Routing in Ad Hoc Wireless Networks

PROF. MICHAEL TSAI / DR. KATE LIN

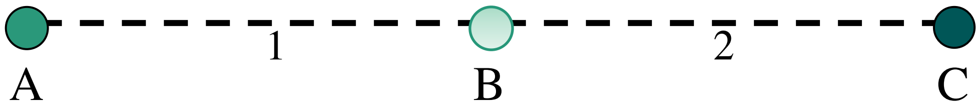
2014/05/14

Routing Algorithms

- **Link-State algorithm**
  - Each node maintains a view of the whole network topology
  - Find the shortest path over the network
  - Maintain the topology information by periodical flooding

- **Distance-Vector algorithm**
  - Each node maintains the distance of each destination and the corresponding next hop
  - Periodically send the table to all neighbors
  - Also known as distributed bellman-ford

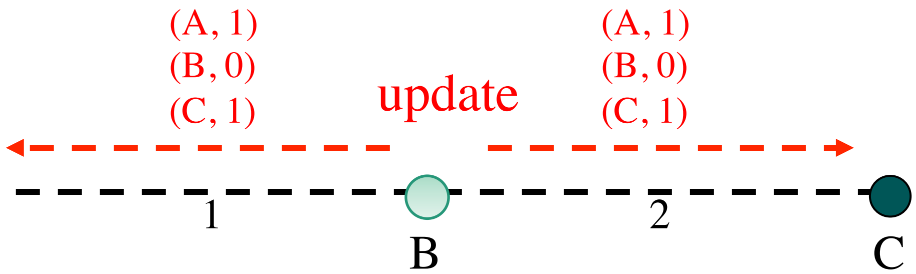
Distance Vector



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	



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

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

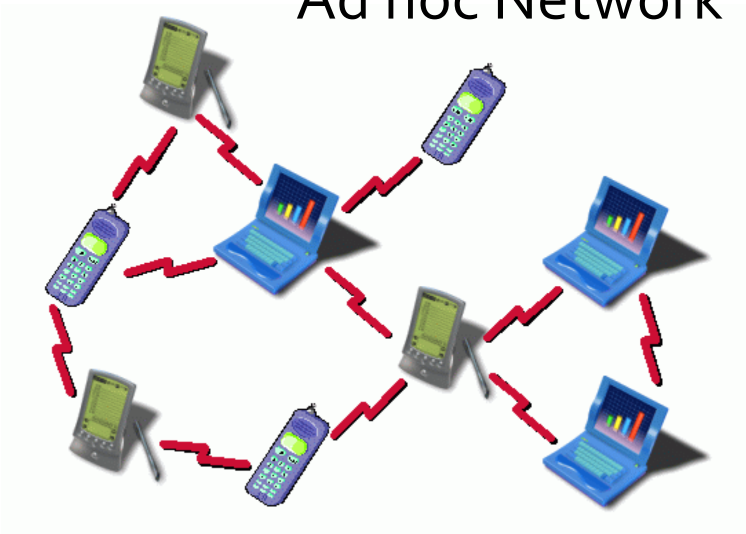
Dest.	Next	Metric	...
A	B	~~3~~ 2	
B	B	2	
C	C	0	

Ad Hoc Wireless Networks

- **No base station or access point to relay the packets**
- **Relaying is necessary to send information to destinations out of our range**
- **Initial application: military usage**
- **Other applications: mesh networks, vehicular networks, etc**

Ad hoc Network

Infrastructure-based Network



Why do we need new protocols?

- **No centralized control**
- **No dedicated routers**
- **Unpredictable network topology changes**
- **Time-variant wireless channel**
  - Link breakage is common in wireless network → Connectivity problem
  - Links are not always bidirectional and/or symmetric
- **Power Limitation**

