Routing Protocols For Ad-hoc Networks

Parikshit Machwe  Neha Jha  Sunil Arora  Raghav Mittal
(9947105)  (9947110)  (9947112)  (9947114)

1 Introduction

Wireless communication between mobile users is becoming more popular than ever before. This is due to recent technological advances in laptop computers and wireless data communication devices, such as wireless modems and wireless LANs. This has led to lower prices and higher data rates, which are the two main reasons why mobile computing continues to enjoy rapid growth.

There are two distinct approaches of enabling wireless communication between two hosts. One is the infrastructured approach or the cellular approach and the other is the non-infrastructure or the ad-hoc approach. A cellular network consists of fixed gateways in each cell which themselves form a wired network. A mobile host communicates with a bridge in the network (called the base station) if it is within its communication radius. The mobile unit can move geographically while it is communicating. When it goes out of range of one base station, it connects with a new base station and starts communicating through it. This is called a handoff. Handoff is a major problem because there is a considerable delay between when one base station hands off the user to another base station. During this delay, packets could be lost.

In contrast to the infrastructure based networks, in ad hoc networks all nodes are mobile and can be connected dynamically in an arbitrary manner. All nodes of these networks behave as routers and take part in discovery and maintenance of routes to other nodes in the network. This form of networking is limited in range by the transmission ranges of the individual nodes’ transmission ranges and is typically smaller compared to the range of cellular networks. But ad hoc networks have their own advantages. These advantages include:

- On demand setup.
- Fault tolerance.
- Unconstrained connectivity.

Ad-hoc networks do not rely on any pre-established infrastructure and can therefore be deployed in places with no infrastructure. This is useful in disaster recovery situations and places with non-existing or damaged communication infrastructure where rapid deployment of a communication network is needed. Ad-hoc networks also allow the creation of temporary networks without engaging the services of pre-existing networks.

Because nodes are forwarding packets for each other, some sort of routing protocol is required to make the routing decisions. Before going into the routing protocols, we discuss the issues related to routing in ad-hoc networks.

1.1 Desired Characteristics of Ad-hoc Routing Protocols

The ad-hoc networks enjoy a very dynamic topology. The nodes are allowed to enter or leave the network whenever they wish. The limited transmission range of the nodes results in multi-hop paths to the destinations. Another characteristic of the nodes in an ad-hoc network is their limited CPU capacity, storage capacity, battery power and bandwidth.

The above restrictions directly imply demanding constraints on the routing protocols. Ad-hoc routing protocols are desired to have the following properties:

- **Distributed Operation**: The protocol should be distributed. It should not be dependent on a centralized controlling node. The reason is obvious. The nodes are allowed to enter/leave the network whenever they wish.
- **Loop-free**: To improve the overall performance, the routing protocol is required to supply loop-free routes so that the wastage of bandwidth and CPU capacity could be avoided.
- **Demand-based operation**: To minimize the control overhead in the network and thus not wasting the network resources more than necessary, the protocol should be reactive. This means that the protocol must react only when needed.
- **Multiple routes**: To reduce the number of reactions to topological changes and congestion, multiple routes could be used. If one route has become invalid, it is possible that another stored route could still be valid and thus saving the routing protocol from initiating another route discovery procedure.
- **Power conservation**: The nodes in an ad-hoc network, in general, have limited power resources. Hence, a power saving or sleep mode must be supported.

Other characteristics include the support for Quality of Service, support for unidirectional links (due to the na-
ture of radio medium) and security (again due to the nature of the medium).

In the next sections, we will discuss the proposed routing protocols for the ad-hoc networks.

1.2 Routing Protocols for Ad-hoc networks

The routing protocols for ad-hoc networks can be divided into two categories: table-driven and on-demand routing based on when and how the routes are discovered. In table-driven routing protocols consistent and up-to-date routing information to all nodes is maintained at each node whereas in on-demand routing the routes are created only when desired by the source host. We will discuss current table-driven protocols as well as on-demand protocols in the following sections.

