ENSC 835: HIGH-PERFORMANCE NETWORKS
CMPT 885: SPECIAL TOPICS: HIGH-PERFORMANCE NETWORKS

Simulation and Performance Study of Ad Hoc Routing with Dynamic Connectivity Management Based on Network Saturation

Fall 2003

FINAL REPORT

Mohamed Soliman
School of Engineering Science
Simon Fraser University
http://www.sfu.ca/~mohameds/ENSC835/adhoc.html

msoliman@cs.sfu.ca
Mohamed.Soliman@nrc.ca

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. ABSTRACT

The purpose of this project is to investigate and undertake simulation-based performance evaluation of ad hoc networks. The project has three folds: Firstly, to investigate ad hoc networks revealing what constitutes its performance criteria for evaluation. The second fold is to utilize and study the simulation model. Thirdly, to propose and suggest enhancements to overcome possible discovered shortcomings observed during the simulation stage. The Simulation environment is based on the AODV model (Ad hoc On Demand Distance Vector Routing Protocol) implemented in OPNET (from the contributed model directory).

Simulation criteria of the ad hoc network include end-to-end-delay and route acquisition latency. Results of the simulation show that the average end-to-end delay increase while the number of nodes in the network increases. While conducting the simulation, observations have shown that control traffic, such as hello messages, consumes network resources. This is sometimes to the benefit of network connectivity and other times in situations that don't necessarily need such amount of control overhead. The proposal of this project is to dynamically adjust the use of hello messages based on the ad hoc network state of mobility and density of nodes. This leads to the importance of studying the degree-of-saturation of the network. Simulations resulted from this modification have shown that this approach is promising to enhance the performance of the ad hoc network. Such enhancement contributes to achieving a better scalability and saves energy power, which is important to ad hoc networks.

2. ACRONYMS

AODV: Ad Hoc On Demand Distance Vector Routing Protocol

IETF: Internet Engineering Task Force

MAC: Medium Access Control

MANET: Mobile Ad Hoc Network

NIST: National Institute for Standards and Technology

PDU: Protocol Data Unit

RFC: Request For Comments

RREQ: Route REQuest message

RREP: Route REPly message

RRER: Route Error/Repair message

TTL: Time To Live

3. INTRODUCTION

An ad hoc network is an interesting special case of networks. It is an infrastructure-less environment that operates on wireless nodes that get connected only when the nodes are into proximity with each other. This paradigm has gained much interest in recent years due to several inherent applications and benefits. Therefore, The MANET (Mobile ad hoc networks) group within the Internet Engineering Task force (IETF) has been formed to work on the aspects of standardizing the protocols and its Internet connectivity [7] .

Benefits of MANETS are in environments where it is needed to build networks on the fly. For example, conference participants can communicate with each other without prior settings. In a battlefield environment, there is no chance to build a network infrastructure in advance. Another interesting application is having a large number of small sensor elements to sense the environment and communicate with each other to build a model of that environment. The later application has gained more interest recently as sensor networks' applications. In this model, there is a need to allow thousands of nodes to communicate within the same MANET. Therefore, supporting such big number of nodes imposes more requirements on the network such as scalability requirements.

In a MANET, nodes are free to join or leave the network. When a node comes into proximity of the network, other nodes will discover the new node and it will eventually become a member of the network. In an opposite scenario, when a node travels away from this proximity, it becomes out of range of the network elements, i.e. leaves the MANET. Obviously, this paradigm imposes extra challenges in network research. Since this environment is infrastructure-less with no previously configured routers, nodes should act as a sender, receiver, and most importantly as a router. In Figure 1, node A is the sender, node D is the receiver while nodes B and D act as routers to construct a route from A to D. In order for a packet sent from a mobile node to arrive to its destination, it may traverse other intermediate mobile nodes that acted as routers. Therefore routing is of much importance to MANETS.

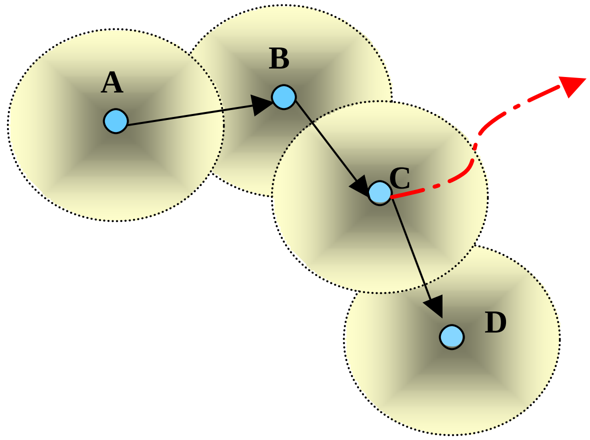

Figure 1: A four nodes network with route from A to D.
If C moves in the direction shown, the route will break.

While the nodes move closer or a way from each other, routes are created or destroyed. For example, in Figure 1 if node C moves in the indicated direction, the route from B to D will break and A won't be able to send to D unless there is an alternative route. Therefore, the routing protocol has an extra requirement of continuously validating routes to destinations considering the fact that mobility of nodes affects links validity. That is in addition to its original routing purpose, which is finding the optimal path that gives minimum delay or cost.

Several protocols have been proposed for MANET routing such as the DSDV (Destination-Sequenced Distance Vector) routing [8] , TORA (Temporally Ordered Routing Algorithm) [5] , DSR (Dynamic Source Routing) [4] , and AODV (Ad Hoc On Demand Distance Vector) routing protocol [6] . A MANET routing protocol could be a proactive protocol (table-driven), a reactive protocol (demand-driven), or a hybrid one. DSDV falls in the first category while AODV, TORA, and DSR fall in the second category.

Proactive protocols maintain routing information in tables prior to actual data transmission. When requests arrive, the protocol uses such data for forwarding. This approach has challenges in the mobile environments since links are frequently destroyed and new links are newly generated. With such changing environment, maintaining an up-to-date route data will be costly. Demand driven routing protocols create routes only when a route is needed to send data from a source to destination. Thus the demand driven protocol adapts to frequent link changes. Hybrid approaches attempt to combine the advantages of being proactive and reactive.

3.1 Project Description

The purpose of this project is to simulate an ad hoc network routing protocol to study its performance. The Ad Hoc on Demand Distance Vector Routing Protocol (AODV) is an on-demand protocol that has recently received much attention [1] [6] [7] . In this project AODV is considered as the main protocol for simulation. OPNET is used to simulate AODV. An AODV OPNET initial simulation model can be obtained from [9] and is modified to reflect proposed changes to the protocol. Several simulation scenarios have been conducted before and after applying the modifications.

