Simulating DYMO in OMNeT++

Isabel Dietrich, Christoph Sommer and Falko Dressler

Computer Networks and Communication Systems
University of Erlangen-Nürnberg, Germany
{isabel.dietrich,christoph.sommer,dressler}@informatik.uni-erlangen.de

Technical Report 01/07
Univ. of Erlangen, Dept. of Computer Science 7

Abstract. Mobile Ad Hoc Networks (MANETs) have evolved in the last years into standards in the communication world. By definition, they do not need any network infrastructure to ensure communication between the nodes. Therefore, they are dealing with new challenges in the context of ad hoc routing. This paper deals with our new implementation of the routing protocol Dynamic MANET On Demand (DYMO). DYMO offers adaptation to changing network topology and determines unicast routes between nodes within the network on demand. We developed a simulation model of DYMO for the network simulation environment OMNeT++. Based on the developed model, we performed several simulation experiments to analyze its performance. The implementation of DYMO and the results of the simulation experiments are described in this report.

1 Introduction

The research on Mobile Ad Hoc Networks (MANETs) includes many objectives such as scalability and energy efficiency of ad hoc routing techniques. A number of protocols have been proposed in the last decade [1–3]. These routing protocols build the basis for all communications in MANETs as well as for even more resource restricted networks such as Wireless Sensor Networks (WSNs).

The developments of protocols and solutions are covering multiple problem domains. It turned out that in most MANET scenarios, reactive routing protocols outperform proactive approaches to a certain extent. The probably best known protocol in this context is Ad Hoc on Demand Distance Vector (AODV) [4–6]. This protocol searches routes through the network on demand when data needs to be transmitted. This protocol has been intensively studied in the last years [7]. Based on all these findings, a new protocol has been developed, the Dynamic MANET On Demand (DYMO) routing protocol [8].

In this paper, we describe a simulation model of DYMO, which we developed for the OMNeT++ simulation environment[1]. OMNeT++ already provides the

[1] The DYMO model is available online at `http://www7.informatik.uni-erlangen.de/~sommer/omnet/dymo/`. It reflects the protocol as described in draft-ietf-manet-dymo-06.

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

framework for comprehensive simulation setups. A great number of network protocols used in the context of MANETs are available as simulation models including the complete TCP/IP stack and an AODV implementation. Based on the implemented DYMO model, we performed a comprehensive performance evaluation to study the effects of DYMO. The results of this analysis are also provided in this report.

2 Dynamic MANET On Demand

DYMO is a new reactive (on-demand) routing protocol, which is currently developed in the scope of the Internet Engineering Task Force's MANET working group. DYMO builds upon experience with previous approaches to reactive routing, especially with the routing protocol AODV. It aims at a somewhat simpler design, helping to lower the nodes' system requirements and simplifying the protocol's implementation. DYMO retains proven mechanisms of previously explored routing protocols like the use of sequence numbers to enforce loop freedom. At the same time, DYMO provides enhanced features, such as covering possible MANET-Internet gatewaying scenarios and implementing path accumulation.

Besides route information about a requested target, a node will also receive information about all intermediate nodes of a newly discovered path. Therein lies a major difference between DYMO and AODV, the latter of which only generates route table entries for the destination node and the next hop node, while DYMO stores routes for each intermediate hop. This is illustrated in figure 1. When using AODV, node A knows only the routes to B and D after the route request is satisfied. In DYMO, the node additionally knows a route to node C.

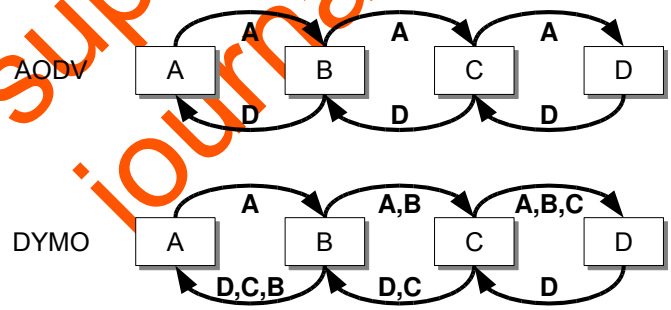

Fig. 1. Routing information dissemination in AODV and DYMO

DYMO is able to set up and maintain unicast routes in IPv4/v6 scenarios by using the following mechanism:

1. In order to discover a new route to a peer, a node transmits a route request message (RREQ) to all nodes in range. This can be achieved by sending

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

the message to a special link local multicast address, which addresses all MANET routers. When an intermediate node receives such an RREQ, it takes note of previously appended information, deducing routes to all nodes the message previously passed through. The node then appends information about itself and passes the message on to all nearby nodes. This way, the RREQ is effectively flooded through the MANET and eventually reaches its destination.