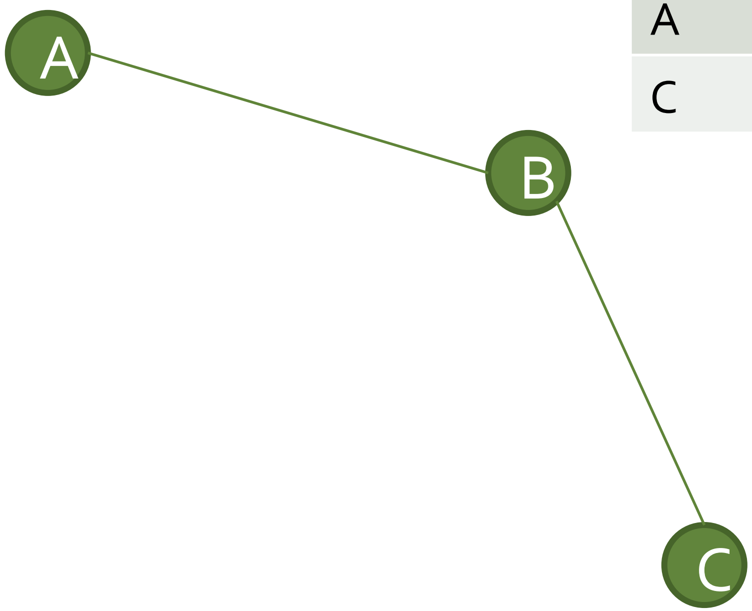
Conventional Routing Protocols

- **Not designed for highly dynamic and low bandwidth networks**

- **Loop formation when topology changes**

- **Flooding causes high control overhead (e.g., Link State)**

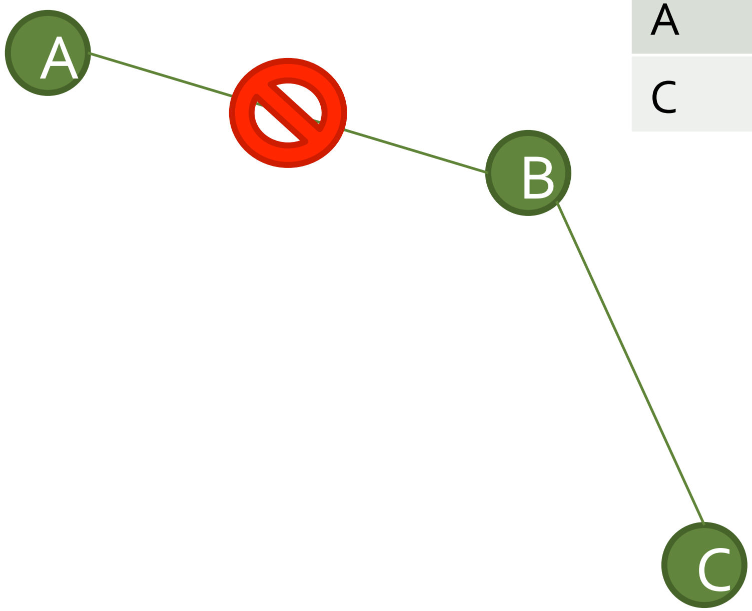
Count-to-infinity Problem



Dest	Cost	Next Hop
A	1	A
C	1	C

Dest	Cost	Next Hop
A	2	B
B	1	B

Count-to-infinity Problem

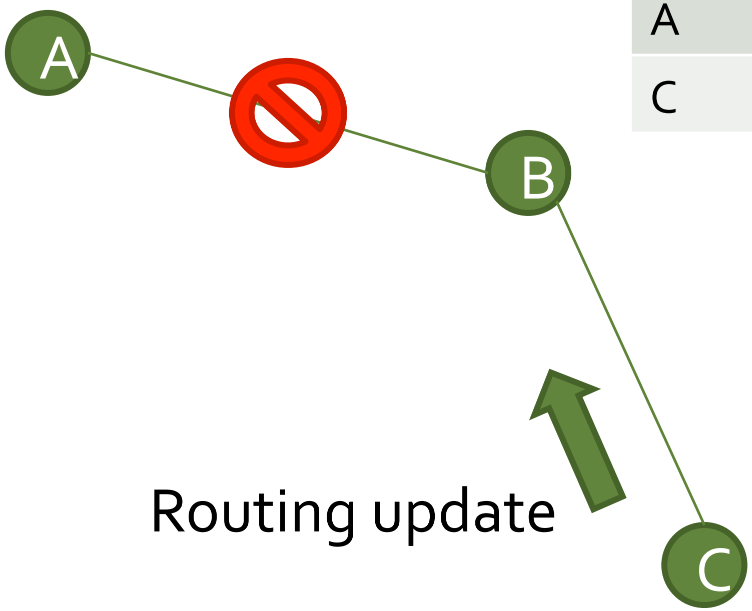


Dest	Cost	Next Hop
A	Infinity	Null
C	1	C

Dest	Cost	Next Hop
A	2	B
B	1	B

Count-to-infinity Problem

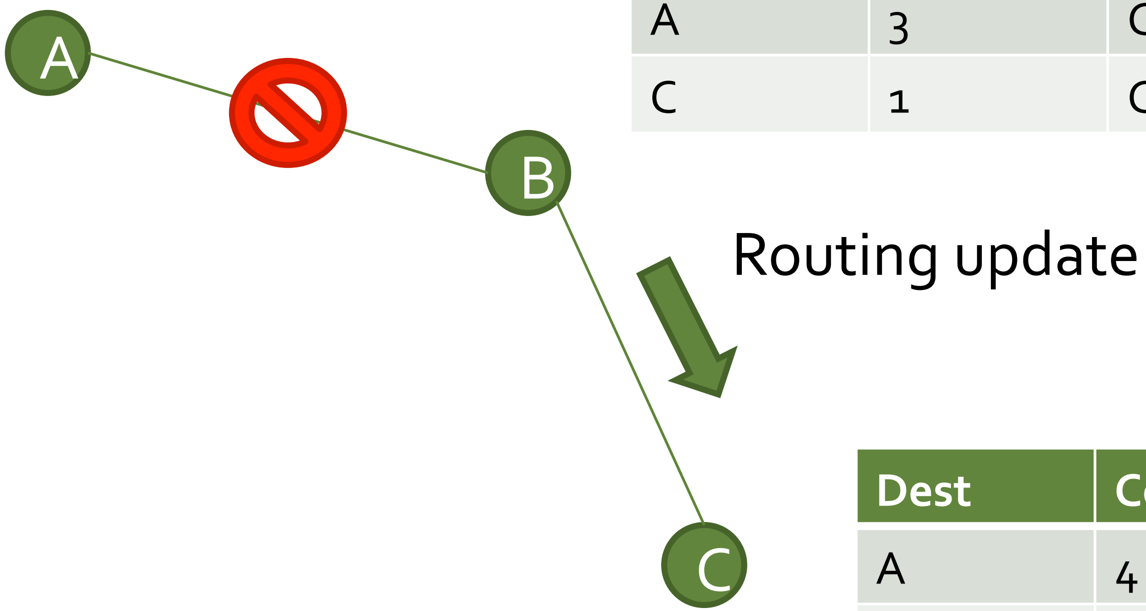
Dest	Cost	Next Hop
A	3	C
C	1	C



Dest	Cost	Next Hop
A	2	B
B	1	B

Count-to-infinity Problem

Dest	Cost	Next Hop
A	3	C
C	1	C



Dest	Cost	Next Hop
A	4	B
B	1	B

This continues until the cost reaches infinity (unreachable). During the process, the packets destined for A will bounce back and forth between B and C

Existing Routing Protocols

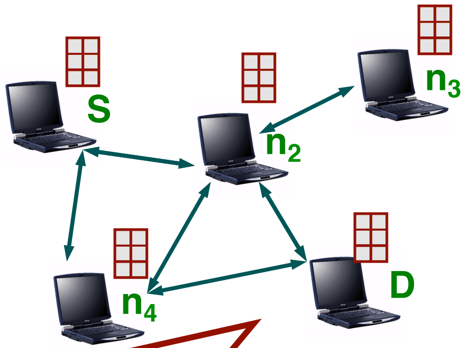
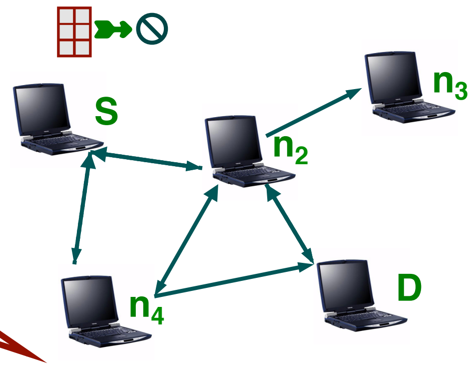


Table-Driven:

- S and all other nodes maintain full routing information
- Require periodic table update