In this project, an enhancement is suggested to dynamically adjust protocol parameters (such as the frequency of hello messages) in response to network state. We consider the network state in terms of the amount of mobility the nodes exercise at certain point of time in relation to the density of nodes per unit area. As a result, a higher efficiency can be achieved when there is a high load on the ad hoc network.

3.2 Report Organization

This remainder of this report is structured as follows: section 4 gives a quick overview of the AODV protocol. Section 5 provides a description of the contributed simulation model in OPNET. Section 6 discusses the criteria, provided by IETF, that is important in simulating an ad hoc network. Section 7 presents the results obtained in simulating AODV and analyzes some shortcomings. Section 8 presents an enhancement to AODV. Section 8 presents the implementation in OPNET and its corresponding performance evaluation. Section 9 provides final discussions and conclusions.

4. AODV PROTOCOL OVERVIEW

As mentioned earlier, AODV is an on demand protocol. Routes are established only on requests to transmit data from a source to destination. To create routes, AODV performs route discovery cycles in two stages: *reverse route setup* and *forward path setup*. When the discovery cycle is complete, the route will be stored in routing tables and be available for routing. The route discovery cycle uses two messages: Route Request (RREQ) and Route Reply (RREP). When a sender wants to send data to destination, and if there is not an already available route, a route should be established between the source and the destination (path discovery). First, the sender broadcasts a RREQ to its neighbors. Nodes within the transmission range of the sender node receive the RREQ. If the neighboring node has enough information (in route table) on how to reach the destination, it will notify back to the sender. Otherwise, it will rebroadcast the RREQ to its neighbors again. The RREQ will be rebroadcast until it reaches either the destination or a node with up-to-date information of a route to the destination. While the RREQ is propagated, each node records obtained data from the source on how to reach it in the reverse path. For that reason, this process is called *reverse route setup*. Figure 1 illustrates the propagation of a RREQ message from a source to destination.

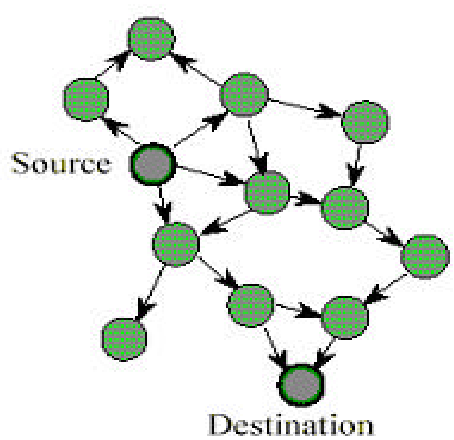

Figure 2: RREQ propagation - reverse path setup

The node with information to the destination (or the destination itself) will issue a RREP message that will propagate in the reverse path. While the RREP is being retransmitted in the back direction, nodes along the path update their route table entries to indicate the route information to the destination. This process is called *forward path setup*. Figure 3 illustrates the RREP traversing the prior established reverse route thus forming the forward path.

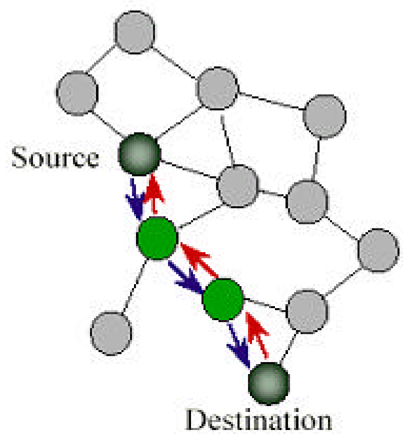

Figure 3: RREP propagation - forward path setup

Periodically (by a specified interval), nodes announce their presence by issuing Hello messages. If a node doesn't hear from a prior neighbor for a period that is larger than the specified interval (*hello_interval*), it concludes link failure that may imply route invalidity. AODV uses a fixed hello_interval of one second. When link failures occur, route tables residing in other nodes should be repaired. A RRER is propagated in the MANET until it finds an alternative route. Please refer to [6] [7] for further details about AODV. The OPNET implementation is described in the next section and the extension to the implementation is provided in section 8.

5. CONTRIBUTED AODV OPNET MODEL OVERVIEW

The National Institute for Science and Technologies (NIST) previously implemented an OPNET model for AODV. This model is obtained from the OPNET contributed models' repository [9] . Some documentation about NIST-AODV is also available at [10] . The following section provides a general description of the model as well as notes encountered on experimenting NIST_AODV.

5.1 Node Model

The node model is illustrated in Figure 4. The *src* module is responsible for generating packets. This acts as the source of data packets in the AODV_DATA_PAYLOAD format. The packets are sent to the lower layer (*app_manager*). The application manager process (*app_manager*) acts as the application layer that issues directed requests to specific destinations (thus initiating routes discovery). Since such service requests are directed to recipients, they need to be routed. Therefore, the application level PDU's are sent to the routing layer (*aodv_routing*) that implements the AODV routing protocol. The routing process is described in detail in section 5.3. Only data packets arrive to the application layer. When the data packet arrives to the destination, it is destroyed simulating the processing of the packet by the application.

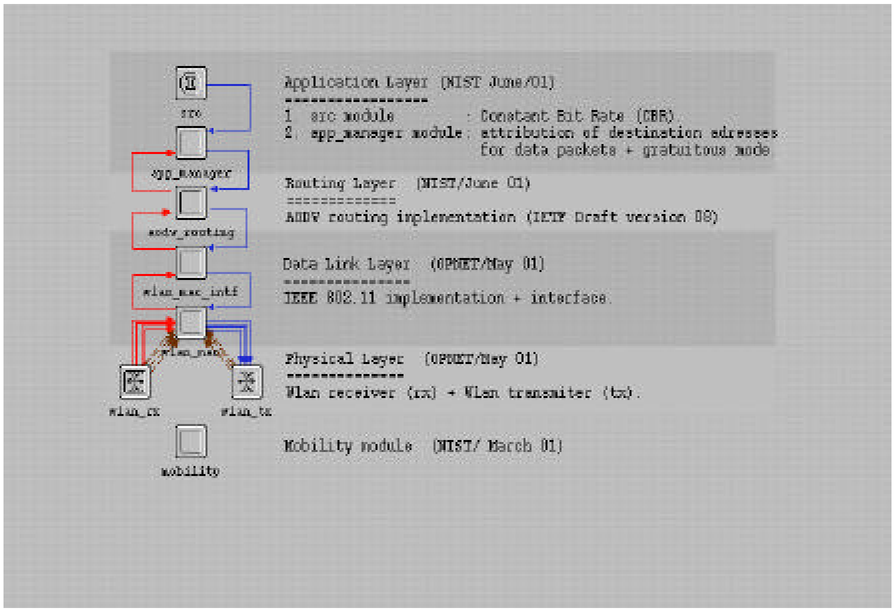

Figure 4: Contributed AODV node model

This model relies on IEEE 802.11 as a standard for Medium Access Control (MAC). The data link layer and the physical layer are represented by *wlan_mac_intf* and *wlan_mac* processes respectively.

