A Comprehensive Comparison of MANET Routing Protocols in Simulation, Emulation and the Real World

Erik Nordström, Per Gunningberg, Christian Rohner, Oskar Wibling
Uppsala University

Abstract

This paper presents a comprehensive comparison of the ad hoc routing protocols AODV, DSR and OLSR. We compare the protocols in three predictable mobility scenarios using UDP, Ping and TCP traffic and we also study how their relative performance change between simulation, emulation and the real world. Identical protocol implementations are used in all three environments. The real world experiments are done inside an office building with highly variable radio signal strength. Laptops with MANET protocols are carried around in the building according to scenarios in the form of instructions on the screens telling when and where to go. The scenario approach makes the mobility pattern repeatable from one experiment to another and reproducible in the simulation and emulation. The approach allows us to do a relative side-by-side comparison of the routing protocols in all three environments and to compare results across the environments. We present an analysis of altogether 270 real world experiments with different scenarios, traffic characteristics on the three protocols. Our scenarios use four nodes. Already at this scale, we find considerable and unexpected performance discrepancies between protocols in the different evaluation environments.

We emphasize two important conclusions from our results. First, the relative real world performance between the protocols is not consistent across scenarios and traffic types compared to simulation. For example, between simulation and the real world AODV experiences a reduction in UDP delivery ratio of around 10% in one of our scenarios. The same figures for DSR and OLSR are 35% and 20%, respectively. Second, the real world performance variance for some protocols is so large that it makes the comparison less conclusive compared to simulation or emulation. The most important factor to this observed variance is the large variance in radio channel quality.

1 Introduction

The mobile ad hoc network (MANET) routing protocols, AODV [19], DSR [13] and OLSR [8] have been researched for many years and are considered for standardization in the IETF[1]. The research behind these protocols has mainly been performed using simulations and emulations. Most of them have been based on fairly simple radio models (e.g., open space models), which have problems to capture signal strength distributions of complex surroundings such as inside buildings. Furthermore, synthetic mobility models that are normally used, seldom incorporate structural layouts, such as rooms, walls, and corridors. As a consequence there is a high uncertainty regarding how well simulated protocols cope with these more complex environments. Still, existing simulation models are excellent tools to study scalability issues as well as sensitivity to variations in parameter values. They are also attractive since simulation experiments are perfectly repeatable.

Previous studies indicate considerable performance discrepancies between real world and simulations [14], [16]. These studies observe the difference but do not quantify the impact on different routing protocols in comparison. Given this observation we formulated our problem with the following question: If one protocol performs better than another in simulation, is it possible to assume the same for the real world? The big picture result of our study says: ”No - it is not the case. None of the protocols do consistently better than the others when varying mobility and traffic scenarios. Instead, our results indicate that all have severe and protocol specific problems in our indoor surroundings, making their performance inconsistent between the real world and simulation.”

The main contribution is our comprehensive performance comparison of the MANET protocols AODV, DSR, and OLSR, using a set of real world scenarios, recreated in simulation and emulation. In our comparison we:

1. Compare and discuss, side-by-side, the performance of the major MANET routing protocols to understand the trade-offs in the design choices of the different protocols and whether these choices are affected differently by simulation or emulation compared to the real world.

2. Explore the utility of simulation and emulation in relation to the real world. We study discrepancies in the results that could either improve the confidence in previous simulation models or that indicate the need for further exploration or a refinement of models.

The fact that we can study the *relative* performance of routing protocols and at the same time evaluate them in the real world, simulation and emulation we consider as novel.

[1] Internet Engineering Task Force.

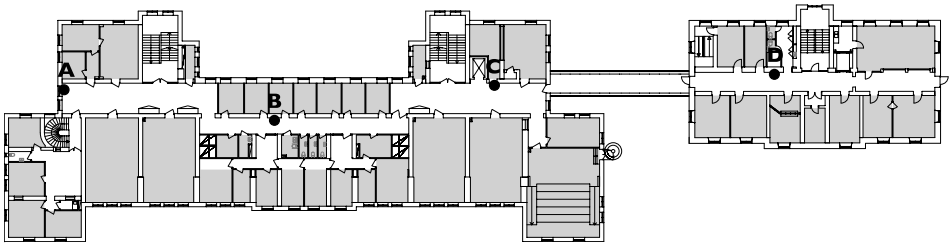

Figure 1: The indoor test environment consisting of offices and corridors. Labels A, B, C and D represent positions, also called *waypoints*, where nodes are either stationary or move between during the course of a scenario.

Previous work that couples real world and simulation, e.g., [12, 14], explore how simulations compare to the real world when feeding (e.g., GPS) traces into a simulator. With our dual approach we can achieve more conclusive results. Furthermore, we are not aware of any comparison of the same extent and with a similar statistical analysis. There are 27 combinations of our scenarios, using UDP, Ping and TCP traffic and AODV, DSR and OLSR routing, resulting in 270 independent experiments. The complete traces comprise around 2 Gbytes of data and will be made available for other researchers along with all software.

The paper is outlined as follows. The next section describes our experimental setup and methodology in detail. Section 3 presents the main results from our comparison through an extensive analysis. Section 4 presents related work and section 5 concludes the paper with a discussion and future work.

2 Experimental Setup and Methodology

The real world experiments feature people that carry laptops and move according to a scenario choreography that is displayed on the screens. All experiments take place in our building, see Figure 1.

The emulations are run on the same platform as the real world experiments but the nodes are stationary in a room and use MAC filters to emulate the mobility and connectivity changes. In our ns-2 simulations we use the same scenarios translated into a schedule and a commonly used radio propagation model.

We use the same protocol implementations that run natively in real world, for all three environments. This is contrast to previous studies which have relied on implementations that use emulation or translation layers to be able to run simulator code in the real world [20, 17, 10]. These approaches may suffer from considerable overhead and sometimes require specific scheduling between real time and simulator time which increases the uncertainty in the results and the conclusions.

We limit the scale of our mobility scenarios to four nodes and three hops, which is the minimum size multi-hop network for which interesting and repeatable real world mobility patterns can be achieved. Already at this scale network, we can observe considerable differences between routing protocols.

2.1 Scenario Descriptions

Our comparisons comprise three mobility scenarios: *End node swap*, *Relay swap* and *Roaming node*. They are choreographed to test different aspects of the routing protocols. Their simplicity makes it feasible to recreate them in the simulation and emulation environments. Figure 2 depicts logical overviews of the scenarios. Positions A, B, C and D correspond to the physical locations in Figure 1. At these positions, nodes only have connectivity to their adjacent neighbors. In all experiments, there is only one traffic stream between nodes 3 and 0, consisting of either UDP packets, Ping messages or TCP segments. The scenarios are constructed so that there are always connectivity between the source node (3) and the destination node (0) over one or more hops. Nodes move at normal walking speed. We measured it to about 1,3 m/s. Each scenario has a warm-up phase and a cool-down phase of at least 10 seconds, during which the routing protocols have time to converge (in the case of OLSR) or, in the case of cool-down, to deliver delayed data packets before the experiment ends. The traffic streams start sometimes after the warm-up phase, depending on scenario.

