CMPE 257: Wireless and Mobile Networking

Spring 2002
Week 4
Dynamic Source Routing (DSR) [Johnson96]

When node S wants to send a packet to D, but does not know a route to D, node S initiates a route discovery.

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

Broadcast transmission

[S]

Represents transmission of RREQ

[X,Y] Represents list of identifiers appended to RREQ
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
Nodes J and K both broadcast RREQ to node D
Since nodes J and K are hidden from each other, their transmissions may collide
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

RREP [S,E,F,J,D]

Represents RREP control message
Route Reply in DSR

RREP can be sent by reversing the route in RREQ only if links are guaranteed to be bi-directional.

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

Unless D already knows a route to S.

If a route discovery is initiated by D for a route to S, then the RREP is piggybacked on D’s RREQ.
Processing RREP

- Node S on receiving RREP, caches the route.
- 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 Route Discovery

When node S wants to send data to D, but does not know a valid route D.
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 F forwards Route Reply RREP [S,E,F,J,D], F learns route [F,J,D] to D.

When node E forwards Data [S,E,F,J,D] it learns route [E,F,J,D] to node D.

Nodes may also learn route when it overhears data.
Use of Route Caching

- When S learns that a route to D is broken, it uses another route from its local cache, if such a route to D exists in its cache; otherwise, S initiates route discovery.

- Node X on receiving a RREQ for some node D can send a RREP if X knows a route to 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
Route Caching:

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.
Assume that there is no link between D and Z. Route Reply (RREP) from node K limits flooding of RREQ.
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 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.

Adverse impact on TCP performance.
DSR: Advantages

- Routes maintained only between nodes who need to communicate.
  - Reduces overhead of route maintenance.
- Route caching can further reduce route discovery overhead.
- 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.
- Flood of route requests may potentially reach all nodes in the network.
- Care must be taken to avoid collisions between route requests propagated by neighboring nodes.
- Insertion of random delays before forwarding RREQ.
DSR: Disadvantages

- Increased contention if too many route replies come back due to nodes replying using their local cache.
  - "RREP" 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 RREP 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.
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 nodes, so that data packets do not have to contain routes. Like DSR, AODV is reactive.
AODV

- Route Requests (RREQ) are forwarded similarly to DSR.
  - When a node re-broadcasts a RREQ, it sets up a reverse path pointing towards the source.
  - AODV assumes symmetric (bi-directional) links.
- When the intended destination receives a RREQ, it replies by sending a RREP.
- RREPs travel along the reverse path set-up when RREQ is forwarded.
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

Represents links on Reverse Path
AODV Route Discovery: Observations

- RREQ contains source and destination IP address, current dest. seq. number, and broadcast id (incremented for every RREQ).
- Source IP + bcast id uniquely identifies RREQ: nodes don’t forward recently seen RREQs.
- RREQ processing: node creates reverse route table entry for RREQ source with TTL.
- If node has “unexpired” route to destination in its table with sequence number >= RREQ’s, replies to RREQ with Route Reply (RREP) back to source.
- Otherwise, broadcast RREQ onward.
Destination Sequence Number

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

Node’s own sequence number is monotonically increasing.

Sequence number is incremented after neighborhood topology change.
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
• Node D does not forward RREQ, because node D is the intended target of the RREQ
Route Reply in AODV

In intermediate node has current route to destination, responds to RREQ with RREP.

RREP contains source and destination IP, current sequence number, number of hops to destination.

- If destination, then destination seq. #.
- Else, node’s current record of destination’s seq. #.

Node receiving RREP sets up forward path to destination.

If multiple RREPs received, node forwards first one. Later RREPs discarded unless greater seq. # or smaller # of hops.
Route Reply Example

Represents links on path taken by RREP
Forward Path Setup in AODV

Forward links are setup when RREP travels along the reverse path

Represents a link on the forward path
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.

Routing table entry maintaining a forward path is purged if not used for 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

Link failures are propagated by means of Route Error messages, which also update destination sequence numbers.

