

Achieving Spatial Disjointness in Multipath Routing without Location Information

Juan J. Gálvez
DIIC, Computer Science Faculty
University of Murcia, Spain
Email: jjgalvez@um.es

Pedro M. Ruiz
DIIC, Computer Science Faculty
University of Murcia, Spain
Email: pedrom@dif.um.es

Antonio F. G. Skarmeta
DIIC, Computer Science Faculty
University of Murcia, Spain
Email: skarmeta@dif.um.es

Abstract—We develop an on-demand multipath routing protocol for multi-hop wireless networks (MWNs), capable of finding spatially disjoint paths without the need of location information. Multipath routing can enable various applications and enhancements in MWNs, such as load balancing, bandwidth aggregation, reliability and secure communications. The use of spatially disjoint routes is important to effectively achieve these gains, due to the non-interfering nature of the paths. Most of the proposed multipath protocols for MWNs focus on reliability and do not find spatially disjoint paths. We propose a new on-demand protocol called Spatially Disjoint Multipath Routing (SDMR), capable of finding multiple paths in one route discovery, measuring the distance between them and choosing paths with most separation. Simulations demonstrate the effectiveness of the protocol in finding spatially separate routes.

I. INTRODUCTION AND MOTIVATION

Multi-hop wireless networks (MWNs) are collections of autonomous nodes capable of dynamically establishing a multi-hop wireless network without the need of an existing infrastructure. They can support a dynamic topology and can have significant resource constraints, such as a shared wireless channel. Developing routing protocols that react to the dynamic nature of the network while efficiently using its resources is a challenging task.

Multipath routing permits the establishment of multiple paths between a source and a destination. This can provide various benefits over single path protocols in MWNs, increasing reliability, enabling load balancing and bandwidth aggregation, and secure communications.

In dynamic networks with mobility, the knowledge of multiple routes can provide reliability when a route breaks. The use of this strategy has also been shown to reduce route discovery overhead in on-demand protocols, because nodes can switch to alternate paths when the current path fails, thus avoiding a new route discovery which accounts for the majority of protocol overhead [1]. Spatial separation between alternate paths can further increase reliability, by gaining tolerance to situations of regional failure, where all paths that cross a specific region fail (e.g. due to a blackout).

Load balancing permits a source to send data simultaneously along multiple paths, providing a higher aggregate bandwidth to cope with the limited capacity of MWNs, or to better balance the load across the network. To effectively achieve load balancing in MWNs, the problem of *route coupling* has

to be addressed. Route coupling refers to the interference between two or more paths which are physically close. In most multipath protocols alternate paths tend to be very close to each other [2], making it difficult to achieve the goals of balancing load or aggregating bandwidth.

The knowledge of spatially disjoint routes enables the effective exploitation of multipath routing. It can help avoid radio collisions between alternate paths (useful in load balancing and bandwidth aggregation), avoid regional failures (translates into better reliability) and can provide secure routing. Situations that threaten the security of a communication include interception of data, man-induced geographically localized failures or black-hole routers. The use of spatially disjoint routes can help in these situations (e.g. by encrypting communication and distributing across multiple separate paths, thus reducing the probability of full interception).

In this paper we develop a new on-demand multipath protocol based on source routing called SDMR, capable of finding spatially disjoint paths. The protocol can discover multiple paths between a source and multiple destinations in one route discovery and measure the distance between the paths. The route discovery mechanism enables the discovery of paths that traverse all geographical regions (not only shortest routes). We define a new metric based on hop distance to measure the separation between paths. In this paper we explain the protocol and study its effectiveness.

The rest of the paper is organized as follows. In section II we review related work. Section III explains the SDMR protocol. Section IV shows and analyzes protocol performance based on simulations with $ns-2$. Finally, in Section V we summarize our conclusions and discuss some future work.