These scenarios are selected since they stress the ability of the routing protocols to adapt to different situations as discussed below.

The **End node swap** scenario (Figure 2 a) aims to test a routing protocol's ability to adapt when both source and destination move and the shortest path changes from three hops, through two hops, to one hop and back. At time 31s, data transmission from node 3 to 0 starts. At time 51s, end nodes 0 and 3 start to move toward the other end node's position (A and D) where they arrive at time 113s. Nodes 1 and 2 are stationary during the course of the scenario.

The **Relay node swap** scenario (Figure 2 b) instead tests how a routing protocol handles mobility among intermediate nodes while the end nodes are stationary. The traffic is initiated between node 3 to node 0 at time 61s, the

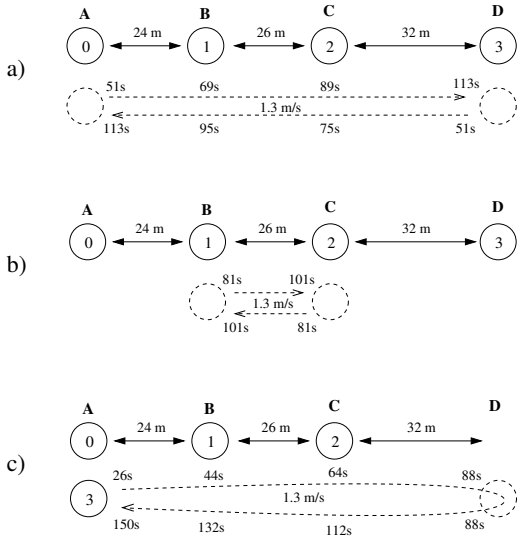

Figure 2: Logical overview of the three scenarios: (a) End node swap, (b) Relay node swap and (c) Roaming node. The dotted lines indicate movement and the times indicate when nodes start to move and when they pass the waypoints A, B, C and D.

relay nodes 1 and 2 start to change positions at time 81s. When they meet in the middle, our node placement allows, depending on the current connectivity, a two hop route between end nodes 0 and 3 using either one of the relay nodes as an intermediary. The relay nodes reach their destinations at time 101s.

The **Roaming node** scenario (Figure 2 c) starts with one hop between node pairs 0 and 3 in contrast to the other scenarios. Also, there is no movement among potential relay nodes. Instead, node 3 "roams" the network, moving from position A to position D and back during the course of the scenario. All other nodes are stationary and only forward traffic. When initiating the traffic at time 26s, node 3 starts its movement from position A toward position D. At time 88s, node 3 has reached position D and heads back toward position A, which it reaches at time 150s. The Roaming node scenario aims to mimic, for example, a mesh network where a user (node 3) is mobile and communicates with a gateway (node 0). Here we study the effect of increasing path length and the route optimization behavior when node 3 moves back.

2.2 Coupling the Real World, Emulation and Simulation

Table 1 lists a number of factors in which the real world, emulation and simulation differ for our experiments. We know from previous work that the radio is an important factor for explaining performance discrepancies between the platforms [14]. In order to study its impact we try to control the other factors. Mobility is handled with choreography and scenarios. Hardware and software are identical while protocol logic is varied with the routing protocols, but otherwise the same between the platforms. Through this harmonization we can use simulation and emulation as a baseline for the protocols and gradually expose the radio factors of the real world.

Environment	Routing Logic	HW	Stack	Mobility	Radio
Real World	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
Emulation	$\surd$	$\surd$	$\surd$	Partial	Partial
Simulation	$\surd$	$\times$	Model	Model	Model

Table 1: Relationship between the real world, emulation and simulation in terms of *Routing logic*, *Hardware*, *Networking stack*, *Mobility* and *Radio environment*. A $\surd$ means that the element is genuine and not a model or, by restrictions, partial. A $\times$ indicates absence of an element.

The real world and the emulated experiments use the Ad hoc Protocol Evaluation (APE) testbed [15], which is publicly available. APE is driven by scenario descriptions to ensure that the mobility pattern is repeatable from one experiment to another, modulo the natural variance caused by people moving around with laptops. We have run complementary experiments to quantify this variance, by varying the mobility artificially and by having people interfere. The results show that this variance is not significant for the experiments in this paper.

When an experiment is conducted, APE reads commands from the scenario file and executes them at the specified time. The scenarios schedule the traffic load and contain instructions to the persons carrying the laptops. As traffic we use either synchronous UDP or Ping traffic (CBR) or a TCP file transfer. APE logs all traffic seen by all nodes during the experiments. By matching the time stamps of all logged packets from all nodes after the experiment, we get a complete global view of the whole experiment. The bulk of our post experiment analysis is based on this information. The logging adds overhead which has some effect on TCP throughput. Our UDP and Ping measurements are not affected by this logging.

All APE computers are identically configured IBM Thinkpad X31 laptops. APE version 0.5, built with Linux kernel 2.6.9 is used for all experiments. The WiFi interfaces are PC-card based Lucent (Orinoco) silver cards supporting the IEEE 802.11b standard. The cards use the Agere Systems Linux driver version 7.18 (March 2004), which we have updated to support Linux kernel 2.6 and an updated wireless extension API. This driver comes with its own firmware that is dynamically loaded on to the card at initialization. All our experiments are run with the driver set to 11Mbps fixed rate with RTS/CTS turned off [21].

2.2.1 Emulation

The emulations use the same HW/SW platform as the real world experiments including the wireless cards. Nodes are stationary and in close proximity, e.g., in the same room and their radios will intentionally interfere with each other. This type of emulation is relative simple, but also quite common and allows comparisons to previous work [10]. The connectivity changes, due to mobility, are emulated using MAC filters by selectively filtering traffic between nodes. The times to enable and disable the filters are extracted from traces generated in the real world experiments. A connectivity change matches the time when the real world signal strengths causes a change of connectivity. We do not introduce any variance in this connectivity time.

This filtering schedule is added to the APE scenario schedule making the connectivity changes completely predictable. The channel quality is high and stable until a change. Besides the predictability the approach eliminates the impact of, for example, gray zones [16] when the signal strength is so weak that the connectivity fluctuate and there are other radio propagation phenomena that degrade the radio channel. The emulation results can therefore, when compared to the real world results, give an indication of the impact of these phenomena on the different routing protocols. Although nodes are stationary in the emulation, it is important to observe that there are still some internal and external radio interferences that may impact the experiments. However, our measurements show that this variance is negligible in our context.