To simulate the mobile environment, it is required to implement node mobility to test the protocol operating on non-stationary nodes. A commonly used mobility model for simulations is the random waypoint mobility model, which is described in section 5.2.

5.2 The Mobility Process Model

The mobility process model is illustrated in Figure 5. Initially, nodes are placed randomly in the physical area (1000m X 1000m). If mobility of nodes is allowed (Mobility attribute is equal to 'enabled'), the node will start (going to Init_Mvt state). It will then select a random destination to follow. In the 'Move' state, the node will start to move to that destination with a random speed (less than a specified maximum, SPEED_LIMIT mobility attribute) according to a predefined step (MVT_STEP, 0.2 seconds). After arrival to the destination, it will pause (Idle state) for a random time and start over again.

If the node is stationary (mobility attribute is set to disabled), the node will remain in the idle state. When the mobility attribute is set to ‘enabled’, it is possible to visualize an animation of the nodes movement versus simulation time by using the Play Animation option available in OPNET (after recording animation). Otherwise, the animation will show the initial location of nodes upon starting the simulation. Two snapshots at two different times of the animation for a three nodes mobile network are given in Figure 6 and Figure 7.

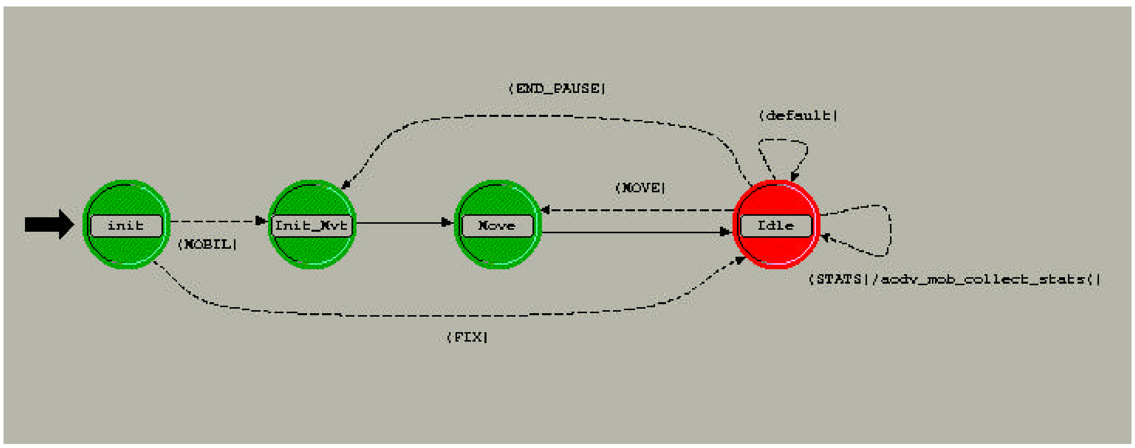

Figure 5: Mobility process model

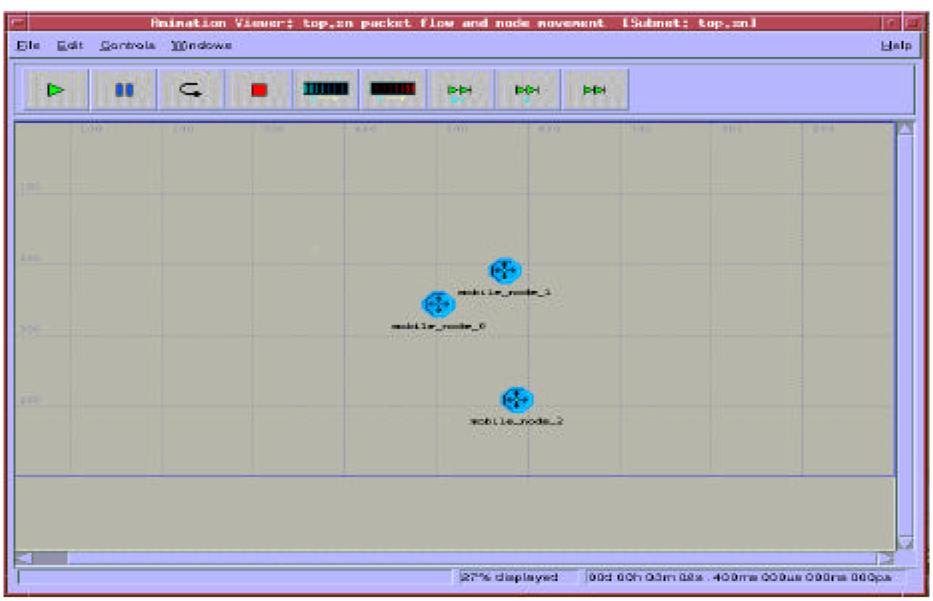

Figure 6: Animation for a three nodes mobile network

Figure 7: Animation for a three nodes mobile network after time

5.3 AODV Routing Process Model

The routing process model (aodv_routing) is the heart of the model as it implements AODV routing protocol. The init state is for initializations and for reading global and object attributes. After the 'Init' sate, control is moved to the idle state. Events will trigger further actions. Events could be scheduled interrupts (initialized at the init state) or events from other states. Every *hello_interval*, an interrupt is generated and state transition occurs to move the state to Say_Hello state in which it broadcasts Hello message and returns back to its idle state. The node moves to the Rcv_Appl state on receiving a request from the application layer (app_manager) to transmit a data packet to a destination. If there is a fresh route to destination, the packet is forwarded to its destination. Otherwise, a route request (RREQ) is initiated (will be described later). An incoming packet arrived to the MAC layer initiates the transition to Rcv_Mac state. Other states are used to process route table, handle link failures, or to acknowledge messages. The *stat* state is used for collecting statistics. In addition to the model-collected statistics, statistics are written to a Comma Separated file (CSV).

