Routing in Mobile Ad Hoc Networks

CS 441

Slides adopted from Nitin Vaidya, UIUC

Mobile Ad Hoc Networks

- Formed by wireless hosts which may be mobile
- Without using a pre-existing infrastructure
- Multi-hop routes between mobile nodes

Why Ad Hoc Networks ?

- Ease of deployment
- Speed of deployment
- Decreased dependence on infrastructure

The Holy Grail

- A one-size-fits-all solution
  - Perhaps using an adaptive/hybrid approach that can adapt to situation at hand
- Difficult problem
- Many solutions proposed trying to address a sub-space of the problem domain

Unicast Routing

in Mobile Ad Hoc Networks (MANET)

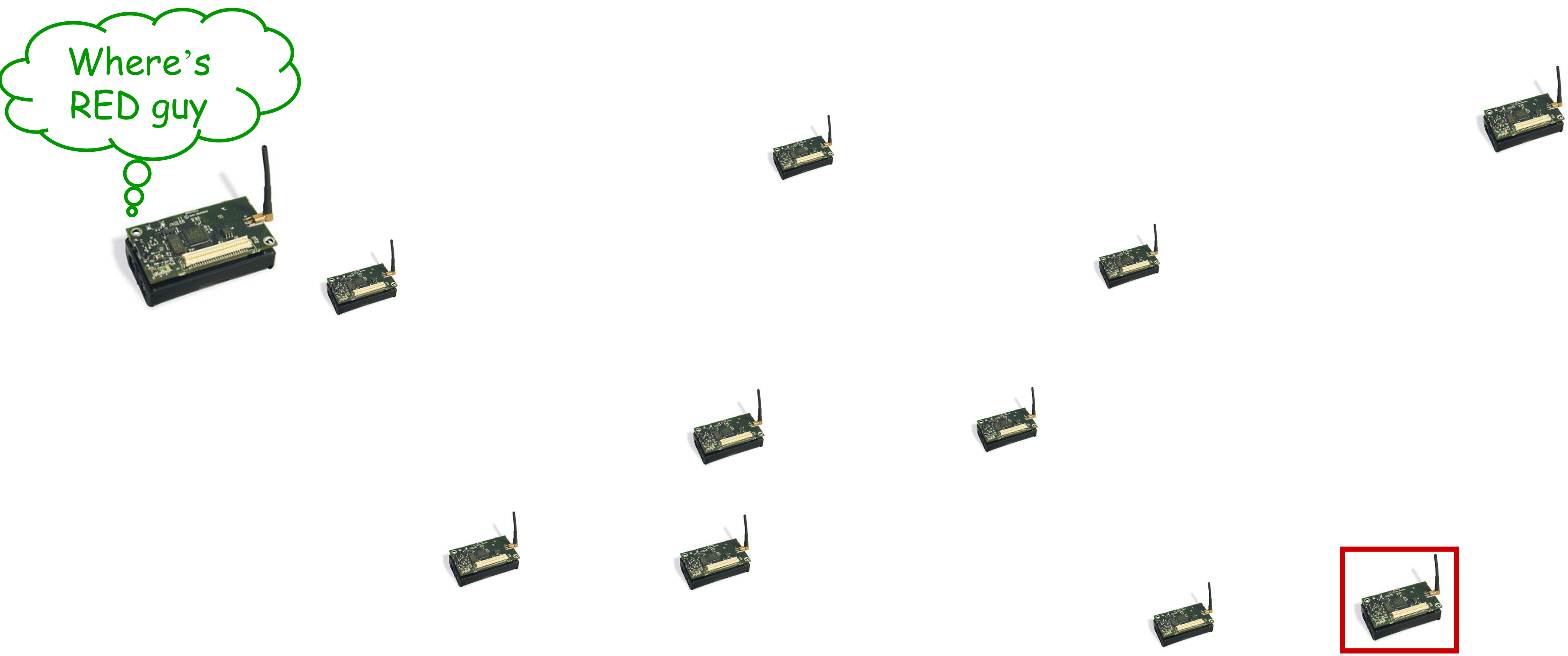
Wireless Routing

- Link instability causes many routing issues
  - Shortest hop routing often worst choice
  - Scarce bandwidth makes overhead conspicuous
  - Battery power a concern
  - Security and misbehavior …

- If that’s not bad enough
  - Add node mobility
    - Note: Routes may break, and reconnect later

Routing in wireless Mobile Networks

- Imagine hundreds of hosts moving
  - Routing algorithm needs to cope up with varying wireless channel and node mobility



Unicast Routing Protocols

- Many protocols have been proposed
- Some have been invented specifically for MANET
- Others are adapted from wired network routing
- No single protocol works well in all environments
  - some attempts made to develop adaptive protocols

Routing Protocols

- Proactive protocols
  - Determine routes independent of traffic pattern
  - Traditional link-state and distance-vector routing protocols are proactive
- Reactive protocols
  - Maintain routes only if needed
- Hybrid protocols
  - Maintain routes to nearby nodes
  - Discover routes for far away nodes

Trade-Off

- Latency of route discovery
- Overhead of route discovery/maintenance
- What is the relationship with mobility?
- What relationship to traffic?

Trade-Off

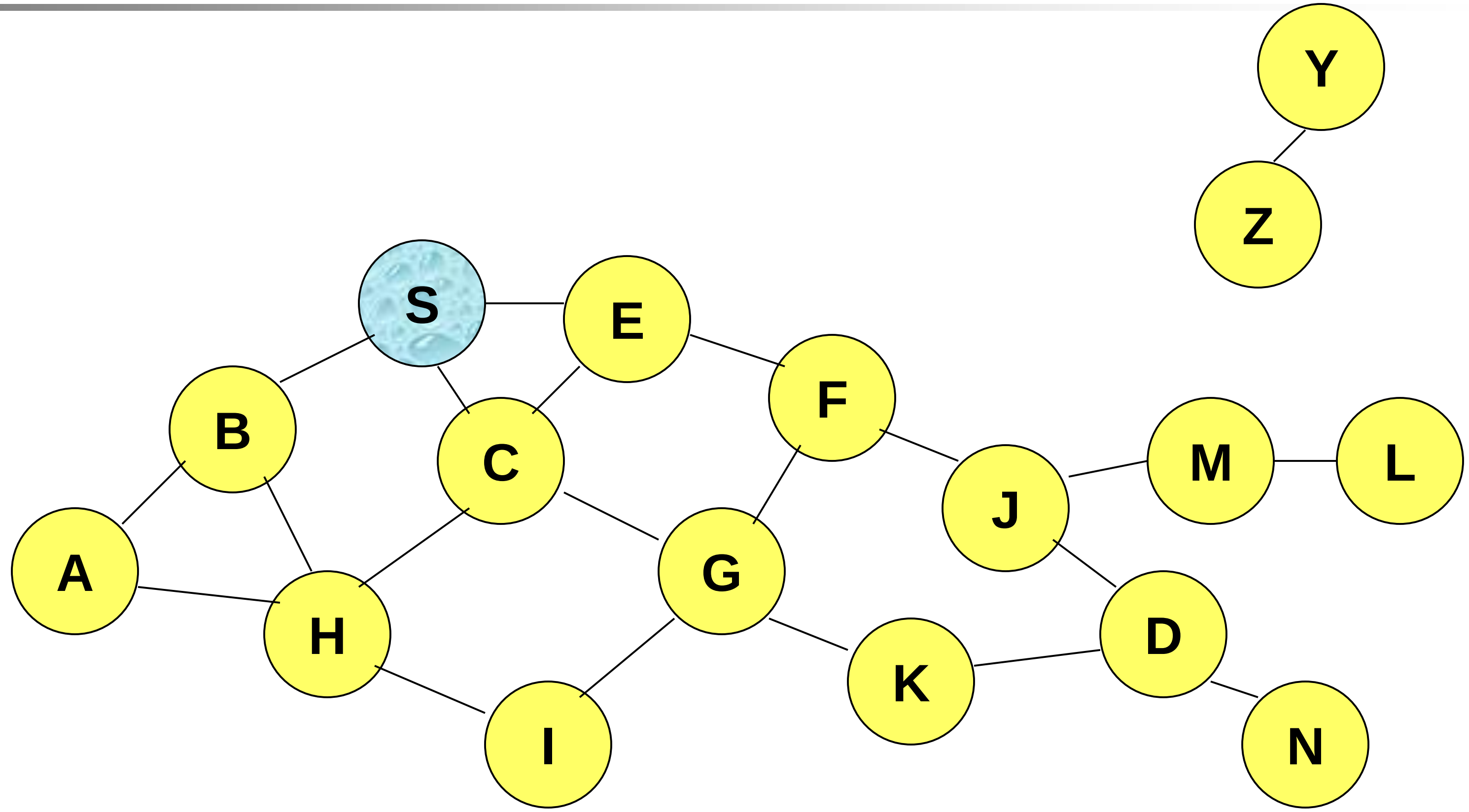
- Latency of route discovery
  - Proactive protocols may have lower latency
  - Reactive protocols higher because a route discovery from X to Y will be initiated only when X attempts to send to Y
- Overhead of route discovery/maintenance
  - Reactive protocols may have lower overhead since routes are determined only if needed
  - Proactive protocols do continuous route updating / maintenance
- Which approach achieves a better trade-off depends on the traffic and mobility patterns

Overview of Unicast Routing Protocols

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

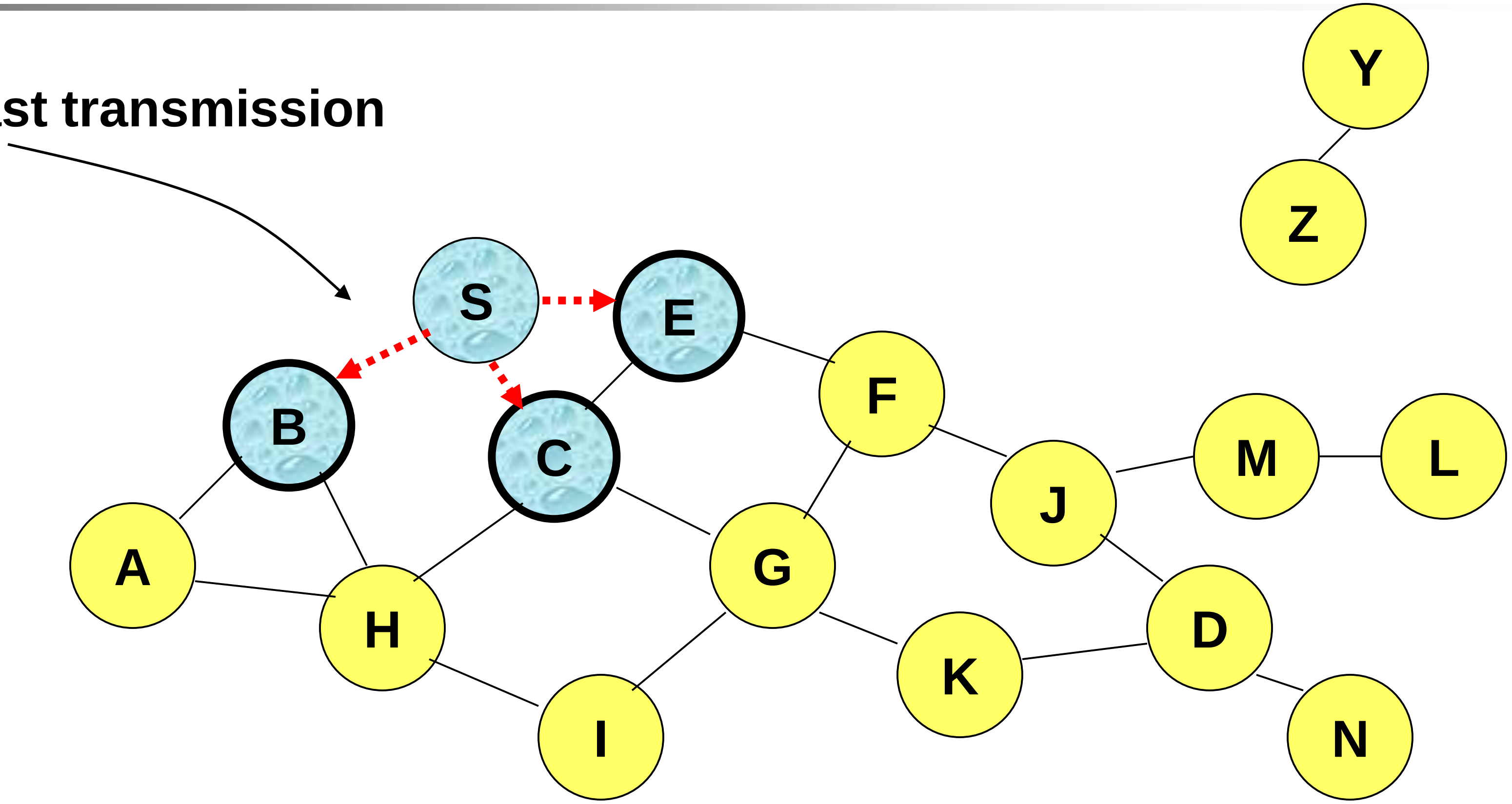


Represents a node that has received packet P

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

Flooding for Data Delivery

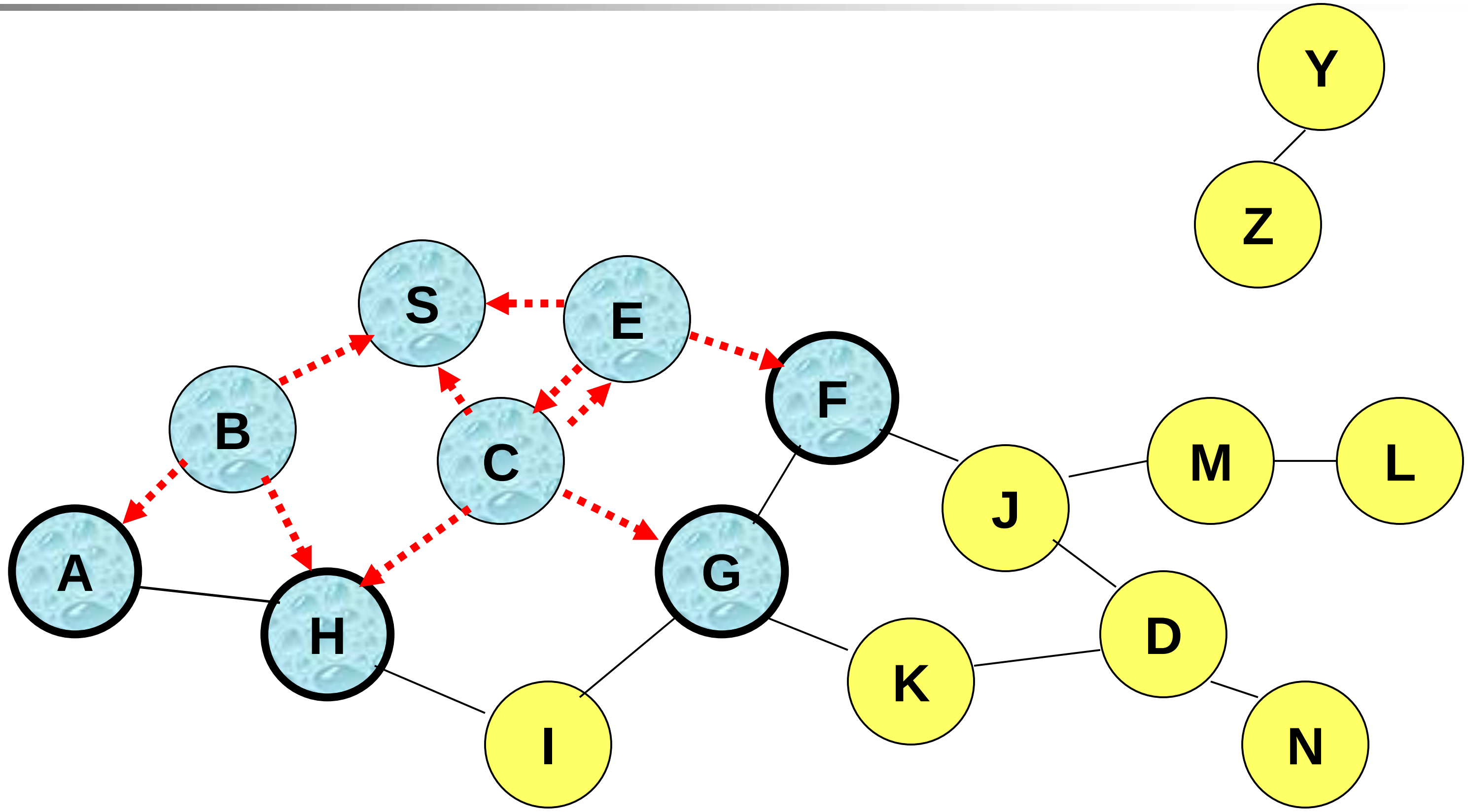
Broadcast transmission



 **Represents a node that receives packet P for the first time**

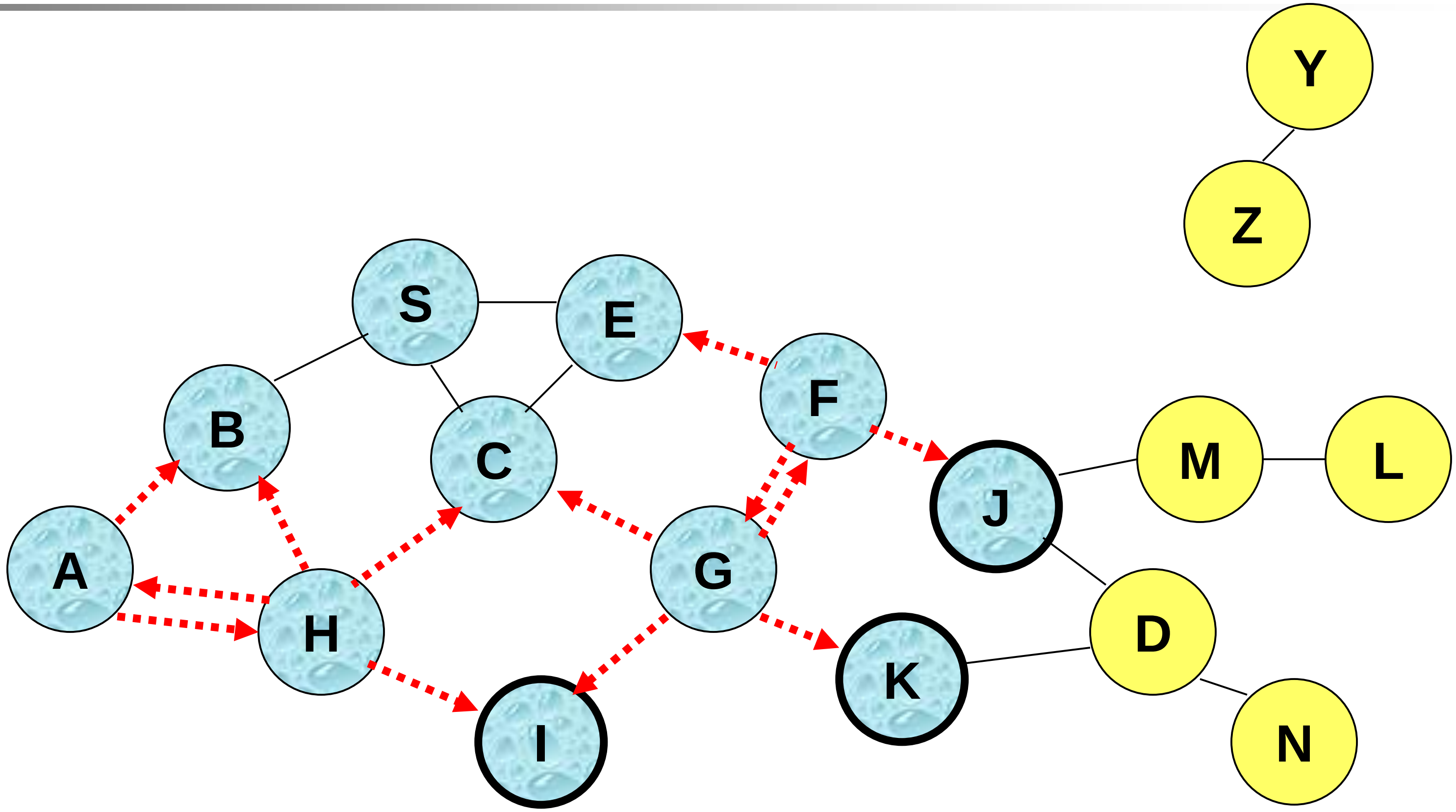
 **Represents transmission of packet P**

Flooding for Data Delivery



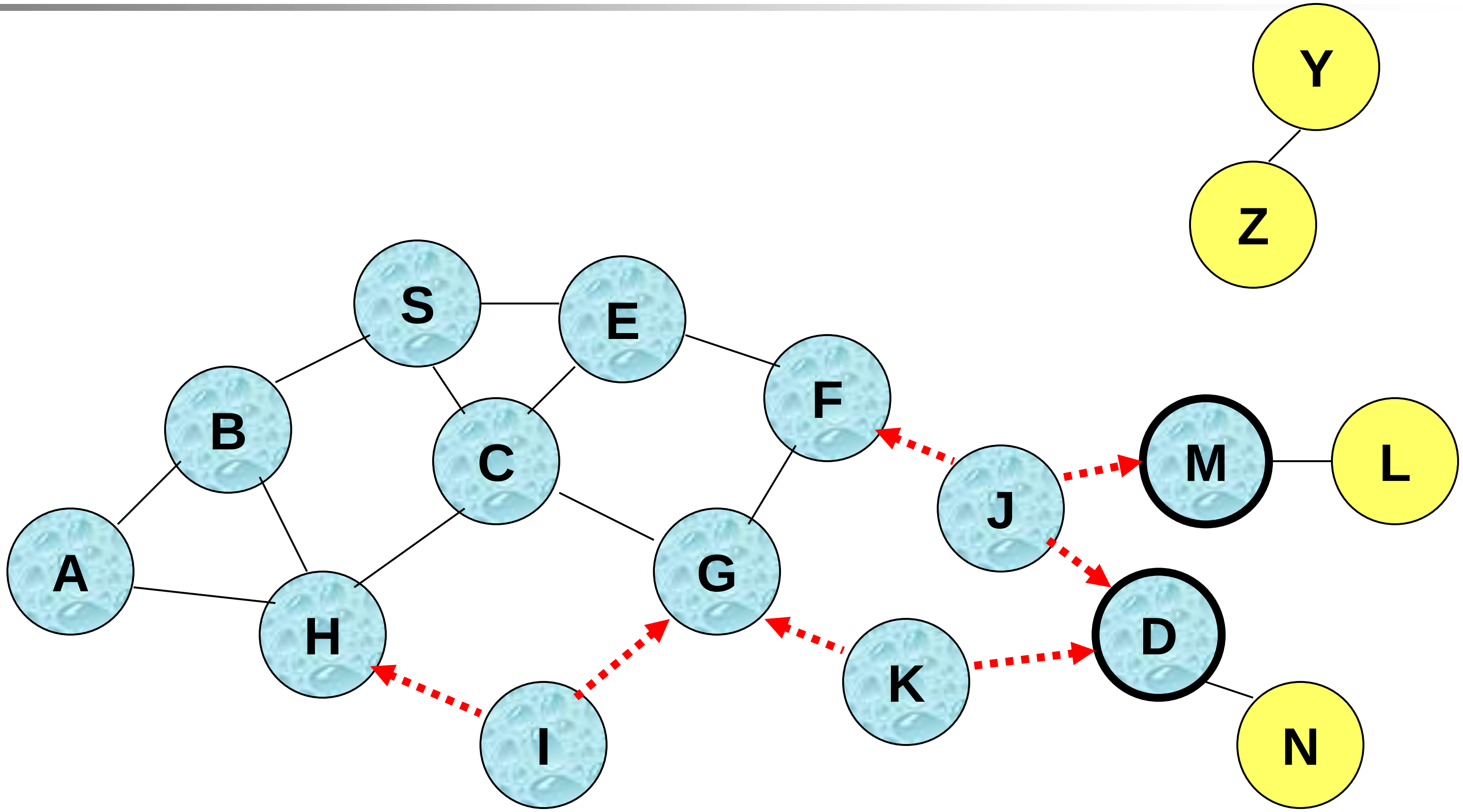
- **Node H receives packet P from two neighbors:**
  potential for collision

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**

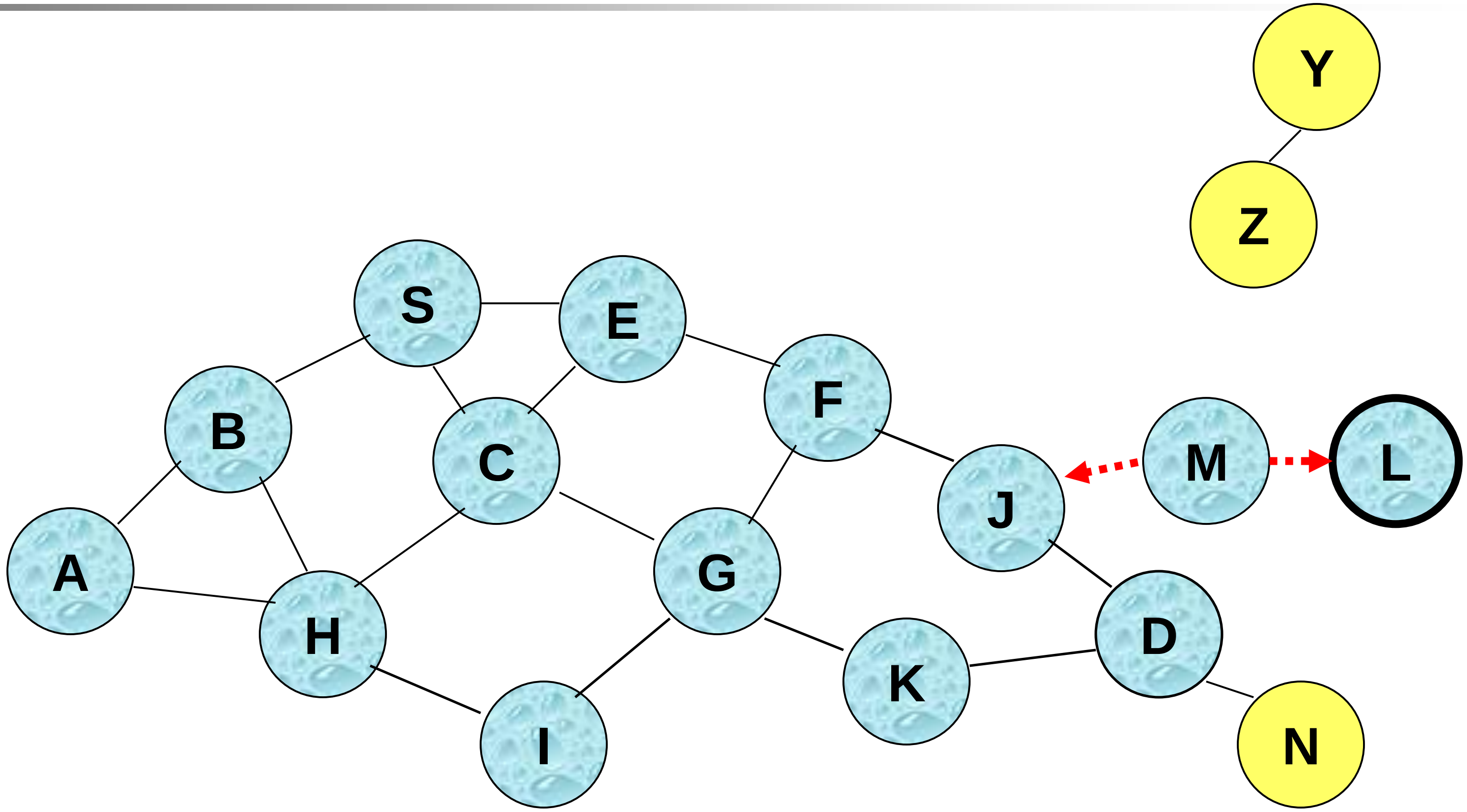
Flooding for Data Delivery



- 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

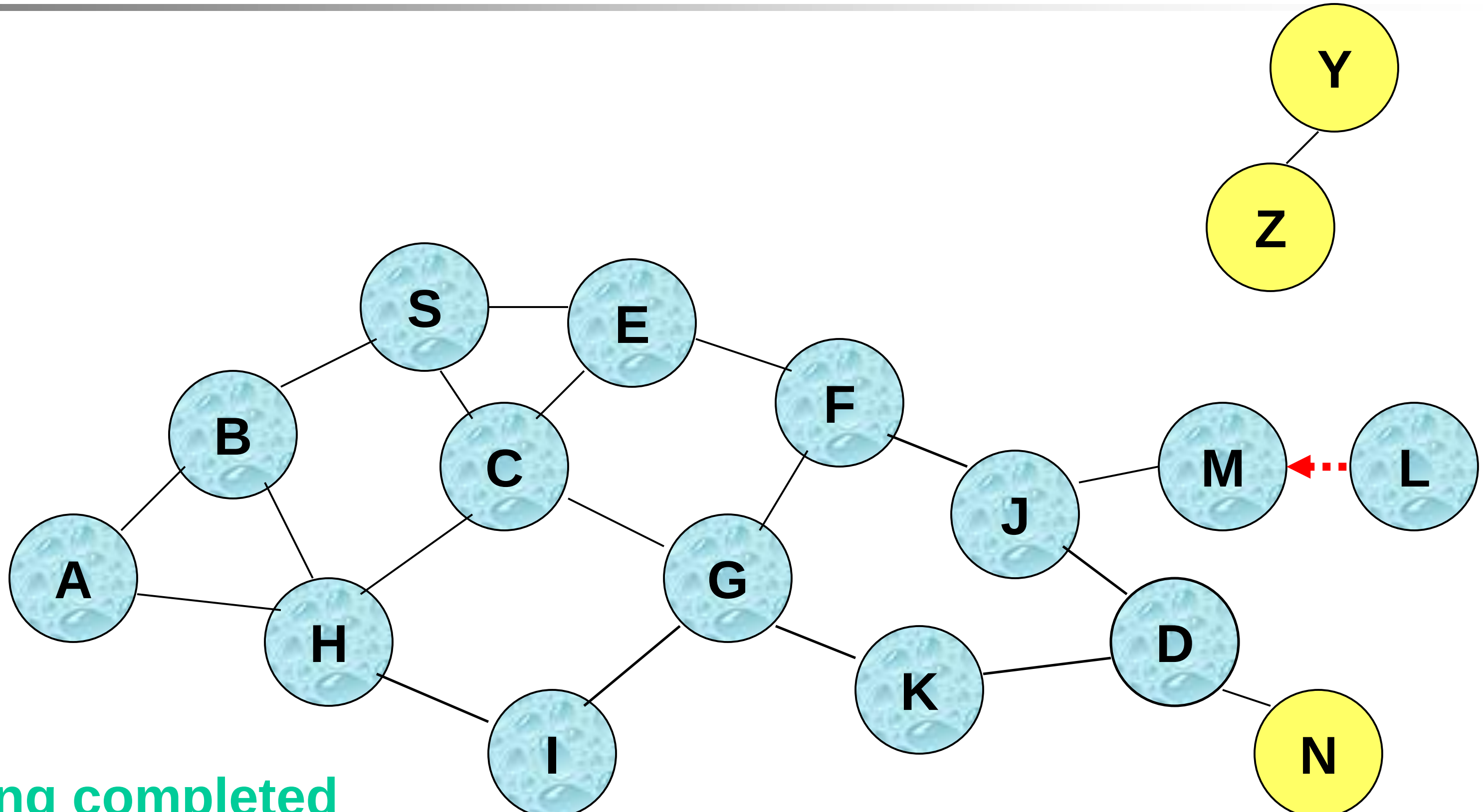
  => Packet P may not be delivered to node D at all, despite the use of flooding

Flooding for Data Delivery



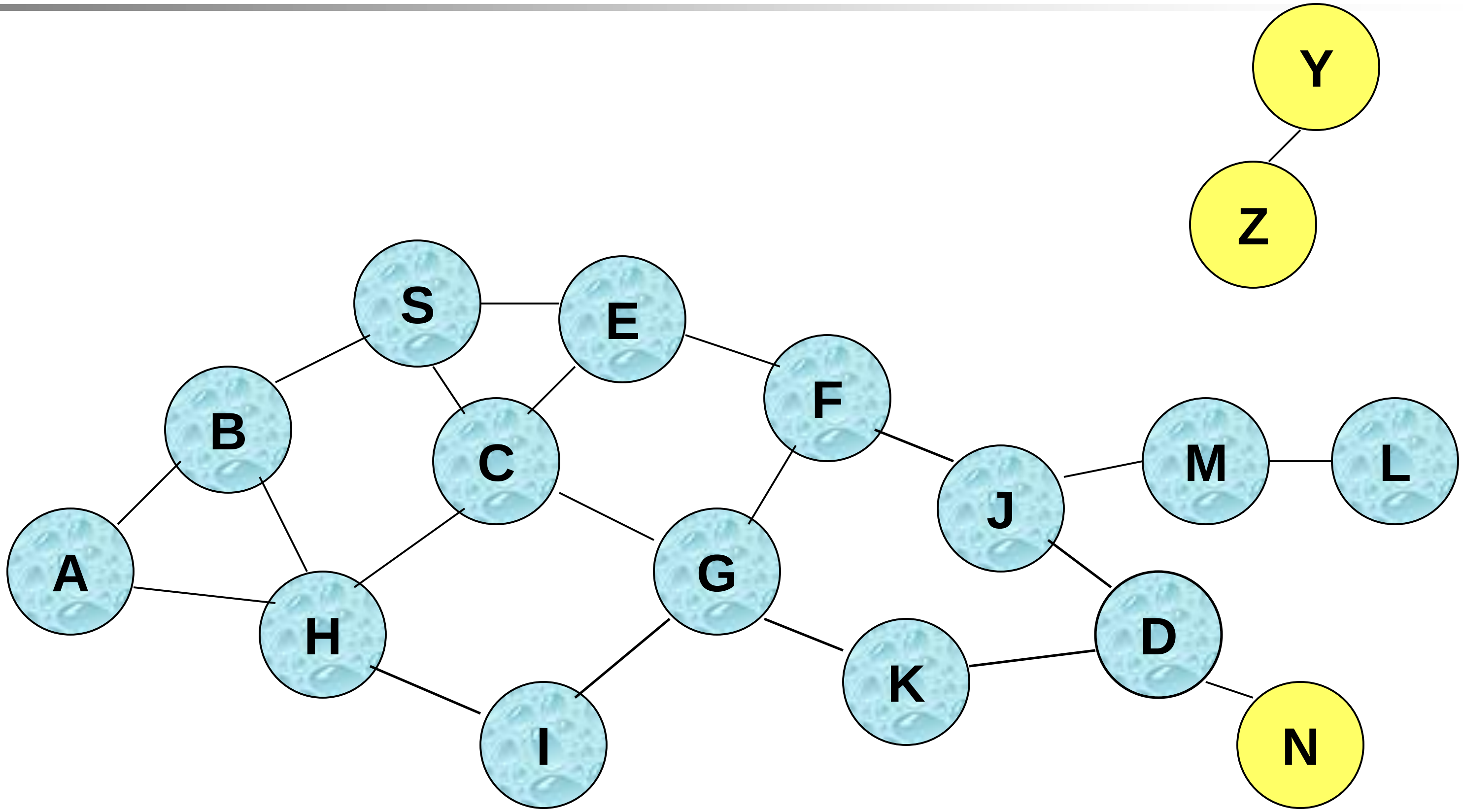
- **Node D does not forward packet P, because node D is the intended destination of packet P**