2.2.2 Simulation

For the simulation we use ns-2 version 2.29. We recreate mobility in ns-2 to match the scenarios from APE. Nodes are configured in a chain topology with logical placement and distance matching real world measurements. The node movement speed is programmed to a speed between 1.33 ± 0.0125 m/s. This gives a variance in the times when each waypoint is passed of up to two seconds. In the real world experiments, each waypoint is reached within 1-2 seconds of the scripted time.

Parameter	Value
Pt_	0.031622777
freq_	2.472e9
CSThresh_	5.011872e-12
RXThresh_	1.45647e-09
RTSThreshold_	2000

Table 2: Simulation parameters to mimic an 802.11b WiFi card with a transmit radius of 45 meters (indoors) and RTS/CTS turned off.

The choice of a radio model to match the actual environment is delicate. We settled on using the standard ns-2 TwoRayGround model to be comparable to other simulation studies and to determine whether this commonly used model can be used to predict the real world performance of our routing protocols. However, to make this simple model match our experimental indoor set-up better, we tuned the WiFi transmission range to 45m. It is slightly longer than the measured average value. The real values vary, of course, much more unpredictable with the actual building layout. We believe that the simulations still provide a convincing reference to the emulation and real world experiments. The chosen parameters for the radio model are listed in Table 2.

2.3 Routing Protocols

The MANET working group [1] intends to standardize one reactive and one proactive protocol based on AODV, DSR and OLSR. Current candidates are DYMO [6] and OLSRv2 [4]. There are two main reasons why these two protocols are not in our comparison. First, they are not yet as mature, e.g., in terms of implementations. Second, DYMO and OLSRv2 are evolutionary steps from AODV and OLSR, mainly differing in packet header format. Therefore, we anticipated that by comparing AODV, DSR and OLSR[2], there could be valuable input for the design choices of both DYMO and OLSRv2. In the following sections we give a brief overview of AODV, DSR and OLSR focusing on the differentiating aspects and implementation specific details. For more complete descriptions we refer to the literature or respective RFCs.

2.3.1 Ad hoc On-demand Distance Vector Routing

AODV only disseminates routing updates on-demand, when a route to a new destination is needed. The source node floods the network with a broadcast route request (RREQ). Upon reception of this RREQ, the destination or an intermediate node with a route to the destination replies with a unicast route reply (RREP). Forwarding state is configured on intermediate nodes as these request-reply messages traverse the network. The routing tables are soft state and entries time out when packets are no longer forwarded on a route. Because routing updates are not periodic, AODV must monitor links between neighbors to detect link failures, either with periodic HELLO beacons or using *link layer feedback*. Link layer feedback is often the more efficient, but is only available in the ns-2 simulation. HELLO messages are sent using broadcast and do not guarantee symmetric link connectivity. The AODV implementation used in this evaluation is AODV-UU v0.9.1 [3].

2.3.2 Dynamic Source Routing

DSR is also an on-demand protocol and features similar route discovery as AODV. However, forwarding state is not configured on intermediate nodes in the request-reply

[2]TBRPF [18] is also a MANET protocol, but it is excluded from our comparison because there are no implementations due to intellectual property right (IPR) issues [5].

phase. Instead, routing information is accumulated in control messages as they traverse the network. Each node caches this information and builds its own local view of the network connectivity. End nodes use the information to build complete *source routes*, listing all nodes from source to destination. A source route is appended to all packets and intermediate nodes only use this source route to forward data. DSR's link monitoring is based on network layer acknowledgements (nlACKs), which is a mechanism that periodically exchanges ACK-request – ACK messages over active links to monitor connectivity. The timeout value for a link is dynamically determined from link RTT measurements, similarly to TCP timeouts. In contrast to AODV, DSR can support *automatic route shortening*. Using promiscuous mode a node can overhear packets and, by inspecting the source route, discern whether an optimization can be performed and then notify the sender. We use the DSR-UU implementation v0.2 [3]. DSR-UU implements link monitoring using nlACKs and alternatively link layer feedback in ns-2. DSR ACK requests are piggybacked on data if possible, but ACKs require an extra transmission. DSR-UU reserves 50 bytes of space for the variable length DSR header in each data packet, effectively reducing the optimal amount of data in each packet. This fixed size reduces implementation complexity, but also reduces the capacity.

2.3.3 Optimized Link State Routing

Unlike AODV and DSR, OLSR is a proactive link state protocol similar to OSPF, but with optimizations for ad hoc networks that reduce control traffic overhead and increase reactivity to topological changes. OLSR minimizes control traffic overhead in two ways. First, by using *multi-point relays* (MPRs) to transmit control messages through the network. Second, by only requiring partial link state information to be flooded.

OLSR relies on HELLO messages to maintain a *neighbor set*. In a HELLO message, a node announces its *link set*, *neighbor set* and *MPR set*. These messages only reach direct neighbors. In contrast to AODV, OLSR requires a symmetric link to establish connectivity with a neighbor. Actual link states are only propagated throughout the network by MPRs in Topology Control (TC) messages. TC messages contain sufficient link state to build the topology information base and to perform route calculation. Because of the proactive nature of OLSR, the protocol needs time to converge and reacts more slowly to topological changes. We use the OOLSR implementation v0.99.15 in our experiments [2].

2.4 Traffic Configuration

All experiments have one data flow between a source node to a sink node consisting of either synchronous UDP packets, Ping or a TCP file transfer session. The CBR rate for UDP and Ping is 20 packets per second while TCP transmits with the highest achievable rate. UDP and Ping have no adaptive mechanisms such as congestion control. Therefore, UDP is used to sample the network connectivity and to measure the route latency. Ping requests are sent to the sink node, which then generates a reply packet for each received request. The request-reply mechanism is used to examine bidirectional connectivity and to measure the round trip times (RTT). TCP is used to study the effect of congestion control and reliable delivery.

We wanted to use the same packet size for all three traffic cases to minimize differences in size induced loss. We settled on 1378 bytes to allow for the DSR header in UDP and Ping. Still, DSR pays a performance penalty for TCP due to this header overhead.

3 Evaluation

In this section we show that, using our approach to couple simulation, emulation and the real world, we can single out the radio modelling as the major contributor to performance discrepancies between our experiment platforms. Our results also show that different routing logic will react differently to times of reduced stability in the radio channel. These reactions are not visible in simulations with the models we use and therefore those dissimilarities are evened out between the protocols. We identify the important routing protocol design choices that account for most of the discrepancies and study them in detail.

The rest of the section is organized as follows. First, we describe the measurements and the metrics we use in the analysis. Then we give an overview of the basic performance observations and explain how they support our claims. Finally, we describe in detail the designs of each routing protocol that explain their different their performance in the real world compared to simulation. A discussion about our findings ends the section.

