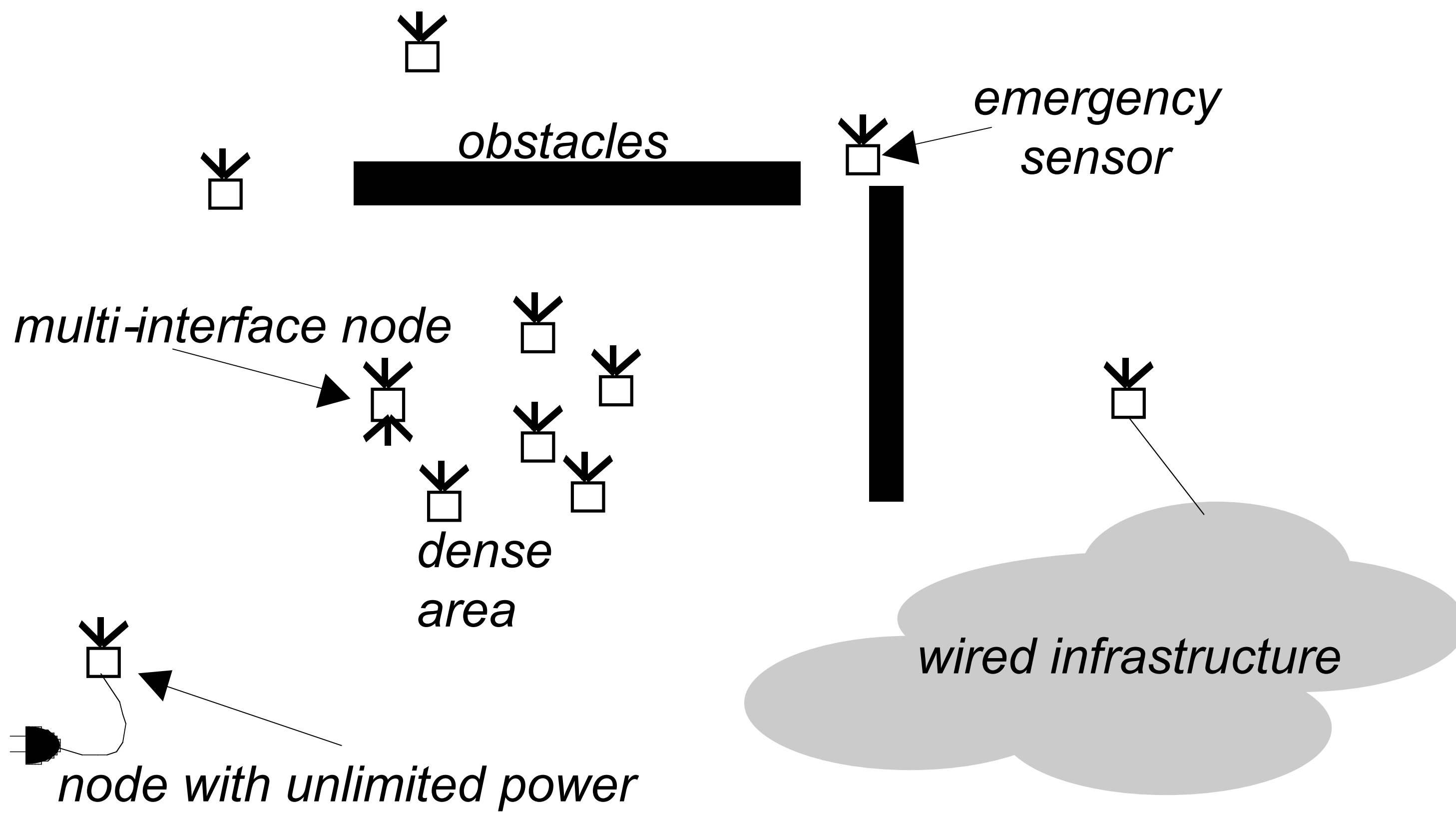
Multi-hop Wireless Networks

Infrastructure vs. multi-hop

- Infrastructure networks:
  - One or several Access-Points (AP) connected to the wired network
  - Mobile nodes communicate through the AP
- Ad hoc network:
  - Mobile nodes communicate directly with each other
  - Multi-hop ad hoc networks: all nodes can also act as routers
- Hybrid (nodes relay packets from AP):
  - Goal: increase capacity, reduce power consumption, and guarantee a minimum service


obstacles
emergency
sensor
multi-interface node
dense
area
node with unlimited power
wired infrastructure

Constraints

- Limited radio spectrum
- Broadcast medium (collisions)
- Limited power available at the nodes
- Limited storage
- Connection requirements (e.g., delay, packet loss)
- Unreliable network connectivity (depends on the channel)
- Dynamic topology (i.e., mobility of nodes, nodes failing or temporarily unavailable)
- Need to provide a full coverage
- Need to enforce fairness

Approaches

- Physical layer:
  - Coding/modulation schemes
  - Smart antennas and MIMO systems
  - Multiple RF interfaces (multiple IF characteristics)
- MAC layer:
  - Controlling transmission power level
  - Packet scheduling schemes
- Network layer:
  - Packet fragmentation
  - Reactive packet routing schemes
  - Clustering and backbone formation
- Planning of the fixed nodes location
- Application-specific optimizations

Adaptivity and Cooperation

- Classical networking stacks have only minimum interaction between adjacent layers
- Multi-hop wireless ad hoc networks require more cooperation between layers because:
  - Channel variation and network topology changes affect the application
  - Routing in a multi-hop considerably affects the medium access control (MAC) performance
  - Collisions and channel fading affect both the physical layer and the MAC
  - Battery power has implications on all layers

Adaptive Coding

- Example:
  - o 1/2 rate convolutional code versus uncoded communication
  - o Channel with two states: E_b/N_0 = 6.8 dB or 11.3 dB (AWGN), L=200 Bytes

E_b/N_0	BER		FER		Nb_Transmit		Total_Tx_Bytes	
	UC	½ CC	UC	½ CC	UC	½ CC	UC	½ CC
6.8dB	10^{-3}	10^{-7}	0.8	$1.6\ 10^{-4}$	5	~ 1	5*200	2*200
11.3dB	10^{-7}	~ 0	$1.6\ 10^{-4}$	~ 0	~ 1	~ 1	200	2*200

- Need to estimate the channel and adapt to it
- Differentiate between congestion and a bad channel condition

Adaptive Fragmentation

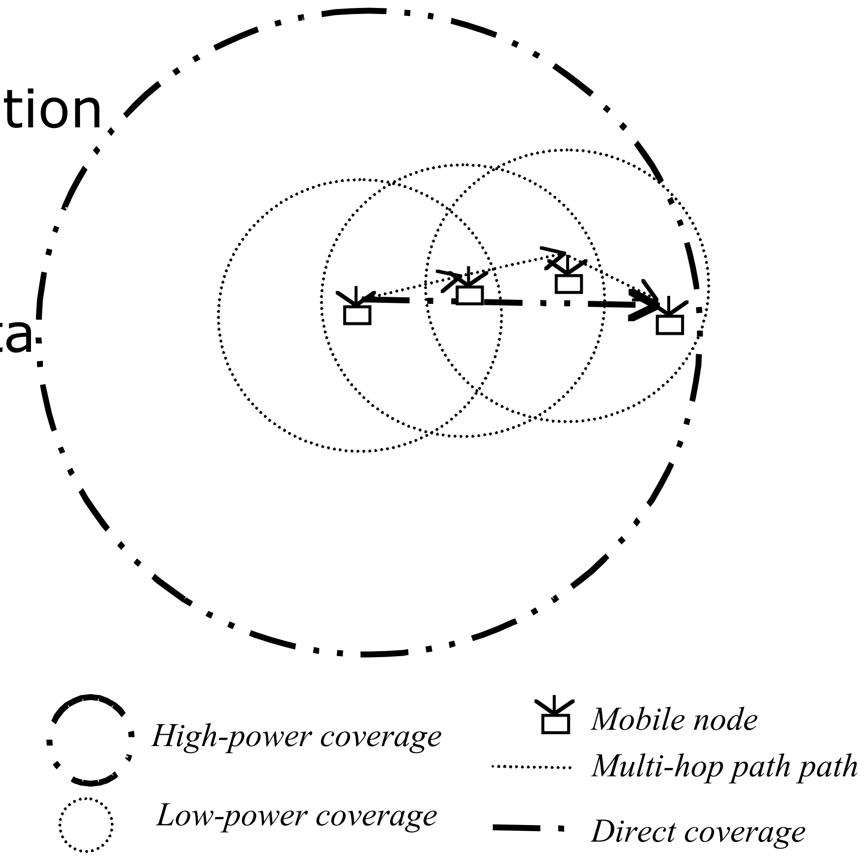
- Example:
  - To transmit a frame of length 200 Bytes, we can fragment into 4 frames of length 50 Bytes (+ 10 Bytes overhead)

BER	FER		Nb_Transmit		Total_Tx_Bytes (incl. overhead)	
	L=60B	L=200B	L=60B	L=200B	L=60B	L=200B
10^{-3}	0.38	0.8	1.6	5	384	1000
10^{-7}	~ 0	~ 0	~ 1	~ 1	240	200

- Need to estimate the channel and adapt to it

Multiple Power Levels

- Using multi-hop transmission (*h* hops) and reducing the transmission power accordingly
  - Increases capacity (factor of *h*)
  - Reduces overall power consumption (by a factor of *h*)
- In asymmetric environments
  - Low power node can encode data and transmit it at low power



Parameters of IEEE 802.11

- IEEE 802.11 has three mechanisms that can be used to improve performance under dynamic channels:
  - Fragmentation (also used to avoid collision)
  - Multiple coding/modulation schemes (8 schemes)
  - 8 power levels

- Multiple coding/modulation schemes are available with 802.11a products over 5GHz
- Currently parameters are statically configured

Problems in Multi-Hop Routing

- Routing:
  - How to maintain up-to-date information on the network topology?
  - How to determine number of hops
  - How to estimate buffer size
- Higher delay
- Risk of congestion on nodes

Existing Unicast Routing Protocols

- Proactive Routing:
  - *"keep routing information current at all times"*
  - good for static networks
  - examples: distance vector (DSDV), link state (LS) algorithms
- Reactive Routing:
  - *"find a route to the destination only after a request comes in"*
  - good for more dynamic networks
  - examples: AODV, dynamic source routing (DSR), TORA
- Hybrid Schemes:
  - *"keep some information current"*
  - example: Zone Routing Protocol (ZRP)
  - example: Use spanning trees for non-optimal routing
- Geometric routing:
  - Assume location-awareness
  - Take advantage of the geometry of plane
  - Example: GPSR
- We will survey some of the popular and well-studied ad hoc network routing protocols:
  - Some slides are based on a tutorial by Nitin Vaidya (UIUC)

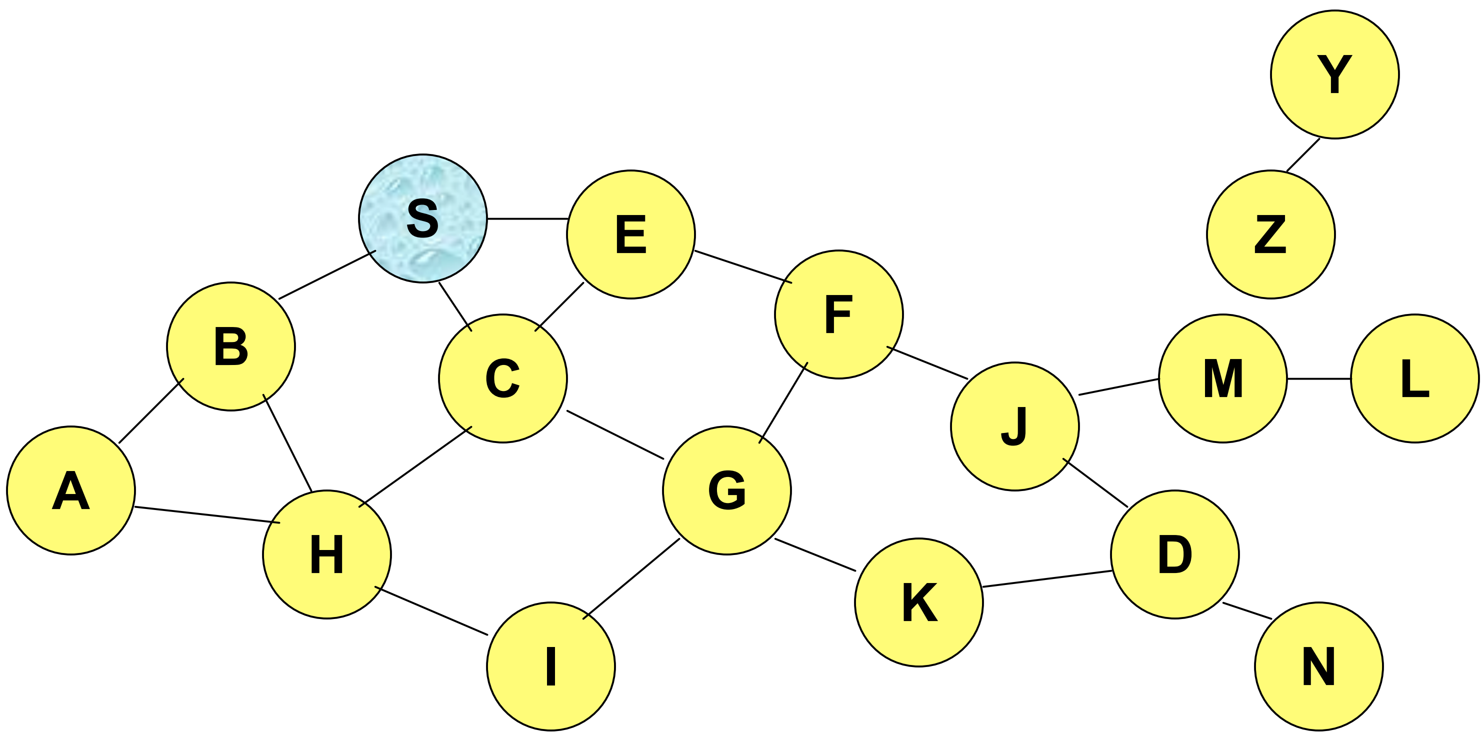
Proactive vs Reactive Routing

- 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 trade-off depends on the traffic and mobility patterns

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

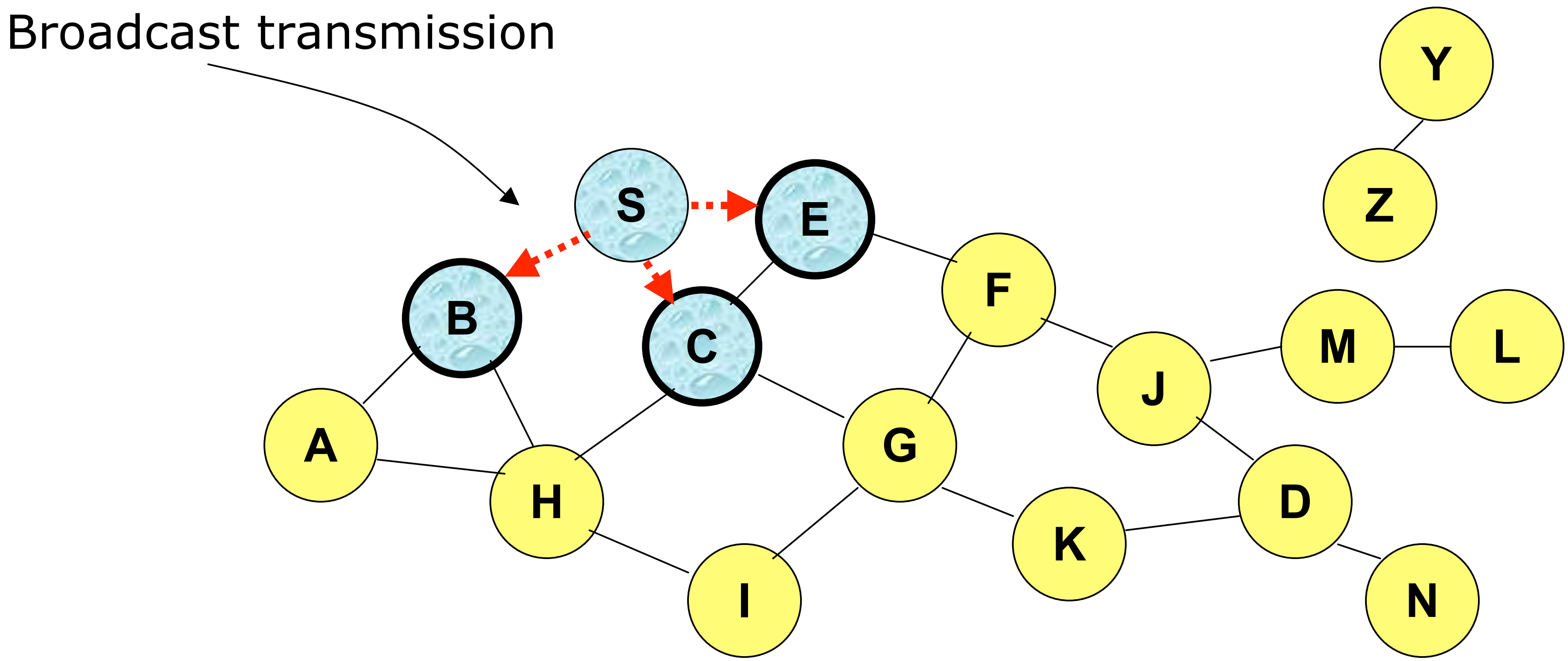
Flooding for Data Delivery



Represents a node that has received packet P

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

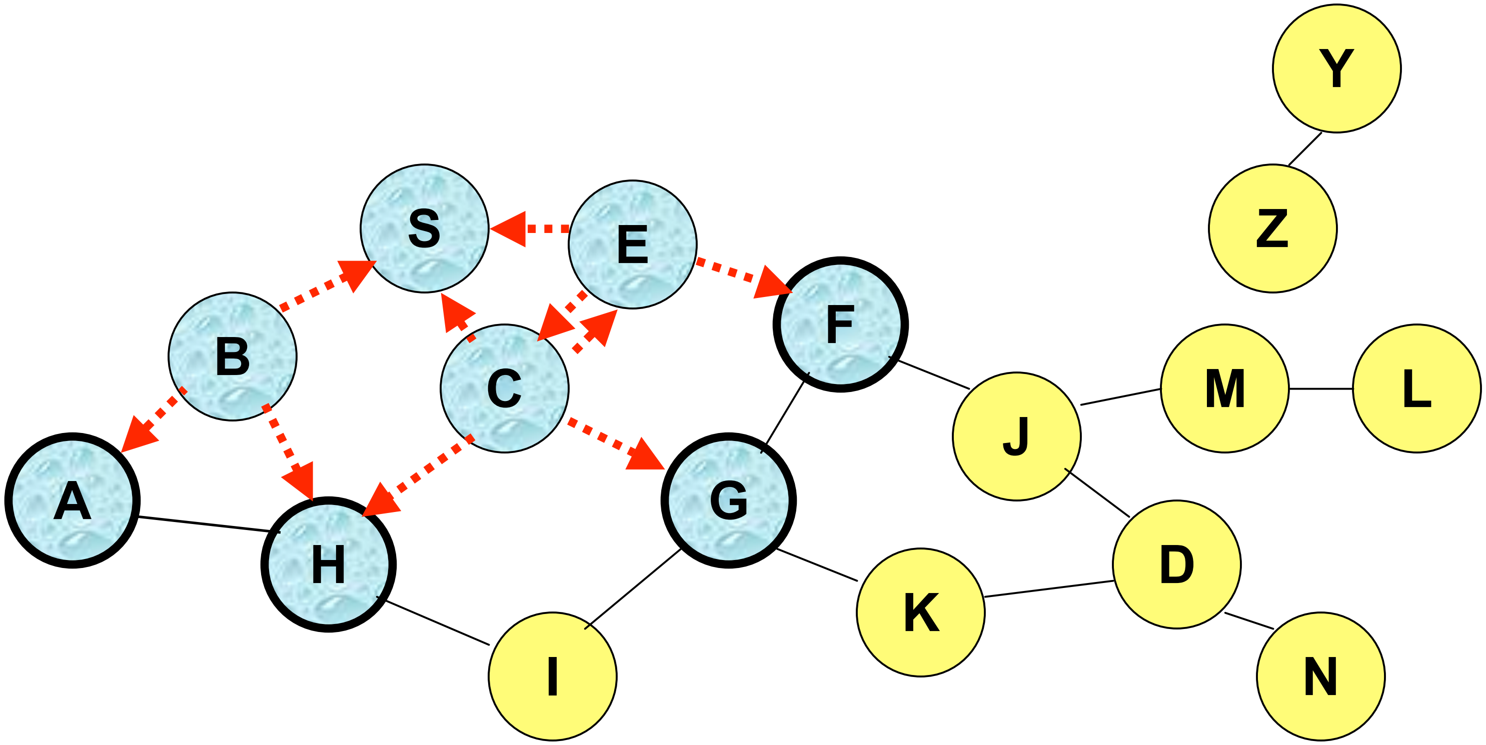
Flooding for Data Delivery



Represents a node that receives packet P for the first time

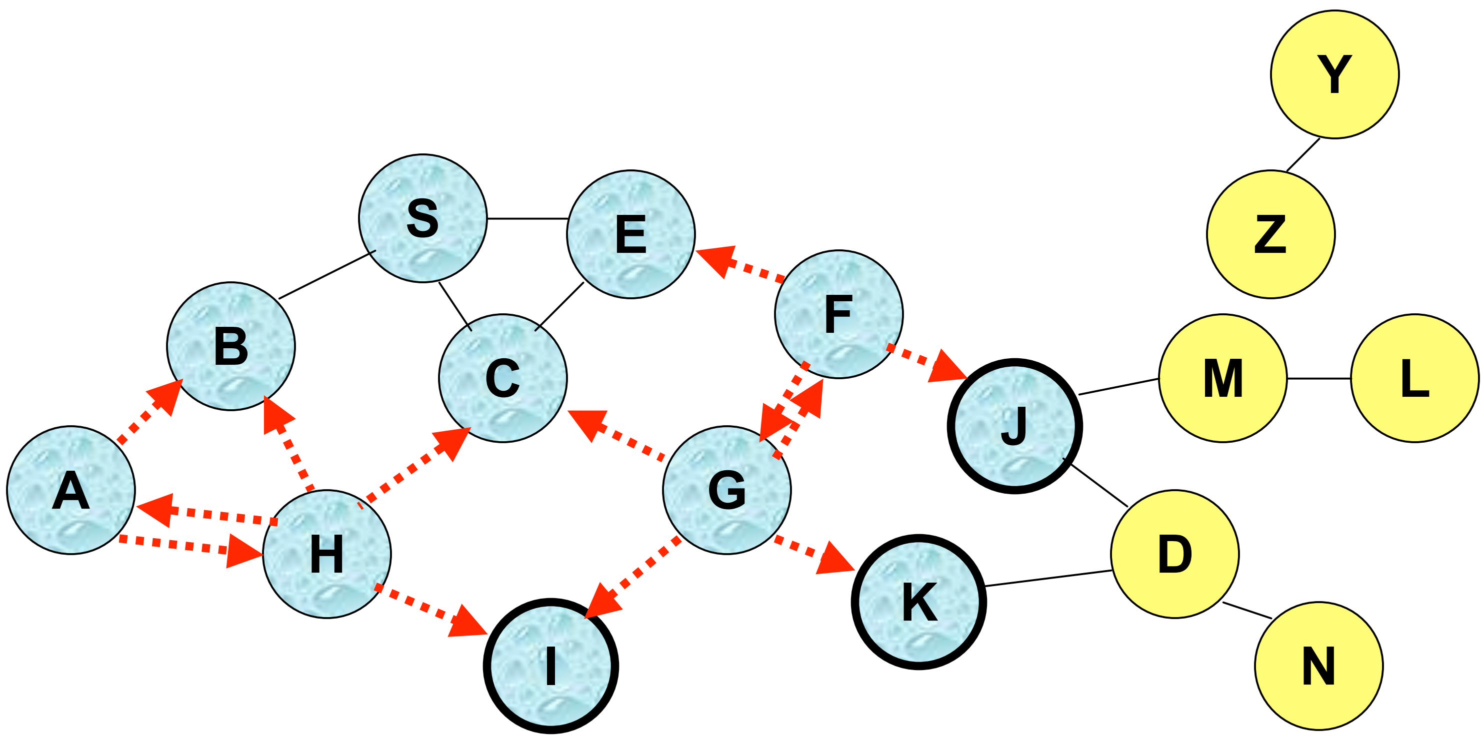
Represents transmission of packet P

Flooding for Data Delivery



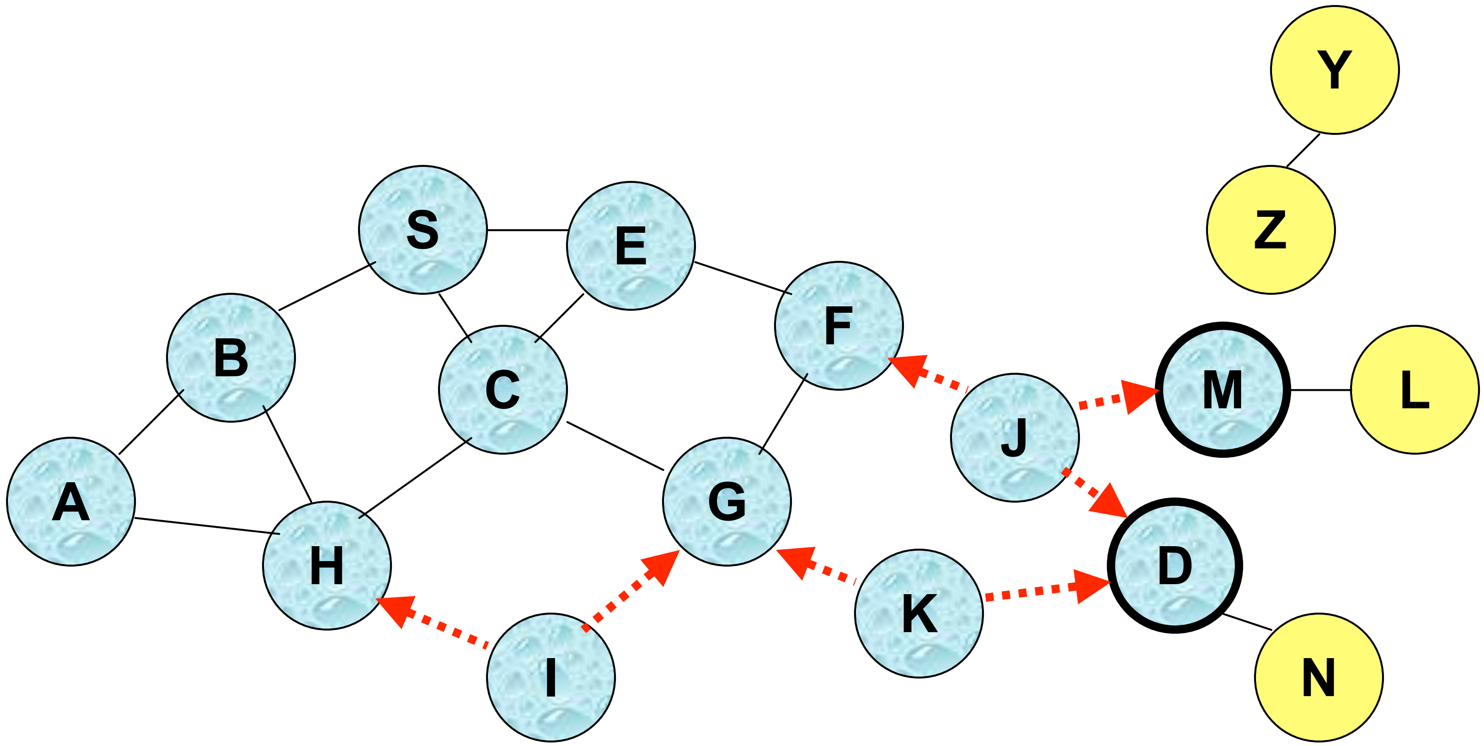
- Node H receives packet P from two neighbors: potential for collision

