A Flexible Family of Multi-Path Routing Protocols over a MANET

Abstract: This paper's contribution is a family of multi-path routing protocols that emphasize different aspects of multi-path routing, though with a central theme of path reliability within a MANET. Tests demonstrate that for all members of the family, the network performance (end-to-end delay and packet loss ratio) compares favorably to the single-path DSR protocol at moderate to high speeds, though as might be expected there is a trade-off against control packet overhead. Where the design effort differs from other multi-path protocol reviewed is that different aspects of the performance are emphasized in different members of the protocol family. The paper gives analysis on how a member of the family is constructed to perform in a particular way. In a general sense, this is the main message of the paper: that such protocol families can be constructed with a particular application type in mind, which in this case was media streaming.

Keywords: *DSR protocol, MANET, media streaming, MPDSR family, multi-path routing.*

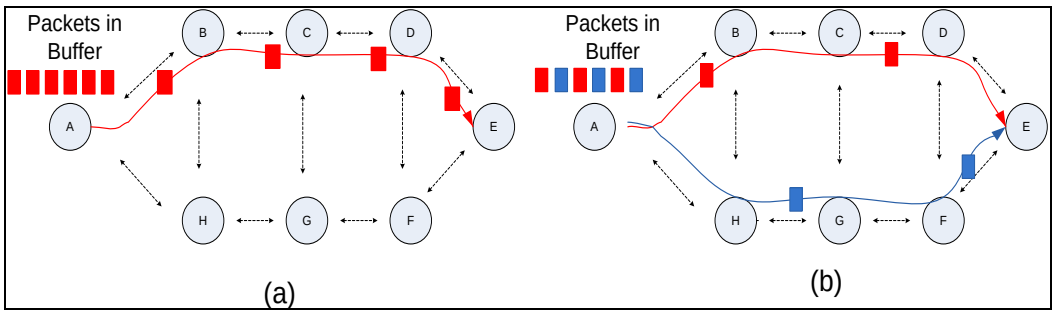
1 Introduction

Within a Mobile Ad Hoc Network (MANET), multi-path routing [1] has been advocated for a variety of reasons. Compared to single-path routing protocols, there may be no need to initiate route discovery again if a link in the path is broken, as suitable alternatives may already exist, though there is no guarantee of this. Because multiple paths are sought it is also possible that, amongst these, routes of better quality can be selected. The ability to counteract congestion through a selection of routes is also an attractive prospect. Various general benefits may arise such as reduced end-to-end delay [2], increased path reliability [3], an increase in aggregate bandwidth [4], load-balancing to equalize energy consumption [5], and congestion avoidance [6]. Multipath routing can also act as a security mechanism [7], though this paper does not examine that aspect further as ways to increase path reliability are mainly explored. For media streaming applications in particular, path reliability reduces packet loss. Data dependencies arise within such compressed bit-streams due to the predictive

structure of motion estimation and compensation, which serves to remove temporal redundancies within video sequences [8]. In fact, one of the members of the protocol family developed within this paper was already employed [9] for media streaming within a vehicular ad hoc network, though the current paper demonstrates entirely new results, as the testing scenario is now more general and all members of the protocol family are investigated.

Figure 1 illustrates the difference between single-path and multi-path routing. In the Figure, a single application transfers data packets across two node-disjoint paths (ones in which no node is in common). It is also possible to route data from two or more co-located sources across multiple paths, or, in other ways, vary the traffic allocation granularity [10]. An example of using co-located sources (or multiple connections) is Multiple Description Coding for resilient video stream transport [9, 11]. In which case, the terms 'multi-path' and 'multiple path' are closely related. It is not always possible to establish node-disjoint routes and, hence, link-disjoint routes may occur, though these are less resilient [12] as several paths may rely on the same node and the same node will spend more time transmitting, leading to disproportionate expenditure of energy. As a compromise, maximally-disjoint routes [13] with a minimal number of shared nodes/links can be sought.

Figure 1 (a) Single path for transmitting data. (b) Multi-paths instead of a single one to distribute the data over the network.



This paper's contribution is a family of multi-path routing protocols that emphasize different aspects of multi-path routing, though with a central theme of path reliability. Unlike much previous work the requirements of the multi-path protocol are molded to the application. Therefore, we do not consider that there is one ideal multi-path routing protocol but rather a family that stresses different aspects of the performance. Consider scenarios

involving real-time transmission such as media streaming. Packets are usually sent frequently to meet presentation deadlines at the destination device. For instance, the compressed video bit-stream is normally decoded at a rate of 25 or 30 frame/s. Because of process scheduling at the source, packets bearing the bit-stream may be sent in bursts to meet these deadlines. In this case, employing a routing protocol capable of using disjoint paths instead of a single route for transmitting data, leads to greater load balancing through the availability of different routes. The more uniform distribution of the payload over all the available routes leads to more efficient use of network resources, including energy consumption by transmitting nodes. Therefore, both the reliability of the transmission session also improves. Besides, traffic handling by different nodes prevents congestion of packets on certain nodes. Consequently, the average end-to-end latency is reduced, which suits a real-time application.

Three different algorithms are initially proposed to provide balanced delivery of the data packets and gain reduced end-to-end delay. This is achieved by making enhancements and amendments to the single-path Dynamic Source Routing (DSR) protocol [14]. The resulting family of protocols allows tuning of different aspects of DSR's performance. These protocols are named by us as Multipath-DSR-1 (MPDSR-1), MPDSR-2, and MPDSR-3. Besides, MPDSR-3 is implemented by two different mechanisms of route refreshment to further control its performance. Notice that there is an earlier MPDSR protocol [3], reviewed in Section 2, which is not the work of the authors and is unrelated to this family of protocols.

In MPDSR-1, the destination node returns more than one route back to the source on the basis of maximally node-disjointness, the property adopted from SMR [13]. However, unlike SMR, intermediate nodes are allowed to also return route replies that contain routes that are not necessarily disjoint. At packet transmission time, the source node selects the shortest path on the basis of the least number of hops, as this is expected on average to increase the reliability of the path. In that sense, though MPDSR-1 selects from multiple paths it is not strictly a multi-path transmission system as it does not transmit simultaneously over different routes. Instead, it exploits multiple paths to select a path at any one time. Like SMR it provides a more efficient way compared to DSR [14] of selecting higher quality routes. DSR itself is more indiscriminate in its selection of routes, as in DSR every route request is

responded to even if a node has already responded. A number of factors are balanced in MPDSR-1. For example, allowing intermediate nodes to reply reduces overhead even though there is a risk that these replies are based on stale route caches. The net effect of MDSR-1 is expected to be a reduction in end-to-end delay compared to DSR at moderate to high speeds.

The MPDSR-2 protocol is based on the MPDSR-1 but is assumed to provide still more reliable conditions for data transmission. Consequently, in this variant, intermediate nodes no longer supply route replies. The destination also delays its choice until several route request packets have arrived. It is expected that, at moderate to high speeds, compared to DSR, the packet loss ratio will be reduced, though no gain in end-to-end delay will occur. At low speeds, DSR's indiscriminate selection of routes is no longer a disadvantage as routes change less quickly. On the occasions that routes do fail, DSR has a wider selection of routes to try, with more of these likely to be valid and not stale, because of low mobility. Because of low mobility there are fewer routes available to select from under the MPDSR family as the routes have been selected with more discrimination.

The MPDSR-3 routing algorithm acts similarly to the MPDSR-1 algorithm, but the algorithm was made more adaptive to conditions within the network. Two different and variant mechanisms were added to this protocol to help avoid stale route caches, which are the principle problem of DSR. The adaptive route-reply mechanism together with the route refreshment techniques are expected at moderate to high speeds to deliver a significant reduction in end-to-end delay and an improvement in the packet delivery ratio of the protocols.

However, using maximally-disjoint routes with a high number of hops has at least one disadvantage: as the channel capacity is limited, if a packet remains longer within the network available resources are utilized inefficiently. Additionally, as a packet travels over a path with a greater number of hops, it is more exposed to the high levels of uncertainty within the physical medium. Consequently, packet delivery failure becomes more probable. These conditions necessitate delivering the packets over routes which have fewer hops and are more reliable. Therefore, in addition to a reliability factor for the routes, the number of hops in each route should be taken into account during route selection to reduce the risks

from disjoint paths taking too long a path (one with many hops). Consequently, in addition to the algorithms presented for data delivery over disjoint routes, a routing algorithm is proposed that uses a reliability factor instead, namely MPDSR-Reliable(R). Before proceeding to analysis of this family of protocols, the paper now considers other research on multi-path routing protocols within dynamic networks.