3.1 Measurements

The presented results for each protocol, scenario and traffic is the average of 10 runs. The variance and min/max values are also given. The same type of results are presented for simulation. The emulation results, on the other hand, are from single experiments since the variance between tests using UDP and Ping was negligible under the deterministic connectivity. The emulation with TCP transfers suffers from contention limitations because nodes interfere. We still provide the data here for completeness. The real world experiments suffer from the overhead of logging. Therefore the TCP results should also only be compared over the routing protocols. For AODV-UU and DSR-UU we include link layer feedback in some of our simulation results. The majority of simulations in related work use link layer feedback.

UDP	Protocol	Delivery Ratio			Std. Dev.		Min		Max		Latency σ [ms]		Avg. Hop count	
		Sim	Emu	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW
Relay Node Swap	**AODV-UU**	0.95	0.97	0.86	0.01	0.03	0.94	0.78	0.96	0.90	1.1	85.2	2.8	3.0
	DSR-UU	0.95	0.92	0.63	0.02	0.18	0.93	0.43	0.98	0.88	0.6	231.5	2.9	3.0
	OOLSR	0.83	0.84	0.67	0.03	0.04	0.81	0.63	0.89	0.73	0.7	60.1	2.9	2.9
End Node Swap	**AODV-UU**	0.97	0.94	0.65	0.00	0.13	0.96	0.42	0.98	0.83	1.8	124.2	2.2	2.1
	DSR-UU	0.95	1.00	0.69	0.00	0.15	0.95	0.43	0.96	0.94	36.2	802.0	2.2	2.2
	OOLSR	0.83	0.88	0.63	0.01	0.14	0.80	0.40	0.85	0.84	1.7	69.1	2.1	2.2
Roaming Node	**AODV-UU**	0.97	0.98	0.91	0.00	0.02	0.97	0.88	0.98	0.93	1.7	80.8	1.7	1.9
	DSR-UU	0.98	1.00	0.91	0.00	0.01	0.97	0.90	0.98	0.93	1.4	52.1	1.6	1.8
	OOLSR	0.91	0.91	0.72	0.00	0.05	0.91	0.64	0.92	0.80	1.4	38.9	1.5	1.7

PING	Protocol	Delivery Ratio			Std. Dev.		Min		Max		Latency σ [ms]		Avg. Hop count	
		Sim	Emu	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW
Relay Node Swap	**AODV-UU**	0.95	0.94	0.55	0.01	0.10	0.94	0.45	0.96	0.69	1.3	193.1	5.7	6.0
	DSR-UU	0.92	0.97	0.46	0.00	0.06	0.91	0.39	0.92	0.57	74.3	3514.3	5.6	6.0
	OOLSR	0.83	0.85	0.50	0.03	0.11	0.81	0.37	0.88	0.70	1.3	311.6	5.7	6.0
End Node Swap	**AODV-UU**	0.96	0.95	0.72	0.01	0.06	0.94	0.60	0.98	0.81	3.4	157.0	4.3	4.7
	DSR-UU	0.93	0.99	0.64	0.01	0.19	0.92	0.35	0.94	0.90	52.0	3296.6	3.8	4.5
	OOLSR	0.81	0.83	0.58	0.02	0.10	0.79	0.41	0.86	0.74	3.5	164.6	4.2	4.1
Roaming Node	**AODV-UU**	0.98	0.98	0.71	0.00	0.08	0.97	0.55	0.98	0.80	3.0	182.9	3.3	3.2
	DSR-UU	0.96	0.97	0.80	0.01	0.05	0.95	0.75	0.97	0.89	3.8	4621.2	2.8	3.2
	OOLSR	0.91	0.91	0.57	0.00	0.07	0.90	0.46	0.91	0.66	2.7	1959.0	3.0	3.0

TCP	Protocol	Throughput (Mbps)			Std. Dev.		Min		Max		Latency σ [ms]		Avg. Hop count	
		Sim	Emu	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW	Sim	RW
Relay Node Swap	**AODV-UU**	1.65	0.61	0.14	0.04	0.12	1.59	0.01	1.70	0.38	42.1	356.3	5.7	6.0
	DSR-UU	1.63	0.62	0.18	0.02	0.04	1.59	0.10	1.66	0.23	49.3	249.1	5.5	6.0
	OOLSR	1.08	0.40	0.19	0.14	0.06	1.01	0.09	1.35	0.29	263.0	268.6	5.6	6.0
End Node Swap	**AODV-UU**	2.77	1.33	0.40	0.05	0.23	2.70	0.08	2.82	0.68	51.6	515.1	3.4	2.4
	DSR-UU	2.40	1.07	0.39	0.24	0.09	2.21	0.29	2.68	0.60	53.3	338.9	3.0	2.8
	OOLSR	2.44	1.32	0.43	0.08	0.13	2.37	0.21	2.52	0.62	158.7	230.6	3.1	3.5
Roaming Node	**AODV-UU**	3.58	2.00	0.85	0.04	0.06	3.53	0.75	3.64	0.93	41.4	172.5	2.7	2.5
	DSR-UU	3.45	1.83	0.93	0.38	0.23	2.36	0.48	3.60	1.26	38.7	176.6	2.6	2.4
	OOLSR	3.44	1.78	0.47	0.03	0.24	3.37	0.19	3.46	0.94	68.8	364.8	2.5	2.1

Table 3: UDP, Ping and TCP results showing mean packet delivery ratio and throughput and their standard deviation, minimum and maximum. The performance results are complemented by latency standard deviation and average hop count.

3.2 Metrics

The following metrics are used in our analysis:

Delivery Ratio The number of packets received divided by the number of packets generated during an experiment.

Throughput The number of bytes useful data delivered divided by the time over which data is sent. This is also refered to as *Goodput*.

Latency standard deviation (σ) The variation in time for a packet travelling from the sender to the receiver. For Ping and TCP we calculate the round trip latency standard deviation. We calculate the standard deviation instead of the mean for two reasons: First, the standard deviation can measure the stability of routes. Second, in the case of UDP, accurate calculation of the mean is not possible due to the lack of good time synchronization between the nodes.

Average Hop Count For UDP the average hop count is calculated from the source node to the destination node. For Ping, the hop counts for the Ping request and the Ping reply are added. Similarly, for TCP the sum of the hop counts for data packet and ACK is used.

3.3 Basic Performance Observations

Our results are summarized in Table 3. From the tables we can make basic performance observations in the following terms:

Protocol Logic Correctness. Because our scenarios are constructed with a potential path between source and destination at all times, the protocols should achieve high delivery ratios under ideal circumstances. AODV-UU and DSR-UU consistently achieve over 90% packet delivery ratio for all scenarios in simulation and emulation. OOLSR

achieves slightly lower ratios at 80-90%, which is expected due to its slower convergence. Since the performance is good in both environments we can, with some confidence, exclude any serious routing logic problems as a source of errors.