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

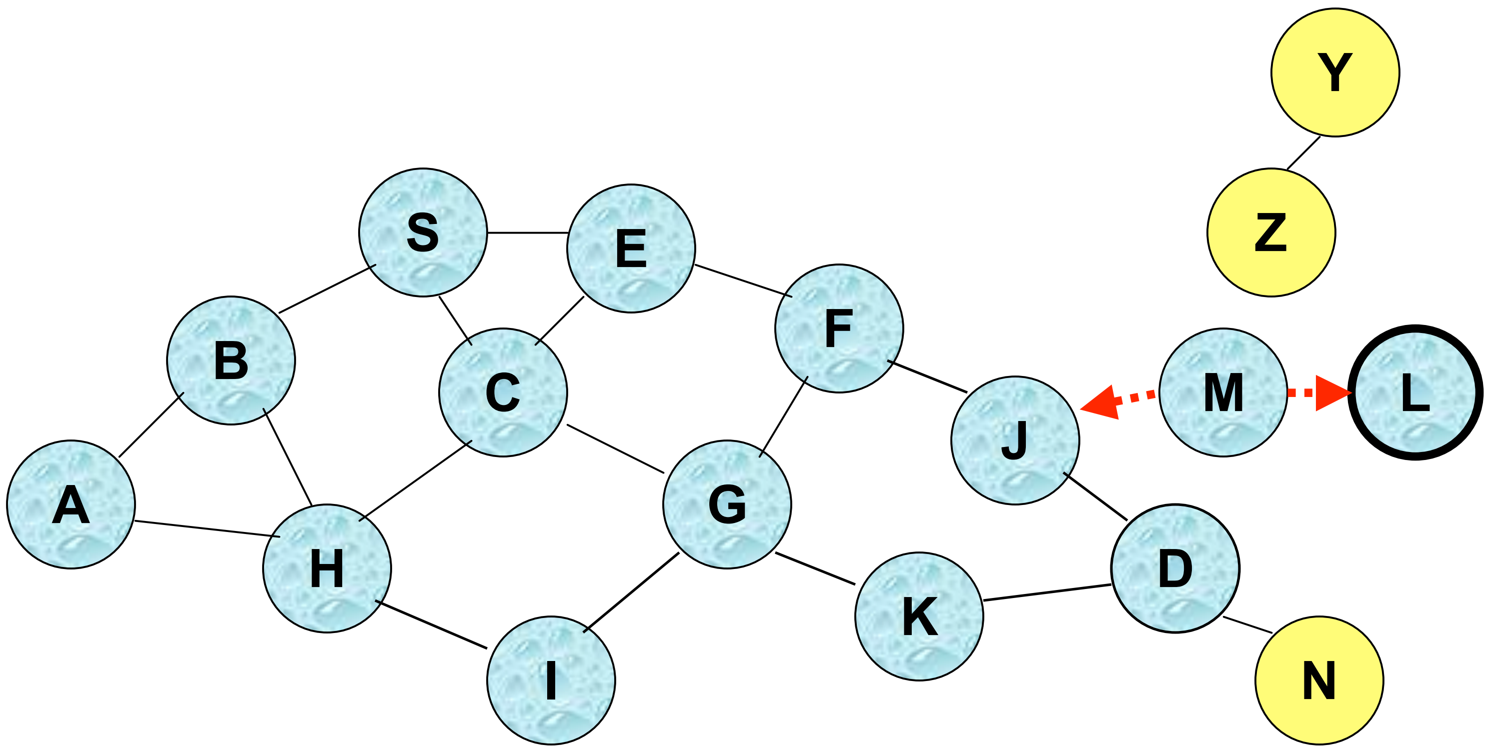
Flooding for Data Delivery



- 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

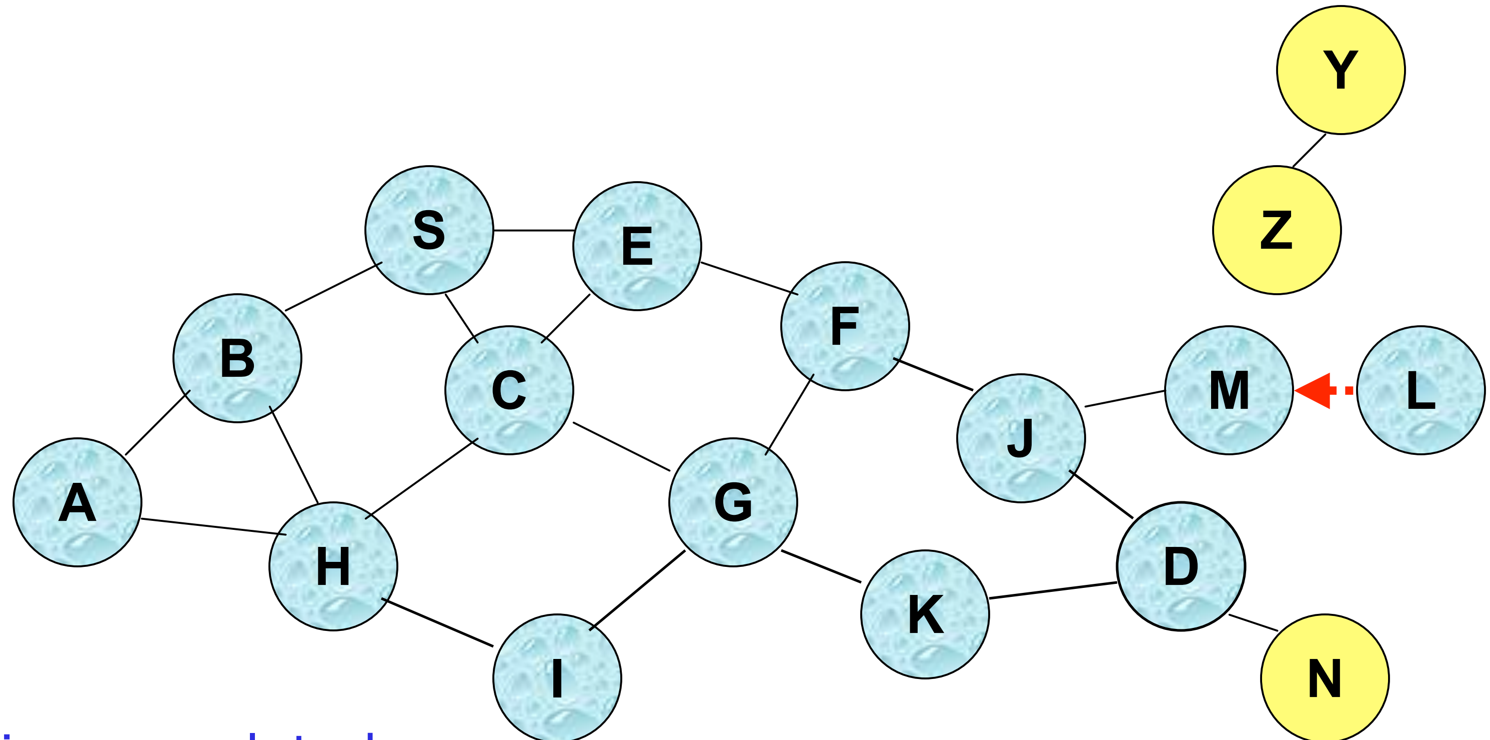
  => Packet P may not be delivered to node D at all

Flooding for Data Delivery



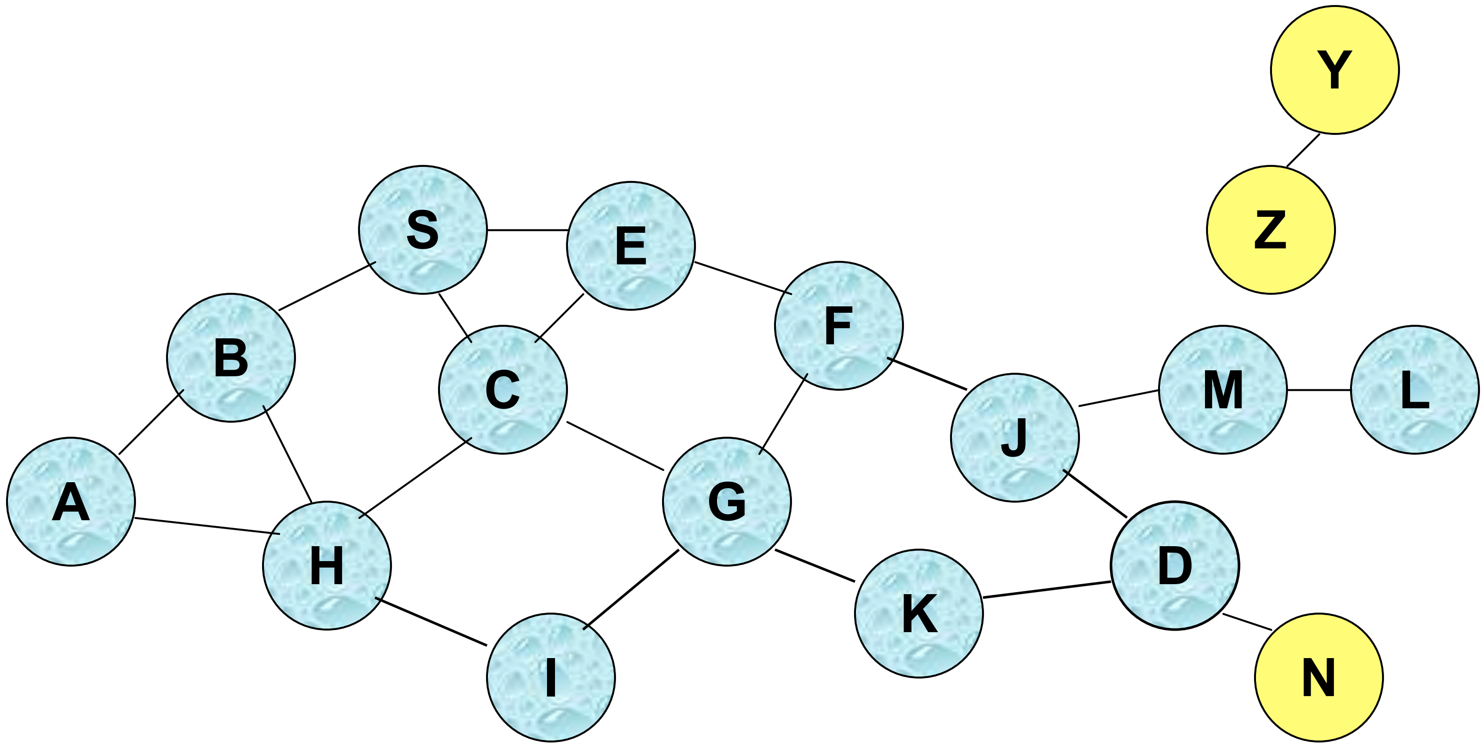
- Node D does not forward packet P, because node D is the intended destination of packet P

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: 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
- Potentially higher reliability of data delivery
    - Because packets may be delivered to the destination on multiple paths

Flooding: Disadvantages

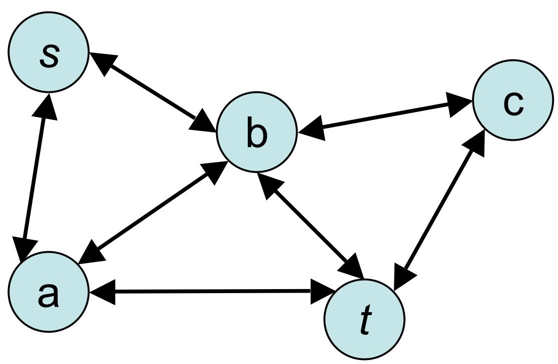
- Potentially, very high overhead
  - Data packets may be delivered to too many nodes who do not need to receive them
- Potentially lower reliability of data delivery
  - Flooding uses broadcasting -- hard to implement reliable broadcast delivery without significantly increasing overhead
    - Broadcasting 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

Proactive Routing: Link-State Routing Protocols

- Link-state routing protocols are a preferred iBGP method (within an autonomous system – think: service provider) in the Internet
- Idea: periodic notification of all nodes about the complete graph



Link-State Routing Protocols

- Routers then forward a message along (for example) the shortest path in the graph

+ message follows shortest path

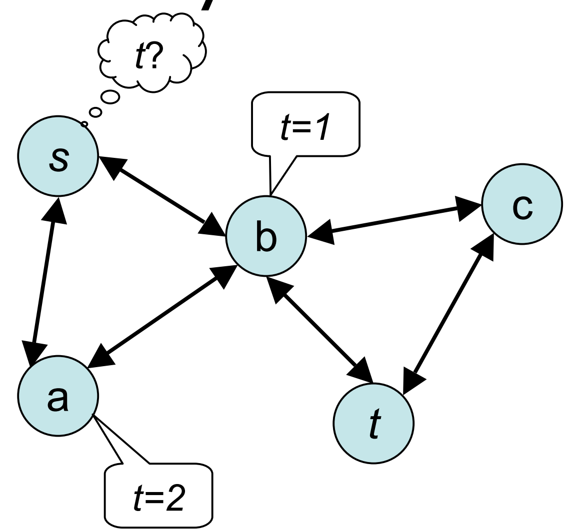
– every node needs to store whole graph, even links that are not on any path

– every node needs to send and receive messages that describe the whole graph regularly

Proactive Routing: Distance Vector Routing Protocols

- Often used in wired networks
- Idea: each node stores a routing table that has an entry to each destination (destination, distance, neighbor); each node maintains distance to every other node

Dest	Dir	Dst
a	*a*	1
b	*b*	1
c	*b*	2
t	*b*	2



Distance Vector Protocols

- If a router notices a change in its neighborhood or receives an update message from a neighbor, it updates its routing table accordingly and sends an update to all its neighbors

+ message follows shortest path
+ only send updates when topology changes
– most topology changes are irrelevant for a given source/destination pair
– Single edge/node failure may require most nodes to change most of their entries
– every node needs to store a big table
– temporary loops

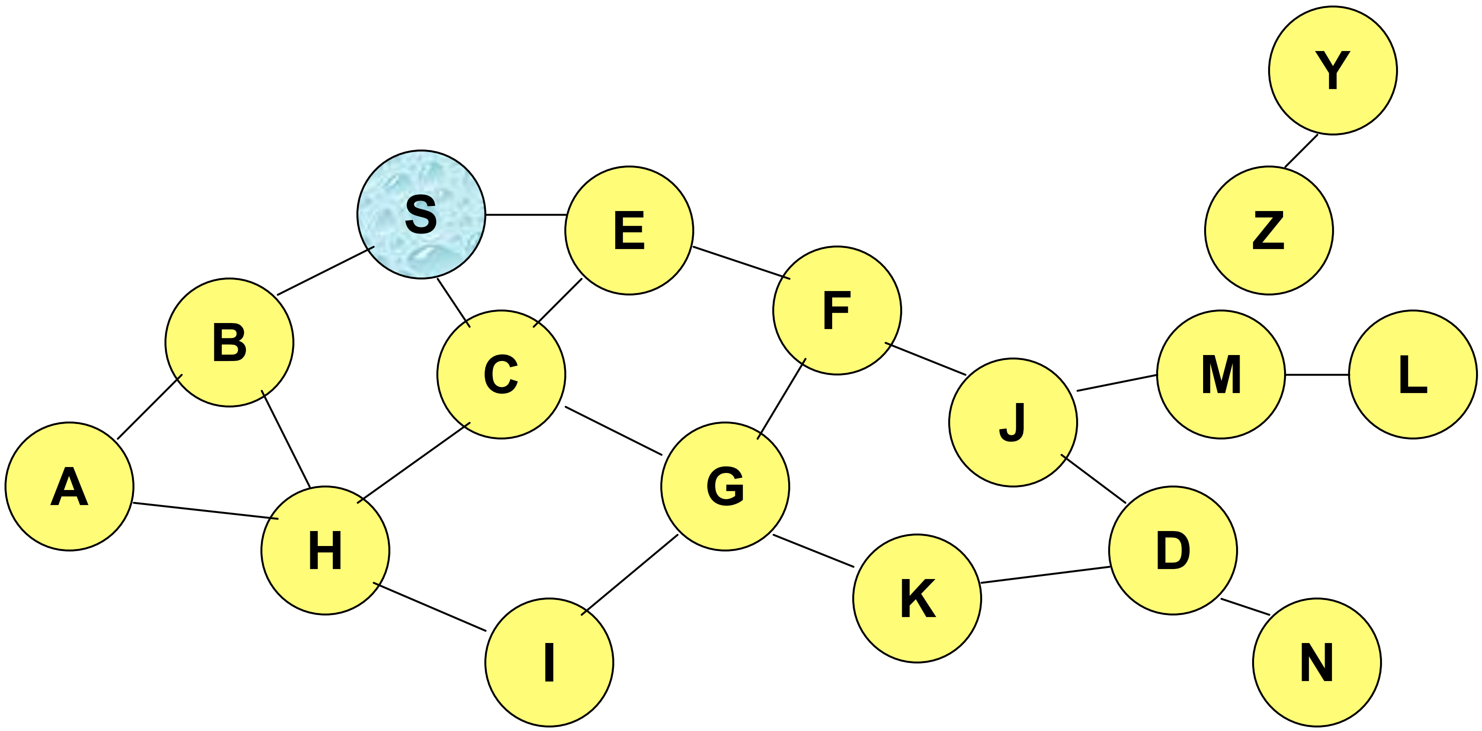
Destination-Sequenced Distance Vector

- [Perkins-Bhagwat 1996]
- Each entry in routing table (distance vector entry) has a sequence number
- Each mobile periodically advertizes its routing table entries
- Each node only needs to consider the entries with highest sequence number it has seen thus far
- Advantage: Quicker response time at time of routing
- Disadvantage: Too much control traffic when many changes in the network

Dynamic Source Routing (DSR) [Johnson96]

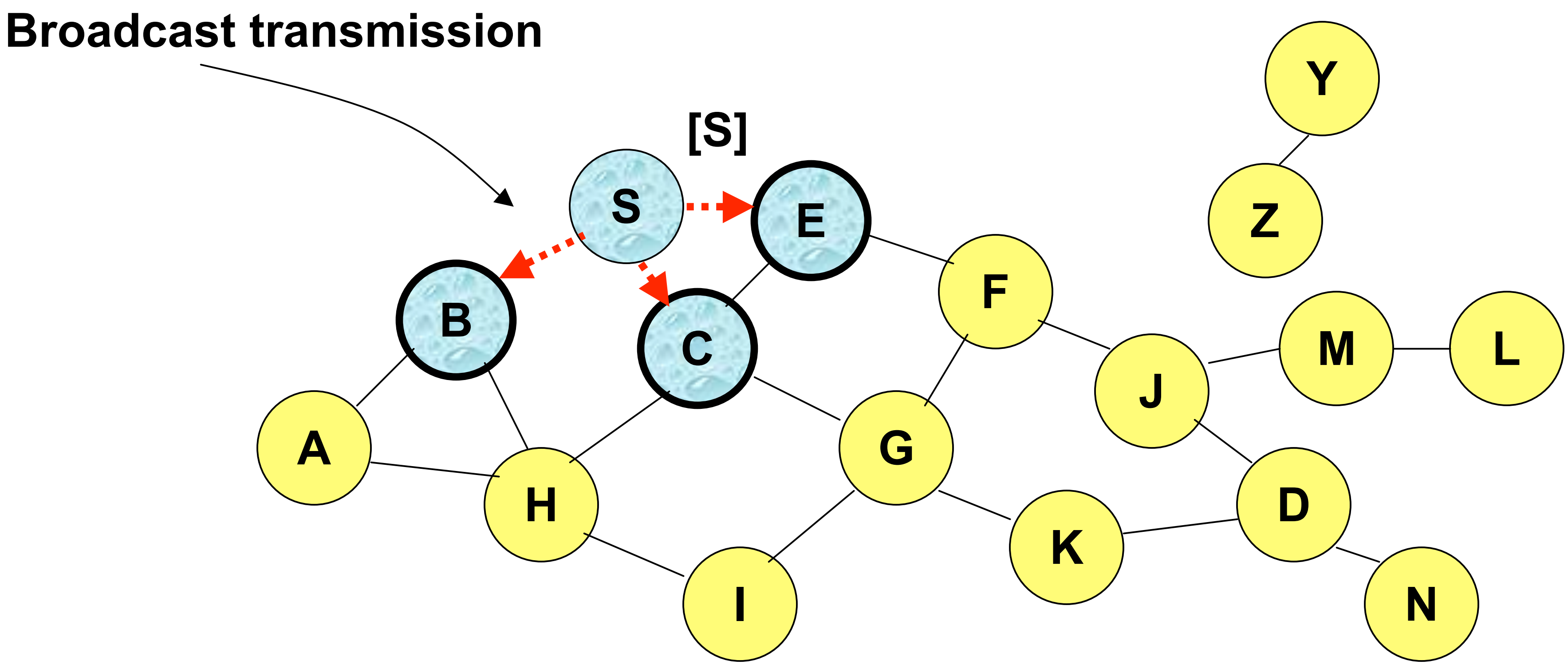
- When node S wants to send a packet to node D, but does not know a route to D, node S initiates a route discovery
- Source node S floods Route Request (RREQ)
- Each node appends own identifier when forwarding RREQ