2 Related Research

There are many multi-path routing algorithms, even when their scope is limited to ad hoc networks. Consequently, for more comprehensive reviews the reader is referred to surveys such as [1, 15, 16]. SMR [13] was an influential algorithm for us, as it was based on DSR and it introduces the notion of maximally disjoint paths based on finding the number of similar nodes that two paths have in common. The main motivation of SMR appears to have been to demonstrate the advantage of multi-path routing, whereas, given the experience of SMR, it is possible to draw lessons from its example and make some adaptations for different test scenarios. SMR prevents intermediate nodes replying in order to ensure enough route requests reach the destination. In that way, the destination can select maximally disjoint routes. We sometimes also prevent intermediate nodes from replying. SMR also alters the RREQ forwarding mechanism of DSR by forwarding duplicate RREQs, provided they originate on different input links. The intention of this modification is to allow the arrival of more disjoint routes at the destination. However, it imposes a burden on the network, because of the number of circulating RREQs. Consequently, we do not adopt this aspect of SMR. In fact, in the performance analysis of [16], SMR's performance was found to degrade with increasing node density because of excessive control overhead. In [3] for original MP-DSR, each node maintains an estimate of the reliability of links by monitoring the signal strength of beacon packets that all nodes periodically broadcast. The product of link availabilities along a path is a measure of link reliability. Link availability is also the basis of the decision by intermediate nodes to relay RREQs, which takes place even when duplicate RREQs arrive up to a preset limit to accept duplicate RREQs. The destination node accumulates reliable but

disjoint paths up to a pre-determined number of such paths. The path selection is subject to a time-out to prevent waiting for too long for a set of paths.

As an alternative to DSR-based protocols, the Ad hoc On demand Multipath Distance Vector (AOMDV) [17] was also designed with reliability in mind in networks with frequent topology changes. Because of hop-by-hop routing, it is possible to have routing loops unless destination sequence numbers increase monotonically (as in the well-known Ad hoc On Demand Distance Vector (AODV) protocol [18]) along the path. However, unlike single-path AODV, it is also possible to have different hop counts associated with any destination if multiple paths are propagated. Therefore, to avoid this possibility, AOMDV chooses the maximum hop count to a particular destination as the destination's advertised hop count. In AOMDV, link disjointness is established by a distributed mechanism in which only the first RREQ to arrive from the same source over different links is relayed.

In [16] AOMDV was found to perform well in high-mobility scenarios but another AODV variant, AOMDV_Multipath [19] was preferable if the node mobility was low and node density was high. In AODV_Multipath, duplicate RREQs are forwarded by nodes in order to maximize the number of routes found. However, as RREPs return and nodes overhear RREP transmissions, nodes remove duplicate RREQs from their RREQ tables. Thus, disjoint routes are established at RREP time.

After this brief review of multi-path routing and the issues that arise, the paper now turns to the background to the design of the multi-path protocol family.

3 Background

3.1 DSR protocol

This study is based around the well-known single-path DSR protocol [14] as a basis for the design of the family of multi-path routing algorithms. The reasons for this choice are:

1- The scenario which was under study does not involve interactive communication between nodes. Consequently, it is more efficient to select an on-demand routing protocol for irregular communication sessions between random nodes. However, unlike AODV [18], and many

other on-demand protocols, DSR does not route on a hop-by-hop basis, which reduces the overhead from sending beacon or 'HELLO' messages.

2- Under the DSR protocol, each node holds a route cache in which it keeps track of any (multiple) routes to particular destinations. The route cache in DSR has the benefit of storing a number of routes for a set of destinations. Storing sets of available routes allows more control over what routes can be chosen for data transmission. Besides faster transmission normally occurs, as there is no need for renewed route discovery. DSR ordinarily accumulates these routes during route request operations. In promiscuous mode, a node can also listen-in to messages that are not sent to it and do not pass through it, thus building-up its cache. The route-cache facility of DSR makes it appropriate for adaptation to multiple path data delivery.

3- The DSR protocol is a source-initiating protocol in which the source node determines which path the data packet should take to get to the destination. Unlike the hop-by-hop structure of AODV, DSR allows selection of routes by knowing which nodes are present on each route. DSR is a routing protocol which has two main phases in its functioning structure: 1) route discovery; 2) route maintenance; with the addition of route caching.

Route discovery: is based on two main functions, route request and route reply. As in all on-demand protocols, a source floods route-request messages via its nearest neighbors. As mentioned previously, unlike some other on-demand protocols, DSR does not employ beacon or 'hello' messages. There are also various pruning methods to reduce the number of circulating request messages. If an intermediate node already has a route to the destination it replies with a route-reply message. On receiving a route-request, the destination also replies through the reverse path that the route-request message has taken. Most of the modifications in this paper occur in the route discovery phase for improving and extending the DSR protocol into a protocol which supports multi-path networking. Hence, these issues are considered in Section 4.

Route maintenance: in the DSR protocol is based on the acknowledgment from a receiving node that a sending node receives after each transmission over a link. When a receiving node on the path realizes an error has probably occurred because of an acknowledgment time-out, an error packet is generated and sent back to the originator of the lost packet in order to inform that node about an error

in a link on the path. Henceforth, the source node and all the listening nodes will truncate paths which transmit over that link. The source node must then re-start route discovery.

Route caching: each node employs a route cache in order to store all the routes, whether it has received them in the route-reply packets from the destination node or by extracting the route from the header of overheard packets. This helps DSR to be time efficient by taking advantage of the routes already obtained and stored in the route cache. In doing so, control overhead traffic is also reduced.

3.2 Multi-path routing evaluation

A drawback of multi-path routing is that a higher number of out-of-order packets may arrive at the destination, because different packets take different routes to travel. An increasing number of available paths may create re-ordering problems. Out-of-order arrival necessitates the implementation of reordering buffers in the nodes. This creates another issue of how much buffer space is needed for specific applications considering energy consumption and storage memory constraints. Therefore, the degree to which the multipath routing protocol is optimized in delivering data in their original order can be an evaluation factor of the routing protocol performance for specific scenarios.

Another disadvantage of multi-path routing is the increased complexity due to the implementation of routing buffers and the need for extra computations for different multi-path mechanisms. The increase in control overhead can be another issue in using multi-path techniques due to the necessary modification and functions. Multi-path routing protocols such as SMR) [13] and AOMDV [17], unlike many single-path protocols, do not allow nodes other than destination node to reply to route-request packets even if they have routes to the requested destination, because the routes which are not initiated originally by the destination for the source node may not be the optimum disjoint routes for the use of source node. Thus, many route-request packets are re-flooded into network and get re-propagated until they reach the destination node.

4 Multi-path protocol family

This Section now describes the proposed family of routing protocols.

4.1 MPDSR-1

For route discovery, the broadcast of route-request packets is similar to that of DSR. A node broadcasts route-request packets (*RREQ*s) whenever it has packets in the buffer for a specific route,

provided it has no existing route to the destination node in its route cache. Aspects of this protocol are captured in the pseudo-code of Figure 2.

As for all the algorithms in this study, another route cache named *RREQ*-route-cache is implemented in the protocols other than the originally route cache employed by DSR This cache is used to store the routes obtained from *RREQ* packets at the destination node for the purpose of selecting *RREP*s. The fastest route is sent in a route-reply (*RREP*) from the destination node. This route is simply the reverse route to that in the first *RREQ* header to be received in this communication session. The destination node also performs a check to ensure that an *RREQ* is from the current communication session and not a prior session. At the same time as receiving the first *RREQ*, a timer is set, after which further *RREQ*s are received. After the time-out, the destination node searches among its cached routes to find the route which is the maximally node-disjoint route to the first route-reply path. If there are multiple routes meeting this condition, the route with the shortest hop count is taken for the second *RREP* and sent back to the source with this route. This process can extend beyond two routes by finding the next candidate node-disjoint path.

In this protocol, the intermediate nodes are also allowed to send RREPs back to the originator of the route requests. This is permitted despite the fact that routes sent back by intermediate nodes are not particularly reliable as they can come from stale route caches in DSR [14] or the original MPDSR [3]. Letting intermediate nodes reply has the advantage of limiting the number of control packets flooded into the network for finding new routes, which leads to less control packets and faster data transmission. Therefore, we see that the main motivation of MPDSR-1 is to reducing control overhead. To implement this, intermediate nodes also set a timer after receiving their first *RREQ*.

Each node may generate gratuitous route replies. A gratuitous route-reply is generated by a node which overhears a sent packet. If that node is on the route of the packet, the node makes an *RREP* to the source if the route reduces the hop count of the path. Gratuitous route-replies are a feature originally appearing in DSR [14]. This is not specifically mentioned as a feature of SMR [13] or original MPDSR [3].

When the source node receives the first *RREP* it starts transmitting its buffered data to the destination immediately. For each new packet the node rechecks its route cache to use the route with minimal hop count to achieve fast delivery of the packet. Therefore, though there are multiple routes available, the source node selects the route with the shortest number of hops at each transmission time. This is similar to DSR, with no aim of selecting disjoint routes.

Author

If a route fails, route recovery to find new routes to the destination is not employed unless no routes are found in the route cache of the source node. As in DSR, route error (*RERR*) packets are generated by nodes that realize the misbehavior of their neighboring nodes. The error packets inform other nodes about a failed link and affected paths are torn down.

The major effect of the applied modifications in this algorithm compared to the original DSR protocol is a reduction in end-to-end delay, thanks to an efficient selection of routes and use of *RREP*s. Besides, a reduction in packet drops is expected as more reliable routes (routes with shorter hop counts) are selected by the source. However, while using routes from intermediate nodes helps the speedy functioning of the protocol, it also leads to more packet drops, which counteracts the gain from utilizing more reliable routes at the source.

