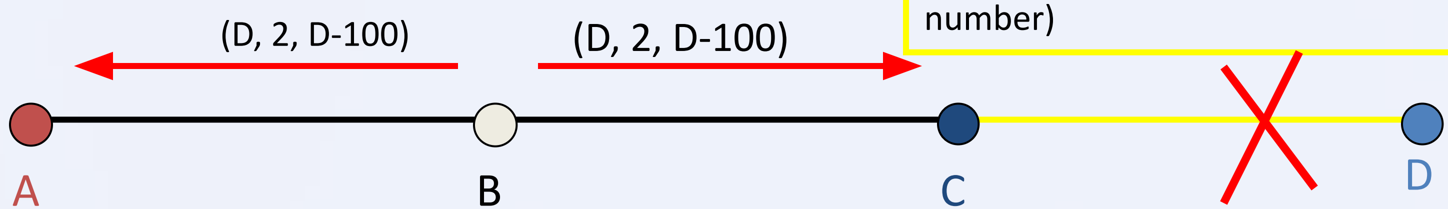
Dest.	Next	Metric	Seq.
A	A	1	A-550
B	B	0	B-102
C	C	1	C-592
D	C	2	D-000

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 loops, no count to infinity)

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)



A

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

B

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

C

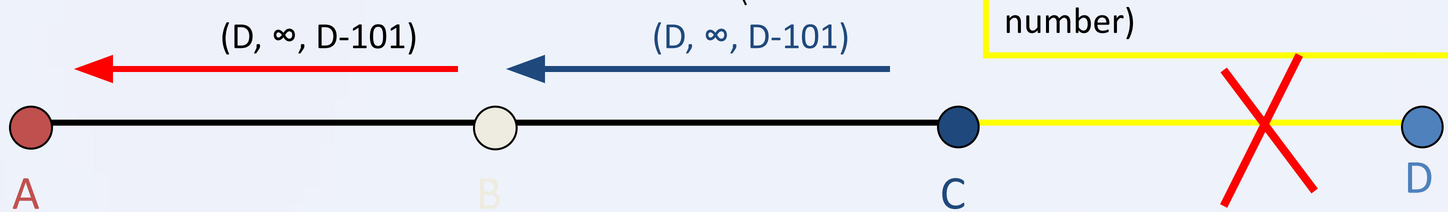
Dest.	Next	Metric	Seq.
…	…	…	
D	D	∞	D-101

DSDV (Immediate Advertisement)

3. Immediate propagation
B to A:
(update information has higher Seq. Nr. -> replace table entry)

2. Immediate propagation
C to B:
(update information has higher Seq. Nr. -> replace table entry)

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



A:

Dest.	Next	Metric	Seq.
…	…	…	…
D	B	3	D-100
D	B	∞	D-101

B:

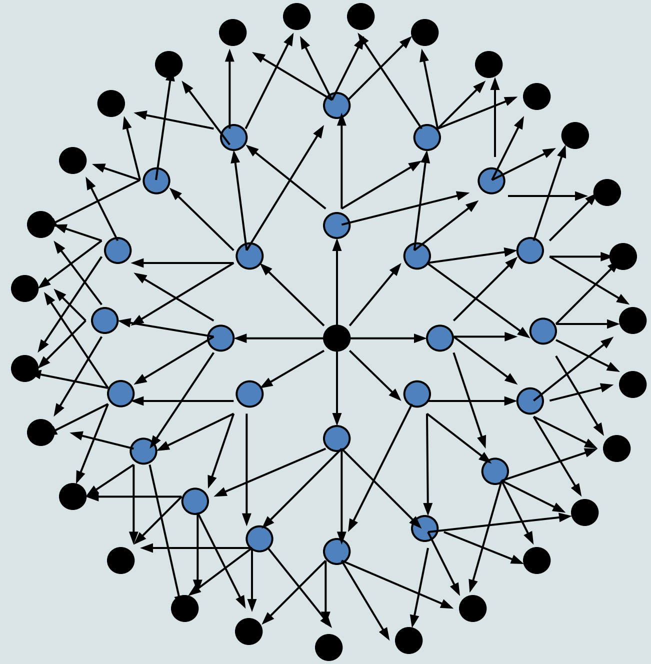
Dest.c	Next	Metric	Seq.
…	…	…	…
D	C	2	D-100
D	C	∞	D-101

C:

Dest.	Next	Metric	Seq.
…	…	…	
D	D	1	D-100
D	D	∞	D-101

Link State Routing

- Each node periodically floods status of its links
- Each node re-broadcasts link state information received from its neighbour
- Each node keeps track of link state information received from other nodes
- Each node uses above information to determine next hope to each destination