Route Discovery in DSR



Represents a node that has received RREQ for D from S

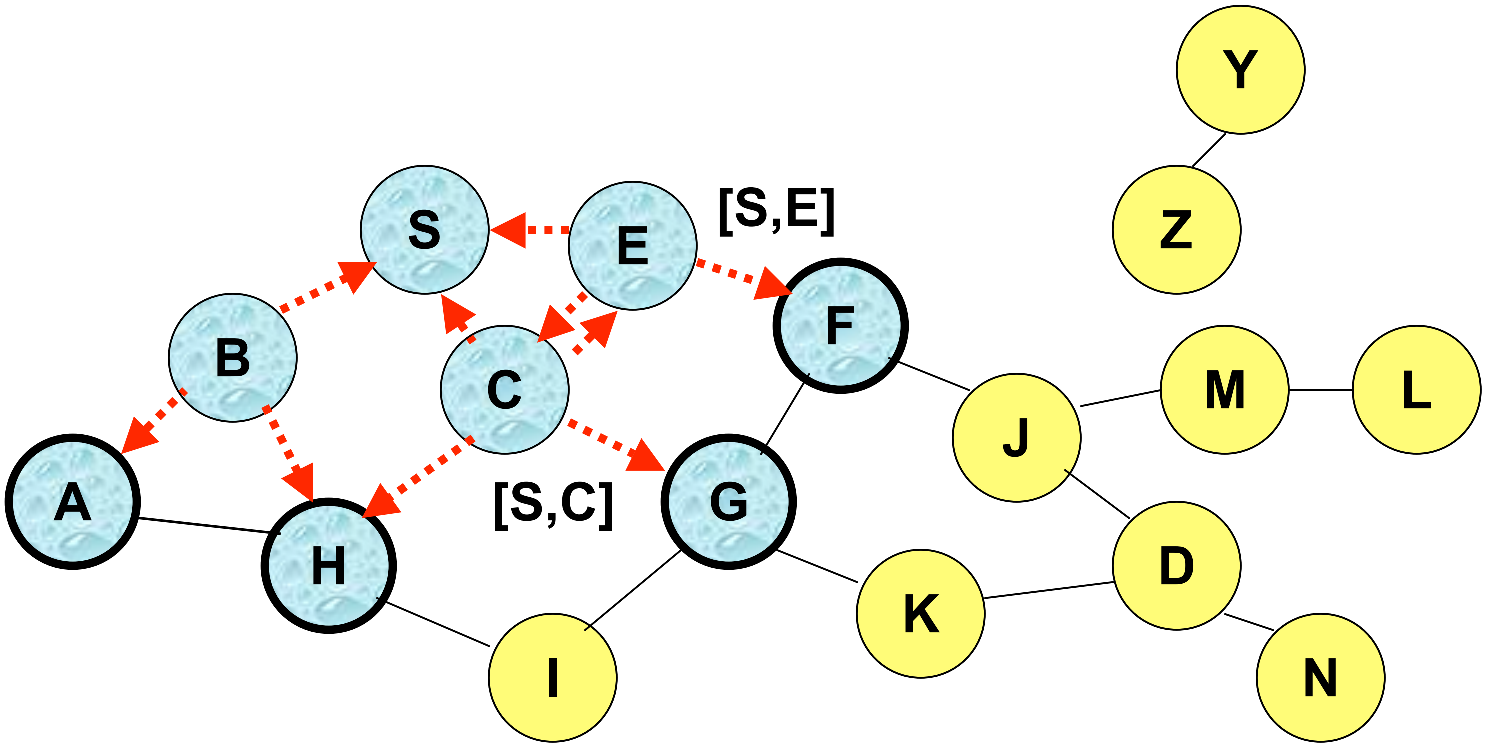
Route Discovery in DSR



Represents transmission of RREQ

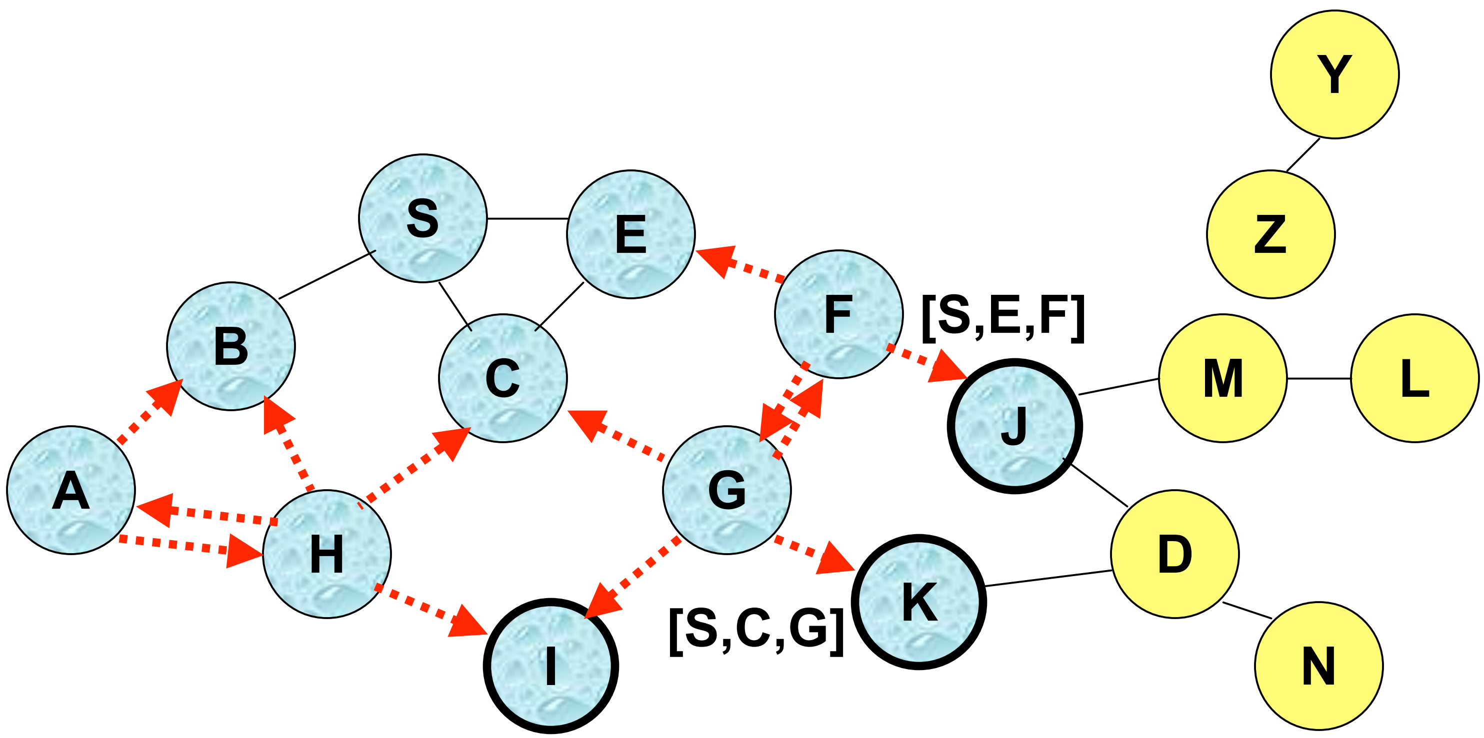
[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR



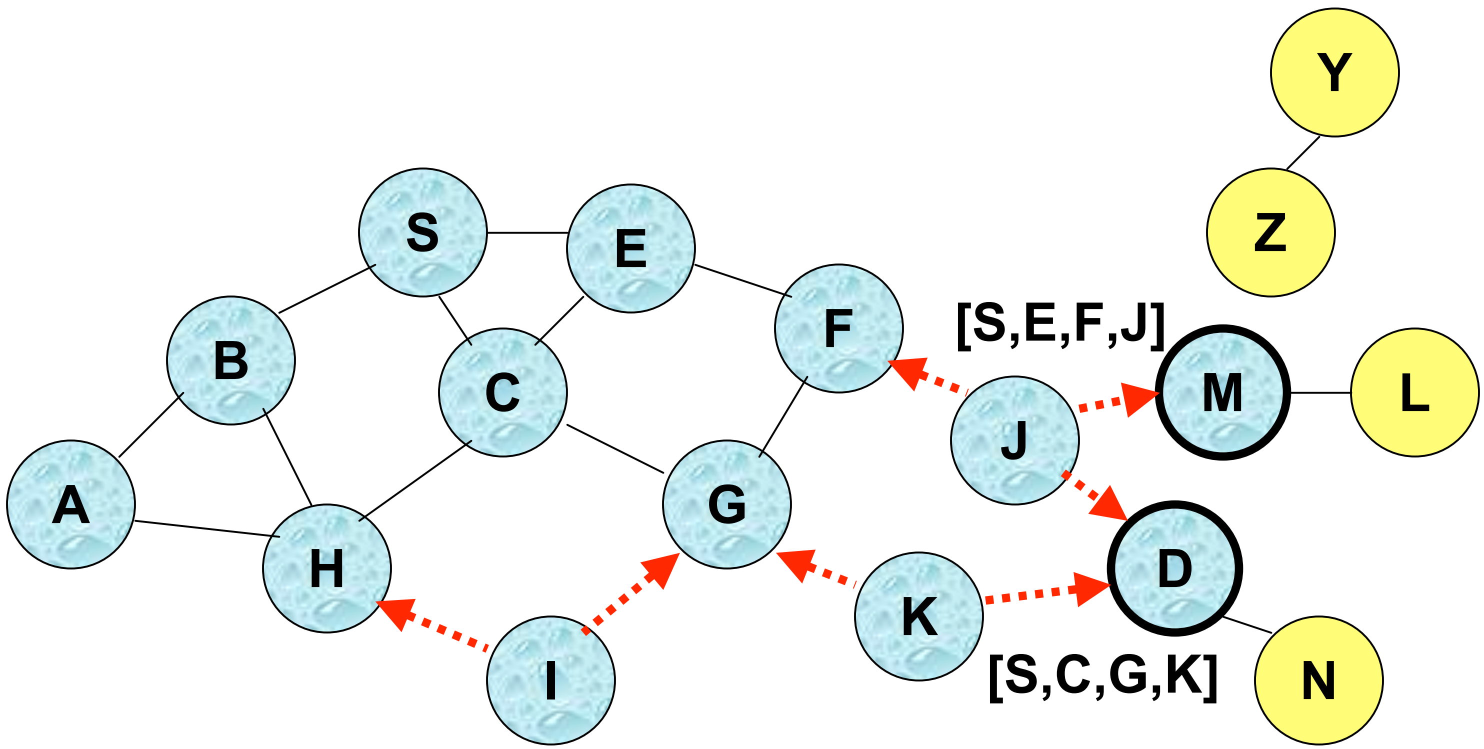
- **Node H receives packet RREQ from two neighbors: potential for collision**

Route Discovery in DSR



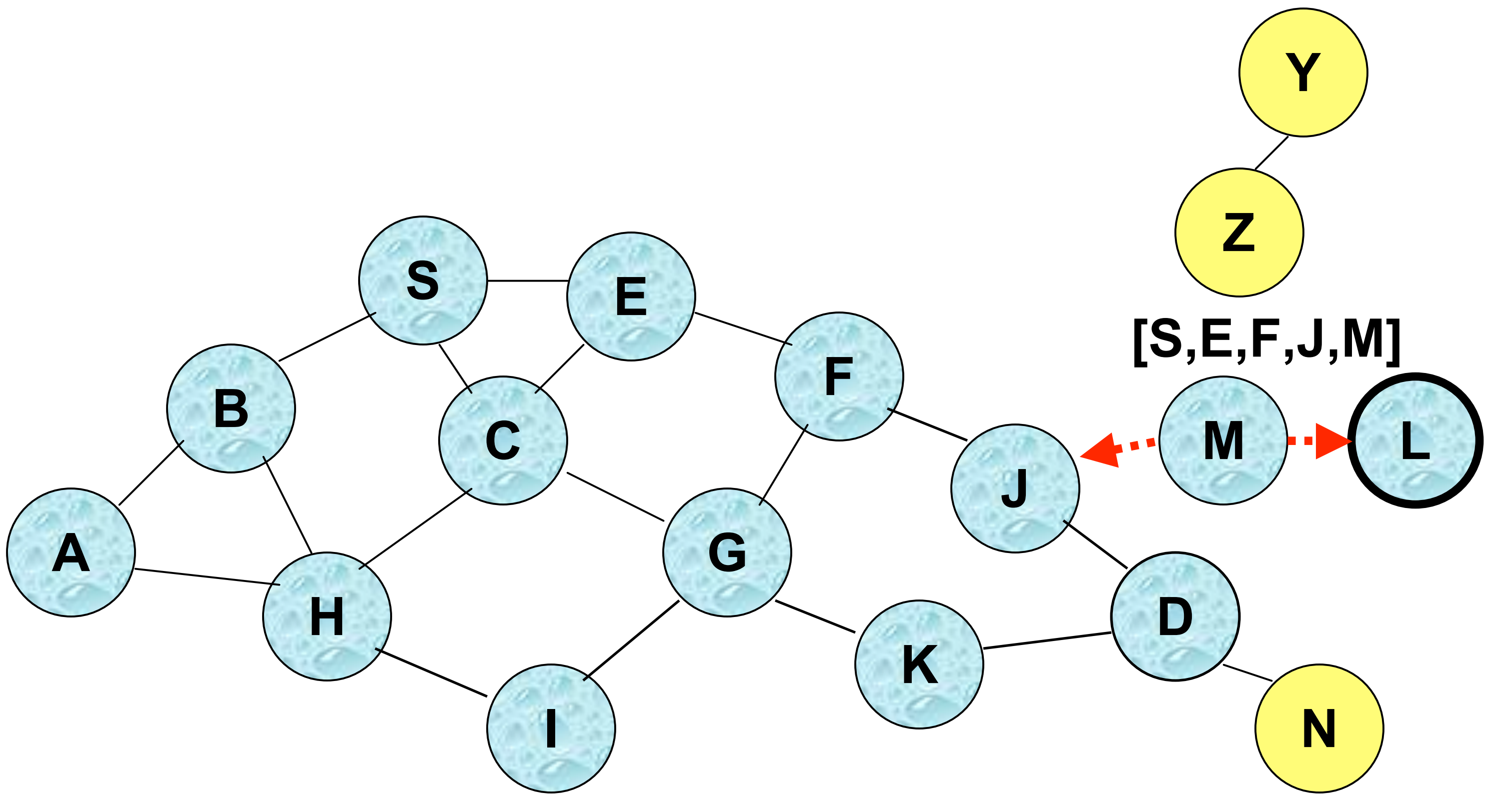
- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

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**

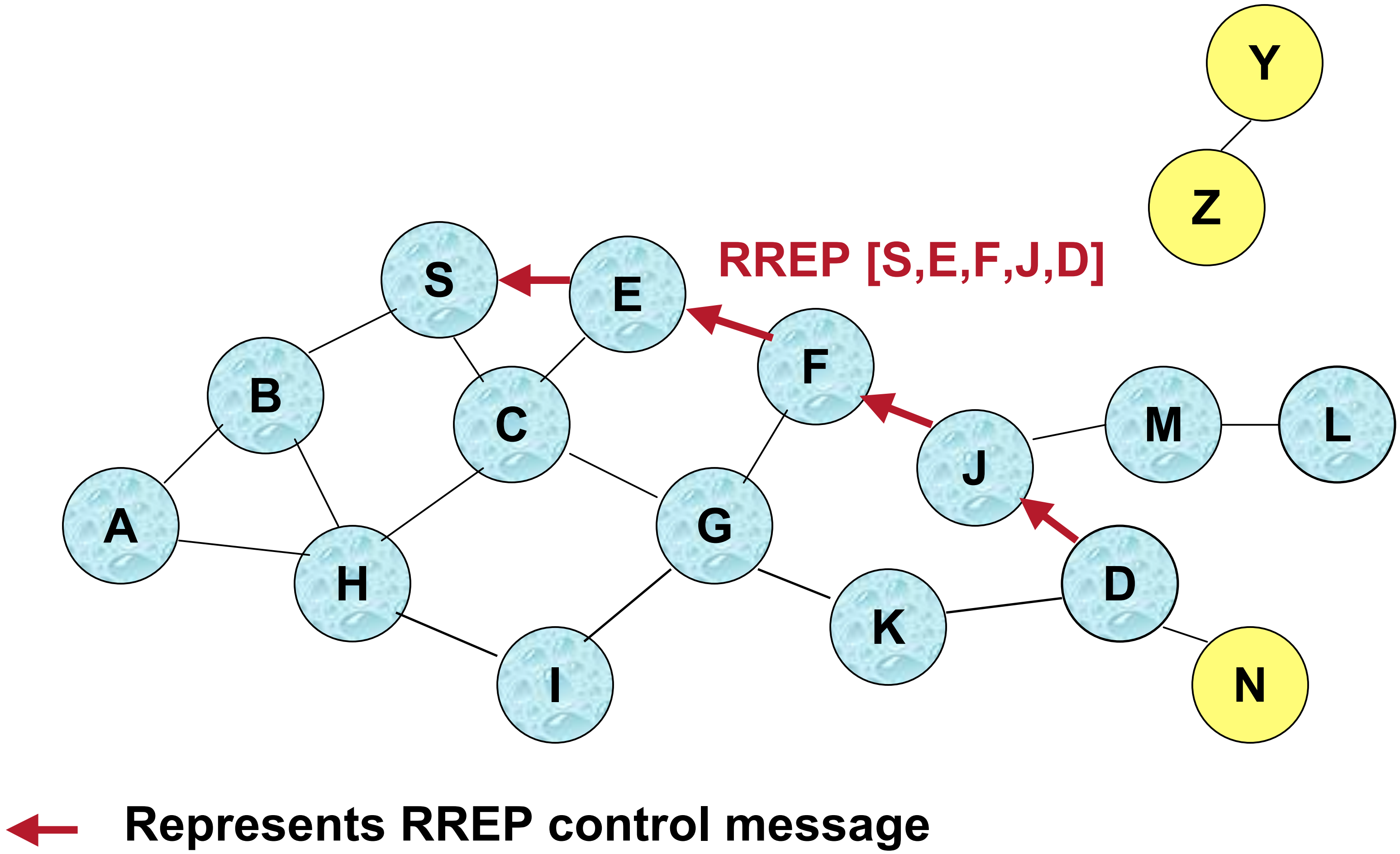
Route Discovery in DSR

- 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



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

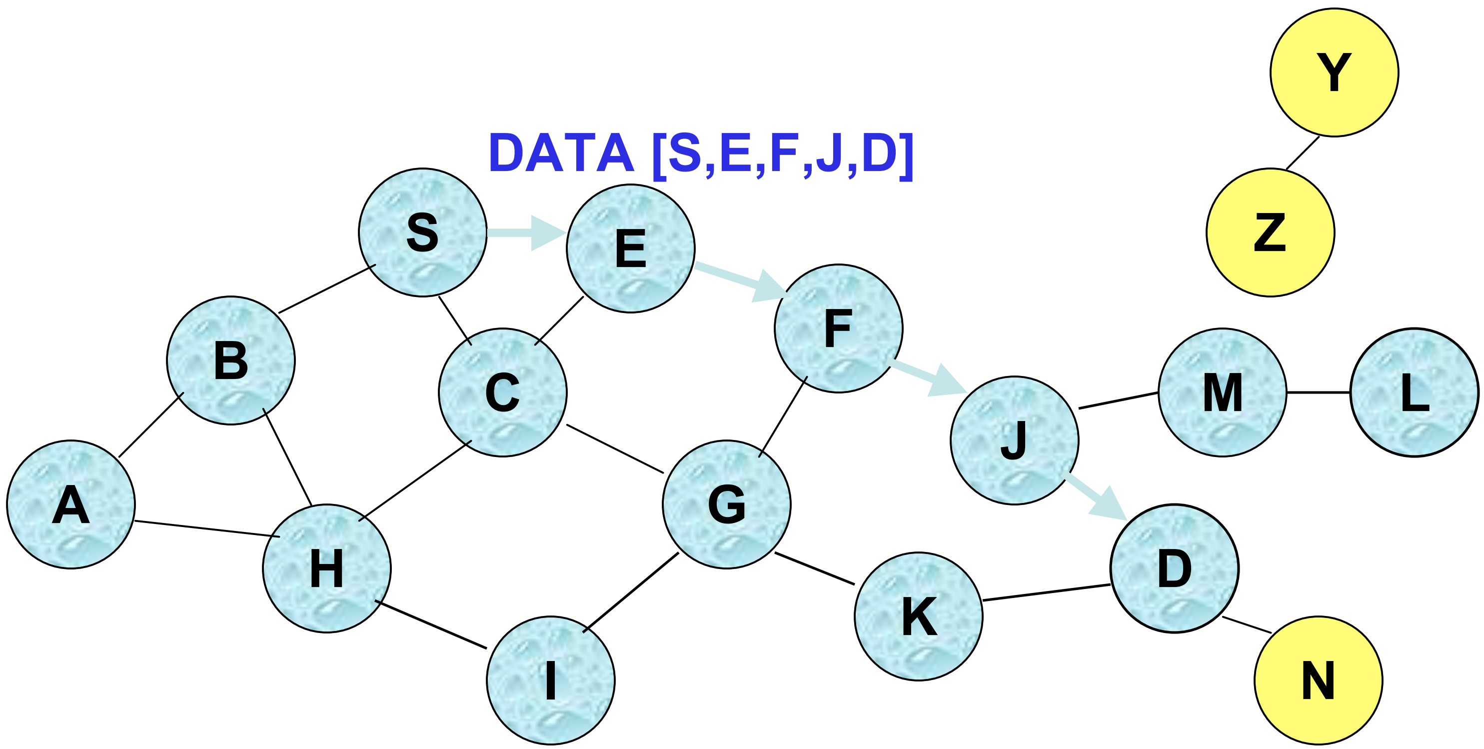
- If unidirectional (asymmetric) links are allowed, then RREP may need a route discovery for S from node D
  - Unless node D already knows a route to node S
  - If a route discovery is initiated by D for a route to S, then the Route Reply is piggybacked on the Route Request from D.