2. The destination responds to the received RREQ by sending a route reply message (RREP) via unicast back to the node it received the RREQ from. As with the passing of an RREQ, this node again appends information about itself and takes note of all routing information contained in the RREP. With the help of the routing information previously obtained while forwarding the corresponding RREQ, the intermediate node is able to send the RREP further back to the start of the chain, until it eventually reaches the originating node. This node will now know a route to the requested destination, as well as routes to all intermediate nodes, and vice versa.

To efficiently deal with highly dynamic scenarios, links on known routes may be actively monitored, e.g. by using the MANET Neighborhood Discovery Protocol [9] or by examining feedback obtained from the data link layer. An implementation may also choose to not actively monitor links, but simply drop inactive routes. Detected link failures are made known to the MANET by sending a route error message (RERR) to all nodes in range, informing them of all routes that now became unavailable. Should this RERR in turn invalidate any routes known to these nodes, they will again inform all their neighbors by multicasting a RERR containing the routes concerned, thus effectively flooding information about a link breakage through the MANET.

DYMO is also designed with future enhancements in mind. It uses a generic MANET packet and message format and offers ways of dealing with unsupported elements in a sensible way.

Regarding related work, several DYMO implementations in different languages and for different systems exist, most notably *DYMOUM*, a GPL implementation for use in both the Linux kernel and ns-2. Also developed for use in the Linux kernel are *NIST-DYMO* (public domain), *EK-DYMO* (closed source) and *DYMO-AU* (GPL, written in LUA and C). Finally, there exist *TYMO* for TinyOS (GPL, written in nesC) and an old implementation of *DYMO for OPNET* (proprietary license).

3 The DYMO simulation model

3.1 Simulation environment – OMNeT++

We modeled DYMO for use as a simulation model in the OMNeT++ 3.2p1 [10] tool, a simulation environment free for academic use, and its *INET Framework* 20060330 extension [11], a set of simulation modules released under the

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

GPL. The OMNeT++ engine runs discrete, event-driven simulations of communicating nodes on a wide variety of platforms and is getting increasingly popular in the field of network simulation. It is also part of the SPEC CPU2006[2] benchmark suite released in August 2006.

Scenarios in OMNeT++ are represented by a hierarchy of reusable modules written in C++. Modules' relationships and communication links are stored as *Network Description* (NED) files and can be modeled graphically. Simulations are either run interactively in a graphical environment or are executed as command-line applications. The *INET Framework* provides a set of OMNeT++ modules that represent various layers of the Internet protocol suite, e.g. the TCP, UDP, IPv4, and ARP protocols. It also provides modules that allow the modeling of spatial relations of mobile nodes and IEEE 802.11 transmissions between them.

3.2 Implementation

For the purpose of evaluating the performance of the routing protocol only, as well as to prevent potential side effects introduced by a transport or network layer, we used our model of DYMO to not only forward RREQs and RREPs, but also take care of delivering our application layer's payload data.

An investigation of effects introduced by the presence of the *INET Framework*'s transport and network layers is being conducted, but out of scope for this document.

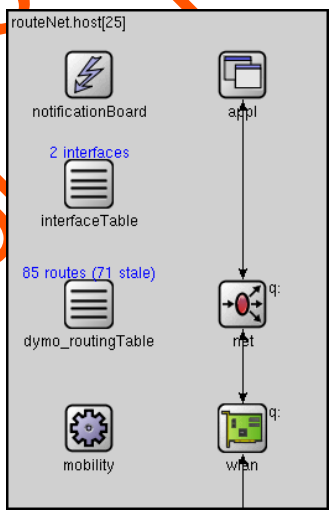

Fig. 2. Screenshot of a simulated node running DYMO

[2] `http://www.spec.org/cpu2006/`

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

As shown in figure 2, the simulated network nodes utilized in this evaluation thus contain only three modules for the handling of messages: Application layer data can be sent and received by an arbitrary module (*appl*), is routed through the DYMO module (*net*) and exchanged with other nodes via an IEEE 802.11 module (*wlan*) provided by the *INET Framework*. The modules on the left hand side of figure 2 do not deal with messages directly, but serve to track the state of the DYMO module's routing table (*dymo_routingTable*) and manage node connectivity (*notificationBoard*, *interfaceTable*) and mobility (*mobility*), as required by the *INET Framework*.

We outfitted our model of DYMO with the capability to queue payload messages received from the application layer, should no usable route be known at the time the data is received. Our model will in this case repeatedly try to establish a route as specified [8, Route Discovery], then dequeue the messages for delivery to the destination or for destruction if no route could be found.

Regarding route maintenance we chose the simplest of the proposed models [8, Active Link Monitoring] for our implementation. Established routes are not actively monitored, but just time out if they not actively used.