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 when infrequent communication is sufficient
  - Route setup / maintenance not worth it
  - Especially, when changing topology / mobility
- Potentially higher robustness to path failure
  - Because of multi-path redundancy

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
  - Reliable broadcast is difficult
  - Hidden terminal because no channel reservation

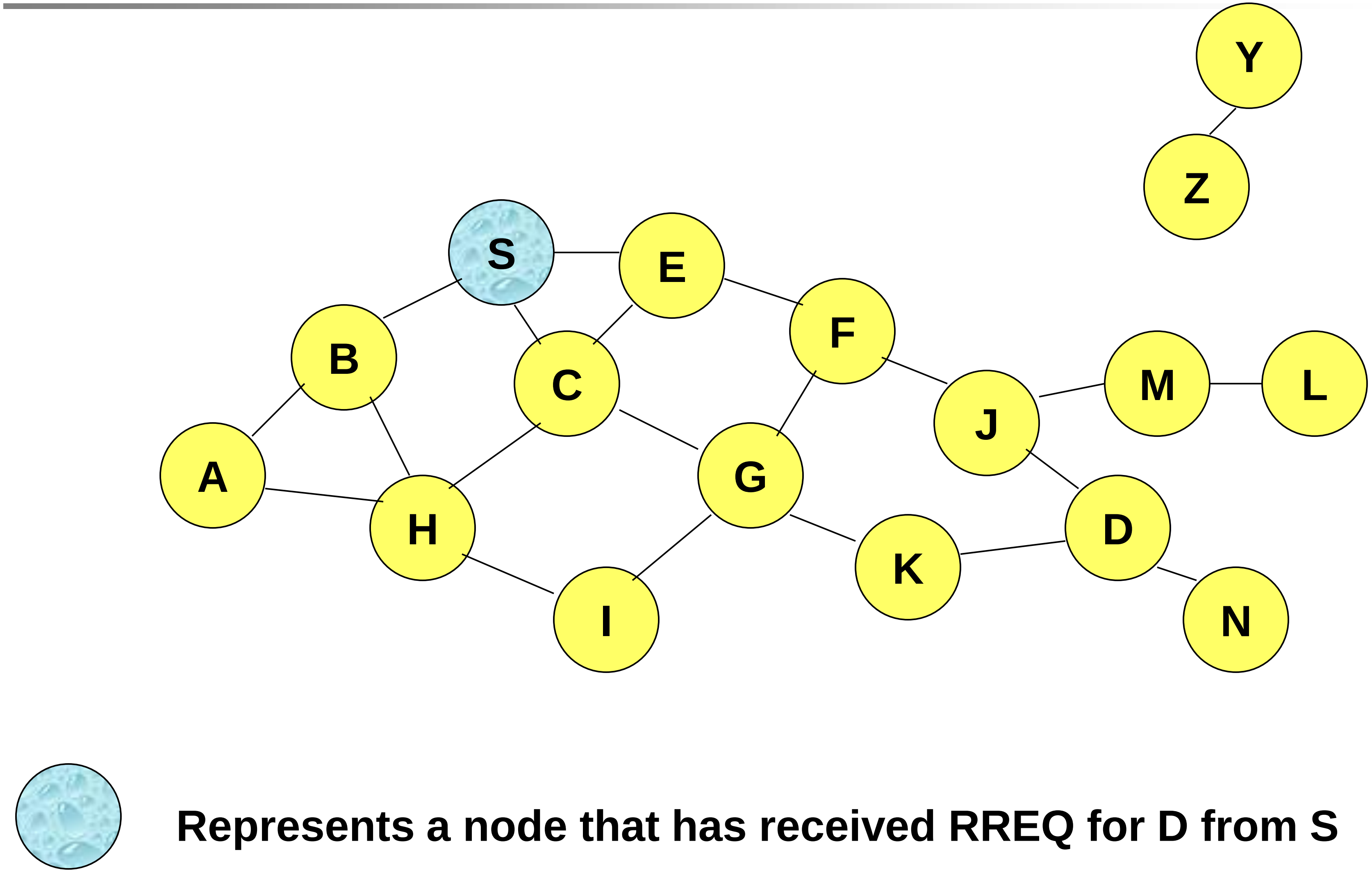
Flooding of Control Packets

- 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) [Johnson96]

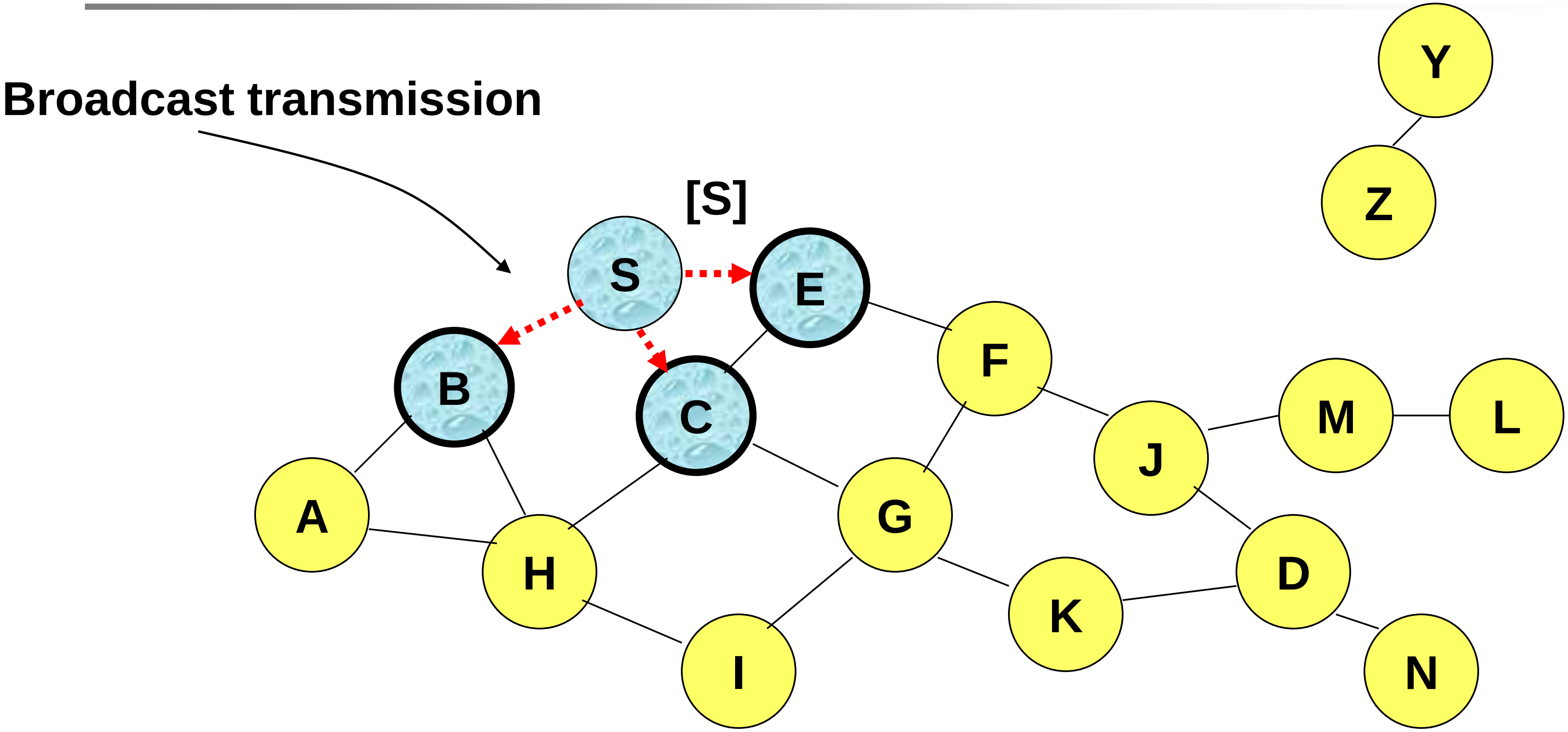
- When node S wants to send a packet to node D, but does not know a route to D, node S initiates a route discovery
- Source node S floods Route Request (RREQ)
- Each node appends own identifier when forwarding RREQ

Route Discovery in DSR



Represents a node that has received RREQ for D from S

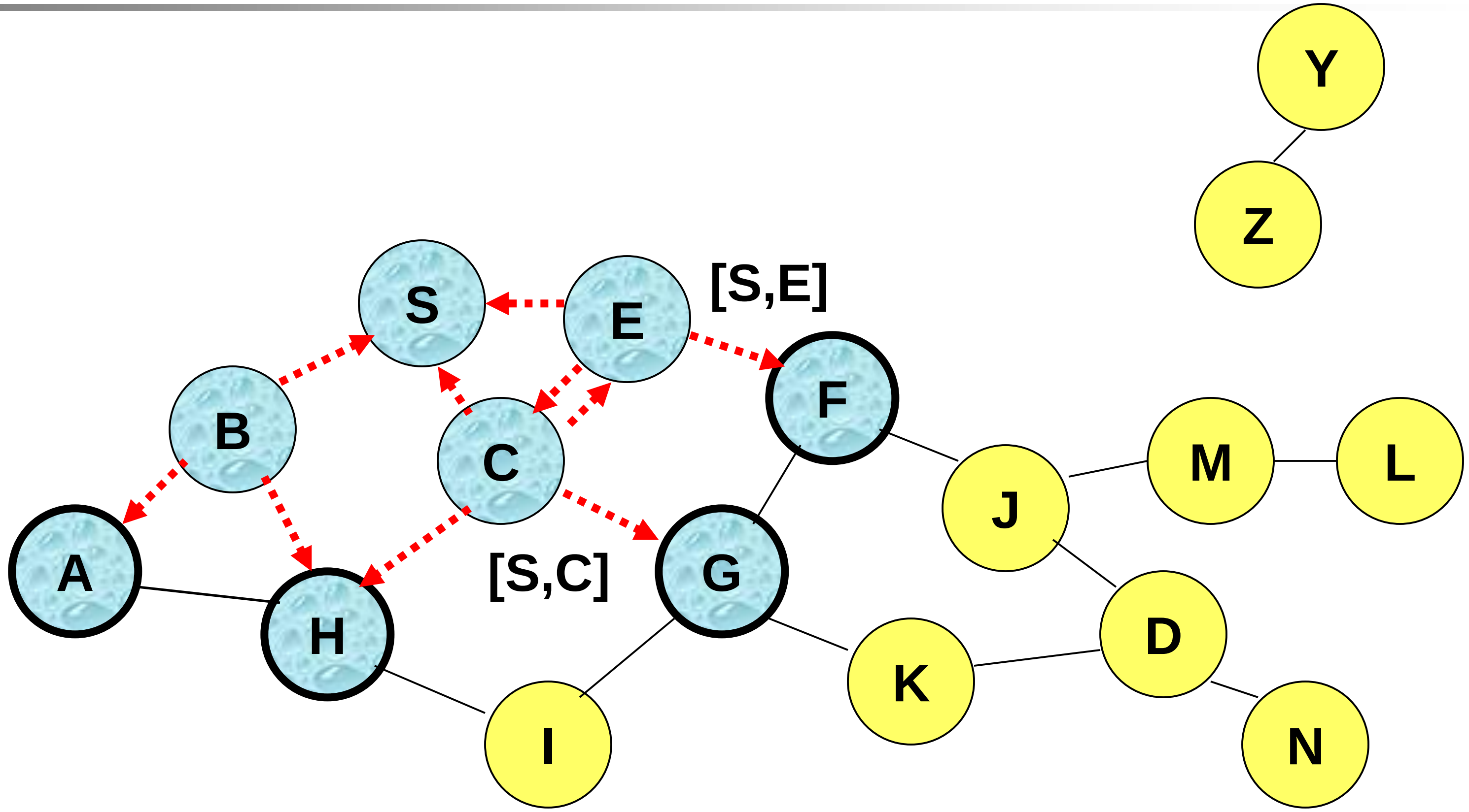
Route Discovery in DSR



Represents transmission of RREQ

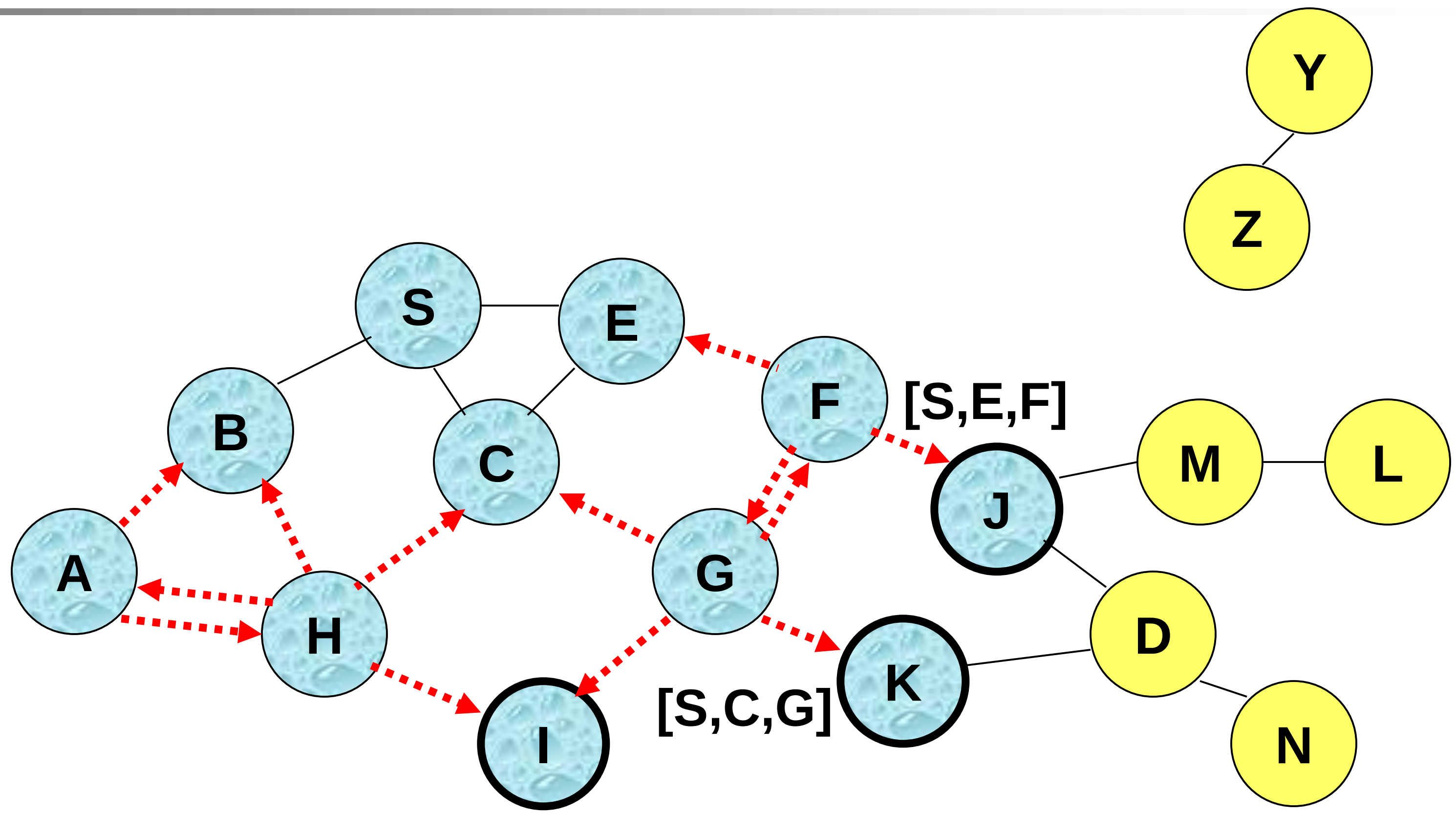
[X,Y] Represents list of identifiers appended to RREQ

Route Discovery in DSR



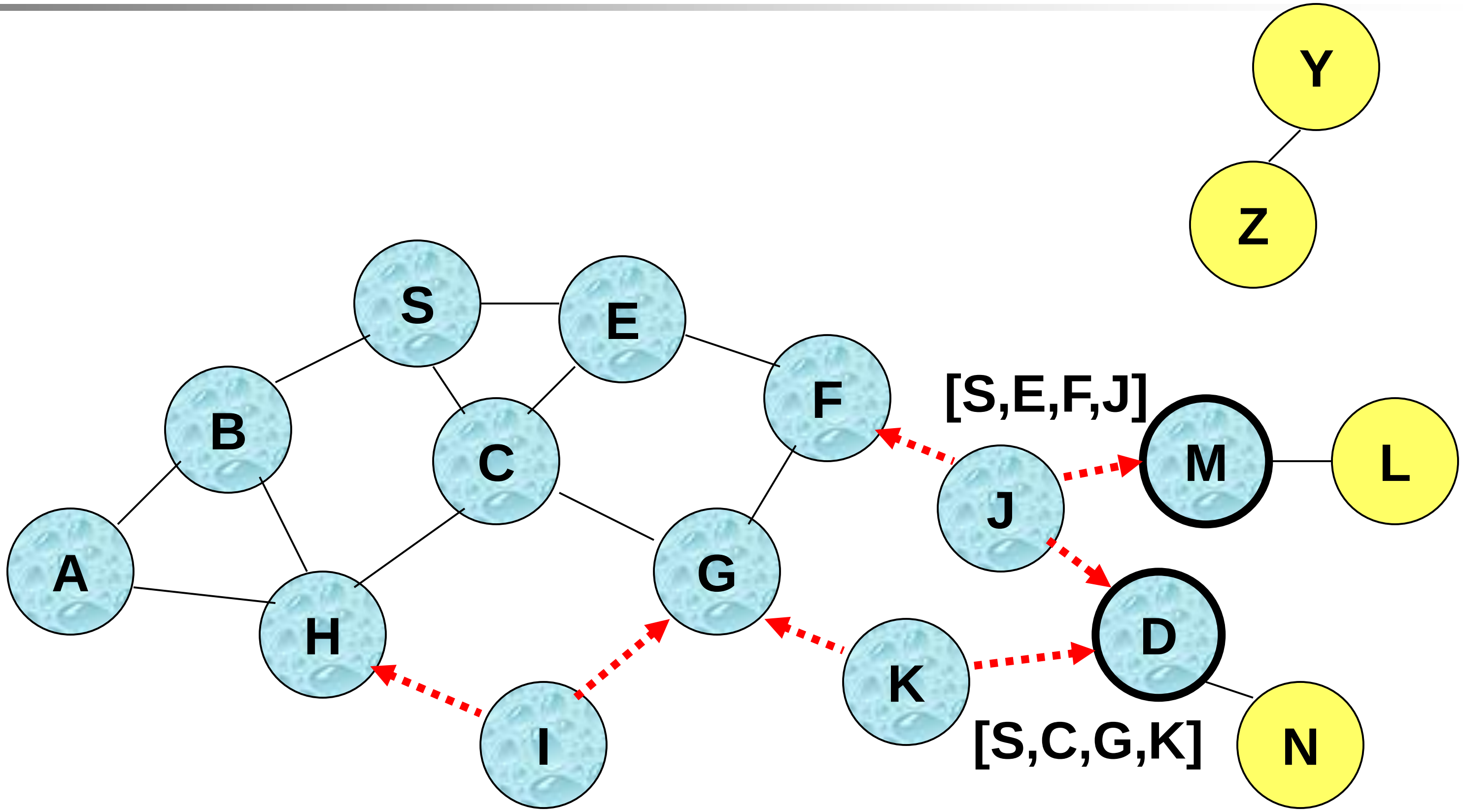
- **Node H receives packet RREQ from two neighbors: potential for collision**

Route Discovery in DSR



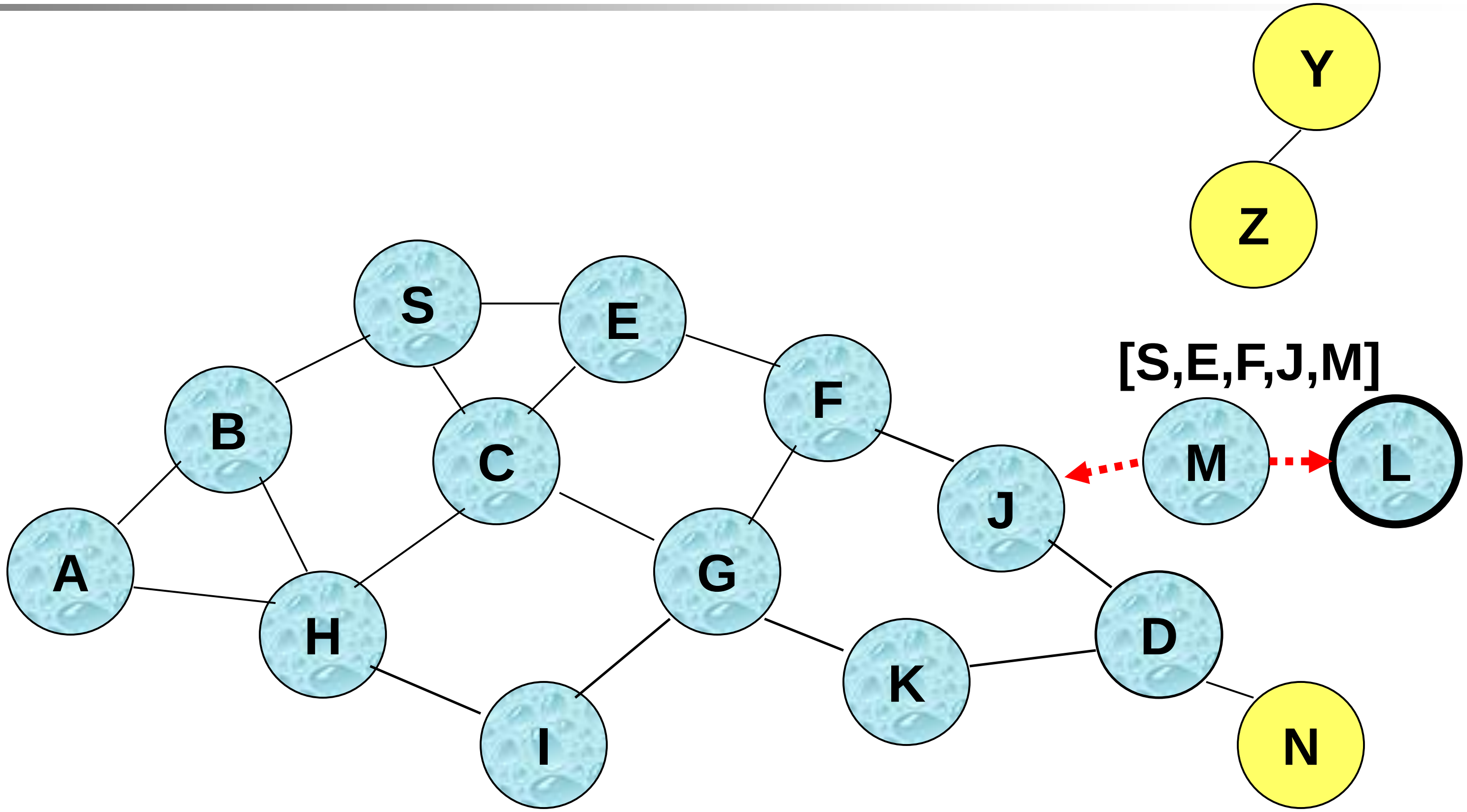
- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

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

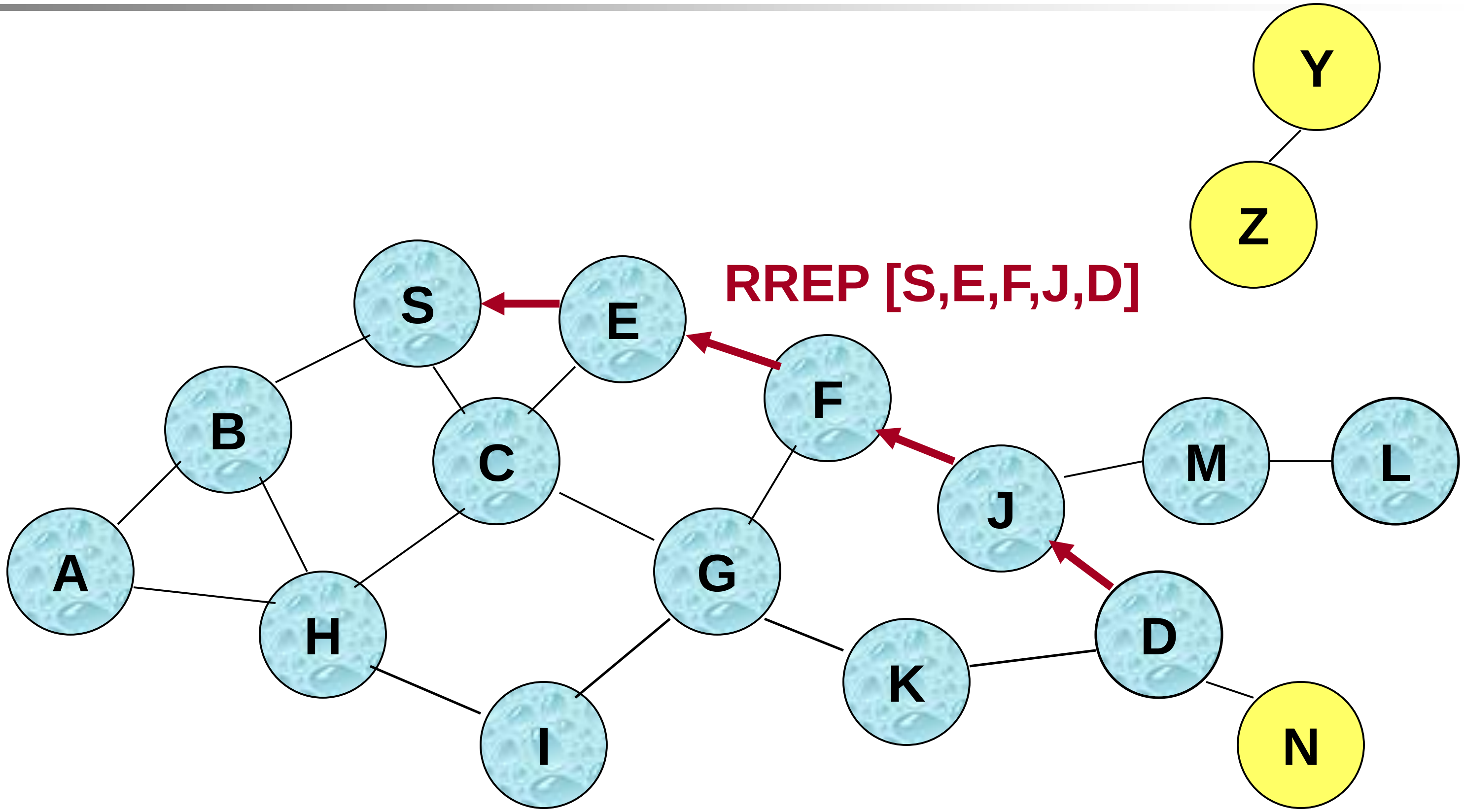


- **Node D does not forward RREQ, because node D is the intended target of the route discovery**

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



Represents RREP control message

Route Reply in DSR

- Route Reply can be sent by reversing route in RREQ
  - But, links need to be bi-directional

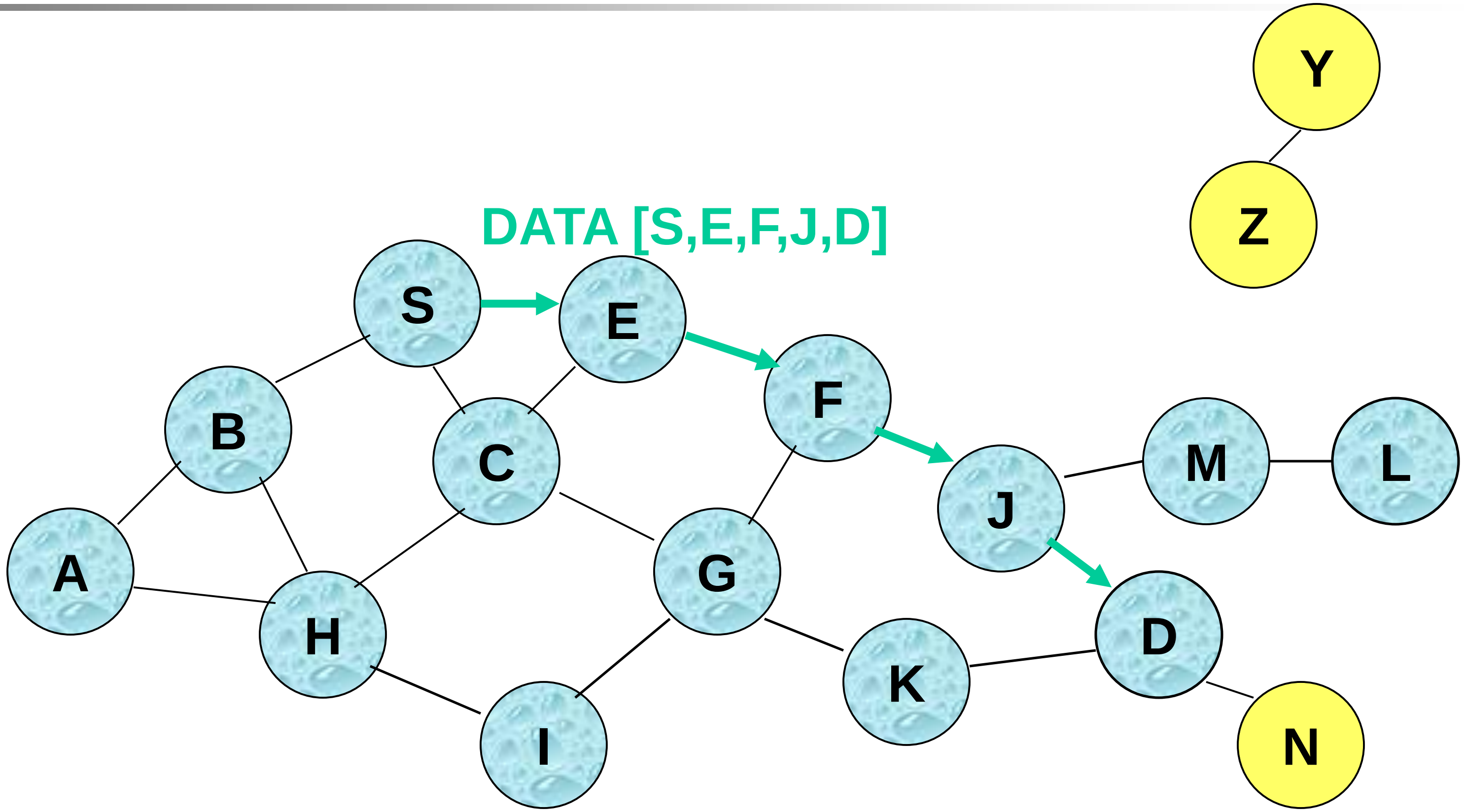
- If unidirectional (asymmetric) links are allowed
  - then RREP may need a route discovery for S from node D

- 802.11 links always bi-directional (since Ack is used)

