# AEROMAN: A Novel Architecture to Evaluate Routing Protocols for Multi-hop Ad-hoc Networks

Lan Tien Nguyen
School of Information Science
Japan Advanced Institute of Science and Technology
Email: lannt@jaist.ac.jp

Razvan Beuran
Hokuriku Research Center
National Institute of Information and Communications Technology
Email: razvan@nict.go.jp

Yoichi Shinoda
School of Information Science
Japan Advanced Institute of Science and Technology
Email: shinoda@jaist.ac.jp

***Abstract*—In this paper, we present AEROMAN (Architecture to Evaluate Routing PrOtocols for Multi-hop Ad-hoc Networks) which is designed and implemented for evaluation of routing protocols for multi-hop wireless networks. AEROMAN uses QOMET, a wireless link emulation tool, to compute parameters of wireless links, such as bandwidth, delay, packet loss rate, in contention-free conditions. In order to take into account the properties of contention-based media access for wireless channel, AEROMAN uses an Adaptive Traffic (AT) model to emulate the sharing feature of CSMA/CA mechanism in IEEE 802.11. The evaluations show that the AEROMAN with AT model effectively captures the characteristics of wireless communications. Several experiments using OLSR as routing protocol with different routing metrics are performed in order to illustrate the main features and usability of AEROMAN.**

***Index Terms*—emulation, modeling, real-time, testbed, routing, wireless networks.**

## I. Introduction

Recently, multi-hop wireless network has been introduced as a promising approach for next generation wireless networks to enable communication, even in the absence of infrastructure. However, before widely deploying multi-hop wireless networks, it is necessary to test the systems to verify their functionalities.

Network emulation is an experimental technique bridging the gap between simulation and real-world experiments, and thus, it has significant impact on the wireless research community. As in real-world testbeds, application testing can be run in real-time in a more realistic environment, for example, adhering to hardware specific limitations, using real protocol implementations, *etc.* Similar to simulation, the wireless medium effects can be controlled to re-enact network constellations. The only thing affecting the accuracy of the model is the fidelity of the emulation layer control. Emulation experiments are more difficult to set up than simulations because of the usage of real devices, but changing conditions is considerably easier than in real-world experiments. The switching process from emulation to real-world deployment is accelerated, since most part of the codes can be reused.

Basically, emulation environments for multi-hop ad-hoc networks can be classified into two types: centralized emulator, and distributed emulator. Further discussions about each type are presented in Section V. Only distributed emulators are able to support real-time evaluation of topology-related protocols.

AEROMAN, which follows the distributed concept, is being developed at Hokuriku Research Center, National Institute of Information and Communications Technology, in Ishikawa, Japan. This architecture allows us to perform various realistic large-scale experiments with different wireless network technologies, such as WLAN and ZigBee, on a wired-network testbed.

The contributions of this paper are as follows:

- We present a novel architecture (Section II) to perform experiments for evaluating wireless multi-hop routing protocols.
- A model, named Adaptive Traffic (AT) model in Section III, is proposed to effectively capture the properties of contention-based media access for wireless channel. This model is used by a module of AEROMAN.
- Using the proposed architecture, we perform several experiments with different routing metrics, such as ETX (Expected Transmission Count) [1] and ETT (Expected Transmission Time) [2], running on OLSR (Optimized Link State Routing Protocol) [3].

## II. AEROMAN

AEROMAN is a highly scalable architecture that can support the design and development of wireless devices and protocols by seamlessly incorporating real applications and protocol implementation with simulation models. Our goal is to allow native wireless routing codes such as OLSR to run on StarBED facility for evaluation purposes.

The routing codes run on StarBED PCs, that are connected by Ethernet switches, and properties of the links (such as bandwidth, delay, packet loss rate, *etc.*) between these PCs are controlled by *dummynet* [10] based on the $\Delta$Q description, which depends on radio parameters and node mobility description. From the view point of the routing agents, it seems that they are working on a real wireless network with typical wireless features, such as low bandwidth, high delay and high packet loss rate

The sections below illustrate overall AEROMAN architecture, which allows us to perform experiments with multi-hop wireless networks as follows:

978-1-4244-7115-7/10/$26.00 ©2010 IEEE

- QOMET wireless link emulator [5]: represents the off-line part of AEROMAN, which is used to compute links parameters in contention-free conditions.
- AEROMAN: We focus on the parts of AEROMAN which support wireless multi-hop routing, as well as properties of contention-based media access for wireless channel. The design in this paper is particularly dedicated to pro-active routing protocols such as OLSR [3] , *etc.* For re-active routing protocols such as AODV [4] , *etc.*, we probably need to design another framework, but all the models in this paper can be re-used.