Mobility and Platform Impact. Despite mobility variations in the simulation, there is no considerable difference in delivery ratios compared to the emulation's clean and deterministic connectivity changes. Therefore, we exclude mobility as a major contributing factor to diverging results between the real world, emulations and simulations for our setup. In the same way we exclude the hardware and the protocol stack as having an impact.

Packet Delivery Ratio. The high delivery ratios of up to 100% in simulation and emulation indicate that the protocols' logic can, with little effort and low loss, handle periods of connectivity changes and multi-hop routes under the idealized circumstances. In the real world we find packet loss concentrated to those periods. Therefore, the radio environment's gradual reduction in predictability passes a threshold at those periods and cause ambivalent feedback to the routing protocols. Protocols may differ in how well they adapt and how much they suffer aftereffects from these periods.

TCP Throughput. For TCP we observe smaller differences between the protocols compared to UDP and Ping. This is because the real world numbers are dominated by the throughput achieved during periods of low variance in link quality and stable routes, while in the rest of the scenario TCP stalls. For UDP and Ping we determined that the discerning periods are concentrated to when links are fluctuating and immediately after. The average hop count supports this claim, as it is much lower for TCP than Ping, i.e., the majority of the TCP packets are sent over shorter routes. Therefore, any advantage of a particular routing strategy never manifests itself. Roaming node is the exception, where the frequent route updates allow AODV-UU and DSR-UU to excerpt their convergence advantage over OOLSR.

Latency Variation. In the real world, many orders of magnitude higher latency variance is observed compared to simulation.

Relative Performance. In simulation and emulation the relative performance between protocols is consistent. However, in the real world this is not the case. Therefore, it is not possible to draw any general conclusions about the routing protocols performance in the real world by using the simple simulation models we use.

However, we want to understand the designs of each protocol that account for the different reactions to the real world environment. In the following sections we look at the design choices of each protocol that, from the above observations, might have a crucial impact on the performance of the protocols.

3.4 Impact of Link Monitoring

The link monitoring strategies used by the different protocols appear to work well in simulation and emulation where some stochastic elements are not present or are modeled in a simplified way. In the real world, however, the link monitoring mechanisms are less accurate, leading to unnecessary link timeouts or increased channel contention.

3.4.1 Network Layer Acknowledgments

One reason for DSR-UU's considerably higher latency standard deviation compared to the other protocols is that nlACKs increase channel contention. Such *self-interference* has been reported by Draves et al. [9], but in that case for static multi-hop ad hoc networks. The interference is higher for Ping and TCP compared to UDP, because nlACK-pairs are sent in both directions on each link. Whilst the ACK request is piggybacked on data, the ACK is not. The interference could be reduced by also piggybacking the ACK, but that is an optimization and only works in case there is traffic in both directions.

The impact of self-interference on latency is evident in both simulation and the real world, but is far more severe in the latter case. Figure 3 illustrates the self-interference in simulation by comparing nlACKs to link layer feedback.

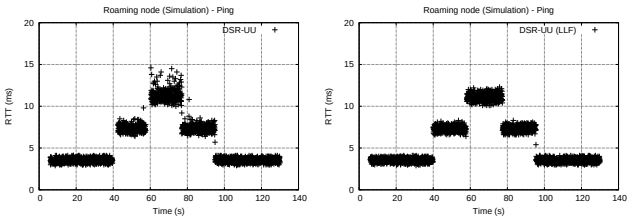

Figure 3: Comparison of the DSR-UU link monitoring from two simulation runs of the Roaming node scenario. Because of *self-interference*, DSR-UU with network layer ACKs has a higher variance in Ping RTT compared to DSR-UU with link layer feedback (LLF). The variance increases with hop count.

The high variance affects DSR's ability to derive a proper retransmission timeout (RTO), leading to premature route timeouts. Simulation and emulation are less affected by this problem due to a partial or modelled radio channel. There are two other factors that explain premature timeouts. First, by borrowing its RTO calculation from TCP, DSR inherits its inability to derive optimal RTOs from RTT measurements in wireless networks. A difference is also that TCP estimates an end-to-end RTT, whilst DSR estimates a per link RTT

that fluctuates more. Second, when a premature link timeout occurs, packets that are either salvaged or buffered during route discovery, will be sent more or less back-to-back when the link is re-established. This causes another type of self-interference, which also increases latency variance. We discuss this problem in detail in Section 3.5

3.4.2 HELLO Messages

AODV and OLSR use broadcast HELLO messages to perform link monitoring. Previous work has reported problems with using HELLO messages to determining link connectivity [7, 16]. The main cause is the difference in transmission range between broadcast and unicast. Another consequence of broadcast is that HELLO messages are sensitive to interference and hidden terminals, likely to be frequently occurring during multi-hop configurations. Delayed or lost HELLO messages cause temporary route breaks, leading to route discovery and increasing latency. Although this causes a slight increase in delay and occasional loss for CBR traffic, TCP is affected more severely because it might go into a timeout. In Figure 4 we see that AODV-UU achieves virtually no TCP progress in the beginning of the End node swap scenario. To explain this we manually inspect our log files and find that in seven out of ten runs there are lost HELLO messages between the node pair 3 and 2, the seconds following the start of data traffic at time 31s. The slow start in TCP builds contention, which is emphasized by the adjacent hops. Hello messages collide with transmissions further down the path due to the hidden terminal effect. When TCP starts at one hop as in the Roaming node scenario there is no such interference and TCP may proceed in slow start without interruptions. We have observed similar problems with HELLOs in OOLSR, but the effect is less prominent and OOLSR seems more resilient, possibly due to its use of link hysteresis. Therefore, OOLSR makes steady TCP progress in the beginning of all scenarios.

3.5 Impact of Buffering

AODV and DSR buffer packets during route discovery. DSR also queues unacknowledged packets in a *maintenance buffer*, so that it salvage those packets in case of a link break. Buffering increases latency and may incur spikes in contention and queue build-up when the buffers are emptied. For some experiments using CBR traffic, we observe latencies of up to 10 seconds for DSR-UU and a very high standard deviation. The default parameters for DSR allow 16 route requests using a back-off algorithm to calculate the timeout between each transmission. The maximum timeout value is 10 seconds. This allows packets to be buffered for very long times during route discovery. AODV-UU employs more modest buffering whilst OOLSR does not buffer at all.