Data Delivery in 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

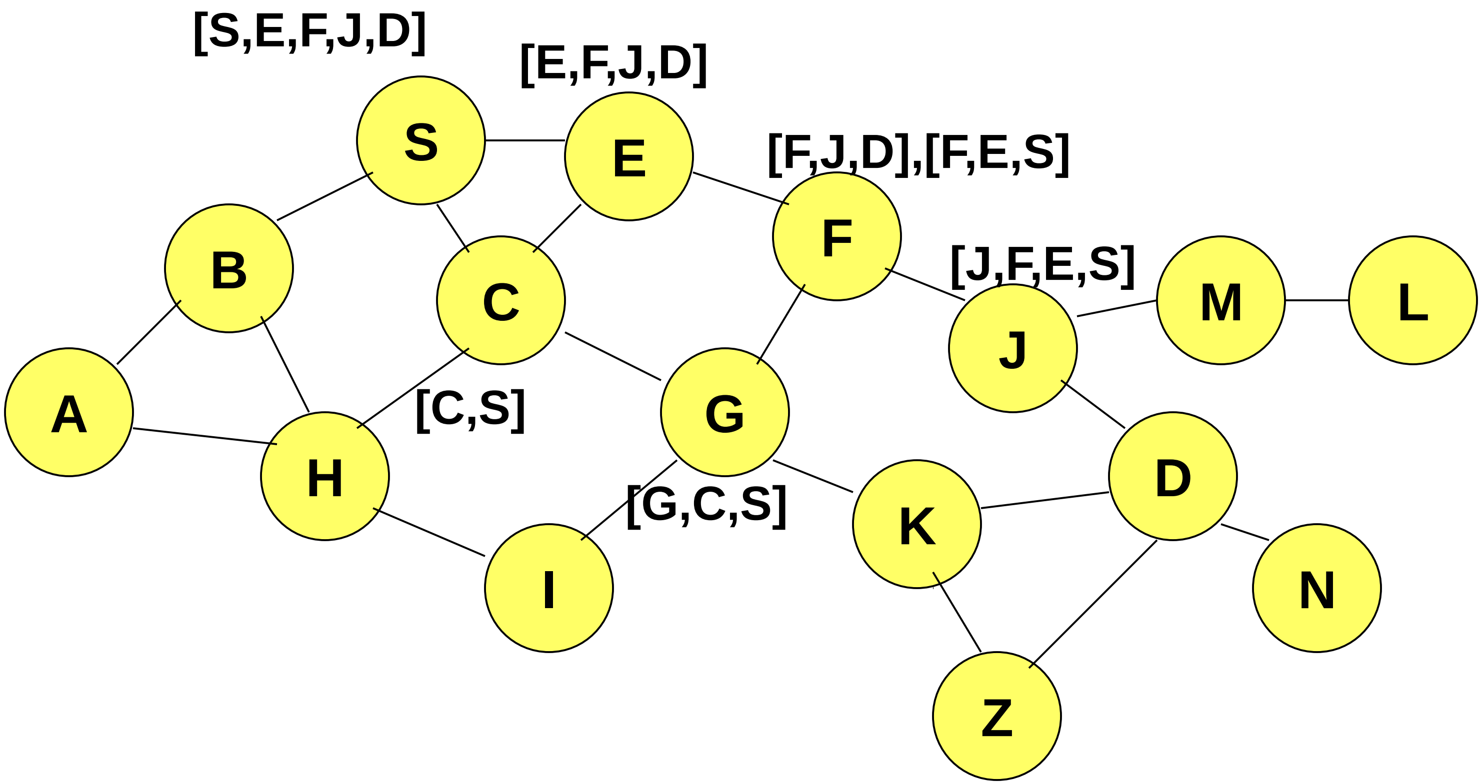
When to Perform a Route Discovery

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

DSR Optimization: Route Caching

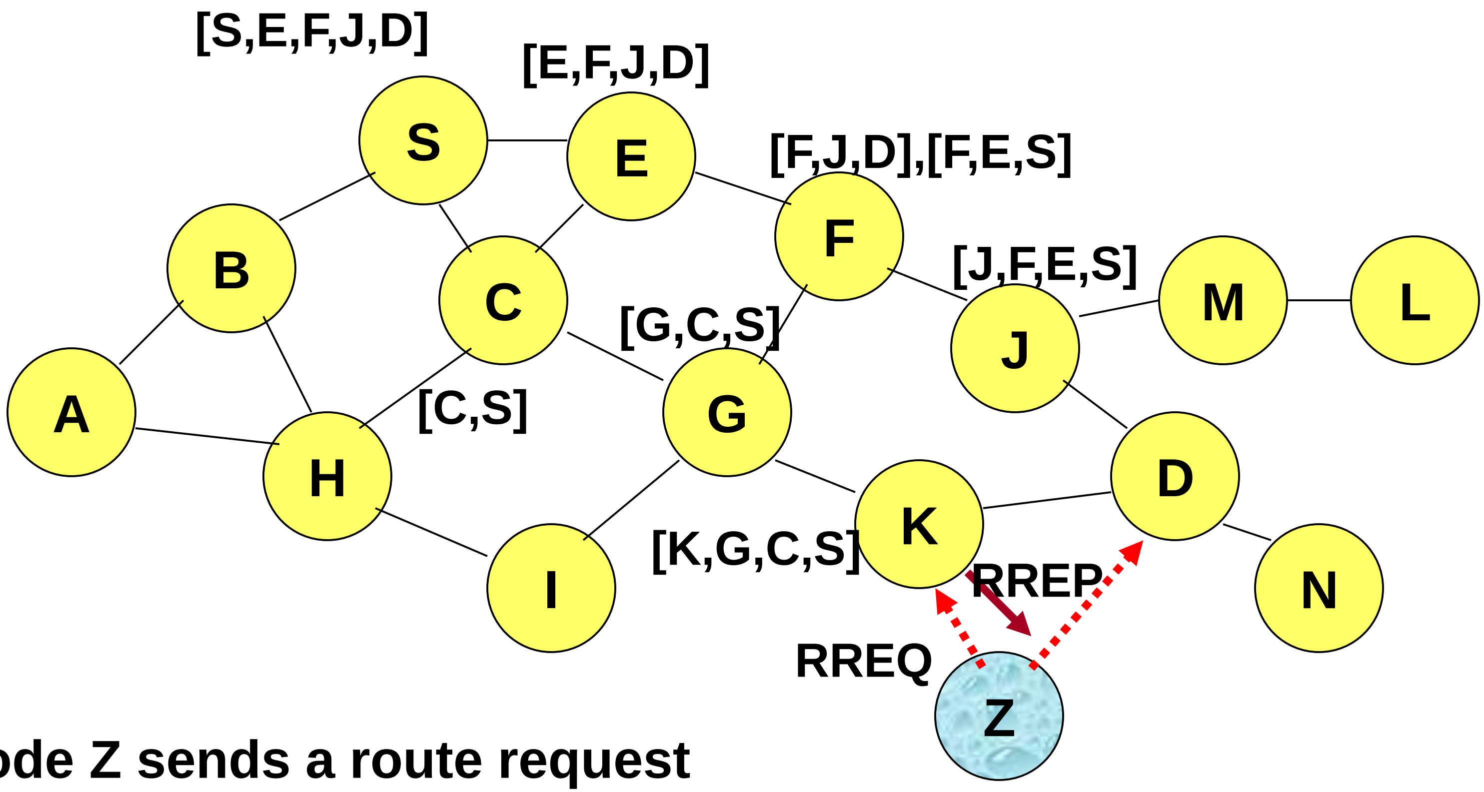
- Caches a new route it learns by *any means*
- When node S finds route [S,E,F,J,D] to node D, node S also learns route [S,E,F] to node F
- When node K receives Route Request [S,C,G] destined for node, node K learns route [K,G,C,S] to node S
- When node F forwards Route Reply RREP [S,E,F,J,D], node F learns route [F,J,D] to node D
- When node E forwards Data [S,E,F,J,D] it learns
- Learn by overhearing Data packets

Use of Route Caching



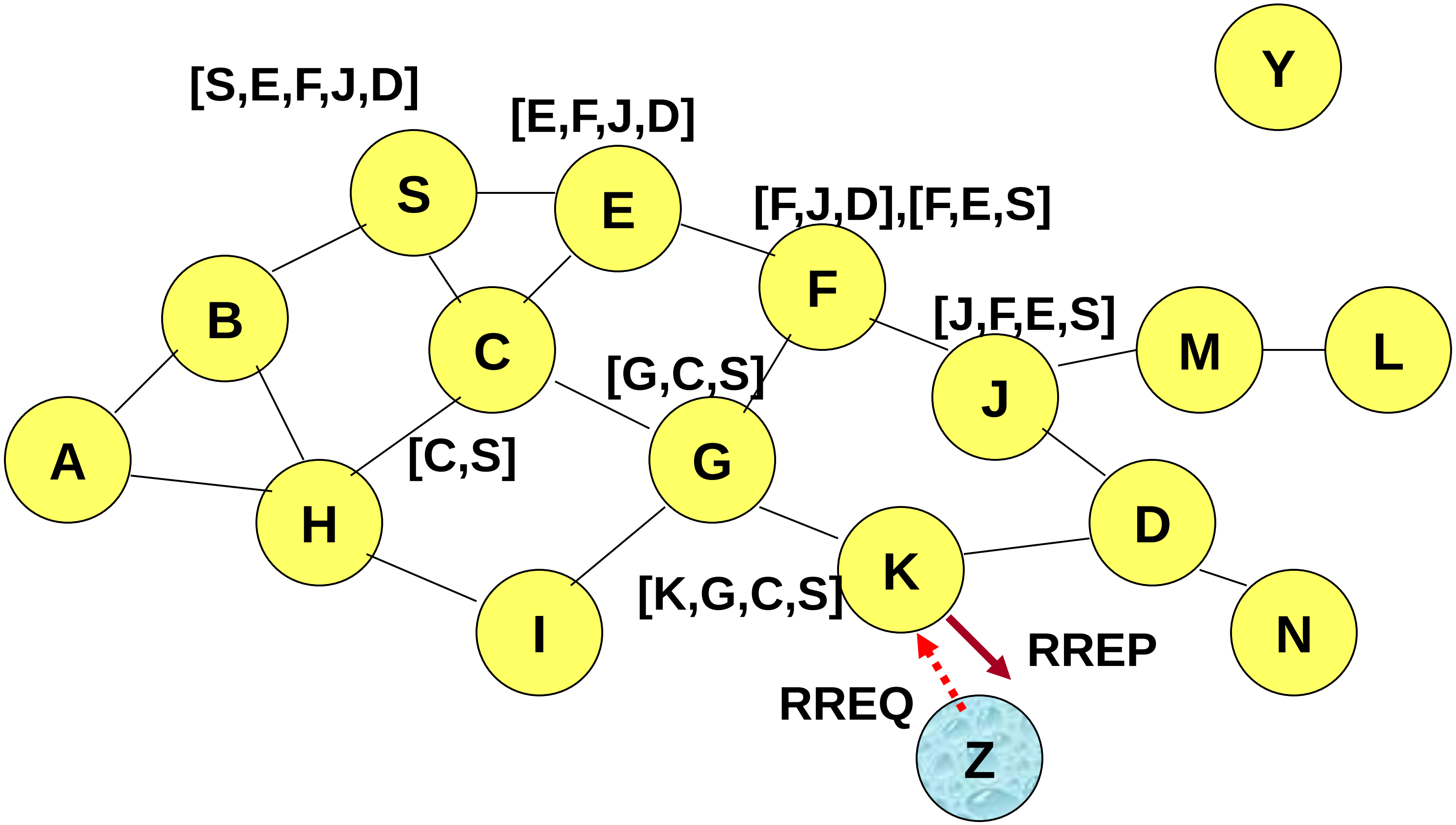
[P,Q,R]   Represents cached route at a node
(DSR maintains the cached routes in a tree format)

Use of Route Caching: Can Speed up Route Discovery



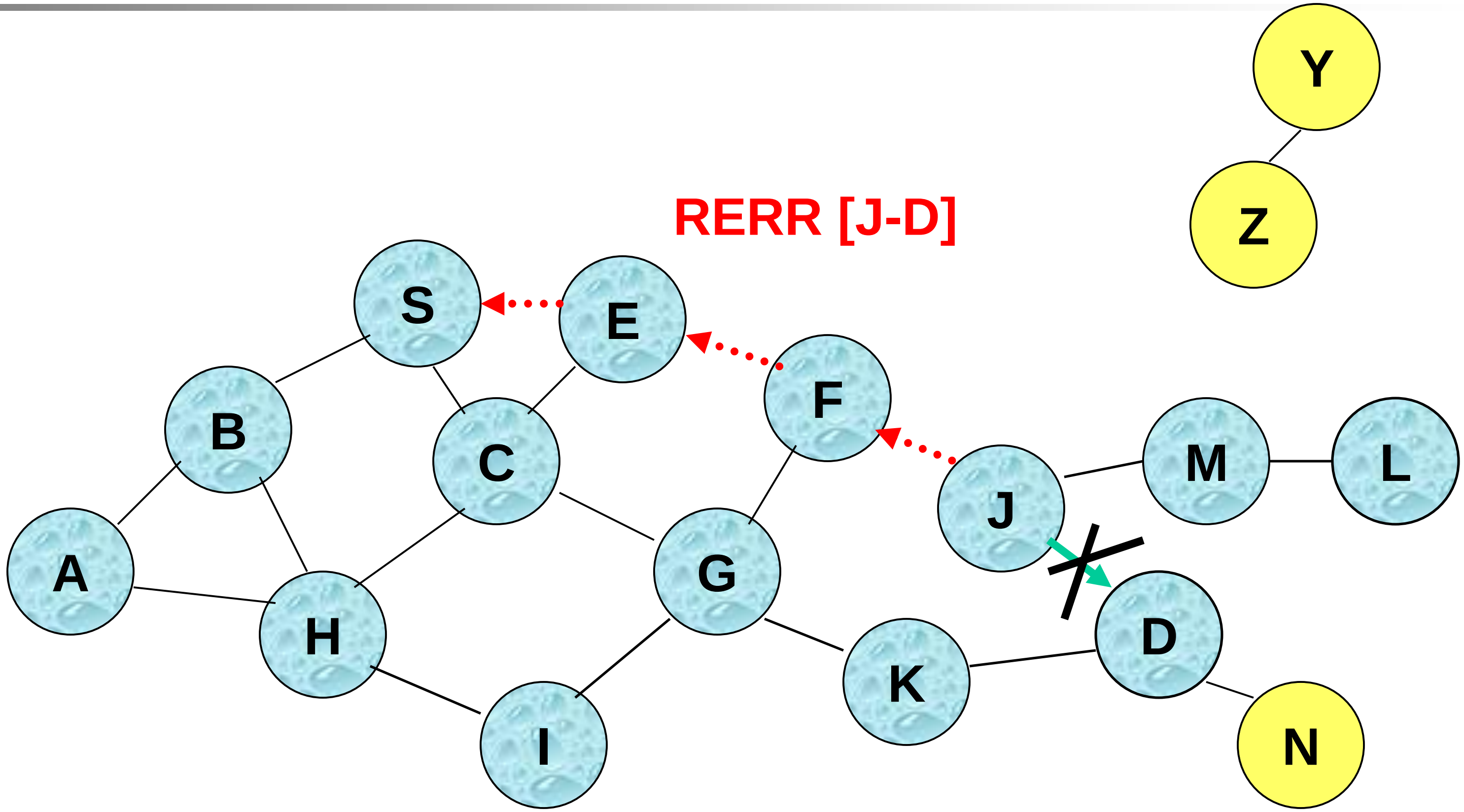
When node Z sends a route request for node C, node K sends back a route reply [Z,K,G,C] to node Z using a locally cached route

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

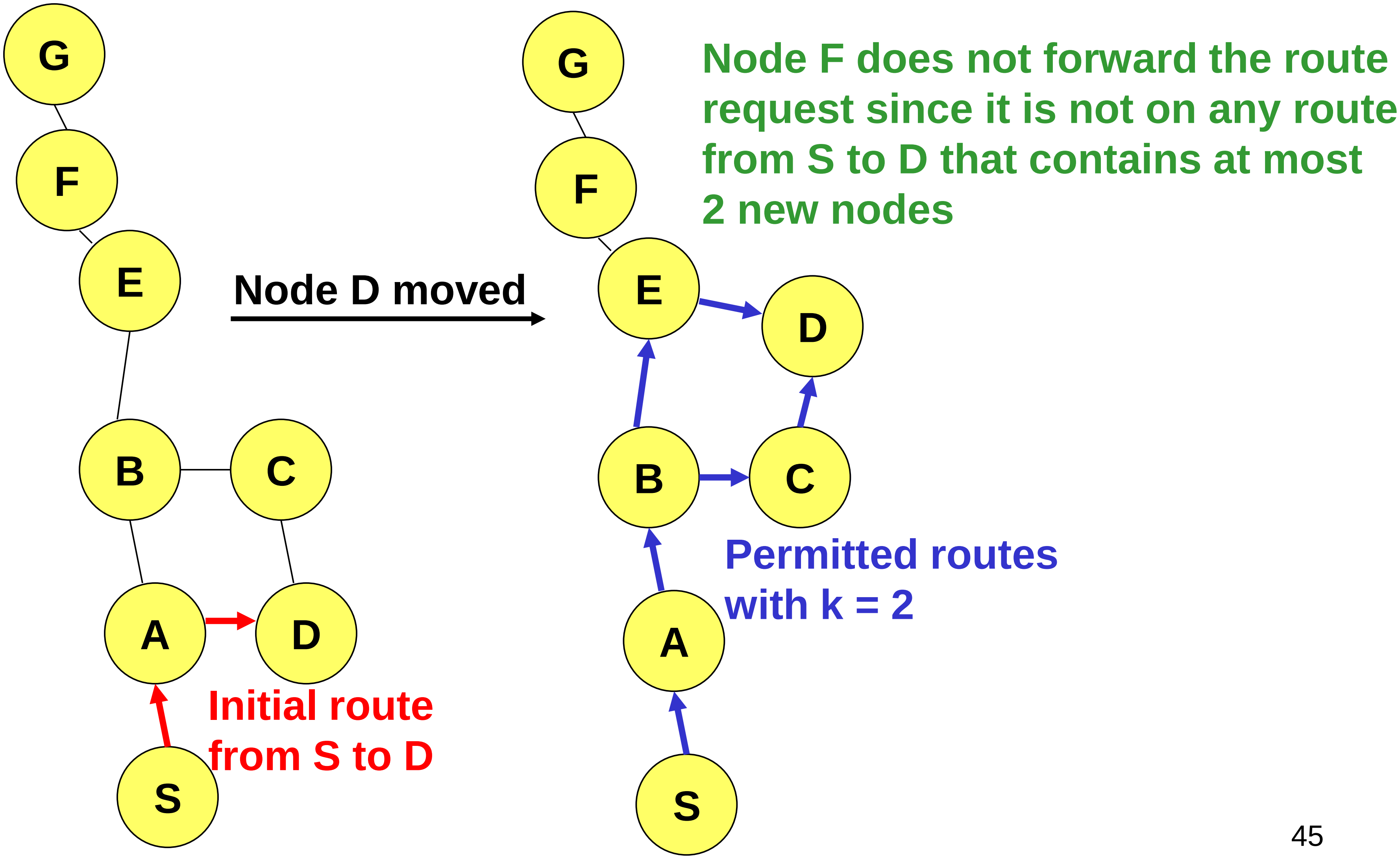
Query Localization

- Path locality heuristic: Look for a new path that contains at most *k* nodes that were not present in the previously known route

- Old route is piggybacked on a Route Request

- Route Request is forwarded only if the accumulated route in the Route Request contains at most *k* new nodes that were absent in the old route
  - this limits propagation of the route request

Query Localization: Example



Dynamic Source Routing: Advantages

- Routes maintained reactively
  - reduces overhead of maintenance
- Route caching can reduce route discovery overhead
- Discovery of multiple routes at D

Dynamic Source Routing: Disadvantages

- Packet header size grows with route length
- Flood of route requests may potentially reach all nodes
- 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

Dynamic Source Routing: 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.
- For some proposals for cache invalidation, see [Hu00Mobicom]
  - Static timeouts
  - Adaptive timeouts based on link stability

Distance Vector Routing

Ad Hoc On-Demand Distance Vector Routing (AODV) [Perkins99Wmcsa]

- DSR includes source routes in packet headers
- Resulting large headers can degrade performance
  - particularly when data contents of a packet are small
- AODV attempts to improve on DSR
  - By maintaining routing tables at the nodes
  - Data packets do not contain long routes
- AODV also reactive

AODV

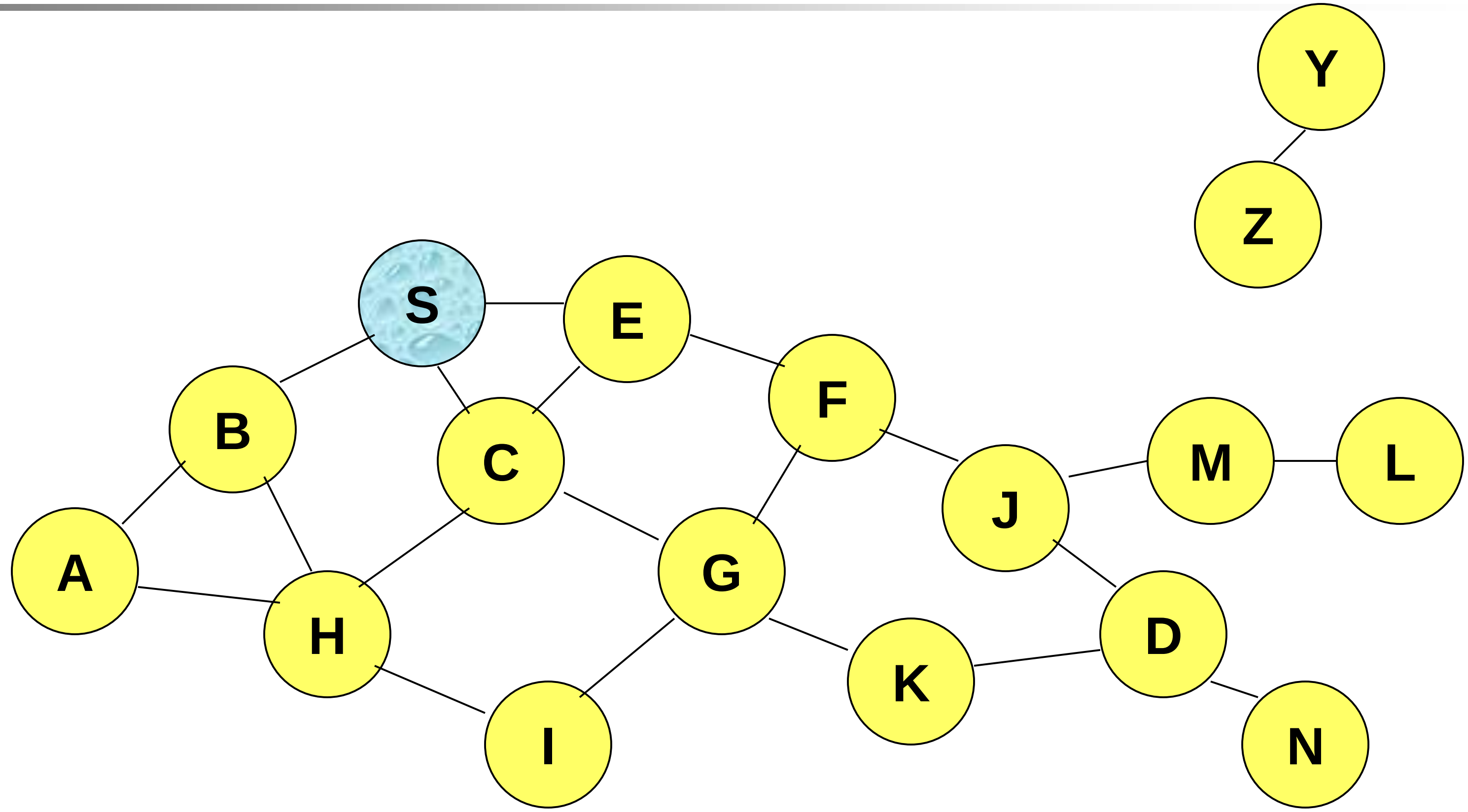
- Route Requests (RREQ) forwarded like DSR

- When intermediate node re-broadcasts RREQ
  - It sets up a reverse path pointing towards previous node
  - AODV assumes symmetric (bi-directional) links

- Destination replies by sending a Route Reply

- Intermediate nodes forward RREP up the reverse path
  - They also remember the downstream path in local cache