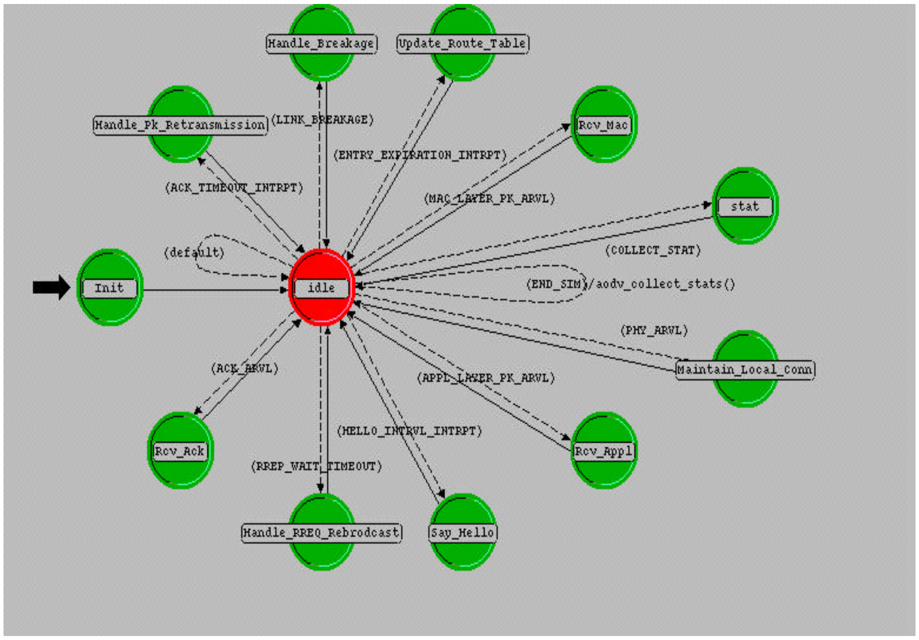

Figure 8: The routing process model (aodv_routing)

Each route table entry has a timer, unless the entry is renewed (e.g. receiving hello messages), the route entry will expire to reflect changes in network topology. Updates to the route table are handled by the Update_Route_Table state. Other C functions for handling the route tables are provided by the: aodv_route_table_entry and aodv_route_table_stat external c programs.

5.4 Simulation Scenarios and Configuration Parameters

Table 1 and Table 2 list important attributes used for the simulation scenarios given below along with their explanation.

Table 1: Global attributes description

Attribute	Description
MAX_NB_FLOWS	Maximum number of flows (35)
Wireless LAN range	802.11 transmission range (set to 250 m)

Table 2: Promoted object attributes

Attribute	Description
HELLO SUPPORT	Connectivity notification type of support. Whether hello messages are supported
AODV parameters	Aodv parameters
Mobility Parameters	Mobility model parameters
SAT_CHECK	Whether saturation_status is checkable or not
SAT_CHECK_INTRVL	Interval in which saturation_status state to be triggered (only if SAT_CHECK is 1)

Gathered statistics are given in a subsequent section after a brief discussion of IETF considerations on MANET performance.

6. IETF PERFORMANCE METRICS

Ad hoc networks are special network paradigms characterized by the operation of wireless nodes that are possibly mobile. The IETF – RFC 2501 [2] discusses performance considerations for MANETs. It reinforces the following significant characteristics of the MANET:

1. Dynamic Topology: As discussed above, when nodes move, links are dynamically created or destroyed.
2. Bandwidth constrained: Interference conditions of the wireless links will hinder the effective bandwidth.
3. Energy constrained: Wireless nodes operate on battery power. Therefore, it is important to save lifetime of the batteries to allow extended duration of operation. The ad hoc routing protocol should eliminate unnecessary transmissions.
4. Security: in this work, this factor is not studied.

The RFC in [2] suggested quantitative metrics to assess the performance of the MANET routing protocol. The end-to-end delay statistical measure is an important performance measure of the ad hoc routing protocol. Another important measure, especially for a demand-driven protocol, is the route acquisition time. This is the time taken to establish a route from the time it was requested.

Since the MANET experiences frequent changes to its topology, the routing protocol of the ad hoc network will impose control traffic overhead to maintain the network connectivity. In AODV, the use of Hello messages is an example of such control traffic. The excessive use of control traffic will limit the network capacity to deliver data packets. Since the MANET is sensitive to bandwidth and energy consumption, as mentioned above, effective use of control packets is important to improve the efficiency of the ad hoc network.

7. SIMULATION RESULTS AND COMPARISONS

This section presents the simulation results for experiments conducted on the AODV model. The collected statistics include the following global statistics:

1. Average Delay: this is the average end-to-end delay (in seconds) versus the simulation time. This is collected by measuring the difference of time between sending the data from the source till it is received to its destination normalized to the size of the data.
2. Average Discovery Time (seconds): Is an indication of the time taken to establish a route from a source to destination (route acquisition latency)

Other statistics gathered by writing to a CSV file include:

1. Control Traffic (bits): Is the number of bits transmitted across the network that does not count as data packets.
2. Mobility Factor (calculated at the end of the simulation): Total movement of nodes / (num of nodes * simulation time)
3. Data Dropped (in packets)

7.1 Effect of Mobility

Simulation scenarios are performed on stationary and mobile nodes controlled by the mobility attribute. Also it is possible to allow higher mobility by changing the mobility attributes (MAX_SPEED). Simulation results indicate that with increased mobility (measured by mobility factor), the average end-to-end delay increases and the average route acquisition latency increases. This is attributed to the increased frequency of link breakages and the incurred route repairs when nodes move. Some values obtained from various simulation runs (on 40 nodes with hello interval = 1 sec.) are given in Table 3.

Table 3: Mobility effect on discovery time (sec) and delay (sec) for three scenarios

Mobility Factor	Average End-to-end Delay	Average Discovery Time
0	0.636	3.534881
0.785586	2.120	4.588857
1.06443	2.244	5.054393

7.2 Delay and Number of Nodes

Several simulation runs show that the average end-to-end delay increase when the number of nodes in the network increases. This is attributed to the increased chance of collisions in the wireless transmissions. Table 4 shows results obtained for running the simulation scenarios for 3, 16, and 40 nodes respectively. All the simulation scenarios are conducted with parameters set to force stationary network and with hello support. An illustration of the average delay progression for the three nodes network is given in Figure 9 and the delay measure for the 40 nodes network is given in Figure 10. Comparing the two results, it can be clearly shown that there is difference of time it takes the average delay to settle to its final value. This is attributed to the big difference of the number of destination nodes. In a three nodes network, it is easier to find the destination (if connected). Only two subsequent nodes will receive the RREQ, and there is a higher chance that the route is discovered in the first step. In Contrary, for the 40 nodes network, the RREQ needs to traverse many nodes (for example 20 nodes) in the reverse path and on the forward path as well. This process is repeated for all nodes transmitting to other destinations. This process is repeated until all the required routes are discovered.