2 Table driven routing protocols

In table-driven routing protocols each node maintains one or more tables containing routing information to every other node in the network. All nodes update these tables so as to maintain a consistent and up-to-date view of the network. When the network topology changes the nodes propagate update messages throughout the network in order to maintain consistent and up-to-date routing information about the whole network. These routing protocols differ in the method by which the topology change information is distributed across the network and the number of necessary routing-related tables.

2.1 Destination Sequenced Distance Vector (DSDV) Protocol

The Destination-Sequenced Distance-Vector (DSDV) Routing Algorithm [1] is based on the idea of the classical Bellman-Ford Routing Algorithm with certain improvements.

The DSDV protocol requires each mobile station to advertise, to each of its current neighbors, its own routing table. The entries in this list may change fairly dynamically over time, so the advertisement must be made often enough to ensure that every mobile computer can almost always locate every other mobile computer of the collection. In addition, each mobile computer must agree to relay data packets to other computers upon request. In this way a mobile computer may exchange data with any other mobile computer in the group even if the target of the data is not within range for direct communication. The data broadcast by each mobile computer will contain its new sequence number and the following information for each new route:

- The destination’s address;
- The number of hops required to reach the destination; and
- The sequence number of the information received regarding that destination, as originally stamped by the destination;

The routing table includes a sequence number created by the transmitter. Routes with more recent sequence numbers are always preferred as the basis for making forwarding decisions, but not necessarily advertised. Of the paths with the same sequence number, those with the smallest metric will be used. By the natural way in which the routing tables are propagated, the sequence number is sent to all mobile computers which may each decide to maintain a routing entry for that originating mobile computer. Routes received in broadcasts are also advertised by the receiver when it subsequently broadcasts its routing information; the receiver adds an increment to the metric before advertising the route, since incoming packets will require one more hop to reach the destination.

One of the most important parameters to be chosen is the time between broadcasting the routing information packets. However, when any new or substantially modified route information is received by a Mobile Host, the new information will be retransmitted soon, effecting the most rapid possible dissemination of routing information among all the cooperating Mobile Hosts. This quick rebroadcast introduces a new requirement for our protocols to converge as soon as possible.

Mobile Hosts cause broken links as they move from place to place. The broken link may be detected by the layer2 protocol, or it may instead be inferred if no broadcasts have been received for a while from a former neighbor. A broken link is described by a metric of INFINITY. When a link to a next hop has broken, any route through that next hop is immediately assigned an INFINITY metric and assigned an updated sequence number. Since this qualifies as a substantial route change, such modified routes are immediately disclosed in a broadcast routing information packet. Building information to describe broken links is the only situation when the sequence number is generated by any Mobile Host other than the destination Mobile Host. Sequence numbers defined by the originating Mobile Hosts are defined to be even numbers, and sequence numbers generated to indicate INFINITY metrics are odd numbers. When a node receives an INFINITY metric, and it has a later sequence number with a finite metric, it triggers a route update broadcast to disseminate this important news about that destination.

In order to reduce the amount of information carried in these packets, two types will be defined. One will carry all the available routing information, called a full dump. The other type will carry only information changed since the last full dump, called an incremental. By design, an incremental routing update should fit in one network protocol data unit (NPDU). The full dump will most likely require multiple NPDUs. Full dumps can be transmitted relatively infrequently when no movement of Mobile Hosts is occurring. When movement becomes frequent, and the size of
an incremental approaches the size of a NPDU, then a full dump can be scheduled (so that the next incremental will be smaller).

When a Mobile Host receives new routing information (usually in an incremental packet), that information is compared to the information already available from previous routing information packets. Any route with a more recent sequence number is used. Routes with older sequence numbers are discarded. A route with a sequence number equal to an existing route is chosen if it has a "better" metric, and the existing route discarded, or stored as less preferable. The metrics for routes chosen from the newly received broadcast information are each incremented by one hop. Newly recorded routes are scheduled for immediate advertisement to the current Mobile Host's neighbors. Routes which show an improved metric are scheduled for advertisement at a time which depends on the average settling time for routes to the particular destination under consideration.