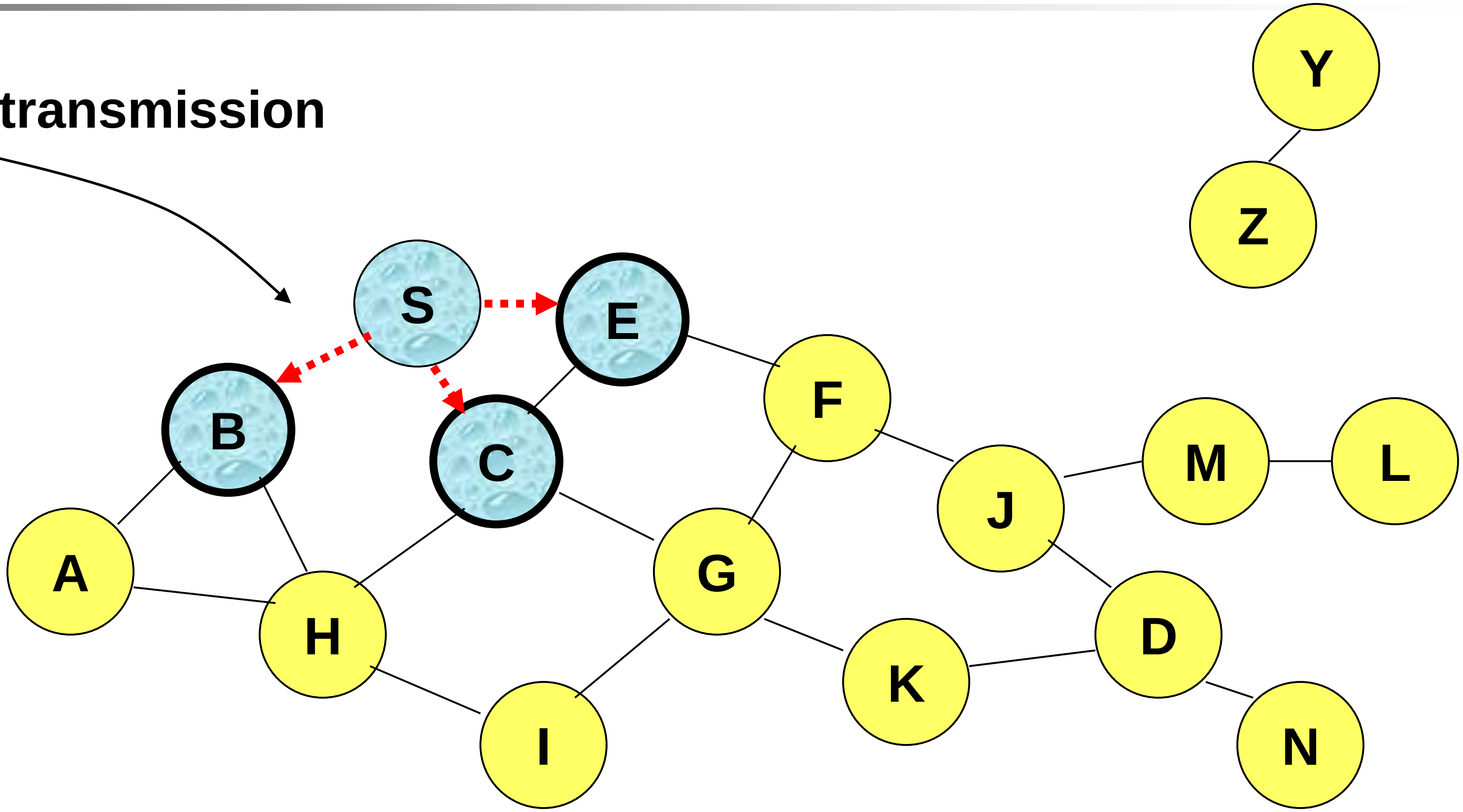
Route Requests in AODV



Represents a node that has received RREQ for D from S

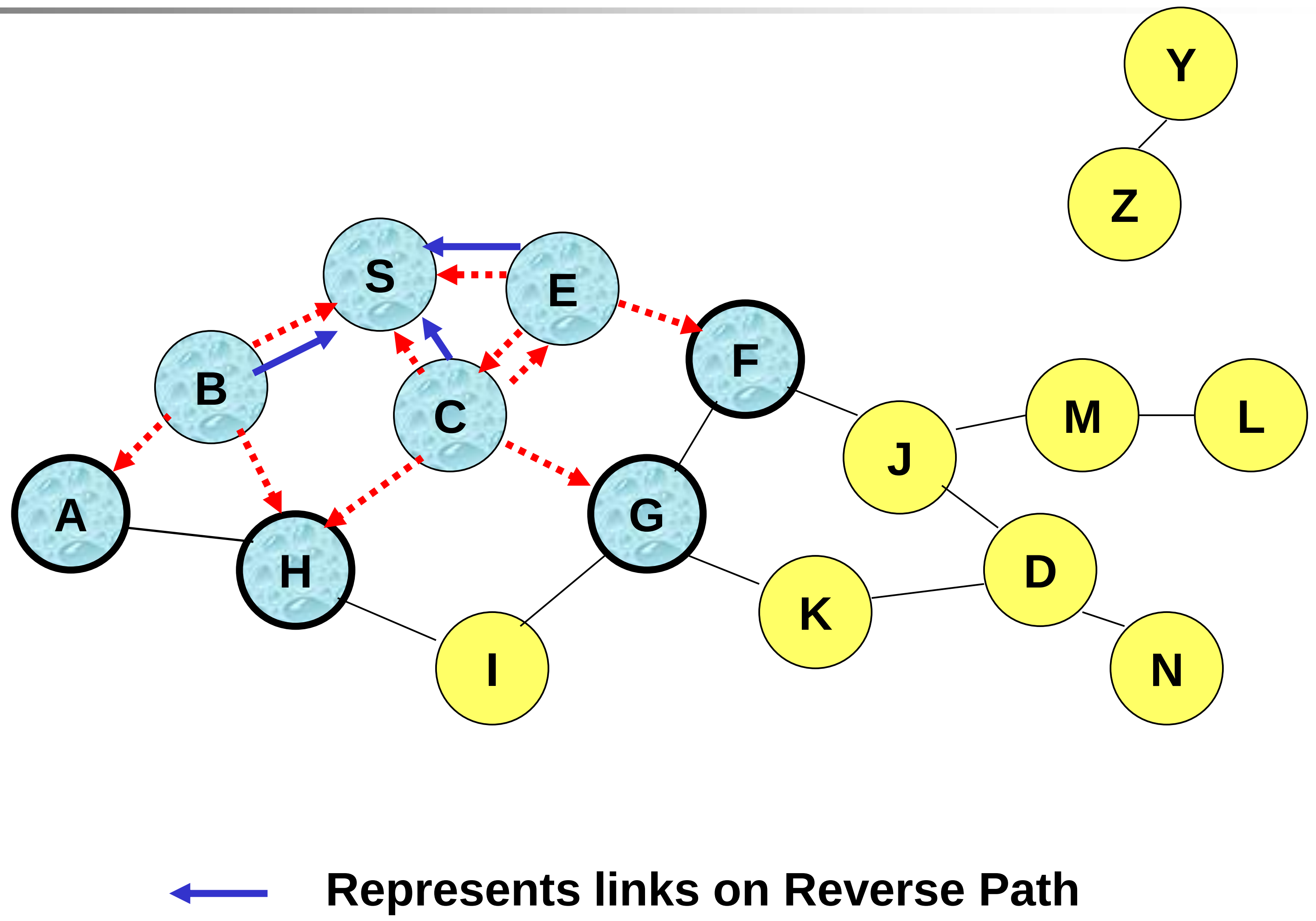
Route Requests in AODV

Broadcast transmission

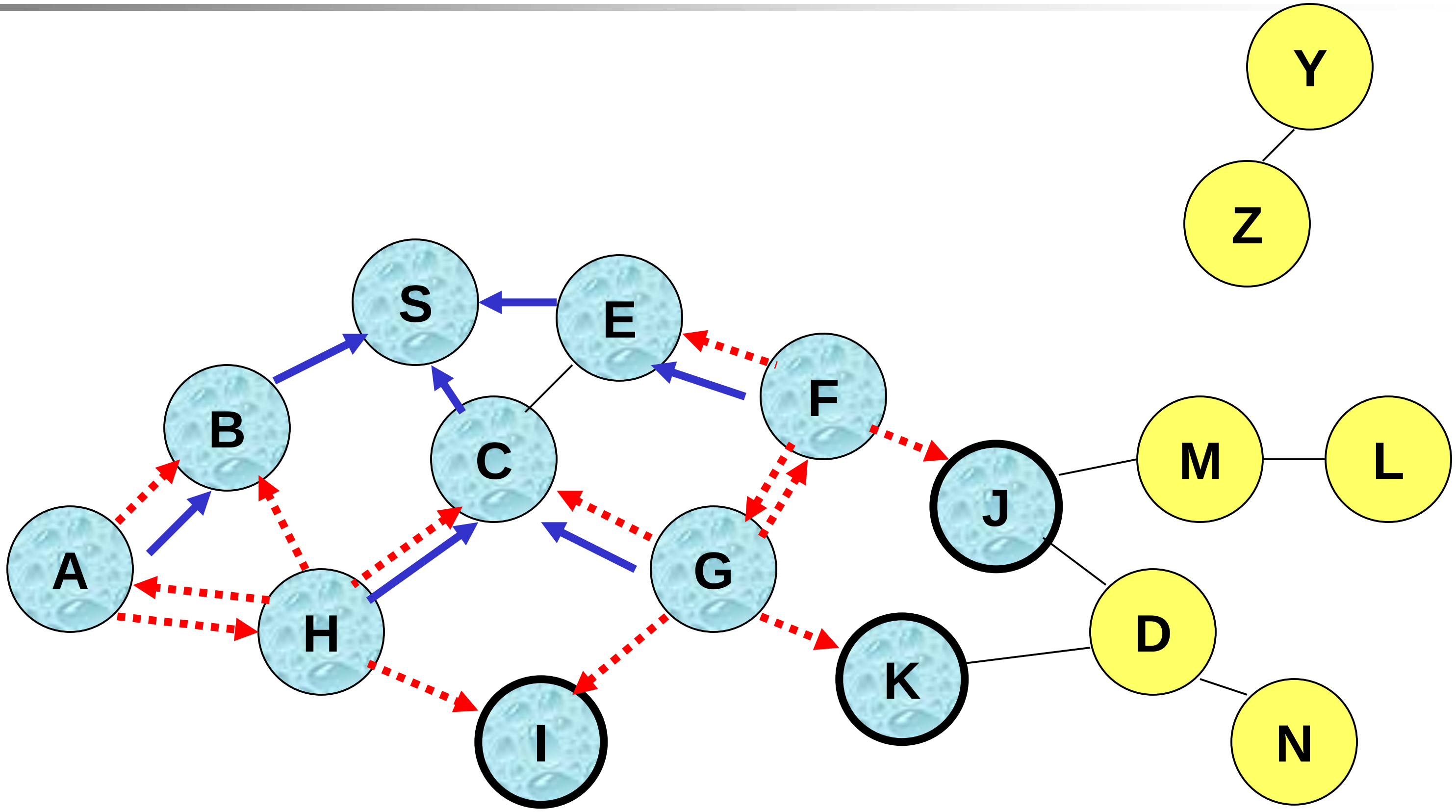


Represents transmission of RREQ

Route Requests in AODV

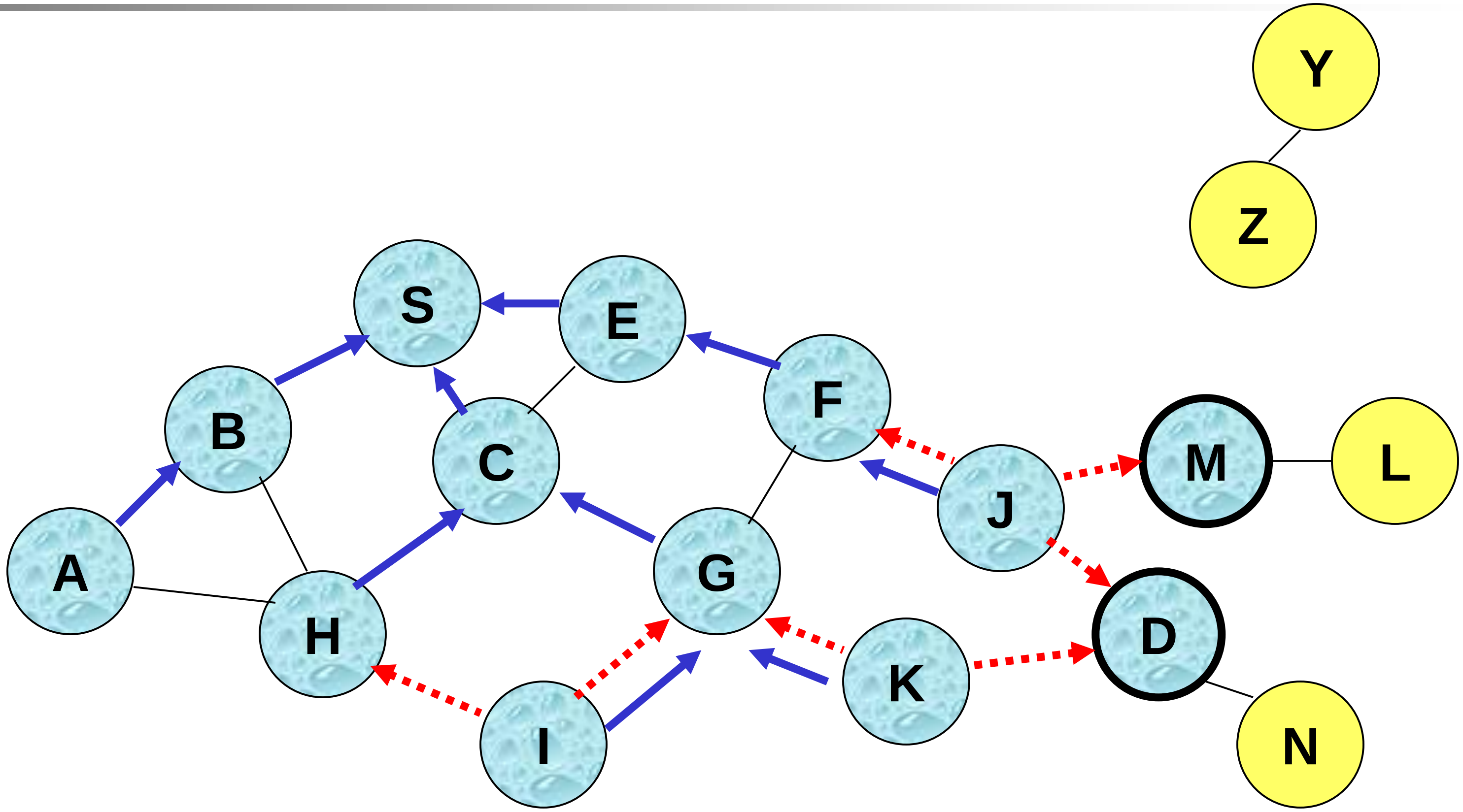


Reverse Path Setup in AODV

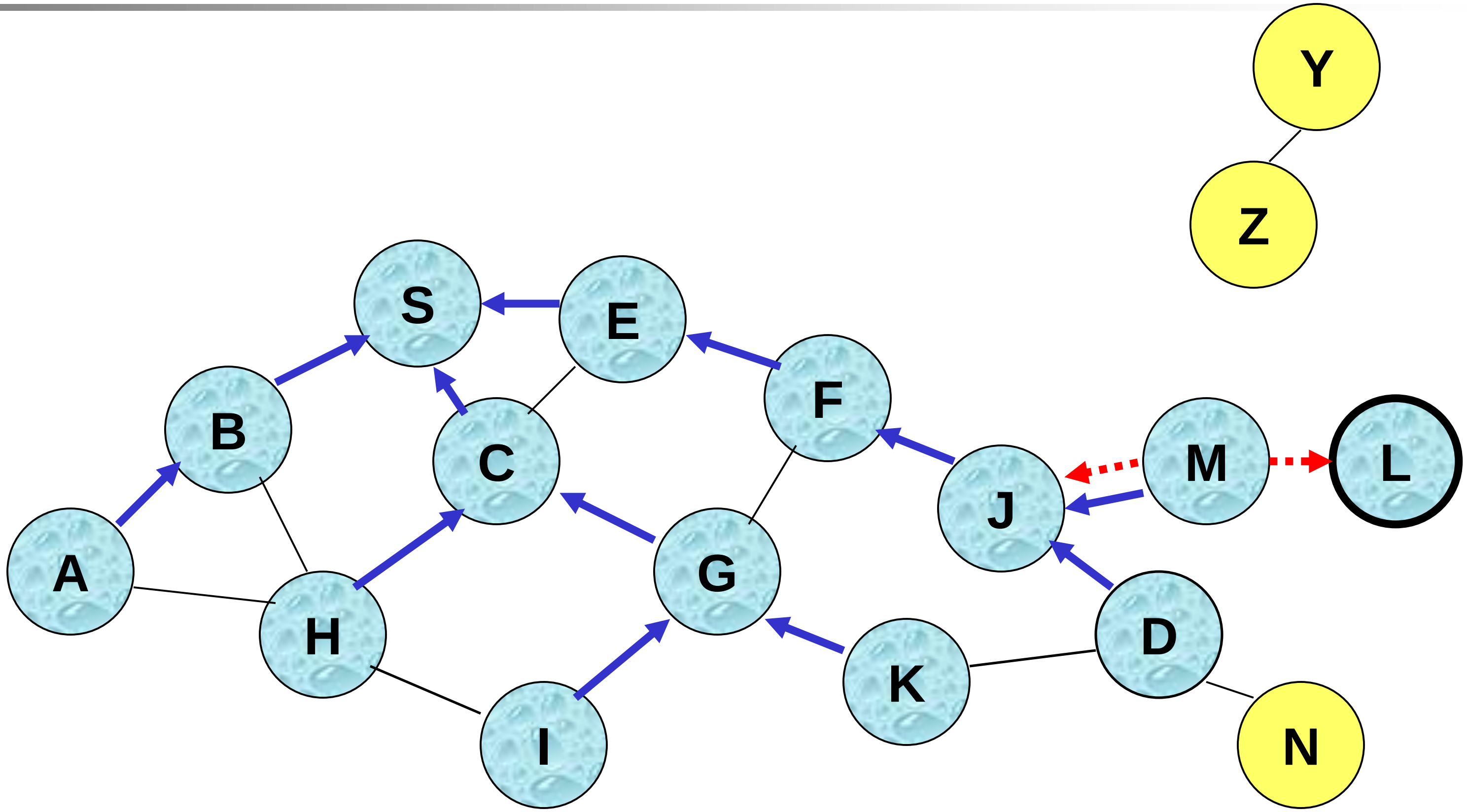


- **Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once**

Reverse Path Setup in AODV

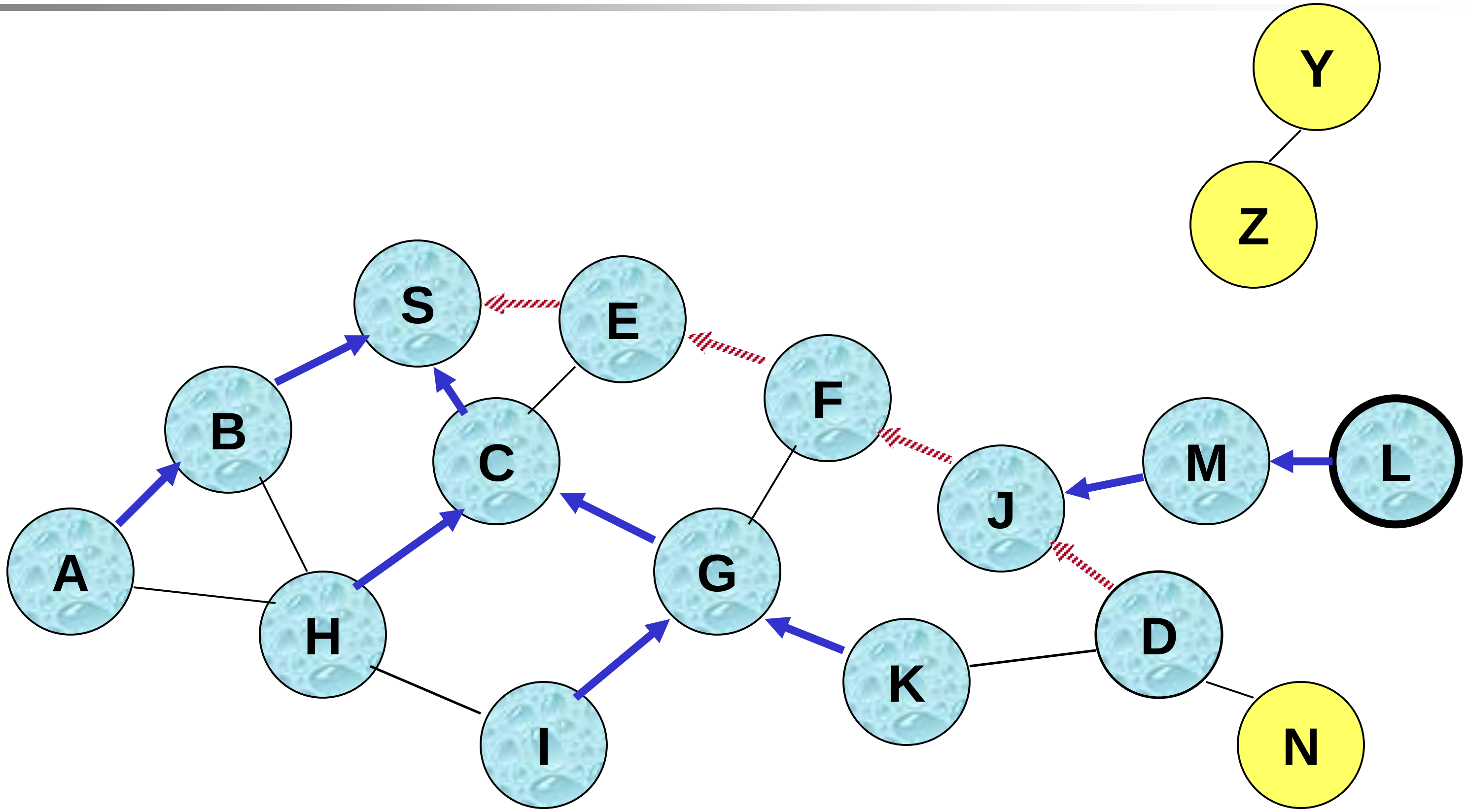


Reverse Path Setup in AODV



- **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

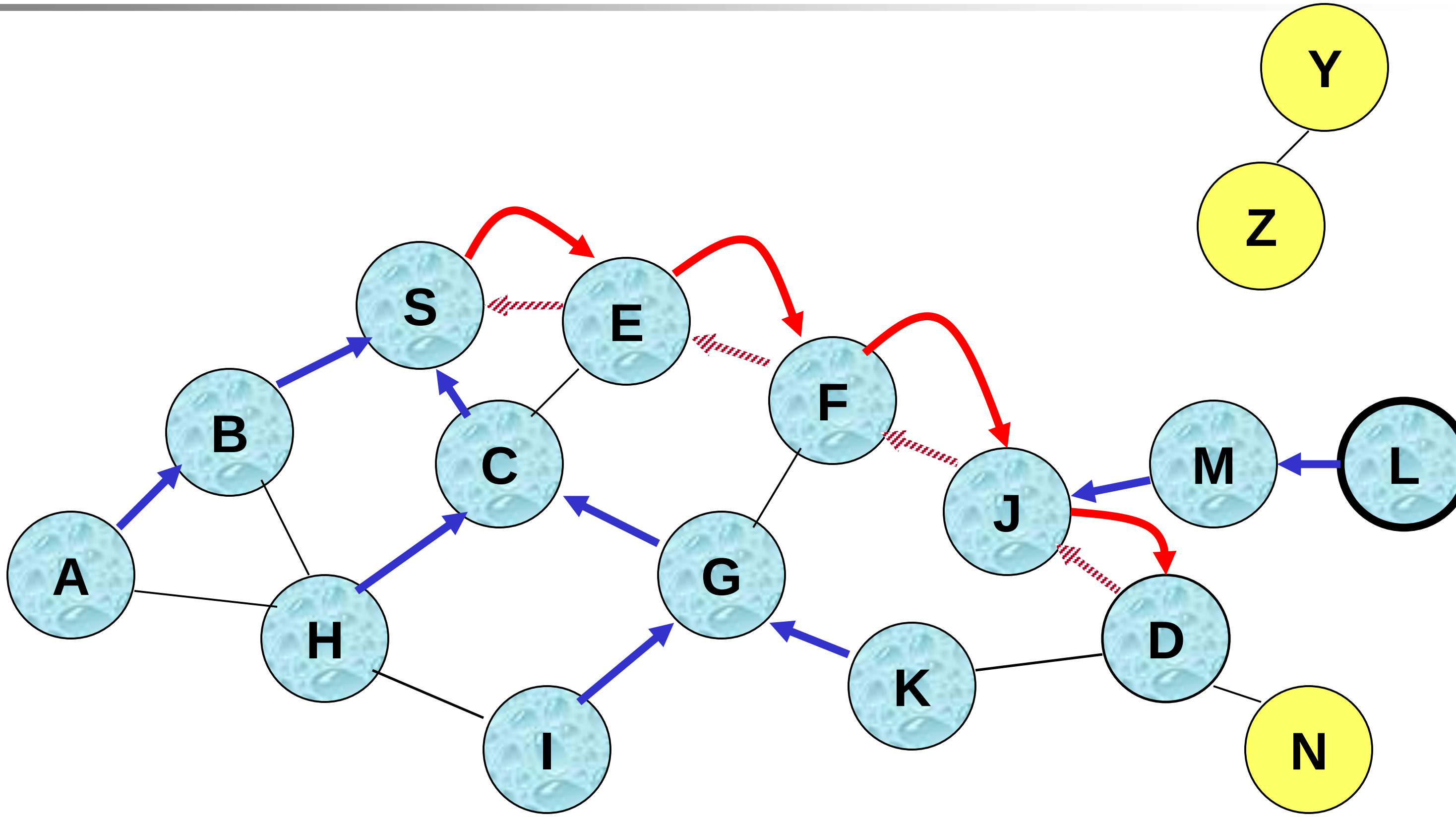
Route Reply in AODV

- 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

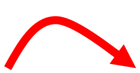
- To determine whether the path known to an intermediate node is more recent, *destination sequence numbers* are used

- 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

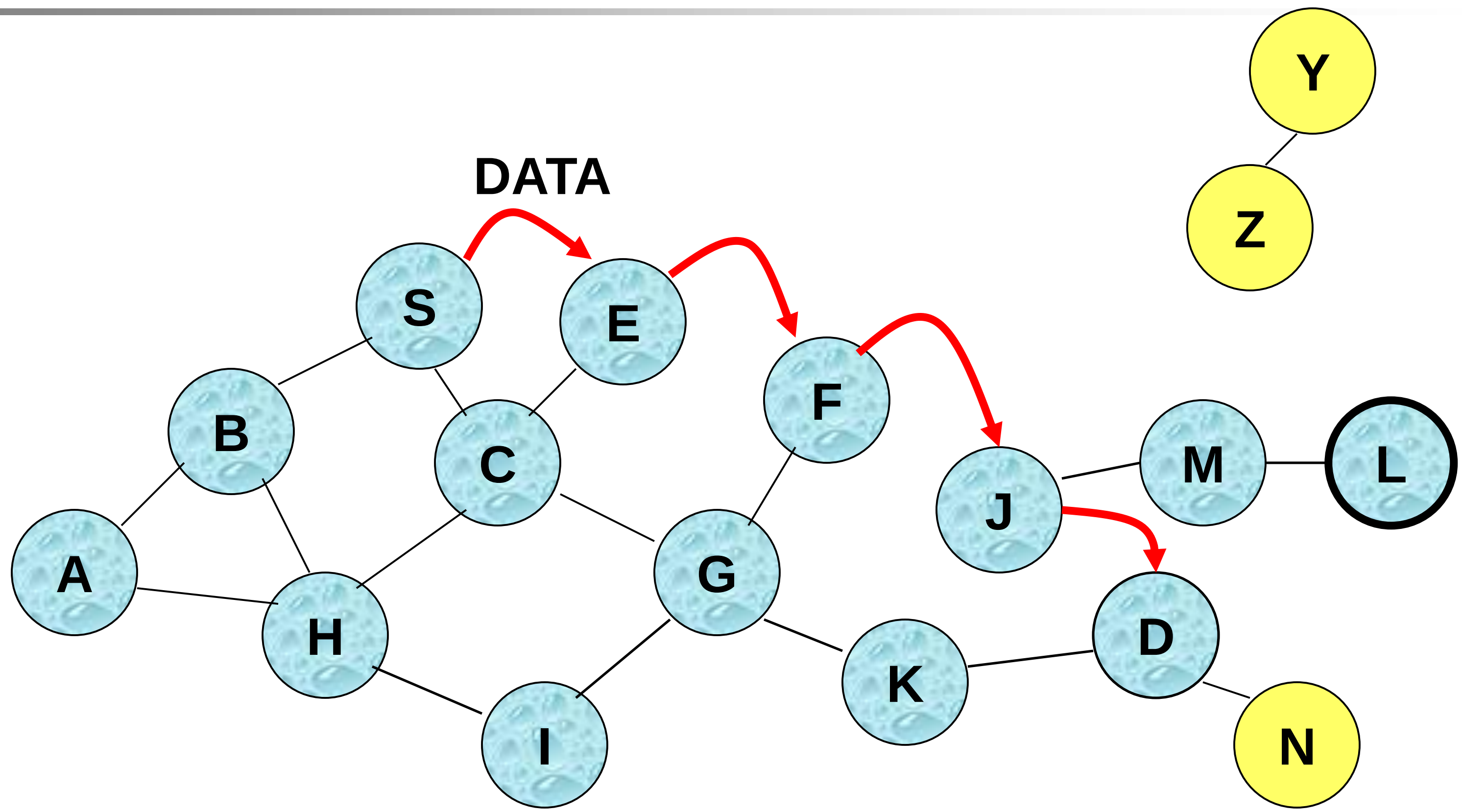


Forward links are setup when RREP travels along the reverse path



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 is 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

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

Exploiting Location Information in routing

Location-Aided Routing (LAR)

- Exploits location information to limit scope of RREQ
  - Location information may be obtained using GPS

- *Expected Zone* is determined as a region that is expected to hold the current location of destination
  - Expected region determined based on potentially old location information, and knowledge of the destination's speed

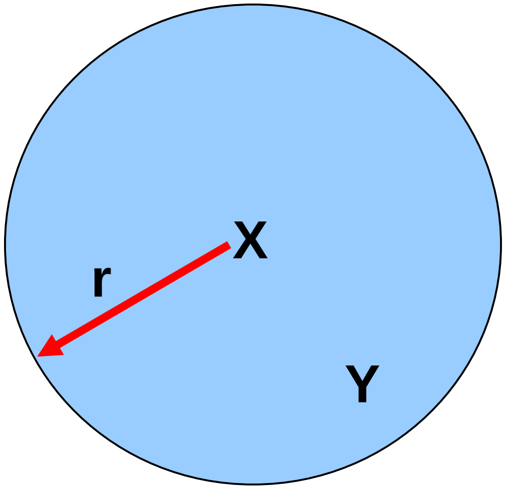
- Route requests limited to a *Request Zone*
  - Such that Expected Zone contained in Request Zone

Expected Zone in LAR

X = last known location of node D, at time t0

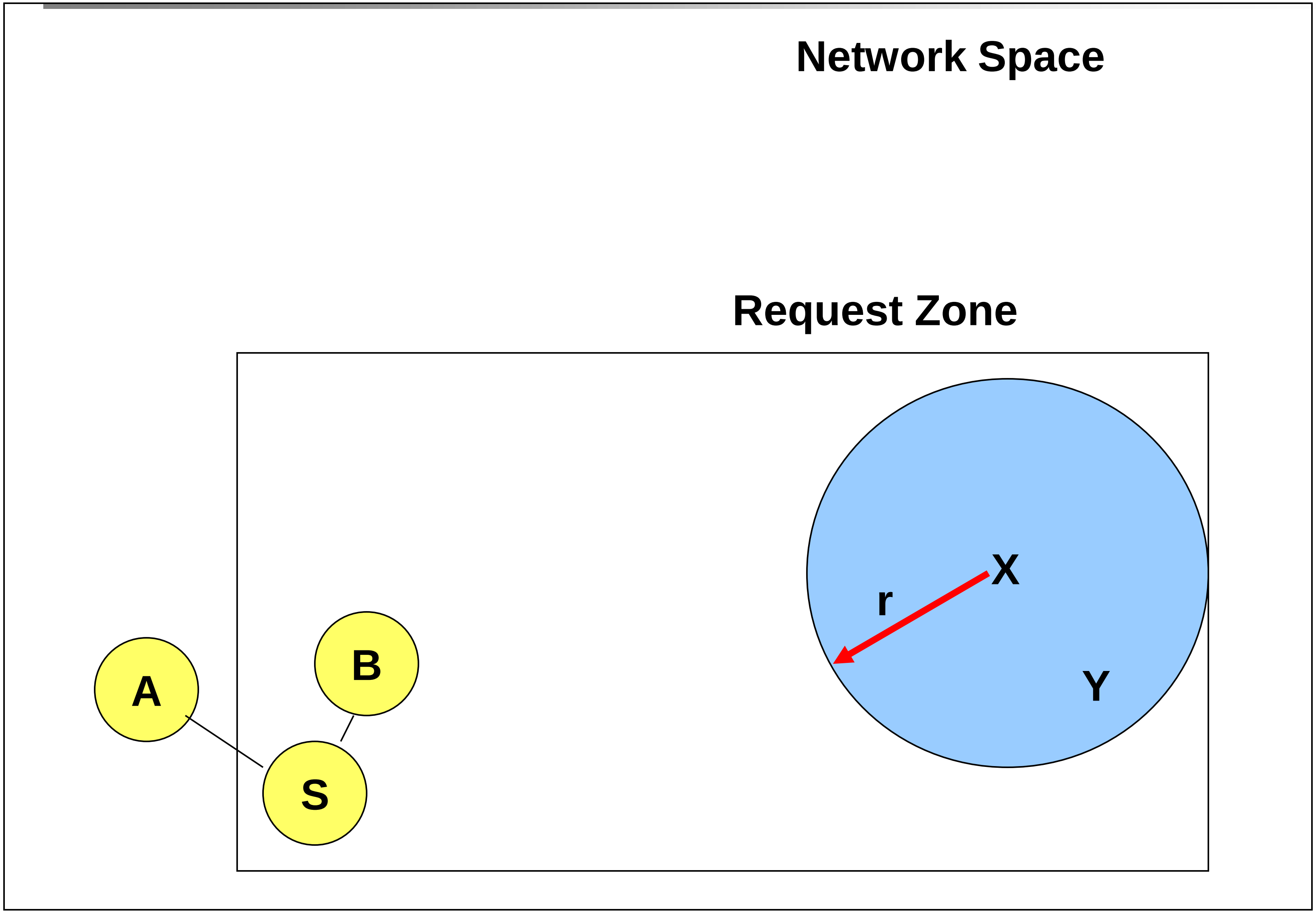
Y = location of node D at current time t1, unknown to node S

r = (t1 - t0) * estimate of D’s speed



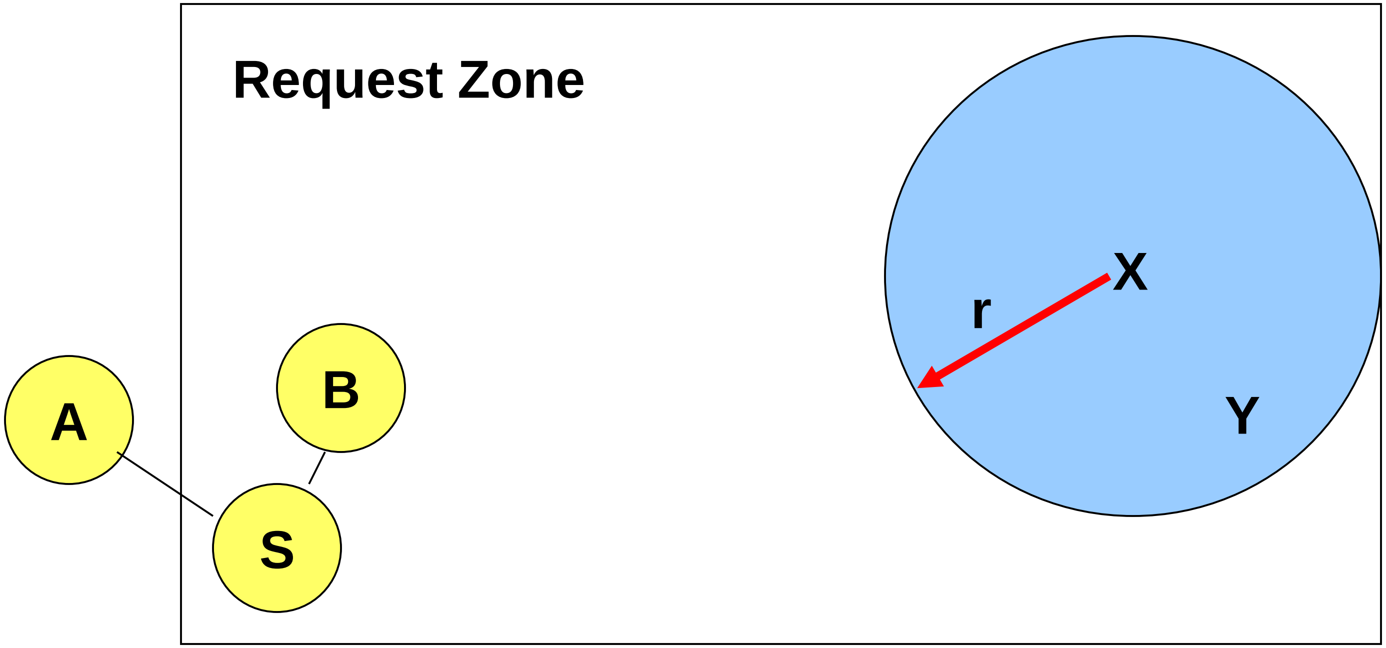
Expected Zone

Request Zone in LAR



LAR

- Only nodes within the request zone forward RREQ
  - Node A does not forward RREQ, but node B does
- Request zone explicitly specified in the route request
  - Each node must know its physical location to determine whether it is within the request zone



LAR

- Only nodes within the request zone forward route requests

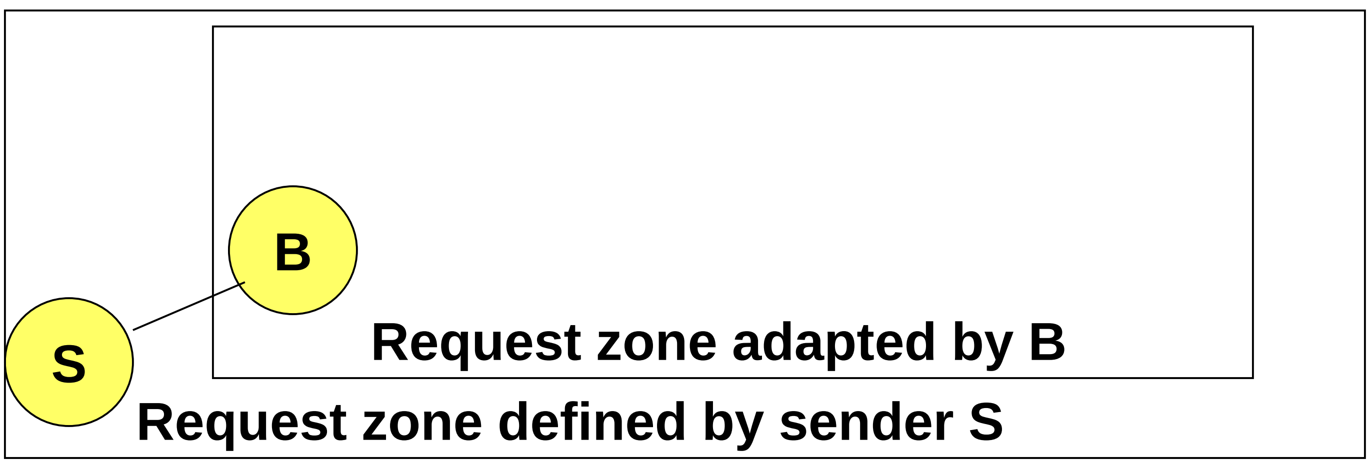
- If route discovery using the smaller request zone fails
  - Initiate new discovery with large zone
  - Perhaps large zone = entire network

- Rest of route discovery protocol similar to DSR

LAR Variations: Adaptive Request Zone

- Each node may modify the request zone
  - And include it in the forwarded RREQ

- Modified request zone may be determined using more recent/accurate information, and may be smaller than the original request zone



Location Aided Routing (LAR)

- Advantages
  - reduces the scope of route request flood
  - reduces overhead of route discovery

- Disadvantages
  - Does not take into account possible existence of obstructions for radio transmissions
  - Assumes that destination’s location information is not too stale

Questions

Brief Overview of Other Ideas

MARP: Multi-Agent Location Routing

- Problem is to obtain global location information proactively
- Location information useful (for routing, geocasting, etc.)
  - Approach: Biologically inspired algorithm (from ants)

- Ants walk randomly in search of food
  - Ants deposit pheromone while walking
  - Ants get attracted toward pheromone smell
  - Pheromones evaporate with time

  - When a route to food found, ants come back home
  - Pheromone deposition increases
  - More ants converge toward this pheromone route
  - Shortest path gets quickly reinforced
  - Other longer routes evaporate with time

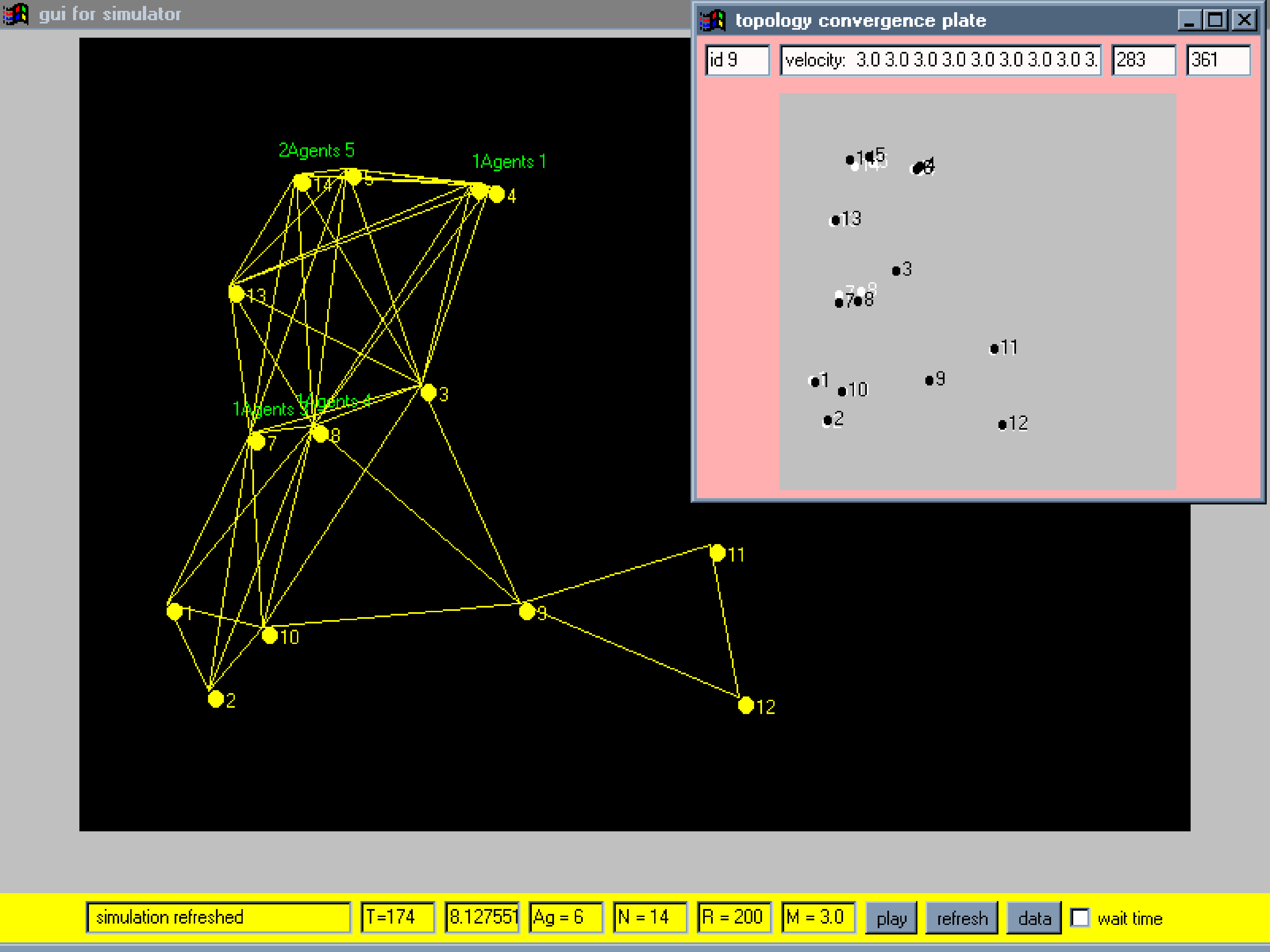
Now …

What happens if

ants were repelled by pheromones

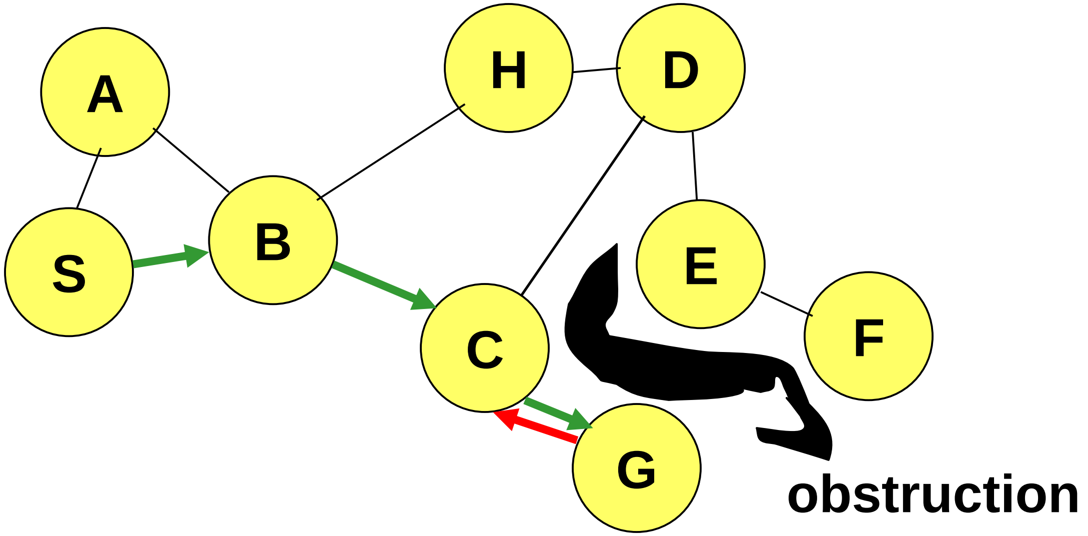
Location Management with Ants

- Each ant (java agent) increments counter
  - Whenever it visits a node
- Other agents repelled by high values
  - Repelled by pheromones
  - Visits directions which have least counter values
- Over time, agents visit nodes with least values
  - This distributes agents homogeneously
  - Every node is kept track of
- Agents exchange information upon meeting
- Any node quickly learns about entire network


gui for simulator
topology convergence plate
id 9
velocity: 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.0 3.
283
361
2Agents 5
1Agents 1
1Agents 3
1Agents 4
simulation refreshed
T=174
8.127551
Ag = 6
N = 14
R = 200
M = 3.0
play
refresh
data
wait time

Geographic Distance Routing (GEDIR)

- Greedy geographic routing can be stuck (local maxima)
  - Packet goes to G for destination F
- Algorithm guarantees delivery
  - Use left-hand rule to guide packets around hole/obstacle
  - Basically, backtrack to nodes on the left side always



Proactive Protocols

Proactive Protocols

- Most of the schemes discussed so far are reactive
- Proactive schemes based on distance-vector and link-state mechanisms have also been proposed

Link State Routing [Huitema95]

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

Fish Eye Routing

- Overhead of LSR too much
  - Every node sends its own link states periodically

- Instead, adapt the periodicity and TTL of updates

  - Transmit updates frequently with low TTL
  - Transmit updates infrequently with high TTL

- Fish Eye: Clarity of vision degrades with distance

- Routing packets can be sent to approx direction
  - It does micro-level course correstion as it approaches dest.