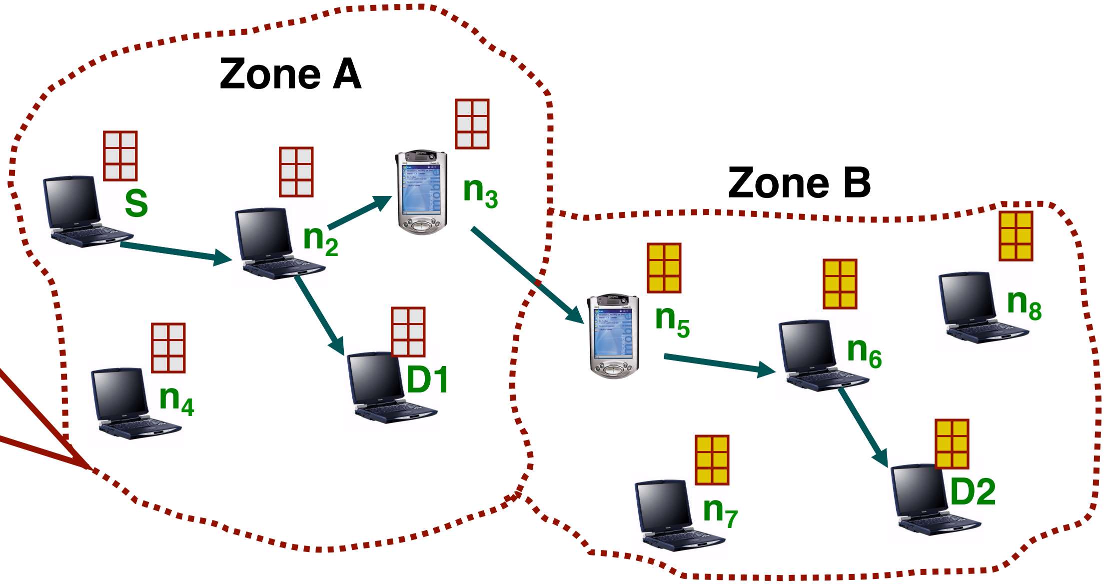
Demand-Driven

- Route is discovered when S wants to talk to D
- A Route only needs to be maintained for as long as S and D are still talking
- EX: Dynamic Source Routing (DSR)



Hybrid Scheme

- Network is divided into multiple zones
- Use Table-Driven within the zone
- Demand-Driven across the zones through boundary nodes



Proactive vs. Reactive Routing

- **Proactive**
  - Table driven
  - Rely on periodic update to keep track of the topology change
  - No latency in route discovery
  - Need large storage space to keep information of the entire network
  - A lot of routing information may never be used

- **Reactive**
  - On demand
  - Route Discovery by local flood or gossiping
  - Additional latency during route discovery
  - Not appropriate for real-time communication
  - Route maintenance
    - Feedback from Link Level ACK
    - Issue new route discovery when link breaks

Destination Sequenced Distance Vector (DSDV)

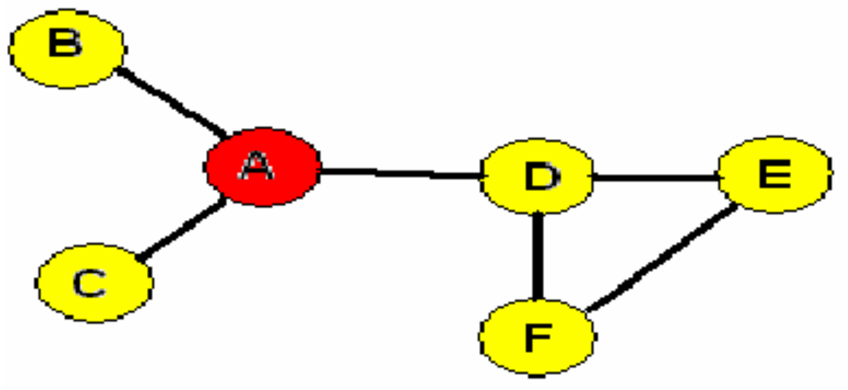
Proactive Routing Protocols

- **Each node advertises a monotonically increasing sequence number**
- **Each Route entry is tagged with a sequence number generated by destination to prevent loops (*count-to-infinity* problem)**
- **Sequence number indicates the "freshness" of a route**
  - Routes with more recent sequence numbers are preferred for packet forwarding
  - If same sequence number, one having smallest metric is used

C. E. Perkins and P. Bhagwat. "Highly dynamic Destination Sequenced Distance-Vector routing (DSDV) for mobile computers", *In Proceedings of the SIGCOMM '94 Conference on Communication Architecture, Protocols and Applications*, pages 234-244, August '94.

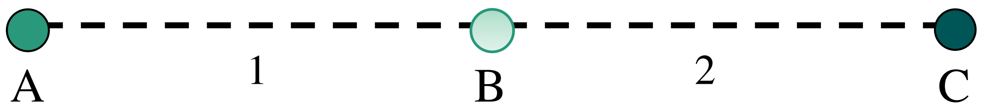
Example: DSDV

- **For each reachable node in the network the routing entry contains:**
  - Destination Address
  - Next Hop
  - Distance (Metric)
  - Sequence Number



Destination	Next Hop	Distance	Sequence Number
A	A	0	S205_A
B	B	1	S334_B
C	C	1	S198_C
D	D	1	S567_D
E	D	2	S767_E
F	D	2	S45_F

