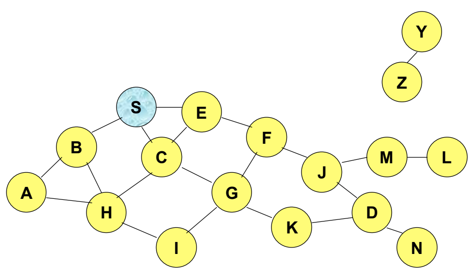
Ad Hoc On-Demand Distance Vector (AODV)

- DSR includes source routes in packet headers
- Resulting large headers can sometimes degrade performance
  - particularly when data contents of a packet are small
- AODV attempts to improve on 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

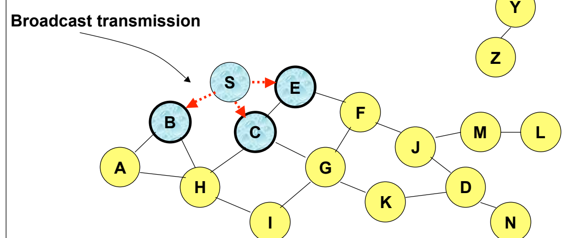
- Route Requests (RREQ) are forwarded in a manner similar to DSR
- When a node re-broadcasts a Route Request, it sets up a reverse path pointing towards the source
  - AODV assumes symmetric (bi-directional) links
- When the intended destination receives a Route Request, it replies by sending a Route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

Route Requests in AODV



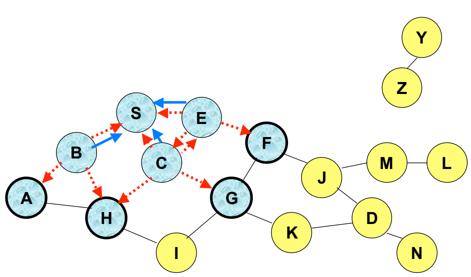
Represents a node that has received RREQ for D from S

Route Requests in AODV



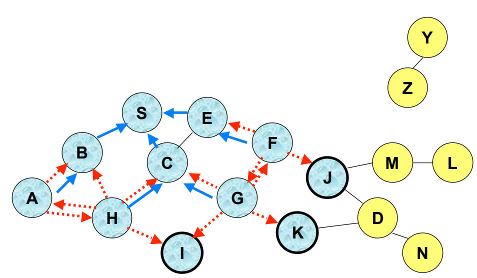
Represents transmission of RREQ

Route Requests in AODV



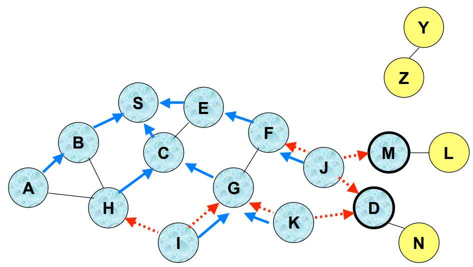
Represents links on Reverse Path

Reverse Path Setup in AODV



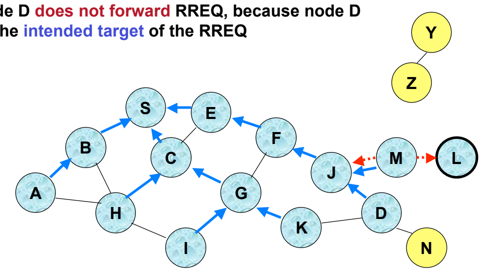
- Node C receives RREQ from G and H, but does not forward it again, because node C has already forwarded RREQ once