All simulation parameters used to parameterize a DYMO module correspond directly to the suggested parameters [8, Configuration Parameters] and are summarized in Table 1, together with the values used in our evaluation. A DYMO messages' TTL is always set to `NET_DIAMETER` and no dedicated mechanism to limit the rate of control messages sent per second was implemented.

Parameter	Value
`ROUTE_TIMEOUT`	3 s
`ROUTE_DELETE_TIMEOUT`	15 s
`NET_DIAMETER`	10 hops
`RREQ_WAIT_TIME`	1 s
`RREQ_TRIES`	3

Table 1. DYMO Module Parameters

4 Performance analysis

We performed a number of simulation experiments in order to check our implementation of DYMO, as well as to evaluate the protocol in different MANET scenarios and under different traffic conditions. In the following subsections, we outline the simulation setup and the selected performance metrics. Finally, we discuss the obtained performance measures.

4.1 Simulation setup

For the performance analysis, we simulated a number of different setups, each consisting of 100 nodes running our implementation of DYMO, 99 of which

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

continuously generated packets addressed to one node acting as a packet sink located in a corner of the playground. We evaluated DYMO's performance for the following combinations of scenarios:

1. Nodes were arranged either to form a *10x10 grid* or in a completely *random* manner.
2. The playground size was adjusted so that the average distance between neighboring nodes corresponded to either *one hop* or *three hops* (according to the grid scenario).
3. Nodes sent a new packet either following an exponential distribution with a mean value of *one second* or a mean value of *ten seconds*, or following a random, *bursty* pattern, which consisted of waiting between zero and five minutes, then sending ten packets spaced 0.49 s apart.

A screenshot for the *10x10 grid* scenario illustrating the *one hop* distance experiment is provided in figure 3. All hosts named `host[xy]` participate in the application by generating data messages and transmitting them to the sink located in the bottom right corner.

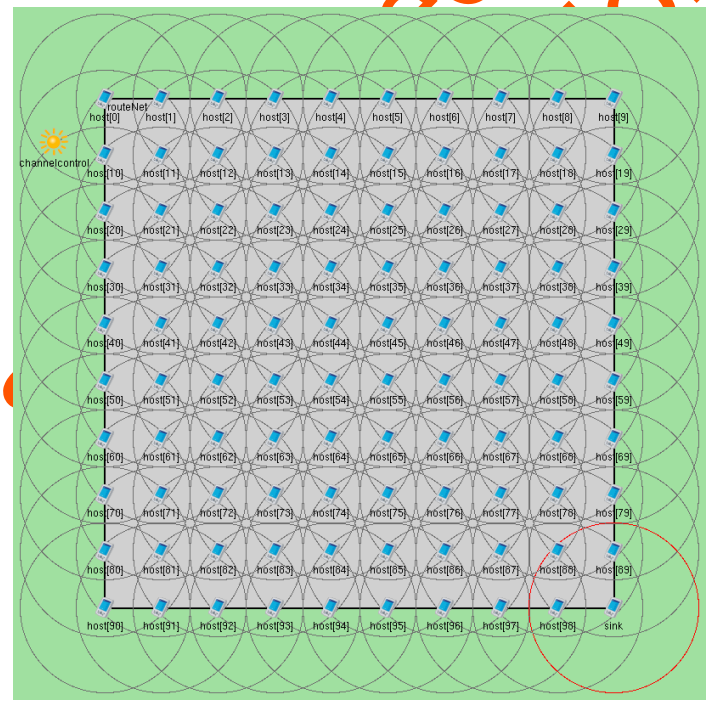

Fig. 3. Screenshot of a *10x10 grid*, *one hop* setup

Additionally, we modeled a network consisting of 11 nodes arranged in a straight *line*, with the distance between neighboring nodes corresponding to

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

one hop. This scenario is shown in figure 4. In this scenario, ten nodes are periodically generating packets and sending this data to a single dedicated sink node located at the end of the line, shown on the left. The time between two packets is exponentially distributed with mean values of *one second* and *ten seconds*, respectively.

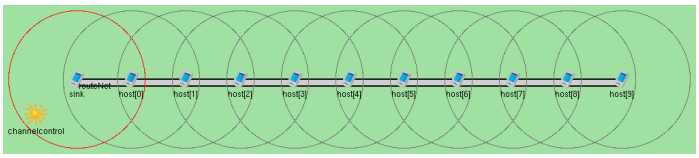

Fig. 4. Screenshot of a *line* setup

4.2 Performance metrics and simulation control

Our model is capable of recording a number of statistics, such as the number of packets sent and received, or message latencies. Overall behavior of the DYMO routing protocol was then examined by recording the following statistical measures:

- Collisions on MAC Layer
- Loss on MAC and Physical Layer
- Frequency of Route Setups
- Data Packets dropped by DYMO
- Route Discovery Delay

For a more detailed analysis of the spatial load distribution, we also examined the following measures in dependence of the recording node's position:

- number of generated RREQs per second, i.e. not including RREQs forwarded on behalf of other nodes.
- number of sent DYMO messages per second, i.e. the rate of RREQs, RREPs and RERRs generated or forwarded on behalf of other nodes.
- number of sent payload messages per second, i.e. including messages forwarded on behalf of other nodes.