Timing skews between the various Mobile Hosts are expected. It could turn out that a particular Mobile Host would receive new routing information in a pattern which causes it to consistently change routes from one next hop to another, even when the destination Mobile Host has not moved. A Mobile Host could conceivably always receive two routes to the same destination, with a newer sequence number, one after another (via different neighbors), but always get the route with the worse metric first. This will lead to a continuing burst of new route transmittal upon every new sequence number from that destination.

One solution is to delay the advertisement of such routes, when a Mobile Host can determine that a route with a better metric is likely to show up soon. The route with the later sequence number must be available for use, but it does not have to be advertised immediately unless it is a route to a destination which was previously unreachable. The wait time could be predicted from the history of the updates received.

Another feature of DSDV that we would like to mention is that if the notification of which other mobile computers are accessible by a particular computer done at the MAC Layer, then DSDV will work with whatever Network Layer protocol used.

### 2.2 Global State Routing (GSR)

GSR [2] is functionally similar to the Link State Routing in that it maintains a topology map at each node.

The key is the way in which routing information is disseminated. In LSR, link state packets are generated and flooded into the network whenever a node detects a topology change. In GSR, link state packets are not flooded. Instead, nodes maintain a link state table based on the up-to-date information received from neighboring nodes, and periodically exchange it with their local neighbors only (no flooding). Through this exchange process, the table entries with larger sequence numbers replace the ones with smaller sequence numbers. The GSR periodic table exchange resembles the vector exchange in DSDV where the distances are updated according to the time stamp or sequence number assigned by the node originating the update. In GSR (like in LS) link states are propagated, a full topology map is kept at each node, and shortest paths are computed using this map.

In a wireless environment, a radio link between mobile nodes may experience frequent disconnects and reconnects. The LS protocol releases a link state update for each such change, which floods the network and causes excessive overhead. GSR avoids this problem by using periodic exchange of the entire topology map, greatly reducing the control message overhead.

The drawbacks of GSR are the large size update message which consumes considerable amount of bandwidth and the latency of the link state change propagation, which depends on the update period. This is where the Fisheye technique comes to help, by reducing the size of update messages without seriously affecting routing accuracy.

### 2.3 Fisheye State Routing (FSR) scheme

The fisheye technique was proposed to reduce the size of information required to represent graphical data. The eye of a fish captures with high detail the pixels near the focal point. The detail decreases as the distance from the focal point increases. In routing, the fisheye approach translates to maintaining accurate distance and path quality information about the immediate neighborhood of a node, with progressively less detail as the distance increases. The Fisheye State Routing (FSR) [2] scheme is built on top of the Global State Routing scheme.

Figure 1 illustrates the application of fisheye in a mobile, wireless network. The circles with different shades of grey define the fisheye scopes with respect to the center node (node 11). The scope is defined as the set of nodes that can be reached within a given number of hops. In our case, three scopes are shown for 1, 2 and 3 hops respectively. Nodes are color coded as black, grey and white accordingly.

The reduction of update message size is obtained by using different exchange periods for different entries in the routing table. More precisely, entries corresponding to nodes within the smaller scope are propagated to the neighbors with the highest frequency. The rest of the entries are sent out at a lower frequency. As a result, a considerable fraction of link state entries are suppressed, thus reducing the message size. This strategy produces timely updates from near stations, but creates large latencies that from stations afar. However the imprecise knowledge of the best path to a distant destination is compensated by the fact that the route becomes progressively more accurate as the packet gets closer to destination.

FSR scales well to large networks, by keeping link state exchange overhead low without compromising route computation accuracy when the destination is near. By retaining a routing entry for each destination, FSR avoids the
extra work of finding the destination (as in on demand routing) and thus maintains low single packet transmission latency. As mobility increases, routes to remote destinations become less accurate. However, when a packet approaches its destination, it finds increasingly accurate routing instructions as it enters sectors with a higher refresh rate.