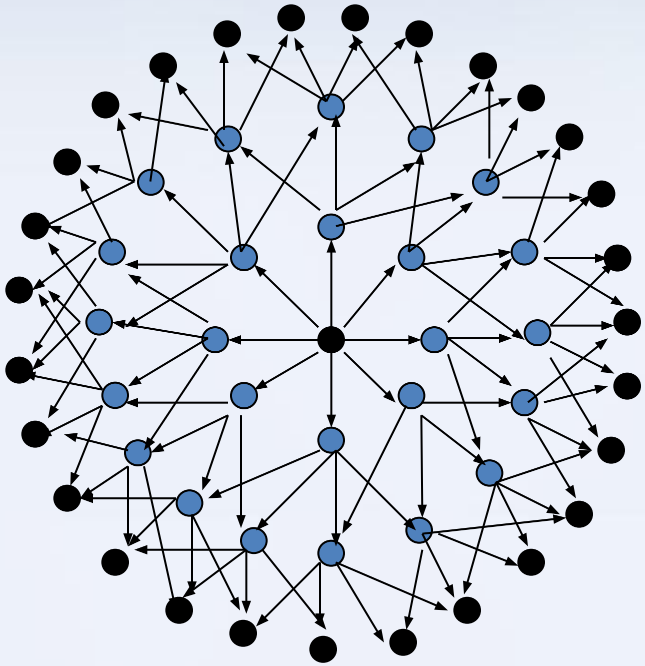


24 retransmissions to diffuse a message up to 3 hops

Optimized Link state routing (OLSR)

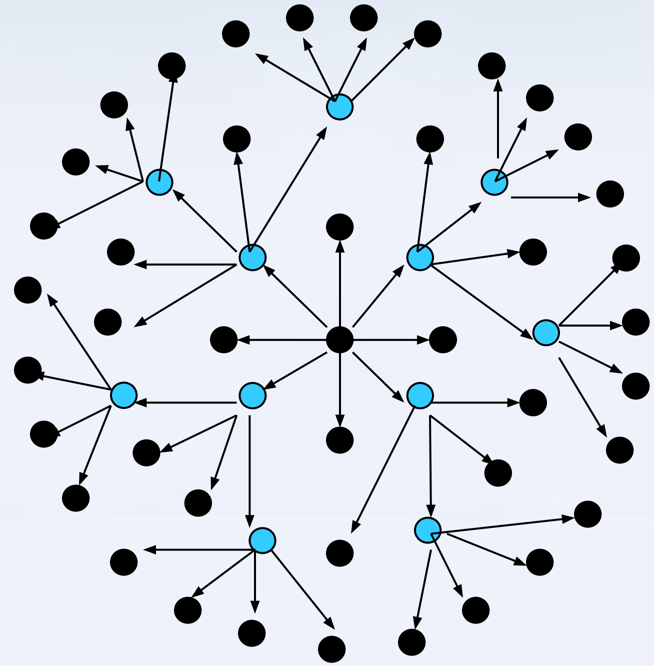
- Proactive protocol based on link state routing
  - overhead of flooding link state packets is reduced by requiring fewer nodes to forward the information to all the nodes
  - a broadcast from node X is only forwarded by its **multipoint relays**
  - very efficient in highly dense networks → less packets with state information

Optimized Link state routing (OLSR)



24 retransmissions to diffuse a message up to 3 hops!