All simulations were performed under the control of the Akaroa2 simulation manager [12,13], a tool "aimed at improving the credibility of results from quantitative stochastic simulation using automated sequential analysis". For each simulation setup, multiple simulation runs were conducted in parallel until the measured *data delay per hop* could be determined with Akaroa's confidence and precision exceeding 95 % and 5 %, respectively.

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

4.3 Collisions on MAC Layer

In order to make sure that none of the effects discussed in later sections occurred due to message loss resulting from collisions in the overloaded shared medium, the first measures we evaluated were the number of collisions and the number of messages successfully received as observed by the MAC layer.

Figure 5 displays the ratio of MAC collisions per link-layer packet sent for a scenario of static, randomly deployed nodes. As can be seen, only in the case of high node density and a mean interval between two application-layer messages of 10 s reaches the ratio approx. 50 %. This is significantly above the ratios for the other scenarios ranging below 15 %. Similar results were obtained if some nodes moved according to a random waypoint model and/or if nodes were arranged in a grid.

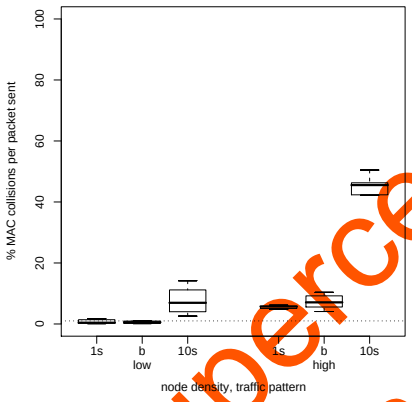

Fig. 5. Collisions on MAC layer. Scenario: random deployment, no mobility

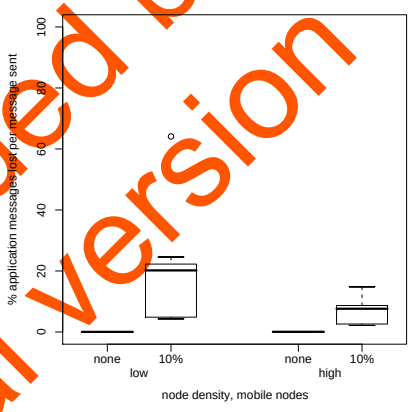

Fig. 6. Loss on MAC and physical layer. Scenario: random deployment

4.4 Loss on MAC and physical layer

To determine the amount of messages lost by either the MAC or the physical layer, e.g. because of node movement, we recorded the total number of application-layer messages passed down by the network layer of all nodes and compared it with the number of messages received by the network layer of the sink.

As shown in figure 6, no such message loss was observed in scenarios where the nodes' positions remained static. Only in dynamic scenarios, as simulated by moving 10 % of the nodes according to a random mobility model, the percentage of packet loss rises noticeably.

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

We also observed that in scenarios with higher node density, node mobility had less impact on packet loss, where packet loss on the MAC and physical layer was approximately cut in half. Similar results were obtained if nodes were arranged in a grid.

4.5 Frequency of route setups

The traffic overhead induced by DYMO was gaged by relating the number of route request messages to the number of application-layer messages sent by DYMO via established routes.

As depicted in figure 7, the number of route request messages exchanged per application-layer message sent decreased significantly if packets were sent at an interval that could keep established routes from expiring. In static scenarios, node density only played a minor role insofar as it improved node connectivity, thus shortening routes, while at the same time it increased the number of collisions thus slightly raising the median number of route request messages that needed to be resent. Simular results were obtained if nodes were arranged in a grid.

However, in the dynamic scenarios shown in figure 8, where 10 % of all nodes moved according to a random waypoint model, the higher node density played a key role in reducing the number of route requests. Here, a larger number of potential routes to the sink meant a higher probability that the chosen route included more static nodes, thus reducing the probability of this route breaking if one of the involved nodes moved out of communication range.