Figure 2 Pseudo-code for aspects of protocol MPDSR-1

Originating RREPs

```
if (RREQ from the source is the first RREQ in a new session)
        send RREP with the fast route(First RREQ);
        Start timer to count till τ milli-seconds;
endif
while (receiving RREQs if there is time left on the timer)
        if (the destination node receives the RREQ)
                Save the paths obtained from RREQs in the RREQ-route-cache;
        else
                Re-flood the RREQ into the network;
        endif
endwhile
if ( timer is up)
        No more RREQs accepted;
        Goto RouteSelecting function;
endif
```

RouteSelecting function

```
if (two routes are in the cache)
        Initiate RREP for the second one;
elseif (more than two routes are available )
        for (the routes in RREQ-route-cache with a different first hop from the route in the
        RREP which has already been sent back)
                Select the route with the least number of hops;
        endfor
endif
```

ScoreSimilarity function

```
   for (nodes in route A from RREQ-route-cache)
       for (nodes in route B from RREQ-route-cache)
            Calculate the similarity variable based on the number of nodes that
            the routes have in common;
       endfor
   endfor
return the value of how similar the routes are;
```

4.2 MPDSR-2

In the route discovery phase of MPDSR-2, similarly to MPDSR-1, each node broadcasts RREQs whenever it has packets in its buffer for a specific route but has no routes to the destination node. Upon receiving the first RREQ packet a timer is triggered at the destination. The destination acquires the routes of all the incoming RREQs during the time allowance of the timer and saves them in order of hop count. Therefore, unlike MDSR-1, MDSR-2 does not send the fastest route back to the source node. Aspects of this protocol are captured in the pseudo-code of Figure 3.

After the time-out occurs, the destination node starts to process the saved routes in its route-reply cache in order to find the most appropriate paths, ordering in terms of 'node-disjointness' and then hop count. Depending on how many paths are required, the selected paths are sent back to the originator of the RREQ packets. Clearly, the fewer the paths selected the more likely it is that the paths will have no nodes in common.

In MPDSR-2, unlike MPDSR-1, intermediate nodes do not send RREPs, to increase route availability. That is more routes arrive from the destination but the replies are less reliable than if they came from intermediate nodes. When the originator of the RREQs receives the first RREP from the destination, it starts a timer and waits for a pre-set amount of time to receive subsequent RREP packets. After receiving a pre-set number of routes or after a time-out, whichever is earlier, the node starts to send its buffered packets to the desired destination. The routes are used in turn to spread the load on the different paths. In the case of no route notification before time-out, the route discovery mechanism is employed to find a fresh disjoint set of routes. Increasing the time to gather routes using the timer, as introduced in this protocol, results in a small delay but potentially results in finding more reliable routes.

The implemented modifications are expected to lead to a greater packet delivery ratio, though there are an increased number of control packets. No significant change is made to the average end-to-end delivery of the protocol compared to DSR.

Author

Figure 3 Pseudo-code for aspects of protocol MPDSR-2

Starting to transmit data by the source node

```
Upon receiving the first route reply packet start a timer;
if  (a second RREP is received from the destination)
        Choose a t route different from the previous route  (using the ScoreSimilarity function)
        used at this node to transmit the data;
        Start sending buffered packets;
endif
if ( timer is up)
        Start sending data packets;
endif
```

Originating RREPs

```
if  (RREQ from the source X is the first RREQ received)
        Start timer to count till τ milli-seconds;
endif
while  (receiving RREQs if there is time left on the timer)
        if (the destination node receives a RREQ)
                Save the paths obtained from the RREQ in the RREQ-route-cache ;
        else
                Re-flood the RREQ in to the network;
        endif
endwhile
if ( timer is up)
        Go to RouteSelecting function;
endif
```

RouteSelecting function

```
if (one route is in the cache)
        Initiate RREP for that;
elseif (at least two routes exist for the source)
        for (A certain number of  routes in cache)
           Give appropriate values to each pair of routes based on the ScoreSimilarity function;
        endfor

        Select two routes which have the smallest value of
        (ScoreSimilarity +  X*hopcount of route A  + X*hopcount of route B);
        Initiate RREPs for the selected routes;
endif
```

ScoreSimilarity function (used by both destination and source nodes)

```
  for (nodes in route A)
      for (nodes in route B)
          Calculate the similarity variable based on the number of nodes each routes have in
          common;
      endfor
   endfor
return the value of how similar routes are;
```

4.3 MPDSR-3

This algorithm was made more adaptive to the uncertainty levels in the acquired paths by considering the distance between the source and destination nodes. This was achieved by the destination sending a different number of route replies depending on the assumed distance between source and destination. Intermediate nodes can also send RREPs in MPDSR-3. Aspects of this protocol are captured in the pseudo-code of Figure 4.

The destination node sends the fastest path to arrive but waits until a timer has expired while it receives further routes. The average number of hops of the stored routes in the RREP cache at the destination is assumed to be a measure of the distance between the source and destination nodes. For closer proximities of source and destination, fewer route-replies are sent back (three replies were used in the tests), and for the further distances more routes are selected (if available) and sent back via RREPs (five replies were used in tests). This is intended to improve data delivery to further nodes by giving more alternative routes from the source node.

As MANETs are highly dynamic, and the conditions inside wireless networks can change so quickly, the routes previously obtained by RREPs (either from the destination or intermediate nodes), or the routes obtained in promiscuous mode can become out of date and, hence, unreliable. The use of routes from stale route caches causes more packet loss and more delay, because those routes are not up to date with the probable network topology.

To remove this problem, two different methods were tried in MPDSR-3: 1) a timer-based mechanism which removes stored routes after a certain amount of time; and 2) a feedback-based mechanism from the destination node to inform the sender of changing conditions along the route. These mechanisms are employed to proactively remove routes from the source's route cache.

4.3.1 Timer-based cache purging

Each route-reply packet carries a route-reply-type field that is determined by the RREP generator in order to allow the source node to distinguish between the RREPs acquired from the destination node and RREPs from intermediate nodes. The source node keeps the received routes in its route cache. Whenever a route is inserted in the route cache, a timer is started. The type of this timer is related to the type of the received route. The general aim is to remove routes other than those generated by the destination node at an earlier time. This is because those routes are less likely to be reliable, which is the reason why the extra route-reply-type information is utilized. Whenever a time

out occurs, the route cache at the source is searched for the matching route, which is deleted in order to avoid the risk of data delivery over an uncertain path.

In this variant of MPDSR-3, whenever a node intends to send data to a specific node in a different session but only after a long duration, it should retry to find fresh paths to the destination instead of relying on previous routes. At first sight, this is more likely to put more overhead traffic on the network by removing the tried paths and redoing route discovery. Nonetheless, using old routes has a higher probability of link breaks due to topology changes; therefore, the packet loss experienced will normally increase. Using fresh routes has another advantage. After topology changes, even if there are no link breaks on the routes, the routes might not be the shortest ones or optimal ones any more. There might be fewer routes still available due to topology changes, which can lead to poorer load balancing performance.

After the source node receives the first route-reply it starts to transmit data. Whenever there are more routes in its cache, this node selects further routes that are maximally-disjoint from previous routes, for the sake of load-balancing.

4.3.2 Feedback-based cache purging

The destination node, on receiving the first packet, starts saving the estimated signal-to-noise ratios (SNRs) of the incoming data packets for a certain number of times, say ten. After the tenth packet's SNR is received, the destination node calculates the average of these ten values as a criterion for the route-SNR delivery. Now the destination node has a value which can be used to predict the trustworthiness of the route for carrying packets. After the tenth packet, the SNR of all the received packets is compared with the average of the average SNRs. Whenever the SNR of a received packet falls below a certain fraction of the averaged SNR, (say) below 75%, an error-probability-counter variable is incremented as a measure of uncertainty of the carrier route in delivering packets. Then when the value of the error-probability-counter reaches a certain number, (say) three, the destination node generates a control packet and attaches the route of the last arrived packet to it, to tell the source node not to use that route again as the route is unreliable. This works like an RERR message, telling all nodes that receive it to delete the route the RERR is on.

The source node, upon receiving the first RREP starts sending the data packets. But unlike the timer-based MPDSR-3, the source node does not change to a fresh set of routes for transmitting data, but sends the data over the path in the route cache with the least number of hops. Thus like MPDSR-

1, one must make the distinction that this variety of MPDSR-3 exploits selection from multiple paths rather than simultaneously uses multiple paths. To continue, after losing the current route, it then uses other routes. This can lead to a greater probability of traffic congestion on some nodes; but the correct functioning of the implemented feedback system requires a gradual change in the status of a route. Route maintenance is not employed unless no routes are found in the route cache of the source node.