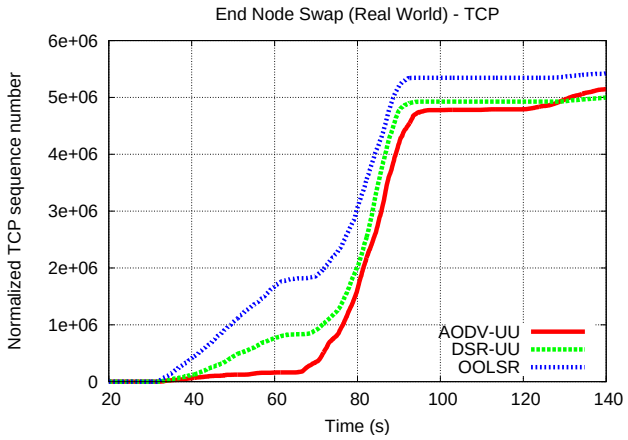

Figure 4: TCP time sequence number trace showing the performance of the routing protocols in the End node swap scenario. AODV-UU struggles in the beginning due to lost HELLO messages. DSR-UU suffers from a too optimistic link RTO. TCP consistently stalls for all protocols when the route switches from one to three hops during a short time period.

In our scenarios, there are only temporary disconnections and packets can therefore be delivered with very long delays. Aggressive buffering, as in the DSR-UU case, can do more harm than good. The mischievous thing here is that the ill effects are not observed until there is already a severe situation in the network. Therefore, emptying buffers quickly might prolong or even deteriorate the situation. This is mainly a problem for constant bit rate UDP and Ping because they continue filling buffers at times of no connectivity. TCP instead adapts its send rate so that buffers do not fill and no floods occur.

In Table 3 we can see the different impact of UDP, Ping and TCP by looking at the real world latency standard deviation. DSR-UU has a considerably higher standard deviation for UDP and Ping traffic, while the TCP latency standard deviation is on par those of the other routing protocols. The effect of buffering over time for Ping and TCP is illustrated in Figure 5. All protocols have a comparable RTT for Ping until the route break. At that point buffers build up for DSR-UU and AODV-UU. DSR-UU's buffering is more aggressive and it never recovers from the flooding that occurs when the new route is established. For TCP, there are temporary fluctuations in RTT from single packets, but since TCP adapts its send rate there will be no floods once the new path is found. There are some indications that the temporary flux in RTT prolongs the timeout for TCP, as no progress is made again until at the end of the scenario.

Aggressive buffering is not observed to the same extent in simulation and emulation because connectivity is either good enough for buffers to never fill the way they do in

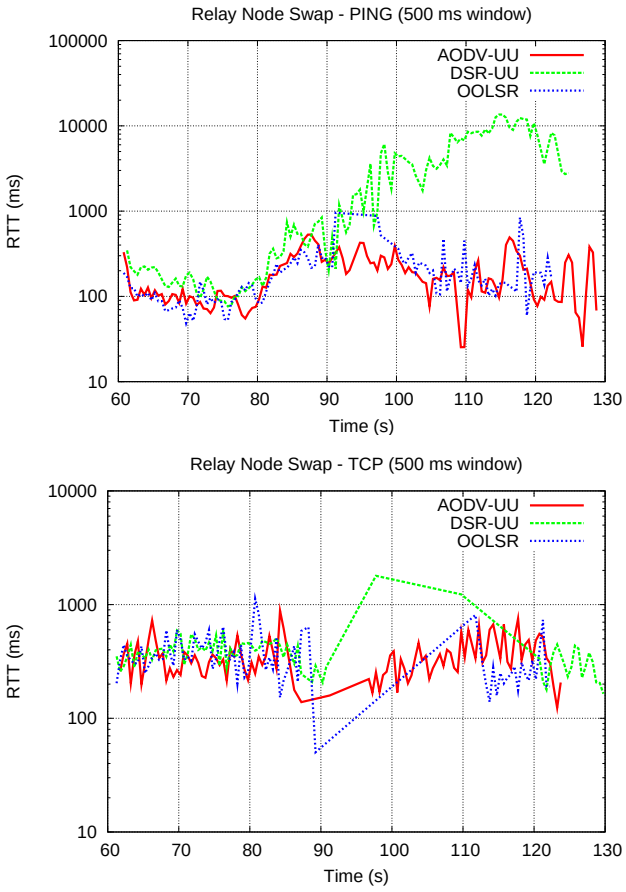

Figure 5: Relay node swap round trip latency for Ping and TCP. The RTT is averaged over 500 ms windows.

the real world, or link breaks are so clean, and bandwidth plentiful, that buffers are easily emptied when a new route is discovered.

3.6 Routing Efficiency

Routing efficiency is determined by two aspects – the time needed to react on a link break and the ability to optimize to a shorter route when one is available. The routing protocols have different mechanisms to react on changes in the network topology. All three protocols optimize their routes by minimizing the number of hops needed from source to destination. A shorter route in general means less interference in the network, higher bandwidth and lower latency. The protocols are, however, more or less sensitive to the radio environment's impact on the stability of links during periods of connectivity changes. Aggressive optimization can, for example, result in packet loss due to premature routes. It is not possible to directly compare routing efficiency in simulation and the real world for only one routing protocol because the timings for the environments are different. However, we can compare simulation and real world by looking at the performance of the protocols relative one another. This gives an understanding of how well they handle the radio environment. In Table 3 the average hop counts in simulation and the real world are quite consistent. The true meaning of the average hop count cannot be understood unless the configuration of the scenario and the delivery ratio are factored in. For example, OOLSR experiences more packet loss during the multi-hop configurations of Roaming node than the other protocols. It has a lower average hop count since the packets successfully sent over one hop are dominant. Average hop count can, despite ambiguity, give an indication of the efficiency of each protocol. In simulation the trend is clear, DSR is the most efficient protocol in terms of shortest path routing. OOLSR appears to be slightly better than AODV-UU, but a comparison is difficult because OOLSR's slow convergence has the effect that few packets are sent on routes that only exist for short time periods ($<$ 10s).

DSR-UU is efficient in terms of hop count because it has *automatic route shortening* and therefore evaluates the route in each packet. OLSR's proactive nature makes it always converge to the shortest routes, but until convergence there is a possibility of non-optimal routing. AODV often uses non-optimal routes because it has no dedicated mechanism for optimization and uses the same route until it breaks. HELLO messages can, however, if enabled act proactively and optimize routes when a node receives a HELLO message from the destination node. HELLO messages sometimes outperform link layer feedback as shown in Figure 6. AODV-UU with link layer feedback never achieves a lower hop count than three as indicated by the constant 3 hop throughput. There is a route break around 110s, but at that time the shorter route from time 60s is no longer available. Therefore, AODV-UU (LLF) always has the same throughput. With HELLO messages, AODV-UU uses the shorter route.