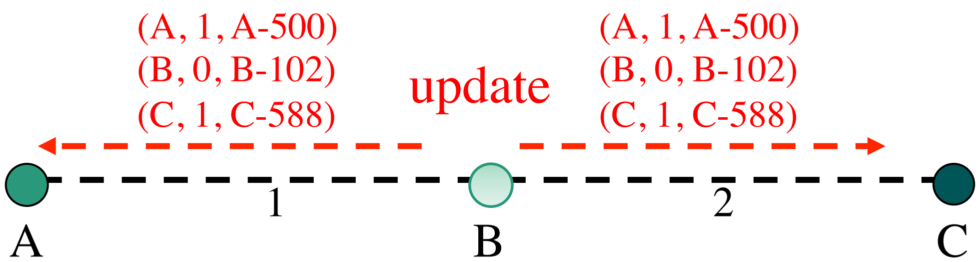
DSDV – Table Update



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

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

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

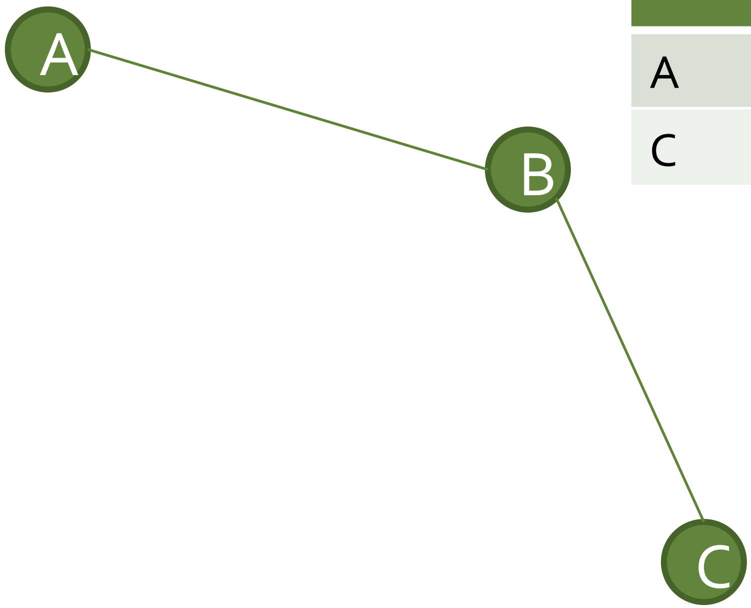


Dest.	Next	Metric	Seq
A	A	0	A-550
B	B	1	B-102
C	B	2	C-588

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

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

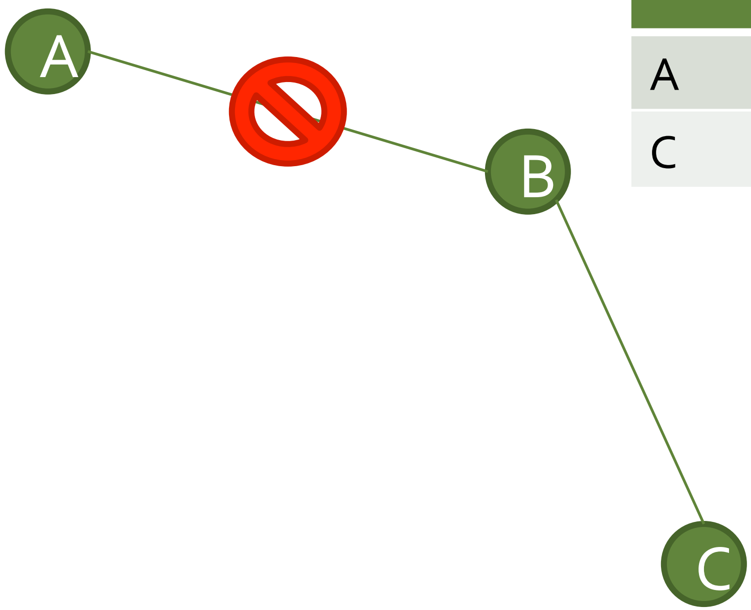
Count-to-infinity Problem



Dest	Cost	Next Hop	Seq. #
A	1	A	1
C	1	C	1

Dest	Cost	Next Hop	Seq. #
B	1	B	1
C	1	C	1

Count-to-infinity Problem

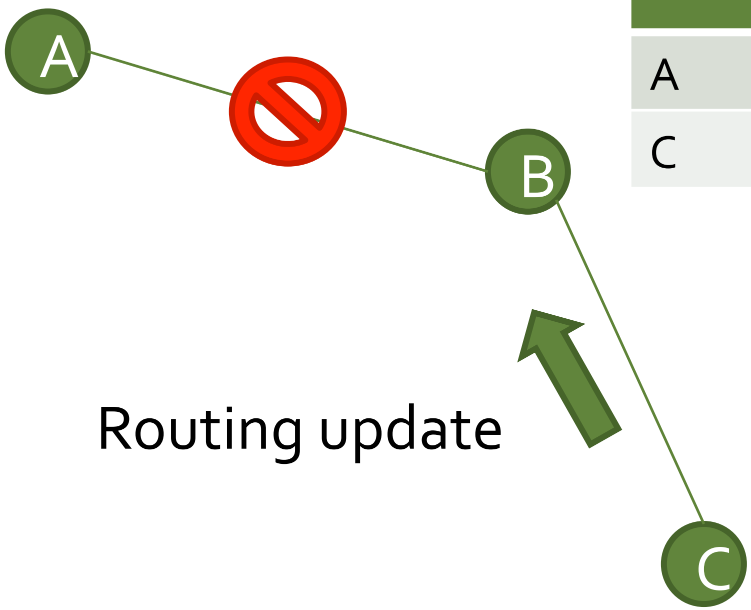


Dest	Cost	Next Hop	Seq. #
A	Infinity	Null	2
C	1	C	1

Dest	Cost	Next Hop	Seq. #
B	1	B	1
C	1	C	1

Count-to-infinity Problem

Dest	Cost	Next Hop	Seq. #
A	Infinity	Null	2
C	1	C	1



Dest	Cost	Next Hop	Seq. #
A	2	B	1
C	1	C	1

C's routing update will not change B's routing table since the sequence number is smaller (older).

DSDV: Topology changes

- **Assign a metric of ∞ to**
  - A broken link
  - Any route through a hop with a broken link
- **"∞ routes" are assigned new sequence numbers by any host and immediately broadcast via a triggered update**
- **If a node has an equal/later sequence number with a finite metric for an "∞ route", a route update is triggered**

DSDV - Summary

- **Advantages**
  - Simple (almost like Distance Vector)
  - Loop free
  - No latency for route discovery

- **Disadvantages**
  - Periodical updates
  - Most routing information never used

Dynamic Source Routing

- **Assumptions**
  - All nodes are willing to participate
  - The network size is small
  - The degree of network dynamics is moderate with respect to the packet transmission latency
  - All nodes are overhearing (promiscuous)
  - Links are symmetric