II. RELATED WORK

Most proposed multipath protocols for MWNs focus on providing fault tolerance, i.e. the capability of switching to alternate routes when a route fails. Recent examples based on AODV [3] are AODVM [4], AOMDV [1] and the proposal in [5]. Other protocols use source routing such as a multipath extension of DSR [6] and SMR [7]. These protocols only consider link or node disjointness of paths (i.e. paths share no links or nodes, respectively), as this is generally sufficient from a fault tolerance perspective. Therefore they don't find

spatially disjoint routes, and in practice routes found are very close to each other.

Route coupling between paths in geographic proximity of each other severely limits the gains of load balancing in MWNs. The effects of load balancing and route coupling have been studied in [2], [7]–[10].

To our knowledge few protocols explicitly attempt to find spatially disjoint paths. Most existing ones are location based and as such use geographic coordinates to measure the distance between paths. Examples are EFR [11], which applies the concept of electric field lines to find spatially disjoint routes, GMP [12] and the work in [13]. The drawback of location based protocols is that they rely on geographic information which is not always present.

A few metrics have been defined to measure the relative degree of interference between paths, namely *coupling* [8] and *correlation* [9]. However, most research concerning these metrics only studies the effects of route coupling on performance. In addition, these metrics present drawbacks. First, they are limited in the degree of separation they can measure. *Correlation* only attempts to avoid interference between paths, i.e., it only considers if nodes of one path are in interference range of nodes of the other path. And their definition of correlation implicitly assumes that the interference range is equal to the transmission range, which generally isn't true. Also, *correlation* is only valid for node-disjoint routes. *Coupling* is difficult to calculate in practice, and again only attempts to avoid interference between paths. In some applications it is desired that paths be further apart than interference range (e.g. to avoid eavesdropping from a node between the two paths).

AODVM/PD [14] is an on-demand multipath protocol based on AODVM [4] that attempts to minimize *correlation*. Besides the limitations of using the *correlation* metric, another drawback of AODVM/PD is that due to the route discovery mechanism of this protocol, it first chooses the shortest path and then attempts to find additional shortest paths with low correlation. This also limits the degree of separation obtainable.

Our distance metric can provide more information regarding the degree of separation compared to the above mentioned metrics, and can be used for both disjoint and non-disjoint routes. Also, our protocol can discover and choose from paths that traverse all geographical regions of the network (not only k shortest paths).

III. SPATIALLY DISJOINT MULTIPATH ROUTING PROTOCOL

In this section we develop the SDMR protocol.

A. SDMR overview

SDMR is a reactive multipath protocol capable of finding spatially disjoint paths which uses source routing. For a source node S to calculate disjoint paths to a destination D , it first needs a graph of the connectivity of the network, in a similar way to link-state routing protocols. S produces this topology graph using connectivity information received from other network nodes. The source node then searches the graph for candidate paths between itself and the destination and

chooses the set of most disjoint paths according to a distance metric.

The route discovery method is thus similar to other link-state routing protocols like OLSR [15], but the main difference stems from the fact that in OLSR all nodes periodically broadcast their connectivity information (link-state) to all nodes in the network, whereas in SDMR this information is only sent to a source node on demand. To achieve this reactively, S requests connectivity information to the set of Multipoint Relays (MPR) in the network and to the destination by flooding a Topology Request (TREQ) message, and these nodes send their 1-hop neighbor set back to the source in Topology Reply (TREP) messages, along the reverse paths formed during TREQ propagation.

Data packets are routed along the paths using source routing.

B. Definitions

G_n	Undirected topology graph of node n used to find and calculate disjoint paths.
$nbset_seqnum_n$	Neighbor set sequence number of n . Incremented each time a change occurs in its 1-hop neighbor set.
$last_nbset_seqnum_n^d$	Value of $nbset_seqnum_n$ when n last sent its 1-hop neighbor set to d .
$pending_trep_timer_n^d$	If this timer at node n expires, n must issue a TREP to d .
$parent_n^d$	Parent node of n on the spanning tree rooted at d .