Figure 4 Pseudo-code for aspects of protocol MPDSR-3

Source node (time-based MPDSR-3)

Upon receiving the first route reply packet start transmitting data;
Start an expiry timer for the received route regarding its route type, to delete the route after a certain time;
if (more than one route available in the route-cache)
    Choose a route different from the previous route (using the **ScoreSimilarity** function) used at this node to transmit the data;
else
    Route the packets;
endif

Originating RREPs

if (RREQ from the source X is the first RREQ received in the new session according to its sequence number)
    Start timer to count till τ milli-seconds;
    Determine the RREP type;
    Send back the first RREP;
endif
while (receiving RREQs until time is left on the timer)
    if (the destination node receives the RREQ)
        Save the path obtained from RREQ in the RREQ-route-cache;
    else
        Re-flood the RREQ into the network;
    endif
endwhile
if (timer is up)
    No more RREQs accepted;
    Go to **RouteSelecting** function;
endif

RouteSelecting function

Author

```
if (two routes are in the RREQ-route-cache)
        Initiate RREP for the second one;
elseif (more than two routes && less than five routes exist for the desired destination)
        for (two times)
                for (A certain number of the routes in RREQ-route-cache)
                        Find the most disjoint route to the route sent using ScoreSimilarity
                        function;
                endfor
                        Wait for a while to avoid collision;
                        Determine the RREP type;
                        Initiate route reply for the selected disjoint route;
        endfor
elseif (more than four routes && average hop count <= X)
        for (two times)
                for (A certain number of the routes in the route cache)
                        Find the most disjoint route to the route sent using the
                        ScoreSimilarity function;
                endfor
                        Wait for a while to avoid collision;
                        Determine the RREP type;
                        Initiate route reply for the selected disjoint route;
        endfor
elseif (more than four routes && average hop count > X)
        for (four times)
                for (a certain number of the routes in route cache)
                        Find the most disjoint route to the route sent using the
                        ScoreSimilarity function;
                endfor
                        Wait for a while to avoid collision;
                        Determine the RREP type;
                        Initiate route reply for the selected disjoint route;
        endfor
endif
```

Feed-back generating function (in the feedback-based MPDSR-3)

```
if (packet SNR is higher than the accepting threshold)
        Get the DSNR;
        if (number of the received packets < 10)
                Add it to the aggregated DSNR;
        endif
        if (number of received packets = 10)
                Calculated the average DSNR;
        endif
        if (number of received packets > 10)
                if (DSNR >=average DSNR)
                        Do nothing;
                elseif(DSNR <average DSNR)
                        Increment the probable-error-variable;
        endif
        if ( the probable-error-variable > threshold for probable-error)
                Get the route in the data packet header;
                Initiate the feedback message;
        endif
        Get packet and prepare it for the MAC layer;
endif
```

The source node upon receiving the feedback message deletes and changes the route in use;

ScoreSimilarity function (used by both the destination and the source nodes)

```
  for (nodes in route A)
      for (nodes in route B)
          Calculate the similarity variable based on the number of nodes each routes have in
          common;
      endfor
   endfor
return the value of how similar routes are;
```

4.4 Linking function

It is possible to list the family of multi-path protocols MPDSR-1, -2, -3 by a common linking function in order to formalize the development summarized in Sections 4.1 to 4.3. The advantage of this approach is that it makes explicit the commonality between the family of multi-path routing protocols. The following is the signature of a programming function that could be called with appropriate parameters to implement any one of the three main members of the family:

```
FuncMultiPath(bool routeDiscovery, bool fastReply, int maxRoutes, int
destTimeout, bool intermedReplies, int intermedTimeout, bool gratuitous);
```

There are two parameter data types: `bool` or Boolean taking values true or false; and `int` or integer taking values from the set of natural numbers including zero. Then for each of the members of the family the function can be called as follows:

MPDSR-1 `FuncMultiPath(true, false, n, t1, true, t2, true);`

Author

MPDSR-2 `FuncMultiPath(true, false, n, t1, false, 0, false);`

MPDSR-3 `FuncMultiPath(true, true, n, t1, true, t2, false);`

Variables `n, t1, t2` are assumed to take values greater than zero except that in MPDSR-3, if t2 is set to zero and `intermedReplies` is `false` then feedback, as in Section 4.3.2, is assumed.

4.5 MPDSR-R

The effectiveness of multi-path protocols is highly dependent on the way that the protocol selects different routes for data transmission. The protocols introduced so far all try to use disjoint routes. MPDSR-R selects routes based on a reliability factor rather than less hops and disjointness of the routes. Aspects of this protocol are captured in the pseudo-code of Figure 5.

The proposed algorithm assumes that the nodes are equipped with a navigation system like a GPS or an Inertial Navigation System (INS). The reliability factor is defined to be:

$$RLBY = \frac{k - d}{v_{avg} + v'_{avg}}, \qquad (1)$$

where *k* ensures *RLBY* is correctly scaled, *d* is the distance from the previous node on the path, *v* is the current node's speed, and *v'* is the previous node's speed. This formula represents a cost function for route selection.

As a *RREQ* packet is relayed by different nodes, the *RLBY* parameter (carried in the packet header of the *RREQ* — and *RREP* — packets) is updated (aggregated) by each intermediate node using the location and the average speed of the node. The number which comes out of this formula is used at the destination as a measure of how reliable the route is based on the aggregation of reliability values of all the links the packet has passed over. At the destination, the reliability factor is used as the main metric in route selection. The selected routes are sent back to the source in the RREP packets.

The rationale behind this mechanism is that nodes that are in closer proximity with lower average speed can have more reliable links. Therefore, routes formed from more reliable links can be trusted more. The direction of movement of the nodes is also of significance for approximating the *RLBY*. However, as this information was not available when running tests, the assumption was made that the two nodes participating in the current link are moving in the opposite direction, which is the worst case.

The propagation of an *RREQ* message inside the network is like DSR. However, as previously mentioned the *RLBY* is updated as the message passes across the network. When the destination node gets the first *RREQ* packet asking for a communication session, starts to collect the routes from route-request packets for a certain amount of time. After that time is up, the node sorts the routes based on their reliability factor. Then, it sends a pre-set number of routes back to the source node. In tests, two paths were returned.

The intermediate nodes are not allowed to generate RREPs, as these may make misleading predictions of the reliability. The source node upon receiving the first route-reply from the destination starts sending its data packets. In the case of having more than one route in the route cache, the source tries each route in turn to send packets over different routes. Despite the fact that there is no disjoint parameter exploited for selecting routes, it is still worth while distributing the traffic amongst available paths.

Having designed the family of protocols, the next step is to systematically evaluate their network performance.

4 Evaluation

4.1 Simulation environment

The Global Mobile System Simulator (GloMoSim) [20] simulation library was employed to generate our results. The simulation parameters now discussed are gathered for convenience in Table 1. GloMoSim was developed based on a layered approach similar to the OSI seven-layer network architecture. IP framing was employed with UDP transport. Each result was obtained by running the GloMoSim simulator for 60 runs and taking the arithmetic mean. In protocols in which more than one route is accumulated at the destination, two routes were accumulated. For evaluation of the results, three metrics of average end-to-end delay, total packet loss ratio, and control overhead traffic were used. 20 Constant Bit-Rate (CBR) communication sessions were set-up by randomly selecting source and destination node, each session consisting of 500 B packets transmitted at 20 packets/s. The node buffer size was defaulted to 100 packets. The simulation time was 600 s, with 100 s to reach steady state. The scenarios differ in the node density and the maximum speed of the nodes in order to evaluate the performance of the proposed algorithm under different conditions. In the results for node speeds, the number of nodes was fifty. Node density is defined as the number of nodes per total

terrain area. When node density is examined then the maximum speed was 15 m/s (with the minimum still 1 m/s).

Figure 5 Pseudo-code for aspects of protocol MPDSR-R

Originating RREQs

```
if (buffered packets to be sent && no routes are known to the destination)
        Initiate Route Request packets, include the current location of the node && the average
        speed of the node in the RREQ, and set the value for the RLBY parameter;
        Broadcast the RREQ;

endif
```

Receiving RREQ's and originating RREP's

```
if (this node is not the destination node)
        Calculate the RLBY factor and add it to the RLBY field in the RREQ;
        Update the location and average speed in the RREQ;
        Relay the RREQ;
endif
if ( this node is the intended destination node for the RREQ && RREQ from the source X is the
first RREQ in a new session)
        send RREP with the fastest route;
        Start timer to count till τ milli-seconds;
endif
while (receiving RREQs if time is left on the timer)
        if (the destination node receives an RREQ)
                update the RLBY;
                Save the paths obtained from RREQs in the RREQ-route-cache structure with
                their related RLBY factor;
      else
                Re-flood the RREQ into the network;
        endif
endwhile
if ( the timer is up)
        No more RREQs accepted;
        Go to RouteSelecting function;
endif
```

RouteSelecting function

```
if (one route is in a cache (whether the original route-cache or in the RREQ-route-cache))
        Initiate RREP for that;
elseif (at least two routes exist for desired source)
        Sort the routes in the cache using bubble sort algorithm regarding their RLBY factor;
        Initiate RREPs for two most reliable routes;
endif
```

```
Starting to transmit data by the source node
Upon receiving the first route-reply packet start sending data;
if (second RREP is received from the destination)
        Choose a route different from the previous route (Using the ScoreSimilarity function)
        used at this node to transmit data;
        Start sending buffered packets;
endif
if ( timer is up)
        restart the Route discovery by sending RREQ's and restarting the timer;
endif
ScoreSimilarity function (used at the destination node)
  for (nodes in route A)
      for (nodes in route B)
          Calculate the similarity variable based on the number of nodes each routes have in
          common;
      endfor
   endfor
return the value of how similar routes are;
```

The MAC layer technology used in the experiments was IEEE 802.11b Distributed Coordination Function (DCF) with Request to Send/Clear to Send (RTS/CTS) support and data capacity of 2 Mbps. The transmission range was 250 m. Two-ray path loss model was used as the path loss model with a Friss free-space model with parameters (exponent, sigma) = (2.0, 0.0) for near line-of-sight and plane-earth path loss (4.0, 0.0) for far line-of-sight. The probabilistic Rician fading model (k = 4.7 dB) was applied additively to the deterministic two-ray path loss model to include the effect of fading. The Rician fading model assumes a line-of-sight component. The heights of omni-directional antennas were all equal and set to 1.5 m. The familiar random waypoint mobility model [21] defined the movement of different nodes in the terrain. The minimum speed of each node was set to 5 m/s and the pause time between each movement was set to 5 s. The terrain size was set to 1 km^2.