- If IEEE 802.11 MAC is used to send data, then links have to be bi-directional (since Ack is used)

Dynamic Source Routing (DSR)

- 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

Data Delivery in DSR



Packet header size grows with route length

When to Perform a Route Discovery

- When node S wants to send data to node D, but does not know a valid route node D

DSR Optimization: Route Caching

- Each node caches a new route it learns by *any means*
- 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, 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
- 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

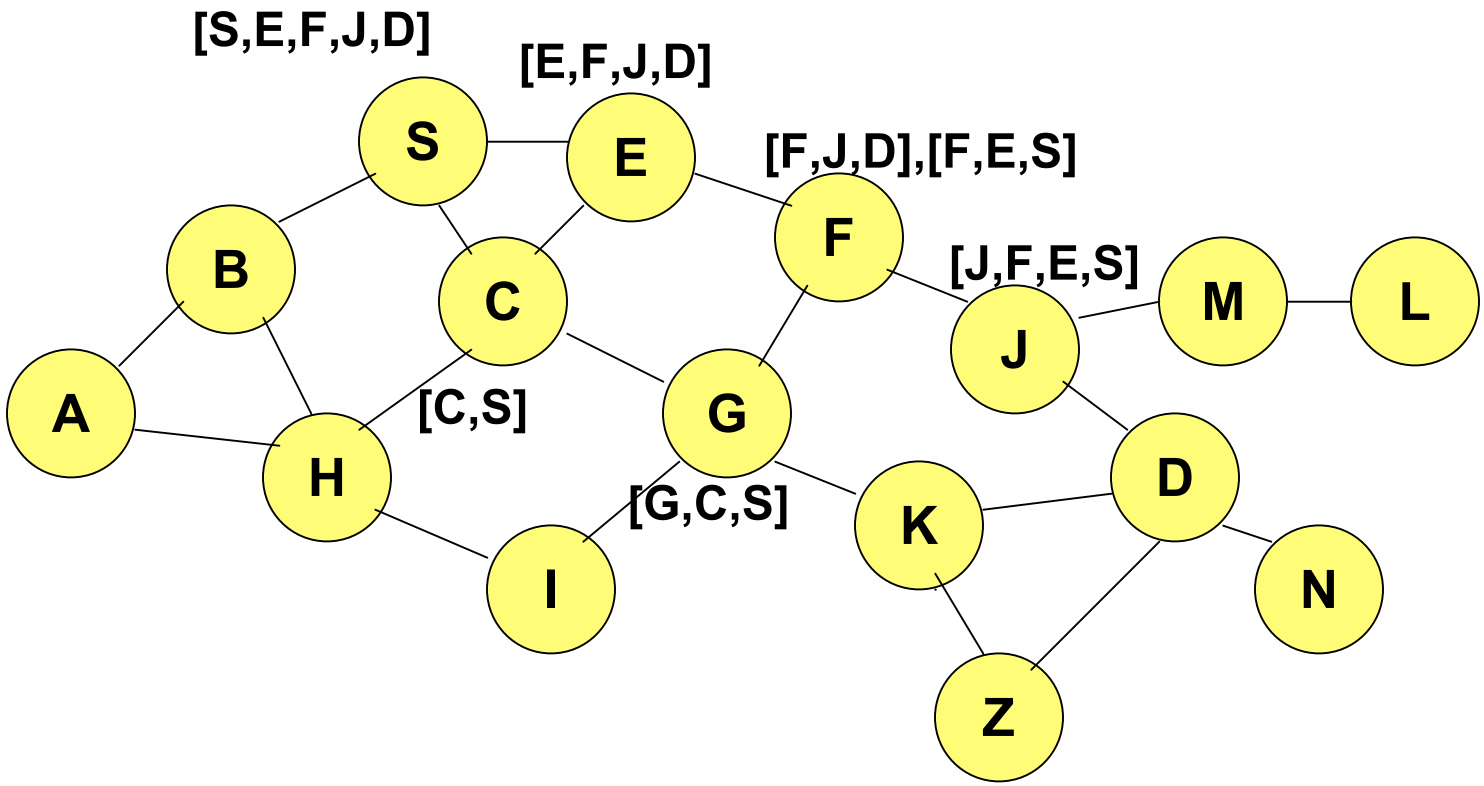
Use of Route Caching

- 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 if node X knows a route to node D

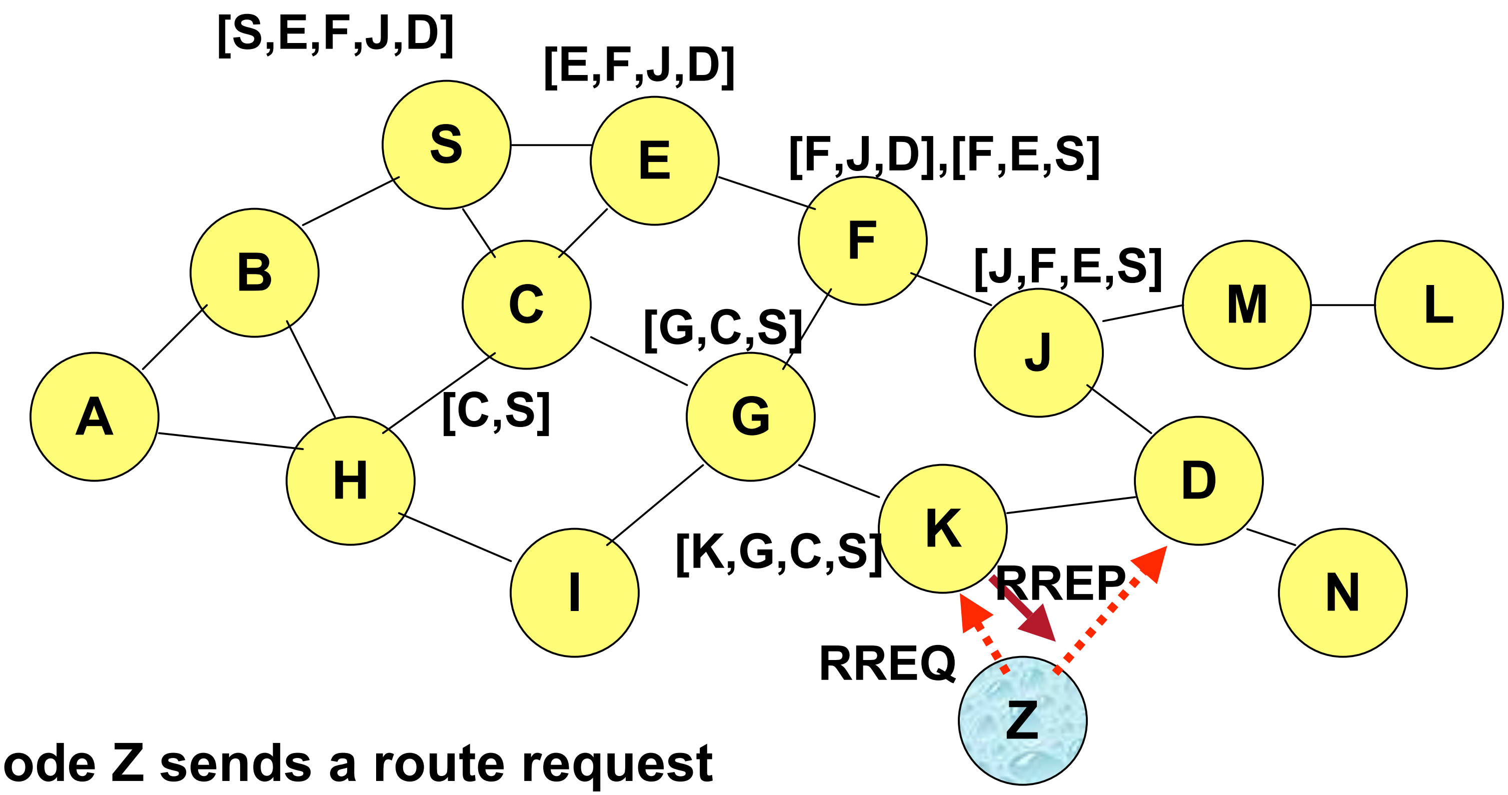
- Use of route cache
  - can speed up route discovery
  - can reduce propagation of route requests

Use of Route Caching



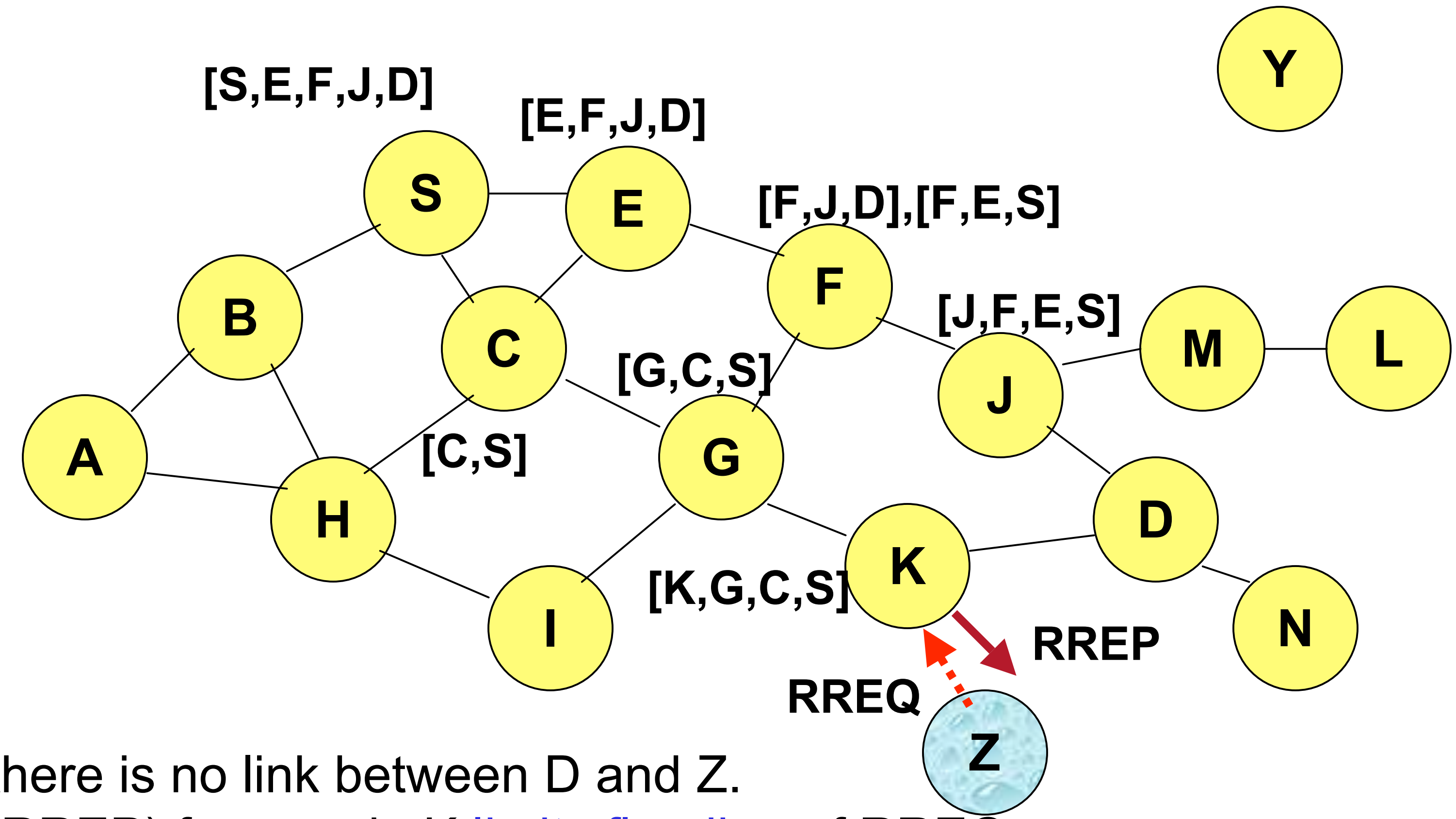
[P,Q,R] Represents cached route at a node
(DSR maintains the cached routes in a tree format)

Use of Route Caching: Can Speed up Route Discovery



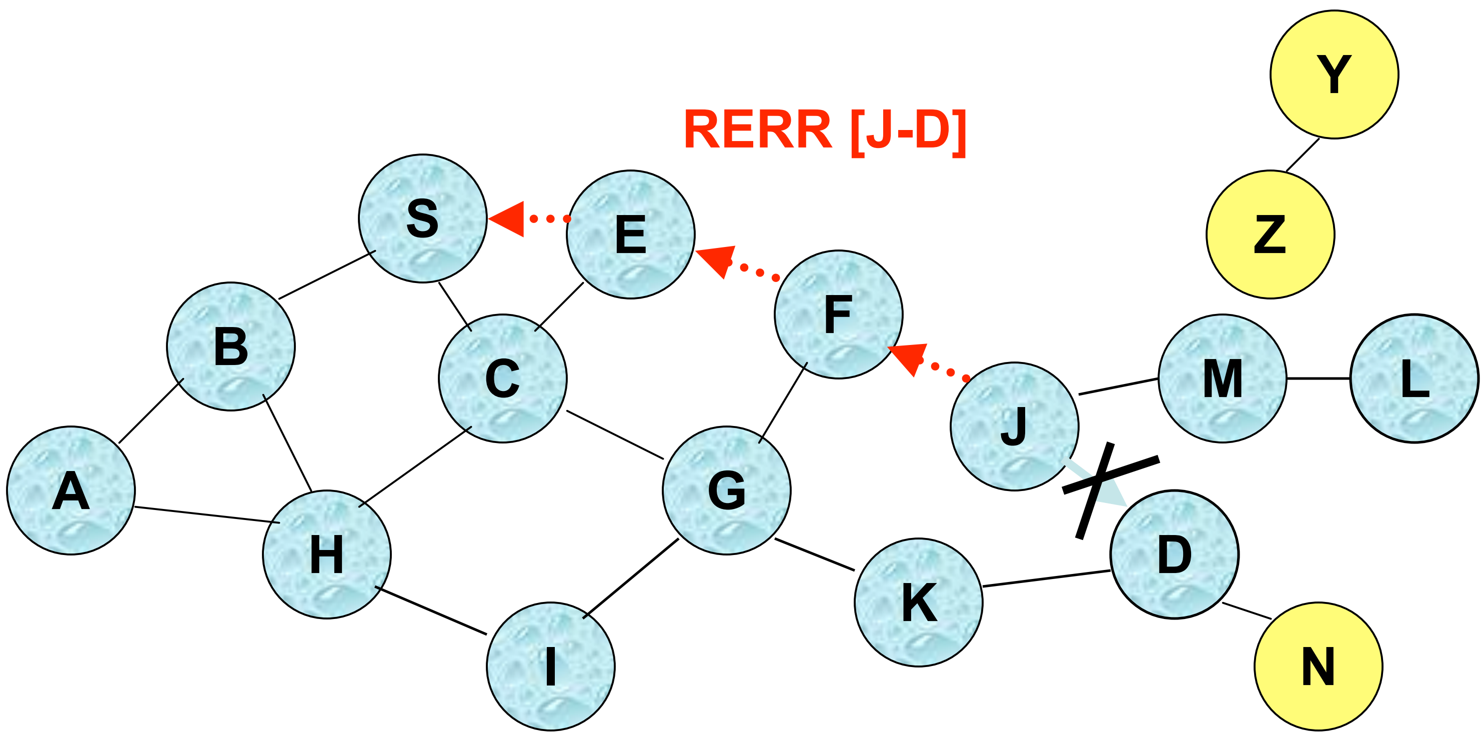
When node Z sends a route request for node C, node K sends back a route reply [Z,K,G,C] to node Z using a locally cached route

Use of Route Caching: Can Reduce Propagation of Route Requests



Assume that there is no link between D and Z.
Route Reply (RREP) from node K limits flooding of RREQ.
In general, the reduction may be less dramatic.

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

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
- 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

- 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 link stability

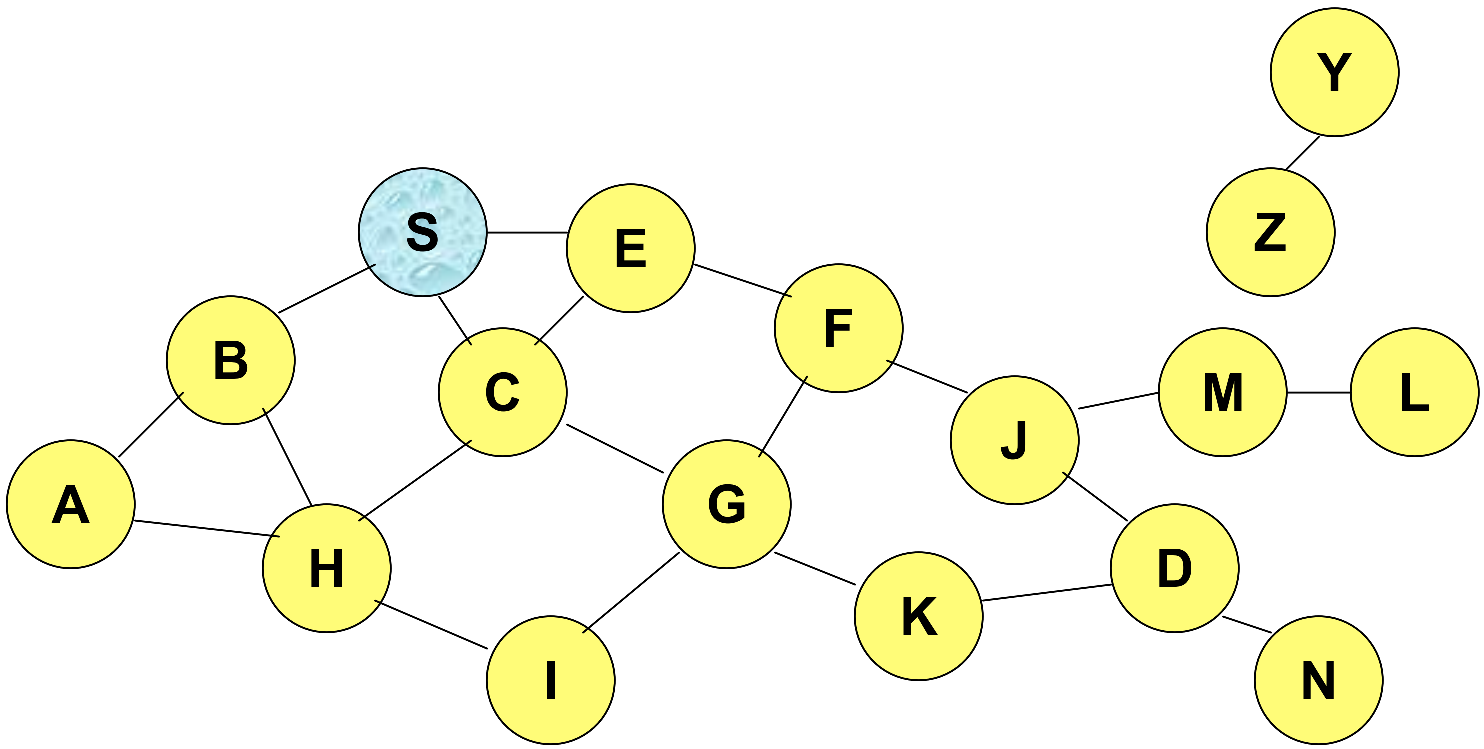
Ad Hoc On-Demand Distance Vector Routing (AODV) [Perkins99]

- 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 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

AODV

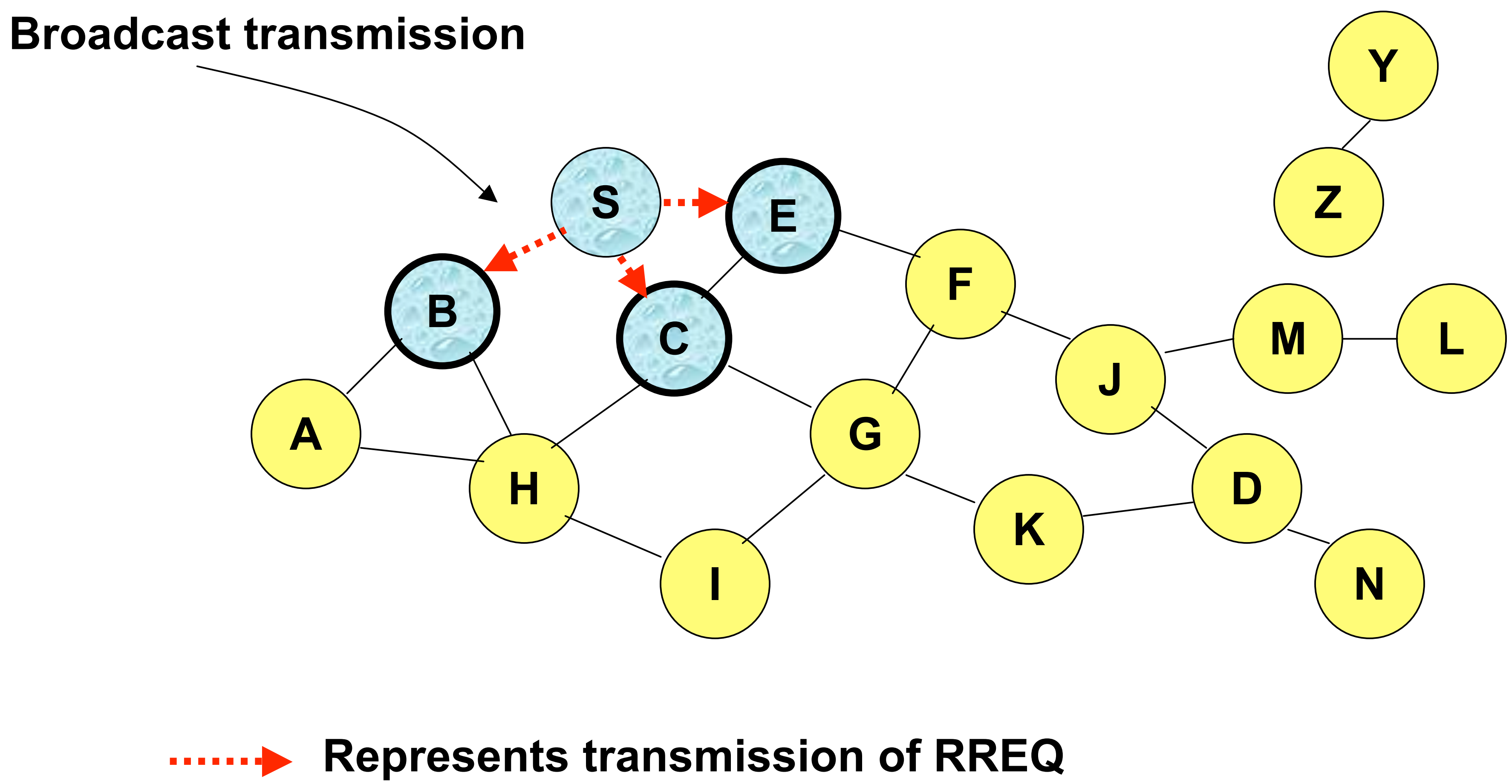
- Route Requests (RREQ) are forwarded in a manner similar to DSR
- When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source
  - AODV assumes symmetric (bi-directional) links
- When the intended destination receives a Route Request, it replies by sending a Route Reply
- Route Reply travels along the reverse path set-up when Route Request is forwarded

Route Requests in AODV

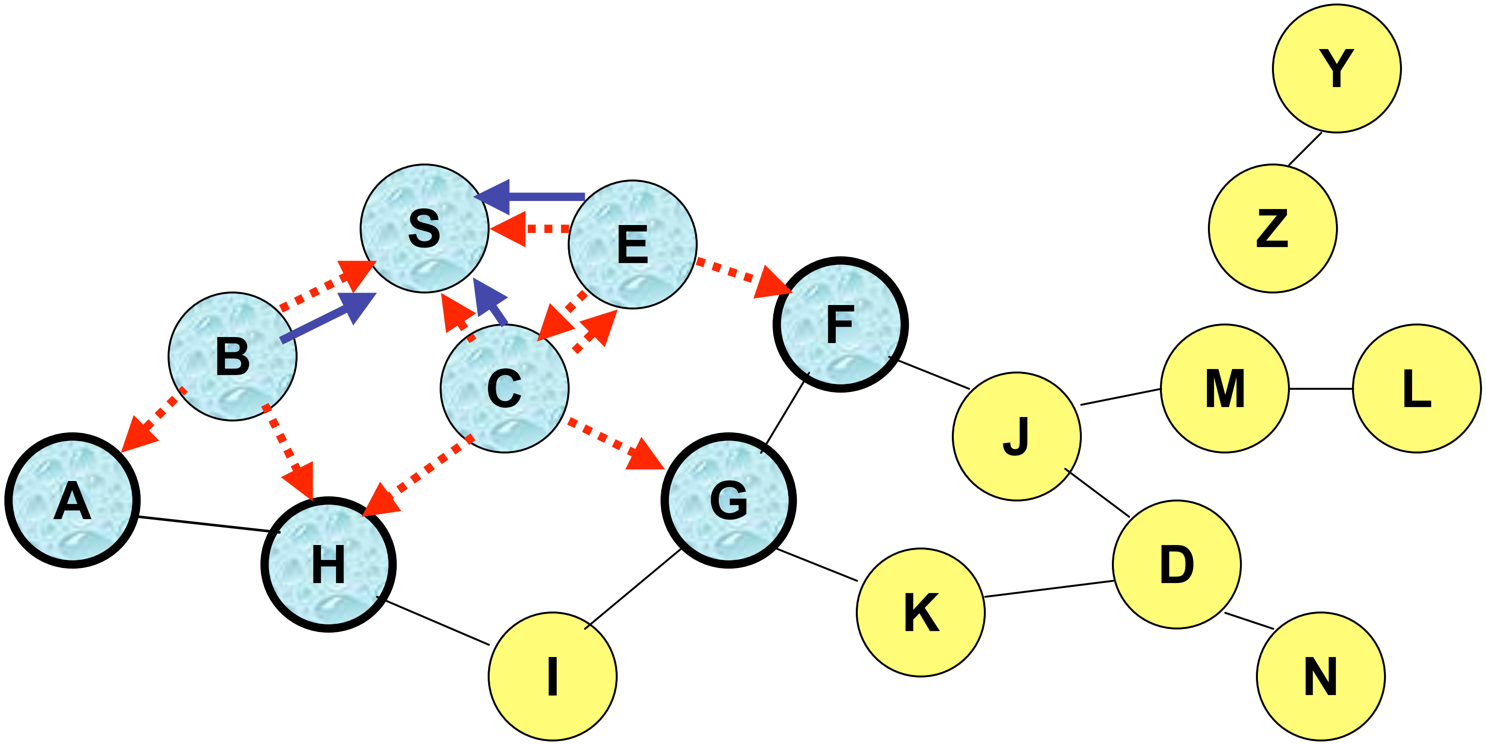


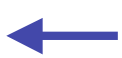
Represents a node that has received RREQ for D from S