C. Neighbor detection

In SDMR nodes periodically send HELLO messages. The information exchanged allows nodes to discover their 1-hop and 2-hop neighbors. Based on this information nodes select their Multipoint Relays (MPR). This functionality is the same as in OLSR, refer to this protocol for details.

Whenever the 1-hop neighbor set of a node n changes, n increments $nbset_seqnum_n$.

D. Path discovery

If S needs paths to D and its topology graph G_S has expired, it must discover a new topology graph. If the topology graph is fresh it can directly calculate disjoint paths (see section III-E).

1) *Topology Request*: To build a new topology graph, S initiates a network-wide flood of a Topology Request (TREQ) by broadcasting the message to its neighbors. A TREQ contains the following fields:

source_addr	broadcast_id	dest_addr	parent
-------------	--------------	-----------	--------

The pair ($source_addr$, $broadcast_id$) uniquely identifies a TREQ. $broadcast_id$ is incremented whenever S issues a new TREQ. When a node receives a TREQ from a neighbor

with a $(source_addr, broadcast_id)$ pair not seen before, it records the neighbor as the parent node to reach S . The *parent* field of a TREQ stores the parent of the last node that forwarded the TREQ. The TREQ will be flooded by MPRs, forming reverse paths in the network to S , which results in a spanning tree rooted at S . The use of MPRs reduces the number of redundant retransmissions in the same region while at the same time enabling all nodes in the network to receive the TREQ, thus reducing overhead.

Algorithm 1 details the steps followed by nodes upon receiving a TREQ.

Algorithm 1 A node n receives a TREQ from neighbor m .

```

1: if  $n \neq D$  then // is not target
2:   if  $n$  has not heard  $(source\_addr, broadcast\_id)$  then
3:     Cache  $(source\_addr, broadcast\_id)$ 
4:      $parent_n^S := m$ 
5:     if  $n$  is a multipoint relay of  $m$  then
6:       Set  $pending\_trep\_timer_n^S$  to  $\delta_r$  ms
7:        $treq.parent := m$ 
8:       Forward  $treq$ 
9:   else
10:  if  $pending\_trep\_timer_n^S$  and  $treq.parent = n$  then
11:    Set  $pending\_trep\_timer_n^S$  to  $\Delta_r$  ms
12:  else // is target
13:    if  $D$  has not heard  $(source\_addr, broadcast\_id)$  then
14:      Cache  $(source\_addr, broadcast\_id)$ 
15:       $parent_D^S := m$ 
16:      Send TREP to  $S$ 

```

When a node n receives the first copy of a TREQ from a neighbor m , it sets a reverse pointer to S via m (lines 2-4 and 13-15) and forwards the message if it is a MPR of m (lines 5-8). When a node forwards a TREQ it records its parent node to reach S in the message (line 7). The flood of the TREQ by MPRs produces a spanning tree rooted at S .

Every MPR that receives a TREQ has to send a response to S . The response will contain its 1-hop neighbor set if it has changed since the last response sent to S . The neighbor set is carried in Topology Reply messages (TREP). To reduce overhead, only the leafs of the spanning tree should generate TREPs. The remaining nodes will aggregate their response in TREPs received from upstream nodes. Initially, every MPR that receives a TREQ sets a timer to generate a TREP message (line 6) after δ_r ms. When a node determines that it is not a leaf, it resets the timer to a higher value ($\Delta_r > \delta_r$) to wait for a TREP from an upstream node (lines 10-11). A node knows that it is not a leaf when it receives a TREQ from a node that has chosen it as parent (line 10). D immediately sends a TREP when it receives a TREQ (line 16).

By only sending the connectivity information of MPRs and D to the source, TREP overhead and bandwidth is reduced. This doesn't prevent SDMR from finding spatially disjoint paths because MPRs cover all geographical regions of the network.

