

Advanced Topics in Computer Networks

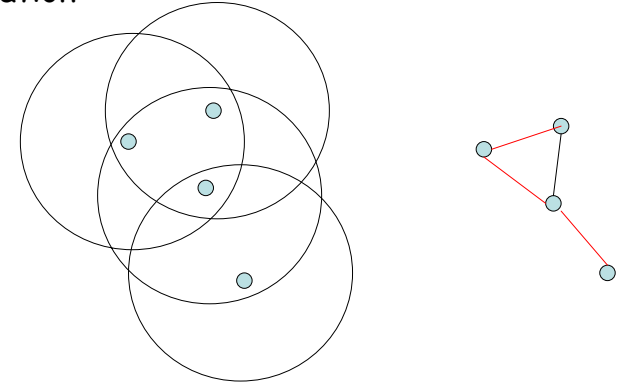
Ad Hoc Networks

Chalermek Intanagonwiwat

Slides courtesy of Golden G. Richard III, Jim Thompson, Musenki, and Nitin Vaidya

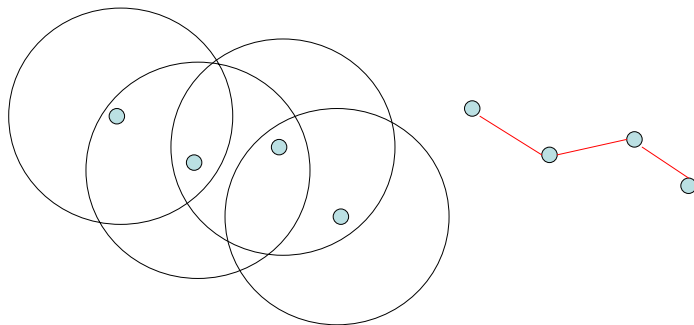
Mobile Ad Hoc Networks

- May need to traverse multiple links to reach a destination



Mobile Ad Hoc Networks

- Mobility causes route changes



Mobile Ad Hoc Networks

- Formed by wireless hosts which may be mobile
- Don't need a pre-existing infrastructure
 - ie, don't need a backbone network, routers, etc.
- Routes between nodes potentially contain multiple hops
- Why MANET?
 - Ease, speed of deployment
 - Decreased dependence on infrastructure
 - Can use in many scenarios where deployment of a wired network is impractical or impossible
 - Lots of military applications, but there are others

Many Applications

- Personal area networking
 - cell phone, laptop, ear phone, wrist watch
- Military environments
 - soldiers, tanks, planes
- Civilian environments
 - meeting rooms
 - sports stadiums
 - groups of boats, small aircraft (wired REALLY impractical!!)
- Emergency operations
 - search-and-rescue
 - policing and fire fighting

Many Variations

- Fully Symmetric Environment
 - all nodes have identical **capabilities** and **responsibilities**
- Asymmetric Capabilities
 - transmission ranges and radios may differ
 - battery life at different nodes may differ
 - processing capacity may be different at different nodes
 - speed of movement different
- Asymmetric Responsibilities
 - only some nodes may route packets
 - some nodes may act as **leaders** of nearby nodes (e.g., "cluster head")

Many Variations

- Traffic characteristics may differ
 - bandwidth
 - timeliness constraints
 - reliability requirements
 - unicast / broadcast / multicast / geocast
- May co-exist (and co-operate) with an infrastructure-based network

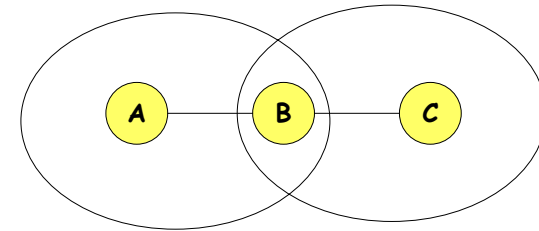
Many Variations

- Mobility patterns may be different
 - people sitting at an airport lounge (little mobility)
 - taxi cabs (highly mobile)
 - military movements (mostly clustered?)
 - personal area network (again, mostly clustered?)
- Mobility characteristics
 - speed
 - predictability
 - direction of movement
 - pattern of movement
 - uniformity (or lack thereof) of mobility characteristics among different nodes

Challenges

- Limited wireless transmission range
- Broadcast nature of the wireless medium
- Packet losses due to transmission errors
- Environmental issues ("chop that tree!!")
- Mobility-induced route changes
- Mobility-induced packet losses
- Battery constraints
- Potentially frequent network partitions
- Ease of snooping on wireless transmissions (security hazard)

Hidden Terminal Problem



Nodes A and C cannot hear each other

Transmissions by nodes A and C can collide at node B

On collision, both transmissions are lost

Nodes A and C are **hidden from each other**

Why is Ad hoc Routing Different?

- Host mobility
 - link failure/repair due to mobility may have different characteristics than those due to other causes
 - traditional routing algorithms assume relatively stable network topology, few router failures
- Rate of link failure/repair may be high when nodes move fast
- New performance criteria may be used
 - route stability despite mobility
 - energy consumption

Routing Protocols

- Proactive protocols
 - Determine routes independent of traffic pattern
 - Traditional link-state and distance-vector routing protocols are proactive
 - Maintain routes between every host pairs at all times
 - Based on periodic updates
 - High routing overhead
 - Example: DSDV (Destination Sequenced Distance Vector)

Routing Protocols (cont.)

- Reactive protocols
 - Maintain routes only if needed
 - Sources initiates route discovery
 - Example: DSR (Dynamic Source Routing) and AODV (Ad hoc On-demand Distance Vector)
- Hybrid protocols
 - Adaptive
 - Combination of proactive and reactive
 - Example: ZRP (Zone Routing Protocol)

Trade-Off

- Latency of route discovery
 - Proactive protocols may have lower latency since routes are maintained at all times
 - Reactive protocols may have higher latency because a route from X to Y will be found only when X attempts to send to Y
- Overhead of route discovery/maintenance
 - Reactive protocols may have lower overhead since routes are determined only if needed
 - Proactive protocols can (but not necessarily) result in higher overhead due to continuous route updating
- Which approach achieves a better tradeoff depends on the traffic and mobility patterns

Destination-Sequenced Distance-Vector (DSDV)

- Each node maintains a routing table which stores
 - next hop, cost metric towards each destination
 - a sequence number that is created by the destination itself
- Each node periodically forwards routing table to neighbors
 - Each node increments and appends its sequence number when sending its local routing table
- Each route is tagged with a sequence number; routes with greater sequence numbers are preferred

Destination-Sequenced Distance-Vector (cont.)

- Each node advertises a monotonically increasing even sequence number for itself
- When a node decides that a route is broken, it increments the sequence number of the route and advertises it with infinite metric
- Destination advertises new sequence number

Destination-Sequenced Distance-Vector (cont.)

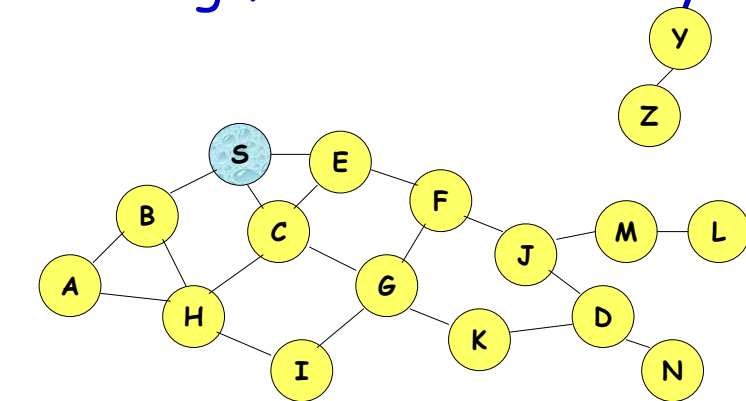


- When X receives information from Y about a route to Z
 - Let destination sequence number for Z at X be $S(X)$, $S(Y)$ is sent from Y
 - If $S(X) > S(Y)$, then X ignores the routing information received from Y
 - If $S(X) = S(Y)$, and cost of going through Y is smaller than the route known to X, then X sets Y as the next hop to Z
 - If $S(X) < S(Y)$, then X sets Y as the next hop to Z, and $S(X)$ is updated to equal $S(Y)$

Flooding for Data Delivery

- Sender S broadcasts data packet P to all its neighbors
- Each node receiving P forwards P to its neighbors
- Sequence numbers used to avoid the possibility of forwarding the same packet more than once
- Packet P reaches destination D provided that D is reachable from sender S
- Node D does not forward the packet