### A. QOMET: Wireless Link Emulator

AEROMAN uses *deltaQ* library of QOMET to compute the parameters, such as Signal to Noise Ratio, Frame Error Rate, upper bound of links throughput due to physical conditions (distance, transmitting power, reception sensitivity, noise, etc.), of all the links in the target network (Point-to-Point $\Delta$Q). These parameters represent links parameters in the contention-free conditions. The effects which come from contentions for transmitting on the same channel are not considered in this step. This computation is made in advance, and a file which contains the parameters of all links is distributed to all the nodes. This parameters are used by AEROMAN during the experiment as presented on the left side of Figure 1.

### B. AEROMAN Design

In order to make AEROMAN support wireless multi-hop routing, and properties of contention-based media access for wireless channel, we use the approach depicted in Figure 1.

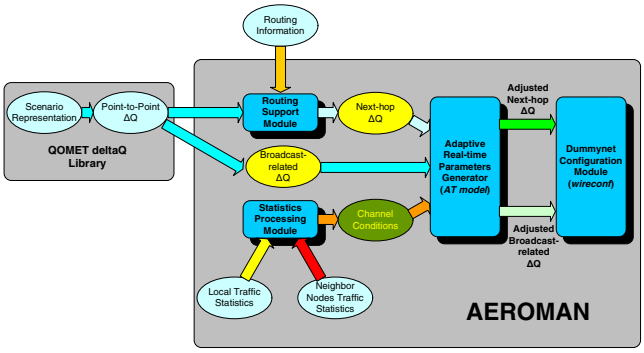


Fig. 1. AEROMAN approach.

For unicast traffic, *dummynet* pipes for this traffic are created at the sender, and the pipes' parameters are applied following the $\Delta$Q parameters. For broadcast traffic, since there is no specified destination in the packets, $\Delta$Q must be applied when packets arrive at the destination through a set of *dummynet* pipes.

The Point-to-Point $\Delta$Q parameters correspond to contention-free conditions. Hence, to take into account the effects when there are several nodes competing for transmission, we need to collect information of existing traffic on the channel. *Statistics Processing* module periodically collects statistical information of traffic at both current node and its neighbor nodes. This information is used to compute channel utilization.

The Channel Utilization is used by the *Adaptive Real-time Parameters Generator* module to adjust the Next-hop $\Delta$Q and Broadcast-related $\Delta$Q parameters following *Adaptive Traffic* model (see Section III). Then, *Dummynet Configuration* module (wireconf) configures the wired-network emulator *dummynet* with the Adjusted Next-hop $\Delta$Q for the Sending pipes, and with Broadcast-related $\Delta$Q for the Receiving pipes.

The advantages of AEROMAN compared with other distributed emulators include:

- AEROMAN allows us to use real-world implementations of routing protocols for multi-hop ad-hoc networks
- Parameters of a wireless link, such as bandwidth, delay and packet loss rate, are computed by considering the properties of contention-based media access for wireless channel.

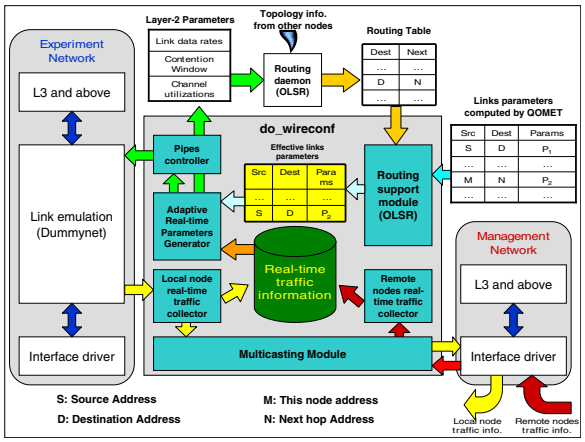


Fig. 2. AEROMAN node internals.

Figure 2 shows AEROMAN node internals, the implementation of the approach in Figure 1, including 6 modules: Pipes controller (*wireconf*), Routing Support Module, Adaptive Real-time Parameters Generator, Local Node Real-time Traffic Collector, Remote Nodes Real-time Traffic Collector, and Multicasting Module. The functionality of each module is presented as follows:

*1) Pipes Controller (wireconf):* The function of this module is to apply the appropriate link parameters which are generated by Adaptive Real-time Parameters Generator module to *dummynet*.

*2) Multicasting Module:* The purpose of this module is to send (multicast manner) the traffic information of this node to other nodes, and receive (multicast manner) the traffic information of its neighbor nodes. The traffic information is achieved by querying statistical information of *dummynet* pipes.

*3) Local Node Real-time Traffic Collector:* This module collects from *dummynet* the statistic information of traffic sent by this node, such as channel utilization, number of packets,

and average packet size, and sends them to other nodes in multicast manner.