2) *Topology Reply*: When a MPR sends a TREP, it follows the steps of algorithm 2. The response will contain its 1-hop neighbor set only if it has changed since the last response sent to S (lines 2-4). The rule of line 2 enables a node to determine if its neighbor set has changed. This is possible because a node increments $nbset_seqnum_n$ with every change in its neighbor set, and records the sequence number when it last sent a response to S (line 4). Different sequence numbers don't guarantee that the neighbor set differs. For example, a node can temporarily lose some neighbors and recover them later. In this case, the $nbset_seqnum_n$ prior to losing the neighbors and after recovering them is different but refers to the same neighbor set. However, in many cases this rule will avoid sending connectivity information which hasn't changed.

Intermediate nodes can either generate or forward TREPs. When a node receives a TREP, algorithm 3 is executed. If it receives a TREP before $pending_trep_timer_n^S$ expires, it cancels the timer, appends its 1-hop neighbor set to the TREP (if neighbor set has changed since last TREP to S) and forwards the TREP via the parent (lines 1-6). If the node doesn't have a pending TREP to S it forwards the TREP via the parent (lines 7-8). If the pending TREP timer expires it sends a new TREP (Alg. 2).

Algorithm 2 A node n sends a TREP to S .

```

1:  $trep = \text{Create TREP}$ 
2: if  $nbset\_seqnum_n > last\_nbset\_seqnum_n^S$  then
3:   Append 1-hop neighbor set to  $trep$ 
4:    $last\_nbset\_seqnum_n^S := nbset\_seqnum_n$ 
5: Send  $trep$  via  $parent_n^S$ 

```

Algorithm 3 A node n receives a TREP destined to S .

```

1: if  $pending\_trep\_timer_n^S$  then
2:   if  $nbset\_seqnum_n > last\_nbset\_seqnum_n^S$  then
3:     Append 1-hop neighbor set to  $trep$ 
4:      $last\_nbset\_seqnum_n^S := nbset\_seqnum_n$ 
5:     Cancel  $pending\_trep\_timer_n^S$ 
6:     Forward  $trep$  via  $parent_n^S$ 
7: else
8:   Forward  $trep$  via  $parent_n^S$ 

```

A path discovery timeouts TREQ_TIMEOUT seconds after the TREQ is issued, after which, if no paths have been found, S must issue a new TREQ.

When S receives a TREP it adds the contained connectivity information to G_S . If the source of the TREP is D , S sets a timer to calculate paths, called *calculate_paths_timer*. This timer cannot expire after the path discovery timeout. When *calculate_paths_timer* expires, S proceeds to calculate paths (see III-E). If valid paths are found, G_S is set to expire after TOPOLOGY_TIMEOUT seconds. If no valid paths are calculated and the path discovery timeout has not been reached, S waits until the path discovery timeout to attempt to recalculate paths. Note that in this time S might receive

more TREPs. If paths are found, G_S is set to expire after TOPOLOGY_TIMEOUT seconds, otherwise S must issue a new TREQ.

E. Path calculation

We briefly describe the path search method due to space restrictions. The source first searches the topology graph G_S for a set of paths P between S and D . To this end we use a breadth-first search algorithm, applying a number of heuristics to discard paths. This is important to allow a subsequent fast comparison of paths which is costly. From the set of paths P , the source will choose a set of most disjoint paths. For simplicity as of now the source only calculates the pair of most disjoint paths, by comparing the distance of all pairs in P , and choosing the pair with most distance.

F. Path distance metric

We have developed a metric to measure the distance between paths. The main reason to develop a new metric is that existing metrics such as *coupling* [8] and *correlation* [9] present limitations in the degree of separation they can measure. In some respects our metric can be considered similar to correlation, but whereas correlation is based on the number of 1-hop links between two paths, our metric is based on the distance in hops between nodes of both paths, where the distance measured can be potentially unbounded. Note that SDMR can function with other metrics such as correlation.

We denote P a set of different paths connecting two endpoints. Assume a path p is a sequence of unique nodes that connects both endpoints, i.e. $p = (n_1, n_2, \dots, n_i)$. Endpoints are not included in the sequence because they are common to all paths in P .