Flooding for Data Delivery

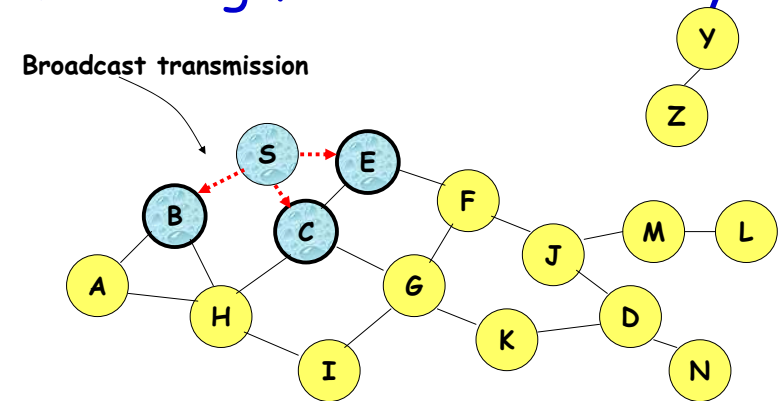


Represents a node that has received packet P



Represents that connected nodes are within each other's transmission range

Flooding for Data Delivery



Broadcast transmission

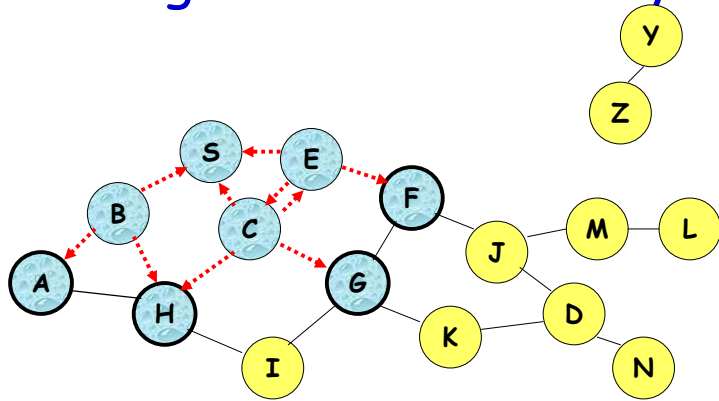


Represents a node that receives packet P for the first time



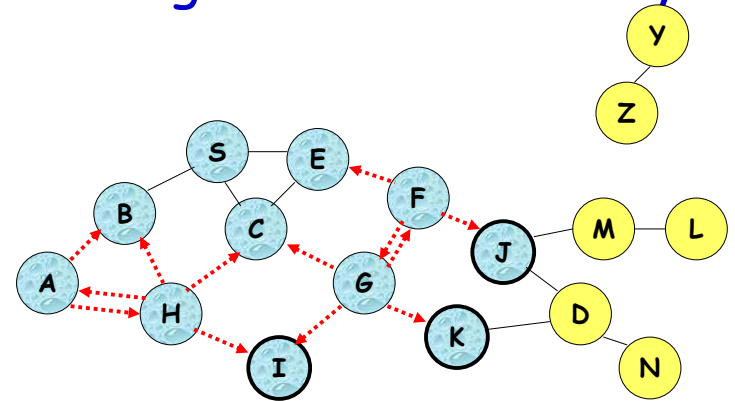
Represents transmission of packet P

Flooding for Data Delivery



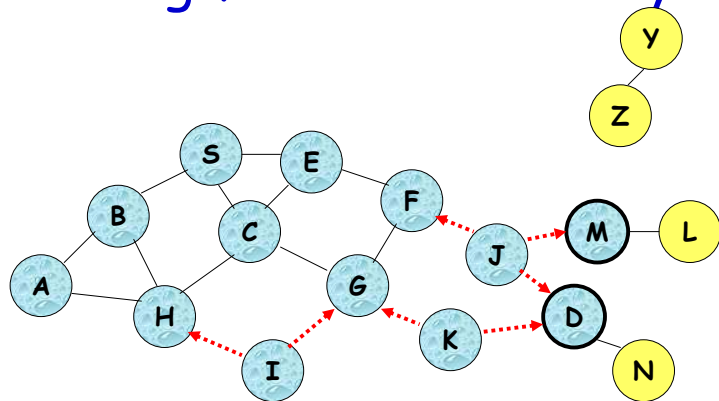
Node H receives packet P from two neighbors:
potential for collision

Flooding for Data Delivery



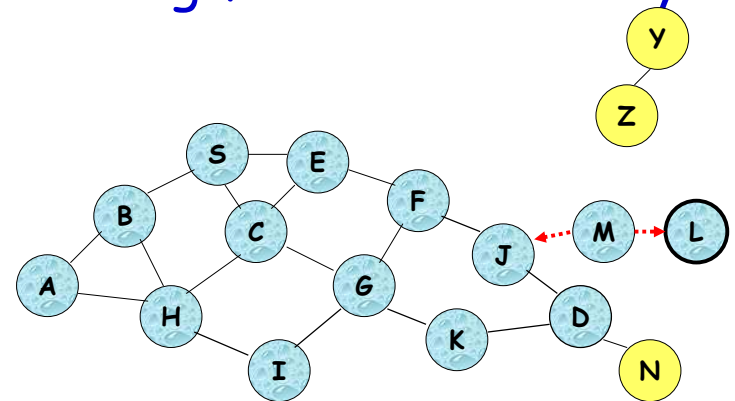
Node C receives packet P from G and H, but does not forward it again, because node C has **already forwarded packet P** once

Flooding for Data Delivery



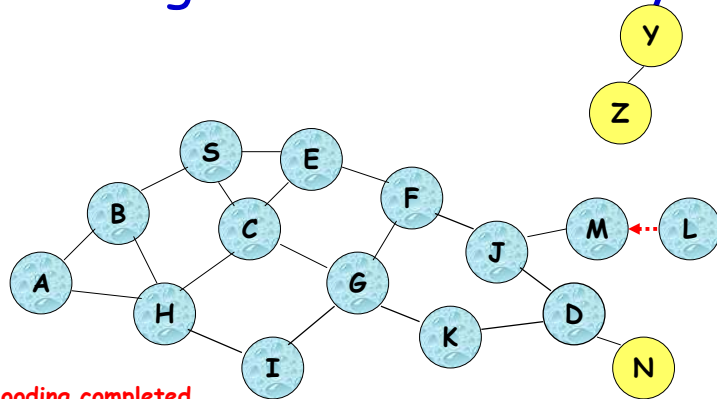
Nodes J and K both broadcast packet P to node D
Since nodes J and K are **hidden** from each other, their transmissions may collide
⇒ Packet P may not be delivered to node D at all, despite the use of flooding!!

Flooding for Data Delivery



• Node D **does not forward** packet P, because node D is the **intended destination** of packet P

Flooding for Data Delivery

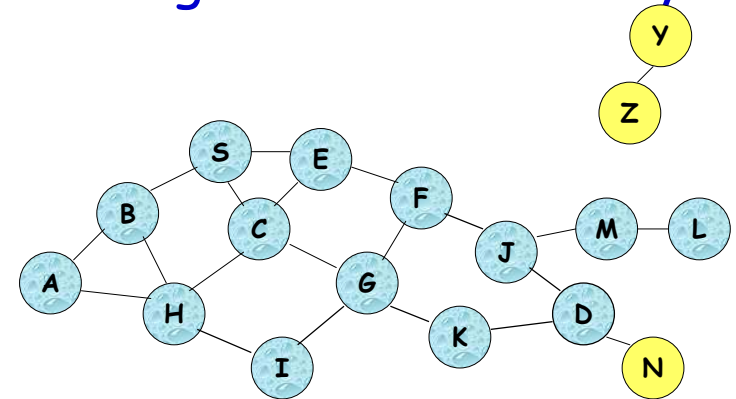


Flooding completed

Nodes **unreachable** from S do not receive packet P (e.g., node Z)

Nodes for which all paths from S go through the destination D also do not receive packet P (example: node N)

Flooding for Data Delivery



Flooding may deliver packets to too many nodes
(in the **worst case**, all nodes reachable from sender may receive the packet)

Flooding for Data Delivery: Advantages

- Simplicity
- May be more efficient than other protocols when rate of information transmission is low enough that the overhead of explicit route discovery/maintenance incurred by other protocols is relatively higher
 - this scenario may occur, for instance, when nodes transmit **small data packets** relatively infrequently, and many topology **changes occur** between consecutive packet transmissions

Flooding for Data Delivery: Advantages (cont.)

- Potentially higher reliability of data delivery
 - Because packets may be delivered to the destination on multiple paths
- **For high mobility patterns, may be the only reasonable choice?**

Flooding for Data Delivery: Disadvantages

- Potentially, very high overhead
 - Data packets may be delivered to too many nodes who do not need to receive them
- Potentially, lower reliability of data delivery
 - Flooding uses broadcasting -- hard to implement reliable broadcast delivery without significantly increasing overhead
 - Broadcasting in IEEE 802.11 MAC is unreliable
 - In our example, nodes J and K may transmit to node D simultaneously, resulting in loss of the packet
 - in this case, destination would not receive the packet at all

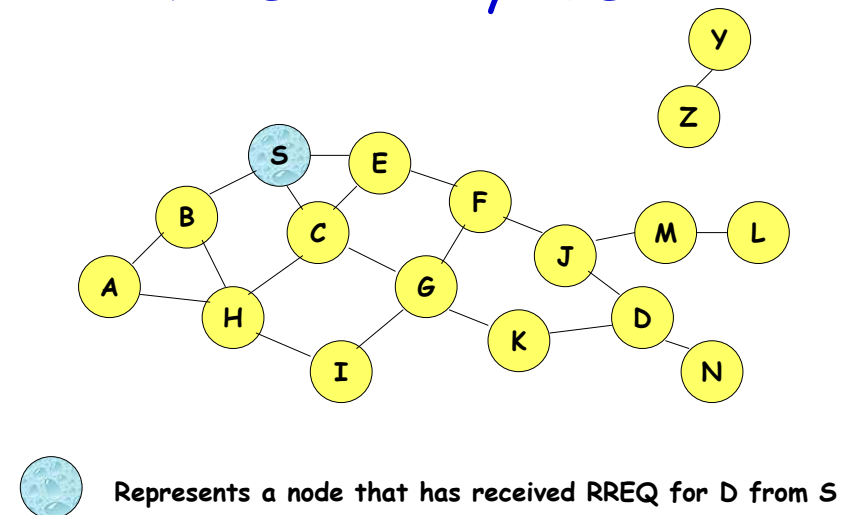
Flooding of Control Packets

- Many protocols perform (potentially *limited*) flooding of **control** packets, instead of **data** packets
- The control packets are used to discover routes
- Discovered routes are subsequently used to send data packet(s)
- Overhead of control packet flooding is amortized over data packets transmitted between consecutive control packet floods