Consequently, there is no need to re-establish the routes unless the routes expire. Since the simulation is run on stationary network, the routes will eventually become available for subsequent transmissions and the average-end-to-end delay will stabilize.

Table 4: Average end-to-end delay obtained for three scenarios

Nodes	Average End-to-end Delay (sec.)
3	0.0046
16	0.00464
40	0.636

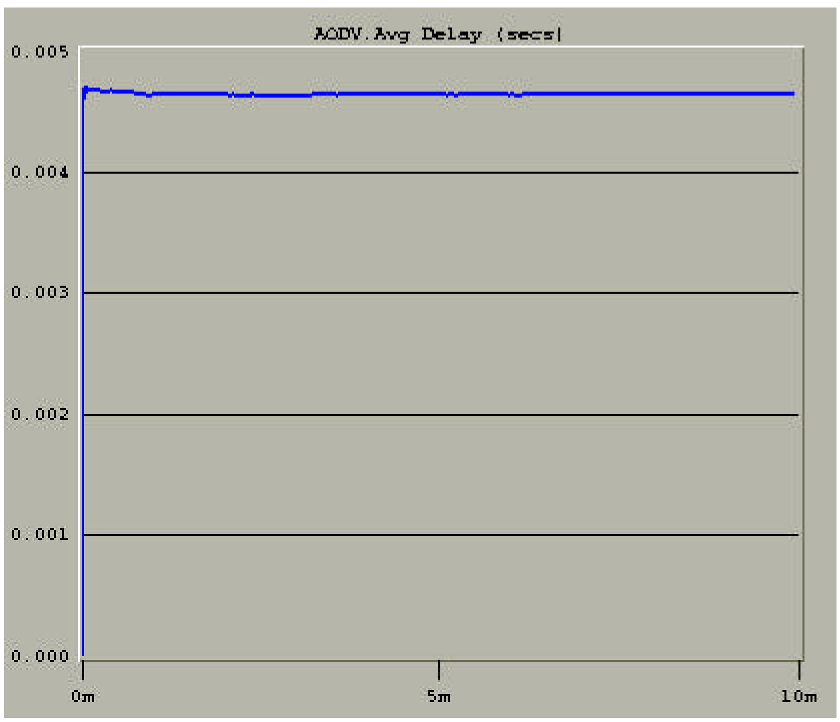

Figure 9: Average end-to-end delay progression for a three nodes stationary network

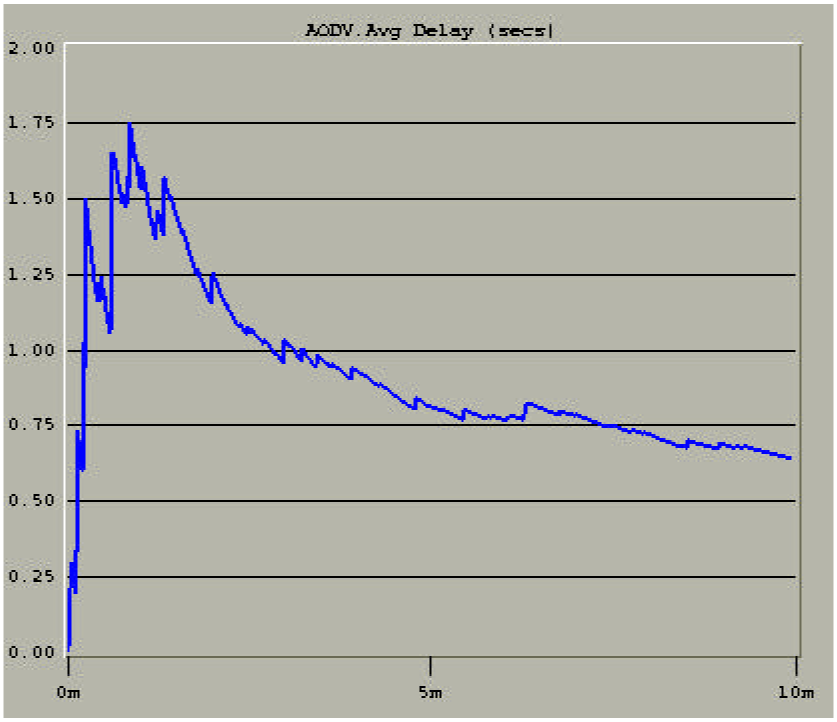

Figure 10: Average end-to-end delay progression for a 40 nodes stationary network

7.3 Connectivity Management Performance

The simulation scenarios' results provided in sections 7.1, 7.2 above have shown the increase of delay upon increasing the number of nodes as well as upon increasing the mobility of nodes. An important characteristic of the AODV protocol is the use of hello messages to maintain connectivity. Nodes announce their presence by frequently broadcasting hello messages to their neighbors (with TTL=1). Once a node receives a hello message from a neighboring node, it will renew its routing table to indicate that the node is reachable with hop count =1. Therefore, the use of hello messages is important to maintain the network connectivity if the nodes are moving. However, if the nodes are stationary, it imposes extra load on the network since it is a broadcast message. To verify the impact of hello messages on a static network, two simulation scenarios were conducted to show this impact. The common attributes setting for the two scenarios are given in Table 5 below.

Table 5: Common simulation parameters

Attribute	Value
Network Dimension	1000m x 1000m
Nodes	40
Transmission Range	250 m
Simulation Duration	10 minutes
Number of Flows	35
Data Rate	1MB/sec.
Hello Interval (If hello supported)	1 sec.

The first scenario consists of a stationary network of 40 nodes while hello messages are not allowed. Figure 11 illustrates the progression of the average-end-to-end delay (Scenario 1). The oscillation is due to the time needed to establish routes. However, it gives a relatively good value (0.26 seconds).

Table 6: Scenario 1 parameters

Attribute	Value
Mobility	No – Stationary Nodes
Hello Support	No Hello Support

The second scenario is based on 40 nodes network with all nodes are stationary while hello messages is allowed. The average end-to-end delay is plotted in Figure 11 (Scenario 2). Obviously, adding hello messages while the network is stationary gives a negative effect on the performance of the MANET. On the other hand, the use of hello messages is important in situations that the network is mobile. A quick overview of the results obtained for running scenarios for mobile networks show that the use of hello messages improves the MANET performance (measured by end-to-end delay). A dynamic adjustment of hello messages frequency based on the network state will enhance the performance of the MANET as discussed in the next section.