Some general remarks on the metrics employed in the simulation are in order. Real-time services such as video streaming are most affected by the packet loss rate (number of data packets lost to number of data packets sent), as packets cannot be re-sent. The use of multiple paths is of benefit, because despite the out of order delivery of packets to the destination, more packets are delivered in total, as link breakages on separate paths causes fewer packet losses when a traffic flow is distributed across multiple paths. Due to the use of several routes for transmitting consecutive packets, although the average hop count of the chosen routes is probably more than the shortest route chosen by DSR,

Table 1. Simulation parameter settings

Parameter	*Value*
Terrain size	1000 x 1000 m^2
Default no. of nodes	50
Default max. speed	15 m/s
Default min speed	1 m/s
Mobility model	Random waypoint [21]
Pause time	5 s
Path loss model	Two-ray ground
Propagation model	Rician (k = 4.7 dB)
Antenna height	1.5 m
Antenna type	Omni-directional
Range	250 m
Upper layers headers	IP/UDP
Wireless MAC	IEEE 802.11 DCF+RTS/CTS
Data rate	2 Mbps
Runs per data point	60
Simulation time	600 s (100 s start-up)
Packet size	500 B

the congestion and buffer overflow on some intermediate nodes is less, so that packets stay a shorter time in the network. Therefore, the average end-to-end delay may be improved. Besides, in single-path protocols like DSR, failure of the routes is more likely as there is no quality factor for the routes in the cache of the nodes. Hence, the packets spend more time in travelling, because of route changes from using packet salvaging mechanisms.

Another important issue in MANETs is the overall control packet overhead percentage. As nodes need to broadcast many control packets at different stages of the communication sessions, in route discovery and route maintenance. This implies a considerable traffic overhead on the network which participates in collisions, congestion and the wastage of the limited resources available in ad hoc networks. For this reason, it is desirable to limit or to optimize the number of propagating control packets to a number which is adequate for the needs of specific scenarios. In particular, if energy consumption is a concern then a reduced number of control packets is desirable.

4.2 Comparisons with DSR

In Figure 6a, MPDSR-1 shows less data packet end-to-end delay than DSR, as the paths used for data transfer are more reliable. Consequently, delays caused by link breakages are infrequent. Equally in Figure 6b, MPDSR-1 performs better as its routes are more stable, and many of the longer routes

are not sent back by the destination. In Figure 7a, with increase in node speed MPDSR-1 gives better data packet delivery, as it uses less control packet overhead (see Figure 8) with more reliable routes. In Figure 7b, as node density increases, routes can have more hops. This leads to a greater probability that routes will be broken. MPDSR-1 has better performance in respect to packet loss, as the control overhead traffic of *RREP*s is limited and the routes are more reliable. Figure 8a confirms that in MPDSR-1 there is less control overhead as destination nodes are restricted in sending *RREP*s. However, from Figure 8b, increased node density has a significant effect on the overhead traffic because of the many broadcasts. This is a severe problem for DSR as well as MPDSR-1.

Figure 6a and 7a indicate that at speeds around 5 m/s and below the advantage of MPDSR-1 over DSR diminishes. DSR's maintenance of an indiscriminate route cache is beneficial for networks with low mobility, as at these speeds routes will be valid for a longer period. When routes do fail, due to low mobility, there is a greater selection of valid cached routes available than for MPDSR-1. At higher speeds, DSR's indiscriminant selection of cached routes is likely to lead to more stale routes in caches.

At lower speeds it is generally more difficult to make connections with new nodes leading to a general drop in performance, which is reflected in the increasing delay and packet losses for both protocols under test. This behavior occurs in other later protocol comparisons, resulting in better performance at 5 ms than speeds lower or higher. Because this is a general observation, it is not repeated again.

Another phenomenon is that from 25 m/s to 30 m/s in Figure 6a end-to-end delay slightly decreases and for MPDSR-1 in Figure 7a packet loss does not increase. This suggests that increasing delay and packet loss due to stale routes as mobility increases is countered at high speed by the ability to more quickly re-make routes, which leads both to a reduction in delay and a reduction in lost packets while the route is repaired.

Author

Figure 6 Data packet end-to-end delay of MPDSR-1 and DSR in terms of (a) node speed (b) node density

(a) (b)

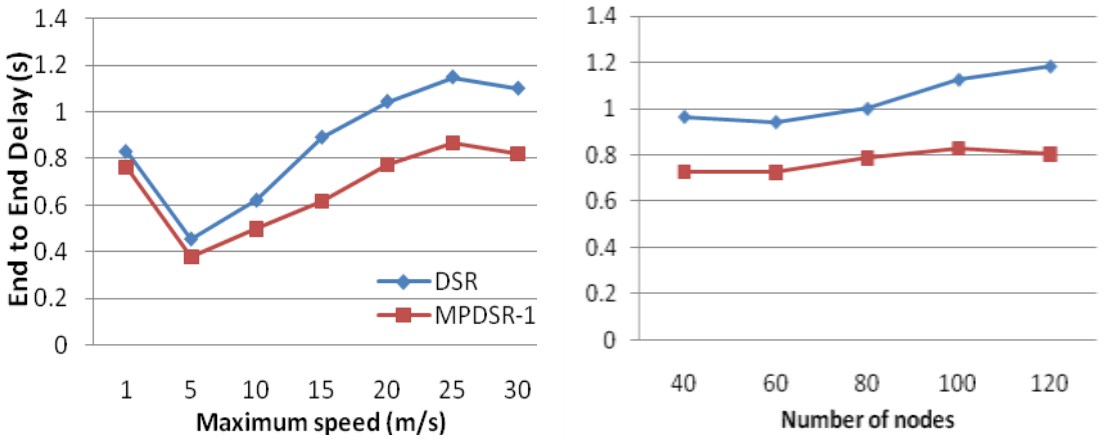


Figure 7 Data packet loss ratio of MPDSR-1 and DSR in terms of (a) node speed (b) node density

(a) (b)

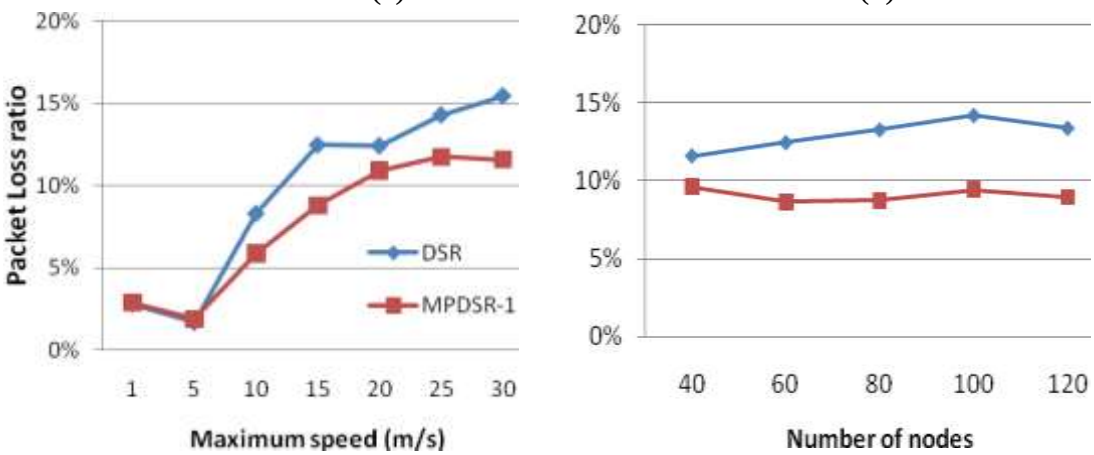
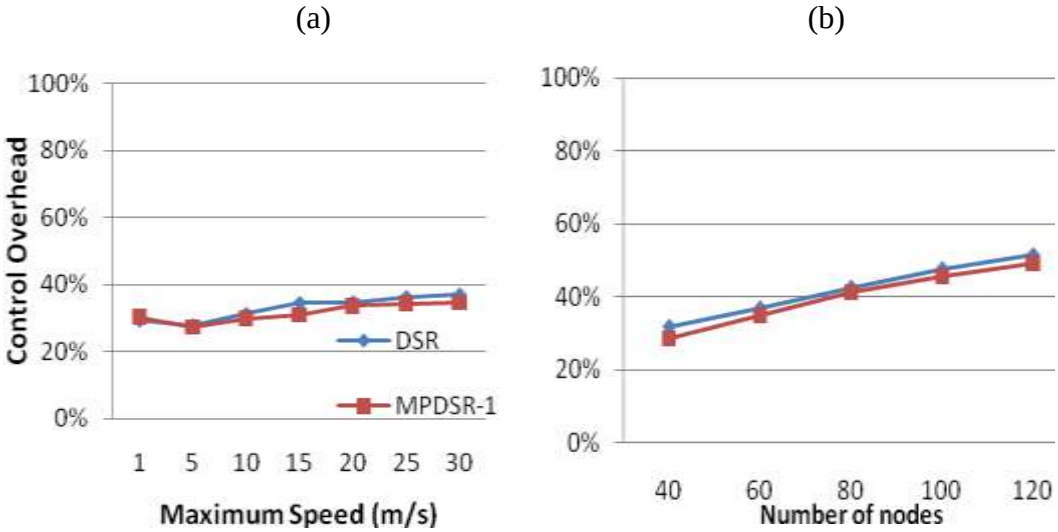