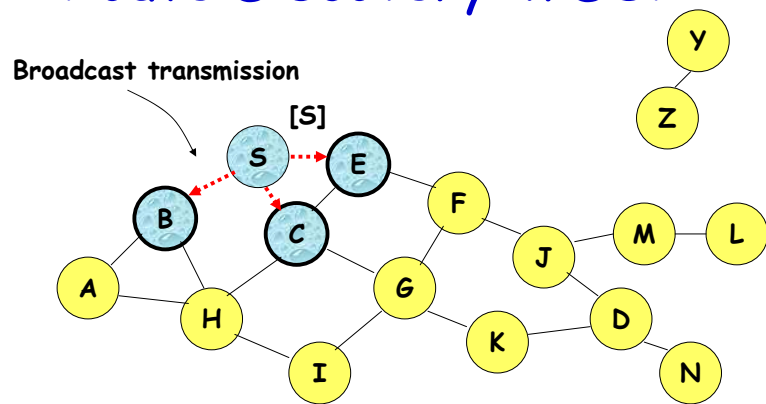
Dynamic Source Routing (DSR)

- When node S wants to send a packet to node D, but does not know a route to D, node S initiates a **route discovery**
- Source node S floods **Route Request (RREQ)**
- Each node **appends own identifier** when forwarding RREQ

Route Discovery in DSR



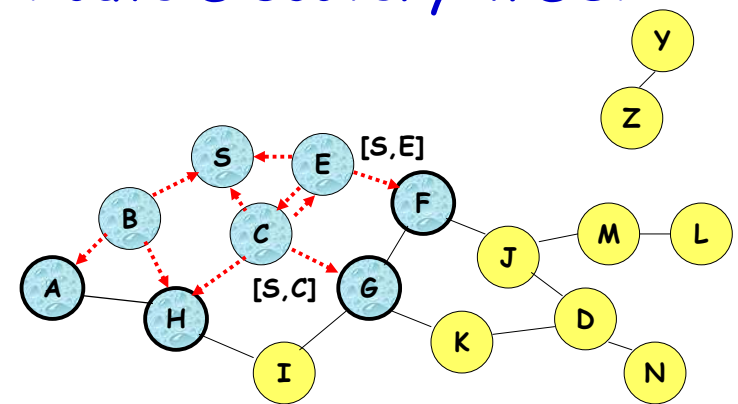
Route Discovery in DSR



.....→ Represents transmission of RREQ

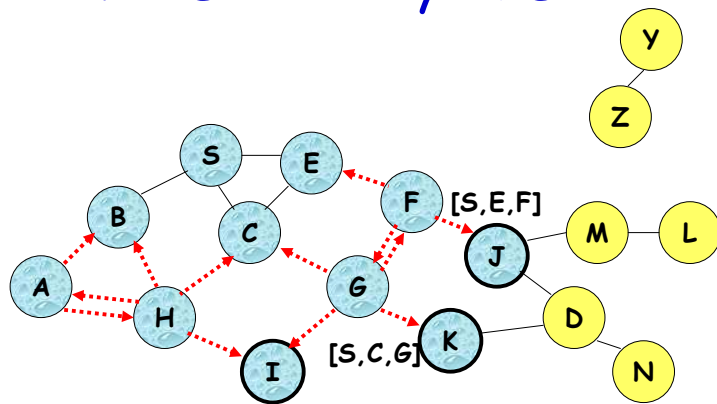
[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR



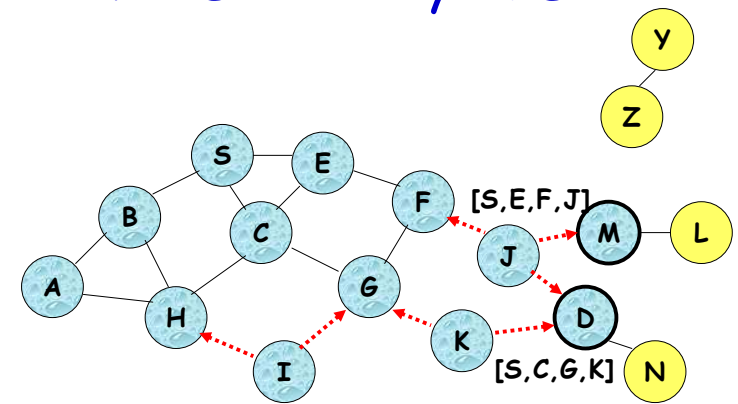
Node H receives packet RREQ from two neighbors:
potential for collision

Route Discovery in DSR



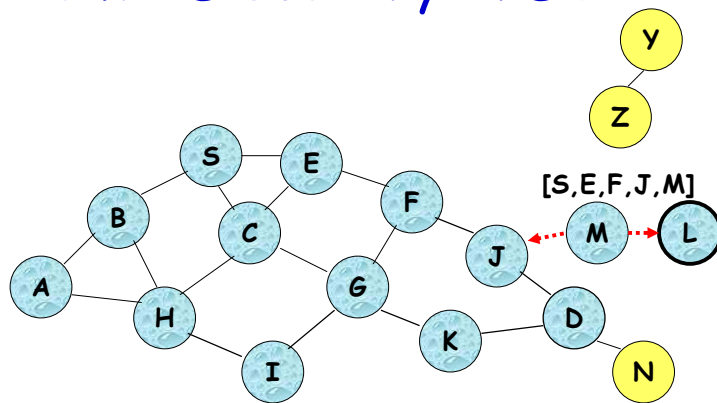
Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ** once

Route Discovery in DSR



Nodes J and K both broadcast RREQ to node D
Since nodes J and K are **hidden** from each other, their **transmissions may collide**

Route Discovery in DSR

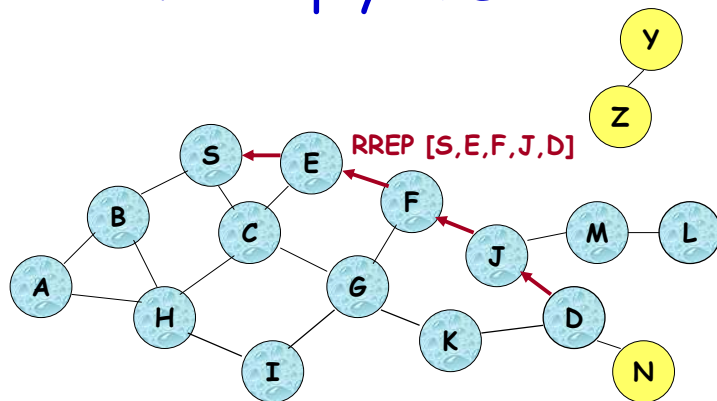


Node D **does not forward** RREQ, because node D is the **intended target** of the route discovery

Route Discovery in DSR

- Destination D on receiving the first RREQ, sends a **Route Reply (RREP)**
- RREP is sent on a route obtained by **reversing** the route appended to received RREQ
- RREP **includes the route** from S to D on which RREQ was received by node D

Route Reply in DSR



← Represents RREP control message

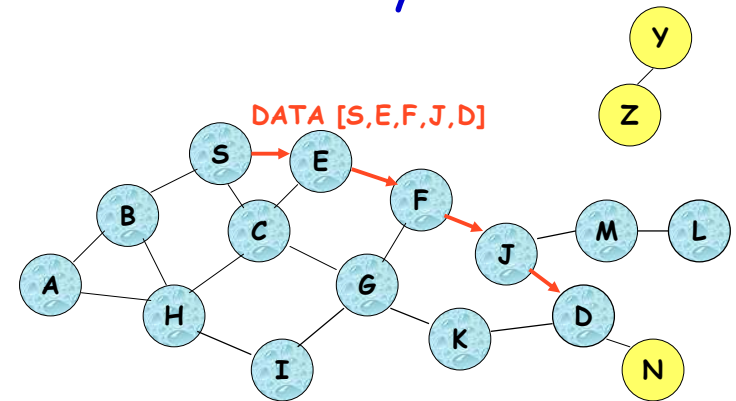
Route Reply in DSR

- Route Reply can be sent by reversing the route in Route Request (RREQ) only if links are **guaranteed to be bi-directional**
 - To ensure this, RREQ should be forwarded only if it received on a link that is known to be bi-directional
- If unidirectional (asymmetric) links are allowed, then RREP may need a route discovery for S from node D
 - Unless node D already knows a route to node S
 - If a route discovery is initiated by D for a route to S, then the Route Reply is piggybacked on the Route Request from D.

DSR: More

- Node S on receiving RREP, caches the route included in the RREP
- When node S sends a data packet to D, the entire route is included in the packet header - hence the name **source routing**
- Intermediate nodes use the **source route** included in a packet to determine to whom a packet should be forwarded
- Note that routes are discovered **ONLY** when a node wants to send data to a node and no route to that node is available

Data Delivery in DSR



Packet header size grows with route length

DSR Optimization: Route Caching

- Each node caches a new route it learns by *any means*
 - E.g., When node S finds **route [S,E,F,J,D]** to node D, node S also learns route [S,E,F] to node F
- When node K receives **Route Request [S,C,G]** destined for node D, node K learns route [K,G,C,S] to node S
- When node F forwards **Route Reply RREP [S,E,F,J,D]**, node F learns route [F,J,D] to node D

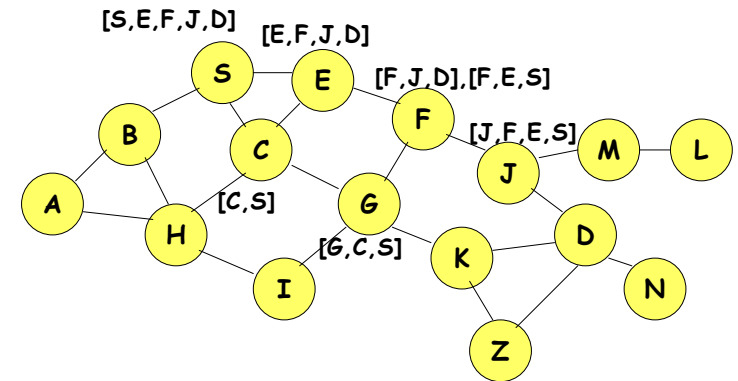
DSR Optimization: Route Caching (cont.)