3 On-demand Routing Protocols

For very dynamic topologies, proactive table-driven protocols can introduce a large overhead in bandwidth and energy consumption on the network. Reactive protocols trade off this overhead with increased delay. A route to a destination is established when it is needed based on an initial discovery between the source and the destination, and hence these protocols are termed as on-demand routing protocols.

3.1 Cluster based Routing Protocols

The protocol divides the nodes of the ad hoc network into a number of overlapping or disjoint clusters in a distributed manner. A cluster consists of a group of nodes with one of them elected as a cluster head. Inter-cluster routes are discovered dynamically using the cluster membership information kept at each cluster head. By clustering nodes into groups, the protocol efficiently minimizes the flooding traffic during route discovery and speeds up this process as well. Furthermore, the protocol takes into consideration of the existence of uni-directional links and uses these links for both intra-cluster and inter-cluster routing.

The protocol can be described in three sections mainly Cluster formation, maintenance and routing considerations.

3.1.1 Cluster formation

The Cluster Formation algorithm is a simple "lowest ID" clustering algorithm in which the node with a lowest ID is elected as the Cluster Head. When a node comes up, it enters the "undecided" state, starts a timer and broadcasts a Hello message. When a cluster-head gets this hello message it responds with a triggered hello message immediately. When the undecided node gets this message it sets its state to "member". If the undecided node times out, then it makes itself the cluster-head if it has bidirectional link to some neighbor otherwise it remains in undecided state and repeats the procedure again. A node uses the information obtained from the HELLO message for Cluster Formation. A node that has the lowest ID among all its bi-directionally linked neighbors (a node that has given up the role as a Cluster Head is not counted, i.e. a cluster member node) is chosen to as the cluster head. The new Cluster Head will change the first field in its subsequently broadcast HELLO messages from "undecided" to "cluster head" thereafter. A cluster head regards all the nodes it has bi-directional links to as its member nodes. A node regards itself as a member node for a particular cluster if it has a bi-directional link to the corresponding cluster head. Note that a member node may hear from several cluster heads and therefore have several host clusters. Each node maintains a neighbor table. Each entry contains the ID\(^*\) of a neighbor that it has connectivity with and the status of that link (uni- or bi-directional) and the role of the neighbor (a leader or a member). Each node broadcasts its Neighbor Table periodically in a HELLO message as shown below every HELLO_INTERVAL.

<table>
<thead>
<tr>
<th>My_Own_ID</th>
<th>My_Member_ship_Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Cluster Head/Cluster member/Undecided)</td>
</tr>
<tr>
<td>Neighbor Table</td>
<td></td>
</tr>
</tbody>
</table>

Upon receiving a HELLO message from its neighbor B, node A modifies its own Neighbor Table as follows:

1. It checks if B is already in the Neighbor Table, if not, it adds one entry for B.

2. If B’s Neighbor Table contains A, A marks the link to B as bi-directional in the relevant entry.

3. If B is cluster head, marks B as a cluster head in the entry.

Each entry in the Neighbor Table is associated with a
timer. A table entry will be removed if a HELLO message from the entry’s node is not received for a time bound.

### 3.1.2 Cluster Maintenance

In order to avoid frequent changes in cluster heads, following rules are followed for changing cluster head

1. A non-cluster head never challenges the status of an existing cluster head, i.e. if X is a non-cluster head node with a bi-directional link to cluster head Y, X does not become a cluster head even if it has an ID lower than Y’s.

2. Only when two cluster heads move next to each other (i.e. there is a bi-directional link between them), one of them loses the role of cluster head.

A node X is removed from a host cluster either because it loses the bi-directional links to the cluster head, or because of node failures. In either case, the cluster head and node X will time out the corresponding entries in their Neighbor Table so that the cluster information will be updated.