Figure 8 Control overhead of MPDSR-1 and DSR in terms of (a) node speed (b) node density



From Figure 9a, the average packet delivery latency for MPDSR-2 and DSR is almost the same, because MPDSR-2 generates more reliable routes which means less link breaks will occur on average, a consequence of which is that packets stay longer in the network. Again, in Figure 9b, as node density increases, paths on average have more hops with increased delay. More reliable routes decrease packet loss, see Figure 10, and, as node density increases, the decrease is considerable. However, as remarked already in respect to the comparison between DSR and MPDSR-1, at low speeds (in the region of 5 m/s and below) DSR reduces delay and packet loss. This is because DSR gains the advantage of route caching without suffering the disadvantage of stale cached routes. Non-local repair of broken routes also hampers DSR's general performance at higher mobility. Just as in Figure 7a, between 25 m/s and 30 m/s there is a slight decrease in the end-to-end delay experienced by DSR. The same explanation as for Figure 7a, i.e. speed of repair countering increase of stale routes, can be applied.

MPDSR-2 generates more control packets than DSR, as intermediate nodes are not allowed to respond to RREQ packets. Therefore, each control packet remains in the network until it reaches the intended destination. Thus, packet loss decreases in MPDSR-2 at the expense of greater overhead, Figure 11, and longer latencies.

Author

Figure 9 Data packet end-to-end delay of MPDSR-2 and DSR in terms of (a) node speed (b) node density

(a) (b)

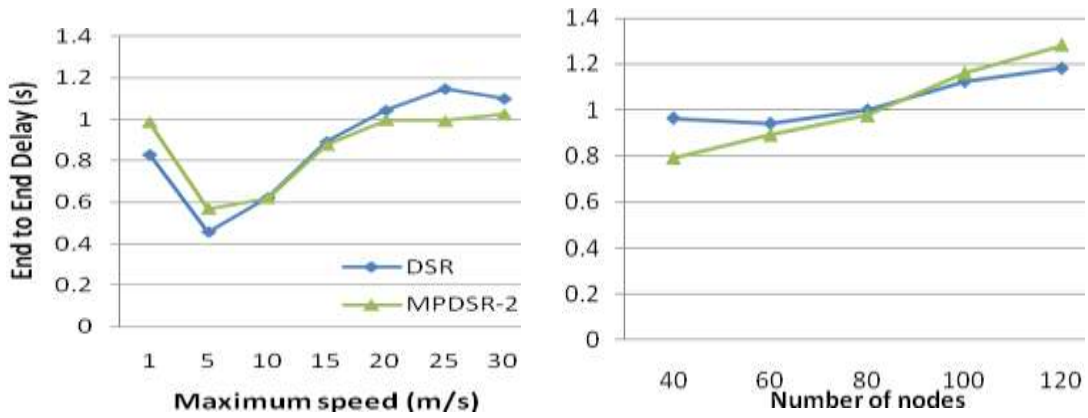


Figure 10 Data packet loss ratio of MPDSR-2 and DSR in terms of (a) node speed (b) node density

(a) (b)

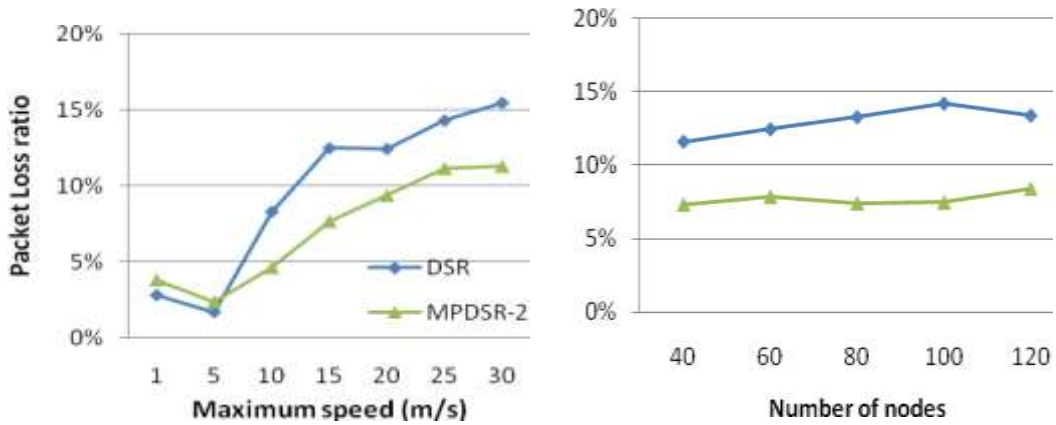
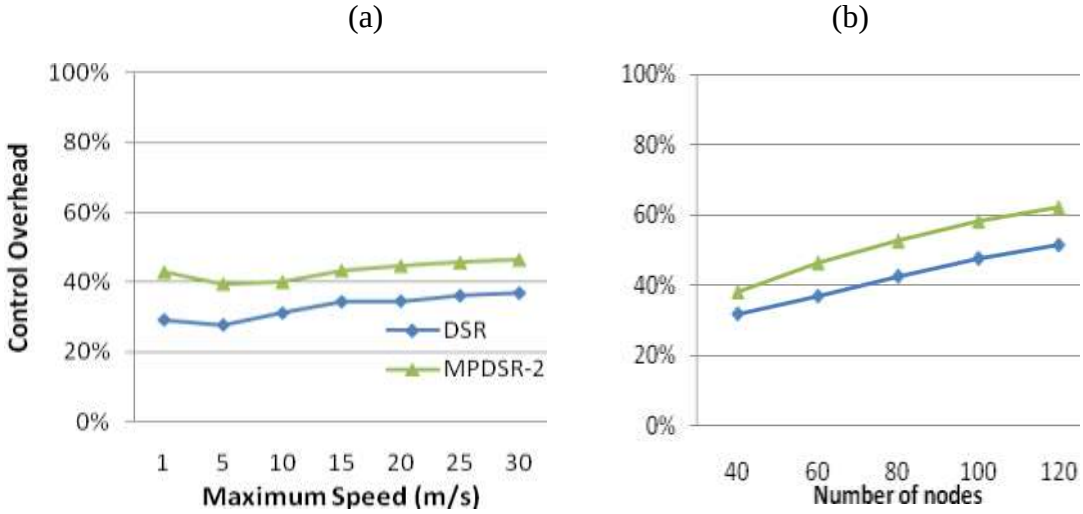


Figure 11 Control overhead of MPDSR-2 and DSR in terms of (a) node speed (b) node density



Both versions of MPDSR-3 generally deliver packets faster than DSR, Figure 12, as the source starts data transmissions as soon as it gets the first route reply. In that respect, as a result of feedback messages, MPDSR-3 outperforms other members of the protocol family. However, at 5 m/s, DSR shows a slight improvement over MPDSR-3 with feedback, for the same reason that has been remarked upon earlier in regard to MPDSR-1 and MPDSR-2. There is also a significant improvement in packet loss rates in all but low mobility, refer to Figure 13, from using MPDSR-3. As nodes do not employ routes from stale route caches, the routes are more reliable and, again, spreading a traffic flow over different paths leads to less congestion and collisions. The adaptive route replying mechanism also gives more reliability to communication over longer distances. At lower mobility, the disadvantage of stale route caches is much reduced which again explains the advantage of DSR around and below 5 m/s. However, from Figure 14 overhead traffic is higher in the MPDSR-3 protocol, as there are more route-replies sent back. The overhead traffic is the result of MPDSR-3's periodic updates and its feedback messages. Therefore, route discovery messages are an important reason for the relative growth in overhead. In fact, increased node density causes a rapid rise in overhead traffic, due to the number of nodes broadcasting control packets.