- **RERR** lists destinations now unreachable.

If upstream node has neighbors as precursors for the affected destinations, it broadcasts RERR. Nodes receiving the RERR update cost to destination to infinity and forward RERR if needed.

Upon receiving RERR, source will initiate route discovery if still needs route.
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 $N$. 
Link Failure Detection

- **Hello** messages: neighbor nodes periodically exchange hello messages.
- Absence of hello message is used as an indication of link failure.
- Alternatively, failure to receive several MAC-level ACKs may be used as an indication of link failure.
Sequence Numbers in AODV

- To avoid using old/broken routes.
- To determine which route is newer.
Sequence Numbers in AODV

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. A receives the RREQ (say, via path C-E-A).

A will reply since it knows a route to D via B.

Results in a loop (for instance, C-E-A-B-C).
Optimization: Expanding Ring Search

RREQs are initially sent with small Time-to-Live (TTL) field, to limit their propagation.

DSR also includes a similar optimization.

If no RREP 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.
Flooding of Control Packets

How to reduce the scope of the route request flood?
Location-Aided Routing (LAR)

Ko98Mobicom

Exploits location information to limit scope of route request flood.
- Location information may be obtained using GPS.

Expected Zone: region expected to hold the current location of the destination.
- Expected region based on old location information, and knowledge of destination’s speed.

Route requests limited to a Request Zone that contains Expected Zone and location of sender node.
Expected Zone in LAR

\( X = \) last known location of node D, at time \( t_0 \)

\( Y = \) location of node D at current time \( t_1 \), unknown to node S

\( r = (t_1 - t_0) \times \) estimate of D’s speed
Request Zone in LAR

Network Space

Request Zone

A

B

S

r

X

Y

CMPE 257  Spring 2002
LAR

- Only nodes within request zone forward RREQs.
  - Node A does not forward RREQ, but node B does.
- Request zone explicitly specified in the RREQ.
- Each node must know its physical location to determine whether it is within the request zone.
If route discovery using smaller request zone fails to find a route, sender initiates another route discovery (after a timeout) using larger request zone.

Larger request zone may be the entire network.

Rest of route discovery protocol similar to DSR.
LAR Variations: Adaptive Request Zone

- Each node may modify the request zone included in the forwarded request.
- Modified request zone may be determined using more recent/accurate information, and may be smaller than the original request zone.
Adaptive Request Zones

- Request zone adapted by B
- Request zone defined by sender S
LAR Variations: Implicit Request Zone

In the previous scheme, RREQ explicitly specified request zone.

**Alternative approach:** node X forwards RREQ received from Y if X is deemed to be closer to expected zone as compared to Y.

The motivation is to attempt to bring the RREQ physically closer to the destination node after each forwarding.
Implicit Request Zone

RREQ includes position of D and distance of current node to D.
More on LAR…

Basic LAR assumes that, \textit{initially}, location information for X becomes known to Y only during route discovery.

This location information is used for future route discovery. Why?

Variations

Location information can also be piggybacked on any message from Y to X.

Y may also proactively distribute its location. Similar to other protocols (e.g., DREAM, GLS).
LAR Summary

Advantages:
- Reduces scope of route request flood.
- Reduces overhead of route discovery.

Disadvantages:
- Nodes need to know their physical locations.
- Choice of request zone.
So far ...

All protocols discussed so far perform some form of flooding.

Now we will consider protocols which try to avoid flooding.
Link Reversal Routing

- Try to keep minimum state at nodes.
- Instead of maintaining state to support shortest-path routing computation, keep only state for DAG rooted at destination.

DAGs:
- Loop-freedom and multiple routes.
- Reduce frequency and scope of route updates.

But, nodes don’t know their position (distance) relative to destination.
- Shortest-path computation not possible.
Link Reversal Algorithm

[Gafni81]
Link Reversal Algorithm

Maintain a directed acyclic graph (DAG) for each destination, with the destination being the *only sink*. This DAG is for destination node D.