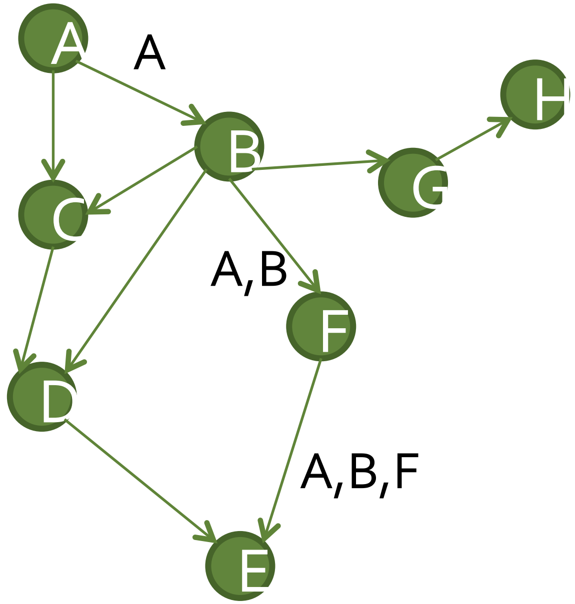
Dynamic Source Routing

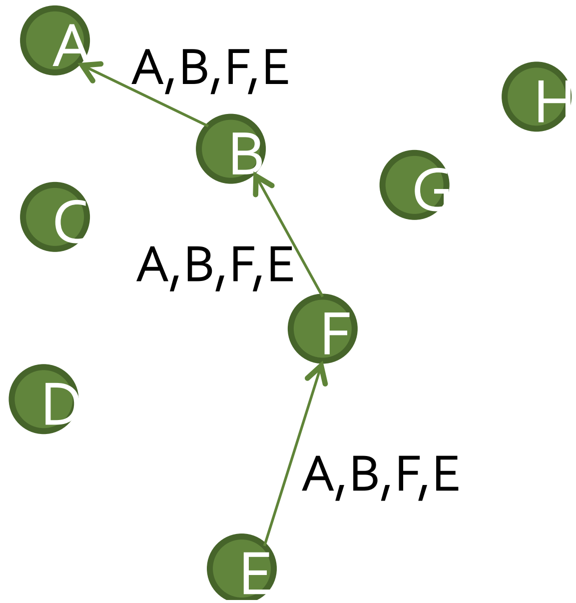
- **Route Discovery**
  - Route Request (RREQ)
  - Route Reply (RREP)

- **Route Maintenance**
  - Route Error (PERR)

Dynamic Source Routing [DSR]
Route Discovery

- **Source node**
  - Broadcasts the Route Request (RREQ) <id, target>
- **Intermediate node**
  - Discards if the *id* has been seen before, or node is in the *route record* (header of RREQ)
  - Else append address in the *route record and* rebroadcast
- **Destination Node**
  - Return Route Reply (RREP)
  - Use previously cached route to source node
  - Call Route Discovery for source node, with route reply piggy backed
  - Use reverse sequence of Route Record, in case of bidirectional links

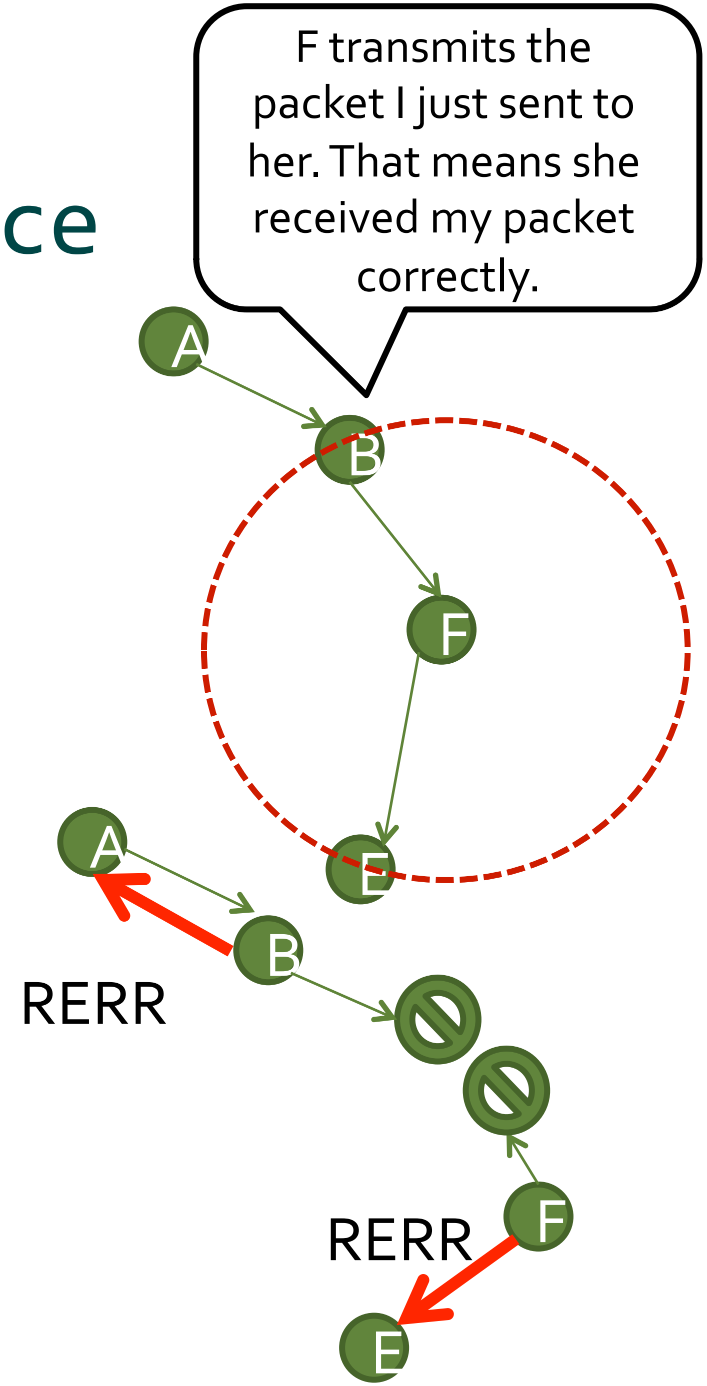




D. B. Johnson, D.A. Maltz, and J. Broch. "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks", *Ad Hoc Networking*, pages 139-172, 2001.