Distance between nodes is measured as the minimum distance in hops between the nodes, as defined in eq. 1. It ranges in $[0, \alpha]$, where α represents a desired upper bound on the distance. If $\alpha = \infty$ the distance is unbounded. The use of a bound can simplify distance calculation, and can be set to the maximum degree of separation desired between paths.

The distance of a node to a path is the minimum distance in hops from the node to the path, as shown in eq. 2.

The distance from path p_i to p_j is the arithmetic mean of the distance of nodes of p_i to p_j , as per eq. 3. Since paths don't have equal length, eq. 3 is not symmetric. Therefore, in order to compare pairs of paths, the distance for each pair (p_i, p_j) is calculated as the average of $p_distance(p_i, p_j)$ and $p_distance(p_j, p_i)$.

$$hop_distance(n_i, n_j) = \min(\alpha, minhopcount(n_i, n_j)) \quad (1)$$

$$n_distance(n, p) = \min_i \{hop_distance(n, n_i)\} \quad (2)$$

where $n_i \in p$

$$p_distance(p_i, p_j) = \frac{\sum_{n \in p_i} n_distance(n, p_j)}{length(p_i)} \quad (3)$$

This metric can be highly correlated with the euclidean distance between both paths, depending on the network density

and chosen value for the bound α . As we shall see in section IV, it shows high correlation in the tests conducted.

IV. PERFORMANCE EVALUATION

Our main objective is to evaluate the capacity of SDMR of finding spatially disjoint paths. We simulate SDMR with ns-2 [16] and compare it against AODV and AOMDV, under transmission scenarios with one source-destination pair. First we measure the distance of paths found by SDMR (with eq. 3 and euclidean distance), showing the spatial disjointness of the paths found. To demonstrate the effectiveness of spatial separation, we analyze the percentage of session packets every node in the network can intercept when routing with each protocol. Both SDMR and AOMDV distribute data packets along available paths to the destination, and AODV uses a unicast route. We show that in SDMR there are substantially less nodes receiving more than 50% of the session.

A. Simulation environment

We have developed a simulation model of SDMR in ns-2. The AODV model of ns-2 is based on [3]. The AOMDV model is based on [1]. Scenarios simulated consist of static networks of 35 nodes placed randomly in a rectangular 1000 meter \times 1000 meter area and 120 nodes placed randomly in 2000 m \times 2000 m area. All topologies generated are connected and with a minimum distance of 160 m between nodes. The traffic pattern consists of one CBR/UDP connection. Right now we focus on scenarios with no mobility and low overhead, in order to measure distance achievable by SDMR without additional variables that could add noise. Each simulation lasts 500 seconds, of which the first 50 seconds represent a warm-up period. The source commences transmission after the first 50 seconds of simulation and stops transmitting 5 seconds before the simulation ends, to give time for packets on flight to reach their destinations. In order to prevent protocols from routing along the same set of paths found in the first route discovery, a new route discovery is made every 50 seconds. Traffic rate is 4 packets/s and size of data packets is 512 bytes. For SDMR we set $\delta_r = 50$ ms and $\Delta_r = 500$ ms. The upper bound on distance of eq. 1 is set to $\alpha = 3$, because this will result in paths which are more than two hops away. This should translate to paths more than 500 m away. This distance is sufficient to avoid interference between paths and to avoid interception of packets by a node between both paths. Results validate this claim.

Confidence intervals are shown at the 95% level. Each point in graphs shown represents an average of ten runs with different topology.