Hybrid Protocols

Zone Routing Protocol (ZRP) [Haas98]

Zone routing protocol combines

- Proactive protocol: which pro-actively updates network state and maintains route regardless of whether any data traffic exists or not

- Reactive protocol: which only determines route to a destination if there is some data to be sent to the destination

ZRP

- All nodes within hop distance at most *d* from a node X are said to be in the routing zone of node X

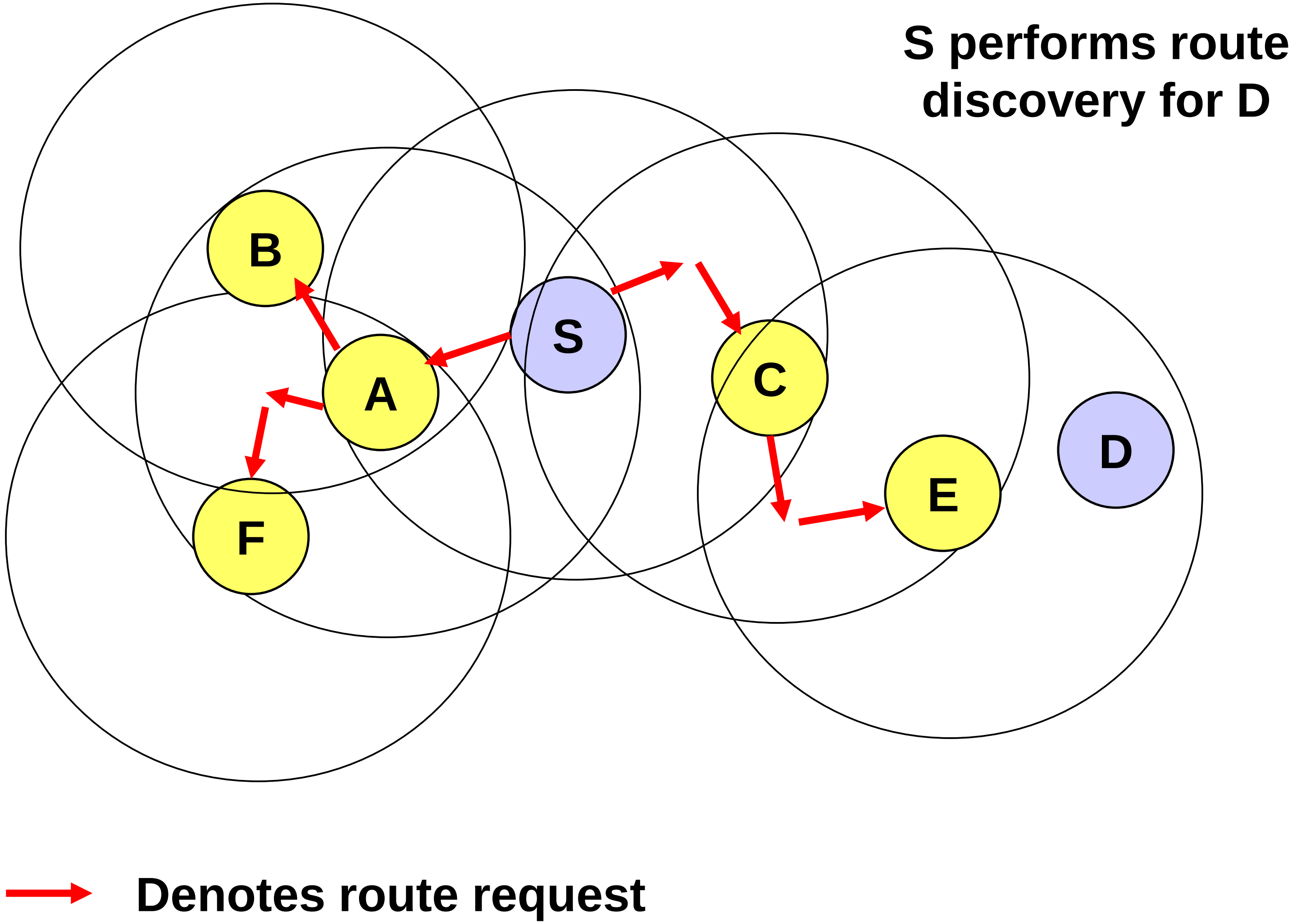
- All nodes at hop distance exactly *d* are said to be peripheral nodes of node X's routing zone

ZRP

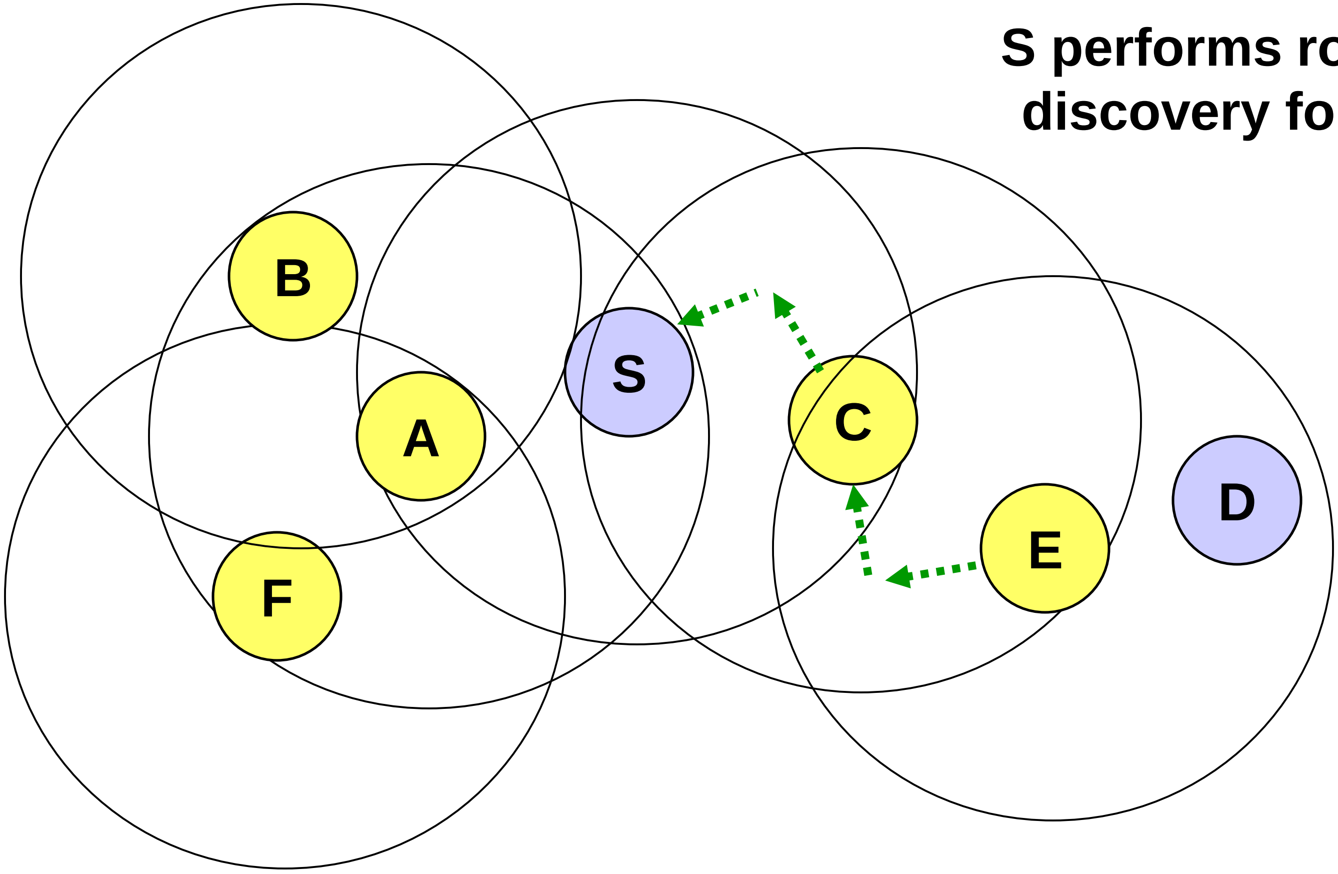
- Intra-zone routing: Pro-actively maintain state information for links within a short distance from any given node
  - Routes to nodes within short distance are thus maintained proactively (using, say, link state or distance vector protocol)

- Inter-zone routing: Use a route discovery protocol for determining routes to far away nodes. Route discovery is similar to DSR with the exception that route requests are propagated via peripheral nodes.

ZRP: Example with Zone Radius = *d* = 2



ZRP: Example with *d = 2*

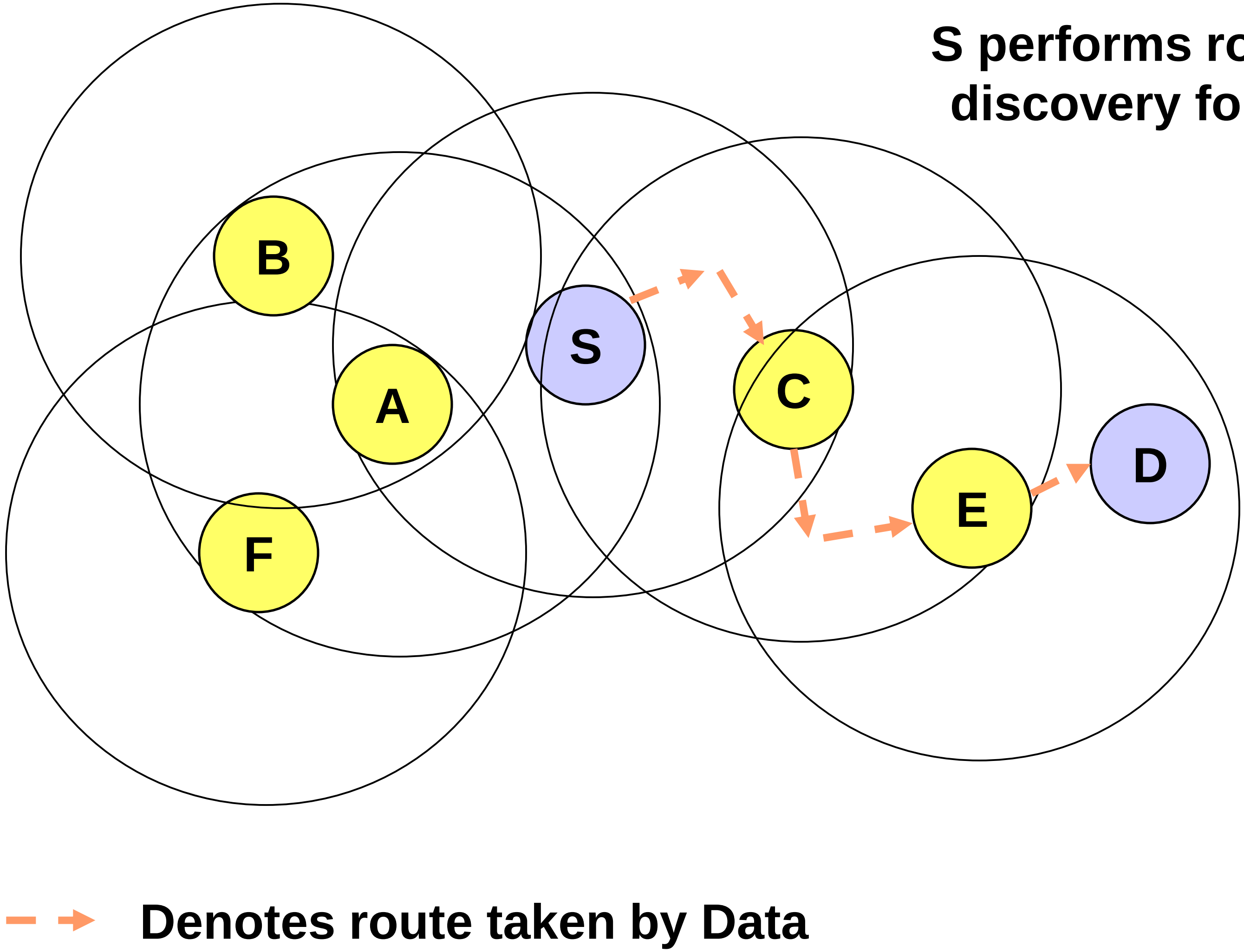


S performs route discovery for D

Denotes route reply

E knows route from E to D, so route request need not be forwarded to D from E

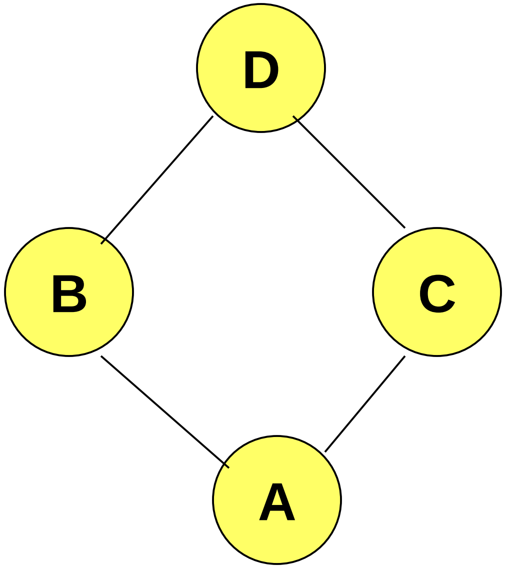
ZRP: Example with *d = 2*



Questions?

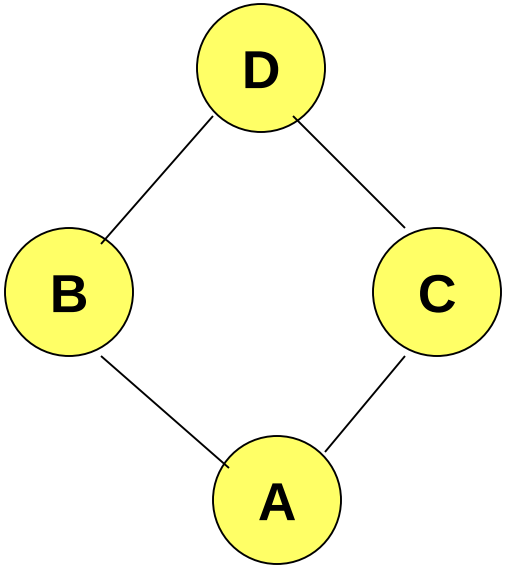
Broadcast Storm Problem [Ni99Mobicom]

- When node A broadcasts a route query, nodes B and C both receive it
- B and C both forward to their neighbors
- B and C transmit at about the same time since they are reacting to receipt of the same message from A
- This results in a high probability of collisions



Broadcast Storm Problem

- Redundancy: A given node may receive the same route request from too many nodes, when one copy would have sufficed
- Node D may receive from nodes B and C both



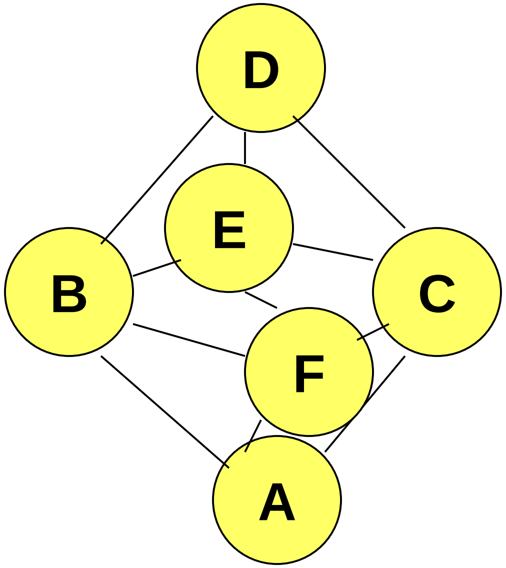
Solutions for Broadcast Storm

- Probabilistic scheme: On receiving a route request for the first time, a node will re-broadcast (forward) the request with probability p

- Also, re-broadcasts by different nodes should be staggered by using a collision avoidance technique (wait a random delay when channel is idle)
  - this would reduce the probability that nodes B and C would forward a packet simultaneously in the previous example

Solutions for Broadcast Storms

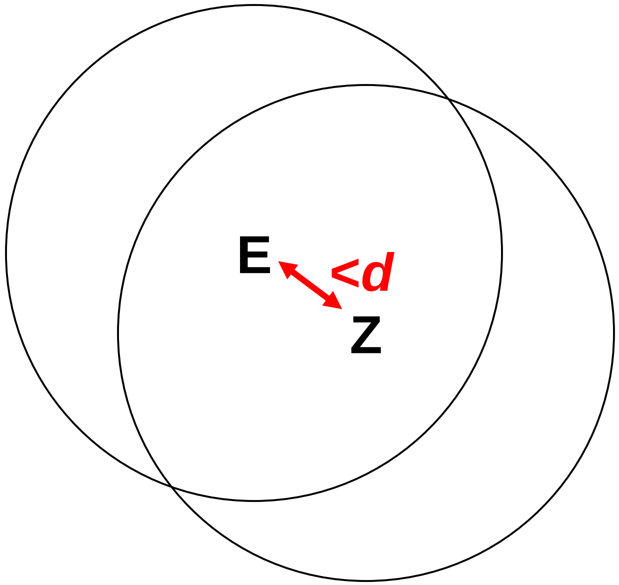
- Counter-Based Scheme: If node E hears more than *k* neighbors broadcasting a given route request, before it can itself forward it, then node E will not forward the request
- Intuition: *k* neighbors together have probably already forwarded the request to all of E's neighbors



Solutions for Broadcast Storms

- Distance-Based Scheme: If node E hears RREQ broadcasted by some node Z within physical distance *d*, then E will not re-broadcast the request

- Intuition: Z and E are too close, so transmission areas covered by Z and E are not very different
  - if E re-broadcasts the request, not many nodes who have not already heard the request from Z will hear the request



Summary: Broadcast Storm Problem

- Flooding is used in many protocols, such as Dynamic Source Routing (DSR)

- Problems associated with flooding
  - collisions
  - redundancy

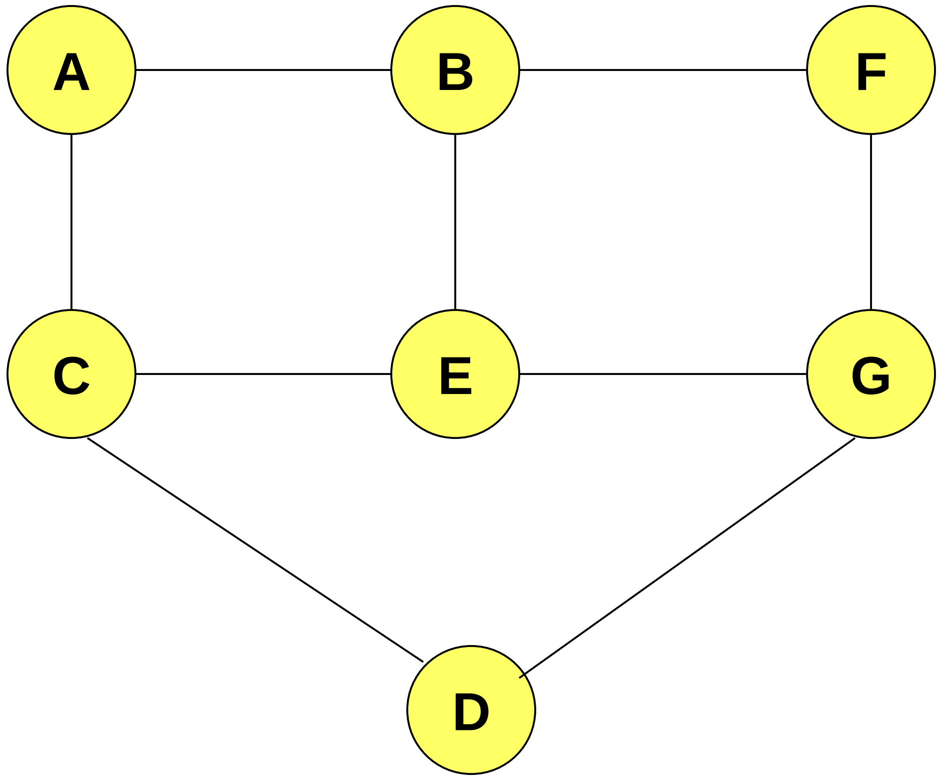
- Collisions may be reduced by “jittering” (waiting for a random interval before propagating the flood)

- Redundancy may be reduced by selectively re-broadcasting packets from only a subset of the nodes

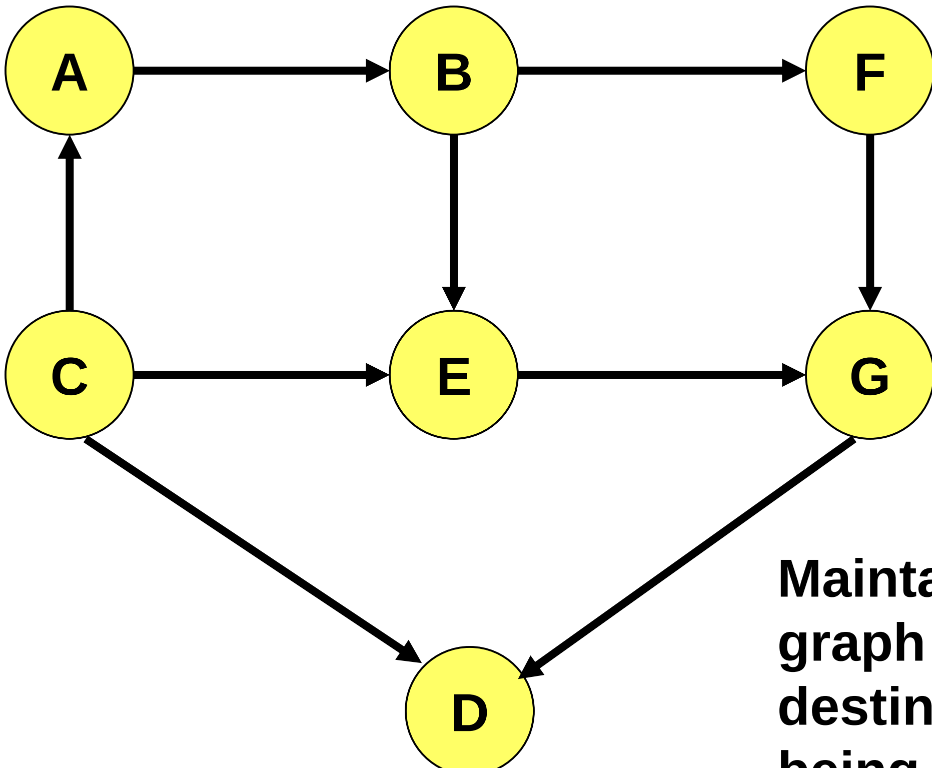
So far ...

- All protocols discussed so far perform some form of flooding
- Now we will consider protocols which try to reduce/avoid such behavior

Link Reversal Algorithm [Gafni81]



Link Reversal Algorithm



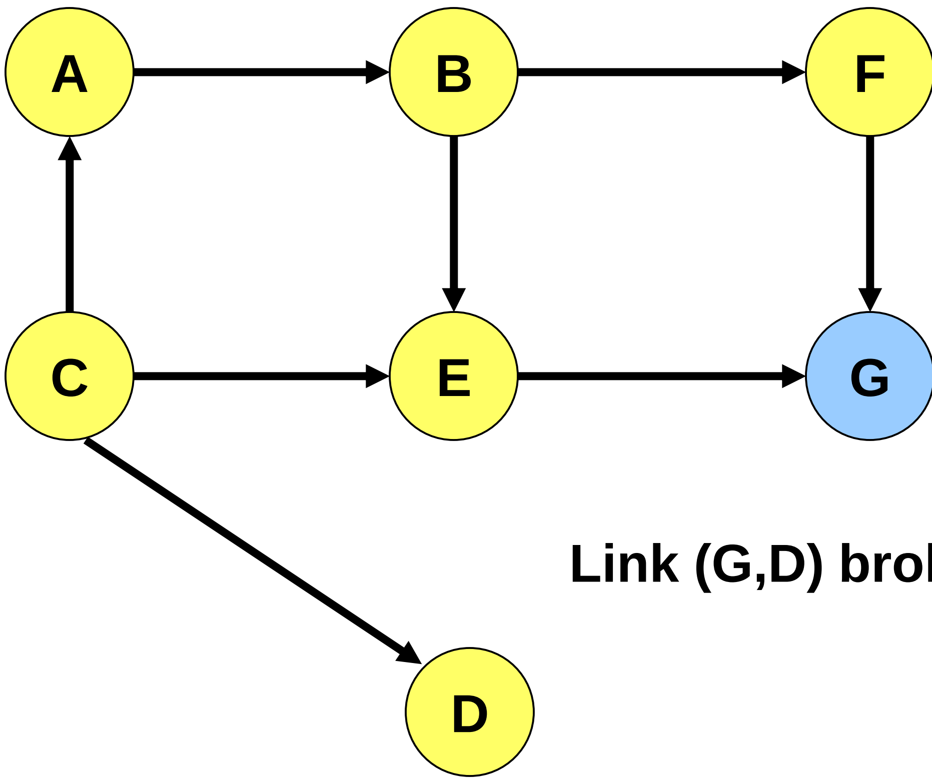
Links are bi-directional

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

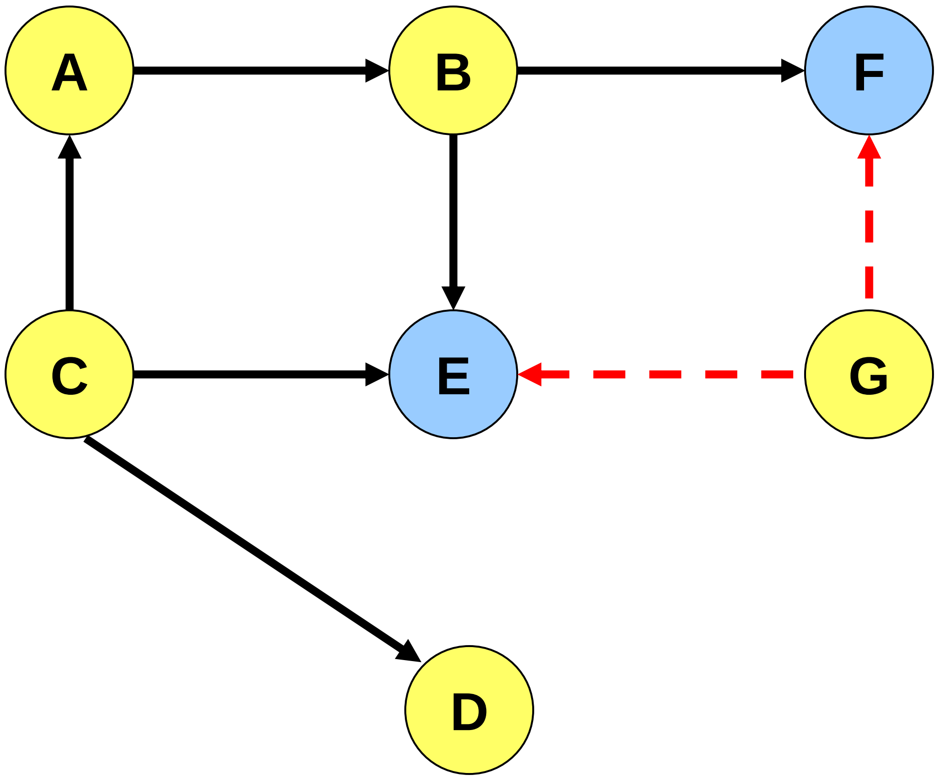


Link (G,D) broke

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

Node G has no outgoing links

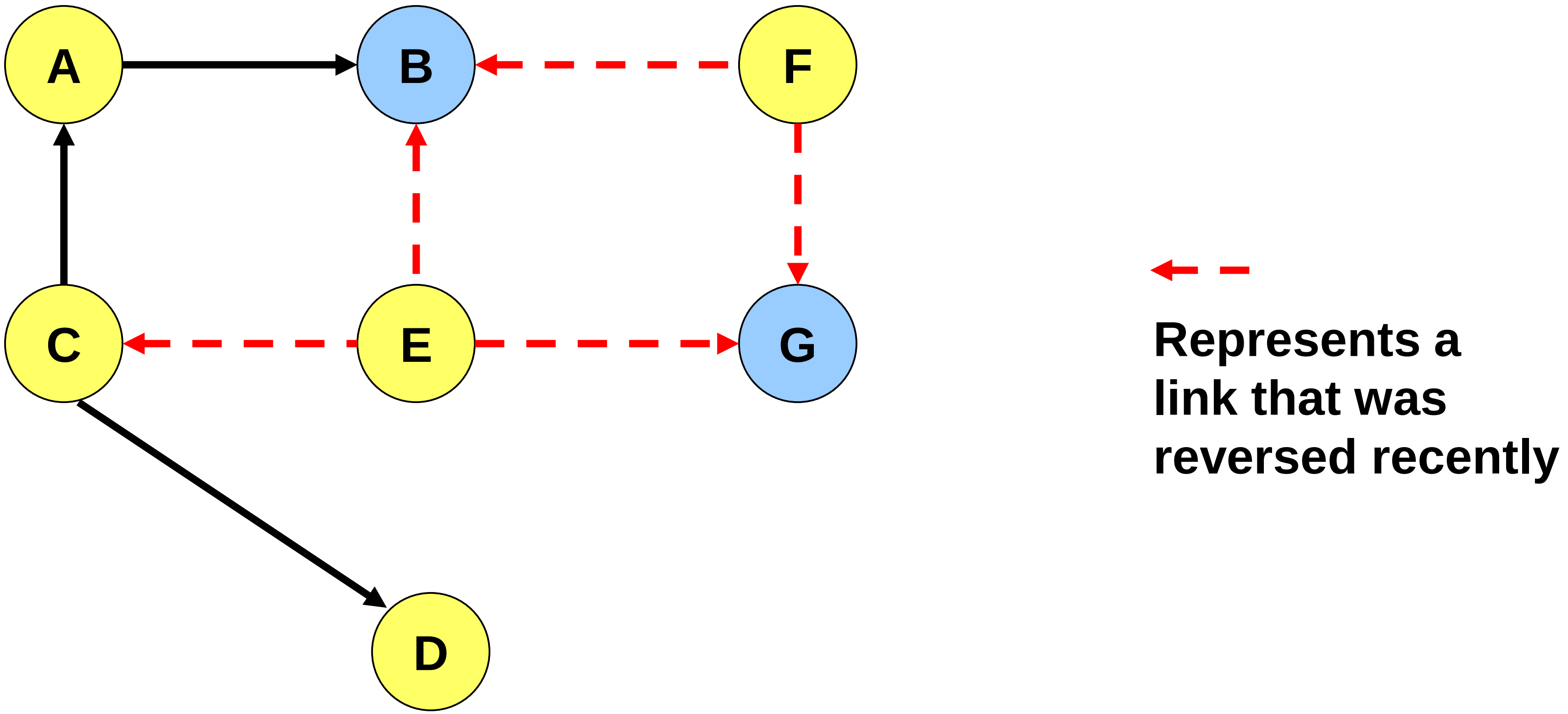
Link Reversal Algorithm



Represents a link that was reversed recently

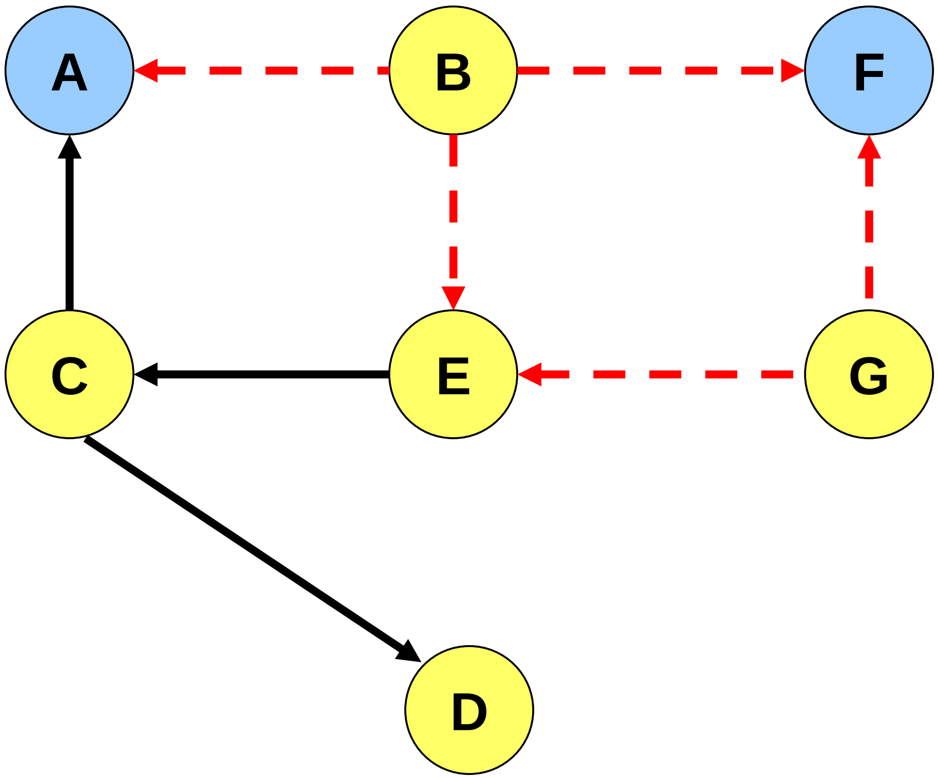
Now nodes E and F have no outgoing links