Table 7: Scenario 2 parameters

Attribute	Value
Mobility	No – Stationary Nodes
Hello Support	Hello Support
Hello Interval	1 sec

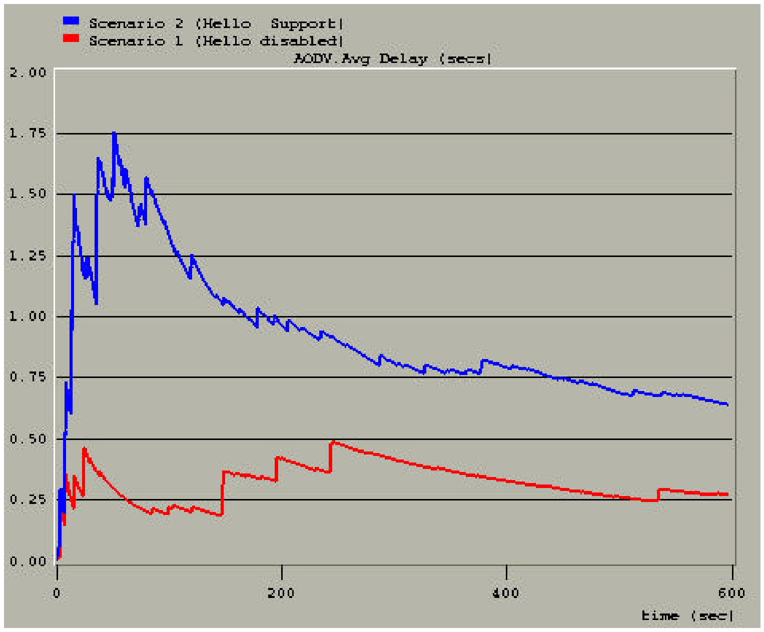

Figure 11: Effect of using hello messages for a stationary network

8. DYNAMIC CONNECTIVITY MANAGEMENT BASED ON NETWORK SATURATION

From the results presented in section 7, we can propose that using hello messages should be dynamic based on the network state. This should also apply to other control traffic of the routing protocol and to the lifetime of the routing entries. The goal is to dynamically allow hello messages in situations that it is useful and to disallow it in situations that its usage will harm the performance. In a wireless environment, a broadcasted message increases the chance of collisions and prevents neighboring nodes from transmitting at the same time. When the number of nodes within the same physical space increases, the number of neighboring nodes will increase leading to a higher cost of broadcast. On the other hand, when the number of neighboring nodes increases, there is a higher chance that an alternative route will be found. This suggests the need to reduce hello messages in situations that have higher density of nodes. Another related factor is mobility. When few nodes are highly moving in small physical area, they frequently get disconnected thus requiring frequent hello messages. On the other hand, for nodes that are moving but with high density, there is a higher chance that nodes will find alternative routes since the network does not become disconnected. In the later case, there is a tradeoff between using hello messages to discover the optimum route versus finding any available one since the later requires less control traffic. Figure 12 suggests the dynamic hello messages' requirements where the higher shading indicates higher need for hello messages.

	Few Nodes/Unit Area	**Many Nodes/Unit Area**
Stationary	?? Less Link Breakages ?? Less Sensitive to Control Overhead	?? Less Link Breakages ?? Sensitive to Control Overhead (Broadcasts)
Mobile	?? Expected frequent Link Breakages ?? Less Sensitive to Control Overhead	?? Sensitive to Control Overhead (Broadcasts) ?? Better Connected

Figure 12: Hello messages' requirement

We believe that both factors (mobility and density of nodes) correlate to form network saturation level. For a highly mobile network, there is a high chance that nodes will become outside the transmission range resulting in a higher dynamic topology. In this case, the use of hello messages is needed more frequently. However, in a dense network, collisions will more likely occur when there is a high amount of traffic. In this case, we need to eliminate control traffic as much as possible knowing that there is good connectivity. In sparse networks, connectivity will be weak. Isolated nodes will need to be discovered to join the network. A measure combining the two factors could be the mobility factor given as: amount of total nodes' movement/number of nodes*simulation time). Since the physical area is constant (1000mX 1000m), a higher number of nodes will decrease the mobility factor. On the other hand, when the nodes move frequently, the mobility factor will increase. Therefore, a low mobility factor can indicate less need to hello messages.

To validate this idea, a modification to the AODV model is implemented in OPNET. This modification is illustrated in the next section.

8.1 Implementation Overview

A new process (Update_Saturation_Status, see Figure 13) is added to the AODV process model, see Figure 13. The process adopts with the following procedure:

1. Read the (SAT_CHECK) attribute value set at the scenario
2. If the attribute value of SAT_CHECK is set to 'enabled' (1), initialize an interrupt in the init state. Set it to fire for the first time after SAT_INTRVL_CHECK interval. This interrupt triggers the Update_Saturation_Status state
3. When the Update_Saturation_Status process is entered (Enter execs), it will check the hello parameter, and calculate the mobility parameter by the following equation:

   Mobility Factor = Total nodes' movement / (number of nodes * sim. time)
4. Will check the network condition and roughly proceed to do one of cases
   a. If the network is stationary (zero mobility factor) and no hello messages is allowed, proceed with no changes
   b. If the network is stationary (zero mobility factor) but supports hello messages:
      i. Turn off hello messages
      ii. Extend the lifetime of the routes in nodes' route tables
   c. If high mobility factor, then allow hello messages by turning on the hello mode.