A member node is added to a new cluster when it discovers a bi-directional link to the respective cluster head even if the cluster head has a higher ID, which is consistent with rule 1. The node will know its new host cluster and the new host cluster head will know its new member through the updated Neighbor Table.

A cluster-head keeps information about the members of its cluster and also maintains a cluster adjacency table that contains information about the neighboring clusters. For each neighbor cluster, the table has entry that contains the gateway through which the cluster can be reached and the cluster-head of the cluster. It also contains the state of the link (uni or bi). Cluster X and cluster Y are said to be bi-directionally linked, if any node in cluster X is bi-directionally linked to another node in cluster Y, or if there is a pair of opposite uni-directional links between any 2 nodes in cluster X and cluster Y respectively. Cluster X is said to be uni-directionally linked to cluster Y if any node in cluster X is uni-directionally linked to any other node in cluster Y. Y is called X’s upstream uni-directionally linked neighboring cluster.

The adjacency table is updated by the periodic HELLO message a node hears. This table is periodically sent to the member node’s host cluster heads. A cluster head uses its members’ Cluster Adjacency table to construct its own cluster adjacency table.

### 3.1.3 Routing

In CBRP, routing is done using source routing. Therefore, it can be viewed as consisting of 3 phases: route discovery, packets routing and stale route removal. Cluster structure is exploited to minimize the flooding traffic during route discovery phase.

To perform route discovery, the source node S floods a Route Request Packet (RREQ) but only to the neighboring cluster-heads, with a recorded source route listing only itself initially. Any node that forwards this packet will append its own ID in this RREQ. Each node forwards a RREQ packet only once and it never forwards it to a node that has already appeared in the recorded route.

On receiving the request a cluster-head checks to see if the destination is in its cluster. If yes, then it sends the request directly to the destination else it sends it to all its adjacent cluster-heads. When the destination receives the request packet, it replies back with the route that had been recorded in the request packet. The recorded route gives the complete information about the SEQUENCE OF CLUSTERS source should traverse in order to reach destination D. Each intermediate cluster head also modifies the recorded route in the Route Reply packet to optimize the recorded route as much as possible using its knowledge of the cluster topology and inter-cluster gateway information. An example of such optimization is to connect two gateway nodes by an intra-cluster link that does not go through the cluster head.

All source routes learned by a node are kept in a Route Cache, which is used to further reduce the cost of Route Discovery. When a node wishes to send a packet, it examines its own Route Cache and performs route discovery only if no suitable source route is found in its cache.

A source route is removed from S if the network topology has changed such that S can no longer use the route to D because a hop along the route no longer works. If S still wants to communicate with D, it can invoke Route Discovery again to find a new route.

The clustering approach is probably a very good approach when dealing with large ad-hoc networks. The solution is more scalable than the other protocols, because it uses the clustering approach that limits the number of messages that need to be sent. CBRP also has the advantage that it utilizes unidirectional links.

### 3.2 Dynamic Source Routing Protocol

Key terminologies: Route Request Packet: This packet is used in the route discovery process. The packet contains the address of the source and the destination, and a unique identification number.

Route Record: It is that part of the route request packet which contains the host information which the route request packet traverses hop by hop.

Route Reply Packet: This packet is generated either by an intermediate node or by the intended destination on receipt of a route request packet, as the case may be.

The Dynamic source routing protocol is a major on-demand routing protocol in adhoc networks. This protocol has two distinct phases:

1. Route Discovery Phase
2. Route Maintenance Phase
Every node of the network maintains a routing cache which contains the source routes known by the node. As and when the knowledge of new routes pour into, it is added to the routing cache.

Whenever a node wants to send a packet to a destination, it looks into its routing cache for an available route. If an unexpired route exists for the intended destination, it is used. Otherwise the route discovery process is invoked.

### 3.2.1 Route Discovery

A route request packet is broadcasted in order to find out the source route to the destination. If an intermediate node has an entry for the destination, it appends that entry to the route record of the route request packet and puts the entire route record in a route reply packet. This route reply packet is then sent to the initial source which had invoked the discovery process. As the route record already contains the sequence of nodes through which the original request packet had arrived the current node, it is routed back to the source using this information.