- When node E forwards **data [S,E,F,J,D]** it learns route [E,F,J,D] to node D
- A node may also learn a route when it overhears data packets, even though it is not directly involved in the transmission
 - Promiscuous Mode

Route Caching (cont.)

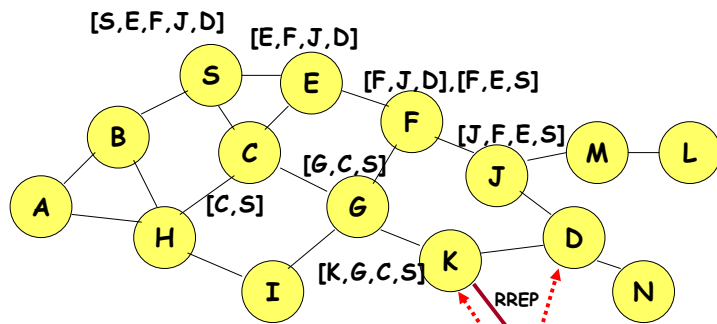
- When node S learns that a route to node D is broken, it uses another route from its local cache, if such a route to D exists in its cache.
- Otherwise, node S initiates route discovery by sending a route request
- Node X on receiving a Route Request for some node D can send a Route Reply directly if node X knows a route to node D
- Use of route cache
 - can speed up route discovery
 - can reduce propagation of route requests

Route Caching (cont.)



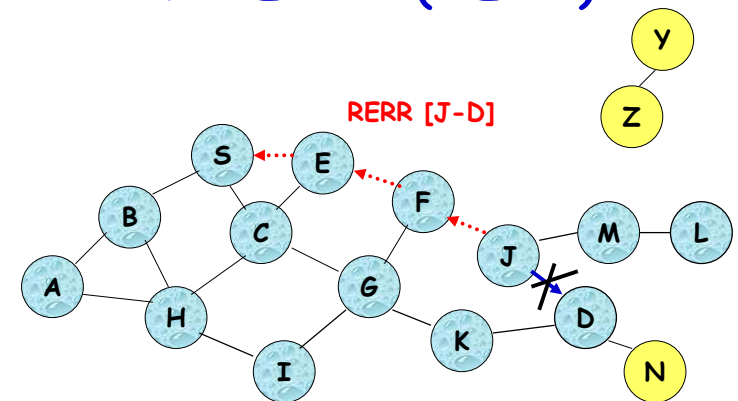
[P,Q,R] Represents cached route at a node
(DSR maintains the cached routes in a tree format)

Route Caching (cont.)



When node Z sends a route request for node C, node K sends back a route reply [Z,K,G,C] to node Z using a locally cached route

Route Error (RERR)



J sends a route error to S along route J-F-E-S when its attempt to forward the data packet S (with route SEFJD) on J-D fails

Nodes hearing RERR update their route cache to remove link J-D

Route Caching: Beware!

- Stale caches can adversely affect performance
- With passage of time and host mobility, cached routes may become invalid
- A sender host may try several stale routes (obtained from local cache, or replied from cache by other nodes), before finding a good route
- It may be more expensive to try several broken routes than to simply discover a new one!!

DSR: Advantages

- Routes maintained only between nodes who need to communicate
 - reduces overhead of route maintenance
- Route caching can further reduce route discovery overhead
- A single route discovery may yield many routes to the destination, due to intermediate nodes replying from local caches

DSR: Disadvantages

- Packet header size grows with route length due to source routing
- Flood of route requests may potentially reach all nodes in the network
- Care must be taken to avoid collisions between route requests propagated by neighboring nodes
 - insertion of random delays before forwarding RREQ

DSR: Disadvantages (cont.)

- Increased contention if too many route replies come back due to nodes replying using their local cache
 - Route Reply Storm problem
 - Reply storm may be eased by preventing a node from sending RREP if it hears another RREP with a shorter route

DSR: Disadvantages (cont.)

- An intermediate node may send Route Reply using a **stale** cached route, thus polluting other caches
- This problem can be eased if some mechanism to purge (potentially) invalid cached routes is incorporated.
 - Static timeouts
 - Adaptive timeouts based on expected rate of mobility

DSR w/ Implicit Source Routing

- Enhancement to make transmission of data packets cheaper
- Don't send entire source route with data packet
- Instead, establish a numbered "flow" between source and destination
- Flow identifier is attached to data packets instead of route

Details of Implicit Source Routing (cont.)

- Replace source route on each data packet with the following tuple:
 - `<source address, destination address, flowID>`
- Source address and destination address can actually be obtained from IP header
- Each node maintains a **flow table** that identifies current flows
- Flows are established with a **flow establishment** packet
 - Contains source route + unique flow ID + timeout

Details of Implicit Source Routing (cont.)

- In addition to forwarding flow packet toward the destination, a forwarding node makes an entry in its flow table
- Entry will be removed from the flow table if a packet doesn't traverse the flow for duration **timeout**
- Subsequent data packets, after initial flow establishment, contain only the tuple previously described

Details of Implicit Source Routing (cont.)

- If a node receives a tuple $\langle \text{source address, destination address, flowID} \rangle$ and does not understand the flowID, then a **FLOW UNKNOWN** error is transmitted toward the sender
- Upon receiving FLOW UNKNOWN, sender must reestablish the flow
- Paper assumes three duplicate flow establishment packets are sent
- Redundancy increases likelihood that at least one flow establishment will be received at each node in the source route

Default Flows

- Can use most recent flow for each $\langle \text{source, destination} \rangle$ as default
- Transmit **no** additional info on data packets
- Nodes on path between source and destination examine TTL of packet to see if they should forward
- If source, destination, TTL match most recent flow, forward as if flow ID were present

Route Shortening

- Illustration:
 - $S \rightarrow A \rightarrow B \rightarrow C \rightarrow D$ is source route
 - C overhears packet when S transmits
 - C sends gratuitous ROUTE REPLY to S , containing $S \rightarrow C \rightarrow D$ to replace old route
- With implicit source routing:
 - Associate a hop count with each flow table entry
 - Hearing a packet with a hop count **less** than expected means gratuitous ROUTE REPLY should be sent

Packet Delivery Ratio

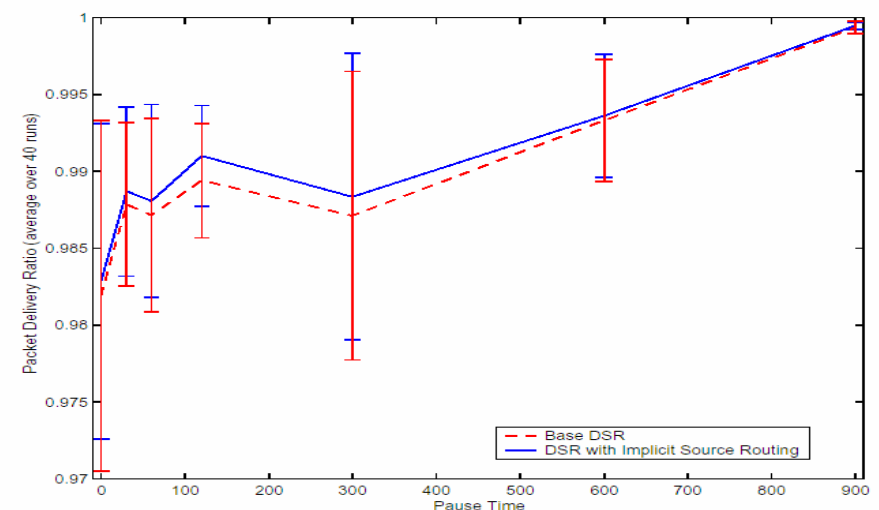


Figure 1 Effect of Implicit Source Routing on Packet Delivery Ratio

Latency

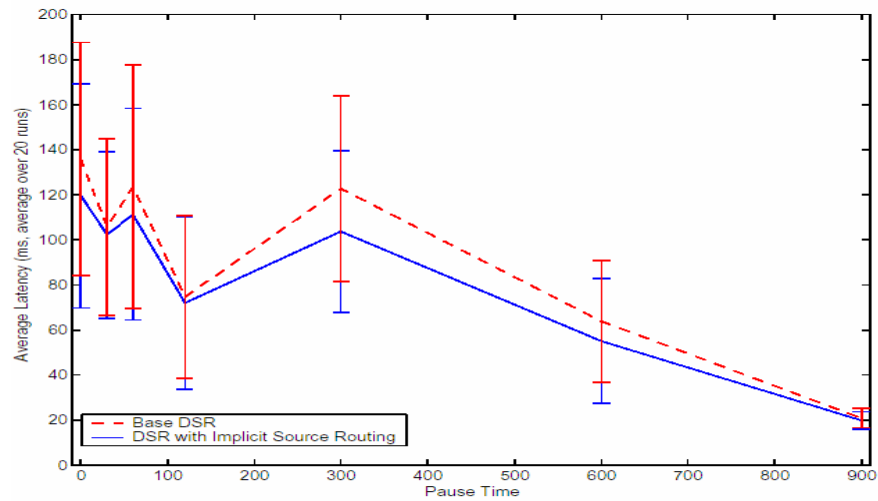


Figure 2 Effect of Implicit Source Routing on Packet Delivery Latency

Path Length (vs. Optimal)