B. Performance metrics

We use the following metrics to study protocol performance: (i) *Path distance* - distance between paths as defined in eq. 3; (ii) *Euclidean path distance* - euclidean distance between paths obtained from eq. 3 by substituting $hop_distance(n_i, n_j)$ in eq. 2 for euclidean distance; (iii) *Data packet delivery ratio (PDR)* - calculated as the ratio of received data packets to

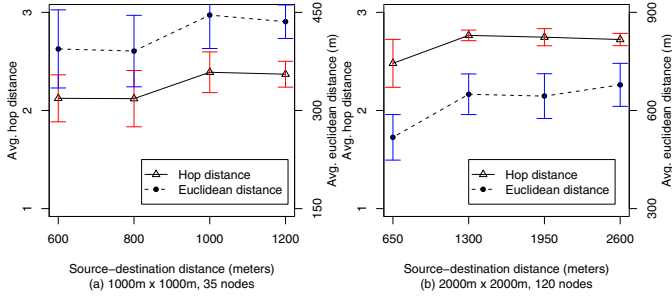


Fig. 1. Distance between paths found by SDMR.

those transmitted by the sources; (iv) *End-to-end delay* - this is the average end-to-end delay of all delivered data packets. Includes all buffering and queuing delays.

C. Distance between paths found by SDMR

Fig. 1 shows the results. We measure with the distance metric defined in eq. 3 ($p_distance$) and with euclidean distance. The left axis shows $p_distance$ and the right axis shows euclidean distance. Note that because euclidean distance represents the average minimum euclidean distance of nodes of one path to another path, the maximum average value it can reach in a square 1000 m \times 1000 m area is roughly 500 m, i.e. although the maximum distance of a node to the nearest node of another path can be 1000 m, on average it is 500 m. And the maximum value in 2000 m \times 2000 m scenarios is roughly 1000 m. As we can see in Fig. 1 (a) the euclidean distance of paths found by SDMR is close to the maximum possible in these scenarios. We can also observe that $p_distance$ and euclidean distance are highly correlated. In the scenarios depicted in Fig. 1 (b), the average euclidean distance is above 600 m, which meets our expectations. Note that euclidean distance doesn't reach its maximum value, because $\alpha = 3$ and $p_distance$ cannot discriminate between paths more than 3 hops away. This is also the cause of the lower correlation between $p_distance$ and euclidean distance in these scenarios.

D. Comparison of protocol performance

We compare the performance between SDMR, AODV and AOMDV. Due to static scenarios with only one data connection, the PDR of all protocols is very high (median value is 1). The average transmission delay is also low due to the existence of only one flow. The median value for AODV and AOMDV is 33.94 and 33.46 ms, respectively. In SDMR it is slightly higher (46.83 ms), because route discoveries take longer. Also, in routing along spatially disjoint paths it is expected that paths will be longer than the unicast shortest path case.

The main difference in performance between SDMR and the other protocols is route discovery overhead. In SDMR it is higher as shown in Fig. 2. The figure shows the total number of RREQs (TREQ in SDMR) and RREPs (TREP) sent. RREQ overhead is less in SDMR because only MPRs forward RREQs, however, RREP overhead is greater. This

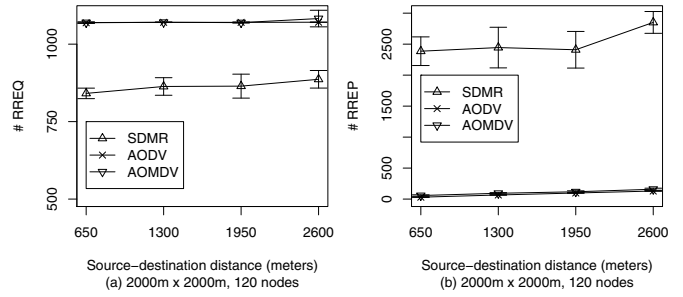


Fig. 2. Performance comparison between SDMR, AODV and AOMDV.

is because a subset of MPRs generate TREPs, which are propagated to the source. Note that, although ideally only the leafs of the spanning tree generate TREPs, in many cases more nodes other than the leafs will generate TREPs, due to not receiving a TREQ from a child node (due to wireless propagation and collisions).