*4) Remote Nodes Real-time Traffic Collector:* This module collects the statistical information which is multi-casted by other nodes, and stores it in *Real-time Traffic Information* database.

*5) Routing Support Module:* This module is created to overcome the fact that QOMET cannot take into account the routing path which is created by routing agents at experiment run time. The information which QOMET computes before performing an experiment are the raw parameters of all possible wireless links on the experiment network in contention-free conditions. Based on the destination address of a packet flow, Routing Support Module looks in the routing table to find the outgoing links through which the packets are routed. Then $\Delta$Q parameters of the links are applied to the corresponding *dummynet* pipes.

*6) Adaptive Real-time Parameters Generator:* The purpose of this module is to allow AEROMAN to support properties of contention-based media access for wireless channel. As we mentioned in the Section 2, QOMET, the off-line part of AEROMAN, computes $\Delta$Q parameters in contention-free conditions. In wireless environment, $\Delta$Q description of a link between two nodes does not only depends on the radio parameters and distance between the nodes, but also depends on the amount of traffic that surrounding nodes put on air. It is very critical for AEROMAN to support interference-aware channels in order to improve its fidelity. Using a realistic traffic model allows the *Adaptive Real-time Parameters Generator* module to compute link parameters on the fly from Real-time Traffic Information database and Effective Link Parameters. The traffic model is presented in Section III.

## III. Adaptive Traffic Model

This model is used by *Adaptive Real-time Parameters Generator* module to compute link parameters on the fly from *Real-time Traffic Information* database and *Effective Link Parameters*. In order to capture all characteristics of layer-2, our model follows closely the operation of the CSMA/CA mechanism, which is used as Medium Access Control in the IEEE 802.11. The following sections describe the details of the model.

### A. Approach

In the AT model, we use the two-stage scenario-driven approach. This approach allows us to compute key parameters at several layers, such as physical, data link and network layer, from real-world scenario. The ultimate goal is to find the network parameters which are used to configure a link emulator.

*1) Real-world scenario to physical layer:* In this step, we compute values of frame error rate, denoted as $FER_{media}$, which is induced by physical conditions such as thermal noise, low received signal strength, *etc.* for each node [5]. This computation is done off-line (before the experiment) in order to reduce the computation of nodes during experiment time.

*2) Physical layer to data link layer:* All the link parameters, such as frame error rate, delay, and bandwidth, are recomputed according to the channel utilization. From our point of view, there are two parameters to be changed, namely frame error rate and defer time. Frame error rate is change due to the variation in probability of a frame colliding with the frames which are sent by other nodes at the same time. The reason of changing in defer time is rather straightforward: a node has to defer from decreasing its counter when the channel is busy. The following paragraphs describe how to adjust these parameters.

Given a set of nodes $\{n_1, n_2, \ldots, n_N\}$, all of which are in the same collision domain, let $\{c_1, c_2, \ldots, c_N\}$ be the set of channel utilization values of the corresponding nodes, and $\{s_1, s_2, \ldots, s_N\}$ be the average frame sizes of the frames sent by the those nodes. Let $\{R_1, R_2, \ldots, R_N\}$ be the data rates of node 1 to node $N$. The channel utilization utilized by node $i$ is computed as:

$$c_i = \frac{M_{i,T_s}}{T_s}\frac{1}{R_j} \tag{1}$$

where $M_{i,T_s}$ is the number of bits (at physical layer) sent by node $n_i$ during a certain time interval $T_s$.

Let us consider a transmission from node $s$ to node $r$. In addition to the frame error rate, $FER_{media}$, which is computed in III-A1), frames from node $i$ have probability of colliding with the frames which are sent by other nodes at the same time, called $P_{s,r,coll}$. There are two main reasons for collisions: two nodes choose the same slot for their transmission, or the hidden node problem. Actually, the probability that two node choose the same slot for transmission is small due to the usage of Network Allocation Vector (NAV) [6] in CSMA/CA. The main reason for collisions is the hidden node problem [7]. The hidden node of a transmission from node $s$ to node $r$ is the node which is out of range of $s$ but close enough to $r$ so that the transmission of this node can collide with a reception at $r$. Consider a transmission from $s$ to $r$ which lasts for $N$ time slots, let $H$ be the set of hidden nodes. The collision probability of the transmission from $s$ to $r$ equals the probability to have a transmission from one node in $H$ in that period:

$$P_{s,r,coll} = \sum_{i \in H} c_i \tag{2}$$

Algorithm 1 outlines the algorithm to compute $P_{s,r,coll}$

**Algorithm 1** Computing collision probability

```
1: procedure COMPUTE_COLLISION_PROB(s,r)
                ▷ FER_(s,r,media) is FER_media of link from s to r
2:     P_{s,r,coll} ← 0
3:     for each node i do
4:         if FER_{i,r,media} < 1.0 then          ▷ i is neighbor of r
5:             P_{s,r,coll} ← P_{s,r,coll} + c_i
6:         end if
7:     end for
8:     return P_{s,r,coll}
9: end procedure
```

