MULTIPATH ROUTING PROTOCOLS FOR MOBILE AD HOC NETWORK

#Amit Sharma¹, kshitij shinghal², Pushpendra Vikram Singh³, Himansu Verma⁴
¹²Dept. of E&C Engg., KIMT, Moradabad, U.P.
³Dept. of E&C Engg., MIT, Moradabad, U.P.,
#Author 1 is a research scholar at IFTM University Moradabad, UP, India
amit211212@rediffmail.com

Abstract: Multi-path routing has been studied thoroughly in the context of wired networks. It has been shown that using multiple paths to route messages between any source-destination pair of nodes balances the load more evenly throughout the network. The common belief is that the same is true for ad hoc networks, i.e., multi-path routing balances the load significantly better than single-path routing. In this paper, we introduce The Temporally Ordered Routing Algorithm (TORA) is a highly adaptive loop free distributed routing algorithm based on the concept of link reversal. TORA is proposed to operate in a highly dynamic mobile networking environment.

Keywords: TORA, RREQ, DSR, RERR.

I. INTRODUCTION

Multi-path routing consists of finding multiple routes between a source and destination node. These multiple paths between source and destination node pairs can be used to compensate for the dynamic and unpredictable nature of Ad hoc networks. Multipath routing consists of three components viz. route discovery, traffic allocation and route maintenance.

Route Discovery

Route discovery consists of finding multiple routes between a source and destination node. Multipath routing protocols can attempt to find node disjoint, link disjoint, or non-disjoint routes. Node disjoint routes[1,2], also known as totally disjoint routes, have no nodes or links in common. Link disjoint routes have no links in common, but may have nodes in common. Non-disjoint routes can have nodes and links in common. Fig 1.1 shows different kinds of multipath routes. Routes SXD, SYD, and SZD in Fig 1.1 (a) have no links or nodes in common and are therefore node disjoint. Routes SXYZD and SYD in Fig 1.1 (b) have node Y in common and are therefore link disjoint. Routes SXD and SXYD in Fig 1.1(c) have node X and link SX in common and are therefore non-disjoint.

II. SPLIT MULTIPATH ROUTING (SMR)

It is a multipath version of DSR. Unlike many prior multipath routing protocols, which keep multiple paths as backup routes, SMR is designed to utilize multipath concurrently by splitting traffic onto two maximally disjoint routes. Two routes said to be maximally disjoint if the number of common links is minimum. SMR uses one route discovery process to accumulate as many as possible routes to the destination node. This route discovery process runs in the same way as in DSR. However, there are more steps involved in processing RREQ packets at intermediate and destination nodes[2]. If an intermediate node receives a RREQ packet, it adds its own address and rebroadcasts the RREQ packet. Whenever an intermediate node receives another RREQ from the same source node and with the same request id, i.e. a duplicated RREQ, the node checks the following two things (Fig 1.2(a)).

If the first RREQ packet arrives at the destination node, a RREP is generated and sent back on the reverse path, which is the “Shortest delay path”. Then the destination node waits a period of time and selects multiple disjoint routes, according to the first path, and sends RREP packets back to the source via the selected routes (Fig 1.2(b)).

III. DYNAMIC MULTI-PATH SOURCE ROUTING (DMSR)

DMSR extends DSR’s routing mechanism to deal with multi-path routing couple with bandwidth constraint. It consists of three major phases, namely routing discovery, multi-path route selecting and routing maintenance. In multi-path route selecting phase, the ideal number of multi-path routing is achieved to compromise between load balancing and network overhead[4]. A DMSR RREQ includes Source id, RREQ id, Routing list, Hmax and Bmin. When a source node initiates a QoS request, it firstly checks whether it has the

Fig1.1: Different Kinds of Multipath Routes
(a) Node Disjoint (b) Link Disjoint (c) Non Disjoint

(a) RREQ Propagation
(b) Available Paths

Fig 1.2: Route Discovery
routing information to the destination node. If not, it begins to broadcast RREQ to its neighborhoods. Once relay nodes receive this RREQ, it can’t respond to the RREQ even if it has the routing information to the destination node. Rather, it follows the steps:

**Step 1** - If current node itself is in the Routing list of RREQ, it will discard the RREQ because of the routing loop. Otherwise, it goes to step 2;

**Step 2** - If the tuple (SourceID, RREQID) of RREQ is not included in the routing table, which means the current node is the first time to receive this RREQ, it calculates the corresponding value of the bandwidth. If the value can’t meet the requirement denoted by Bmin, and the RREQ will be discarded. Otherwise, it goes to step 3.

**Step 3** - Adds the value of bandwidth to the corresponding fields of the RREQ. Then the RREQ will be continually forwarded and it goes back to step 1.