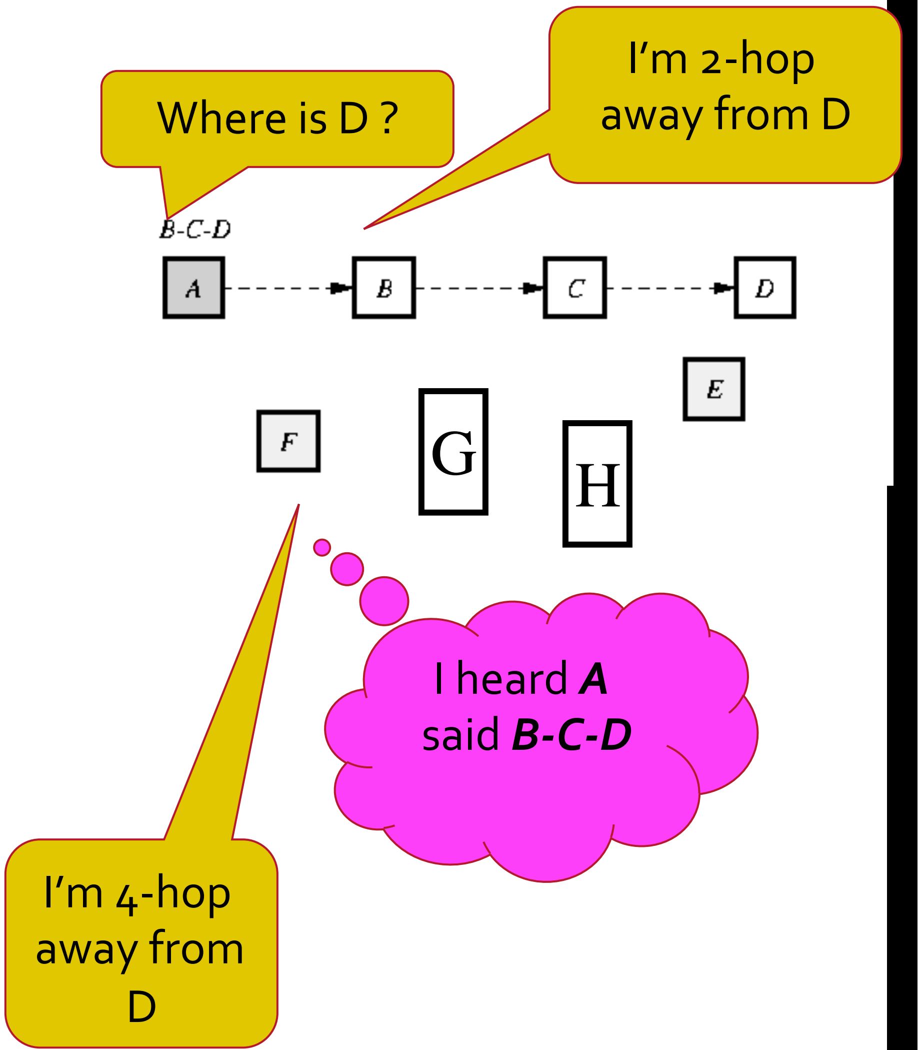
DSR: Route Maintenance

- **Monitoring the route**
  - Passive Acknowledgement – overhearing the next-hop node sending packet to its next-hop
  - Set a bit in packet to request explicit next hop acknowledgement
- **Route Error**
  - Rely on data link layer to report the broken links
  - Notify source of the broken link via Route Error (RERR)
  - Source truncates all routes which use nodes mentioned in RERR
  - Initiate new route discovery



Optimization 1: Route Caching

- **Use cached entries to create RREP at intermediate node**
  - S finds route [S,E,F,J,D] to D, S also learns route [S,E,F] to F
  - F receives Route Request [S,E,F] destined for some node D, F learns route [F,E,S] to S
- **Promiscuous mode to add more routes**
  - Caching overheard RREQ/RREP



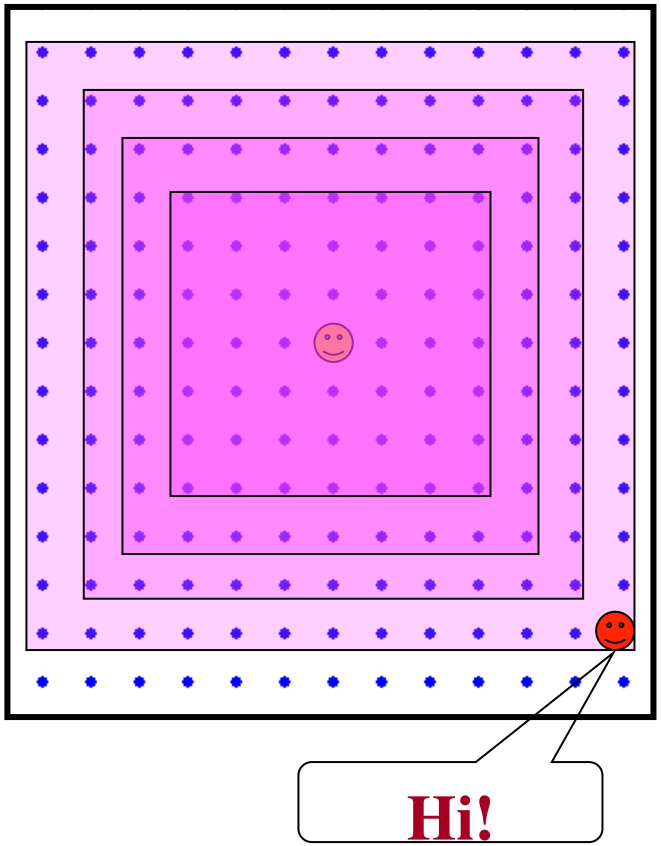
Optimization 1: Route Caching

- **Route reply storm**
  - A lot of neighbors know the route to target and attempt to send RREP in response to RREQ
  - Solution: Delay RREP for a period d=H*(h-1+r)
    - r : random number between 0 and 1
    - H : small constant delay
    - h : number of hops to source from that node

- **Out-of-date cache**
  - Cached routes may become invalid
  - Stale or invalid information may be propagated to whole network