In Figure 7 we plot all the UDP packets sent during the ten runs of the Roaming node scenario. The packets are categorized as successfully received, lost or unoptimally routed over the periods of one hop, two hops and three hops. The figure shows the routing efficiency of each protocol. Packet loss is concentrated to periods of connectivity changes. Note that since packets from all runs are overlayed the loss appears longer and more severe due to time shifts in the loss from one experiment to another. OOLSR suffers more loss than AODV-UU and DSR-UU because of its slower convergence. However, although OOLSR is slow to converge it is quick to optimize routes. DSR-UU is too quick in optimizing the routes. It often chooses premature and unstable routes, causing route flapping between the longer and shorter route. The route flapping occurs because the automatic route shortening tries to optimize the route as soon as a single packet is promiscuously overheard by a node further down the path. If the new link is not stable yet it will

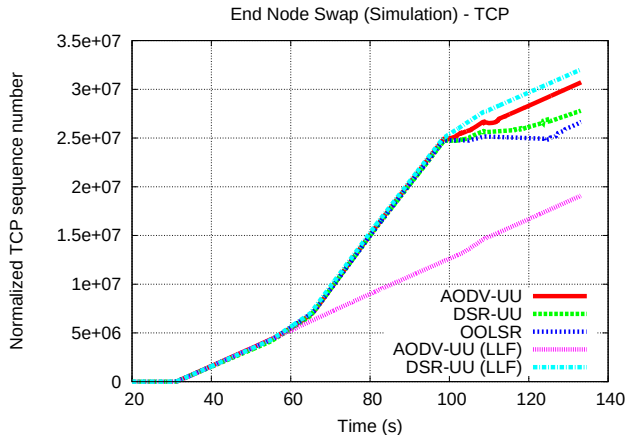

Figure 6: TCP time sequence number trace from simulation showing the route optimization behavior in the End node swap scenario.

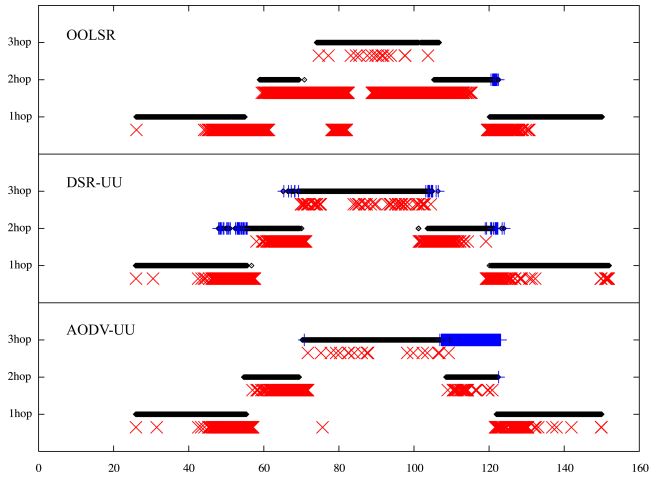

Figure 7: Routing behavior in the Roaming node scenario with UDP traffic. Packets from all ten runs are overlayed and plotted according to their time and route they were sent on (indicated by hop count). A $\diamond$ is a successfully received packet, while a $\times$ is a lost packet. A $+$ over a diamond indicates that connectivity allowed the packet to been sent over a shorter route.

soon time out and the longer route is discovered again. The cycle then starts again. The route request flooding following the route flapping increases overhead and contention in the network. Automatic route shortening sometimes works to DSR-UU's disadvantage at times a longer but more stable route is selected as shown at time 50 s in Figure 7. The optimization will cause route flapping when it switches back to the shorter but lower quality route. In simulation, the radio model works like a binary switch and proves perfect for DSR's route shortening because as soon as an optimization can be made, connectivity is perfect.

AODV-UU's route optimization behavior for Roaming node is visible at time 110 s. The route between node 3 and 0 is not optimized until node 3 receives a HELLO message from node 0, i.e., the destination. The reason some packets take the shorter route is that during some of the ten experiments fluctuations cause a timeout that triggers a route discovery during the period between 110 s and 120 s. The optimal route is than discovered. This is actually a situation when the real world radio environment is an advantage over the stable and predictable model in simulation. At least in a minimal hop count sense.

3.7 Routing Overhead

The overhead pattern for each protocol is very similar in all scenarios. We use the Roaming node scenario with UDP traffic to illustrate the patterns. Figure 8 shows the overhead for all three protocols in bytes. Note that the time axis is different than in the other graphs to show the overhead during the warmup phase.

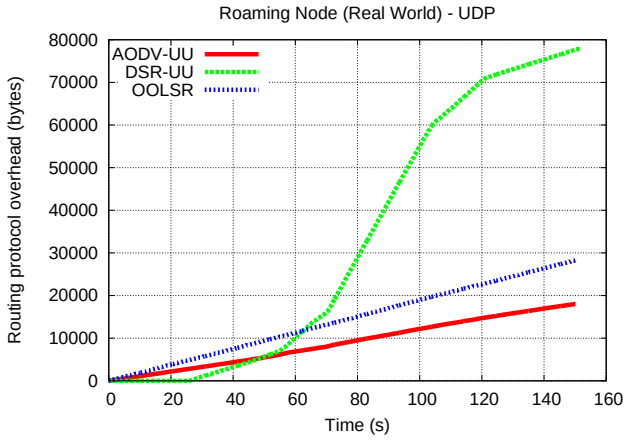

Figure 8: The real world routing protocol overhead for the Roaming node scenario.

OOLSR has, as expected, a constant overhead. AODV-UU also has nearly constant overhead because of the periodic HELLO messages, but the overhead is overall lower than OOLSR. There are periods of increased overhead for AODV-UU caused by broadcast floods during route discovery. DSR-UU has the most deviating overhead pattern. At first glance, the overhead of DSR-UU is considerably larger than the other protocols. However, most of the overhead comes from control information that is piggy-backed on application data frames. Hence, the DSR-UU overhead pattern is not directly comparable to the other protocols' overhead as it is not proportional in the same way to the amount of excess transmissions. Another observation with DSR-UU is that overhead is proportional to hop count and number of data packets, since for each hop, the full source route is (re-)transmitted with each packet. When using network

layer ACKs, this also adds to extra overhead for each hop. However, the overhead of the ACKs is only proportional to the number of active links. The overhead patterns for each protocol occur in all our scenarios. We have found no indications that the overhead is limiting for any of the protocols.

3.8 Discussion

In our comparison we see that AODV and DSR suffer the most when exposed to a real radio environment. Being reactive protocols they are dependent on quickly sensing and responding to events in the surrounding area of a node. Unfortunately, our measurements show that both protocols experience problems related to the sensing efficiency. In particular, DSR sometimes reacts inappropriately; something which could not be anticipated from studying it in simulation. OLSR, on the other hand, is more predictable but being proactive imposes a limit on its performance.

Our results further raise doubts regarding some of the more complex protocol features. Salvaging in DSR and long buffering which is also present in AODV give a better delivery ratio in artificial environments. For real applications, however, the utility of packets degrades over time and it may not be desirable to trade acceptable and predictable latencies for higher total delivery ratios.