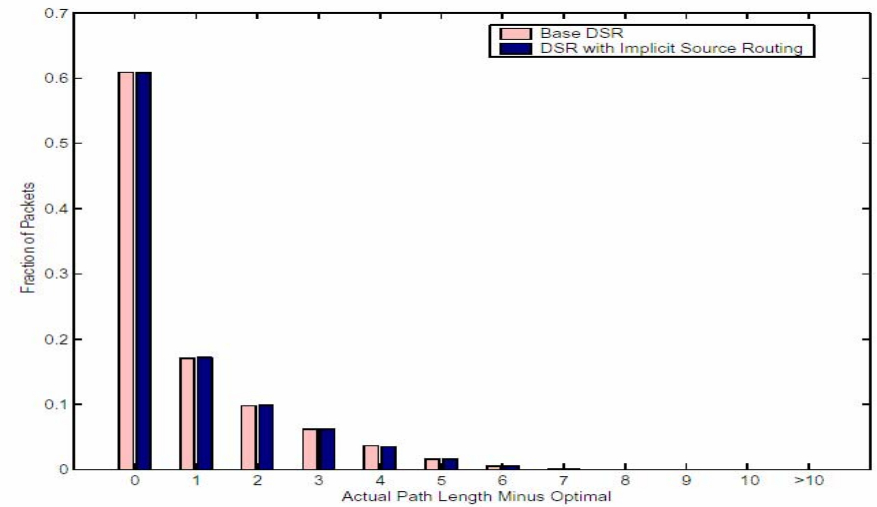


Figure 3 Effect of Implicit Source Routing on Path Length Optimality

Overhead: Routing Packets Only

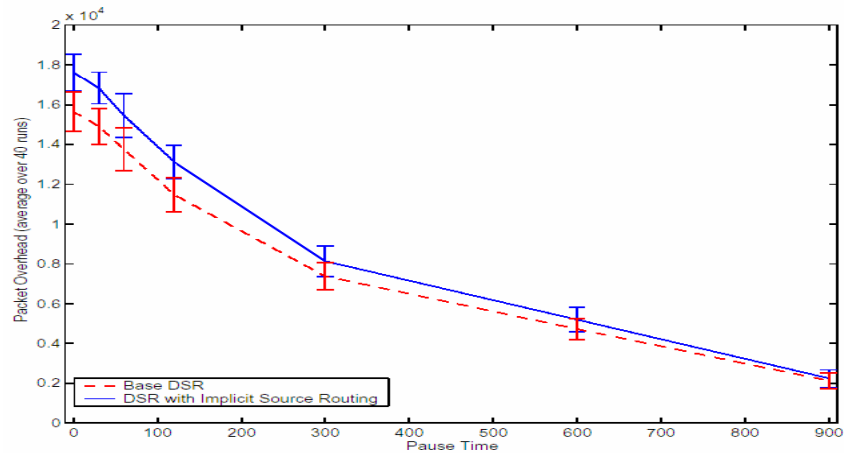


Figure 4 Effect of Implicit Source Routing on Routing Packet Overhead

Routing Overhead (Total)

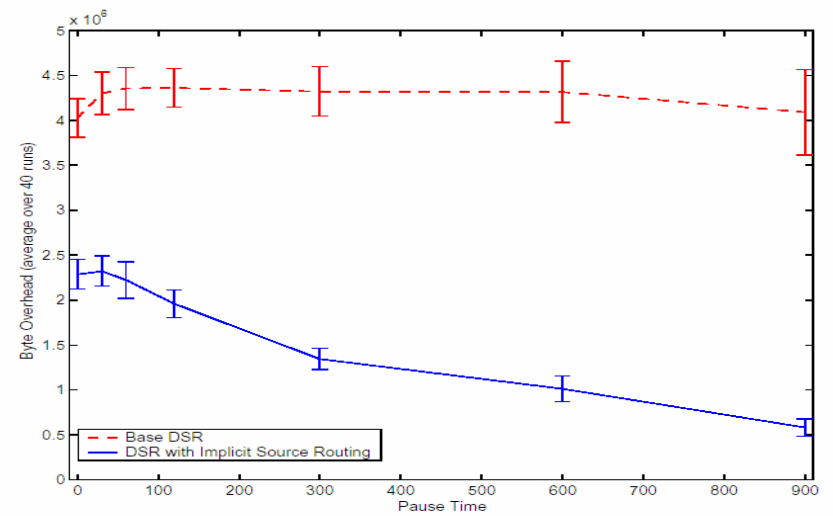


Figure 5 Effect of Implicit Source Routing on Total Bytes of Overhead

Ad Hoc On-Demand Distance Vector Routing (AODV)

- DSR includes source routes in packet headers
- Resulting large headers can sometimes degrade performance
 - particularly when data contents of a packet are small
- AODV attempts to improve on "vanilla" DSR by maintaining routing tables at the nodes, so that data packets do not have to contain routes
- AODV retains the desirable feature of DSR that routes are maintained only between nodes which need to communicate
- Implicit source routing version of DSR has largely removed the improvements that AODV had over vanilla DSR

AODV Message Types

- RREQ (Route Request)
 - Discover a route to a destination
- RREP (Route Reply)
- RERR (Route Error)
 - Propagate info about broken links

AODV: Local Data

- Each node maintains
 - node sequence #
 - broadcast ID (uniquely identifies a particular RREQ)
- Also maintains a routing table, one entry per destination
- Each entry in routing table contains:
 - destination IP
 - next hop IP
 - # of hops to destination
 - Sequence # for destination
 - "precursor" info—who routes THROUGH me to this destination?
 - Expiration time for this entry
- Each time the route entry is used to send data to the destination, the expiration time is pushed forward

AODV: RREQ contents

- Route Requests (RREQ) are forwarded in a manner similar to DSR
- Before sending a RREQ:
 - Source increments the broadcast ID
 - Source increments its source sequence number
- RREQ contains:
 - Source IP
 - Source sequence #
 - Broadcast ID
 - Destination IP
 - Destination sequence #
 - Hop count (initially zero, incremented on each hop toward destination)
 - Flags

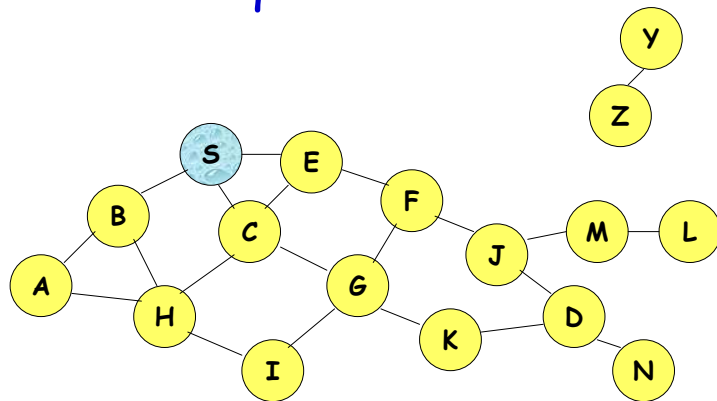
AODV: RREP Contents

- RREP contains:
 - Hop Count
 - Destination IP Address
 - Destination Sequence Number
 - Originator IP Address
 - Lifetime
 - Time in milliseconds for which route associated with RREP is valid
 - Flags

More RREQ

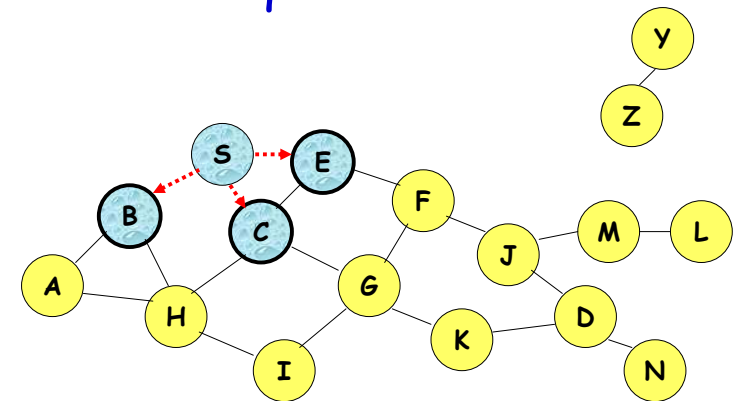
- When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source
 - AODV assumes bi-directional links!
- When the intended destination receives a Route Request, it replies by sending a Route Reply
- Route Reply travels along the reverse path set-up when Route Request is forwarded
- If a node knows a route to the destination, it can reply directly to the node that forwarded the RREQ

Route Requests in AODV



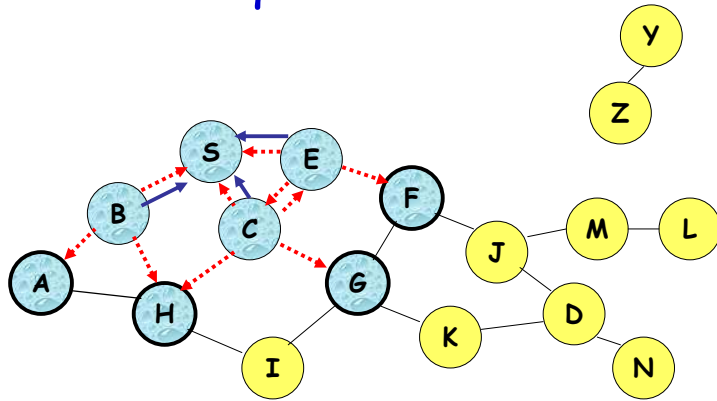
Represents a node that has received RREQ for D from S