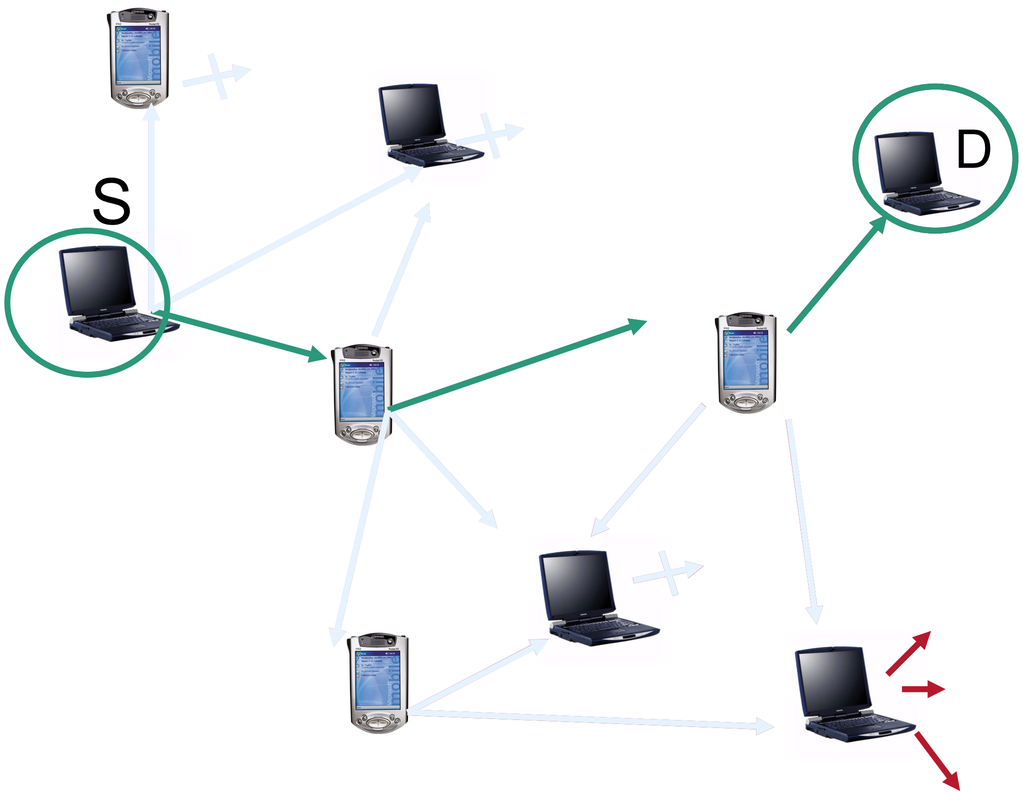
Optimization 2: Expanding Ring

- **Route Request Hop Limit**
- **Use TTL in the packet header to specify the first ring boundary**
- **RREQ is initially forwarded *n* times (*n* hops)**
- **If destination is not within *n*-hop**
  - Increase TTL to a larger value



This is useful if destination is close to the source

Optimization 3: Gossiping



Gossip: Probabilistic Flooding

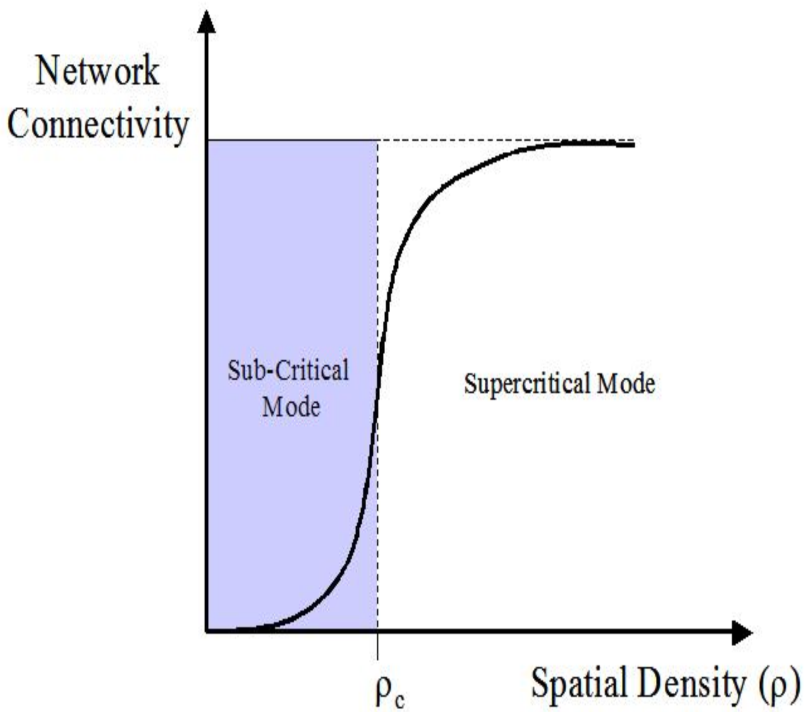
- **Gossip-Based Routing**
  - Node forward packets with some probability $p_G < 1$
- **How good is it?**
  - 35% less overhead than flooding
- **What determine P_G?**

Network Connectivity

Z. Haas, J. Halpern and L. Li, Gossip-based ad hoc routing, in: *IEEE INFOCOM* (2002)

Network Connectivity

Connectivity: Fraction of nodes that is connected to the network



- **Sub-Critical**
  - Low connectivity
  - Mobile nodes are sparsely distributed in the network
  - Performance is limited !!
- **Super-Critical**
  - High connectivity region
  - Most or all the nodes can communicate

DSR – Summary

- **Advantages**
  - Purely on-demand
  - Zero control message overhead
  - Loop-free route
- **Disadvantages**
  - High data latency
  - Space overhead in packets
  - Storage overhead for caching
  - Promiscuous mode consumes extra power

Ad-Hoc On Demand Distance Vector Routing (AODV)

- **Protocol overview - Pure on-demand protocol**
  - Node does not maintain knowledge of another node unless it communicates with it
  - Routes discovered on as-needed basis and maintained only as long as necessary
  - Nodes not involved in the route should not pay any cost
  - No cost to deal with out-of-date
  - Little or no periodic advertisement

C. E. Perkins and E. M. Royer. "Ad-Hoc On Demand Distance Vector Routing", *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90-100, 1999.

AODV – Route Discovery

- **Initiation**
  - Source node sends a Route Request (RREQ) when it has no information about destination node in its table
  - RREQ contains
    - Source and destination's address and sequence number
    - Broadcast id
    - Hop count
  - Source address and broadcast id uniquely identify RREQ
- **Reverse Path Setup**
  - Reverse paths are formed when a node hears a route request
  - Neighbor increments hop count and broadcasts to neighbors
  - Records address of neighbor which first sends the RREQ

AODV – Route Discovery

- **Forward Path Setup**
  - Intermediate node satisfies RREQ if
    - Destination itself
    - Has route entry in table with destination sequence number ≥ that given in RREQ
  - Unicasts RREP to neighbor which sent RREQ
    - Source address
    - Destination address and sequence number (updated)
    - Hop count
    - Lifetime
  - As RREP travels backwards, each node sets pointer to sending node and updates destination sequence number and timeout entry for source and destination routes