Figure 12 Data packet end-to-end delay of MPDSR-3 and DSR in terms of (a) node speed (b) node density

(a) (b)

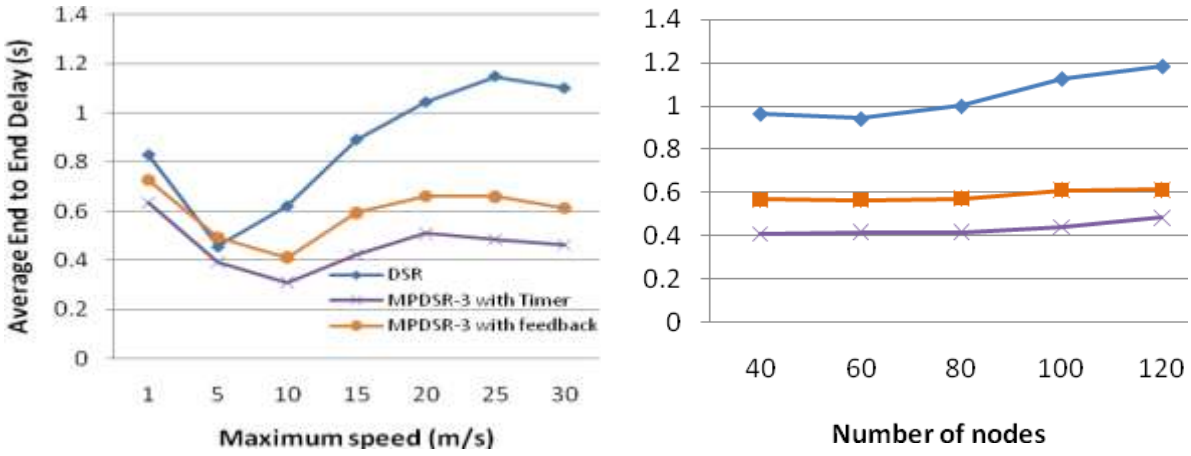


Figure 13 Data packet loss ratio of MPDSR-3 and DSR in terms of (a) node speed (b) node density

(a) (b)

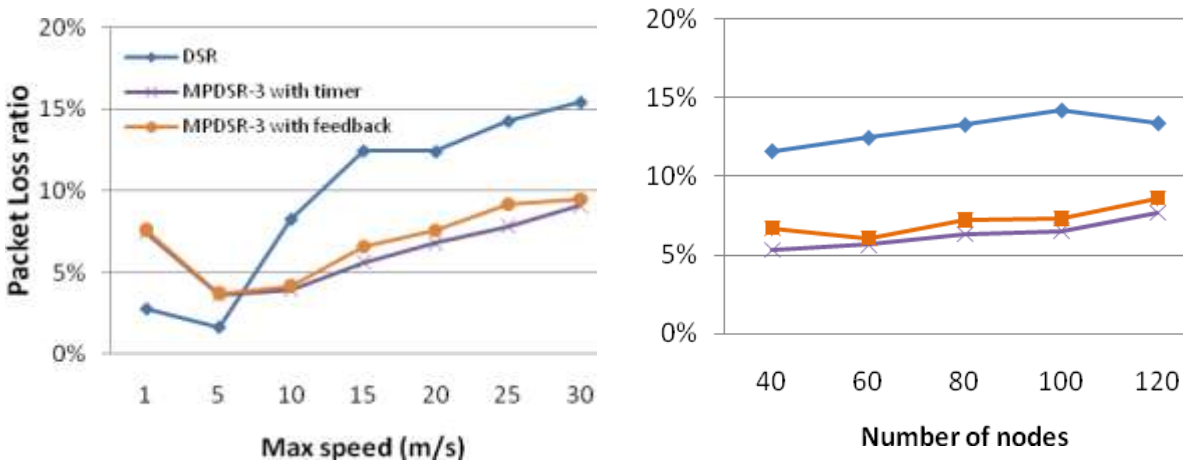
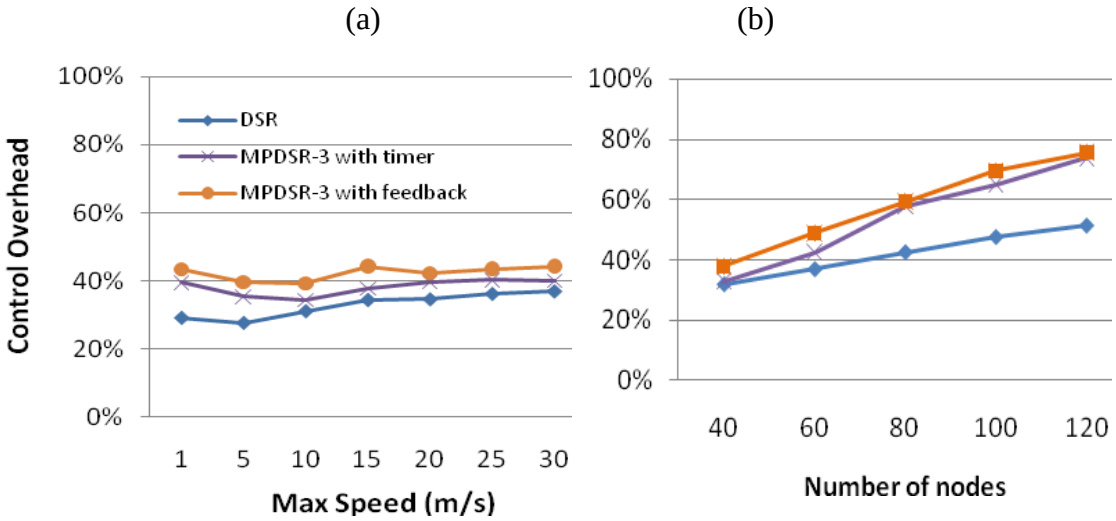


Figure 14 Control overhead of MPDSR-3 and DSR in terms of (a) node speed (b) node density



In MPDSR-R, routes are formed according to the physical proximity of nodes. Therefore, it should not be a surprise, Figure 15, that delay is reduced, as the number of hops is reduced. Again, there is a slight decrease in delay beyond 25 m/s for the reason mentioned in rspect to the other protocol comparisons. MPDSR-R generally reduces packet loss in an uncrowded network but an increase in node density is a problem for this member of the protocol family, because there is an increase in control traffic leading to collisions and congestion. This is clear from a comparison between Figure 16b and 17b. Again though, use of DSR results in lower packet loss than MPDSR-R in a low mobility network, because routes are more stable at these speeds.

Author

Figure 15 Data packet end-to-end delay of MPDSR-R and DSR in terms of (a) node speed (b) node density.

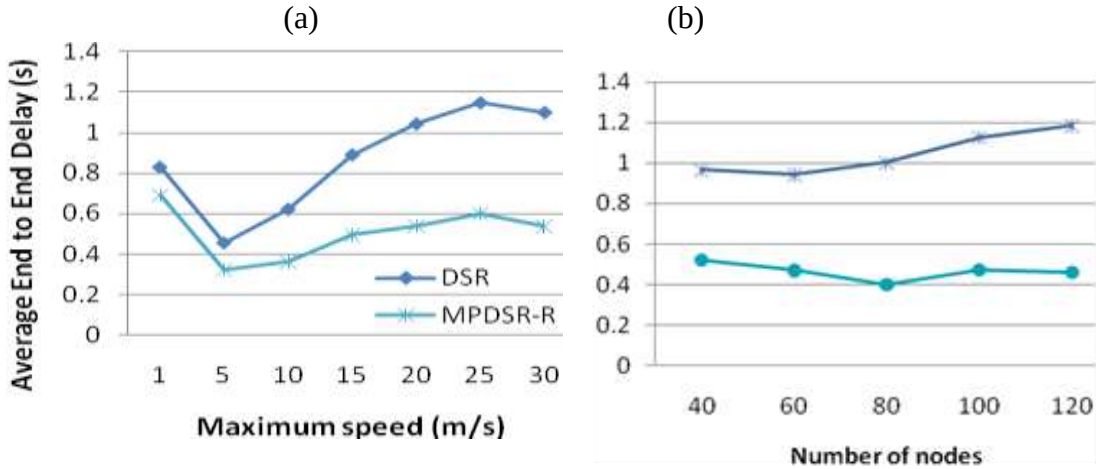


Figure 16 Data packet loss ratio of MPDSR-R and DSR in terms of (a) node speed (b) node density.