Let $FER_{s,r,total}$ be the total frame error rate of the link from $s$ to $r$, we have:

$$FER_{s,r,total} = FER_{s,r,media} + P_{s,r,coll} - FER_{s,r,media} P_{s,r,coll} \quad (3)$$

The delay, denoted as $D_{s,r,k}$, is computed as the weighted average of the delay induced to frames undergoing a number of $k$ retransmissions before being received, with $k$ from 0 to $L$, where $L$ is the maximum number of retransmission (in addition to the first transmission of the frame). Value of $L$ can be 3 or 6 depending on whether Request-To-Send/Clear-To-Send (RTS/CTS) mechanism is used or not.

$$D_{s,r} = \frac{\sum_{j=0}^{L} (1 - FER_{s,r,total}) FER_{s,r,total}^{j} D_{s,r,j}}{\sum_{j=0}^{L} (1 - FER_{s,r,total}) FER_{s,r,total}^{j}} \quad (4)$$

$$FER_{s,r,total} \neq 0 \quad \wedge \quad FER_{s,r,total} \neq 1$$
$$D_{s,r} \mid_{FER_{s,r}=0} = D_0 \quad ; \quad D_{s,r} \mid_{FER_{s,r}=1} = \infty$$

The weights included in Equation (4) represent the probabilities for a frame to undergo $j$ retransmissions. The delay $D_{s,r,j}$ consists of three components: back-off time, defer time and transmission time. These components are computed using the equations below, where $T_{SIFS}$, $T_{ACK}$, $T_{DIFS}$, and $T_{Frame}$ represent the time of a SIFS (Short Inter Frame Space) frame, transmitting an Acknowledgment frame, a DIFS (Distributed coordination function Inter Frame Space), and transmitting frame payload, respectively. From the view point of node $i$, let $U_i$ be the total channel utilization used by other nodes, and $I_i$ be the set of nodes which are inside the interference range of node $i$.

$$U_i = \sum_{j \in I_i} (c_j) - c_i \quad (5)$$

The defer time, in which a node has to stop decreasing its counter due to busy state of the channel, is computed by the following equation:

$$T\_def_{s,r,j} = \frac{U_s}{1 - U_s} (T\_back_{s,r,j} + T_{Frame}) \quad (6)$$

where $T\_def_{s,r,j}$ and $T\_back_{s,r,j}$ are the defer and back-off time of a frame at the retransmission number $j^{th}$.

Then the delay including back-off time, defer time, and transmission time is computed as:

$$\begin{aligned} D_{s,r,0} &= T_{SIFS} + T\_back_{s,r,0} + T\_def_{s,r,0} \\ &\quad + T_{Frame} + T_{DIFS} + T_{ACK} \\ &= T_{SIFS} + \frac{T\_back_{s,r,0} + T_{Frame}}{1 - U_s} \\ &\quad + T_{DIFS} + T_{ACK} \quad (7) \\ D_{s,r,j} &= D_{j-1} + T_{SIFS} + T\_back_{s,r,j} + T\_def_{s,r,j} \\ &\quad + T_{Frame} + T_{DIFS} + T_{ACK} \\ &= D_{j-1} + T_{SIFS} + \frac{T\_back_{s,r,j} + T_{Frame}}{1 - U_s} \\ &\quad + T_{DIFS} + T_{ACK} \quad (8) \end{aligned}$$

with $j \in [0, L]$

In case RTS/CTS is disabled, the time to transmit an RTS frame, denoted $T_{RTS}$, and a CTS frame, denoted $T_{CTS}$, can be ignored. The $T\_back_{s,r,j}$ in equation (7) represents the average back-off time induced by $i$ retransmissions. The value of $T\_back_{s,r,j}$ depends on the size of the congestion window after $i$ times retransmission, $CW_i$, and for its computation the interval $[0, CW_i - 1]$ is uniformly sampled. Therefore, the average back-off time is equal to $T_{slot}.CW_j/2$, where $CW_j$ is the value of Contention Window at the $j^{th}$ retransmission, $T_{slot}$ represents the duration of a time slot, $20\mu s$ in the IEEE 802.11.

$$T\_back_{s,r,j} = \frac{CW_j}{2} T_{slot} \quad (9)$$

*3) Data link layer to network layer:* After data link layer parameters are computed, we get to the last step computing network layer parameters. Based on the change in frame delay at previous step, network layer bandwidth can be calculated as follows:

$$B_{s,r} = \frac{T_{Frame}}{D_{s,r}} R_{s,r} \quad (10)$$

where $R_{s,r}$ is the current data rate of the link from node $s$ to node $r$

Packet loss rate of the link from node $s$ to node $r$, $PLR_{s,r}$, is computed from $FER_{s,r,total}$ (Equation 3) by taking into account the 802.11 MAC retransmission mechanism:

$$PLR_{s,r} = FER_{s,r,total}^{L+1} \quad (11)$$

Delay at network layer is given by Equation 4.