Retransmission node

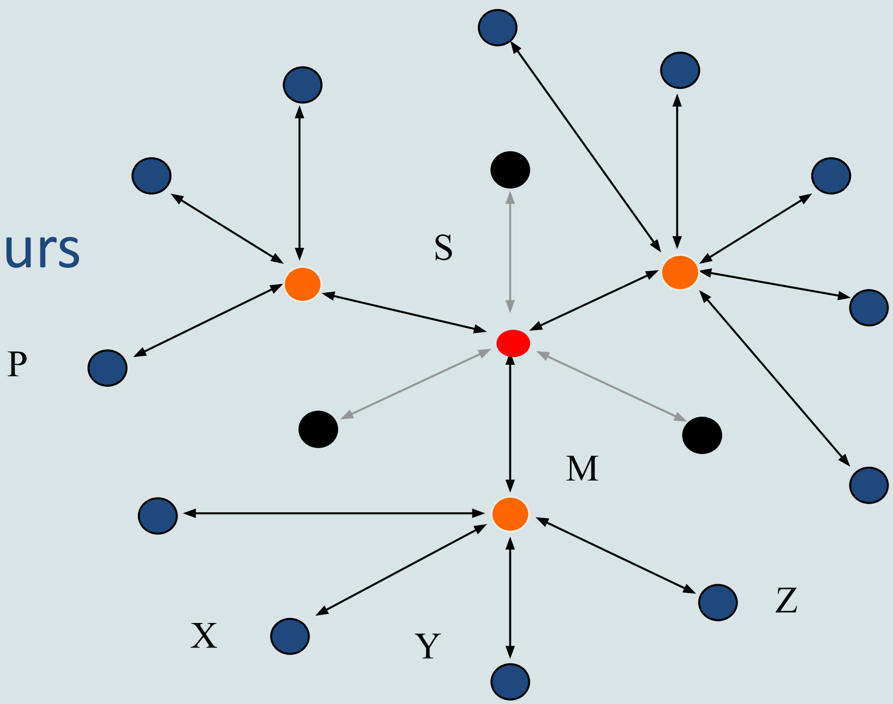


11 retransmission to diffuse a message up to 3 hops!

Retransmission node

Description of OLSR

- MPR (Multipoint relays)
- MPR selector
- Symmetric 1-hop neighbours
- Symmetric strict 2-hop neighbours



Selecting Multipoint relays

- Heuristic approaches to select the *MPRSet*
  - minimum set of one-hop neighbors through which all nodes in the two-hop neighborhood can be reached

$N1(x)$ - one-hop neighborhood of x

$N2(x)$ - two-hop neighborhood of x

$MPRSet(x) \leftarrow \emptyset$

↓

$MPRSet(x) \leftarrow$ {nodes that belong to $N1(x)$ and which are the only neighbors of nodes in $N2(x)$}

↓

while thre exist nodes in $N2(x)$ not covered by $MPRSet(x)$:

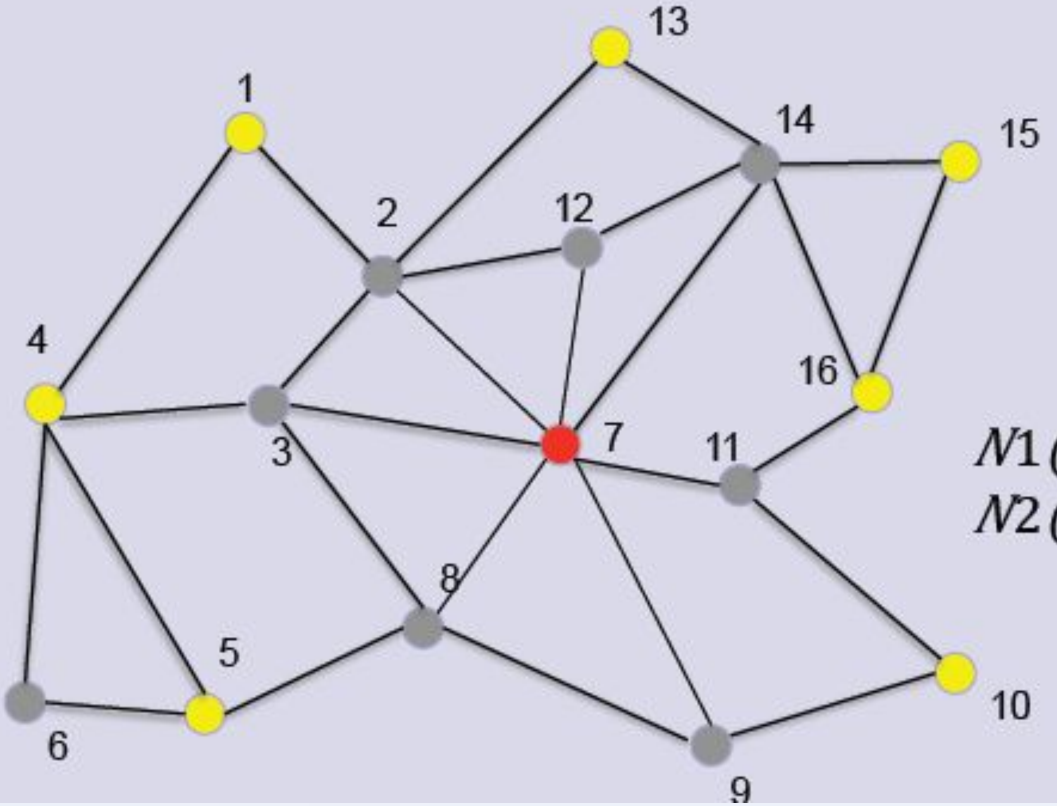
↓

1. for each node in $N1(x)$ which is not in $MPRSet(x)$, compute the maximum no. of nodes it covers among the uncoverd nodes in $N2(x)$

↓

2. add to $MPRSet(x)$ the node in $N1(x)$ for which this number is maximum

Example



$N1(7)=\{2,3,8,9,11,12,14\}$
$N2(7)=\{1,4,5,10,13,15,16\}$