If this node is also unaware of the route to the destination, it appends its own address to the route record and forwards the packet to its neighbors.

If the request packet reaches the destination the entire route record is put into a route reply packet and sent to the source.

One check is made to control the number of route request packets. A node processes the route request packet only if it has not already seen the packet and it’s address is not present in the route record of the packet.

After the route to the destination has been determined either by an intermediate node or the destination itself, the question of sending back this information to the initial source comes into picture. Now the route reply packet has to be routed back to the node which had initiated the request. There are two solutions for this problem. The responding node has a route to the source. If it has a route to the source in its route cache, it can use that route. Otherwise the node can initiate a route discovery to source and piggyback the route reply on this new route request. If symmetric links are supported the reverse of route record can be used.

### 3.2.2 Route Maintenance

Once the routes have been ascertained using the route discovery process, the route maintenance comes into play for various smaller and critical information.

Two types of packets - Route Error packet and Acknowledgement packet - are used for route maintenance.

When a node encounters a fatal transmission problem at its data link layer, it generates a Route Error packet. When a node receives a route error packet, it removes the hop in error from it’s route cache. All routes that contain the hop in error are are truncated at that point.
Acknowledgment packets are used to verify the correct operation of the route links. This also includes passive acknowledgments in which a node hears the next hop forwarding the packet along the route.

### 3.2.3 Protocol analysis

The Dynamic Source protocol is designed mainly for mobile ad hoc networks of up to about two hundred nodes, and is designed to work well with even very high rates of mobility.

- All aspects of the protocol operate entirely on-demand. This allows the routing packet overhead of the protocol to scale automatically to only that needed to react to changes in the routes currently in use.
- The protocol allows multiple routes to any destination and allows each sender to select and control the routes used in routing its packets, for example for use in load balancing or for increased robustness.
- Easily guaranteed loop-free routing,
- Support for use in networks containing unidirectional links,
- Use of only "soft state" in routing,
- Very rapid recovery when routes in the network change.

DSR is officially defined by an Internet-Draft [4] and is on its way to becoming an RFC. Once released, this will be the official definition of DSR. The most current explanation of the DSR protocol can be found in [5].

### 3.3 Ad-hoc On-demand Distance Vector (AODV) Routing

The Ad hoc On-Demand Distance Vector (AODV) algorithm enables dynamic, self-starting, multi-hop routing between participating mobile nodes wishing to establish and maintain an ad hoc network. AODV [6] is essentially a combination of both DSR and DSDV. It borrows the basic on-demand mechanism of Route Discovery and Route Maintenance from DSR, plus the use of hop-by-hop routing, sequence numbers, and periodic beacons from DSDV.

#### 3.3.1 Overview

Route Requests (RREQs), Route Replies (RREPs), and Route Errors (RERRs) are the message types defined by AODV. As long as the endpoints of a communication connection have valid routes to each other, AODV does not play any role. A node running AODV initiates a route discovery process only when it has data packets to send and it does not know any route to the destination node; therefore, route discovery in AODV is on-demand.

For broadcast messages, the IP limited broadcast address (255.255.255.255) is used. This means that such messages are not blindly forwarded. However, AODV operation does require certain messages (e.g., RREQ) to be disseminated widely, perhaps throughout the ad hoc network. The range of dissemination of such RREQs is indicated by the TTL in the IP header.

When a route to a new destination is needed, the node broadcasts a RREQ message. All AODV messages are sent to port 654 using UDP. A route can be determined when the RREQ reaches either the destination itself, or an intermediate node with a fresh enough route to the destination. A fresh route is a valid route entry for the destination whose associated sequence number is at least as great as that contained in the RREQ. The route is made available by unicasting a RREP back to the origination of the RREQ. Each node receiving the request adds a route back to the originator of the request, so that the RREP can be uni-cast from the destination along a path to that originator, or likewise from any intermediate node that is able to satisfy the request.