### B. Computing channel utilization

The above equations follow closely the nature of Media Access Mechanism CSMA/CA used in the IEEE 802.11, however, they cannot really create an adaptive traffic model for the wireless media, in which every node competes for its transmission. For example, considering a scenario with two nodes $x$ and $y$. Node $x$ would like to start sending its traffic while its neighbor, $y$ has used almost all available time slots. In other words, channel utilization used by nodes $y$ is close to 1. The above model computes a big value of delay and small bandwidth for node $x$. Since node $y$ sees that channel utilization used by node $x$ is small, the model assigns $y$ small delay and big bandwidth. In order to make the model adaptive, we use an algorithm when computing the channel utilization used in Equation 6. The algorithm utilizes the fact that if there is no hidden node, a group of nodes with an infinite amount of data to send, more or less equally shares the channel utilization [8]. A node in this group always can utilizes channel up to this equally shared channel utilization regardless of the channel utilization used by other nodes. Value of channel utilization used in Equation 6 is computed by Algorithm 2. The $ACTIVE\_THRES$ value is set to 0.02 to make sure that a node really sends traffic on the channel.

**Algorithm 2** Computing channel utilization at node $m$

**procedure** COMPUTE_CHANNEL_UTILIZATION($m$)
▷ $m$: ID of the current node
  $ACTIVE\_THRES \leftarrow 0.02$
  ▷ Counting number of active nodes and channel utilization used by other nodes
  $active\_nodes_m \leftarrow 1$ ▷ Always consider myself to be active node
  $u\_others_m \leftarrow 0$
  **for** each node $i$ **do**
    **if** $i \neq m$ **then**

      **if** $FER_{m,i,media} < 1.0$ **then**
        ▷ $i$ is neighbor of $m$
        **if** $c_i \geq ACTIVE\_THRES$ **then**
          $u\_others_m \leftarrow u\_others_m + c_i$
          $active\_nodes_m \leftarrow active\_nodes_m + 1$
        **end if**
      **end if**
    **end if**
  **end for**
  ▷ Compute channel utilization based on fairly sharing channel
  **if** $active\_nodes > 1$ **then**
    **if** $(u_m < \frac{1}{active\_nodes_m})$ *and* $(u\_others_m > 1 - \frac{1}{active\_nodes_m})$ **then**
      $u\_others_m \leftarrow 1 - \frac{1}{active\_nodes_m}$
    **end if**
  **end if**
  return $u\_others_m$
**end procedure**

## IV. Experimental results

In this section, we present several experimental results for evaluating the accuracy of the Adaptive Traffic model, as well as for demonstrating the ability of AEROMAN to perform experiments, using OLSR [3] as routing protocol, with Wireless Mesh Networks.

For emulation experiments, the operating system of all the experimental nodes is FreeBSD v5.4. All of the experiments use QOMET v1.5 with AT model integrated in Adaptive Real-time Parameters Generator of $do\_wireconf$ module. We use olsrd-0.5.5 [9] as the implementation of OLSR routing protocol. For generating traffic, we use *iperf* v2.0.1 for TCP and iperf v2.0.4 for UDP traffic. The reason behind using two different versions of *iperf* is that *iperf* v2.0.1 uses too many CPU resources when generating UDP traffic, which interferes with the operation of the *olsrd* routing agent.

For real-world experiments, we used laptops running Windows XP. Communications between those laptops are made in ad hoc mode. We also use the same version of *iperf* as mentioned above for generating traffic.

### A. Adaptive Traffic Model Verifications

In order to verify the AT model, the second experiment is carried out with a scenario in which there are several nodes competing on one channel. We used 5 nodes using 11Mbps PCMCIA Buffalo WLI-PCM-L11 cards, one running iperf server and the others running iperf clients. The clients start one by one every 10 seconds and try to send UDP traffic with throughput and packet size equal to 6 Mbps and 1024 bytes, respectively, to the iperf server. All clients and server are put in a square 3 m x 3 m and distance between any two random nodes is at least 1 m. Since distances between nodes are small compared to transmission range, we assume there is no hidden node in this scenario. Emulation experiment is done with the same scenario.