Link Reversal Algorithm



Now nodes B and G have no outgoing links

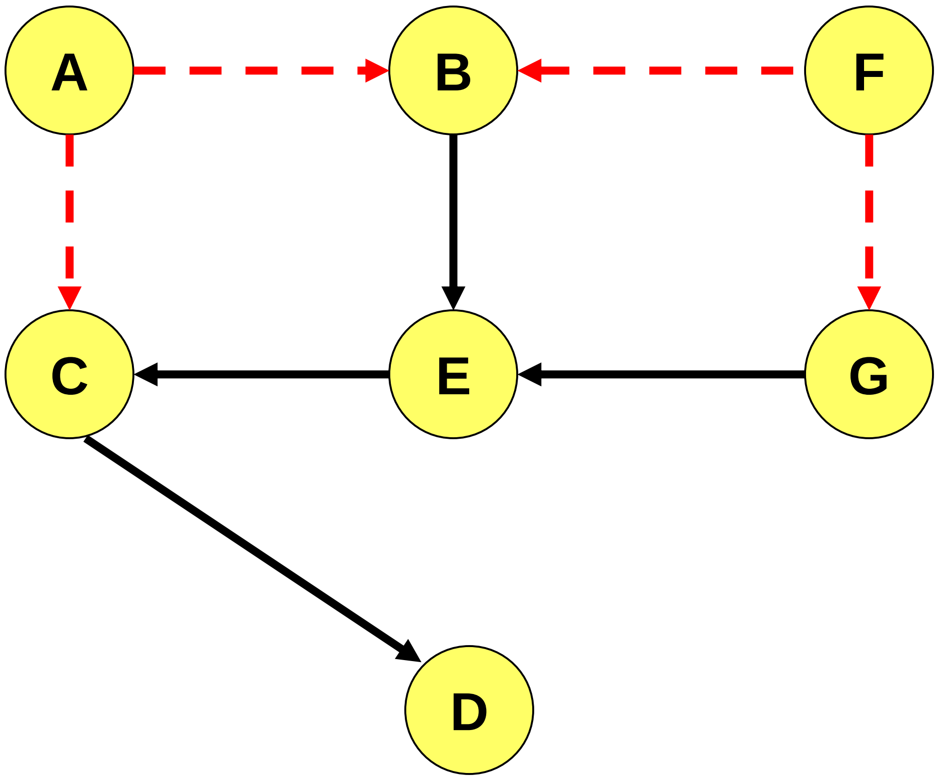
Link Reversal Algorithm



Represents a link that was reversed recently

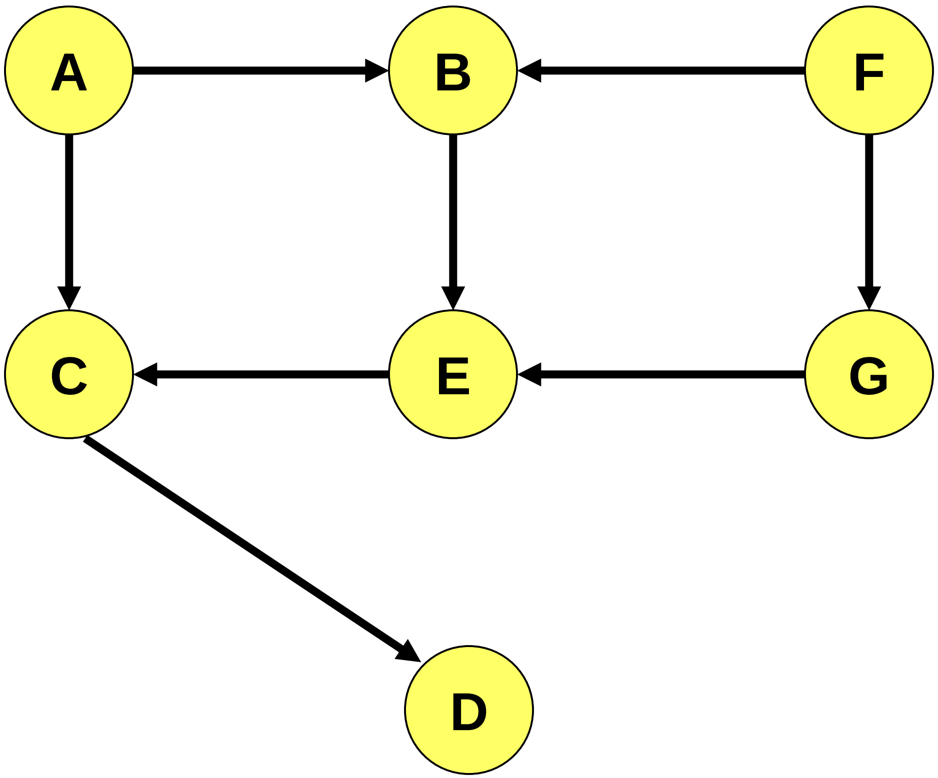
Now nodes A and F have no outgoing links

Link Reversal Algorithm



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

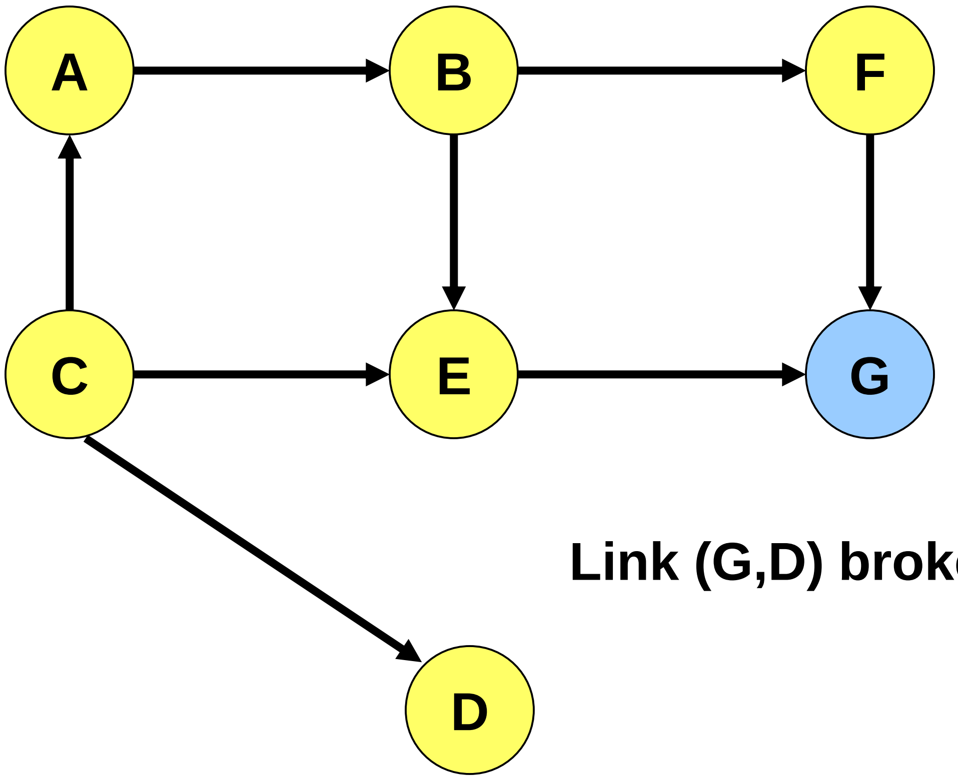
Link Reversal Algorithm

- 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*

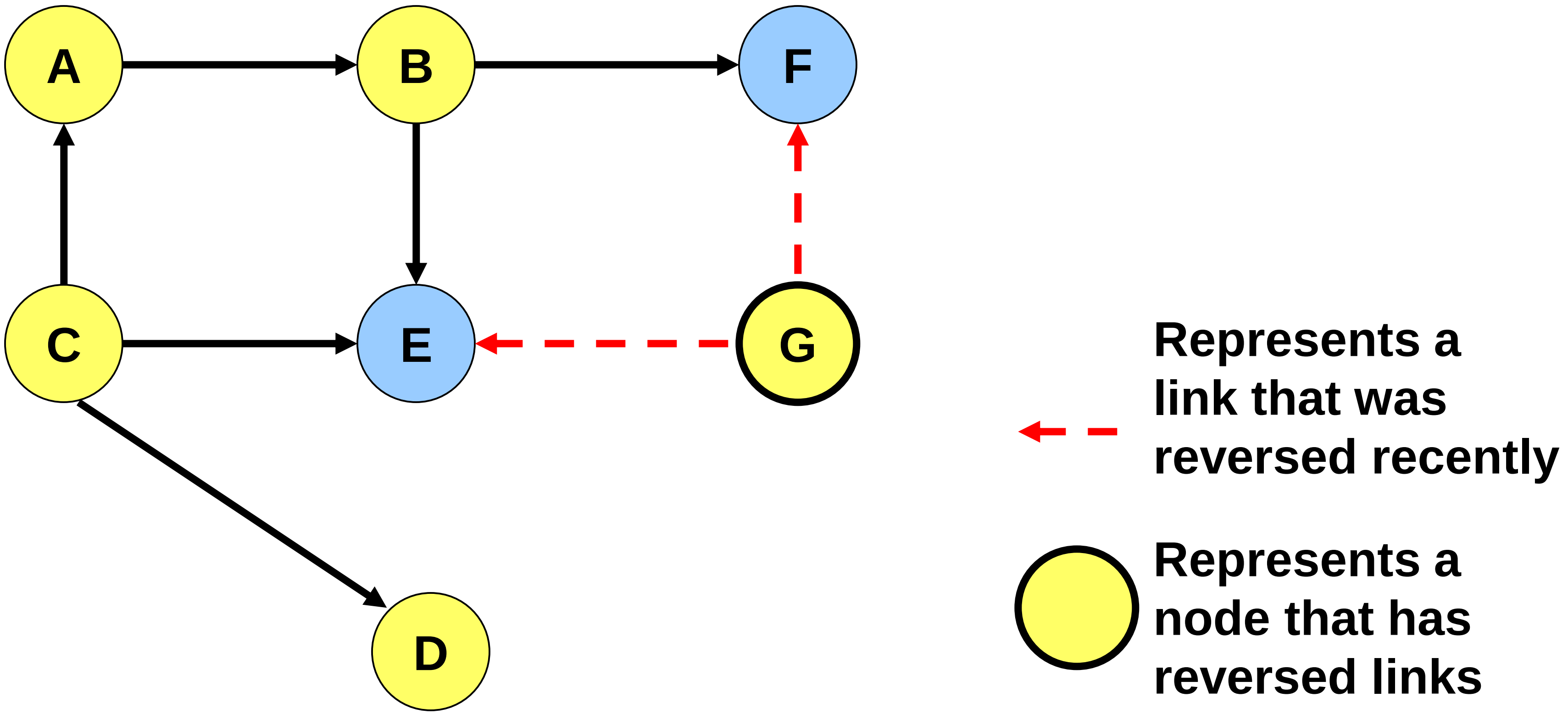
Partial Reversal Method



Link (G,D) broke

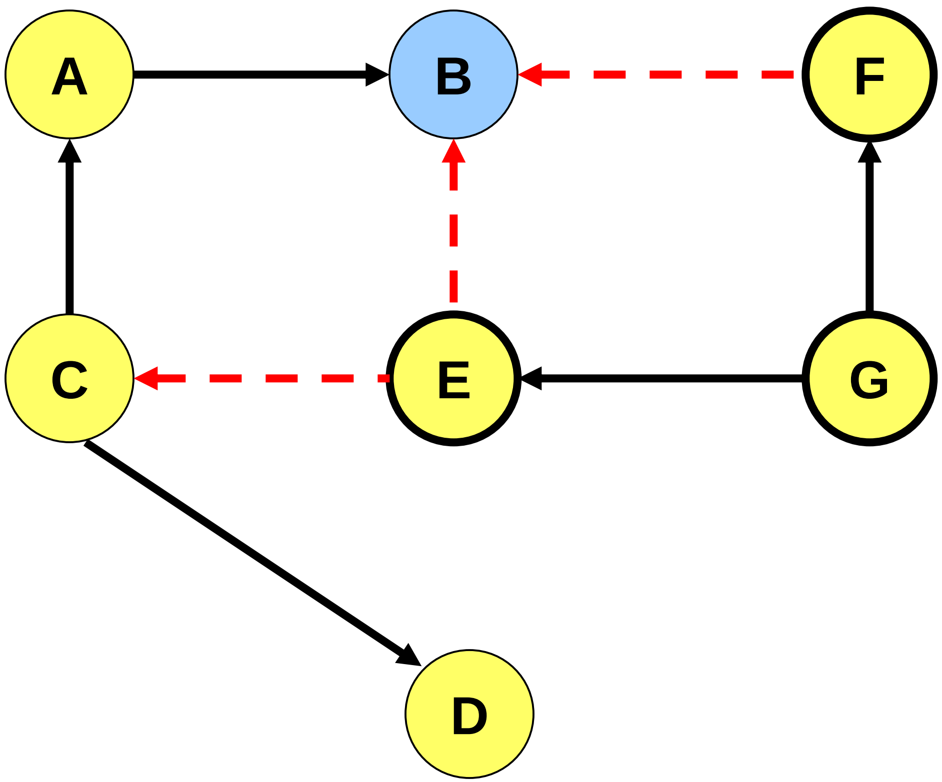
Node G has no outgoing links

Partial Reversal Method



Now nodes E and F have no outgoing links

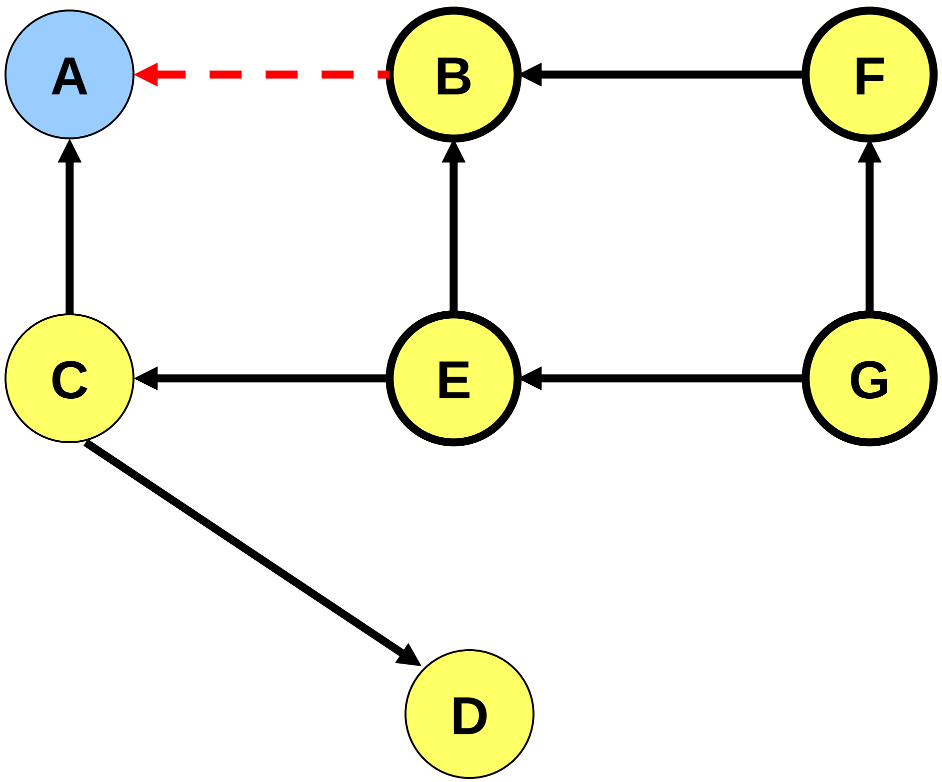
Partial Reversal Method



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

Now node B has no outgoing links

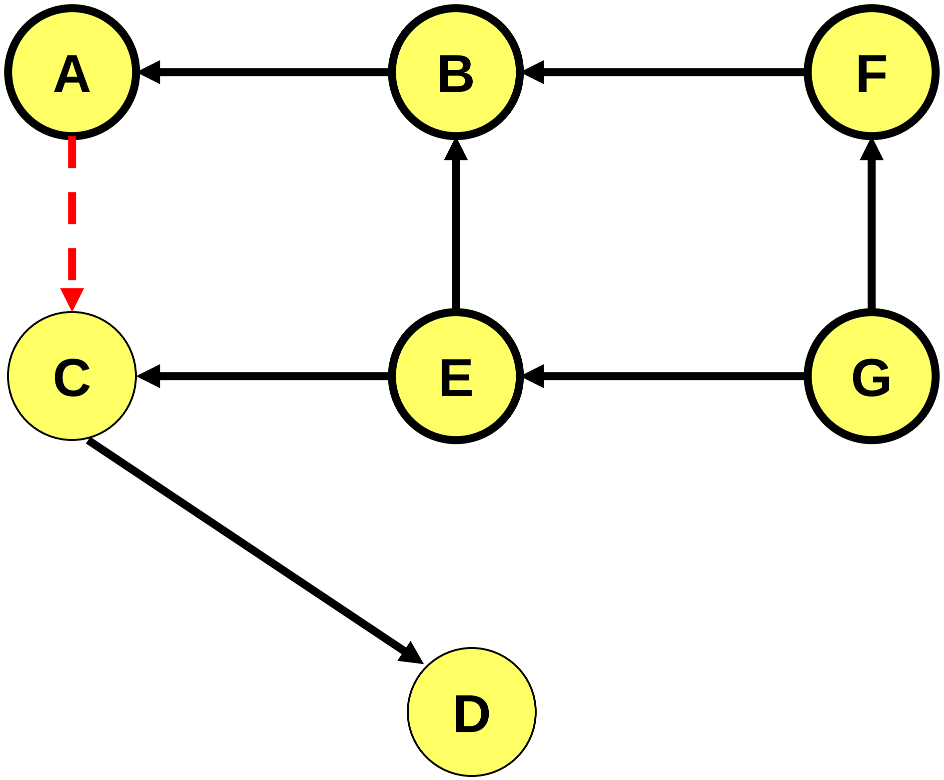
Partial Reversal Method



Represents a link that was reversed recently

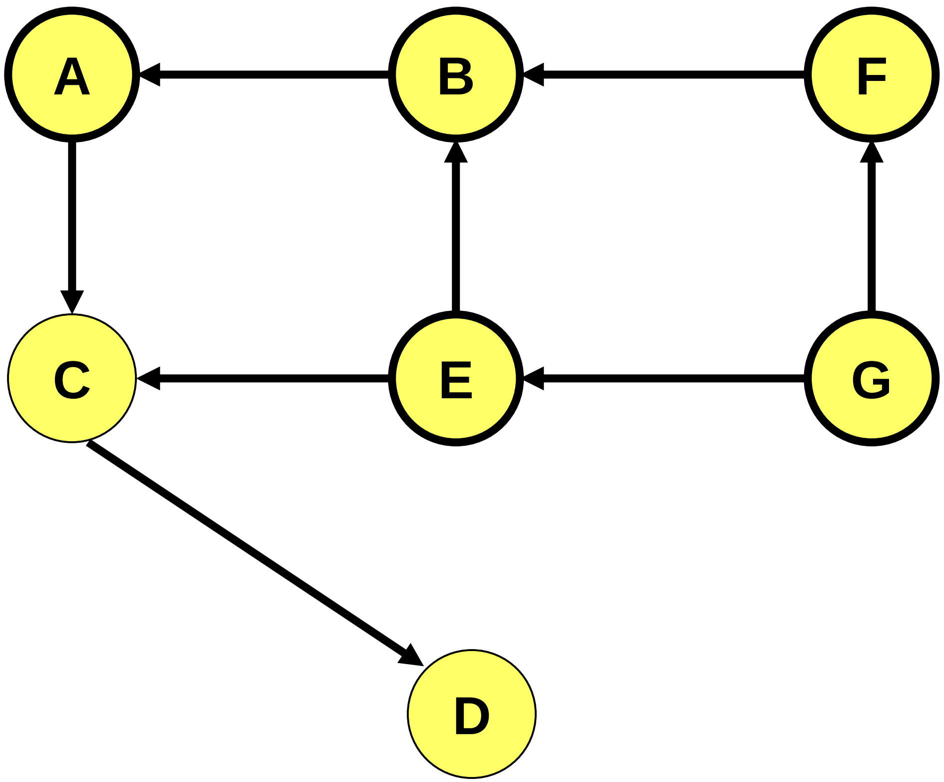
Now node A has no outgoing links

Partial Reversal Method



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 Methods: Advantages

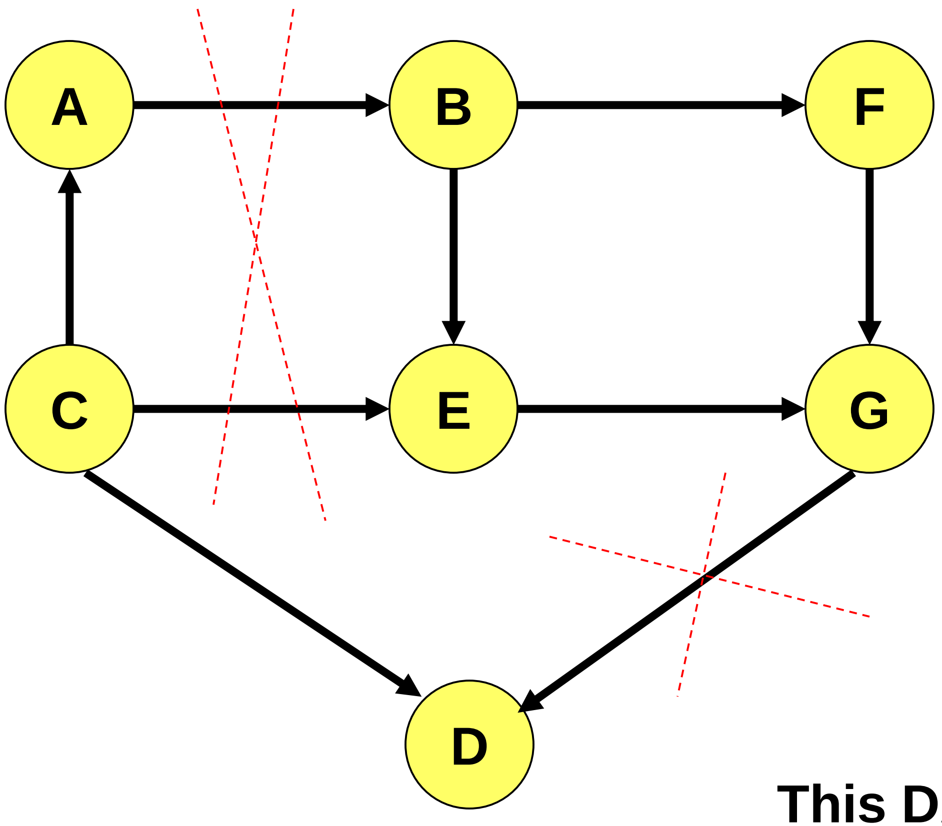
- Link reversal methods attempt to limit updates to routing tables at nodes in the vicinity of a broken link
  - Partial reversal method tends to be better than full reversal method

- Each node may potentially have multiple routes to a destination

Link Reversal Methods: Disadvantage

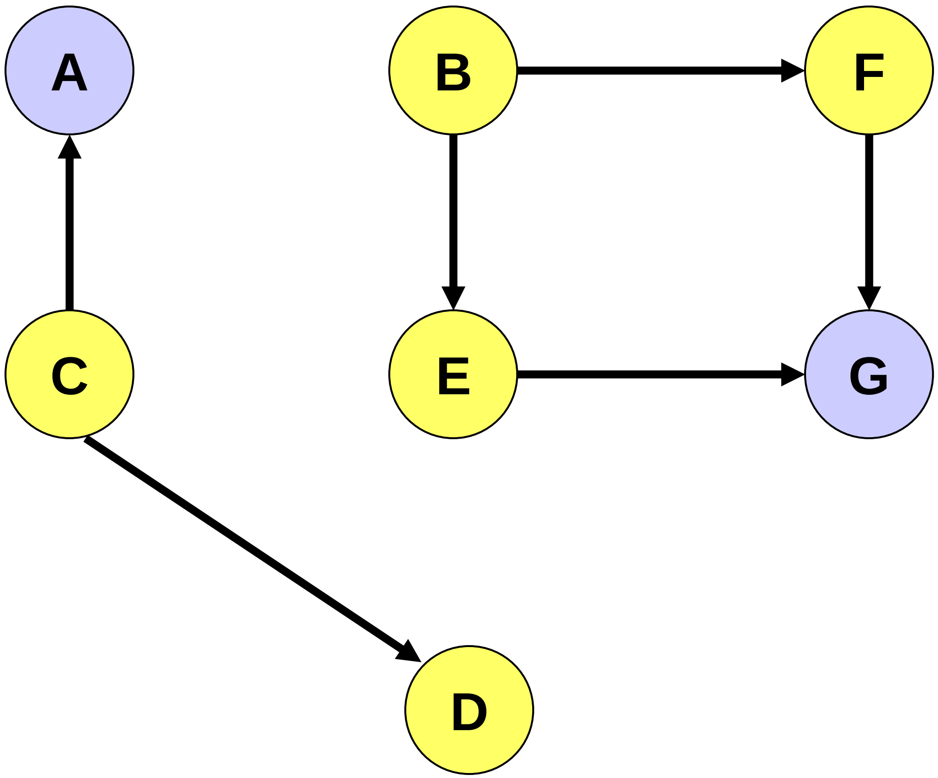
- Need a mechanism to detect link failure
  - hello messages may be used
  - but hello messages can add to contention
- If network is partitioned, link reversals continue indefinitely

Link Reversal in a Partitioned Network



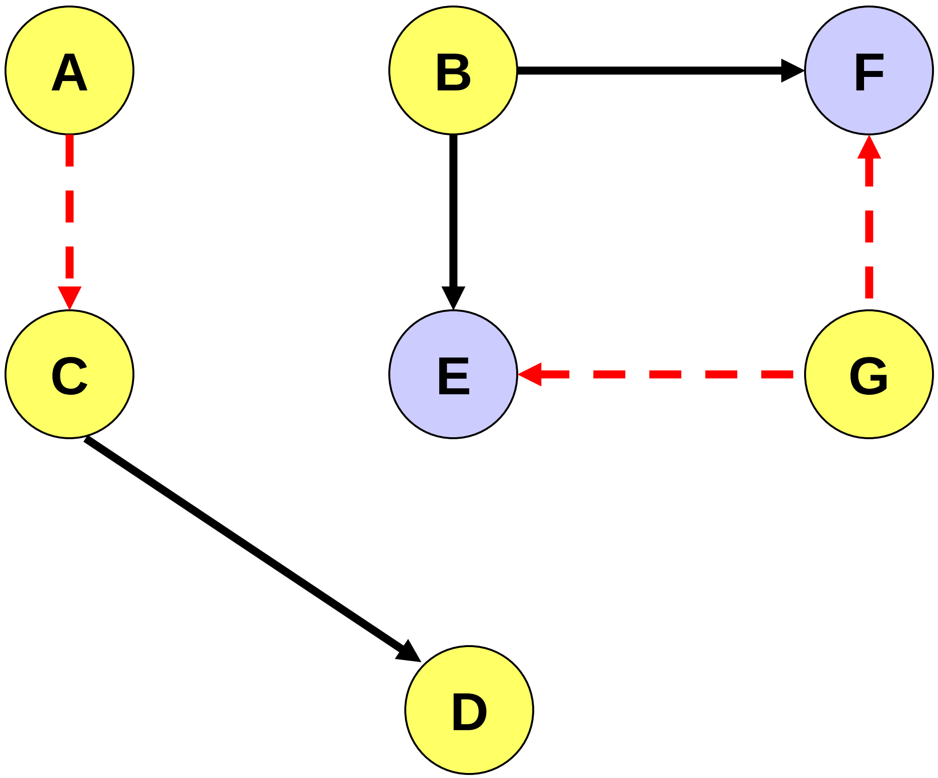
This DAG is for *destination node D*

Full Reversal in a Partitioned Network



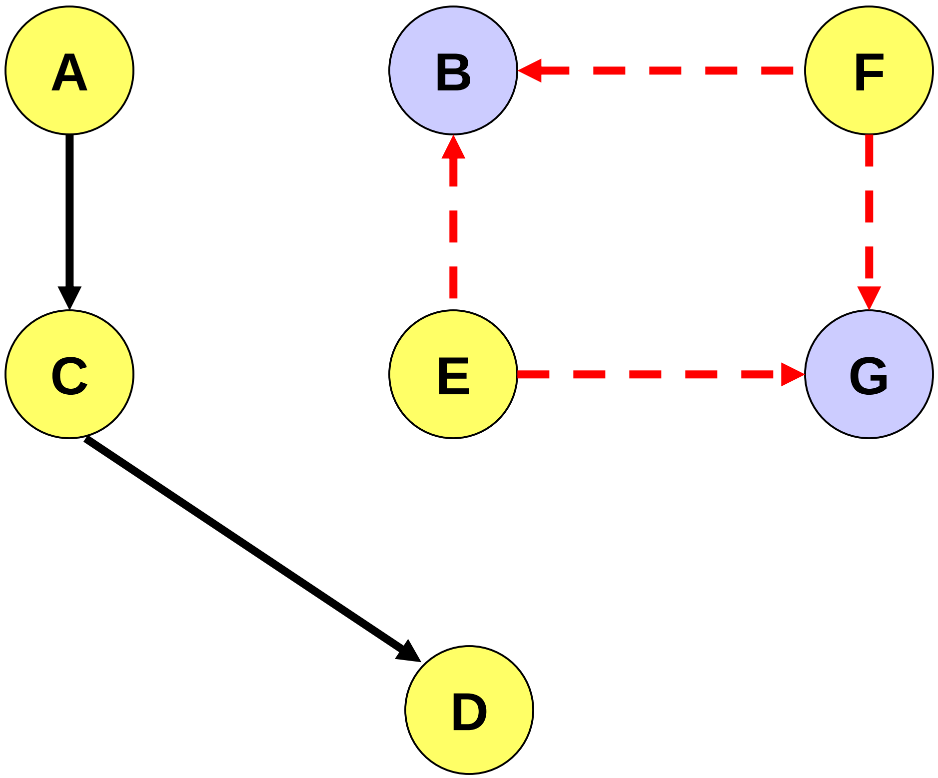
A and G do not have outgoing links

Full Reversal in a Partitioned Network



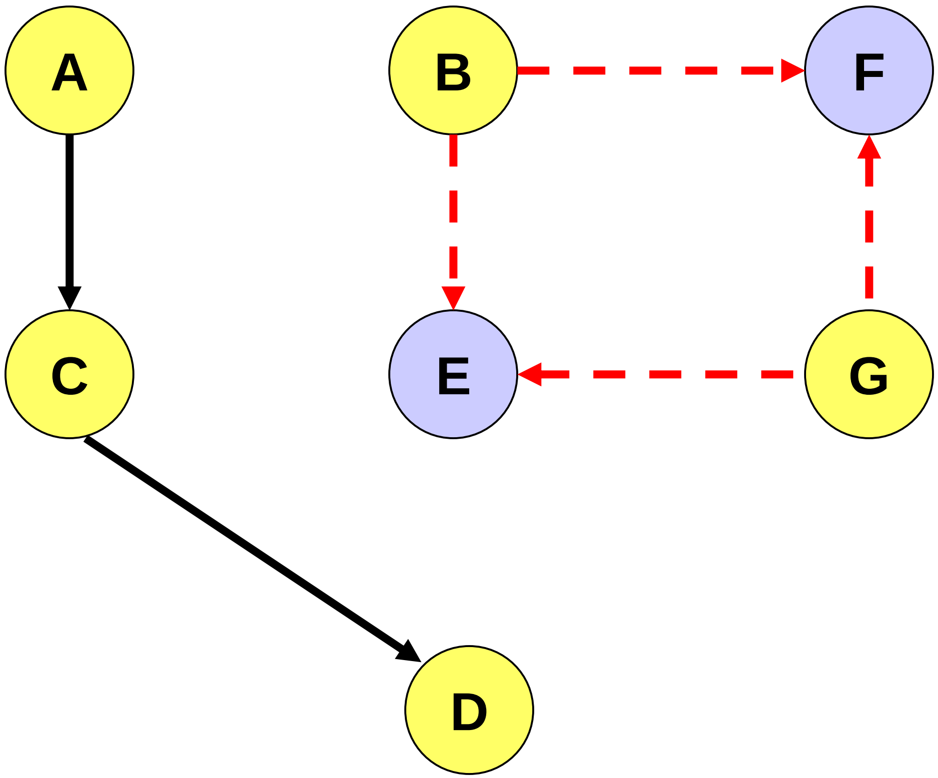
E and F do not have outgoing links

Full Reversal in a Partitioned Network



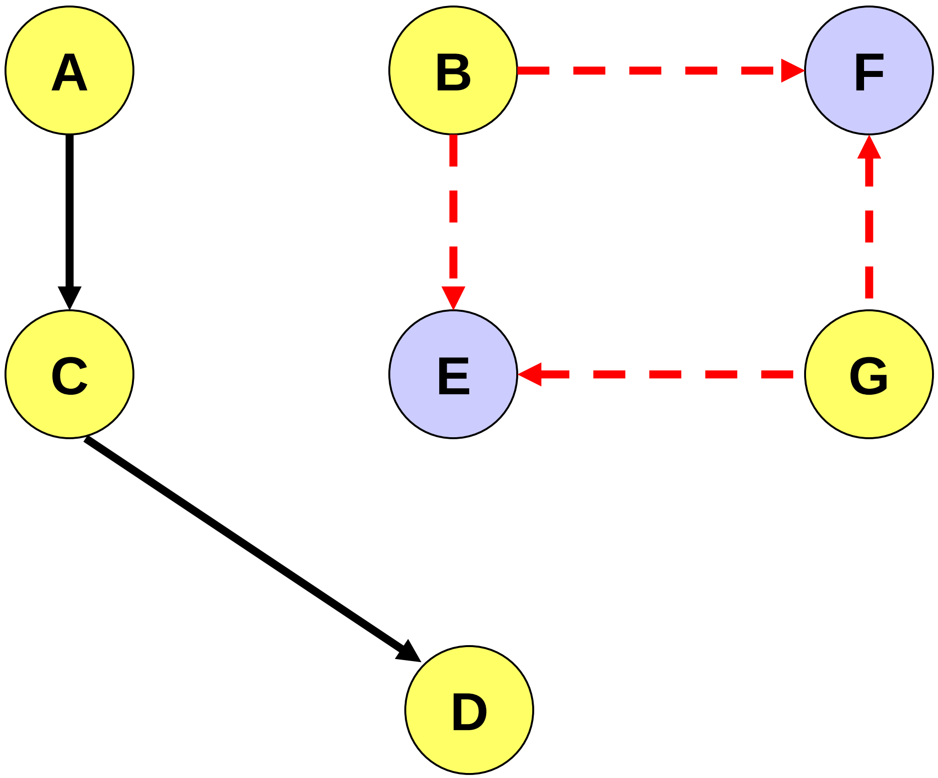
B and G do not have outgoing links

Full Reversal in a Partitioned Network



E and F do not have outgoing links

Full Reversal in a Partitioned Network



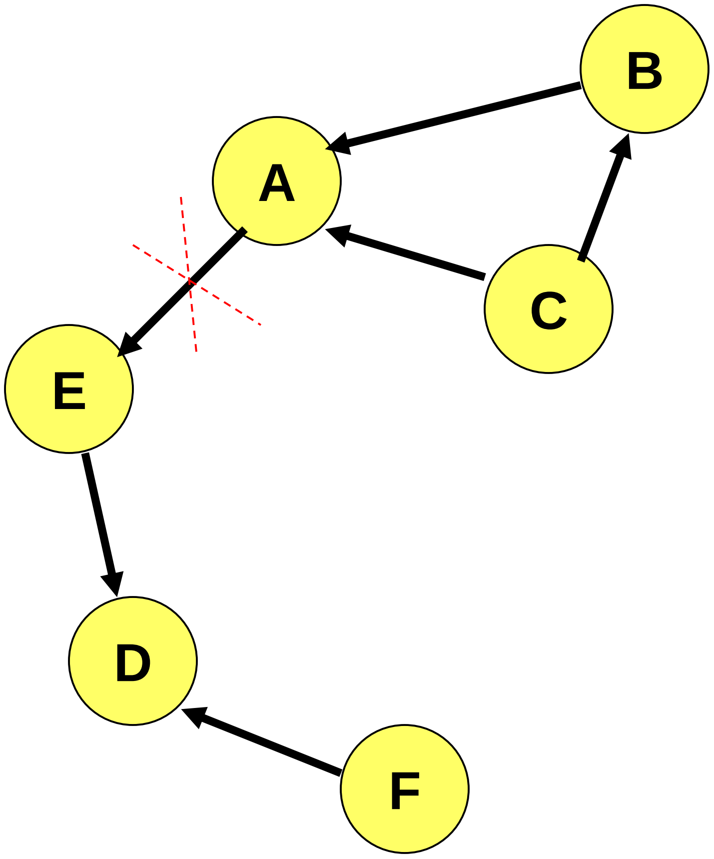
- In the partition disconnected from destination D, link reversals continue, until the partitions merge
- Need a mechanism to minimize this wasteful activity
- Similar scenario can occur with partial reversal method too