Route Requests in AODV

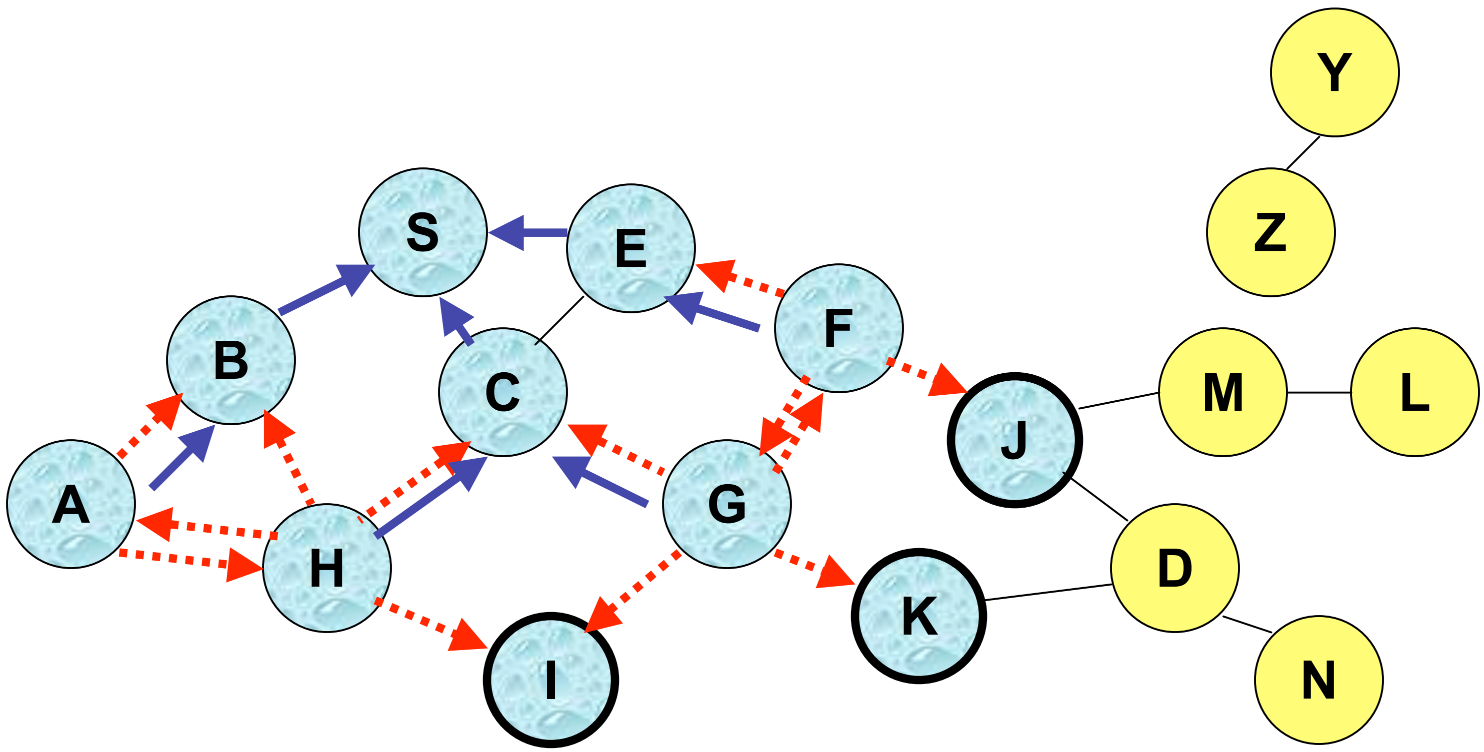


Route Requests in AODV



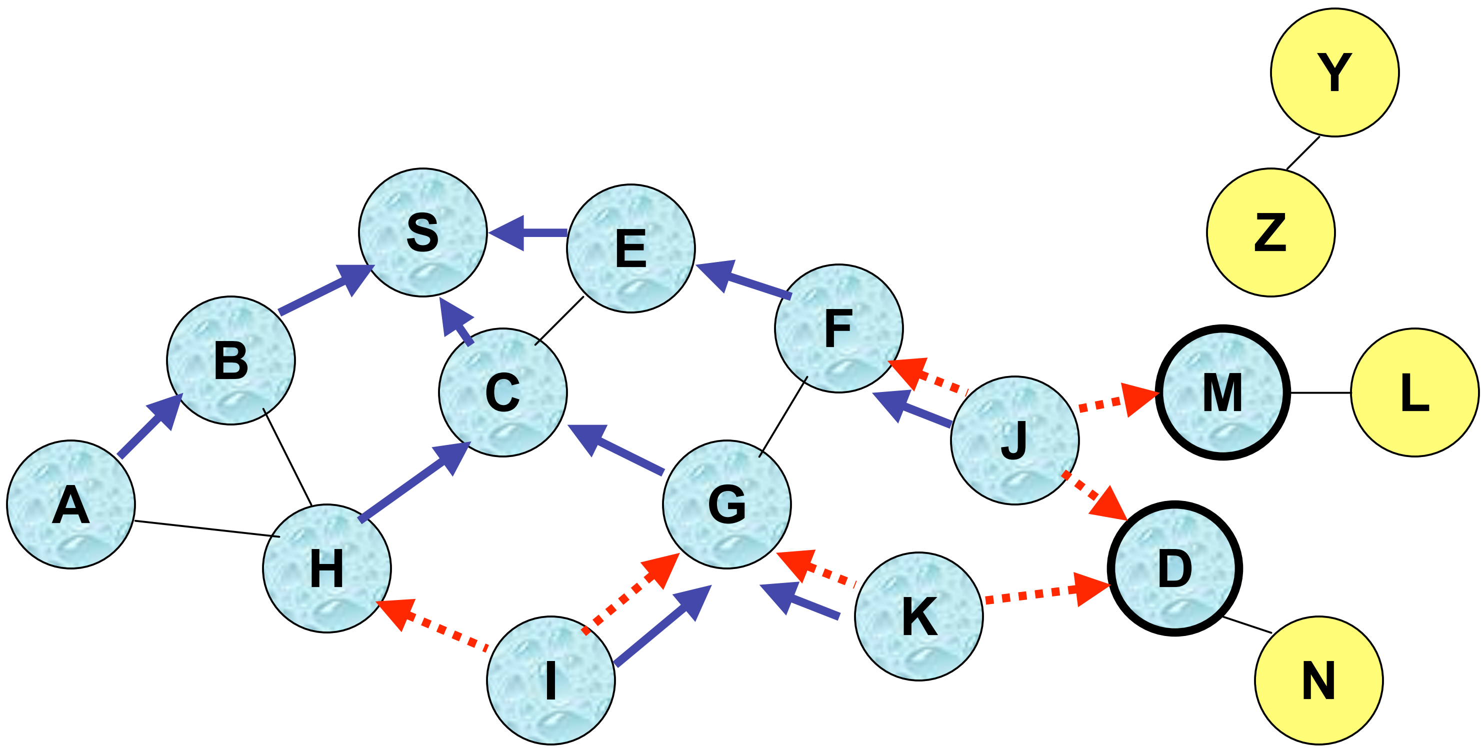
 Represents links on Reverse Path

Reverse Path Setup in AODV

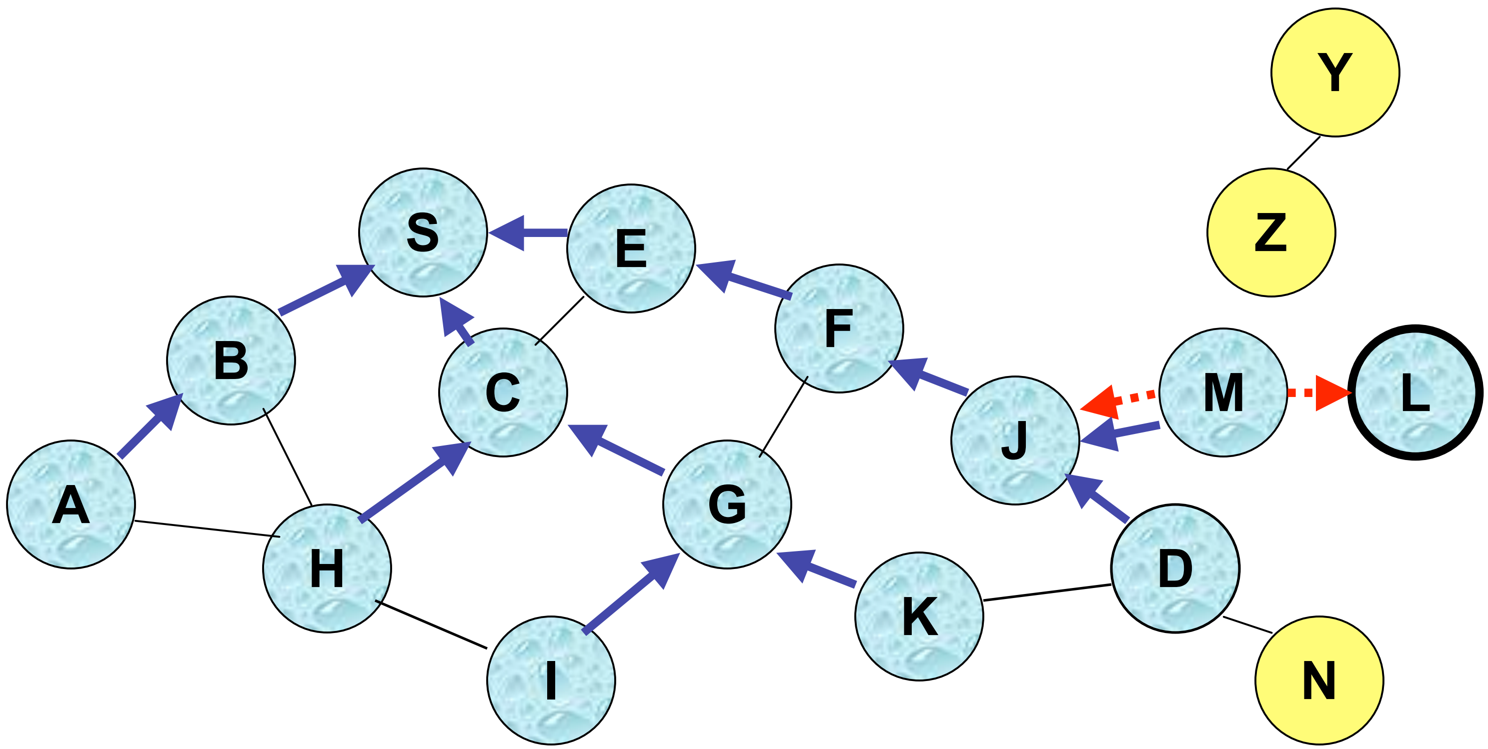


- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

Reverse Path Setup in AODV

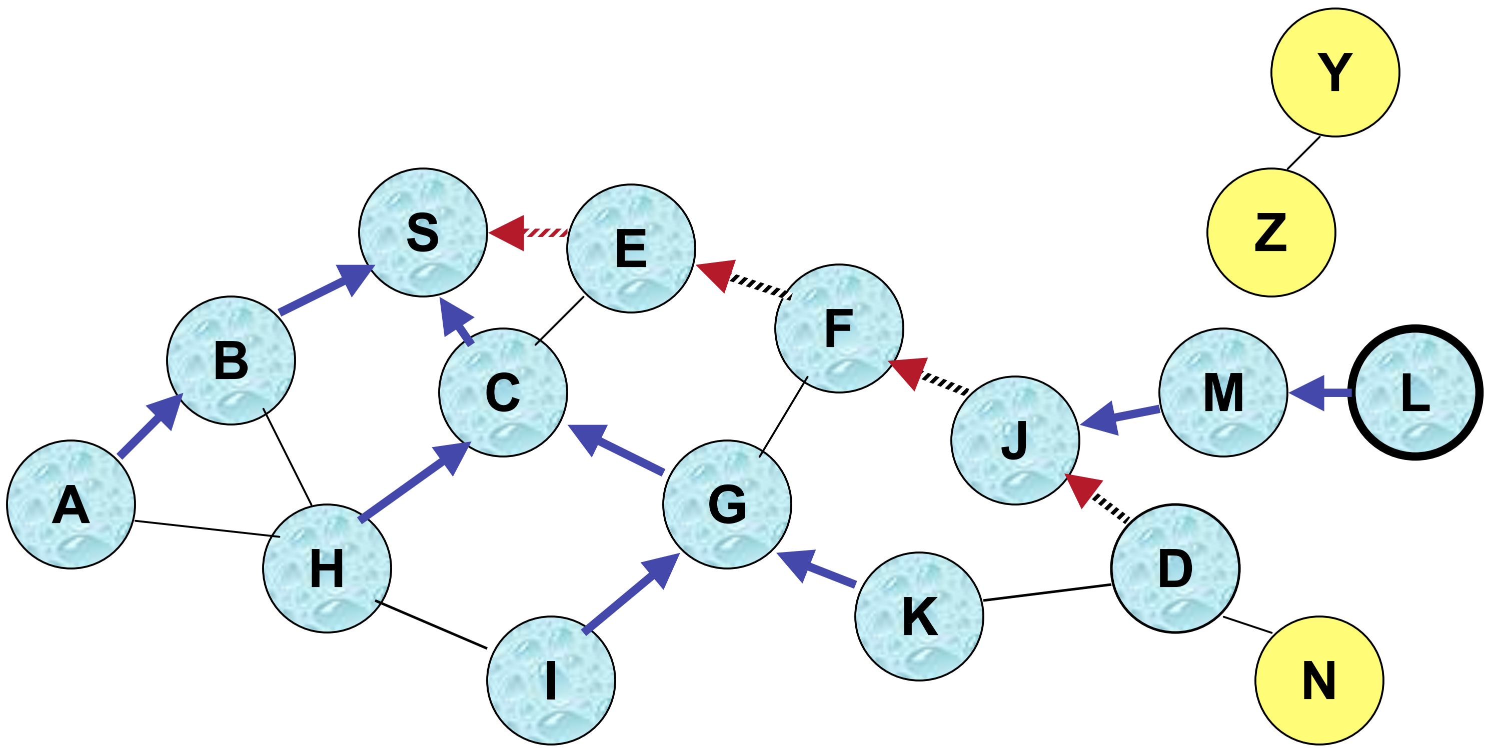


Reverse Path Setup in AODV



- 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

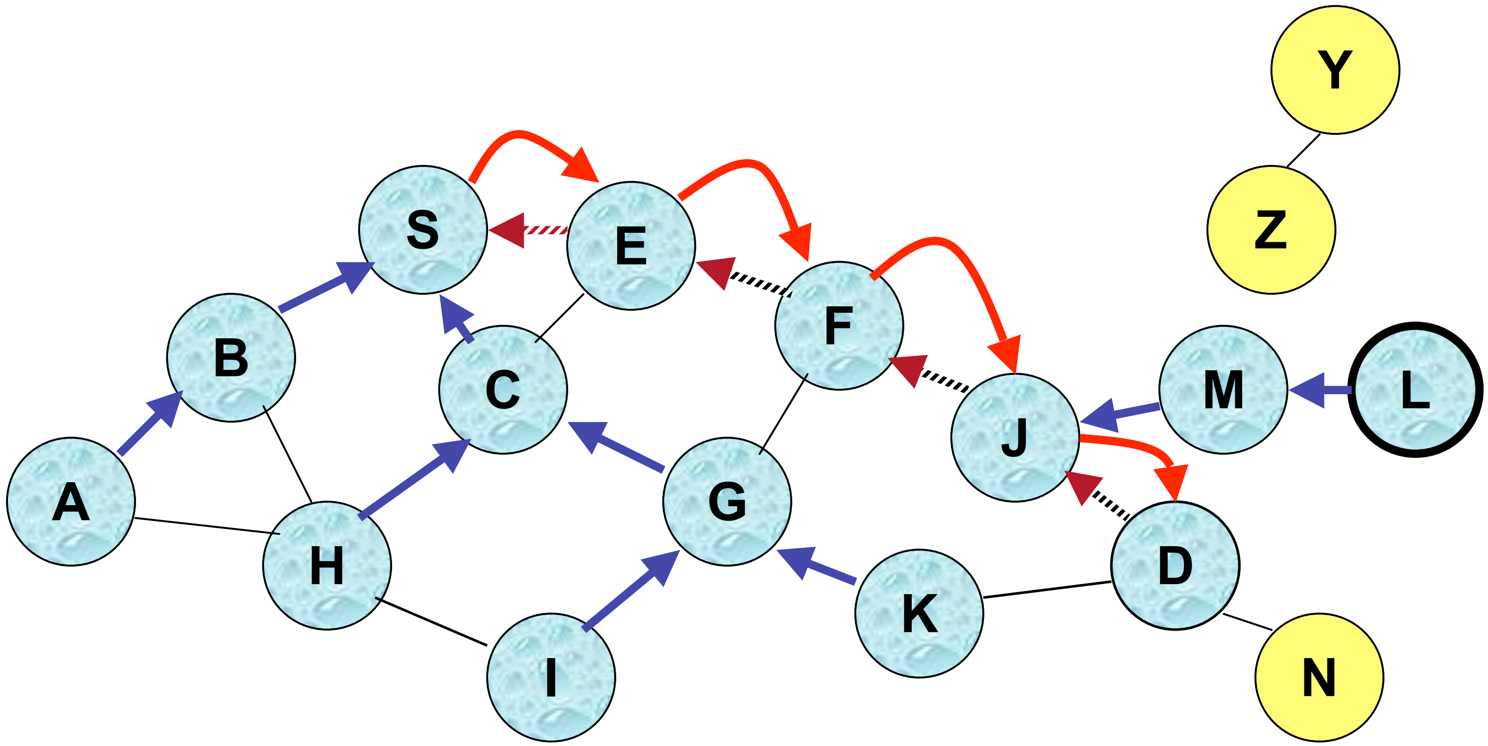
Route Reply in AODV

- An intermediate node (not the destination) may also send a Route Reply (RREP) provided that it knows a more recent path than the one previously known to sender S

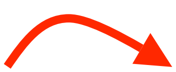
- To determine whether the path known to an intermediate node is more recent, *destination sequence numbers* are used

- The likelihood that an intermediate node will send a Route Reply when using AODV not as high as DSR
  - 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

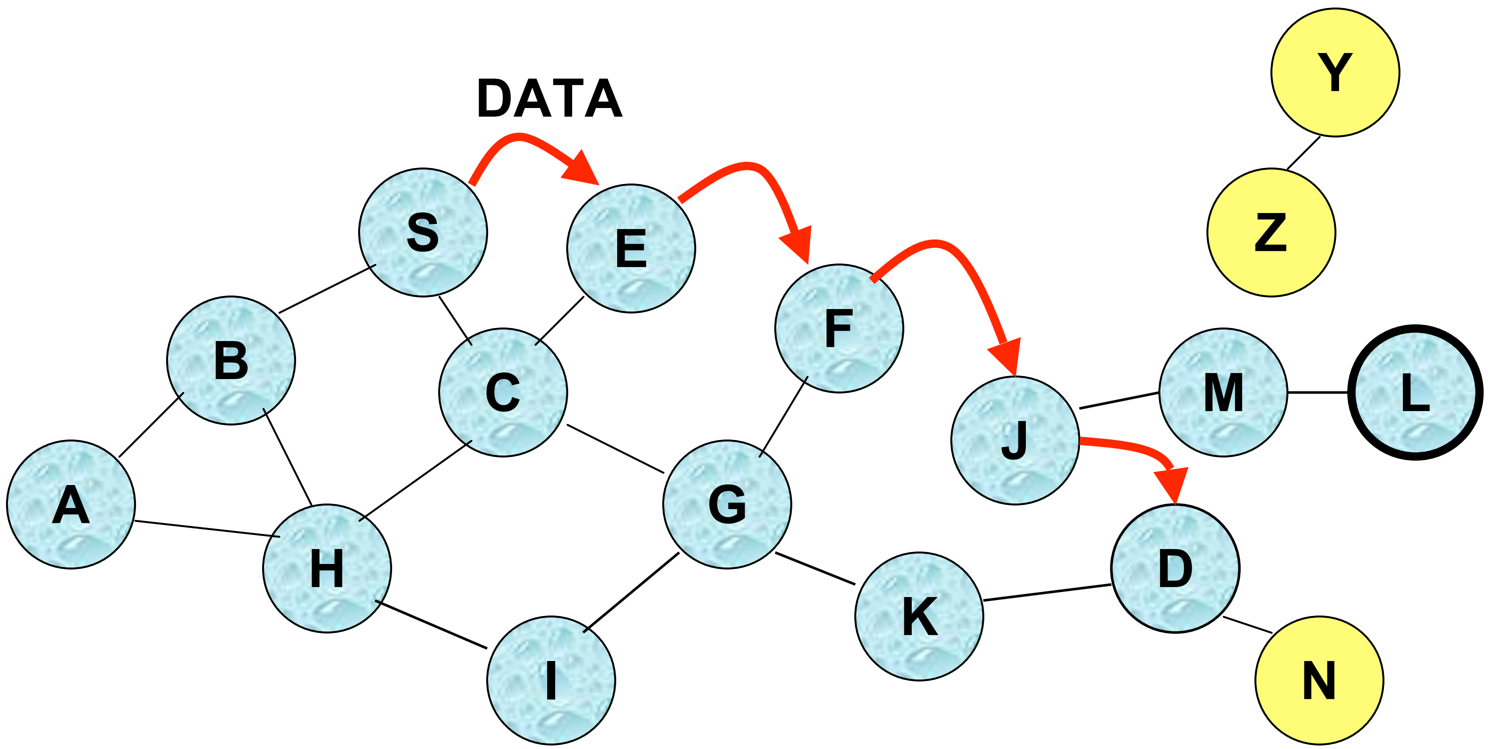
Forward Path Setup in AODV



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 a *active_route_timeout* interval
  - if no data being 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 messages, which also update destination sequence numbers

Route Error

- When node X is unable to forward packet P (from node S to node D) on link (X,Y), it generates a RERR message
- Node X increments the destination sequence number for D cached at node X
- The incremented sequence number *N* is included in the RERR
- When node S receives the RERR, it initiates a new route discovery for D using destination sequence number at least as large as *N*

Destination Sequence Number

- Continuing from the previous slide **...**

- When node D receives the route request with destination sequence number N, node D will set its sequence number to N, unless it is already larger than N

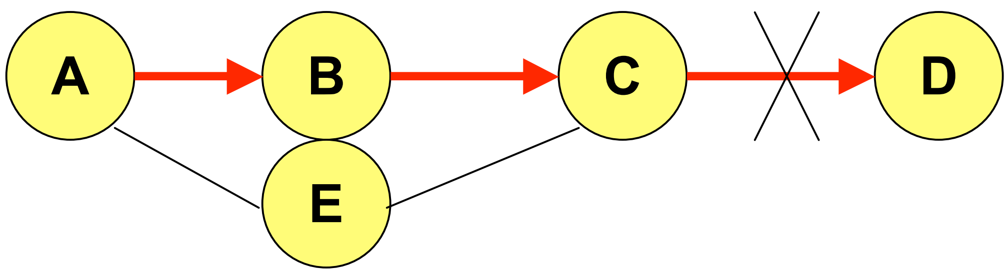
Link Failure Detection

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

Why Sequence Numbers in AODV

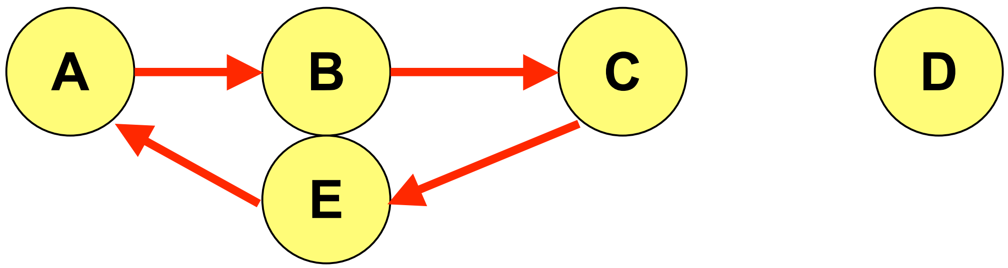
- To avoid using old/broken routes
  - To determine which route is newer