It is important to note that, when a source obtains a new topology graph in SDMR resulting from a topology discovery, this graph can be used to calculate disjoint paths to any destination, avoiding the need for new topology discoveries.

E. Analysis of session interception

We analyze interception of data packets by network nodes with all protocols. Besides studying the effectiveness of spatial disjointness, we also verify if SDMR is suitable for use in scenarios where it is not desired for potentially dangerous nodes in the network to have access to all packets in a session.

Both SDMR and AOMDV route packets along available paths to destination in round robin fashion. AODV routes along one path. Note that paths can change in all protocols throughout the simulation, since each route discovery can result in different routes. We count the number of data packets every node receives, including nodes along routes used.

Fig. 3 shows the fraction of nodes receiving a percentage of session packets. A number of conclusions can be extracted from this figure. In SDMR substantially less nodes receive more than 50% of the session, as compared to AODV and AOMDV. The effect of routing along two disjoint paths in SDMR is that there are much more nodes receiving between 40-50% of the session: the session is evenly distributed along two paths, and the nodes along these paths and their neighbors can receive these packets. In AODV, since packets are routed along one path, less nodes will be able to overhear data packets, but in SDMR less nodes are able to overhear more than 50% of the packets. AODV and AOMDV behave similarly. This is because in practice AOMDV rarely finds more than one path in the scenarios tested, and paths found are not spatially disjoint (AOMDV only aims to find the shortest link-disjoint paths, which in practice are close to each other).

An important conclusion that can be drawn from the figure is that the advantage of using SDMR to prevent nodes from having access to the entire session decreases with the distance between source and destination. This is logical, since the

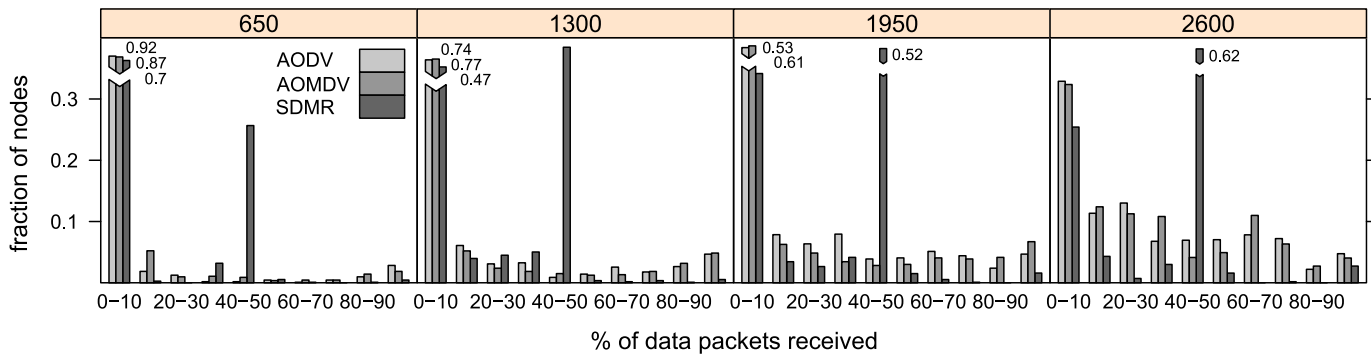


Fig. 3. Histograms of data packet reception with varying source-destination distance. 2000 m \times 2000 m, 120 nodes.

shorter the route, the less the number of network nodes that can overhear packets. If the distance is too short (e.g. 650 m as shown) it might even prove detrimental to attempt to route along spatially disjoint paths, since this will give more nodes access to data as opposed to routing along one path.

V. CONCLUSIONS AND FUTURE WORK

In this paper we have developed a new multipath routing protocol called SDMR. It is an on-demand protocol based on source routing, capable of finding spatially disjoint paths in one route discovery, without the use of location information.