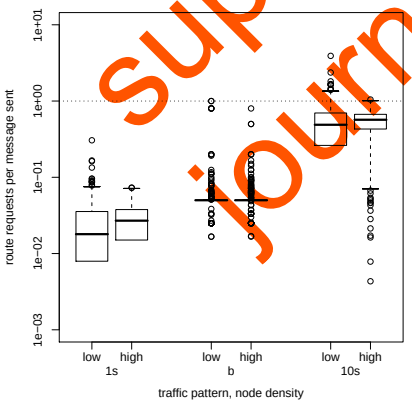

Fig. 7. Frequency of route setups. Scenario: random deployment, static nodes

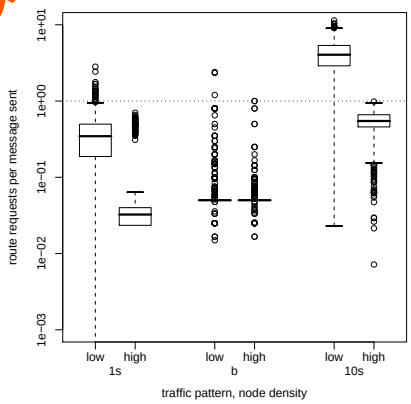

Fig. 8. Frequency of route setups. Scenario: random deployment, 10 % mobile

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

4.6 Data packets dropped by DYMO

As a measure for DYMO's aptitude for finding routes, we compared the amount of data packets dropped by the network layer with the number of packets requested to be sent.

Figure 9 displays the results of this comparison. In the stationary scenario, almost no packets got lost due to problems at the network layer. The few outliers result from particular nodes being in principle unable to establish a path towards the sink in the random deployment. In the mobile scenarios, the probability to successfully set up a path *and* to transmit messages essentially relies on the probability of route failures. Obviously, the *low density* scenario tends to show a higher number of route failures compared to the *high density* example.

Similar results were obtained for other intra-packet spacings and/or if nodes were arranged in a grid.

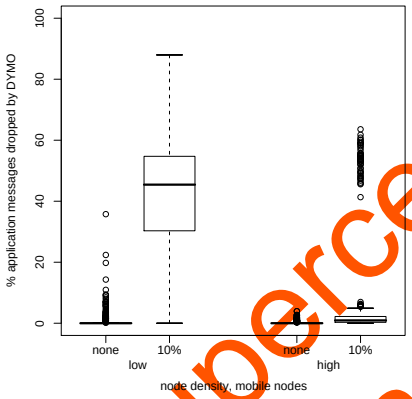

Fig. 9. Data packets dropped by DYMO. Scenario: random deployment, 1 s mean intra-packet spacing

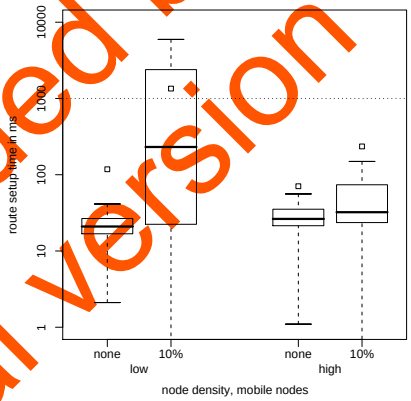

Fig. 10. Route discovery delay. Scenario: random deployment. Mean delays are shown as squares, outliers are not plotted

4.7 Route discovery delay

Another immediate performance measure we evaluated was the delay between a message being queued for delivery by DYMO and its removal from the queue when a route was established. It should be noted that this measure does not reflect cases where a message was not queued because a route was already known, neither does it reflect cases where a message was discarded because no route could be discovered in the time interval set.

As shown in figure 10, for static nodes this delay was well below 100 ms, as opposed to up to 5000 ms if 10 % of the nodes moved in a low-density scenario and up to 200 ms in a high-density scenario.

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

Similar results were obtained if nodes were arranged in a grid.

4.8 End-to-end delay

From a user point of view, one of the most important measures is the delay of messages as observed by the application. Figures 11 and 12 show the results of our simulation experiments. In order to produce comparable results, both figures depict the mean delay per hop, i.e. the end-to-end latency divided by the number of hops for this particular transmission.

Without looking at particular numbers, which mainly depend on the specific network scenario, we need to discuss a number of effects that became visible in these figures. Considering the non-mobile case shown in 11 first and comparing the traffic scenarios with one and ten seconds inter-packet time, we see that the average per-hop delay increases. This effect can be explained by the route timeouts used by DYMO in our experiment. In the *ten seconds* example, DYMO has to set up a route for almost each packet because the available routes have timed out. Thus, each time an additional route setup delay adds to the packet transmission delay.

Comparing the results for the mobile scenario, the effect of node mobility seems to be partly reversed, i.e. the measured delays for the *ten seconds* case are often smaller compared to the *one second* case. Again, this can be explained by the route timeouts. In the *one second* example, DYMO will try to re-use already discovered routes more often, but this time frequently fail because one or more of the involved nodes have moved out of communication range. Analogous to the effect discussed in section 4.5, this reversed effect is more pronounced if node densities are lower, as this means a higher probability of a mobile node participating in a route.