- 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 in AODV



- o Loop C-E-A-B-C

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

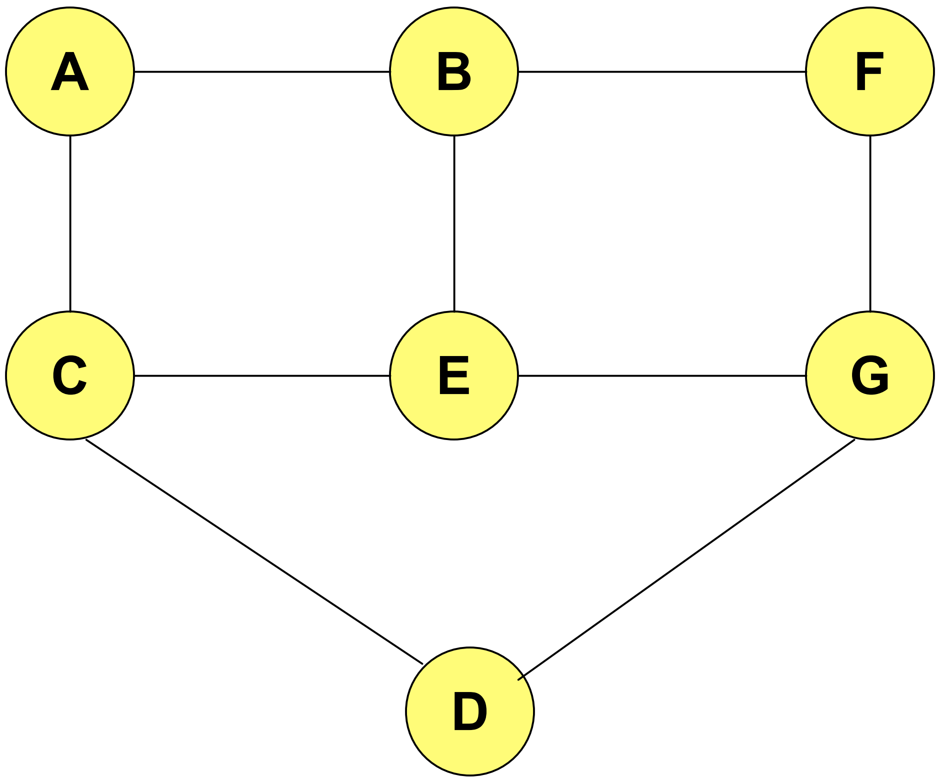
Summary: AODV

- 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

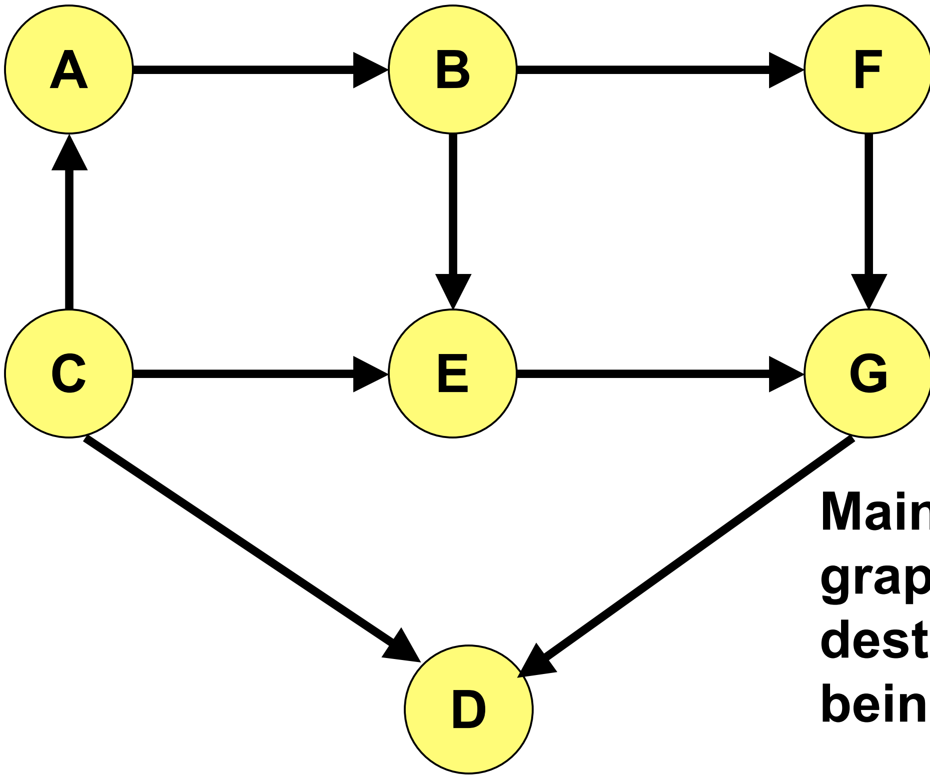
Other novel approaches to ad hoc network routing

- Link reversal
  - Aimed for highly dynamic networks
  - Goal: to identify some path, as opposed to the best path
- Clustering
  - For transmission management
  - For routing
- Geometric routing
  - Take advantage of the underlying physical space
  - Assume that node locations are known
  - Route to a location (as opposed to a node)

Link Reversal Algorithm [Gafni81]



Link Reversal Algorithm



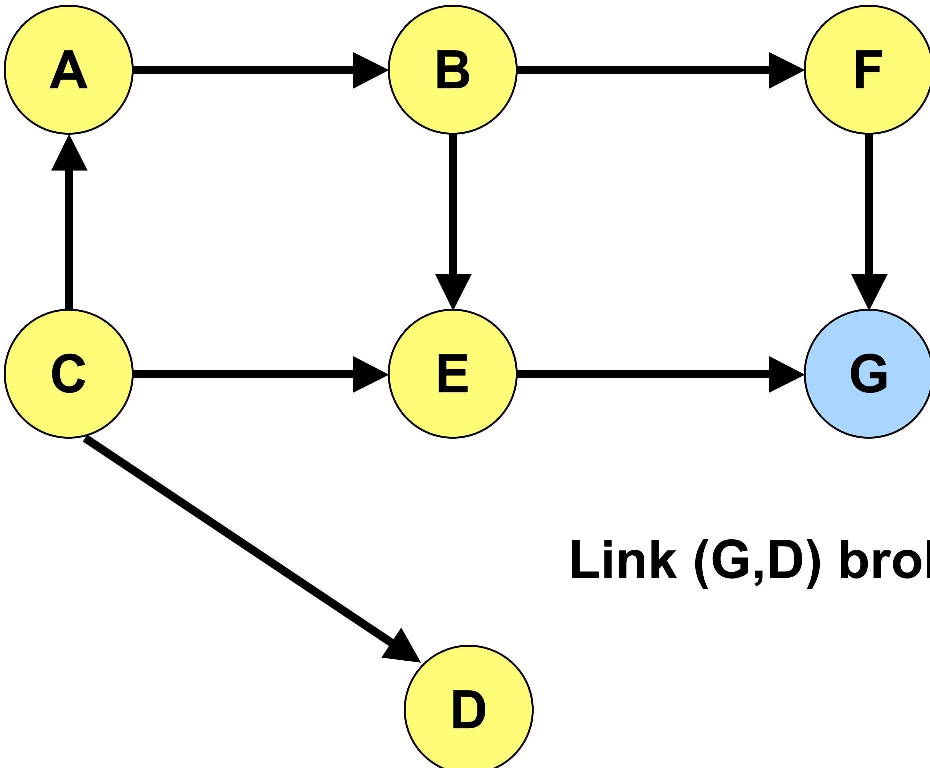
Links are bi-directional

But algorithm imposes logical directions on them

Maintain a directed acyclic graph (DAG) for each destination, with the destination being the *only sink*

This DAG is for *destination node D*

Link Reversal Algorithm

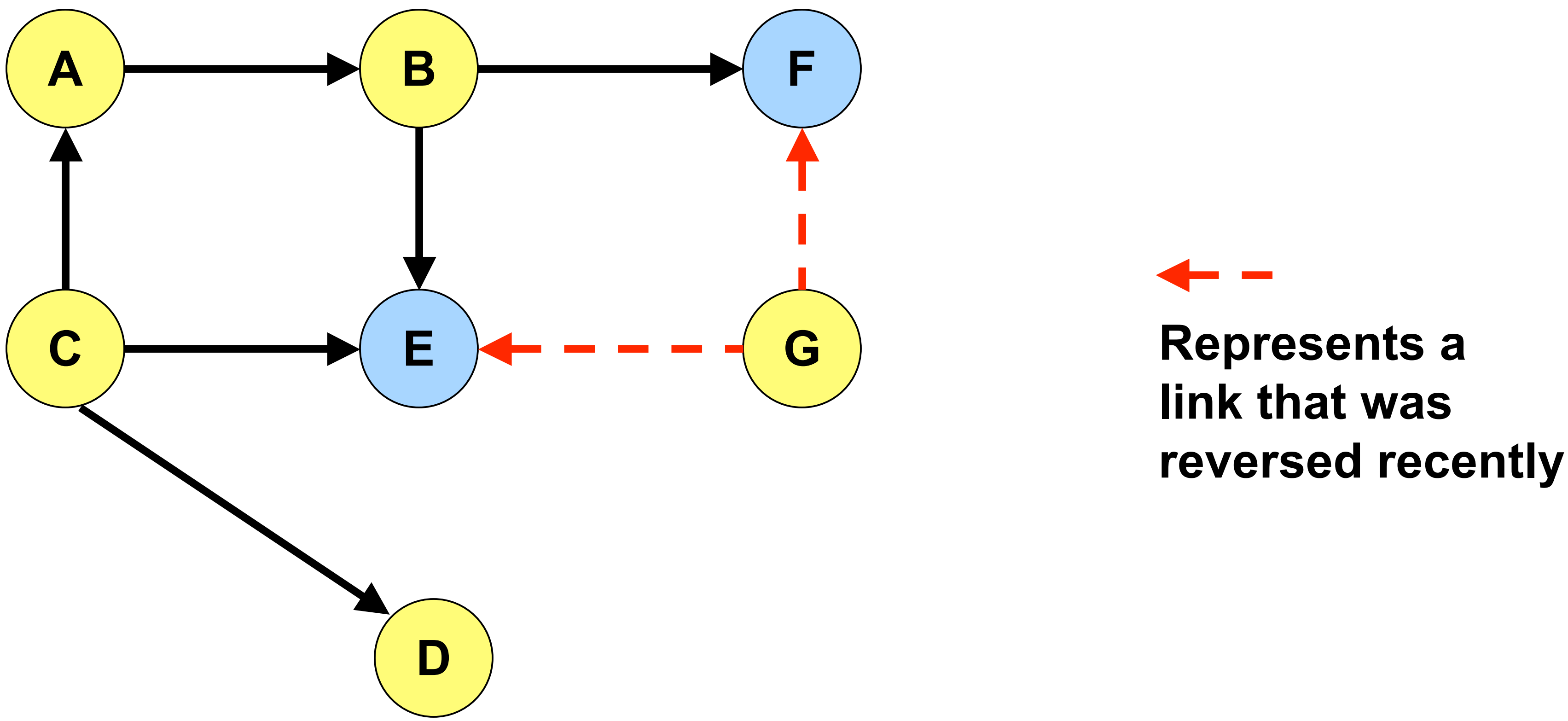


Link (G,D) broke

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

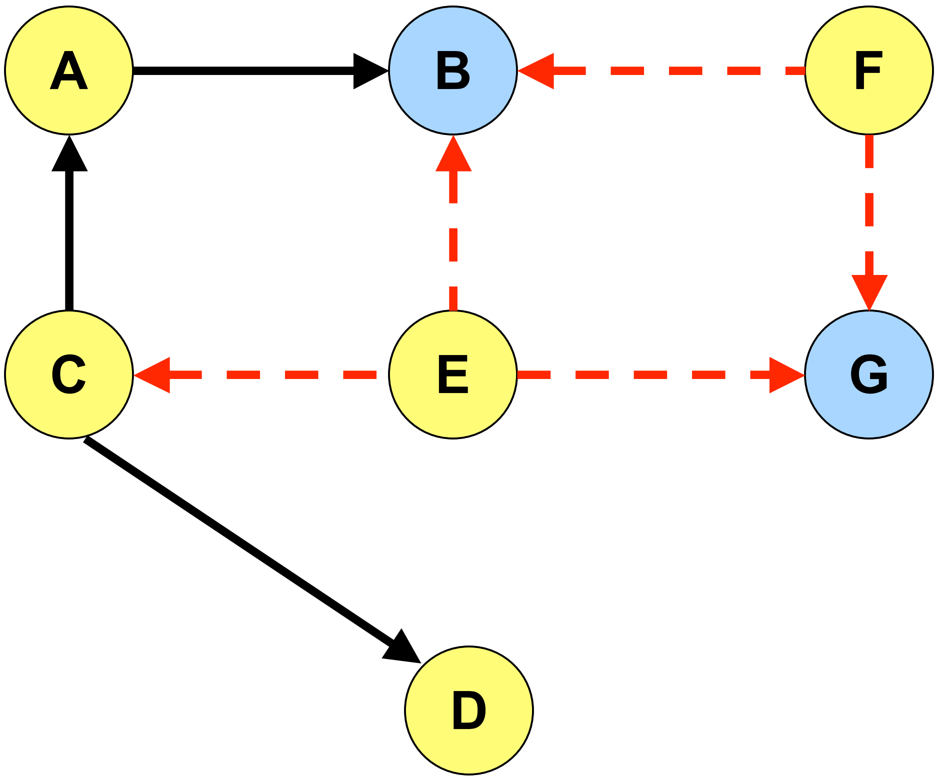
Node G has no outgoing links

Link Reversal Algorithm



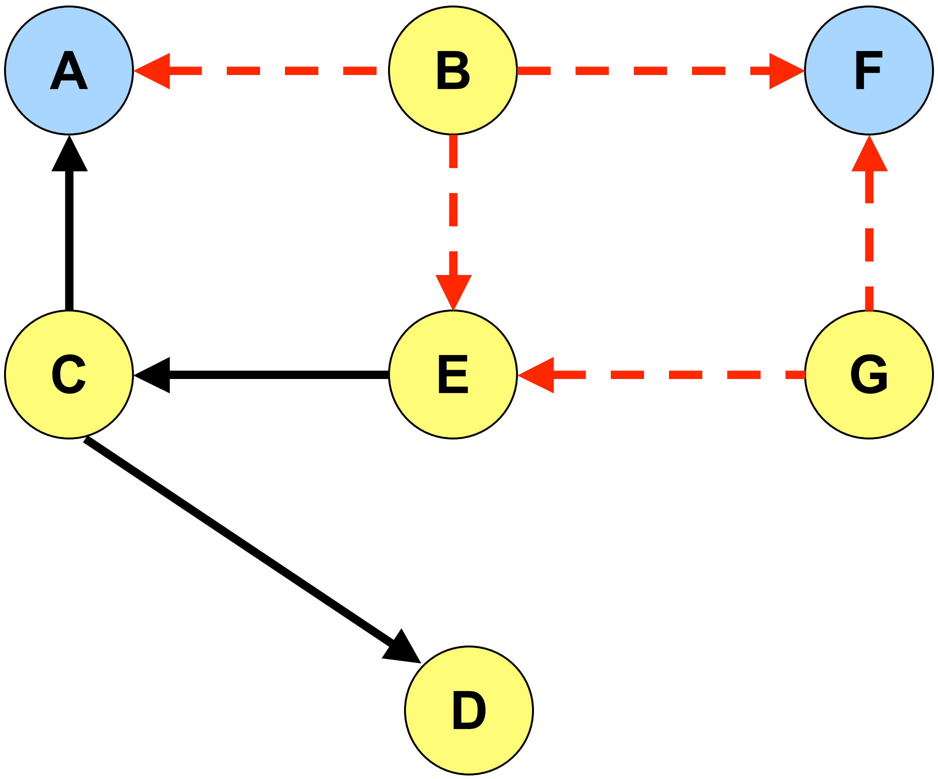
Now nodes E and F have no outgoing links

Link Reversal Algorithm



Now nodes B and G have no outgoing links

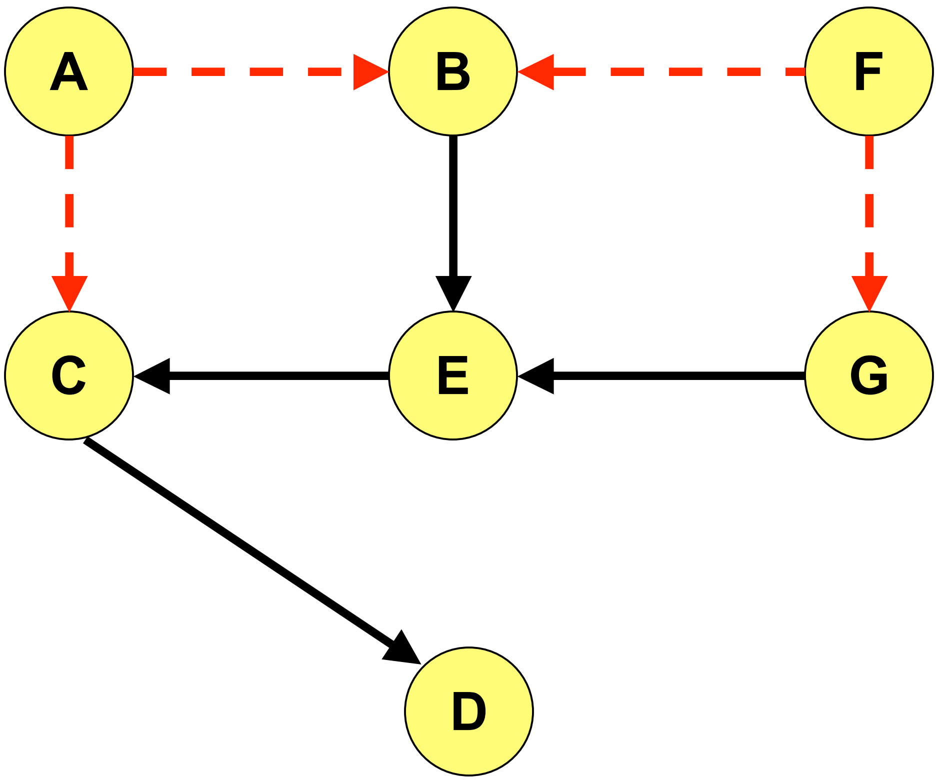
Link Reversal Algorithm



Represents a link that was reversed recently

Now nodes A and F have no outgoing links

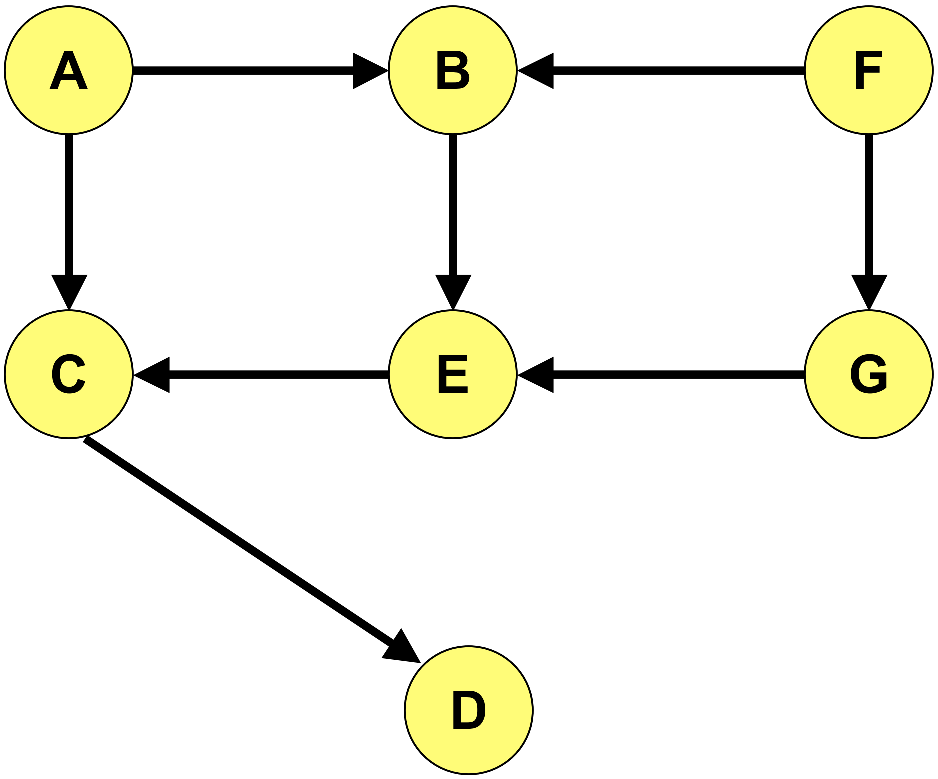
Link Reversal Algorithm




Represents a link that was reversed recently

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

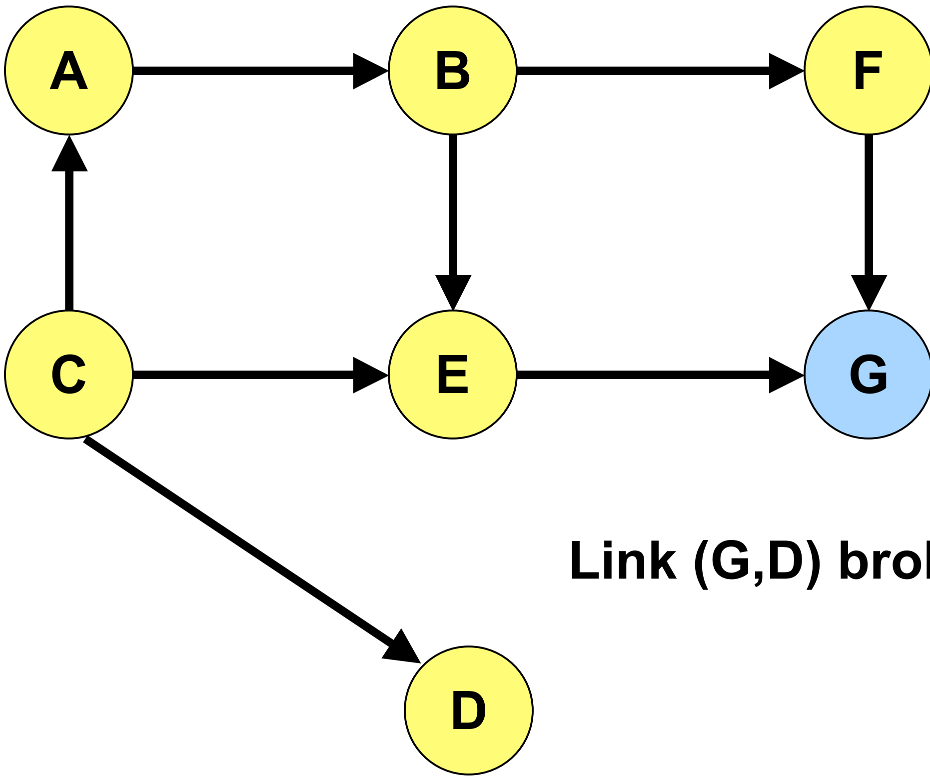
- Attempts to keep link reversals local to where the failure occurred
  - But this is not guaranteed
- When the first packet is sent to a destination, the destination oriented DAG is constructed
- The initial construction does result in flooding of control packets

Link Reversal Algorithm

- The previous algorithm is called a full reversal method since when a node reverses links, it reverses *all* its incoming links

- Partial reversal method [Gafni81]: A node reverses incoming links from only those neighbors who have not themselves reversed links "previously"
  - If all neighbors have reversed links, then the node reverses all its incoming links
  - "Previously" at node X means *since the last link reversal done by node X*