Route Requests in AODV



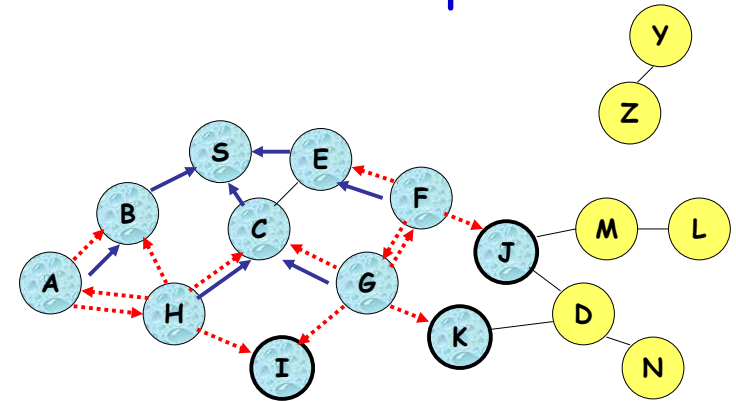
.....→ Represents transmission of RREQ

Route Requests in AODV



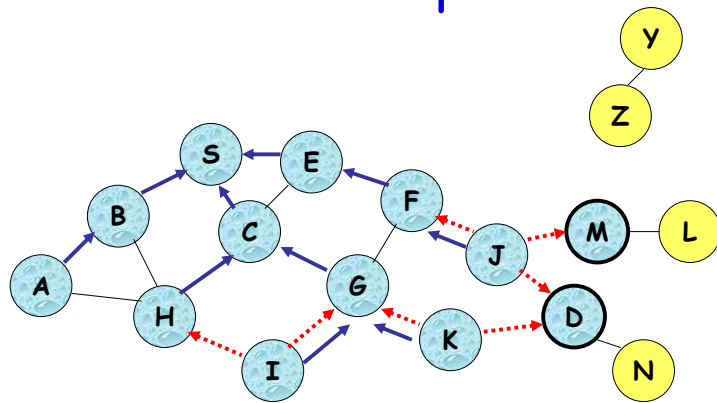
← Represents links on Reverse Path

Reverse Path Setup in AODV

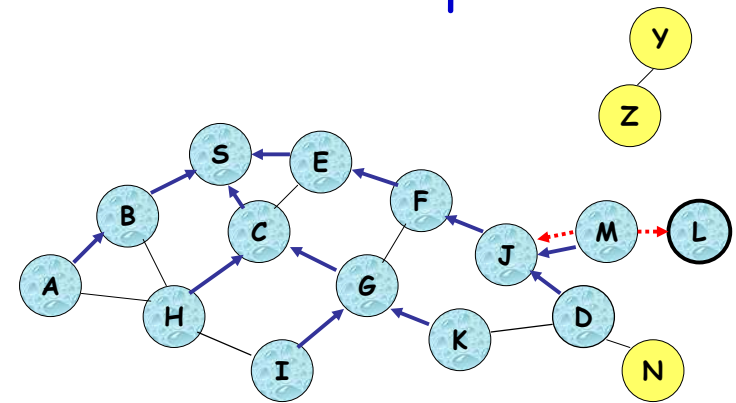


Node C receives RREQ from G and H, but does not forward it again, because node C has **already forwarded RREQ** once

Reverse Path Setup in AODV

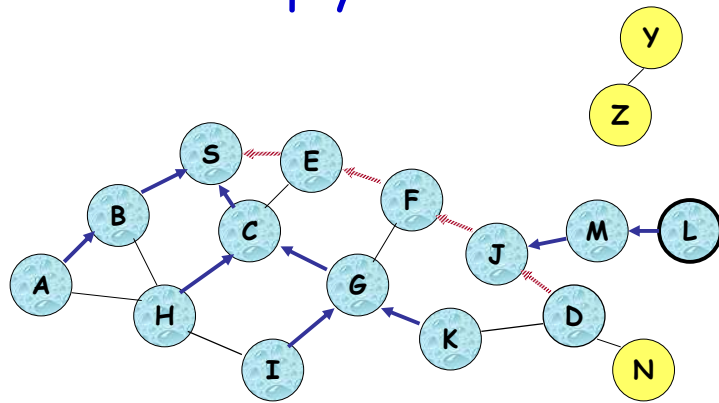


Reverse Path Setup in AODV



• Node D **does not forward** RREQ, because node D is the **intended target** of the RREQ

Route Reply in AODV

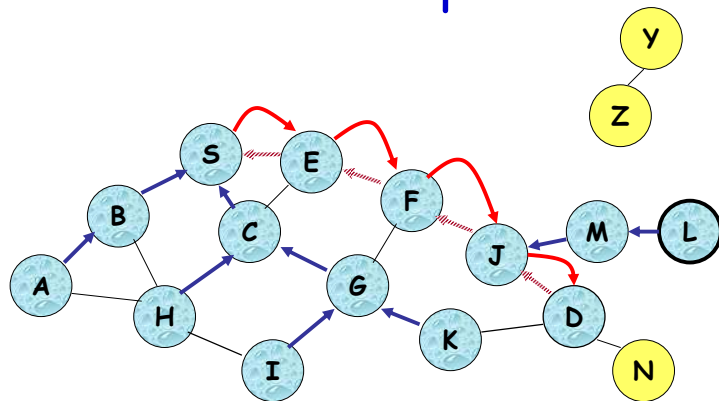


 Represents links on path taken by RREP

Route Reply in AODV

- An intermediate node may also send a Route Reply (RREP) provided that it knows a more recent path than the one previously known to sender S
- To determine whether the path known to an intermediate node is more recent, destination sequence numbers are used
- Works because a new Route Request by node S for a destination is assigned a higher destination sequence number.
- An intermediate node which knows a route, but with a smaller sequence number, cannot send Route Reply

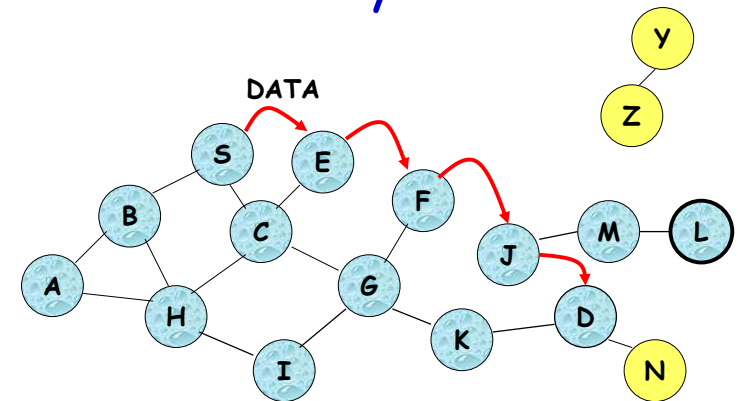
Forward Path Setup in AODV



Forward links are setup when RREP travels along the reverse path

 Represents a link on the forward path

Data Delivery in AODV



Routing table entries used to forward data packet.

Route is *not* included in packet header.

Timeouts

- A routing table entry maintaining a **reverse path** is purged after a timeout interval
 - timeout should be long enough to allow RREP to come back
- A routing table entry maintaining a **forward path** is purged if *not used* for an **active_route_timeout** interval
 - if no data is sent using a particular routing table entry, that entry will be deleted from the routing table (even if the route may actually still be valid)

Link Failure Reporting

- A neighbor of node X is considered active for a routing table entry if the neighbor sent a packet within active_route_timeout interval which was forwarded using that entry
- When the next hop link in a routing table entry breaks, all active neighbors are informed
- Link failures are propagated by means of Route Error (RERR) messages, which also update destination sequence numbers

Route Error (RERR) messages

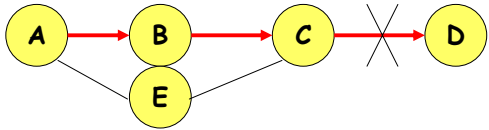
- When node X is unable to forward packet P (from node S to node D) on link (X,Y):
 - X increments the destination sequence number for D in its routing table
 - sets the hop count to D through Y to ∞
 - generates an RERR, which includes the updated sequence number
- Nodes receiving the RERR:
 - set their sequence number for D to the one in the RERR
 - set the hop count for their route to D to ∞
 - this means the known route to D is broken

Link Failure Detection

- Hello messages: Neighboring nodes periodically exchange hello messages
- Absence of hello message is used as an indication of link failure
- Alternatively, failure to receive several MAC-level acknowledgements may be used as an indication of link failure

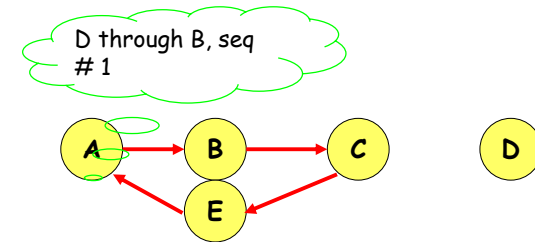
Why Use Sequence Numbers?

- To avoid using old/broken routes
 - To determine which route is newer
- To prevent formation of loops



- Assume that A does not know about failure of link C-D because RERR sent by C is lost
- Now C performs a route discovery for D. Node A receives the RREQ (say, via path C-E-A)
- Node A will reply since A knows a route to D via node B
- Results in a loop (for instance, C-E-A-B-C)

Why Sequence Numbers (cont.)



Loop would look like this—but prevented, because the new RREQ sent by C contains a destination sequence # for D of *2*--this means that the next hop A→B to D can't be used. The sequence number reveals that the route is too old!