IV. MULTIPATH-DISTANCE VECTOR ROUTING (MP-DSR) [3]:

MP-DSR is an extension of DSR with QoS support. It tries to forward packets on multiple disjoint paths with certain end-to-end reliability requirements. This reliability considers the probability of having a successful transmission between the two mobile nodes within the time period from t0 to t0 + t, where t0 is any time instance. The probability successful transmission is shown in the following equation.

\[ P(t) = 1 - \prod_{k \in K} (1 - p(k,t)) \]

Where \( K \) is a set of node-disjoint paths from the source to the destination. \( p(k,t) \) is the path reliability of path \( k \), calculated as the product of link availability of all the links in path \( k \). In other words, \( P(t) \) is the probability that at least one path stays connected for the duration of t. After the value of lowest path reliability requirement and the number of paths to be discovered are set, the source node floods a RREQ packet for a set of paths (neighbor nodes), which can satisfy these requirements as shown in Fig 1.3. Each RREQ packet contains additional parameters e.g. the reliability requirement, time window, path that a RREQ message has traversed, etc. Whenever an intermediate node receives a RREQ, it checks, whether the RREQ meets the lowest path reliability requirement. If yes, the intermediate node adds itself and sends out multiple copies of this RREQ to its neighbors. Otherwise RREQ packet is dropped. After the reception of first RREQ, the destination waits a period of time. Then it selects multiple disjoint paths out of all received RREQ packets. RREP packets are sent along these paths as shown in Fig 1.3.

![Fig 1.3: MP-DSR RREQ & RREP Messages](image)

V. AODV MULTIPATH ROUTER APPROACH (AODVM-R)

When performing route discovery, the source and intermediate nodes maintain multiple routes to the destination. To ensure loop freedom the RREQ packet includes path information (path from the source to the router). Primary and secondary routes will have the same sequence numbers. When a link breaks, a node tries to reestablish the route using alternate paths. If still there is an unreachable destination, the node sends an RERR message to its neighbors. If the primary route works for a long time, alternate paths might timeout because they are not used. While the primary route is being used, send REFRESH message to the alternate routes occasionally to refresh them. The REFRESH packet is sent every active route_timeout /2 seconds. The REFRESH packet is forwarded to the destination, refreshing the routes on the way. If an alternate route is detected to be broken, it is simply discarded from the route table[5,6].

VI. AD HOC ON-DEMAND MULTIPATH DISTANCE VECTOR ROUTING (AOMDV)

AOMDV involves route discovery and route maintenance phases similar to AODV. Noteworthy feature of the AOMDV protocol is the use of routing information already available in the underlying AODV protocol as much as possible. Thus little additional overhead is required for the computation of multiple paths. The AOMDV protocol has two main components[7]:

1. A route update rule to establish and maintain multiple loop-free paths at each node.
2. A distributed protocol to find 1 disjoint paths.

To keep track of multiple routes, the routing entries in each node for each destination contain a list of the next hops along with the corresponding hop counts. All the next hops have the same sequence number. For each destination, a node maintains the “advertised hop count”, which is defined as the maximum hop count for all the paths.

**Table 1: Structure of Routing Table Entries for AOMDV**:

<table>
<thead>
<tr>
<th>field</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>destination</td>
<td>route-destination node</td>
</tr>
<tr>
<td>sequence number</td>
<td>route sequence number</td>
</tr>
<tr>
<td>advertised hopcount</td>
<td>advertised hop count</td>
</tr>
<tr>
<td>route list</td>
<td>list of next hops</td>
</tr>
<tr>
<td>{seqnum1, ...}</td>
<td>list of hop counts</td>
</tr>
<tr>
<td>expiration timeout</td>
<td>time for route entries to expire</td>
</tr>
</tbody>
</table>
The “advertised hop count” is initialized each time the sequence number is updated. A node i updates its “advertised hopcount” for a destination d whenever it sends a route advertisement for d. Specifically, it is updated as follows:

\[
\text{advertised\_hopcount}_i^d := \max_k \{ \text{hopcount}_k | (\text{nexthop}_k, \text{hopcount}_k) \in \text{route\_list}_i^d \}.
\]

The route update rule in algorithm 2 is invoked whenever a node receives a route advertisement. Lines (1) and (9)-(10) of the AOMDV route update rule ensure loop freedom. AOMDV update rule is used whenever a node i receives a route advertisement to a destination d from a neighbor j. The variables seqnum_i^d, advertised_hopcount_i^d, route_list_i^d represent the sequence number, advertised_hopcount and route_list for destination d at node i respectively. AOMDV can be used to find node-disjoint or link-disjoint routes. To find node-disjoint routes, each node does not immediately reject duplicate RREQs. Each RREQ arriving via a different neighbor of the source defines a node-disjoint path. This is because nodes cannot broadcast duplicate RREQs, so any two RREQs arriving at an intermediate node via a different neighbor of the source could not have traversed the same node. In an attempt to get multiple link-disjoint routes, the destination replies to duplicate RREQs regardless of their first hop[8].