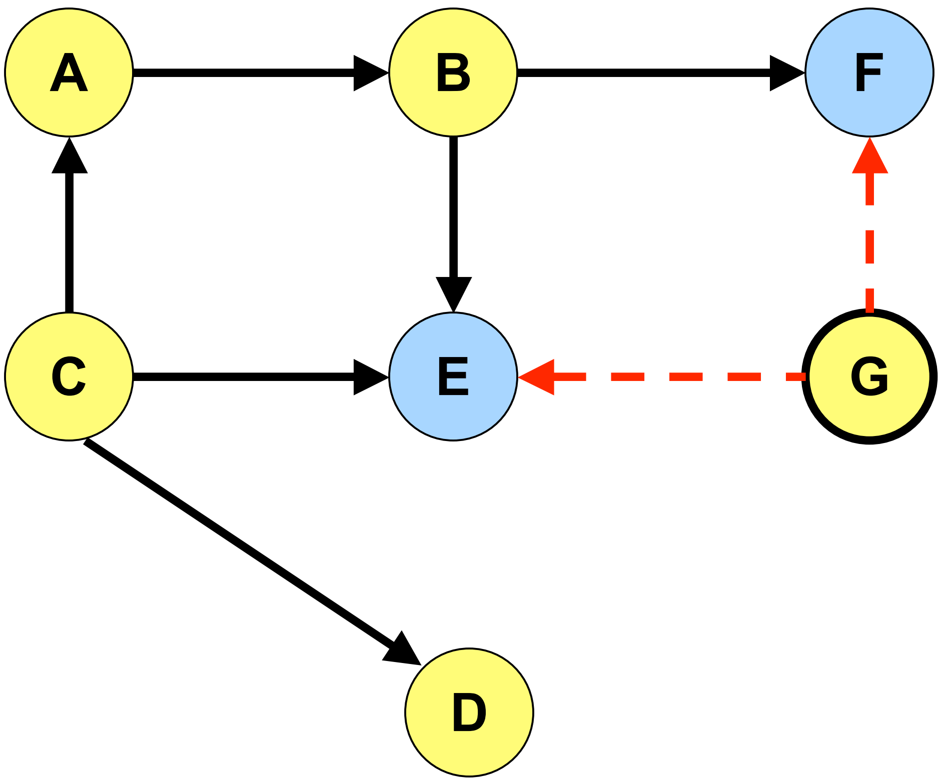
Partial Reversal Method

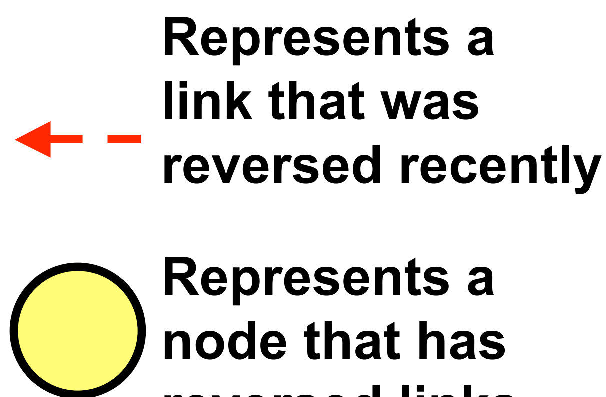


Link (G,D) broke

Node G has no outgoing links

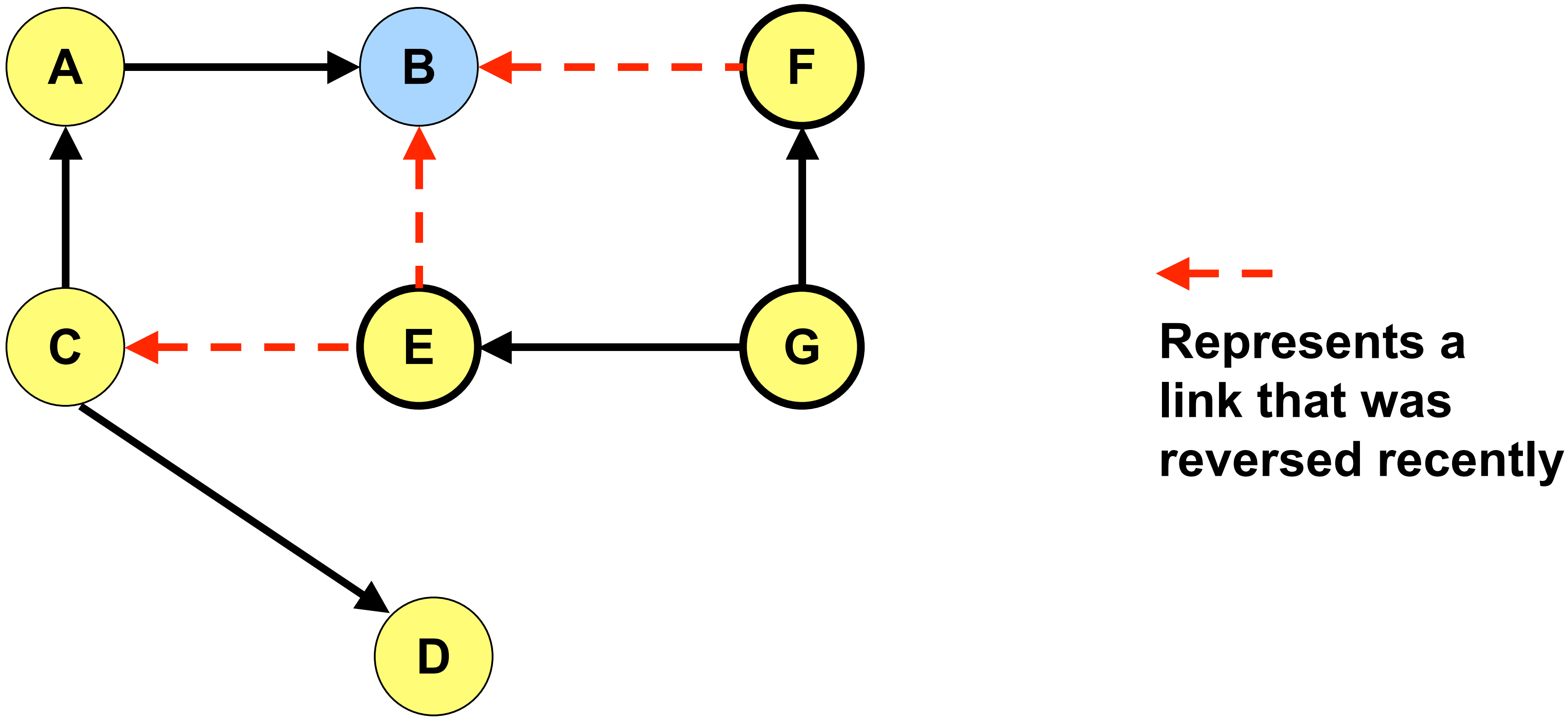
Partial Reversal Method





Now nodes E and F have no outgoing links

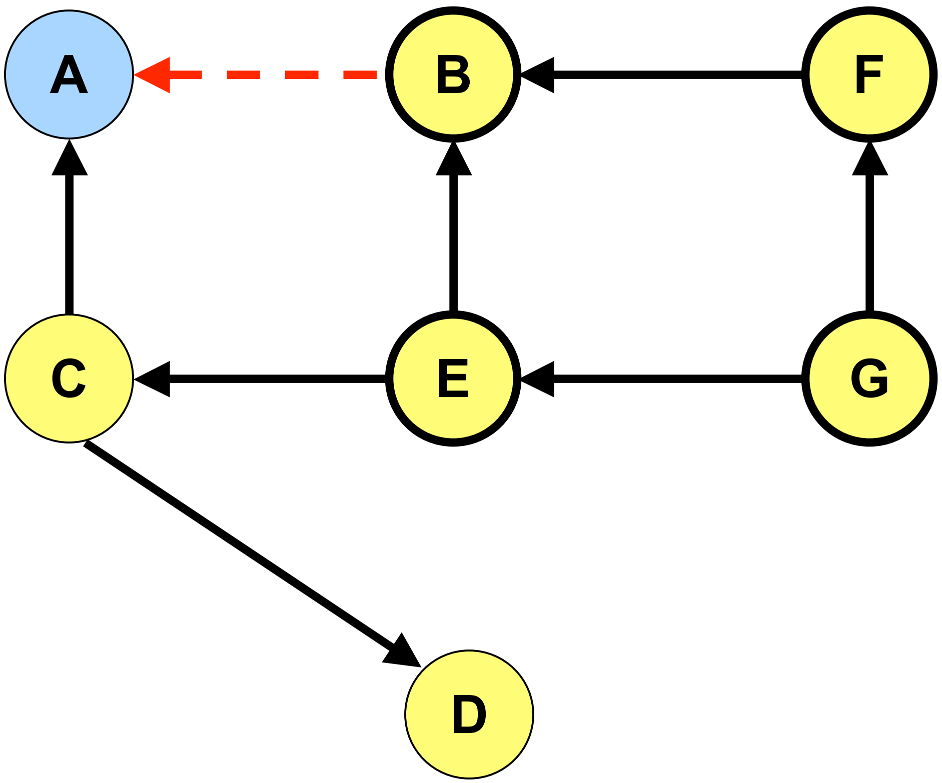
Partial Reversal Method



Nodes E and F *do not* reverse links from node G

Now node B has no outgoing links

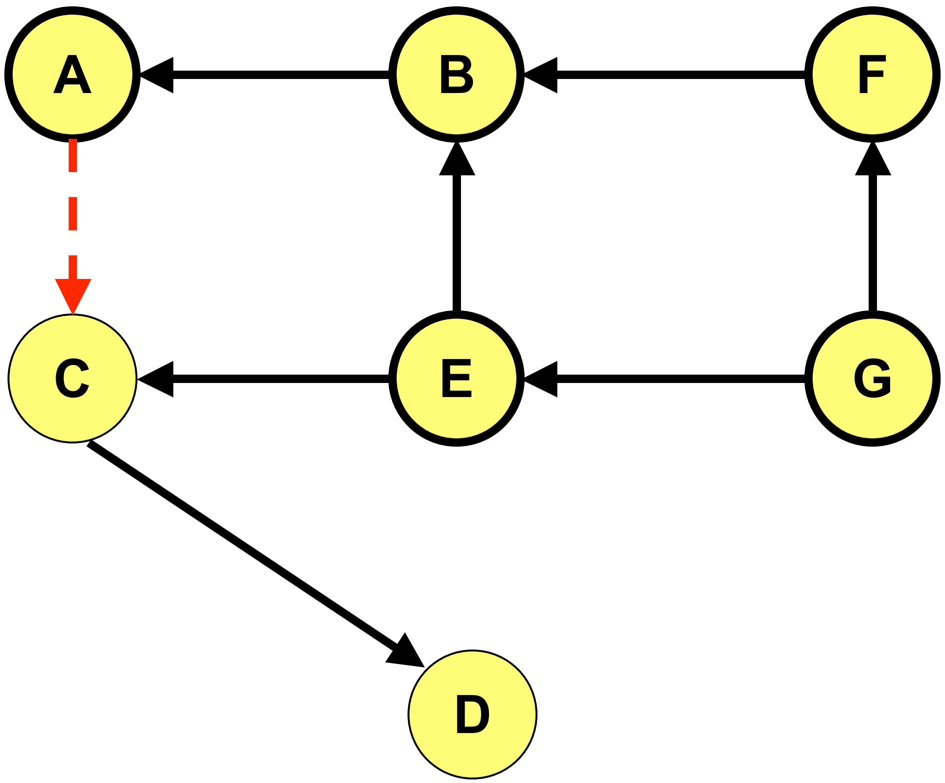
Partial Reversal Method



Represents a link that was reversed recently

Now node A has no outgoing links

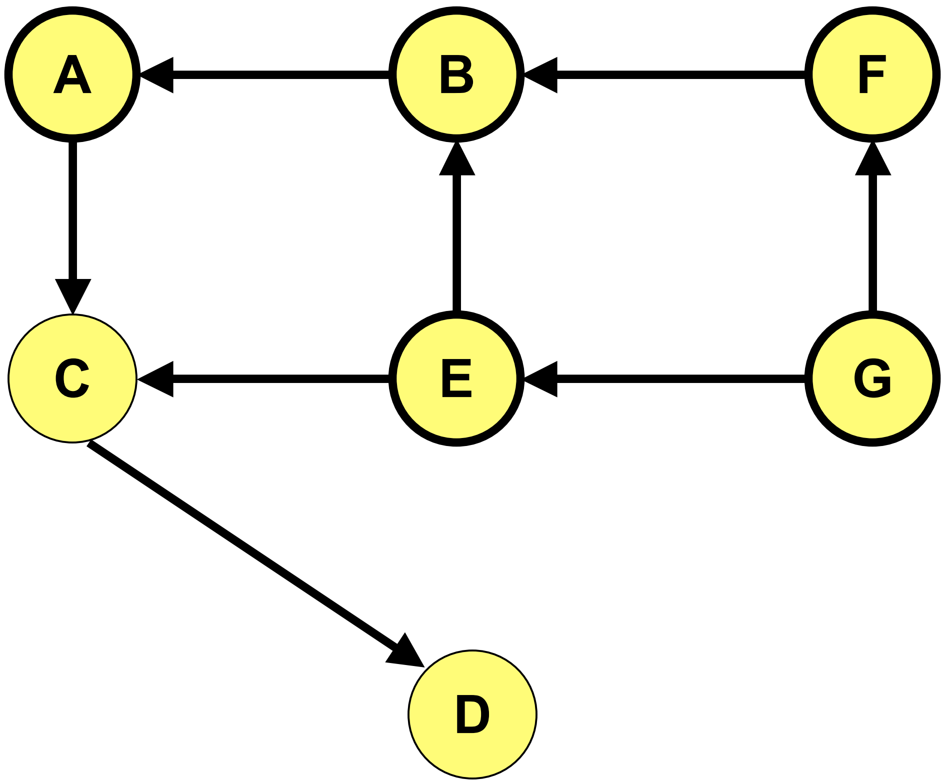
Partial Reversal Method



Represents a link that was reversed recently

Now all nodes (except destination D) have outgoing links

Partial Reversal Method



DAG has been restored with only the destination as a sink

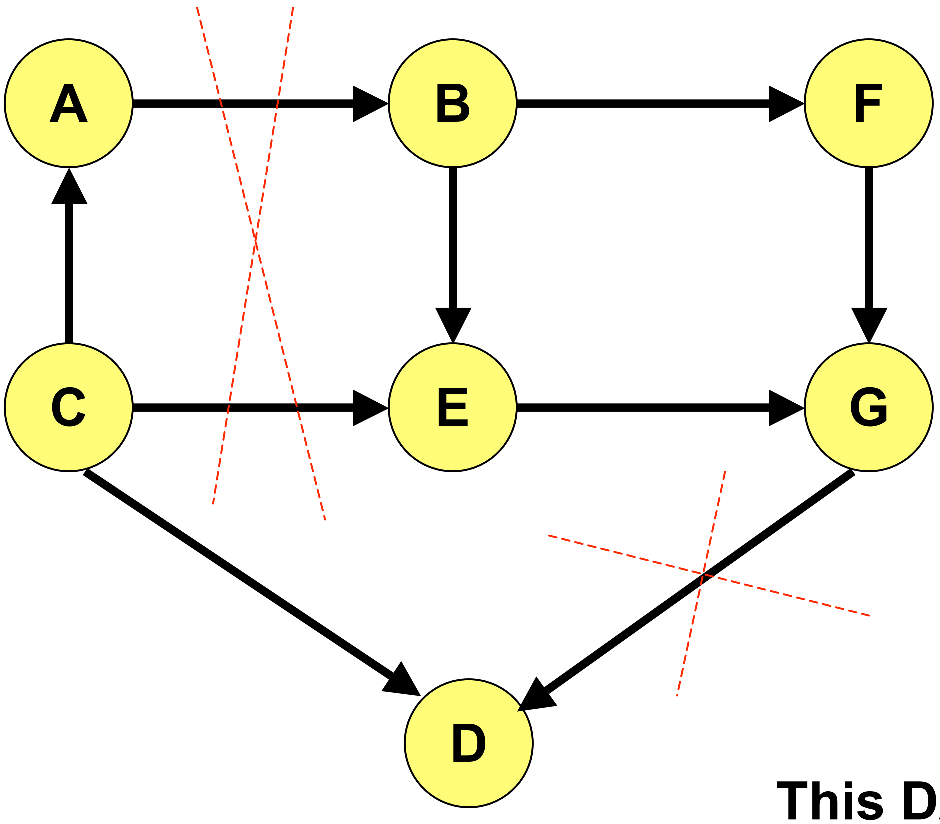
Link Reversal: Advantages

- Link reversal methods attempt to limit updates to routing tables at nodes in the vicinity of a broken link
  - Partial reversal method tends to be better than full reversal method
  - In the worst case, full reversal provably better than partial reversal [Busch-Tirthapura03]
- Each node may potentially have multiple routes to a destination

Link Reversal: Disadvantage

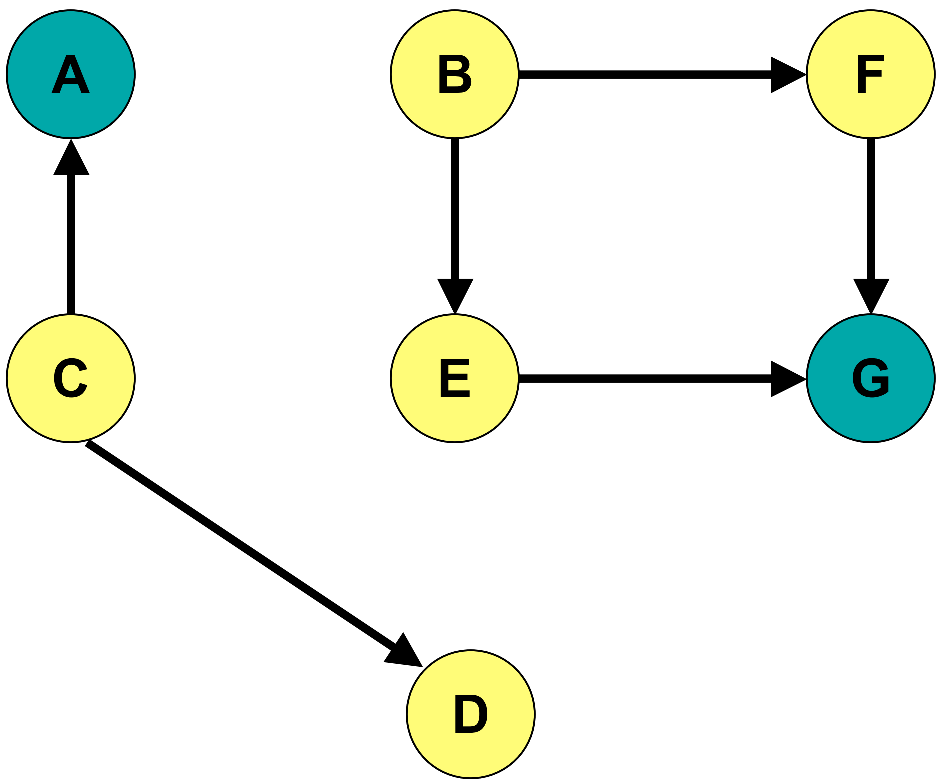
- Need a mechanism to detect link failure
  - o hello messages may be used
  - o but hello messages can add to contention
  - o this disadvantage also present in DSDV
- If network is partitioned, link reversals continue indefinitely

Link Reversal in a Partitioned Network



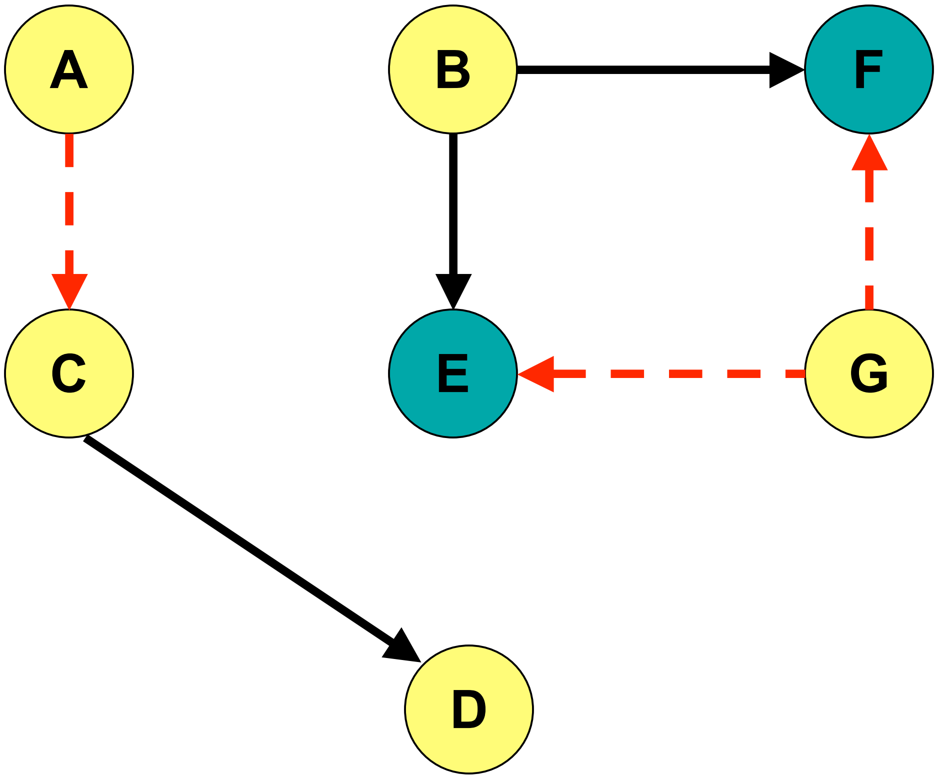
This DAG is for ***destination node D***

Full Reversal in a Partitioned Network



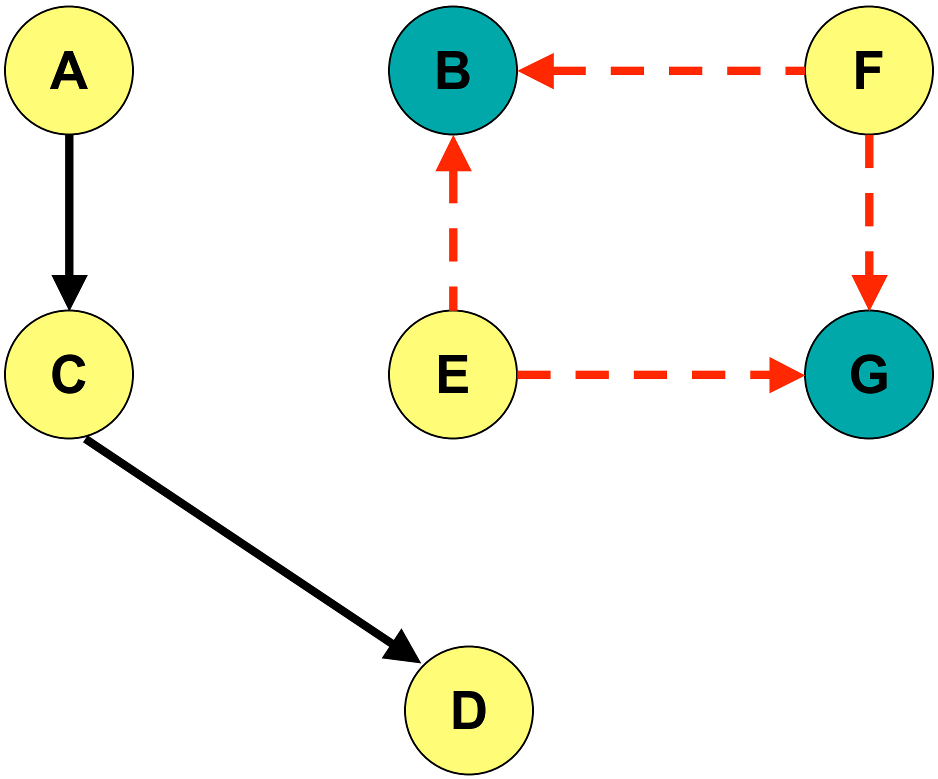
A and G do not have outgoing links

Full Reversal in a Partitioned Network



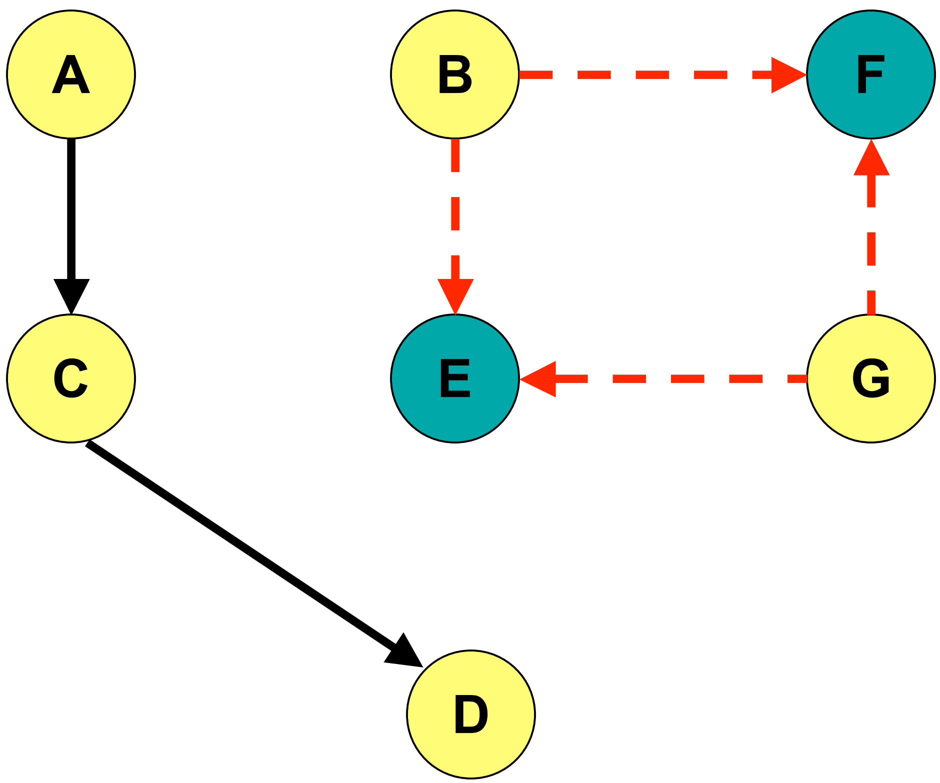
E and F do not have outgoing links

Full Reversal in a Partitioned Network



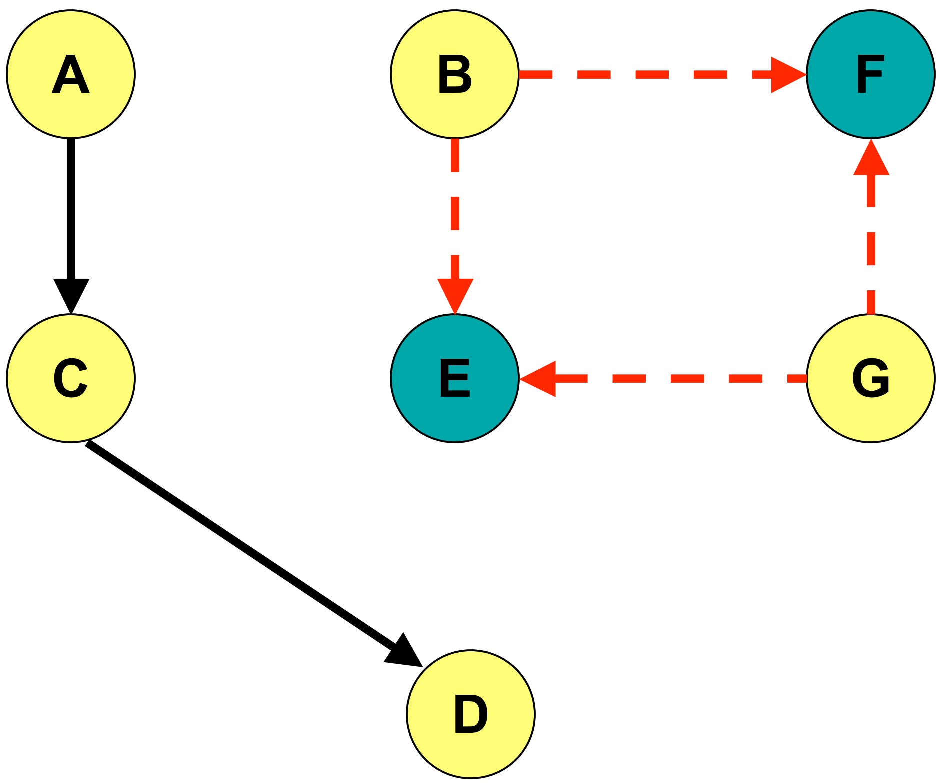
B and G do not have outgoing links

Full Reversal in a Partitioned Network



E and F do not have outgoing links

Full Reversal in a Partitioned Network



In the partition disconnected from destination D, link reversals continue, until the partitions merge