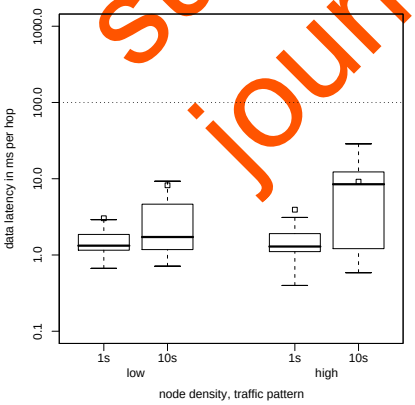

Fig. 11. End-to-end delay. Scenario: random deployment, static nodes

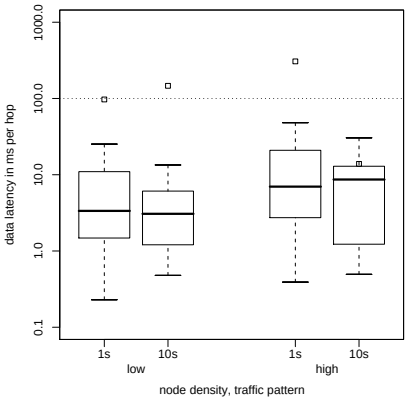

Fig. 12. End-to-end delay. Scenario: random deployment, 10 % mobile nodes

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

4.9 Spatial load distribution (100 nodes in a grid)

Besides the standard performance metrics discussed above, we analyzed the spatial load distribution in the network in order to get more information about the working behavior of DYMO and to identify possibly bottlenecks. In figures 13 and 14, the spatial load distributions for the *low density* and *high density* scenarios are depicted, showing the following three measures: the number of generated RREQs per second, the number of sent DYMO messages per second, i.e. the number of RREQs, RREPs, and RERRs messages, and the number of sent payload messages per second.

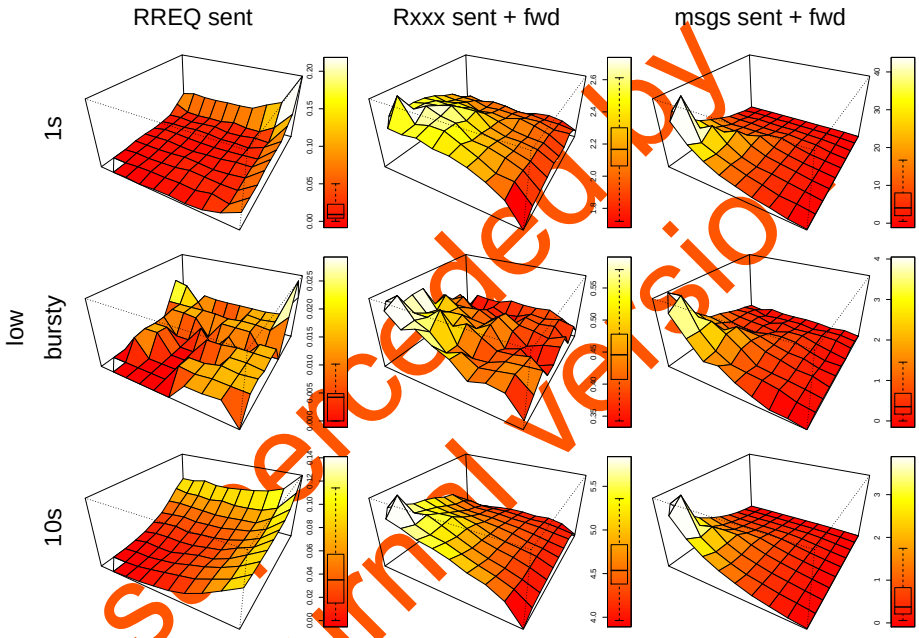

Fig. 13. Spatial load distribution in the 10x10 grid. Depicted is the low density scenario

As can be seen in the first columns of figures 13 and 14, the number of necessary RREQs increases with the distance of the node towards the sink. This behavior results from the key property of DYMO to learn routes from received RREQs and RREPs. Therefore, the probability to know a path prior to initiating a data transfer increases for nodes close to the sink. It can also be seen that the bursty scenarios show less "smooth" results according to the expected less deterministic behavior. Finally, it needs to be mentioned that the steps visible in the *high density* scenarios result from the large broadcast range, i.e. all nodes in this range will learn particular routes simultaneously.