Links are bi-directional; algorithm imposes logical directions on them.
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

Represents a link that was reversed recently
Link Reversal Algorithm

Now nodes B and G have no outgoing links

Represents a link that was reversed recently
Now nodes A and F have no outgoing links

Represents a link that was reversed recently
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 first packet is sent to a destination, the destination oriented DAG is constructed.

The initial construction results in flooding of control packets.
Link Reversal Algorithm

Previous algorithm is called a full reversal method since when a node reverses links, it reverses all its incoming links.

Partial reversal method [Gafni81]: 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

Represents a link that was reversed recently

Represents a node that has reversed links
Nodes E and F *do not* reverse links from node G. Now node B has no outgoing links.
Partial Reversal Method

Now node A has no outgoing links

Represents a link that was reversed recently
Partial Reversal Method

Now all nodes (except destination D) have outgoing links

Represents a link that was reversed recently
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 more efficient than full reversal method.
Link Reversal Methods:

Disadvantage

- Need 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.
TORA [Park97Infocom]

- Temporally-Ordered Routing Algorithm.
- 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
Node C has no outgoing links -- all its neighbor have reversed links previously.
Nodes A and B receive the reflection from node C
Node B now has no outgoing link
Node A has received the reflection from all its neighbors. Node A determines that it is partitioned from destination D.
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 D may disappear.

TORA maintains the DAG for D only if someone needs route to D.

Reactive behavior
Node Height

- Destination has height 0.
- Defines a node’s “location” relative to destination according to current DAG.
So far ...

All nodes had identical responsibilities.

Some schemes propose giving special responsibilities to a subset of nodes.

Even if all nodes are physically identical.

**Core-based** schemes are examples of such schemes.
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.
Optimized Link State Routing (OLSR) [Jacquet00ietf]

- Overhead of flooding link state information reduced by having fewer nodes forward the information.
- Broadcast from X only forwarded by its *multipoint relays (MPRs).*
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 know their 2-hop neighbors.

MPRs of X are 1-hop neighbors of X covering X’s 2-hop neighbors.
C and E are multipoint relays of A.

Node that has broadcast state information from A
Optimized Link State Routing (OLSR)

Nodes C and E forward information received from A.

Node that has broadcast state information from A
Optimized Link State Routing (OLSR)

E and K are multipoint relays for H.

K forwards information received from H.

E has already forwarded the same information once.
OLSR

- OLSR is proactive.
- It floods information through MPRs.
- Flooded information contains links connecting nodes to respective MPRs.
  - i.e., node sends info on nodes that selected it as their MPR.
  - Periodic HELLO messages inform nodes which other nodes selected it as their MPR.
- Routes used by OLSR only include multipoint relays as intermediate nodes.
Hybrid Protocols
Zone Routing Protocol combines:

- **Proactive protocol**: which pro-actively updates network state and maintains routes 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 “Hybridness”

- Limits scope of proactive procedure to a node’s *local neighborhood*.
- Limits scope of topology changes to local neighborhood.
- Reactive protocol executed for routes to destination far-away.
Routing Zone

All nodes within hop distance of at most $d$ from node $X$ are said to be in the routing zone (RZ) of $X$.

All nodes at hop distance exactly $d$ are said to be peripheral nodes of $X$’s routing zone.

Each node maintains its own RZ.
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**: Uses reactive 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 $d = 2$

S performs route discovery for D

→ Denotes route request
ZRP: Example with $d = 2$

S performs route discovery for D

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

Denotes route reply
ZRP: Example with $d = 2$

S performs route discovery for D

Denotes route taken by Data
References

- Charles Perkins, “Mobile IP”.
- Yongguang Zhang’s CS397 page (www.cs.utexas.edu/users/ygz/395T/).
- Nitin Vaidya’s tutorial on Ad Hoc Networks (Mobicom 2001).
- Charles Perkins, “Ad Hoc Networking”