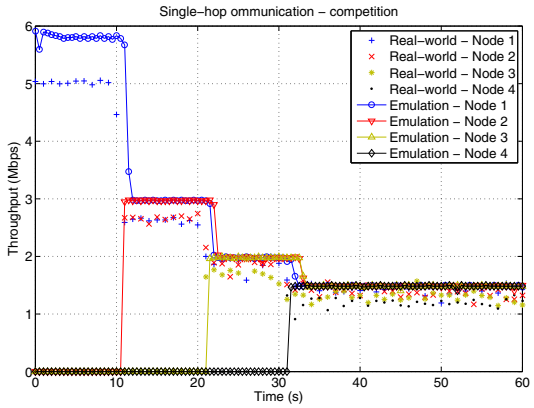


Fig. 3. Experiment of single-hop communication with competition.

The results in Figure 3 show that AT model can capture the behavior of all the nodes in this scenario. When there are several nodes competing to transmit, AT model divides available bandwidth to serve all the nodes. In this scenario, all the nodes try to sent the same amount of traffic, therefore, bandwidth is equally shared between them. We carried out several experiments with different traffic patterns and saw that AT model also works well. Because of the limitations of space, we cannot present all the results in this paper.

### B. Experiments with routing protocol

One of our goals is to create a testbed to evaluate our routing metric for WMNs [11]. The performance values of the routing metric, such as throughput, delay, need to be measured and compared with the performance of other routing metrics, such as ETX, and ETT, in the same scenario. To demonstrate such research, we have created a scenario depicted in Figure 4. In the virtual environment created by using real map data of Kawasaki, Japan, we considered a community wireless network which provides internet connections for residents in this area in emergency conditions. The network consists of 13 nodes including an Internet Gateway, denoted by $GW0$, and 12 WiFi routers, denoted by $N01$ to $N11$, that are located on roofs of resident houses. Transmission power was set to 10 dBm and the environment was considered to have an attenuation of 3.32, typical for outdoor environments [12]. Three random nodes were selected from the 12 WiFi routers to send TCP traffic to $GW0$ for 90 seconds. We generated 20 sets of 3 random nodes, and for each set, we run the experiment

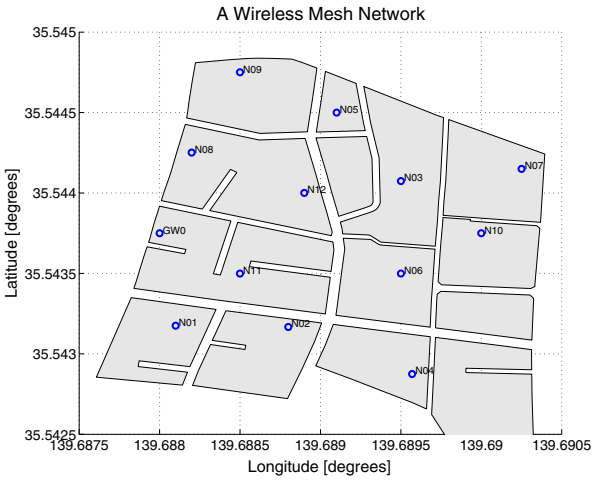


Fig. 4. Experiment on Wireless Mesh Networks.

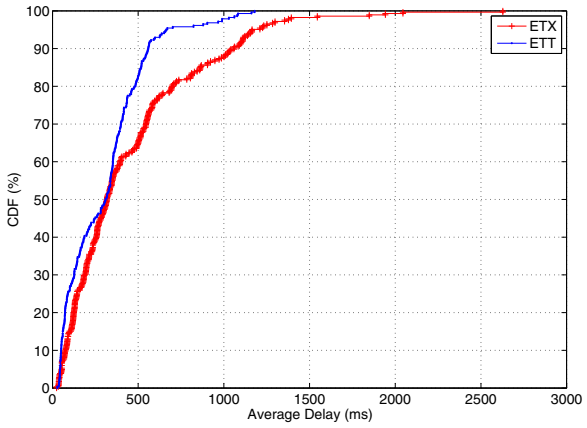


Fig. 6. CDF of the TCP delay from random nodes to gateway.

5 times. Before *iperf* sends traffic, we leave 60 seconds for the *olsrd* routing agent to build routing table based on the emulated environment which is created by AEROMAN.