Need a mechanism to minimize this wasteful activity

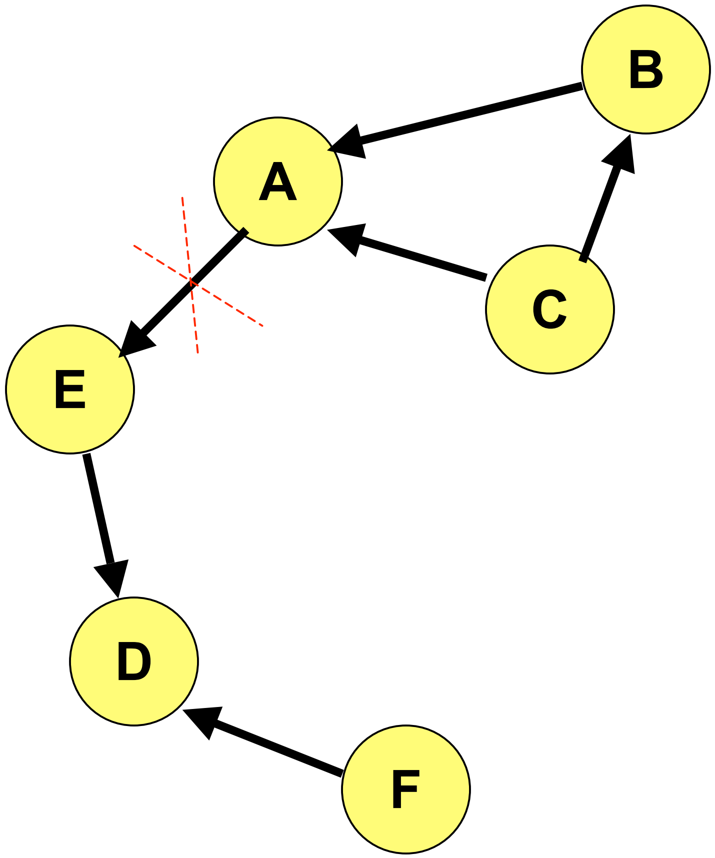
Similar scenario can occur with partial reversal method too

Temporally-Ordered Routing Algorithm (TORA) [Park-Corson97]

- TORA modifies the partial link reversal method to be able to detect partitions

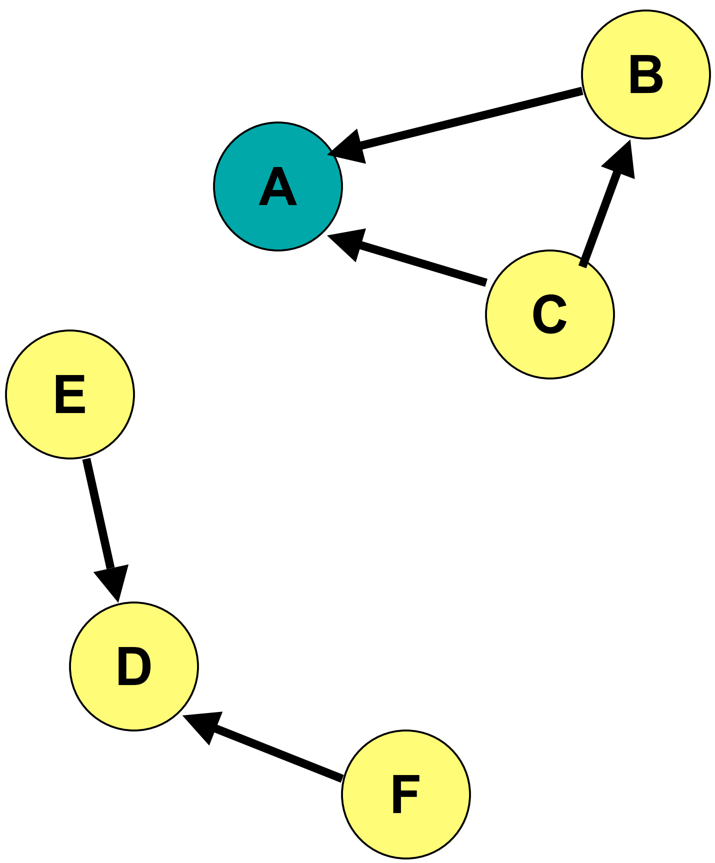
- When a partition is detected, all nodes in the partition are informed, and link reversals in that partition cease

Partition Detection in TORA



DAG for destination D

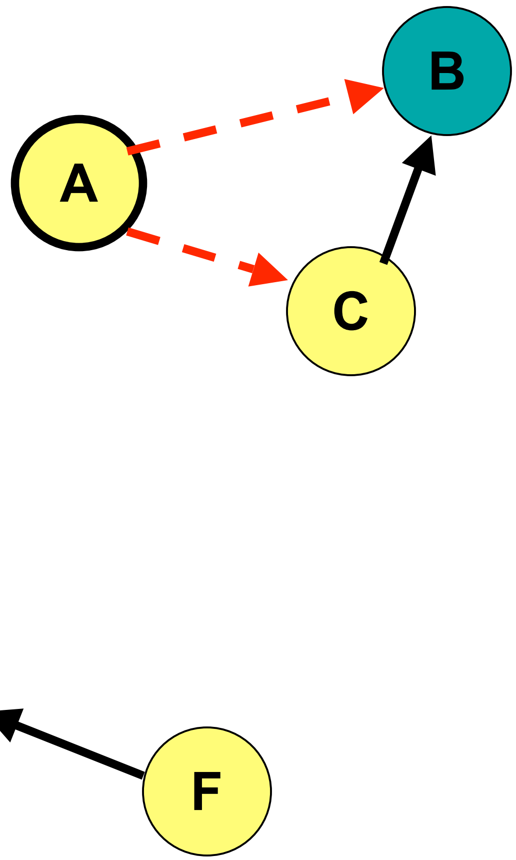
Partition Detection in TORA



Node A has no outgoing links

TORA uses a modified partial reversal method

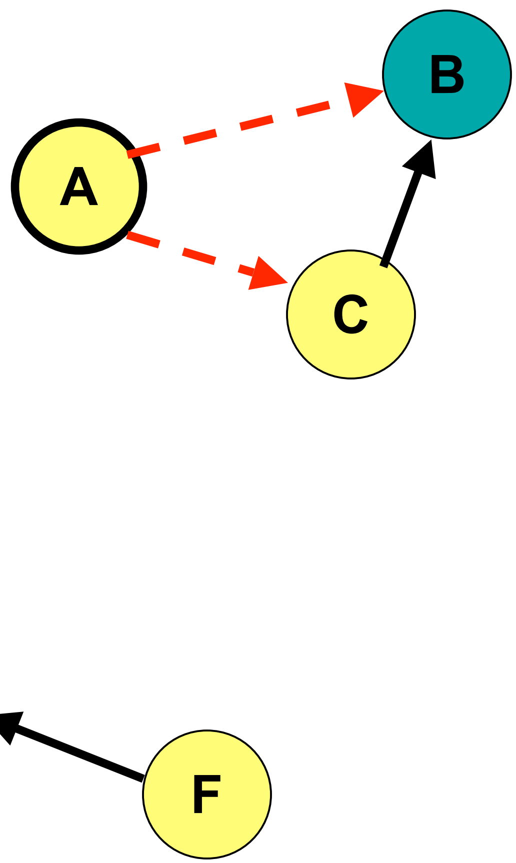
Partition Detection in TORA



TORA uses a modified partial reversal method

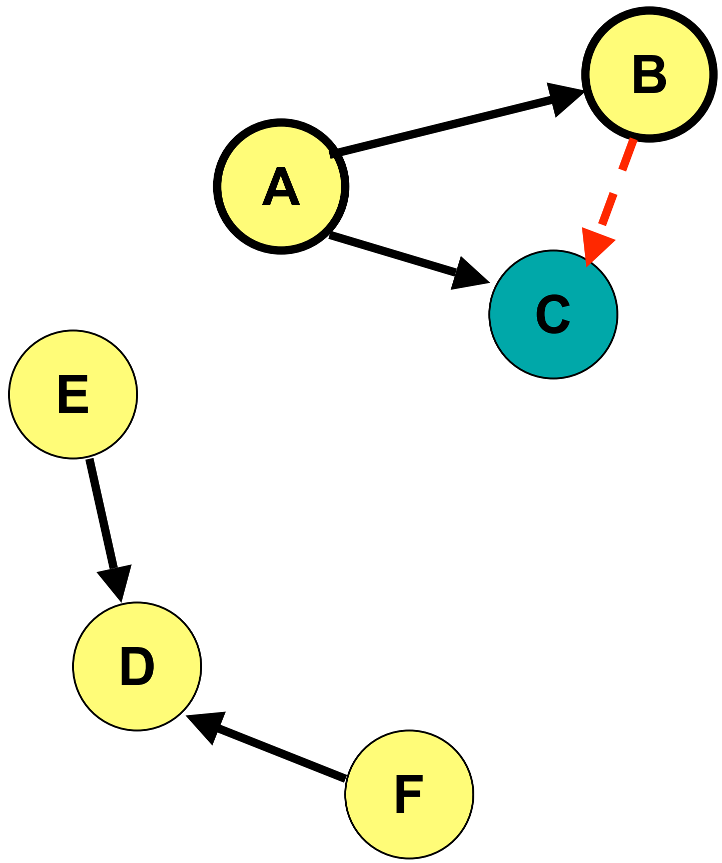
Node B has no outgoing links

Partition Detection in TORA



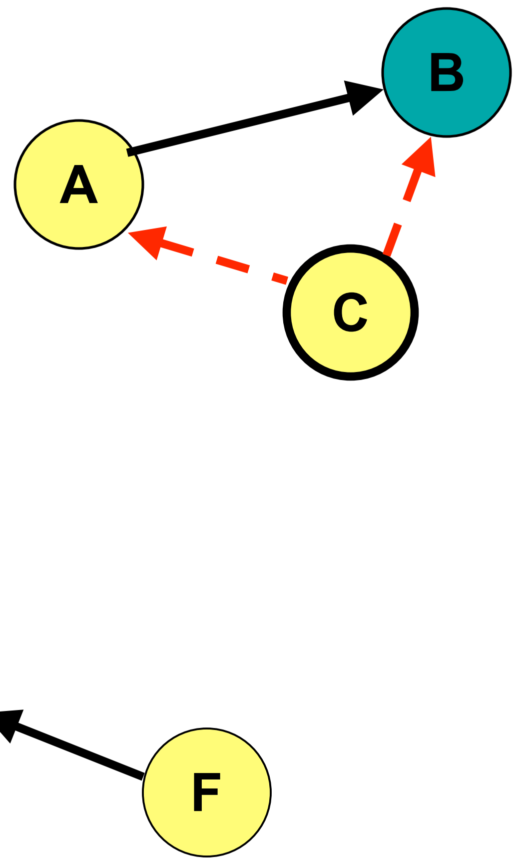
Node B has no outgoing links

Partition Detection in TORA



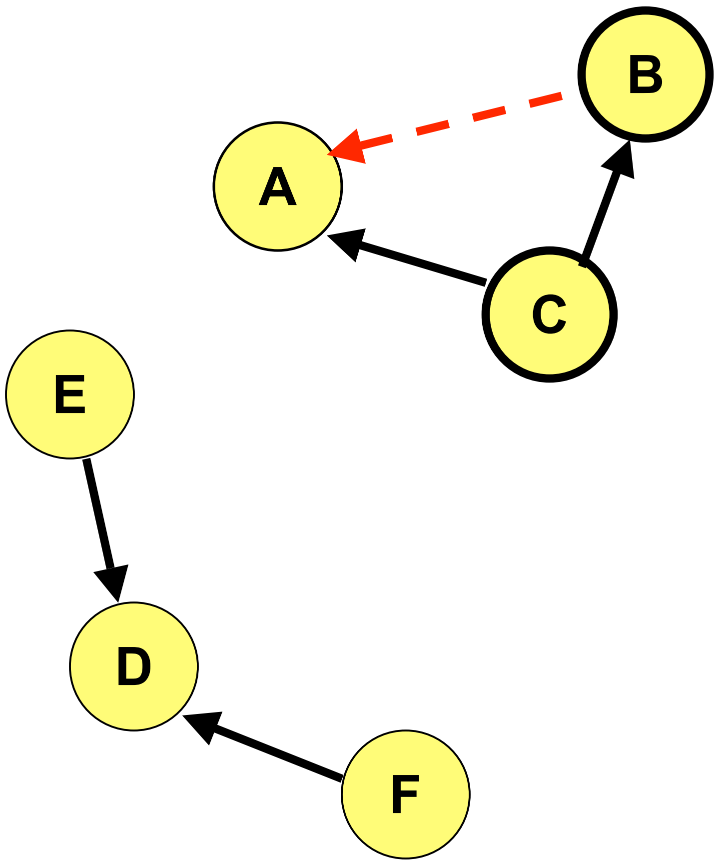
Node C has no outgoing links -- all its neighbor have reversed links previously.

Partition Detection in TORA



Nodes A and B receive the reflection from node C
Node B now has no outgoing link

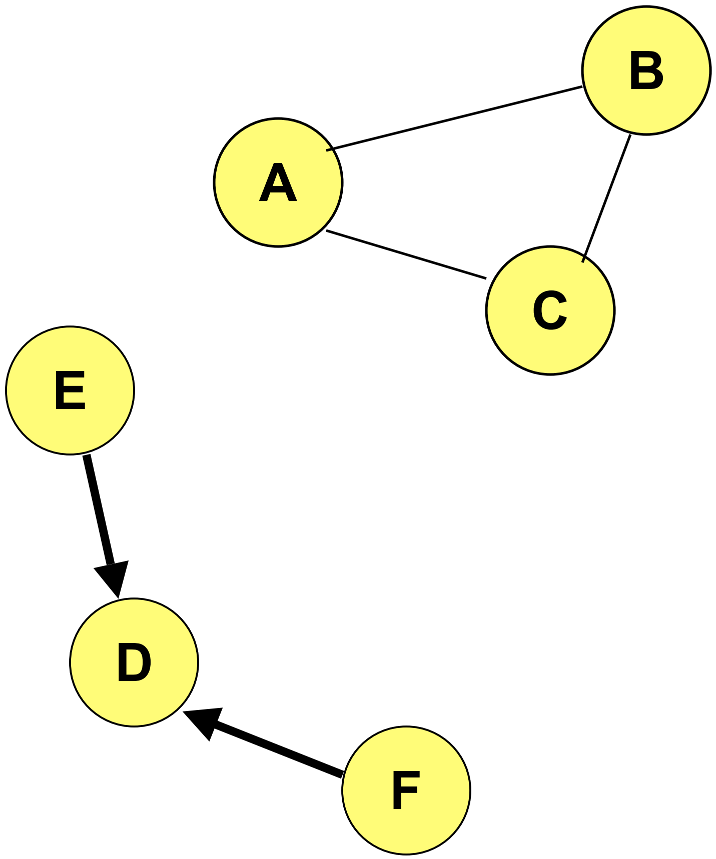
Partition Detection in TORA



Node B **propagates** the **reflection** to node A

Node A has received the **reflection from all its neighbors**.
Node A determines that it is partitioned from destination D.

Partition Detection in TORA



On detecting a partition, node A sends a clear (CLR) message that purges all directed links in that partition

TORA

- Improves on the partial link reversal method in [Gafni81] by detecting partitions and stopping non-productive link reversals
- Paths may not be shortest
- The DAG provides many hosts the ability to send packets to a given destination
  - Beneficial when many hosts want to communicate with a single destination

TORA Design Decision

- TORA performs link reversals as dictated by [Gafni81]
- However, when a link breaks, it loses its direction

- When a link is repaired, it may not be assigned a direction, unless some node has performed a route discovery after the link broke
  - if no one wants to send packets to D anymore, eventually, the DAG for destination D may disappear

- TORA makes effort to maintain the DAG for D only if someone needs route to D
  - Reactive behavior

TORA Design Decision

- One proposal for modifying TORA optionally allowed a more proactive behavior, such that a DAG would be maintained even if no node is attempting to transmit to the destination

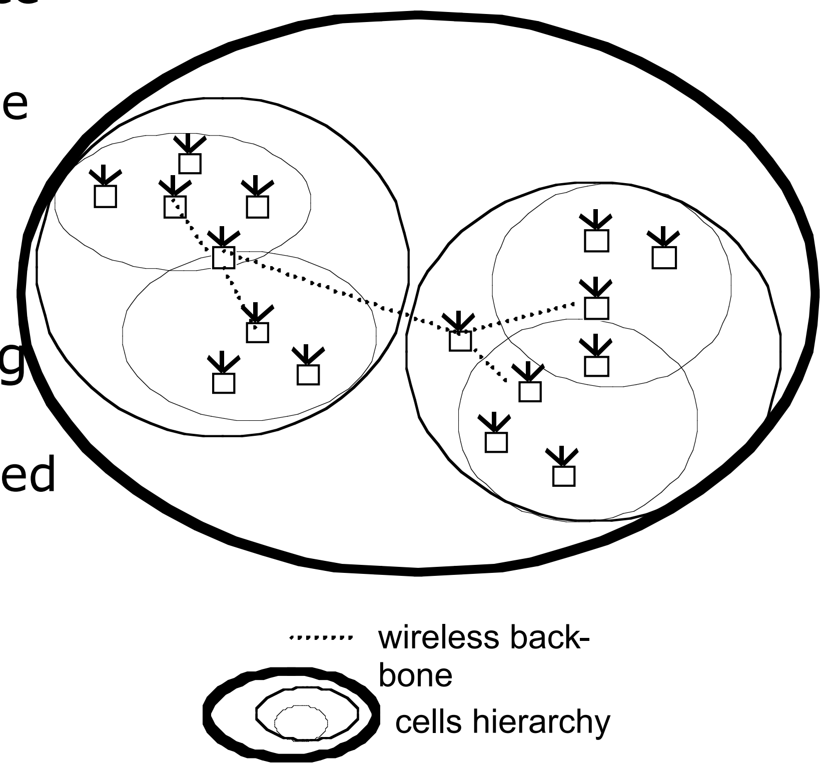
- Moral of the story: The link reversal algorithm in [Gafni81] does not dictate a proactive or reactive response to link failure/repair

- Decision on reactive/proactive behavior should be made based on environment under consideration

Clustering Approaches To Multi-hop Ad Hoc Networks

Clustering

- Goal:
  - Reduce channel contention
  - Form routing backbone to reduce network diameter
  - Abstract network state to reduce its quantity and its variability
- Various approaches to clustering
  - Started in the 70s with Packet Radio Network (PRNet) sponsored by DARPA



Clustering for Transmission Management

- Goal: reduce contention
- Cluster = clusterhead + gateways + ordinary nodes
- Roles:
  - Clusterhead: schedules traffic, allocates resources (tokens, emits busy tone, etc.). Similar to the master in a Bluetooth piconet.
  - Gateways: interconnect clusters
  - Ordinary nodes are 1-hop away from a clusterhead and 2-hops away from other members in the cluster
- Tasks:
  - Connectivity discovery
  - Election of clusterheads
  - Agree on Gateways

Clustering for Transmission Management (Cont)

- Clusterhead election:
  - Centralized/distributed algorithms
  - Node identifier/degree based
  - Principles:
    - Centralized: (1) elect the highest ID node and create a corresponding cluster, repeat step (1) with nodes not already members of a cluster
    - Distributed:
      - a node elects itself as cluster head if it has the highest ID among its neighbors
      - otherwise elect a neighbor that is not member of another cluster
  - Leads to disjoint clusters
- Gateways:
  - If connected to > 1 cluster => gateway candidate
  - When multiple candidates to connect two clusters, choose GW with highest ID

Clustering for Transmission Management (Cont)

- Mobility:
  - When node finds it is not close to a clusterhead, it can initiate an election process
  - When two clusterheads become neighbors, they can merge clusters
    - This may trigger other clusterhead elections and/or mergers
- Routing:
  - To avoid clusterhead congestion and improve robustness, routing is done over the flat network

Clustering for Backbone Formation

- Wireless multiphop networks have high end-to-end delay:
  - link-layer ARQ, MAC delay, FEC/spreading, tx/rx switching time
- Clustering can reduce the end-to-end delay by allowing faster forwarding through the clusterheads backbone
- Approaches:
  - Near-Term Digital Radio Network (NTDR) [Zavgren 1997]
  - Virtual Subnet Architecture [Sharony 1996]

Overview of Geometric Routing

- Assumptions:
  - Each node is aware of its physical location, e.g., using GPS
  - Source knows the location of the destination node
- Use the underlying geometry to direct the packet to the desired destination
- Two components:
  - Greedy routing: send packet to a neighbor that is closer to destination
  - Face routing: route along a planar subgraph of the transmission graph
- [Bose et al 99] and GPSR [Karp-Kung 00]

Greedy Routing

- Send packet to neighbor that is closer to destination
- Greedily arrive closer and closer to destination
- If greedy routing persists, then eventually reach destination
- Problem:
  - Greedy routing not always possible
  - All the neighbors of a node may be farther from the destination

Face routing: Planar subgraph

- If greedy routing not possible, can route along a planar subgraph
- Gabriel graph: A planar graph in which any edge (u,v) satisfies

$$d(u,v)^2 \leq d(u,w)^2 + d(v,w)^2$$

- Given graph is connected iff the Gabriel subgraph is connected
- Can find Gabriel subgraph locally

Face routing in planar subgraph

- If the packet is at node v and destination is d, identify face adjacent to v that intersects line vd
- Route along the face using the right-hand rule, until a node is found that is closer to d than v is
- Combine face routing and greedy routing