Optimization: Expanding Ring Search

- Route Requests are initially sent with small Time-to-Live (TTL) field, to limit their propagation
 - DSR also includes a similar optimization
- If no Route Reply is received, then larger TTL tried

Summary: AODV Unicast

- Routes need not be included in packet headers
- Nodes maintain routing tables containing entries only for routes that are in active use
- At most one next-hop per destination maintained at each node
 - DSR may maintain several routes for a single destination
- Unused routes expire even if topology does not change

Simulation: AODV vs. DSR

- Documented in "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks"
- Vanilla DSR (**NOT** implicit source routes)
- Comparison under a number of different traffic loads and mobility parameters
- *ns-2* simulation package
- 2Mb/sec 802.11, 250m range
- Send queue of 64 packets
- 30seconds max to find a route before it is dropped from the send queue

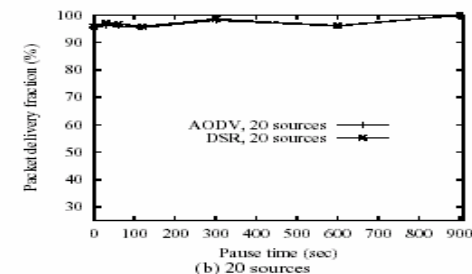
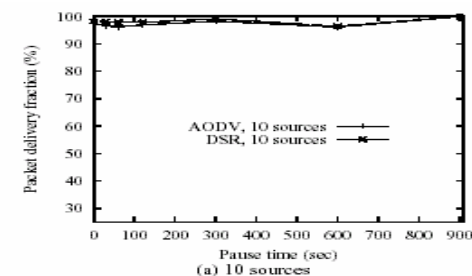
Mobility Model

- "Random waypoint" model
 - Initial random location
 - Choose destination on grid
 - "Move" to destination at speed of 0-20m/sec
 - Pause
 - Choose new destination and repeat
- 1500x300 w/ 50 nodes
 - 900 seconds
 - 4 512-byte packets/sec (except for 40 senders, 3 per sec)
- 2200x600 w/ 100 nodes
 - 500 seconds
 - 4 512-byte packets/sec for 10/20 senders, 2 per sec for 40 senders
- Some limitations imposed by resource requirements of the simulator

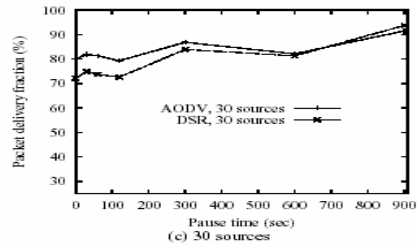
Interesting Metrics

- Delivery fraction (% of packets delivered successfully)
- Average end-to-end delay (# secs to deliver a data packet)
- "Normalized routing load"
 - # of packets (routing, etc.) per data packet delivered
 - E.g., each hop in a RREQ counts as one packet

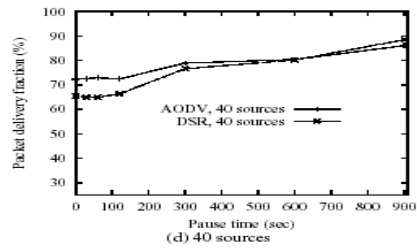
50 nodes, packet delivery %



50 nodes, packet delivery %

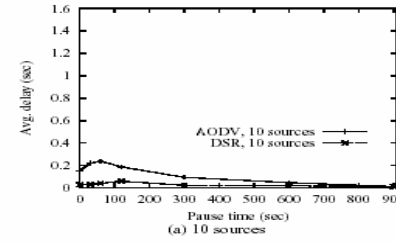


30 sources

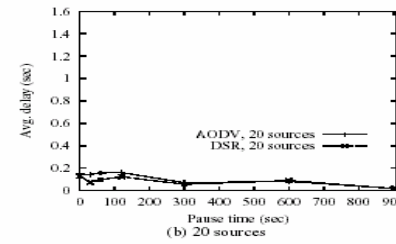


40 sources

50 nodes, delay

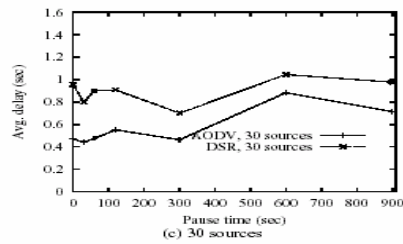


10 sources

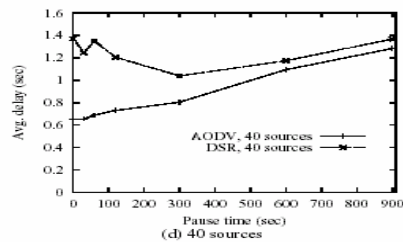


20 sources

50 nodes, delay

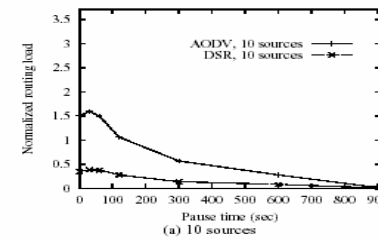


30 sources

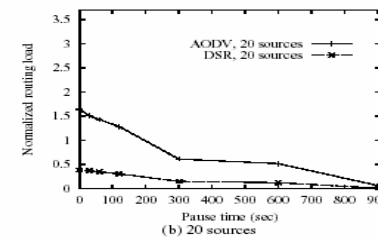


40 sources

50 nodes, routing load

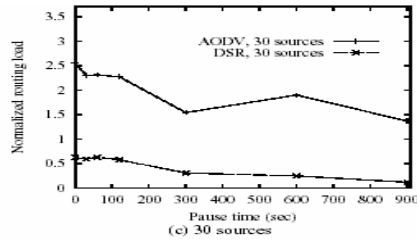


10 sources

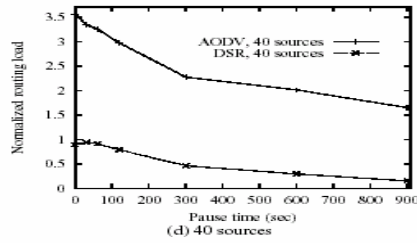


20 sources

50 nodes, routing load

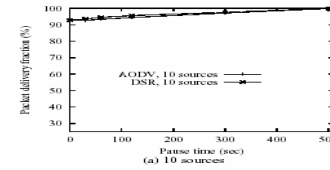


30 sources

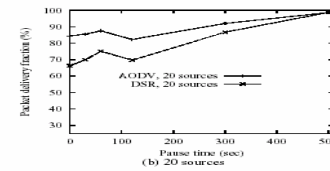


40 sources

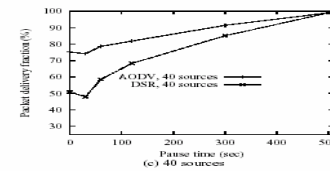
100 nodes, packet delivery %



10 sources

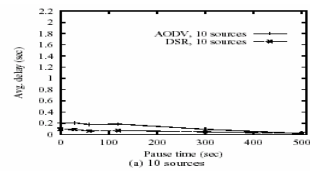


20 sources

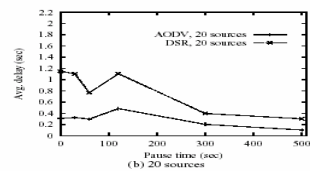


40 sources

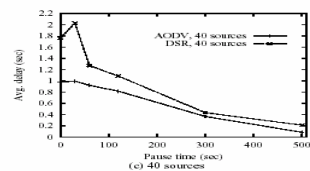
100 nodes, delay



10 sources

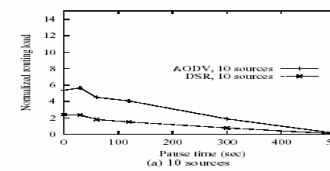


20 sources

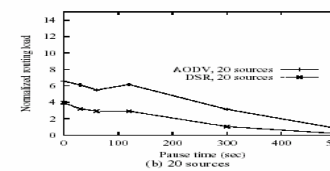


40 sources

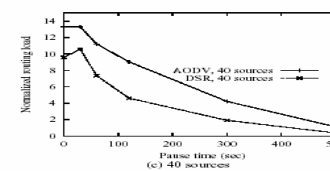
100 nodes, routing load



10 sources



20 sources



40 sources

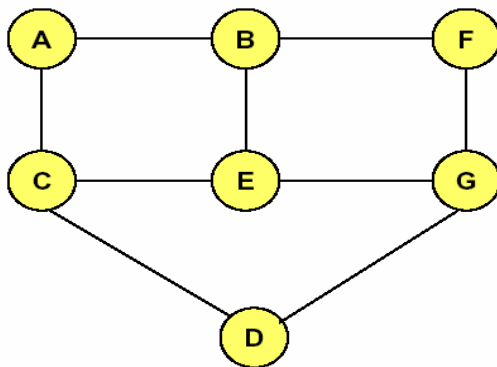
Observations

- More "stressful" situations:
 - AODV better in terms of packet delivery %
 - DSR attempts to use too many stale routes
 - DSR delay much higher
 - DSR loses many packets because route discovery takes too long—send queue fills, drops packets
- Less "stressful" situations:
 - DSR better. Route caching good.
 - DSR delay comparable to AODV

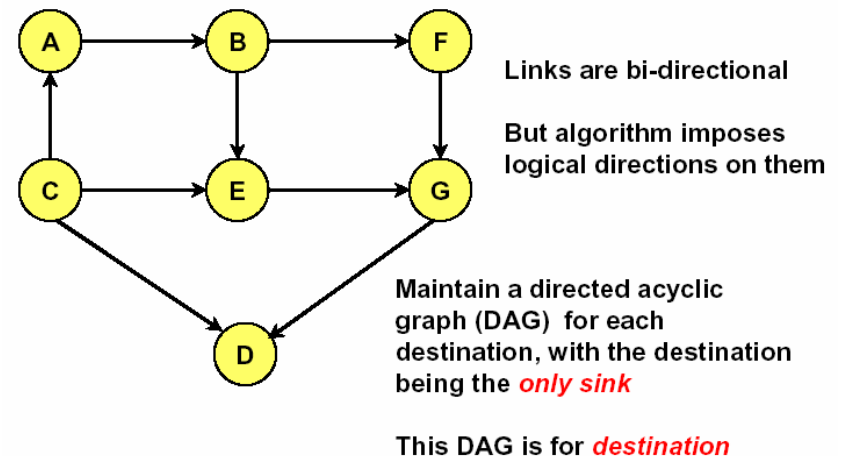
Observations (cont.)