The second column shows the number of all DYMO messages sent, i.e. either generated or forwarded on behalf of other nodes. As in this small network each

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

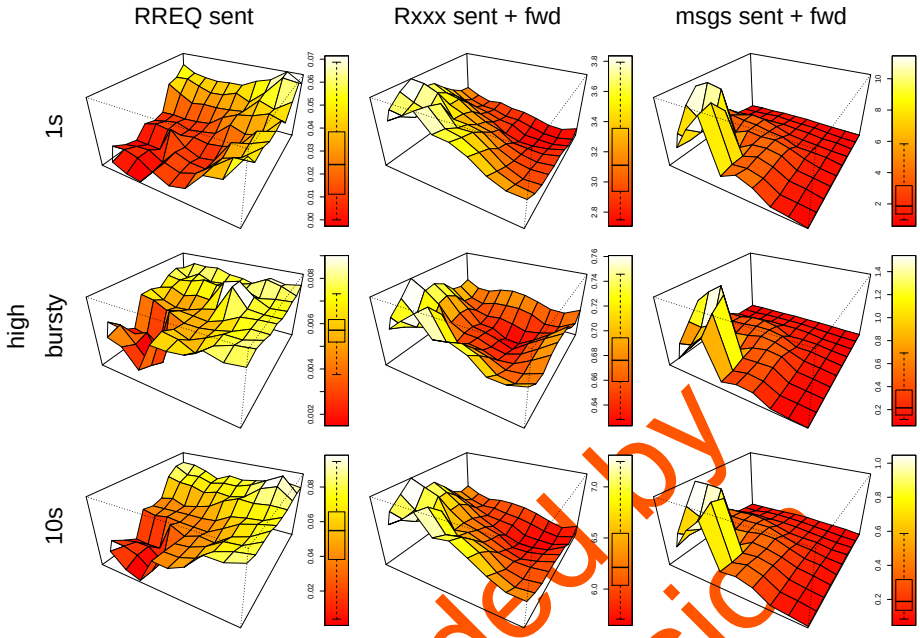

Fig. 14. Spatial load distribution in the 10x10 grid. Depicted is the high density scenario

generated RREQ is forwarded by every node exactly once, the number of RREQs forwarded is almost uniformly distributed across the network. Depending on the node density, only two effects cause a slight variance in this distribution. Because of the TTL being set exactly to the network diameter, in the low node density scenario the number of forwarded RREQs decreases slightly at edges and corners, an effect that can consequently not be observed in the *high density* scenario. In this scenario, however, congestion effects reduce the number of RREQs forwarded along the diagonal that runs through the sink, as well as RREQs forwarded through the center of the playground. In addition to relaying RREQs, each node has to relay RREPs generated by the sink. As RREPs are distributed via unicast along the shortest path found and a particular node in this network will only have to forward RREPs addressed to any node further away from the sink than itself, the number of RREPs relayed increases towards the sink and the diagonal that runs through the sink. In the *high density* scenario, however, the decrease of RREQs forwarded along this diagonal mentioned above is consequently also reflected in a decrease of the amount of RREPs forwarded directly along it. Also visible in the *high density* scenario is a sharp decrease of the number of forwarded RREPs at nodes placed two hops or one hop away from the sink, which stems from the maximum communication distance at this density corresponding to three hops.

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

Lastly, the third column in figure 13 shows the load distribution of the real data transmission. As all nodes uniformly contribute to the generation of messages and forward these messages in a directed way towards the sink node, a permanent increase of the load towards the sink can be seen. Again, steps can be seen in the *high density* scenario as nodes try to forward the message in each hop as far as possible (DYMO uses only the hop count as a metric to determine the shortest path).

4.10 Spatial load distribution (11 nodes in a line)

In order to verify the results, we created a very special scenario, which is commonly used to evaluate the performance of protocols in wireless ad hoc networks. In this scenario, 11 nodes are placed in a straight line. Ten of these nodes generate data using the same traffic characteristics as used in the grid scenario. The 11th node is used as a sink to transmit all data messages to.

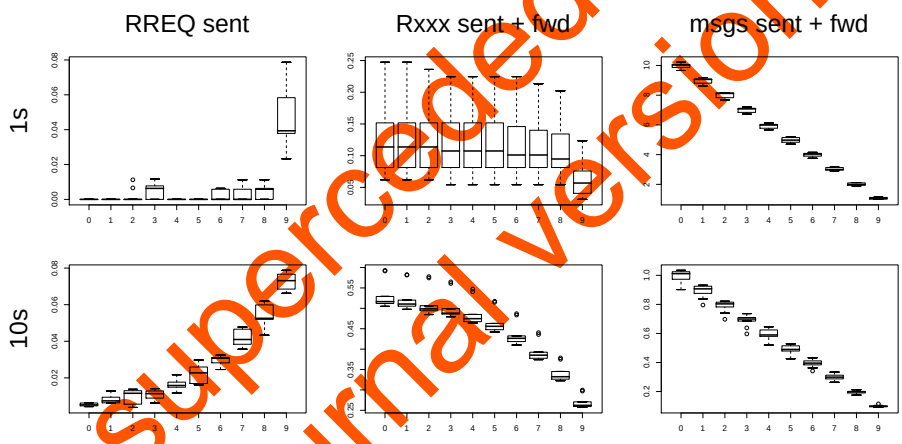

Fig. 15. Spatial load distribution in the line scenario

The measurement results are summarized in figure 15. The left column depicts the number of necessary RREQs to set up routes towards the sink. It can be seen that the number of RREQs increases with the distance to the sink. This validates the results from the grid scenario. Depending on the data rate, routes may time out and additional RREQs are necessary if the inter-packet time is greater than the route timeout used by DYMO.