\[
\text{if} \ (\text{seqnum}_i^d < \text{seqnum}_j^d) \ \text{then} \ (1) \\
\quad \text{seqnum}_i^d := \text{seqnum}_j^d; \ (2) \\
\text{if} \ (i \neq d) \ \text{then} \ (3) \\
\quad \text{advertised\_hopcount}_i^d := \infty; \ (4) \\
\quad \text{route\_list}_i^d := \text{NULL}; \ (5) \\
\quad \text{d}_i \ (j, \text{advertised\_hopcount}_i^d + 1) \ \text{into} \ \text{route\_list}_i^d; \ (6) \\
\text{else} \ (7) \\
\quad \text{advertised\_hopcount}_i^d := 0; \ (8) \\
\text{endif} \ (9) \\
\text{elseif} \ (\text{seqnum}_i^d = \text{seqnum}_j^d) \ \text{and} \ \\
\quad (\text{advertised\_hopcount}_i^d, i) > (\text{advertised\_hopcount}_i^d, j) \ \text{then} \ (10) \\
\quad \text{d}_i \ (j, \text{advertised\_hopcount}_i^d + 1) \ \text{into} \ \text{route\_list}_i^d.
\]

**Algorithm 2: AOMDV Route Update Rules**

To ensure link-disjointness in the first hop of the RREP, the destination only replies to RREQs arriving via unique neighbors. After the first hop, the RREPs follow the reverse paths, which are node-disjoint and thus link-disjoint. The trajectories of each RREP may intersect at an intermediate node, but each takes a different reverse path to the source to ensure link-disjointness. Fig 1.4 illustrates link-disjoint path computation in AOMDV. S, D and I denote the source, intermediate node and the destination, respectively. There are two possible sets of link-disjoint paths between S and D.

**Fig 1.4 Link-Disjoint Paths**

AOMDV offers better performance relative to AODV under a wide range of mobility and traffic scenarios. It has been observed that AOMDV offers a significant reduction in delay, often more than a factor of two. It also provides up to about 20% reduction in the routing load and the frequency of route discoveries. In general, AOMDV always offers a superior overall routing performance than AODV in a variety of mobility and traffic conditions[9].

**VII. TEMPORALLY ORDERED ROUTING ALGORITHM (TORA)**

The Temporally Ordered Routing Algorithm (TORA) is a highly adaptive loop free distributed routing algorithm based on the concept of link reversal. It is designed to minimize reaction to topological changes. A key design concept in TORA is that it decouples the generation of potentially far-reaching control messages from the rate of topological changes. Such messaging is typically localized to a very small set of nodes near the change without having to resort to a complex dynamic, hierarchical routing solution. TORA is also characterized by a multi-path routing capability. Each node has a height with respect to the destination that is computed by the routing protocol. Fig 1.5 illustrates the use of the height metric. TORA is proposed to operate in a highly dynamic mobile networking environment. The protocol performs three basic functions as follows-

a. Route creation
b. Route maintenance
c. Route erasure

**Fig 1.5: TORA Height Metric**
When a node needs a route to a particular destination, it broadcasts a QUERY packet containing the address of the destination for which it requires a route. This packet propagates through the network until it reaches either the destination, or an intermediate node having a route to the destination[10,11].

The node which receives the QUERY then broadcasts an update packet, listing its height with respect to the destination. As this packet propagates through the network, each node that receives the update sets its height to a value greater than the height of the neighbor from which the update has been received. This has the effect of creating a series of directed links from the original sender of the QUERY to the node that initially generated the update.

VIII. CONCLUSION

In this paper we study the different multipath routing protocols for mobile Ad HOC network. The overall summery of above-mentioned multipath routing protocols is shown in the table:

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Base Protocol</th>
<th>Routing Choice Made at</th>
<th>Motivation/Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>AODVM-R</td>
<td>AODV</td>
<td>Intermediate nodes</td>
<td>Reduces number of route discoveries</td>
</tr>
<tr>
<td>AOMDV</td>
<td>AODV</td>
<td>Source node</td>
<td>Reduction in delay, routing load and the frequency of route discoveries</td>
</tr>
<tr>
<td>SMR</td>
<td>DSR</td>
<td>Source node (source routing)</td>
<td>Spliting traffic provides better load distribution</td>
</tr>
<tr>
<td>MP-DSR</td>
<td>DSR</td>
<td>Source node (source routing)</td>
<td>QoS applications with soft end-to-end reliability</td>
</tr>
<tr>
<td>TORA</td>
<td>link reversal</td>
<td>Source node (Source routing)</td>
<td>Operate in a highly dynamic mobile networking environment</td>
</tr>
</tbody>
</table>

REFERENCES