Nodes monitor the link status of next hops in active routes. When a link break in an active route is detected, a RERR message is used to notify other nodes that the loss of that link has occurred. The RERR message indicates those destinations (possibly subnets) which are no longer reachable by way of the broken link. In order to enable this reporting mechanism, each node maintains a precursor list, containing the IP address for all its neighbors that are likely to use it as the next hop towards a destination.

### 3.3.2 Routing Table

Although AODV shares DSR’s on-demand characteristics, it adopts a very different mechanism to maintain routing information. It uses traditional routing tables with one entry per destination. This is a departure from DSR, which can maintain multiple route cache entries per destination. Route table information must be kept even for short-lived routes, such as are created to temporarily store reverse paths towards nodes originating RREQs.

AODV uses the following fields with each route table entry:

- Destination IP Address
- Destination Sequence Number
- Valid Destination Sequence Number flag
- Other state and routing flags (e.g., valid, invalid, repairable, being repaired)
- Network Interface
- Hop Count (number of hops needed to reach destination)
- Next Hop
- List of Precursors
• Lifetime (expiration or deletion time of the route)

3.3.3 Sequence Number

Every route table entry at a node must include the latest information available about the sequence number for the IP address of the destination node for which the route table entry is maintained. This sequence number is called the destination sequence number. It is updated whenever a node receives new information about the sequence number from RREQ, RREP, or RERR messages which are related to that destination. AODV depends on each node in the network to own and maintain its destination sequence number to guarantee the loop-freedom of all routes towards that node.

A destination becomes unreachable when a link breaks or is deactivated. When these conditions occur, the route is invalidated by operations involving the sequence number and marking the route table entry state as invalid.

A node may change the sequence number in the routing table entry of a destination only if:

• it is itself the destination node, and offers a new route to itself, or
• it receives an AODV message with new information about the sequence number for a destination node, or
• the path towards the destination node expires or breaks.

3.3.4 Route Table Entries

When a node receives an AODV control packet from a neighbor, or creates or updates a route for a particular destination or subnet, it checks its route table for an entry for the destination. In the case that there is no corresponding entry for that destination, an entry is created. The sequence number is either determined from the information contained in the control packet, or else the valid sequence number field is set to false. The route is only updated if the new sequence number is either:

• higher than the destination sequence number in the route table, or
• the sequence numbers are equal, but the hop count (of the new information) plus one, is smaller than the existing hop count in the routing table, or
• the sequence number is unknown.

An important feature of AODV is maintenance of timer based states in each node, regarding utilization of individual routing table entries. A routing table entry is expired if not used recently. A set of predecessor nodes is maintained per routing table entry, which denotes the set of neighboring nodes that use this entry to route data packets. These nodes are notified with RERR packets when the next hop link breaks. Each predecessor node, in turn, forwards the RERR to its own set of predecessors, thus effectively erasing all routes using the broken link.

3.3.5 Route Discovery

A node disseminates a RREQ when it determines that it needs a route to a destination and does not have one available. This can happen if the destination is previously unknown to the node, or if a previously valid route to the destination expires or is marked as invalid. The Destination Sequence Number field in the RREQ message is the last known destination sequence number for this destination and is copied from the Destination. If no sequence number is known, the unknown sequence number flag is set. The Originator Sequence Number in the RREQ message is the node's own sequence number, which is incremented prior to insertion in a RREQ. The RREQ message contains a RREQ ID field, which is used along with the IP address of the originator to uniquely identify a request.

3.3.6 Controlling Dissemination of RREQs

After broadcasting a RREQ, the node waits for RREP. If the reply is not received within a certain time, the node may rebroadcast the RREQ using a binary exponential back-off algorithm. If a reply is not received even after a maximum permissible number of retries, the node assumes that no route to the destination exists.