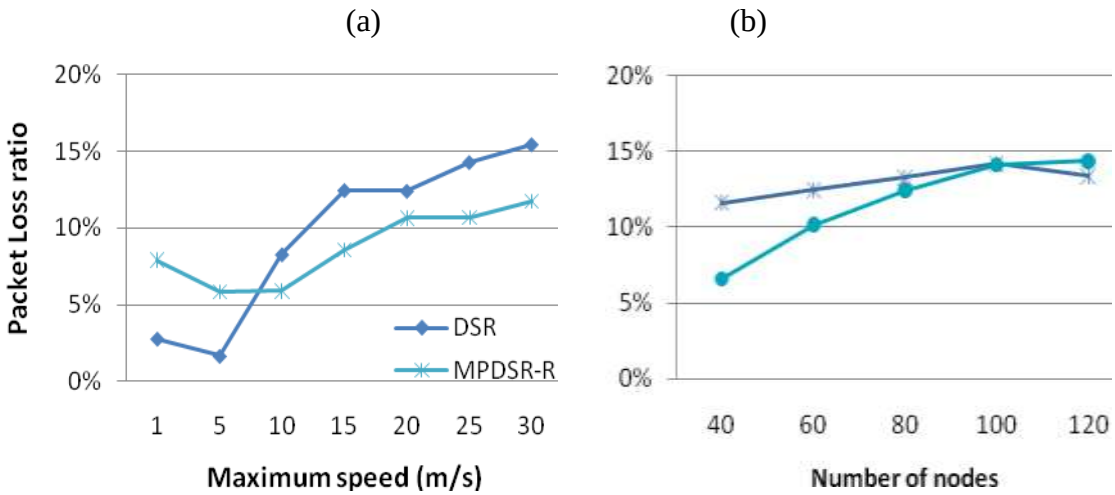
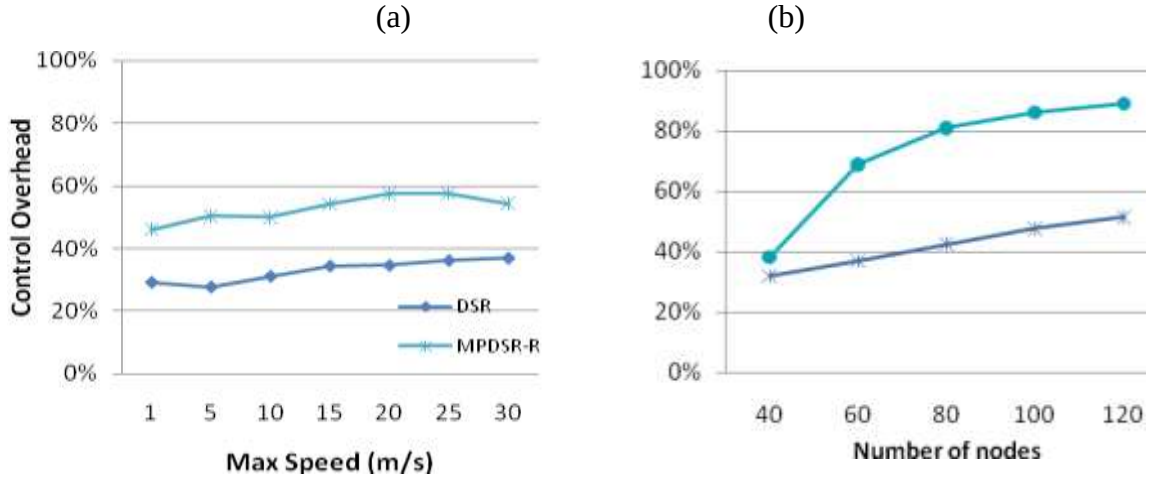


Figure 17 Control overhead of MPDSR-R and DSR in terms of (a) node speed (b) node density.



5 Conclusions

This paper has compared the relative performance of single-path routing under DSR with multi-path routing by means of the proposed MPDSR family. There is probably no one ideal routing protocol exploiting multiple paths, as hopefully the discussion in this paper exposes. This is because the variety of possible applications and network configurations does not allow there to be an all-purpose protocol. By selecting from the techniques that have been applied in multi-path protocol research it is possible to construct a set of related routing protocols. Hopefully, this paper's analysis has shown how it might be possible to design a protocol family each member of which emphasizes a different aspect, such as further reducing end-to-end delay at a cost in reliability. The paper has confirmed that in terms of packet loss rates and latency, each of the members of the MPDSR protocol family outperforms DSR at moderate to high speeds. Conversely, the main advantage that DSR brings at these speeds is a reduction in control packet overhead, implying a reduction in battery usage in transmission. However, even in the latter respect, the distribution of control packets may provide better load balancing than single-path DSR.

Our work in this paper is at the network layer, which is sensible if the MANET is intended for a general traffic types. To achieve a further improved outcome it is necessary to communicate between different layers of the network, which is sensible if the MANET is dedicated to a particular application such as some form of media streaming. This implies tight information exchange between

the application layer and lower layers due to the variation in the data content in streaming. Besides, wireless networking improves if communication is more adapted to the channel conditions.

References

[1] Mueller, S., Tsang, R. P. and Ghosal, D. (Apr. 2004). Multipath routing in mobile ad hoc networks: Issues and challenges. LNCS 2965, pp. 209 – 234.

[2] Javan, N. T. and Deghan, M. (2007). Reducing end-to-end delay in multi-path routing algorithms for mobile ad hoc networks. *Mobile Ad-Hoc and Sensor Networks*. LNCS 4864, pp. 715-724.

[3] Leung, R., Liu, J., Poon, E., Chan, A.-L.C., and Li, B., (2001). MP-DSR: A QoS-aware multipath dynamic source routing protocol for wireless ad-hoc networks. 26th IEEE Ann. Conf. on Local Computer Networks (2001) pp. 132-141

[4] Liao, W.-H., Tseng, Y.-C., Wang, S.-L., and Sheu, J.-P. (2001). A multi-path QoS routing protocol in a wireless mobile ad hoc network. *IEEE Int. Conf. on Networking*, pp. 158-167.

[5] Ganesan, D., Govindan, R., Shenker, S. and Estrin, D. (2002). Highly resilient energy efficient multipath routing in wireless sensor networks. *2nd ACM Int. Symp. on Mobile Ad Hoc Networking & Computing*. pp. 11 - 25.

[6] Hashim, R., Nasir, Q. and Harous, S. (2007). Congestion aware multi-path Dynamic Source Routing protocol (CAWMP-DSR) for mobile ad-hoc network. *Int. Conf. on Advances in Mobile Multimedia*. pp. 199-206.

[7] Lee, C. K-L., .Lin, X.-H., and Kwok, Y.-K. (May 2003). A multipath ad hoc routing approach to combat wireless link insecurity. *IEEE Int. Conf. on Communications*, pp. 448–452.

[8] Ghanbari, M. (2003). *Standard Codecs: Image Compression to Advanced Video Coding*. IEE (now IET), Stevenage, UK.

[9] Qadri, N. Fleury, M., Altaf, M., Rofoee, B. R., and Ghanbari, M. (2009). Resilient P2P multimedia exchange in a VANET, *IEEE IFIP Wireless Days Conf.*

[10] Krishnan, R., Silvester, J.A. (1993). Choice of allocation granularity in multipath source routing schemes. *IEEE INFOCOM*, pp. 322-329.

[11] Lin, S., Wang, Y., Mao, S. and Panwar, S. (2002). Video transport over ad-hoc networks using multiple paths. *IEEE Int. Symp. on Circuits and Systems*. pp. 57-60.

[12] Li, X. and Cuthbert, L. (2004). Node-disjointness-based multipath routing for mobile ad hoc networks. *ACM Int. Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, pp. 23-29.

[13] Lee, S.-J. and Gerla, M. (2001). Split multipath routing with maximally disjoint paths in ad hoc networks. *IEEE Int. Conf. on Comunnications*, pp. 3201–3205.

[14] Johnson, D.B. and Maltz, D.A. (1996). Dynamic Source Routing in ad hoc wireless networks. *Mobile Computing*, vol. 353, pp. 153-181.

[15] Toussaint, M. T., (Aug. 2003). Multipath routing in mobile ad hoc networks. Tech. Report, Technical University of Delft, Netherlands.

[16] Parissidis, G., Lenders, V., May, M., and Plattner, B. (2006). Multi-path routing protocols in wireless mobile ad hoc networks: A quantitative comparison . *6th Int. Conf. on Next Generation Teletraffic and Wired/Wireless Advanced Networking*, pp. 313-326.

[17] Marina, M. K. and Das, S. (2001). On-demand multi-path distance vector routing in ad hoc networks. *IEEE Int. Conf. on Network Protocols*. pp. 14–23.

[18] Perkins, C. E. and Royer, E. M. (Feb. 1999). Ad-hoc On-Demand Distance Vector Routing. *IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90-100.

[19] Ye, Z., Krishnamurthy, S. V., and Tripathi, S. K. (2003). A framework for reliable routing in mobile ad hoc networks. *IEEE INFOCOM*, pp. 270-280.

[20] Zeng, X., Bagrodia, R., and Gerla, M. (May 1998). GloMoSim: A library for parallel simulation of large-scale wireless networks. *12th Workshop on Parallel and Distributed Simulations*, pp. 154-161.

[21] Broch, J., Maltz, D.A., Johnson, D. B., Hu, Y.-C. and Jetcheva, J. (Oct. 1998). A performance comparison of multi-hop wireless ad hoc network routing protocols. *ACM Mobicom*, 85-97.

[22] Wiegand, T., Sullivan, G. J., Bjøntegaard, G. and Luthra, A. (2003). Overview of the H.264/AVC video coding standard, *IEEE Trans. on Circuits and Syst. for Video Technol*, vol. 13, no. 7, pp. 560-576.

[23] Wenger, S. (2003). H264/AVC over IP. *IEEE Trans. on Circuits and Syst. for Video Technol.*, vol. 13, no. 7, pp. 645-656.