We have studied both the capability of the protocol of finding spatially disjoint paths and its performance. Results show that the protocol can discover paths with considerable separation, and our distance metric can correlate highly with euclidean distance. With the tested parameters, paths found by SDMR are more than two hops and 600 m away in many cases, with solution close to the maximum separation possible with the distance metric. In another set of tests comparing SDMR with AODV and AOMDV, we have shown that distributing traffic with SDMR along separate paths, less network nodes are likely to receive more than 50% of the session. This means that SDMR is better suited for applications where it is not desired that all packets traverse the same region, e.g. secure communications, better reliability avoiding regional failures.

For future work we will study the possibility of further optimizing the protocol in order to reduce route discovery overhead. We will also study the distance of paths found setting a higher bound α , and the correlation with euclidean distance in these cases.

ACKNOWLEDGMENT

This work has been partially funded by excellence research group funds from the Spanish CARM and project TIN2008-06441-C02-02.

REFERENCES

- [1] M. K. Marina and S. R. Das, "Ad hoc On-demand Multipath Distance Vector Routing," *Wireless Communications and Mobile Computing*, vol. 6, no. 7, pp. 969–988, 2006.
- [2] Y. Ganjali and A. Keshavarzian, "Load Balancing in Ad Hoc Networks: Single-path Routing vs. Multi-path Routing," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, 2004, pp. 1120–1125.
- [3] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," Jul. 2003, RFC 3561. [Online]. Available: <http://www.ietf.org/rfc/rfc3561.txt>
- [4] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, "A Framework for Reliable Routing in Mobile Ad Hoc Networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, vol. 1, 2003, pp. 270–280. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1208679
- [5] S. Motegi and H. Horiuchi, "AODV-based Multipath Routing Protocol for Mobile Ad hoc Networks," *IEICE Trans. Commun.*, 2004.
- [6] A. Nasipuri, R. Castañeda, and S. R. Das, "Performance of Multipath Routing for On-demand Protocols in Mobile Ad Hoc Networks," *ACM/Kluwer Mobile Networks and Applications (MONET)*, vol. 6, no. 4, pp. 339–349, 2001.
- [7] S. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks," in *Proceedings of the IEEE ICC*, pp. 3201–3205, 2001.
- [8] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi, "On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks," in *MobiHoc '00: Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing*. Piscataway, NJ, USA: IEEE Press, 2000, pp. 3–10.
- [9] K. Wu and J. Harms, "Performance Study of a Multipath Routing Method for Wireless Mobile Ad Hoc Networks," in *MASCOTS '01: Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. Washington, DC, USA: IEEE Computer Society, 2001, p. 99.
- [10] E. Jones, M. Karsten, and P. Ward, "Multipath Load Balancing in Multi-hop Wireless Networks," in *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005)*, vol. 2, August 2005, pp. 158–166.
- [11] N. T. Nguyen, A.-I. A. Wang, P. Reiher, and G. Kuenning, "Electric-Field-Based Routing: A Reliable Framework for Routing in MANETs," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 8, no. 2, pp. 35–49, 2004.
- [12] V. Loscri and S. Marano, "A new Geographic Multipath Protocol for Ad hoc Networks to reduce the Route Coupling phenomenon," in *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, 2006, pp. 1102–1106.
- [13] T. Voigt, A. Dunkels, and T. Braun, "On-demand construction of non-interfering multiple paths in wireless sensor networks," in *In Proceedings of the 2nd Workshop on Sensor Networks at Informatik 2005*, 2005, pp. 277–285.
- [14] S. Mueller and D. Ghosal, "Analysis of a Distributed Algorithm to Determine Multiple Routes with Path Diversity in Ad Hoc Networks," in *Third International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'05)*, 2005, pp. 277–285.
- [15] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," Oct. 2003, RFC 3626. [Online]. Available: <http://www.ietf.org/rfc/rfc3626.txt>
- [16] K. Fall and K. Varadham, "The ns Manual," <http://www.isi.edu/nsnam/ns/ns-documentation.html>. [Online]. Available: <http://www.isi.edu/nsnam/ns/ns-documentation.html>