- DSR almost always has better routing load (in terms of # packets!)
- DSR caches routes, so can eliminate many full route request/reply cycles
- DSR does generate a lot of RREPs..so in terms of # bytes transmitted, difference from AODV is not so great

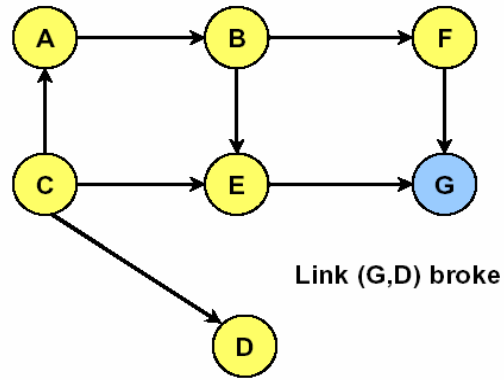
Link Reversal Algorithm



Link Reversal Algorithm

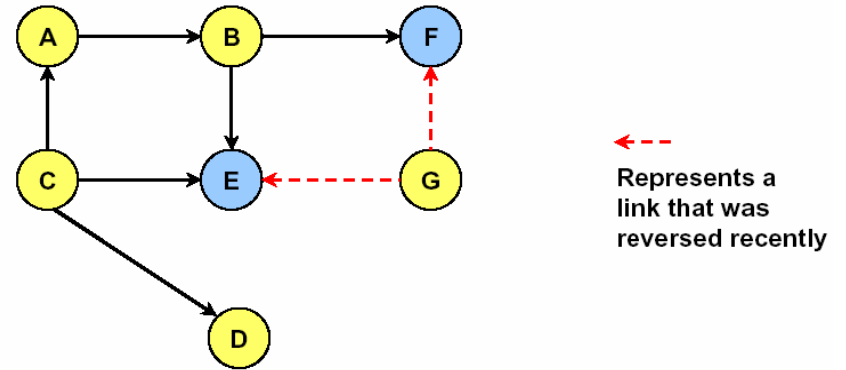


Link Reversal Algorithm



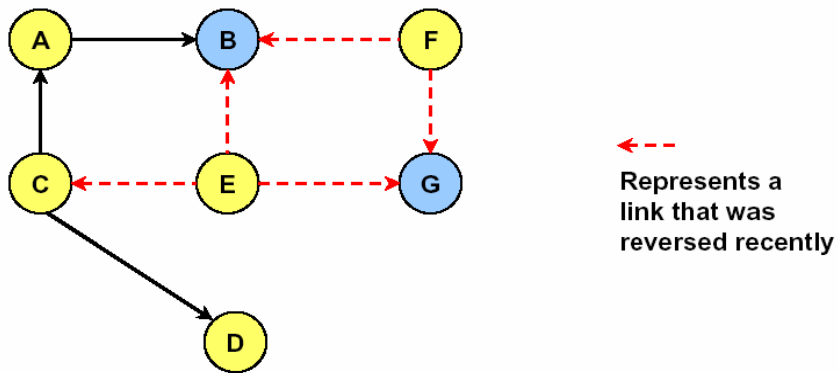
Any node, **other than the destination**, that has no outgoing links reverses all its incoming links.

Link Reversal Algorithm



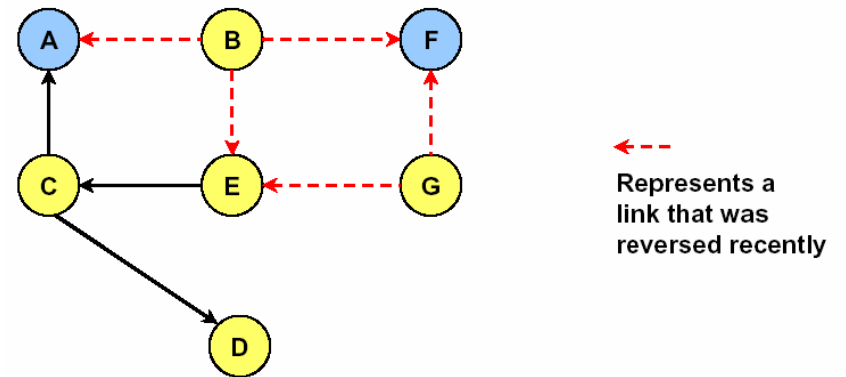
Now nodes E and F have no outgoing links

Link Reversal Algorithm



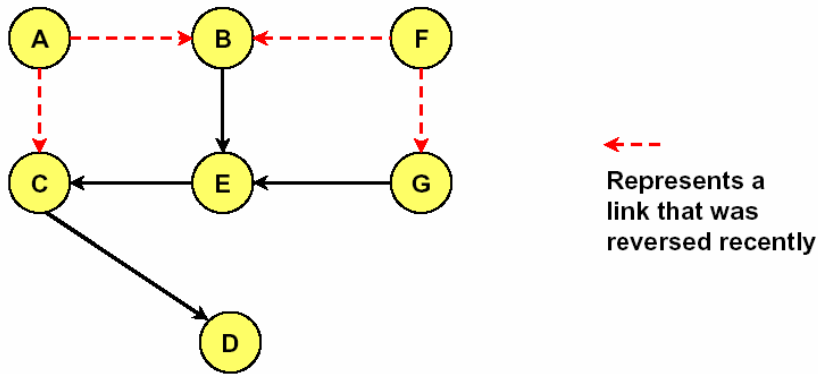
Now nodes B and G have no outgoing links

Link Reversal Algorithm



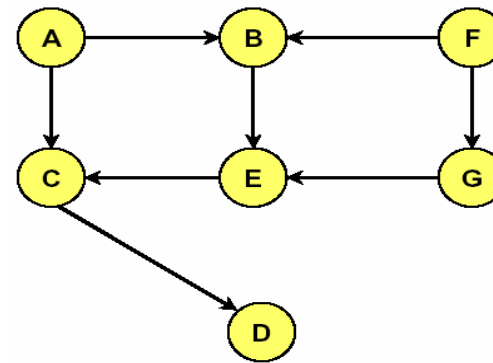
Now nodes A and F have no outgoing links

Link Reversal Algorithm



Now all nodes (other than destination D) have an outgoing link

Link Reversal Algorithm



DAG has been restored with only the destination as a sink

Link Reversal Algorithm

- Attempts to keep link reversals local to where the failure occurred
 - But this is not guaranteed
- When the first packet is sent to a destination, the destination oriented DAG is constructed
- The initial construction does result in flooding of control packets

Link Reversal Methods

- Advantages
 - Link reversal methods attempt to limit updates to routing tables at nodes in the vicinity of a broken link
 - Each node may potentially have multiple routes to a destination
- Disadvantages
 - Need a mechanism to detect link failure
 - hello messages may be used
 - If network is partitioned, link reversals continue indefinitely

Temporally-Ordered Routing Algorithm (TORA)

- Route optimality is considered of secondary importance; longer routes may be used
- At each node, a logically separate copy of TORA is run for each destination, that computes the height of the node with respect to the destination
- Height captures number of hops and next hop

Temporally-Ordered Routing Algorithm (cont.)

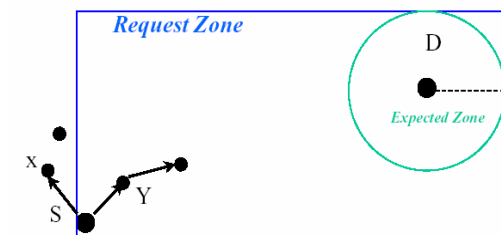
- Route discovery is by using query and update packets
- TORA modifies the partial link reversal method to be able to detect partitions
- When a partition is detected, all nodes in the partition are informed, and link reversals in that partition cease

Location-Aided Routing (LAR)

- Exploits location information to limit scope of route request flood
 - Location information may be obtained using GPS
- *Expected Zone* is determined as a region that is expected to hold the current location of the destination
 - Expected region determined based on potentially old location information, and knowledge of the destination's speed
- Route requests limited to a *Request Zone* that contains the Expected Zone and location of the sender node

Request Zone

- Define a Request Zone
- LAR is same as flooding, except that only nodes in request zone forward route request
- Smallest rectangle including S and expected zone for D



Location Aided Routing (LAR)

- Advantages
 - reduces the scope of route request flood
 - reduces overhead of route discovery
- Disadvantages
 - Nodes need to know their physical locations
 - Does not take into account possible existence of obstructions for radio transmissions

Zone Routing Protocol (ZRP)

- ZRP combines proactive and reactive approaches
- All nodes within hop distance at most d from a node X are said to be in the routing zone of node X
- All nodes at hop distance exactly d are said to be peripheral nodes of node X 's routing zone
- Intra-zone routing: Proactively maintain routes to all nodes within the source node's own zone.
- Inter-zone routing: Use an on-demand protocol (similar to DSR or AODV) to determine routes to outside zone.

Zone Routing Protocol (ZRP)