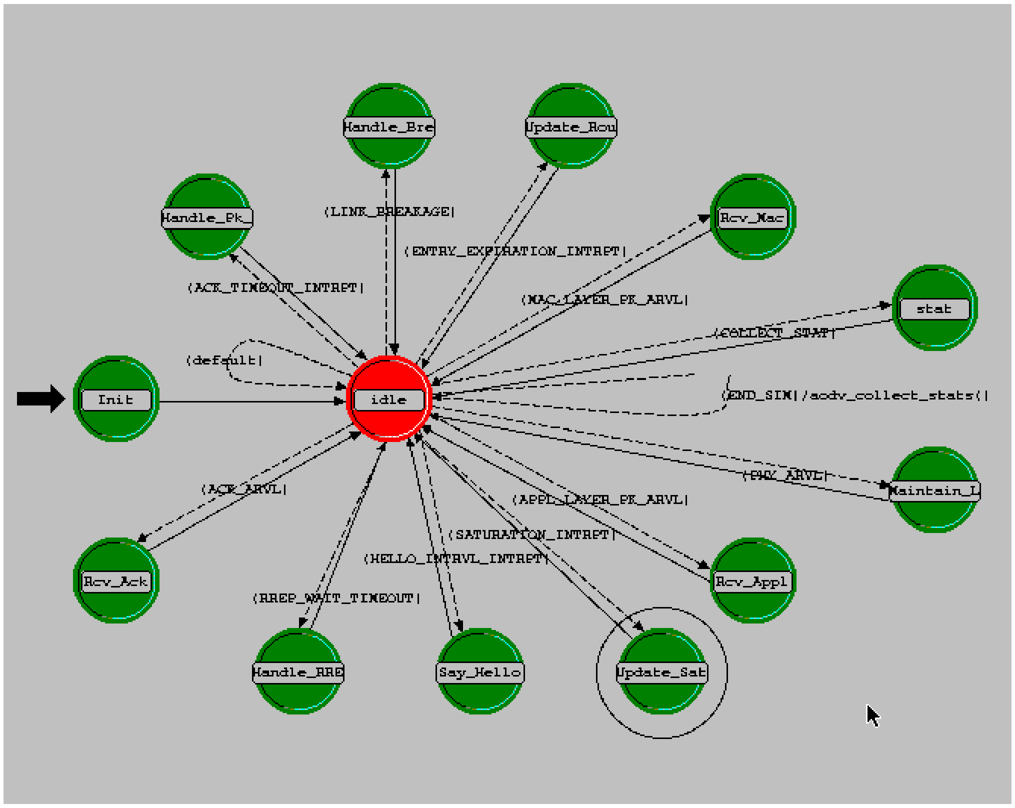

Figure 13: Update saturation status process

The code listing is provided in the Appendix. In the simulation runs, the SAT_INTRVL_CHECK is set to a high value (40 seconds). Note that nodes move much slower than network transmissions. When adjusting the value of the SAT_INTRVL_CHECK, some considerations might be taken. A higher time interval such as 40 seconds might be relatively needed to record/observe the movement of nodes. Using the way point mobility model incorporated in NIST-AODV, the node can move outside the range of another node within (250/20=12.5 seconds) or (500/20=25 seconds) depending on the location of nodes. Also, the node will pause for an interval of 60 seconds, i.e. there is no change to that node for 60 seconds.

In some scenarios, such as in a conference room, people movement or attendees switching off their notebooks leads to the node being neither able to receive nor act as a router. Such changes occur at far lower rates than the actual transmission delay of packets.

It is also important to note that while the overall mobility factor is recalculated (update_saturation_status) for all nodes in the simulation, there is a need to verify how it will be calculated in actual MANETS. In actual MANETS, this measure can be gathered by:

1. Using motion sensors to gather network movement or
2. A node may update its mobility parameters, the protocol will then collect these statistics or
3. A node can count the changes of the neighboring nodes and report changes to give indications on the number of nodes of the network and their mobility

8.2 Simulation Results

Hello messages are part of the control traffic gathered in a CSV file. Figure 14 shows the amount of control traffic incurred when using a static hello message versus the amount of dynamically adjusting hello messages described above. It also shows that the control traffic has been significantly reduced when the hello messages are dynamically eliminated. This method has also shown reduction in the number of dropped packets.

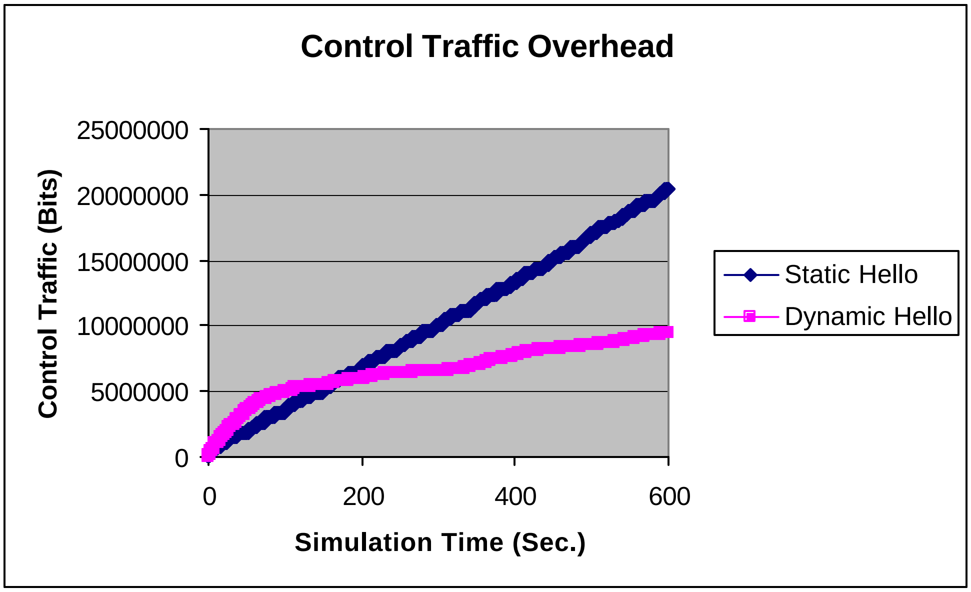

Figure 14: Effect of dynamic hello messages on control traffic

Comparison in delay for the same scenarios is given in Figure 15. It shows that for the simulation time conducted, the average-end-to-end delay with hello message reduction is much better than the one initially set with using hello messages. This verifies that the usage of hello messages is not necessarily useful to the MANET performance in all situations. It also verifies that it is possible to dynamically alter the hello interval to respond to network conditions. While the implementation done in this project was directed to validate this idea, further studies could help to answer the following questions:

1. What are the best values of SAT_INTRVL to update the network status?
2. What is a better measure than the mobility factor to adjust the hello interval accordingly and how?