Temporally-Ordered Routing Algorithm (TORA) [Park97Infocom]

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

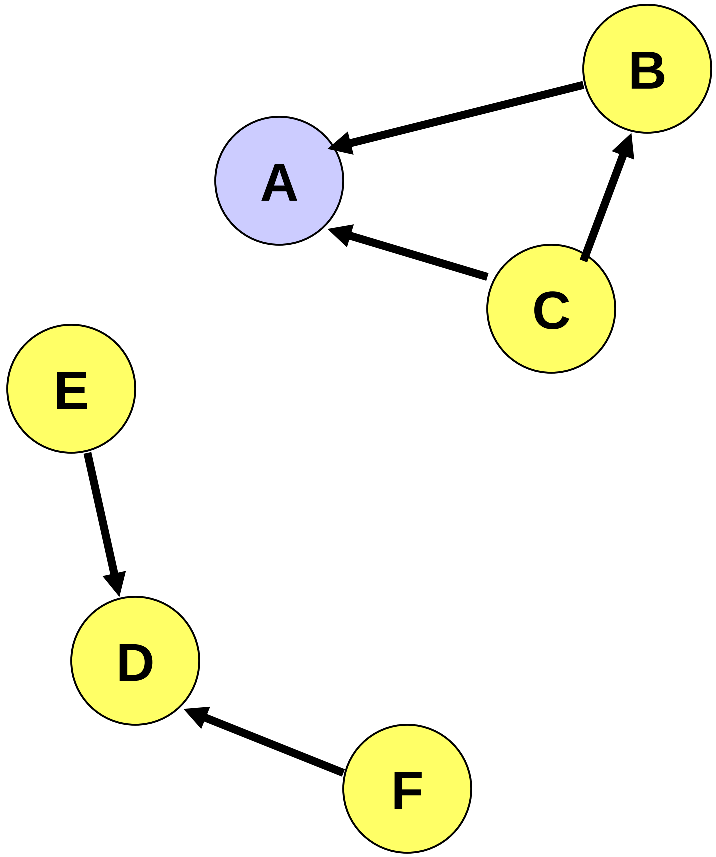
- When a partition is detected, all nodes in the partition are informed, and link reversals in that partition cease

Partition Detection in TORA



DAG for destination D

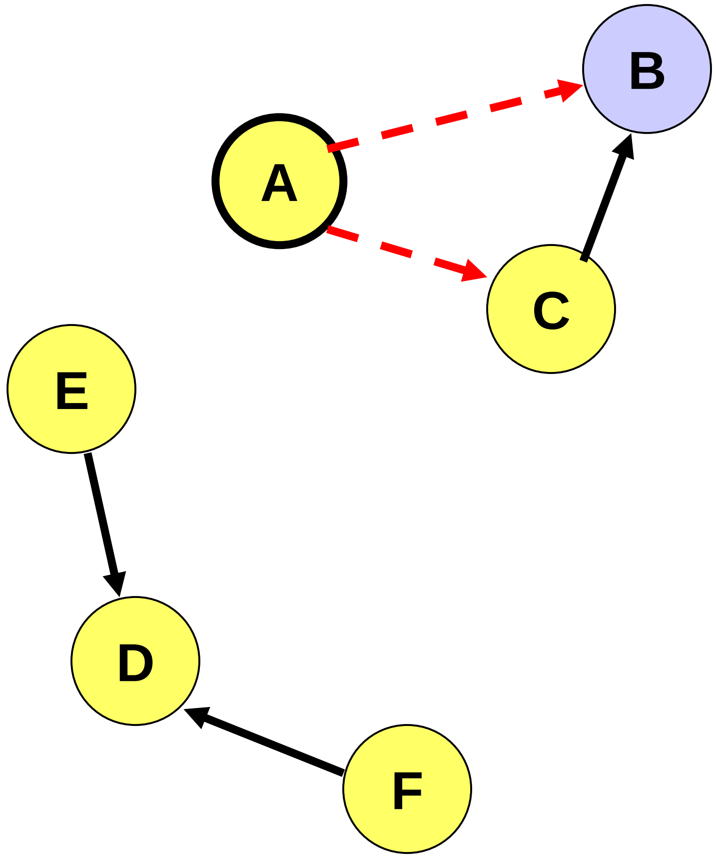
Partition Detection in TORA



TORA uses a modified partial reversal method

Node A has no outgoing links

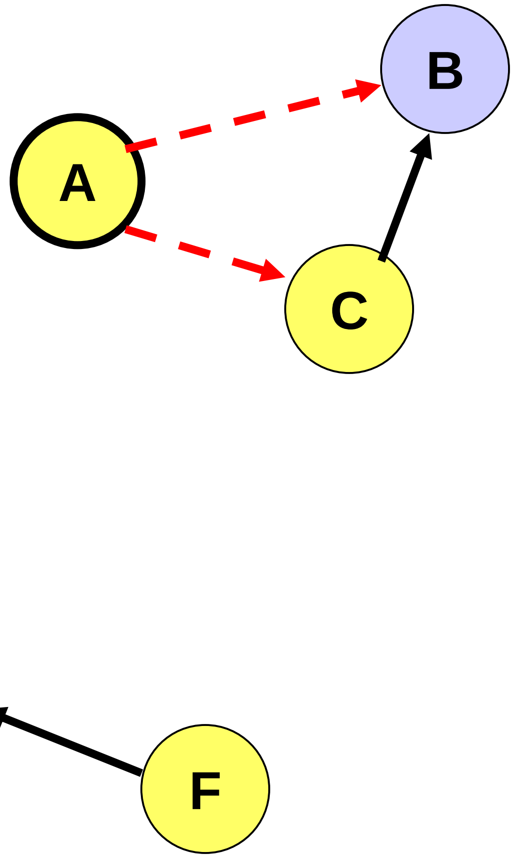
Partition Detection in TORA



TORA uses a modified partial reversal method

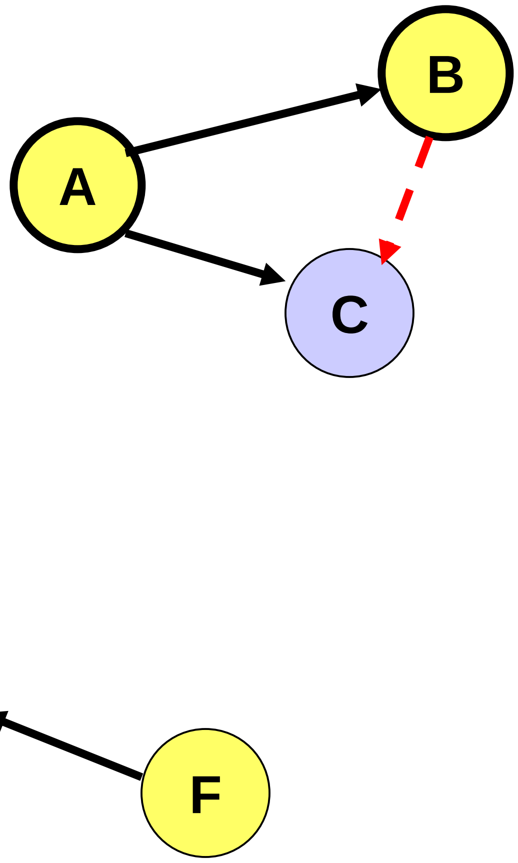
Node B has no outgoing links

Partition Detection in TORA



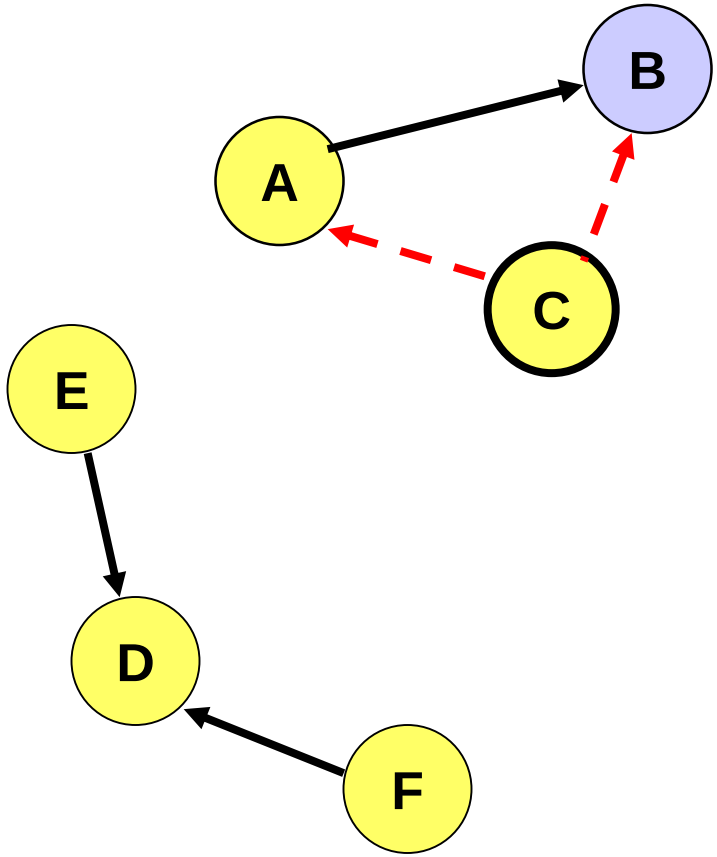
Node B has no outgoing links

Partition Detection in TORA



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

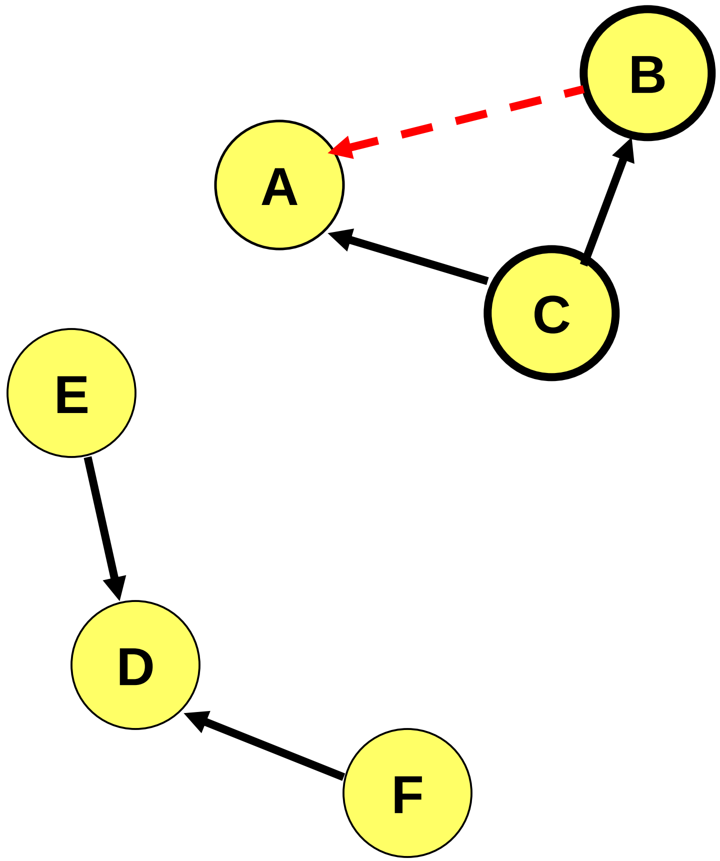
Partition Detection in TORA



Nodes A and B receive the reflection from node C

Node B now has no outgoing link

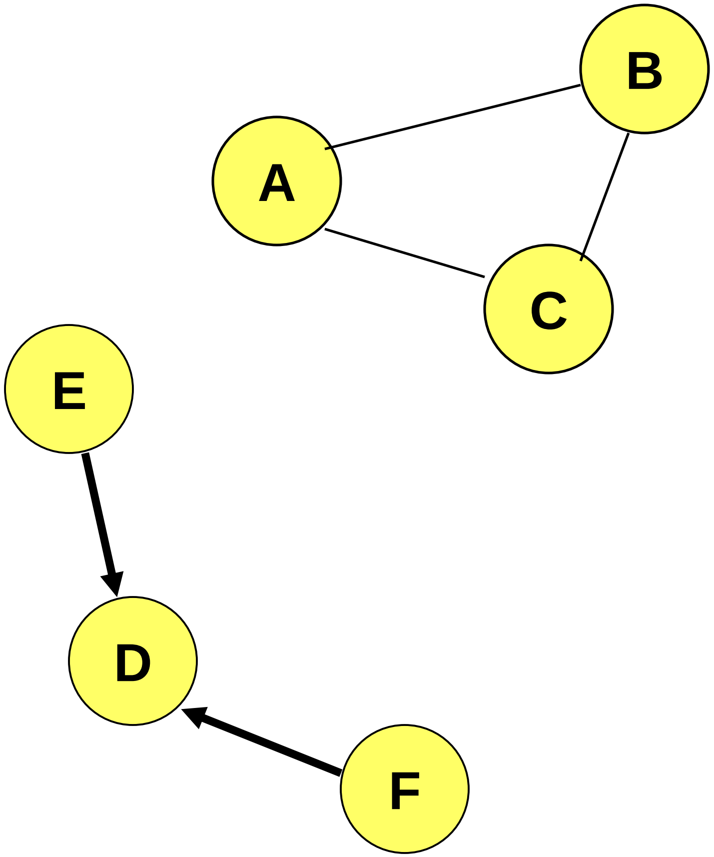
Partition Detection in TORA



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 looses 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

So far ...

- All nodes had identical responsibilities
- Some schemes propose giving special responsibilities to a subset of nodes
  - “Core” based schemes assign additional tasks to nodes belonging to the “core
  - Clustering schemes assign additional tasks to cluster “leaders”
- Not discussed further in this tutorial

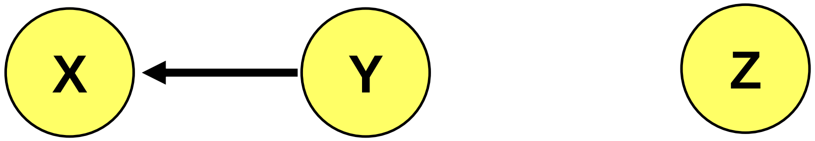
Destination-Sequenced Distance-Vector (DSDV) [Perkins94Sigcomm]

- Each node maintains a routing table which stores
  - next hop towards each destination
  - a cost metric for the path to each destination
  - a destination sequence number that is created by the destination itself
  - Sequence numbers used to avoid formation of loops

- Each node periodically forwards the routing table to its neighbors
  - Each node increments and appends its sequence number when sending its local routing table
  - This sequence number will be attached to route entries created for this node

Destination-Sequenced Distance-Vector (DSDV)

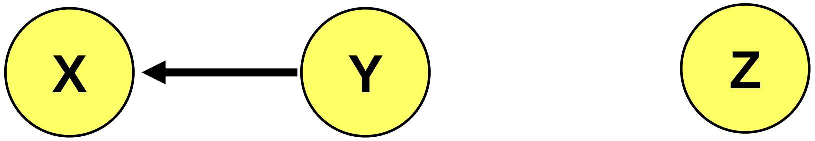
- Assume that node X receives routing information from Y about a route to node Z



- Let S(X) and S(Y) denote the destination sequence number for node Z as stored at node X, and as sent by node Y with its routing table to node X, respectively

Destination-Sequenced Distance-Vector (DSDV)

- Node X takes the following steps:



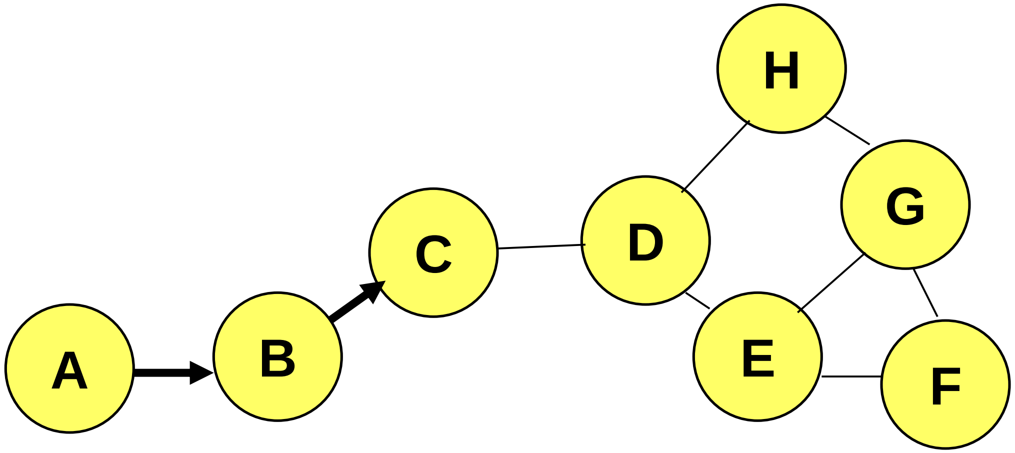
  - If $S(X) > S(Y)$, then X ignores the routing information received from Y
  - If $S(X) = S(Y)$, and cost of going through Y is smaller than the route known to X, then X sets Y as the next hop to Z
  - If $S(X) < S(Y)$, then X sets Y as the next hop to Z, and $S(X)$ is updated to equal $S(Y)$

Landmark Routing (LANMAR) for MANET with Group Mobility [Pei00Mobihoc]

- A *landmark* node is elected for a group of nodes that are likely to move together
- A *scope* is defined such that each node would typically be within the scope of its landmark node
- Each node propagates *link state* information corresponding only to nodes within it *scope* and *distance-vector* information for all *landmark* nodes
  - Combination of link-state and distance-vector
  - Distance-vector used for landmark nodes outside the scope
  - No state information for non-landmark nodes outside scope maintained

LANMAR Routing to Nodes Within Scope

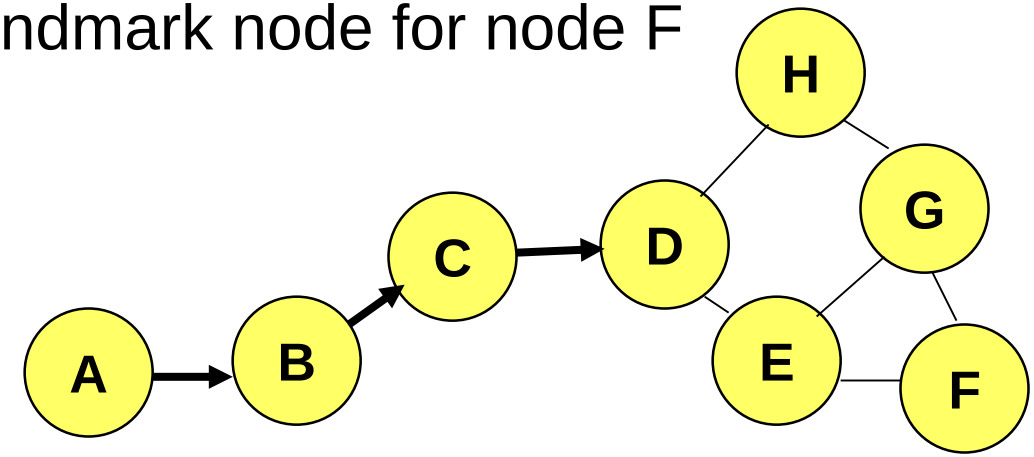
- Assume that node C is within scope of node A



- Routing from A to C: Node A can determine next hop to node C using the available link state information

LANMAR Routing to Nodes Outside Scope

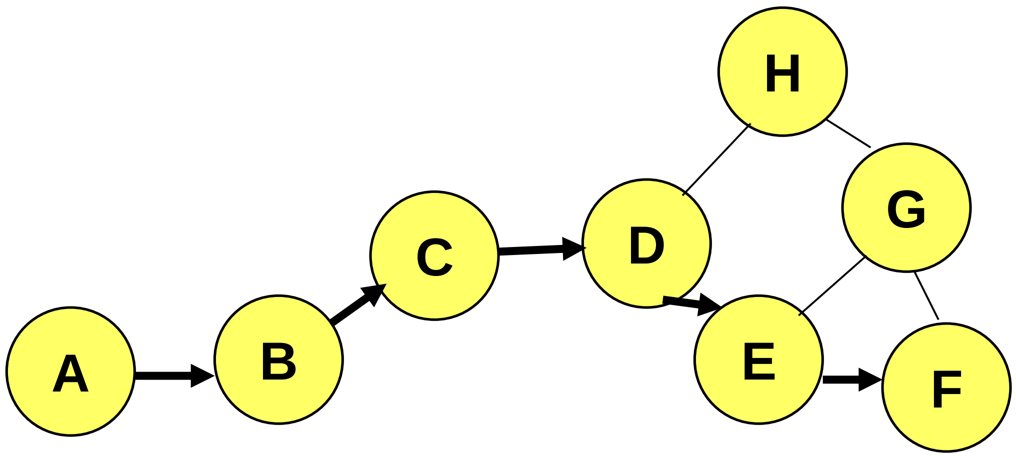
- Routing from node A to F which is outside A's scope
- Let H be the landmark node for node F



- Node A somehow knows that H is the landmark for C
- Node A can determine next hop to node H using the available distance vector information

LANMAR Routing to Nodes Outside Scope

- Node D is within scope of node F



- Node D can determine next hop to node F using link state information
- The packet for F may never reach the landmark node H, even though initially node A sends it towards H

- LANMAR scheme uses node identifiers as landmarks
- Anchored Geodesic Scheme [LeBoudec00] uses geographical regions as landmarks

Routing

- Protocols discussed so far find/maintain a route provided it exists

- Some protocols attempt to ensure that a route exists by
  - Power Control [Ramanathan00Infocom]
  - Limiting movement of hosts or forcing them to take detours [Reuben98thesis]

Power Control

- Protocols discussed so far find a route, on a *given* network topology
- Some researchers propose *controlling* network topology by transmission power control to yield network properties which may be desirable [Ramanathan00Infocom]
  - Such approaches can significantly impact performance at several layers of protocol stack
- [Wattwnhofer00Infocom] provides a distributed mechanism for power control which allows for local decisions, but guarantees global connectivity
  - Each node uses a power level that ensures that the node has at least one neighbor in each *cone* with angle $2\pi/3$

Some Variations

Power-Aware Routing [Singh98Mobicom,Chang00Infocom]

Define optimization criteria as a function of energy consumption. Examples:

- Minimize energy consumed per packet
- Minimize time to network partition due to energy depletion
- Maximize duration before a node fails due to energy depletion

Power-Aware Routing [Singh98Mobicom]

- Assign a weight to each link
- Weight of a link may be a function of energy consumed when transmitting a packet on that link, as well as the residual energy level
  - low residual energy level may correspond to a high cost
- Prefer a route with the smallest aggregate weight

Power-Aware Routing

Possible modification to DSR to make it power aware (for simplicity, assume no route caching):

- Route Requests aggregate the weights of all traversed links

- Destination responds with a Route Reply to a Route Request if
  - it is the first RREQ with a given (“current”) sequence number, or
  - its weight is smaller than all other RREQs received with the current sequence number

Preemptive Routing [Goff01MobiCom]

- Add some proactivity to reactive routing protocols such as DSR and AODV
- Route discovery initiated when it appears that an active route will break in the near future
- Initiating route discover *before* existing route breaks reduces discovery latency

Performance of Unicast Routing in MANET

- Several performance comparisons [Broch98Mobicom,Johansson99Mobicom,Das00Infocom,Das98ic3n]

- We will discuss performance issue later in the tutorial

Address Auto-Configuration

Address Auto-configuration

- Auto-configuration important for autonomous operation of an ad hoc network

- *IPv4* and *IPv6* auto-configuration mechanisms have been proposed
  - Need to be adapted for ad hoc networks

Auto-Configuration in Ad Hoc Networks

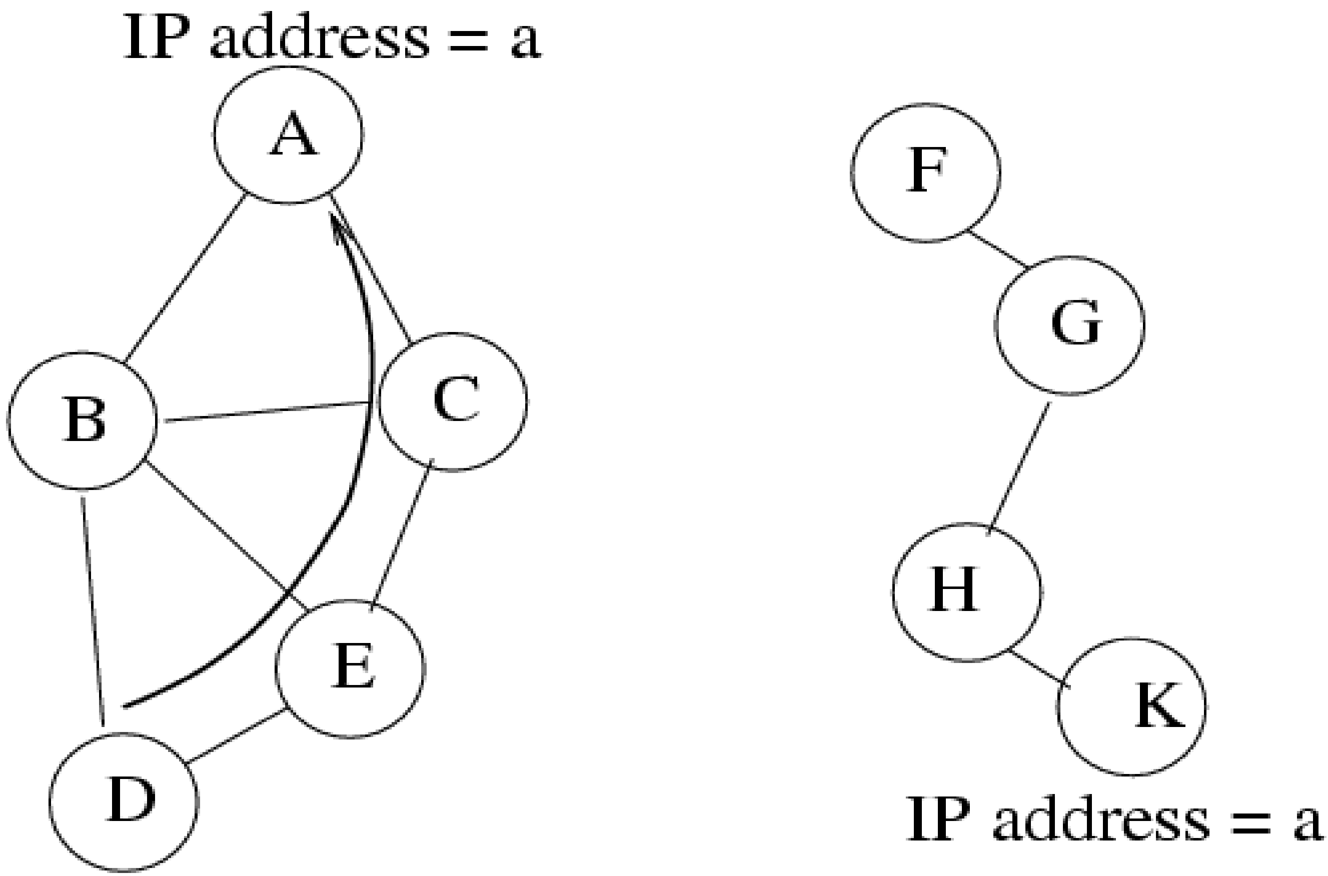
- Worst case network delays may be unknown, or highly variable

- Partitions may occur, and merge

Duplicate Address Detection in Ad Hoc Networks

- Several proposals
- One example [Perkins]:
  - Host picks an address randomly
  - Host performs route discovery for the chosen address
  - If a route reply is received, address duplication is detected

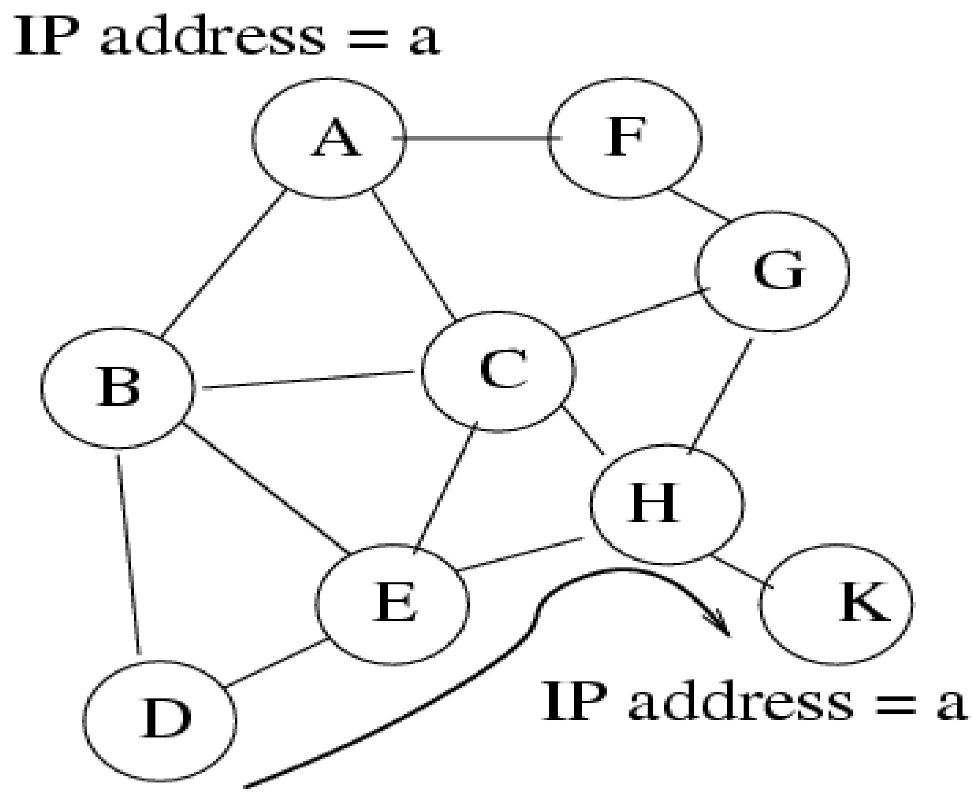
Example:
Initially Partitioned Network



D’s packets for address *a* routed to A

Merged Network

- Duplicate address detection (DAD) important To avoid misrouting



Strong DAD

- Detect duplicate addresses within *t* seconds

- Not possible to guarantee strong DAD in presence of unbounded delays
  - May occur due to partitions
  - Even when delays are bounded, bound may be difficult to calculate
    - Unknown network size

DAD

- Strong DAD impossible with unbounded delay

- How to achieve DAD ?