Whilst some of the mechanisms discussed can certainly be tweaked to handle the radio impact better, the conclusion is that MANET protocols in general suffer from designs that are optimized for a simulation environment. To alleviate this problem, simulators must improve their radio models or protocol implementations must be systematically evaluated in the real world.

4 Related Work

Grey et al. compare in [10] the routing protocols APRL, AODV, ODMRP, and STARA in a thirty-three node outdoor testbed. They use GPS to collect movement traces from experiments in an open field where 40 people walk around randomly with laptops. The traces are later fed into a "tabletop" emulation and a simulator. The simple radio models yields acceptable results for open fields but this is not true for an indoor corridor environment They use *direct execution* to allow protocols developed in simulation to run in the real world, similarly to the work of Saha et al. [20] as well as the *nsclick* [17] project. In contrast to our code, packets are forwarded in user space and separate event-loops and scheduling increase the overhead. While our approach is scenario based, they instead use a larger scale network with random mobility and random traffic using only UDP. Each routing protocol is run separately and subjected to different mobility and traffic. Therefore, it is not feasible to compare them side-by-side. Their focus is instead on validating different propagation models in simulation, which is the topic of a follow-up paper by Liu et al. [14]. The authors conclude that it is possible to achieve fairly accurate results using simple radio models. However, the open field scenario they use in their validation is not likely to reflect realistic settings in comparison to more complex environments, e.g., indoors.

Haq and Kunz [11] have evaluated OLSR using two different simulators as well as an emulated testbed. They study the total number of successfully transmitted packets using CBR traffic (UDP) at two different rates and two different packet sizes. The authors use a single scenario with five nodes and report that at low traffic rates, testbed results match closely with those from simulation. However, at higher rates they see very significant differences. Apart from providing a much more extensive study in terms of ad hoc routing protocols, scenarios and traffic types we compare simulation, emulation and real world testing. Haq and Kunz have further only studied the total number of packets received whereas we look at protocol behaviors during the whole scenario and also report on latencies and protocol overheads.

Johnson [12] recorded traffic traces from laptops, running DSR, mounted in cars whose positions were constantly logged using GPS. Several different traffic types were used and the collected data drove simulations as well as emulations. The author believes that simply comparing the average number of received packets from simulations and real experiments does not provide enough information to answer the question of how closely emulations come to reproducing simulation results. It can even produce an incorrect conclusion. He therefore suggests studying time-sequence number plots as well as other performance metrics over time. In our work we use different performance metrics over time and compare simulations to emulations but also to the real world.

5 Conclusions

We have examined the relative performance of AODV, DSR and OLSR in simulation, emulation and the real world. From our comprehensive comparison we conclude that one of the deciding factors for a protocol's performance is the ability to sense the surroundings. OLSR is the most stable protocol, but not necessarily the best performer. The behaviors of AODV and DSR are more erratic and unpredictable, but they are also able to act more often during periods of frequent connectivity changes. However, the relative performance of the protocols change with scenario and traffic type in a way that is not consistent with the results from our simulations.

Although we can only speculate over how our results can be generalized to other types of networks and setups, we have shown that it is not feasible to use simulations to predict the performance of ad hoc routing protocols for our

scenarios. It is difficult to draw conclusions from simulations without having validated that the models expose the limits of the protocols as the real world do.

References

[1] The official IETF MANET working group webpage. `http://www.ietf.org/html.charters/manet-charter.html`.

[2] Projet hipercom, INRIA rocquencourt, OOLSR implementation. `http://hipercom.inria.fr/OOLSR/`.

[3] The Uppsala University Ad Hoc Implementation Portal. `http://core.it.uu.se/adhoc`.

[4] IETF draft, August 2005. draft-ietf-manet-olsrv2-00.txt.

[5] F. Baker. An outsider's view of manet. IETF draft, February 2002. draft-baker-manet-review-00.txt.

[6] I. Chakeres, E. Belding-Royer, and C. Perkins. Dynamic MANET On-demand (DYMO) routing. Internet draft, March 2006. draft-ietf-manet-dymo-04.txt.

[7] I. D. Chakeres and E. M. Belding-Royer. The utility of hello messages for determining link connectivity. In *5th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, October 2002.

[8] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, and a. Qayyum et L. Viennot. Optimized link state routing protocol. In *IEEE National Multi-Topic Conference (INMIC 2001), December 2001*.

[9] R. Draves, J. Padhye, and B. Zill. Comparison of routing metrics for static multi-hop wireless networks. In *SIGCOMM*, volume 34, pages 133–144, October 2004.

[10] R. S. Gray, D. Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Masone, S. McGrath, and Y. Yuan. Outdoor experimental comparison of four ad hoc routing algorithms. In *MSWiM'04*, pages 220–229, October 2004.

[11] F. Haq and T. Kunz. Simulation vs. emulation: evaluating mobile ad hoc network routing protocols. In *Proceedings of the International Workshop on Wireless Ad-hoc Networks (IWWAN 2005)*, May 2005.

[12] D. B. Johnson. Validation of wireless and mobile network models and simulation. In *Proceedings of the DARPA/NIST Workshop on Validation of Large-Scale Network Models and Simulation*, May 1999.

[13] D. B. Johnson, D. A. Maltz, and Y. Hu. The dynamic source routing protocol for mobile ad hoc networks (DSR), April 2003. IETF Internet Draft, draft-ietf-manet-dsr-09.txt, (work in progress).

[14] J. Liu, Y. Yuan, R. S. Gray, and L. F. Perrone. Empirical validation of wireless models in simulations of ad hoc routing protocols. *SIMULATION*, 81(4):307–323, April 2005.

[15] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin. A large-scale testbed for reproducible ad hoc protocol evaluations. In *Proceedings of IEEE Wireless Communications and Networking Conference 2002 (WCNC'02), March 2002*.

[16] H. Lundgren, E. Nordström, and C. Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *Proceedings of The Fifth ACM International Workshop On Wireless Mobile Multimedia (WoWMoM)*, September 2002.

[17] M. Neufeld, A. Jain, and D. Grunwald. Nsclick: Bridging network simulation and deployment. In *Proceedings of the the Fifth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2002)*, September 2002.

[18] R. Ogier, M. Lewis, F. Tempelin, and B. Bellur. Topology broadcast based on reverse-path forwarding (TBRPF).

[19] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing, July 2003. IETF Internet RFC 3561.

[20] A. K. Saha, K. A. To, S. PalChaudhuri, S. Du, and D. B. Johnson. Physical implementation and evaluation of ad hoc network routing protocols using unmodified simulation models. In *SIGCOMM Asia Workshop*. ACM, April 2005.

[21] K. Xu, M. Gerla, and S. Bae. How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks? In *Global Telecommunications Conference (Globecom)*, 2002.