Forwarding of RREQs is done when the node receiving a RREQ does not have a route to the destination. It then rebroadcasts the RREQ. The node also creates a temporary reverse route to the Source IP Address in its routing table with next hop equal to the IP address field of the neighboring node that sent the broadcast RREQ. This is done to keep track of a route back to the original node making the request, and might be used for an eventual RREP to find its way back to the requesting node. The route is temporary in the sense that it is valid for a much shorter time, than an actual route entry.

The recent specification of AODV includes an optimization technique to control the RREQ flood in the route discovery process. It uses an expanding ring search initially to discover routes to an unknown destination. In the expanding ring search, increasingly larger neighborhoods are searched to find the destination. The search is controlled by the TTL field in the IP header of the RREQ packets. If the route to a previously known destination is needed, the prior hop-wise distance is used to optimize the search.

3.3.7 Reply Messages

When the RREQ reaches a node that either is the destination node or a node with a valid route to the destination, a RREP is generated and uni-casted back to the requesting node. As the RREP is forwarded back towards the node
which originated the RREQ message, the Hop Count field is incremented by one at each hop. Thus, when the RREP reaches the originator, the Hop Count represents the distance, in hops, of the destination from the originator.

In the absence of the source routing and promiscuous listening mechanisms of DSR, AODV can gather only a very limited amount of routing information. In particular, route learning is limited only to the source of any routing packets being forwarded. This usually causes AODV to rely on a route discovery flood more often, which may carry a significant network overhead. Moreover, to make use of route caching aggressively, DSR replies to all requests reaching a destination from a single request cycle. Thus the source learns many alternate routes to the destination, which will be useful in the case the primary (shortest) route fails. Having access to many alternate routes saves route discovery floods, which is often a performance bottleneck. In AODV, on the other hand, the destination replies only once to the request arriving first and ignores the rest. Thus routing table maintains at most one entry per destination.

3.3.8 Route Maintenance

AODV provides the option of using Hello messages for maintaining local connectivity. These messages are broadcasted periodically by a node to its immediate neighbors. They act as local advertisements for the continued presence of the node. Neighbors using routes through the broadcasting node will continue to mark the routes as valid as long as they receive the hello message. If a message does not arrive from the node within a specified time, the neighbor assumes that the node has moved away and marks the link to the node as broken. It then notifies the affected set of nodes by sending a link failure notification to that set of nodes.

3.3.9 Advantages over DSR

The current specification of DSR does not contain any explicit mechanism to expire stale routes in the cache, or prefer fresher routes when faced with multiple choices. Stale routes, if used, may start polluting other caches. Some stale entries are indeed deleted by route error packets; but because of promiscuous listening and node mobility, it is possible that more caches are polluted by stale entries than are removed by error packets. In contrast, AODV has a much more conservative approach than DSR. When faced with two choices for routes, the fresher route (based on destination sequence numbers) is always chosen. Also, if a routing table entry is not used recently, this entry is expired.

Secondly, the route deletion activity using RERR is also conservative in AODV. By way of a predecessor list, the error packets reach all nodes using a failed link on its route to any destination. In DSR, however, a route error simply backtracks the data packet that meets a failed link. Nodes that are not on the upstream route of this data packet but using the failed link are not notified promptly.

4 Conclusion

Many different protocols have been proposed to solve the multi-hop routing problem in ad hoc networks, each based on different assumptions and intuitions. This paper, describes some of the existing protocols originating from the proactive and reactive approaches to routing.

Extensive research has been done in comparing the different proposed adhoc routing protocols under varying network scenarios. Routing overhead, packet delivery ratio, end-to-end delay, path optimality, and throughput are some metrics commonly used in the comparisons. The results of the performance studies indicate that the different protocols are suited to different network conditions. For instance, DSDV performs considerably well under light node density. However, DSR outperforms DSDV at high node density due to the tremendous overhead incurred by DSDV when exchanging routing tables. On the other hand, AODV offers better delay and throughput performance as compared to DSR under higher load and mobility conditions. This may be attributed to the use of aggressive caching mechanism and the lack of any mechanism to expire stale routes in DSR.

References