Figures 5 and 6 plot CDFs (Cumulative Distribution Function) of throughput and delay of TCP flows, respectively. The performance values for both throughput and delay of routing with ETT metric are far better than the ones with ETX metric, since ETT considers data rate when routing traffic. While ETX prefers short paths in number of hops with low data rate of the links, ETT utilizes the paths with high data rate of the links. In this experiment, around 50% of flows have delay higher than 400 ms even though the coverage area is not large. This means that delay is a problem in multi-hop wireless networks, if we want to use the applications that require small packet delay such as VoIP, video conferencing, *etc.*

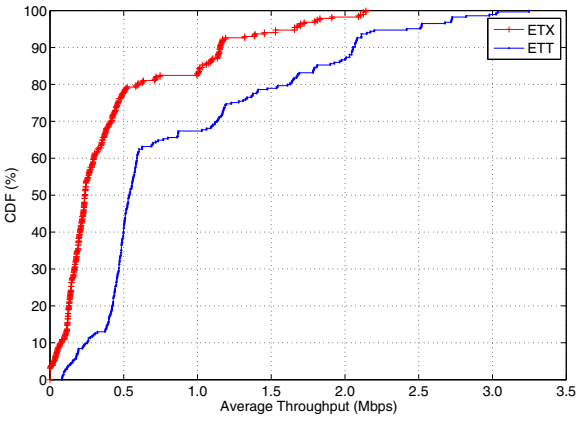


Fig. 5. CDF of the TCP throughput from random nodes to gateway.

## V. Related Work

In this section, we focus on the existing emulators for multi-hop ad-hoc networks. As being mentioned in Section I, emulation environments can be classified using the following categories:

### A. Centralized Emulator

A central server emulates network topology, node movement and actual states of all the wireless links. All the nodes connect to central server and direct their traffic to the server. When the server receives traffic addressed to a destination, the sever forwards the traffic to the destination according to the parameters which characterize the current state of the emulated network, i.e. reach-ability, link quality, collision etc. Each packet must be processed by the server whether to be forwarded or dropped. There are several emulators which fall in this category such as NS Emulator [13], JEmu [14], and MobiNet [15]. However, passing all traffic through a central server makes this server become a bottleneck and limits the scalability of this approach.

### B. Distributed Emulator

Unlike the centralized approach, all nodes are mutually connected via wired or wireless media. The node itself is responsible for directing and forwarding traffic. Since all the node are mutually connected, network topology is created by using logical connectivities which are computed from geographical information, radio parameters, as well as medium information in distributed fashion. In the following paragraph, we only consider the distributed emulators which run on wired networks.

In *MobiEmu* [16] and *NE* [17], logical connectivities are controlled by a central server. This server governs the overall network topology and regulates the configuration of each mobile node. There is no bandwidth adaptation in MobiEmu, while NE offers a limited mechanism for bandwidth limitation in which properties of contention-based media access for wireless channel are not considered. EMWIN/EMPOWER [18], [19] and MASSIVE [20] do not implement any bandwidth limitation mechanism but just determine basic on/off connectivity based on a predefined event list. DINEMO [21]

is a distributed version of NEMAN emulator in which nodes are emulated as virtual nodes inside a single physical machine. However, like NEMAN, DINEMO does not introduce any mechanism to emulate quality of wireless links.

Table I roughly summarizes the properties of related works and AEROMAN architecture. "Virtual node" means the capability to host several virtual nodes on one physical machine. AEROMAN inherits this feature from the design of StarBED. "Contention Aware" means supporting properties of contention-based media access for wireless channel.

TABLE I
PROPERTIES OF NETWORK EMULATORS

| Emulators | Virtual node | Use real routing code | Bandwidth Limitation | Contention Aware |
|---|---|---|---|---|
| MobiEmu | ✓ | | | |
| NE | | ✓ | ✓ | |
| EMWIN | ✓ | | | |
| MASSIVE | | ✓ | | |
| DINEMO | ✓ | | | |
| AEROMAN | ✓ | ✓ | ✓ | ✓ |

## VI. CONCLUSION

We present the design and implementation of AEROMAN as an architecture for evaluating routing protocols for multi-hop wireless networks. Experiments show that AEROMAN can effectively emulate multi-hop wireless networks. In the future, further development is needed to improve the accuracy of AEROMAN, as well as to use it in large-scale experiments on multi-hop wireless networks.

## ACKNOWLEDGMENT

We would like to specially thank Dr. Junya Nakata and Takashi Okada for their valuable comments on the implementations. We also would like to express our gratitude to various people at Hokuriku Research Center, National Institute of Information and Communications Technology, Japan, for their support during the time we did the experiments.

The financial support from Internet Research Center, Japan Advanced Institute of Science and Technology, is gratefully acknowledged.

## REFERENCES

[1] D. Couto, D. S. J., A. Daniel, B. John, and M. Robert, "A High-throughput Path Metric for Multi-hop Wireless Routing," in *Proceedings of the 9th Annual International Conference on Mobile computing and Networking*. New York, NY, USA: ACM, 2003, pp. 134–146.