The second column shows the number of all DYMO messages sent, i.e. either generated or forwarded on behalf of other nodes. As in this small network each generated RREQ is forwarded by every node exactly once, any participating node sends a minimum of the sum of all generated RREQs per second, a measure which is uniformly distributed across the network. Additionally, each node has

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

to relay RREPs generated by the sink. As RREPs are distributed via unicast, a particular node in this network will only have to forward RREPs addressed to any node further away from the sink than itself, which is why this measure decreases with rising distance from the sink. This effect is even more pronounced, because (as explained earlier) nodes further away from the sink generate more RREQs and thus will have more RREPs sent to them by the sink. This can be clearly seen in the top row where the number of DYMO packets sent through the network is almost only due to the RREQs sent by the node farthest from the sink. Each of its RREQs triggers a RREP, both of which need to be relayed by all intermediate nodes, but not by the node itself, which thus ends up having sent the smallest amount of DYMO messages – in spite of being the almost only node generating RREQs.

The third column shows the number of data messages generated and forwarded by all DYMO nodes in the network. As expected, this number linearly increases towards the sink node. Again, the results from the grid scenario are supported by this measure.

5 Conclusion and Future Work

In this work, we motivated, presented, and discussed our implementation of the DYMO routing protocol for OMNeT++ and the *INET Framework*, and we commented on design choices we made in the process. The implementation is available under GPL.

We moved on to demonstrate how our model can be used to evaluate DYMO's performance in a number of different simulation setups and presented the results we obtained in these simulations.

We are currently using this implementation of DYMO, as well as a version which uses the full stack of the Internet protocol family, to evaluate the applicability and performance of DYMO in MANET and especially Vehicular Ad Hoc Network (VANET) scenarios, and are working on comparing the performance of DYMO and AODV.

References

1. Hong, X., Xu, K., Gerla, M.: Scalable Routing Protocols for Mobile Ad Hoc Networks. IEEE Network **16** (2002) 11–21
2. Akkaya, K., Younis, M.: A Survey of Routing Protocols in Wireless Sensor Networks. Elsevier Ad Hoc Network Journal **3**(3) (2005) 325–349
3. Iwata, A., Chiang, C.C., Pei, G., Gerla, M., Chen, T.W.: Scalable Routing Strategies for Ad Hoc Wireless Networks. IEEE Journal on Selected Areas in Communications: Special Issue on Ad-Hoc Networks **17**(8) (1999) 1369–1379
4. Perkins, C., Royer, E.: Ad hoc On-Demand Distance Vector Routing. In: 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA (1999) 90–100
5. Perkins, C., Belding-Royer, E., Das, S.: Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (2003)

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5

6. Chakeres, I., Royer, E.: AODV Routing Protocol Implementation Design. In: International Workshop on Wireless Ad Hoc Networking (WWAN), Tokyo, Japan (2004)
7. Das, S., Perkins, C., Royer, E.: Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. In: IEEE Conference on Computer Communications (INFOCOM), Tel Aviv, Israel (2000) 3–12
8. Chakeres, I., Belding-Royer, E., Perkins, C.: Dynamic MANET On-Demand (DYMO) Routing. Internet-Draft, draft-ietf-manet-dymo-06.txt (2006)
9. Clausen, T., Dearlove, C., Dean, J., Team, T.O.D., the MANET Working Group: MANET Neighborhood Discovery Protocol (NHDP). Internet-Draft, draft-ietf-manet-nhdp-00.txt (2006)
10. Varga, A.: The OMNeT++ discrete event simulation system. In: Proceedings of the European Simulation Multiconference (ESM2001). (2001)
11. Varga, A.: INET Framework. `http://www.omnetpp.org/staticpages/index.php?page=20041019113420757` (2006)
12. Ewing, G., Pawlikowski, K., McNickle, D.: Akaroa2: Exploiting Network Computing by Distributed Stochastic Simulation. In: European Simulation Multiconference (ESM99), Warsaw, Poland (1999) 175–181
13. Sroka, S., Karl, H.: Using Akaroa2 with OMNeT++. In: 2nd International OMNeT++ Workshop, Berlin, Germany (2002)

Christoph Sommer, Isabel Dietrich and Falko Dressler, "Simulation of Ad Hoc Routing Protocols using OMNeT++: A Case Study for the DYMO Protocol," ACM/Springer Mobile Networks and Applications (MONET), Special Issue on Development Tools and Techniques for Mobile Telecommunications, vol. 15 (6), pp. 786-801, December 2010.
doi: 10.1007/s11036-009-0174-5