Reverse Path Setup in AODV

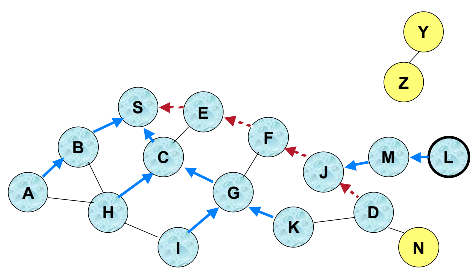


Reverse Path Setup in AODV

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



Route Reply in AODV



Represents links on path taken by RREP

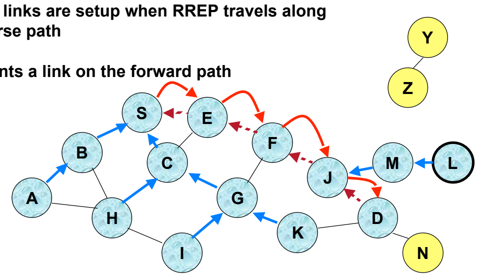
Route Reply in AODV

- An intermediate node (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

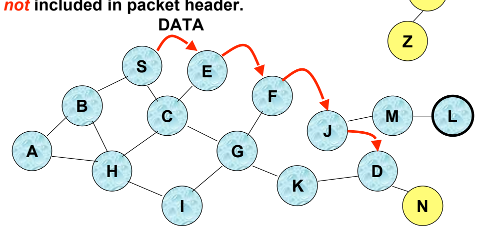
Represents a link on the forward path



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

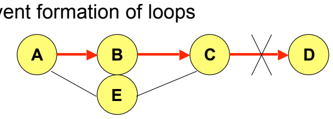
- Continuing from the previous slide …
- When node D receives the route request with destination sequence number N, node D will set its sequence number to N, unless it is already larger than N

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

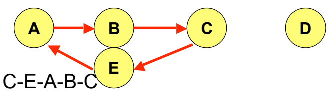
Why Sequence Numbers in AODV

- 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

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

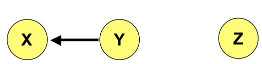
- Routes need not be included in packet headers
- Nodes maintain routing tables containing entries only for routes that are in active use
- At most one next-hop per destination maintained at each node
  - DSR may maintain several routes for a single destination
- Unused routes expire even if topology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

Destination-Sequenced Distance-Vector (DSDV)

- 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)

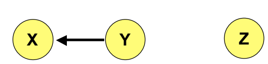
- Assume that node X receives routing information from Y about a route to node Z



- 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:

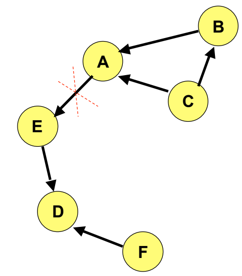


  - If S(X) > S(Y), then X ignores the routing information received from Y
  - If S(X) = S(Y), and cost of going through Y is smaller than the route known to X, then X sets Y as the next hop to Z
  - If S(X) < S(Y), then X sets Y as the next hop to Z, and S(X) is updated to equal S(Y)

Temporally-Ordered Routing Algorithm (TORA)

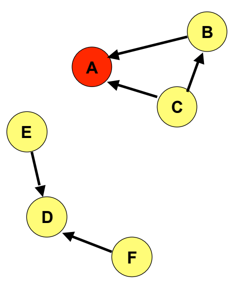
- TORA modifies the partial link reversal method to be able to detect partitions
- When a partition is detected, all nodes in the partition are informed, and link reversals in that partition cease

Partition Detection in TORA



DAG for destination D

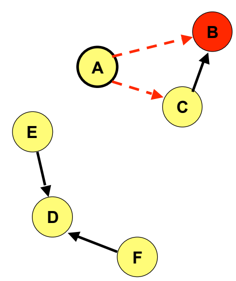
Partition Detection in TORA



TORA uses a modified partial reversal method

Node A has no outgoing links

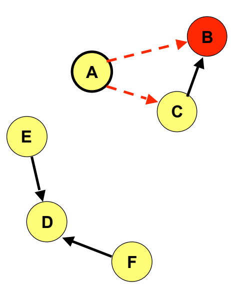
Partition Detection in TORA



TORA uses a modified partial reversal method

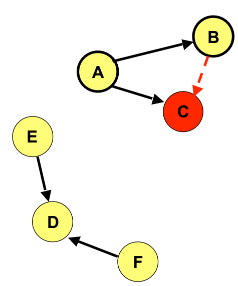
Node B has no outgoing links

Partition Detection in TORA



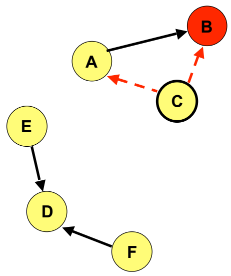
Node B has no outgoing links

Partition Detection in TORA



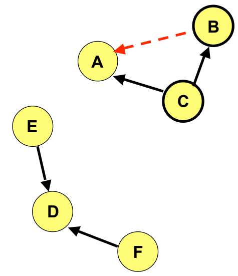
Node C has no outgoing links -- all its neighbor have reversed links previously.

Partition Detection in TORA



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

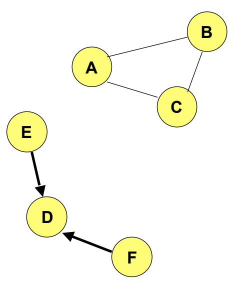
Partition Detection in TORA



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 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