[2] D. Richard, P. Jitendra, and Z. Brian, "Routing in Multi-radio, Multi-hop Wireless Mesh Networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. Philadelphia, PA, USA: ACM, 2004, pp. 114–128.

[3] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol for Ad hoc Networks," in *Proceedings of IEEE International Multi Topic Conference*, Lahore, Pakistan, 2001, pp. 62–68.

[4] C. E. Perkins and E. M. Royer, "Ad-hoc On-demand Distance Vector Routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, Louisiana, USA, Feb 1999, pp. 90–100.

[5] R. Beuran, L. T. Nguyen, K. T. Latt, J. Nakata, and Y. Shinoda, "QOMET: A Versatile WLAN Emulator," in *Proceedings of the 21st International Conference on Advanced Information Networking and Applications*, Niagara Falls, Ontario, Canada, 2007, pp. 348–353.

[6] M. Gast, *802.11 Wireless Networks: The Definitive Guide*. O'Reilly, 2002.

[7] P. Karn, "MACA a New Channel Access Method for Packet Radio," in *Proceedings of the 9th Computer Networking Conference*, vol. 9th, London, Ontario, Canada, 1990, pp. 134–140.

[8] J.-S. Liu, "Achieving Weighted Fairness in IEEE 802.11-based WLANs: Models and Analysis," *WSEAS Transactions on Communications*, vol. 7, no. 6, pp. 605–615, 2008.

[9] An Ad hoc Wireless Mesh Routing Deamon, *http://www.olsr.org/*.

[10] L. Rizzo, "Dummynet: A Simple Approach to the Evaluation of Network Protocols," *ACM Computer Communication Review*, vol. 27, pp. 31–41, 1997.

[11] L. T. Nguyen, R. Beuran, and Y. Shinoda, "A Load-aware Routing Metric for Wireless Mesh Networks," in *Proceedings of the 13rd IEEE Symposium on Computers and Communications*, Marrakech, Morocco, July 2008, pp. 429–435.

[12] D. B. Faria, "Modelling Signal Attenuation in IEEE 802.11 Wireless LANs–Vol. 1," Kiwi Project, Stanford University, Tech. Rep., July 2005.

[13] "Network Emulation in the Vint/NS Simulator," in *Proceedings of the The 4th IEEE Symposium on Computers and Communications*. Sharm El Sheik, Red Sea, Egypt: IEEE Computer Society, 1999, p. 244.

[14] J. Flynn, H. Tewari, and D. O'Mahony, "JEmu: A Real Time Emulation System for Mobile Ad Hoc Networks," in *Proceedings of the SCS Conference on Communication Networks and Distributed Systems Modeling and Simulation*, San Antonio, Texas, USA, 2002.

[15] M. Priya, R. Adolfo, B. David, and V. Amin, "MobiNet: A Scalable Emulation Infrastructure for Ad hoc and Wireless Networks," in *WiTMeMo '05: Papers presented at the 2005 workshop on Wireless traffic measurements and modeling*. Berkeley, CA, USA: USENIX Association, 2005, pp. 7–12.

[16] Y. Zhang and W. Li, "An Integrated Environment for Testing Mobile Ad-hoc Networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad hoc Networking & Computing*. Lausanne, Switzerland: ACM, 2002, pp. 104–111.

[17] W. Liu and H. Song, "Research and implementation of mobile ad hoc network emulation system," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*. Vienna, Austria: IEEE Computer Society, 2002, pp. 749–756.

[18] P. Zheng and L. M. Ni, "EMWIN: Emulating a Mobile Wireless Network using a Wired Network," in *Proceedings of the 5th ACM International Workshop on Wireless Mobile Multimedia*. Atlanta, Georgia, USA: ACM, 2002, pp. 64–71.

[19] Zheng, Pei and Ni, Lionel M., "EMPOWER: A Network Emulator for Wireline and Wireless Networks," in *Proceedings of the 22nd Conference on Computer Communications*, San Francisco, California, USA, 2003, pp. 1933 – 1942.

[20] M. Matthes, H. Biehl, M. Lauer, and O. Drobnik, "MASSIVE: An Emulation Environment for Mobile Ad-Hoc Networks," in *Proceedings of the International Conference on Wireless on Demand Network Systems and Service*, St. Moritz, Switzerland, 2005, pp. 54–59.

[21] Göktürk, Erek and Puvzar, Matija and Akkøk, M. Naci, "Distributing NEMAN Network Emulator Using MICA Component Architecture," in *Proceedings of the AI, Simulation and Planning in High Autonomy Systems and Conceptual Modeling and Simulation Conference*, Buenos Aires, Argentina, February 2007, pp. 199–205.