Design Principle

- If you cannot solve a problem

Change the problem

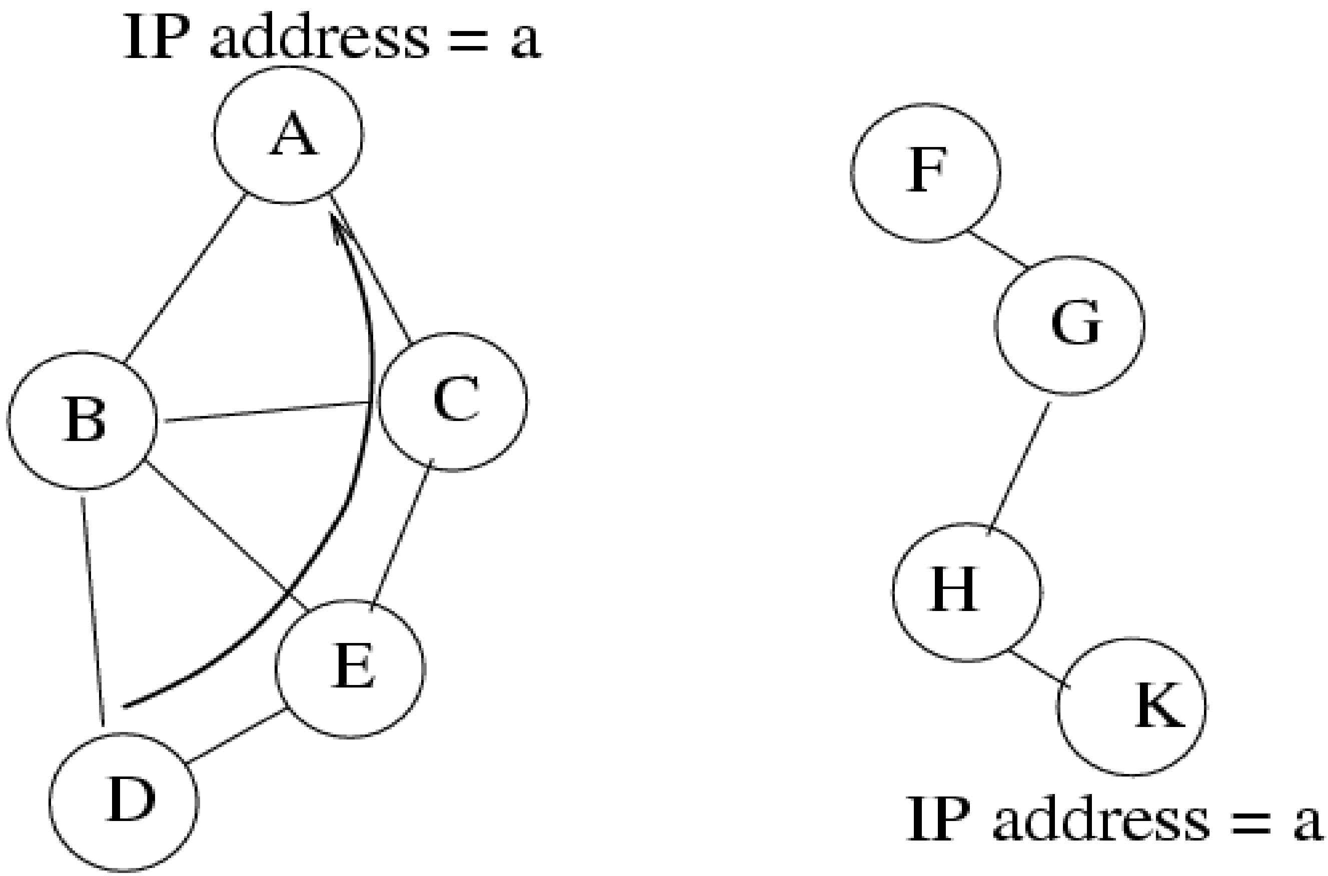
Weak DAD [Vaidya02MobiHoc]

Packets from a given host to a given *address*

should be routed to the same *destination*,

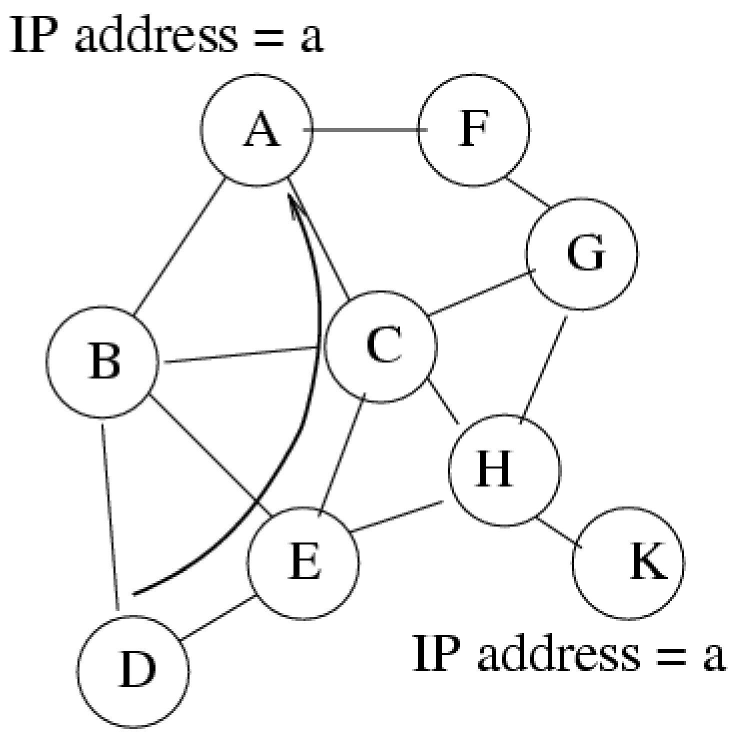
despite duplication of the *address*

Example:
Initially Partitioned Network



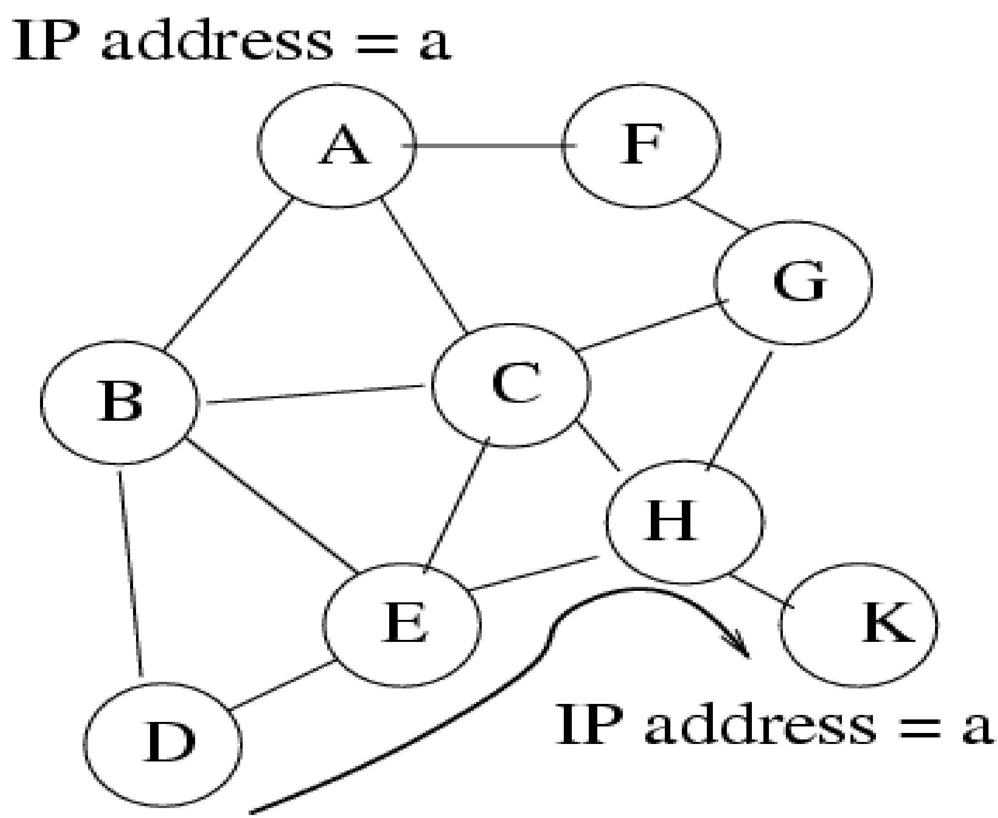
D's packets for address *a* routed to A

Merged Network: Acceptable Behavior with Weak DAD



Packets from D to address *a* still routed to host A

Merged Network: Unacceptable behavior



Packets from D to address *a* routed to host K instead of A

Weak DAD: Implementation

- Integrate duplicate address detection with route maintenance

Weak DAD with Link State Routing

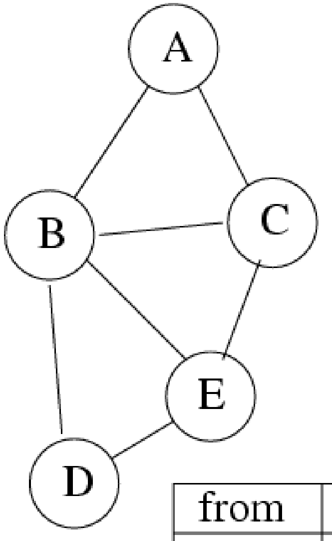
- Each host has a unique (with high probability) key
  - May include MAC address, serial number, …
  - May be large in size

- In all routing-related packets (link state updates) IP addresses tagged by keys
  - (IP, key) pair

Weak DAD with Link State Routing

- Address duplication not always detected
- Duplication detected before misrouting can occur
- Weak
  - ➔ Reliable, but potentially delayed, DAD

Link State Routing (LSR): Example



from	to	cost
IP_D	IP_E	2
IP_D	IP_B	10

link state packet
transmitted by D

Dest	Next Hop
IP_B	IP_B
IP_C	IP_E
IP_A	IP_B
IP_E	IP_E

Routing table
at node D

Weak DAD with LSR

Dest	Key	Next Hop
IP_B	K_B	IP_B
IP_C	K_C	IP_E
IP_A	K_A	IP_B
IP_E	K_E	IP_E

Routing table at node D

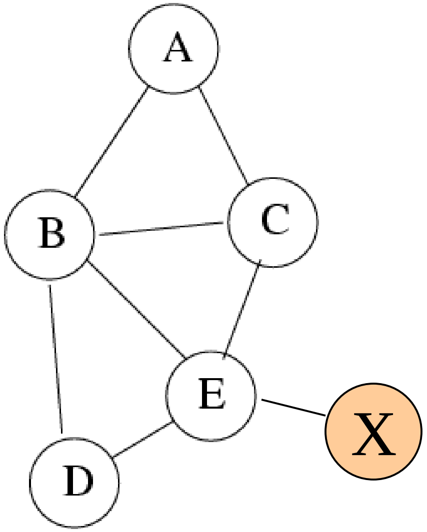
from	key	to	key	cost
IP_D	K_D	IP_E	K_E	2
IP_D	K_D	IP_B	K_B	10

link state packet transmitted by D

Weak DAD with LSR

Dest	Next Hop
IP_B	IP_B
IP_C	IP_E
IP_A	IP_B
IP_E	IP_E

Routing table at node D



Host X with key K_x joins and choose IP_A

(address duplication)

Weak DAD with LSR

Dest	Key	Next Hop
IP_B	K_B	IP_B
IP_C	K_C	IP_E
IP_A	K_A	IP_B
IP_E	K_E	IP_E

Routing table at node D

If host D receives a link state update containing (IP_A, K_x), host D detects duplication of address IP_A

Two pairs with identical IP address but distinct keys imply duplication

Just-in-Time DAD

- Duplication detected before routing tables could be mis-configured

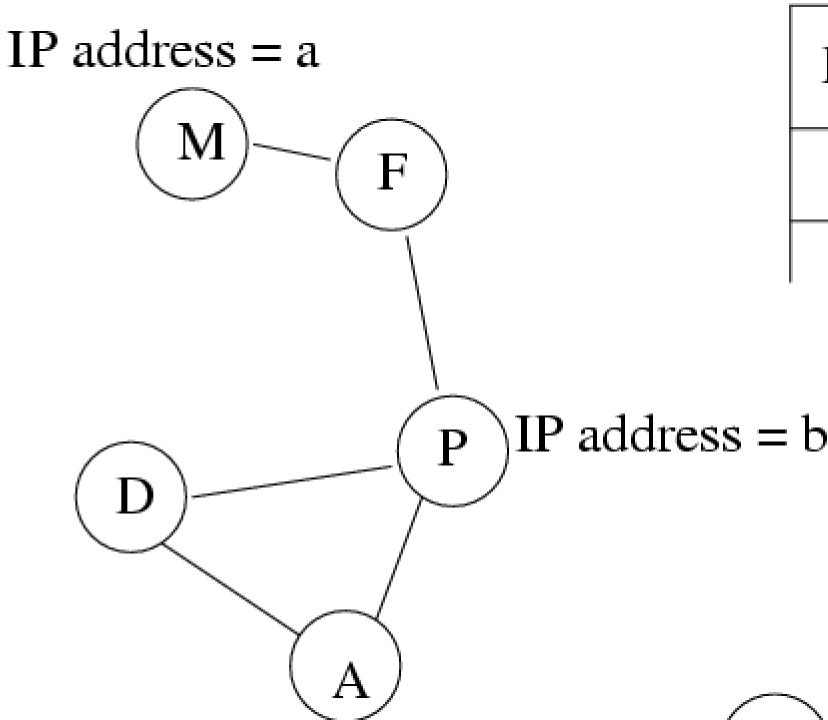
Higher Layer Interaction

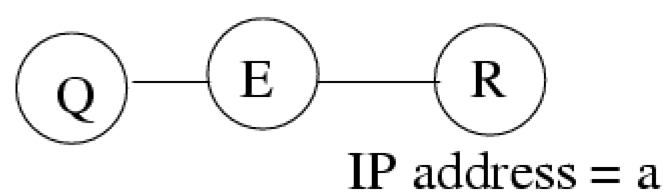
- Higher layers interaction may result in undesirable behavior

Example

An entry in node A's routing table

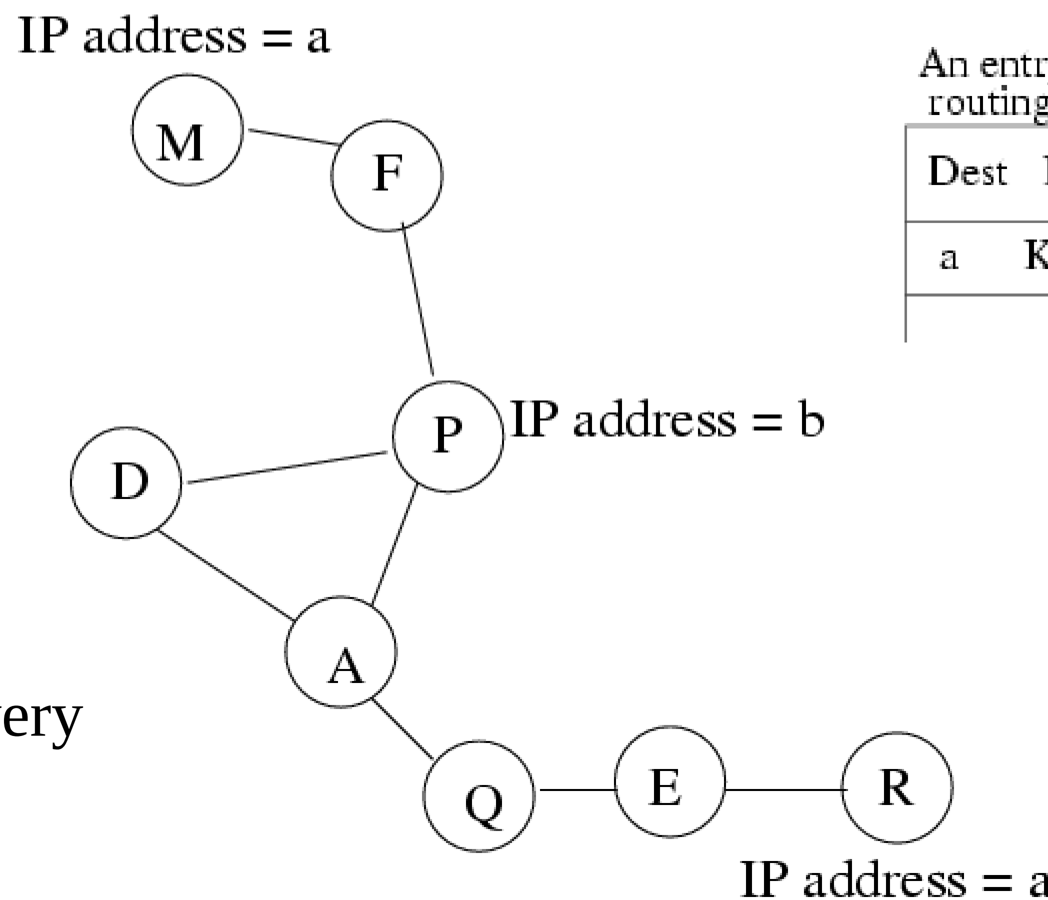
Dest	Key	Next Hop
a	K_M	b





Q discovers service *Foo* at address *a*

Example: Networks merge

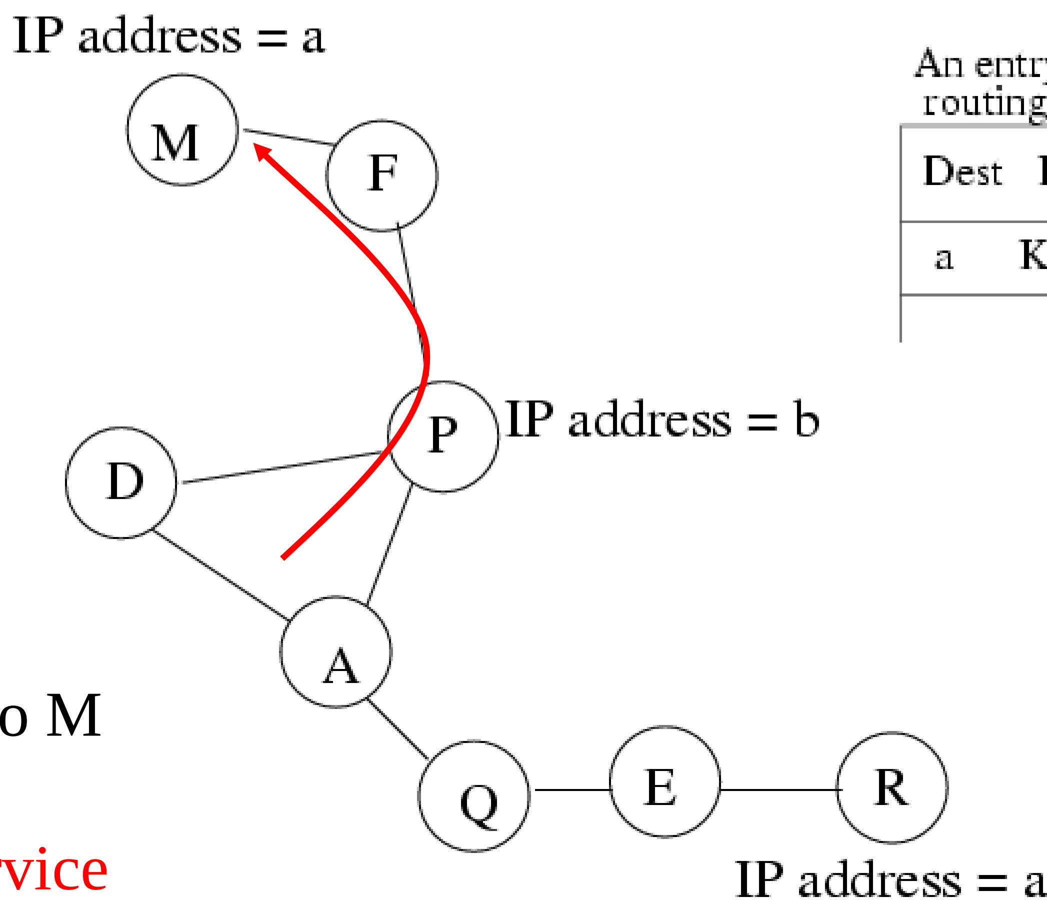


An entry in node A's routing table

Dest	Key	Next Hop
a	K_M	b

Node A performs service discovery for *Foo*, and learns from Q that *Foo* is available at address *a*

Example: Networks merge



An entry in node A's routing table

Dest	Key	Next Hop
a	K_M	b

Node A's packets to a are delivered to M

R provides service *Foo* not M

Enhanced Weak DAD

- If the status of host A above the network layer depends on state of host B

(State A → state B)

→ then network layer of host A should be aware of (IP, key) pairs known to B

Enhanced Weak DAD

- Works despite upper layer interaction

Weak DAD: Other Issues

- Duplicate MAC addresses within two hops of each other bad
  - Need a duplicate MAC address detection scheme
- Network layers performing unicasts using multicast/flooding
- Limited-time address leases
- DAD with other routing protocols
  - Possible. Paper also discusses DSR.

Summary

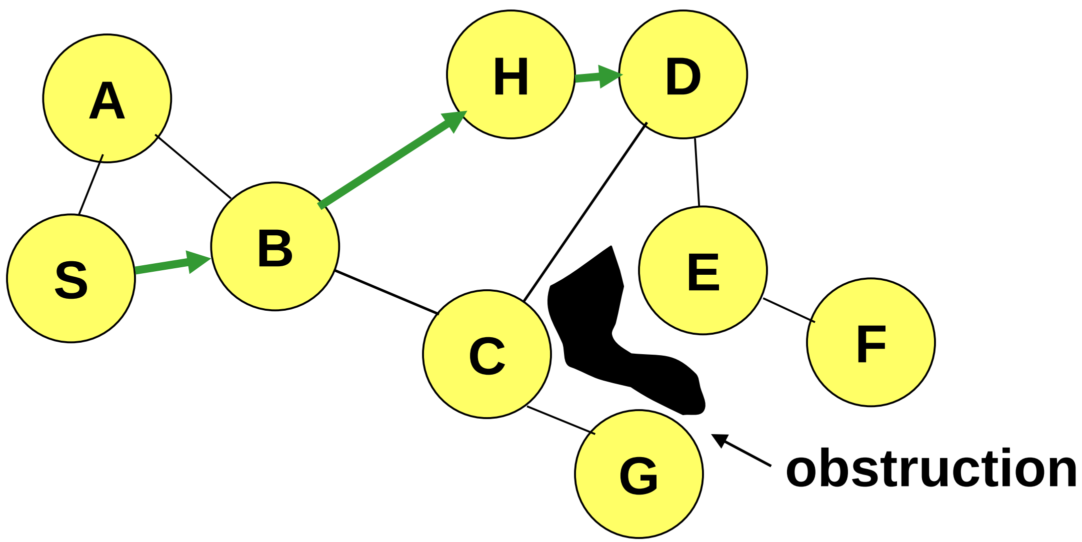
- Strong DAD – Not always possible
- Weak DAD feasible
  - Combines DAD with route maintenance
- Overhead of weak DAD
  - Expected to be low, but unknown presently

Detour

Routing Using Location Information

Geographic Distance Routing (GEDIR) [Lin98]

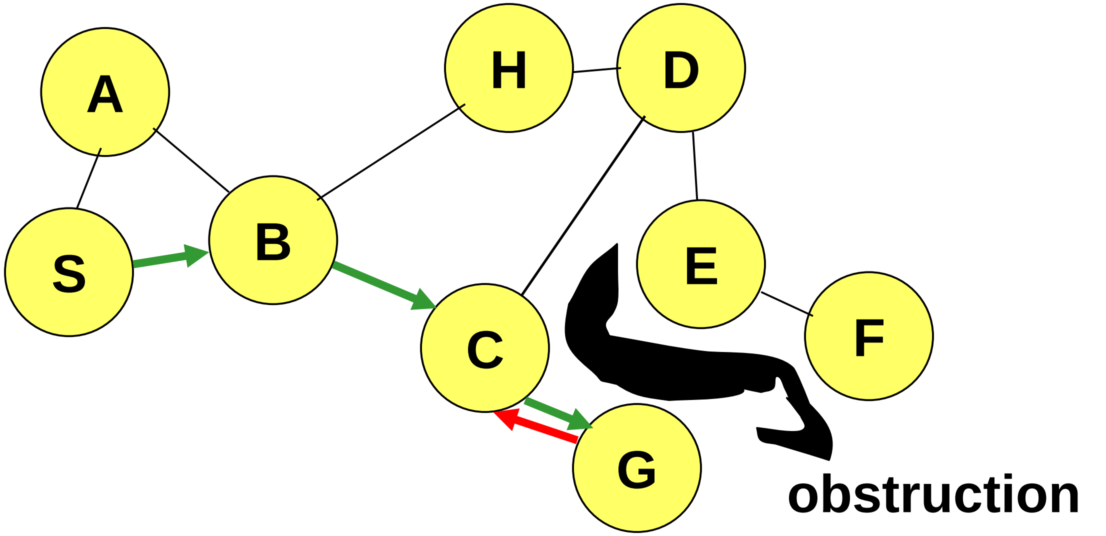
- Location of the destination node is assumed known
- Each node knows location of its neighbors
- Each node forwards a packet to its neighbor closest to the destination
- Route taken from S to D shown below



Geographic Distance Routing (GEDIR) [Stojmenovic99]

- The algorithm terminates when same edge traversed twice consecutively

- Algorithm fails to route from S to E
  - Node G is the neighbor of C who is closest from destination E, but C does not have a route to E



Routing with Guaranteed Delivery [Bose99Dialm]

- Improves on GEDIR [Lin98]
- Guarantees delivery (using location information) provided that a path exists from source to destination
- Routes around obstacles if necessary
- A similar idea also appears in [Karp00Mobicom]

End of Detour

Back to

Reducing Scope of the Route Request Flood

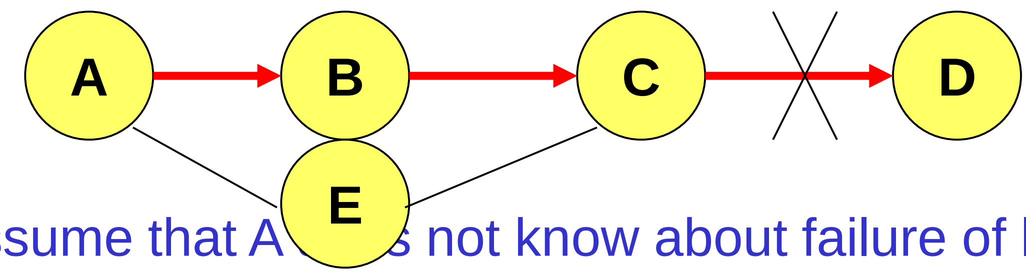
Query Localization [Castaneda99Mobicom]

- Limits route request flood without using physical information
- Route requests are propagated only along paths that are *close* to the previously known route
- The *closeness* property is defined without using physical location information

Why Sequence Numbers in AODV

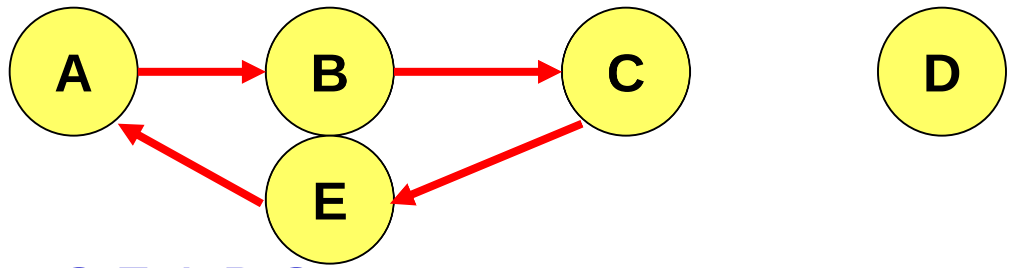
- To avoid using old/broken routes
  - To determine which route is newer

- 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



- Loop C-E-A-B-C

LAR Variations: Implicit Request Zone

- In the previous scheme, a route request explicitly specified a request zone
- Alternative approach: A node X forwards a route request received from Y if node X is deemed to be closer to the expected zone as compared to Y
- The motivation is to attempt to bring the route request physically closer to the destination node after each forwarding

Location-Aided Routing

- The basic proposal assumes that, *initially,* location information for node X becomes known to Y only during a route discovery
- This location information is used for a future route discovery
  - Each route discovery yields more updated information which is used for the next discovery

Variations

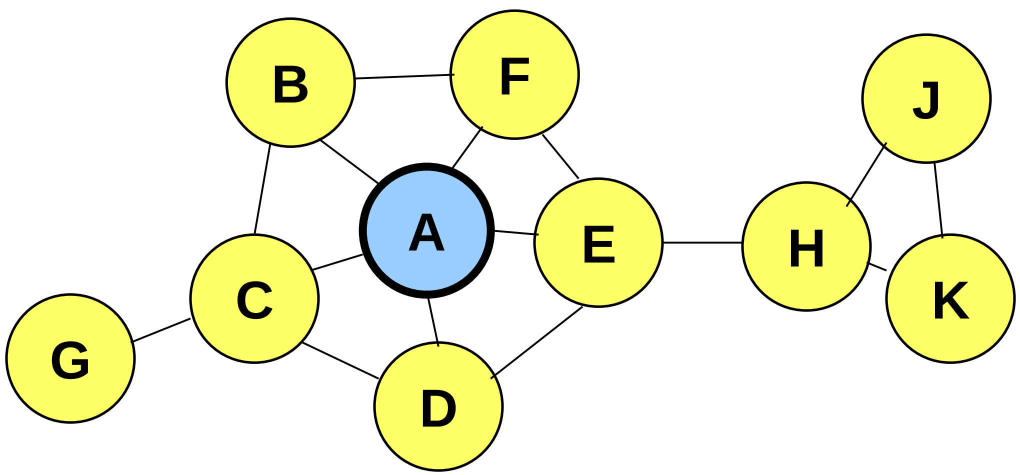
- Location information can also be piggybacked on any message from Y to X
- Y may also proactively distribute its location information
  - Similar to other protocols discussed later (e.g., DREAM, GLS)

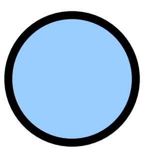
Optimized Link State Routing (OLSR)

- The overhead of flooding link state too high
  - Reduced by requiring fewer nodes to forward the information
- Broadcast from X forwarded by *multipoint relays only*
- Multipoint relays of node X are its neighbors such that each two-hop neighbor of X is a one-hop neighbor of at least one multipoint relay of X
  - Each node transmits its neighbor list in periodic beacons, so that all nodes can know their 2-hop neighbors, in order to choose the multipoint relays

Optimized Link State Routing (OLSR)

- Nodes C and E are multipoint relays of node A

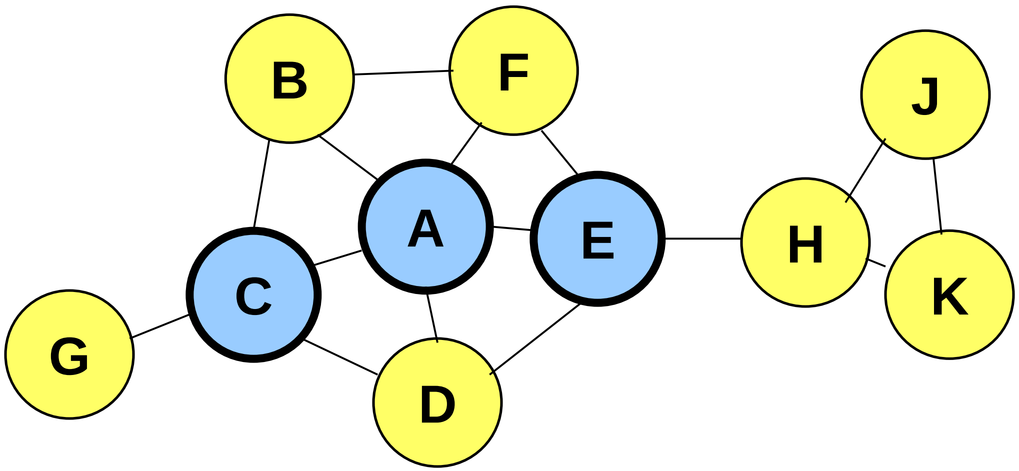


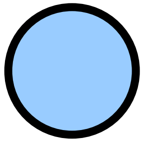


Node that has broadcast state information from A

Optimized Link State Routing (OLSR)

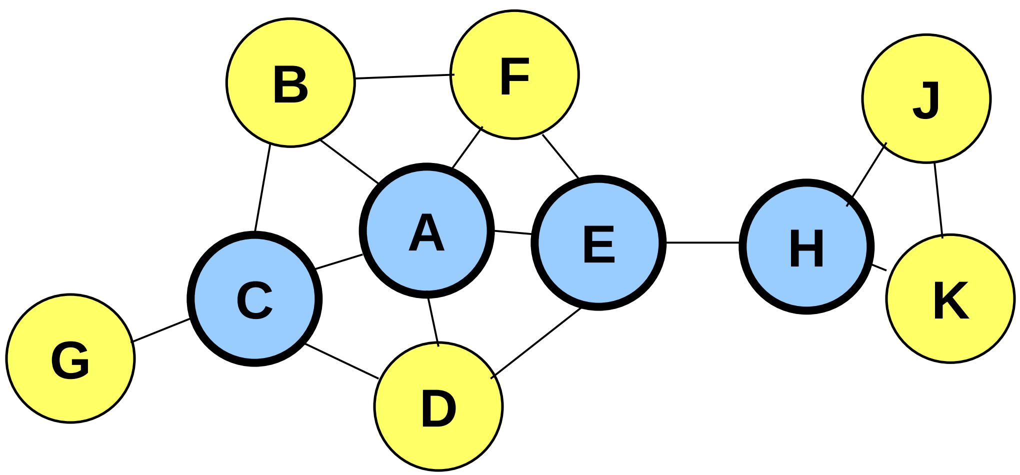
- Nodes C and E forward information received from A

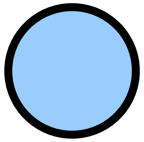




Optimized Link State Routing (OLSR)

- Nodes E and K are multipoint relays for node H
- Node K forwards information received from H
  - E has already forwarded the same information once





Node that has broadcast state information from A

OLSR

- OLSR floods information through the multipoint relays
- The flooded itself is for links connecting nodes to respective multipoint relays
- Routes used by OLSR only include multipoint relays as intermediate nodes