AODV – Route Discovery

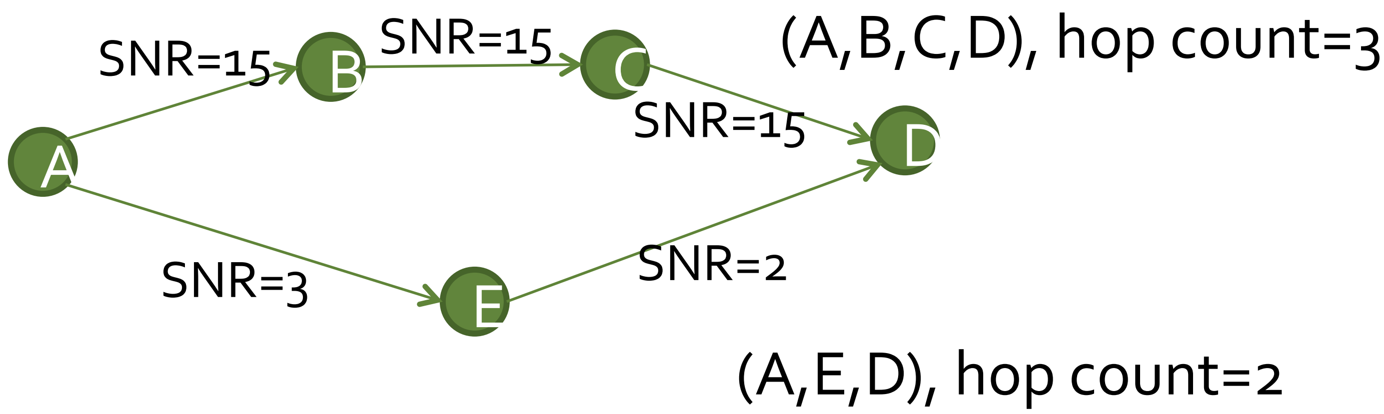
- **Other nodes**
  - RREQ times out : Route Request Expiration Timer
  - Deletes corresponding pointers
- **More than one RREP received**
  - One with greater destination number
  - Lesser hop count
- **Source node starts transmission - updates if a better RREP is received**

C. E. Perkins and E. M. Royer. "Ad-Hoc On Demand Distance Vector Routing", *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, pages 90-100, 1999.

AODV – Route Maintenance

- **Nodes send *hello message* if it has not sent a packet in hello_interval**
- **Failing to receive *allowed_hello_loss* packets consecutively means link is broken**
- **In case of broken link**
  - unsolicited RERR sent to affected source node
  - Source initiates new RREQ
  - Sequence number updated
  - Hop count set = $\propto$
- **Route Caching Timeout after the route is considered invalid**
- **Optional* AODV-LL uses link layer ACK instead of hello messages**

Link Quality Metrics



- **The protocol chooses the route with the smallest hop count →Long hops will be included**
- **Long hops usually have lower SNR → high PER → retransmission!**
- **Original thought: lower hop count = lower bandwidth usage**
- **New thought: retransmission means wasted bandwidth**

Link Quality Metrics

- **Instead of using hop count only, we need to take "link quality" into account!**
- **What is a good metric for link quality?**
  - RSSI (representing SNR)
  - ETX (Expected Transmission Count)
- **Then we combine hop count + link quality to choose an optimal route**

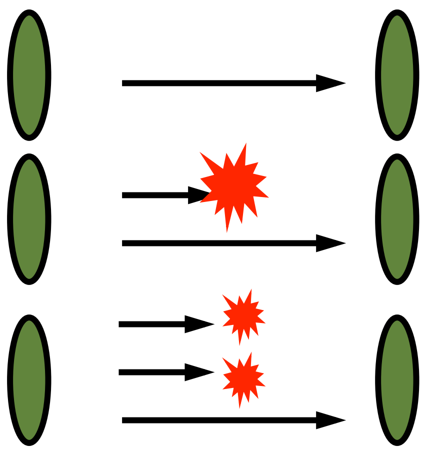
Example: ETX

Minimize total transmissions per packet

(ETX, Expected Transmission Count)

Link throughput ≈ 1/ Link ETX

Delivery Ratio	Link ETX	Throughput
100%	1	100%
50%	2	50%
33%	3	33%



Measuring delivery ratios

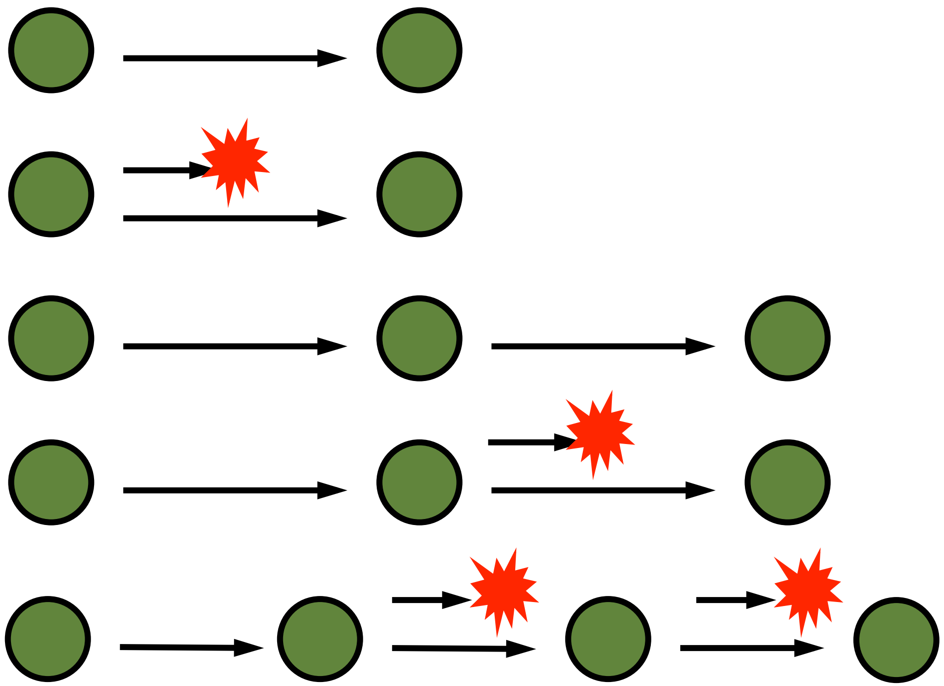
- **Each node broadcasts small link probes (134 bytes), once per second**
- **Nodes remember probes received over past 10 seconds**
- **Reverse delivery ratios estimated as**

$$r_{\text{rev}} \approx \text{pkts received / pkts sent}$$

- **Forward delivery ratios obtained from neighbors (piggybacked on probes)**

Route ETX

Route ETX = Sum of link ETXs



Route ETX	Throughput
1	100%
2	50%
2	50%
3	33%
5	20%