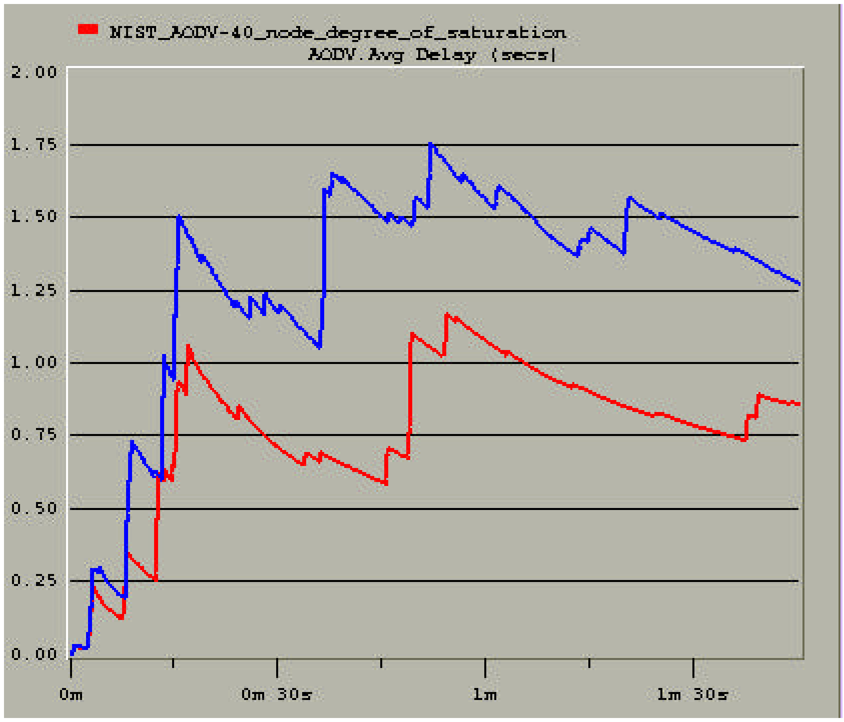

Figure 15: Effect of using degree of saturation on delay

9. DISCUSSION AND CONCLUSIONS

In a mobile ad hoc network, MANET, the network is infrastructure-less, built on the fly, and the nodes are free to join and leave the network on an ad hoc basis. While nodes should act as routers, routes become disconnected and reconnected resulting in a highly dynamic topology. Therefore the routing protocol is central to ad hoc networks. On demand routing protocols, such as AODV, create routes only when needed. It is found that connectivity management is required to update the routing information in response to nodes' movement. Since this challenging paradigm is useful to many applications, the IETF has studied the MANET performance requirement.

In this project, performance studies of AODV were conducted investigating several aspects of its operation including number of nodes effect, mobility, and the use of hello messages. The study has found a trend to a decrease in performance when mobility of nodes or number of nodes increases. It has also demonstrated that the use of hello messages provided less performance when the nodes were static. This performance degradation is attributed to the incurred broadcast overhead in situations that don't require the control messages. Such overload hinders the network efficiency in these situations. Thus AODV suffers from a major problem that the hello interval is always a constant (1 sec.) irrespective of whether nodes are moving or not, the network is crowded or not. While AODV assumes that the nodes are mobile, there are situations that the mobility is less than expected.

The main idea behind the proposed enhancement by this project is to dynamically adjust the control messages to cope with the status of the network (adjusting the hello interval). In this case, a relief of the overhead of control packets will improve performance based on the network state of mobility and density of nodes/unit area. Simulation studies of the proposal have shown improvement in the performance in terms of reduction of average delay, reduction of the overall control overhead, and reduction in dropped packets. Thus verifying that:

1. The use of hello messages is not necessarily useful in all situations.
2. It is possible to dynamically adjust the hello interval, extend or shorten the lifetime of route table entries, which proved success in the simulations conducted.

Currently, the proposed solution suggests the need to dynamically adjust the hello_interval in AODV. The hello interval was adjusted roughly. Future studies can reveal the correlation that will give the best value of hello_interval in relation to the network condition. Also, to investigate how frequently the network state should be checked to allow this modification. Consequently leading to achieve better MANET scalability.

REFERENCES

[1] I. Chakeres and E. Royer, "The Utility of Hello Messages for Determining Link connectivity", The Fifth International Symposium on Wireless Personal Multimedia Communications, Oct. 2002.

[2] S. Corson and J. Macker, "Routing Protocol Performance Issues and Evaluation Considerations" in IETF RFC 2501.

[3] M. Günes, U. Sorges, I. Bouazizi, "ARA The Ant-Colony Based Routing Algorithm for MANETs" Int. Workshop on Ad Hoc Networking (IWAHN 2002), Vancouver, British Columbia, Canada, August 18-21, 2002.

[4] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks, " Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer Academic Publishers, 1996.

[5] V. Park and M. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," In IEEE Conference on Computer Communications, INFOCOM, April 1997.

[6] C. Perkins and E. Royer, "Ad-hoc On-Demand Distance Vector Routing" In IEEE Workshop on Mobile Computing Systems and Applications, February 1999.

[7] C. Perkins, E. Royer, and S. Das, Mobile Ad Hoc Networking (MANET), "Ad-hoc On-Demand Distance Vector (AODV) Routing", IETF RFC 3561.

[8] C. Perkins and P. Bhagwat, "Highly Dynamic Destination Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," ACM SIGCOMM Computer Communications Review, October 1994.

[9] AODV Contributed Model (NIST), OPNET Technologies Inc, www.opnet.com

[10] Simulation Model for the AODV MANET Routing Protocol, Advanced Network Technologies Division, National Institute of Science and Technologies (NIST), http://w3.antd.nist.gov/wctg/manet/prd_aodvfiles.html

APPENDIX

CODE LISTING 1 – Modifications to Init: Enter Execs

PS: This code is added to the end of the init state, enter execs.

```
// updates by Mohamed Soliman
//            Simon Fraser University
//            msoliman@cs.sfu.ca
// The purpose of this update is to trigger the saturation_update state
// only once  - currently the saturation interval
// Triggered if SAT CHECK mode is set

op_ima_obj_attr_get (op_id_self(), "SAT CHECK"
,&NETWORK_SAT_CHECK);

// getting the Checking interval

op_ima_obj_attr_get (op_id_self(), "SAT INTRVL CHECK"
,&SAT_INTRVL_CHECK);

if(NETWORK_SAT_CHECK)
      op_intrpt_schedule_self(op_sim_time()+ SAT_INTRVL_CHECK,
SATURATION_CODE);



if(my_node_addr == (num_nodes -1))
      {
      // init file
      aodv_stat_init_csvf();
      op_intrpt_schedule_self(op_sim_time()+ update_stat_intrvl,
STAT_CODE);

      }
```

CODE LISTING 2 – Update_Saturation_Status: Enter Execs

Update_Saturation_Status is a new process added to NIST_AODV contributed models. It is triggered by the SATURATION_INTRPT. The init process triggers SATURATION_INTRPT initially. Its frequency is defined by a node attribute (SAT_INTRVL_CHECK).

```
//
// BY:  Mohamed Soliman
//      Engineering Science
//      Simon Fraser University
//      Email: msoliman@cs.sfu.ca
//
// Purpose: To check the network saturation status and adjust
frequencies of
//          control packets
// N.B. Triggering of this state is scheduled for first time by the
init state
//      Also attributes of triggering period and whether this state is
enabled
//      are initialized from init as well

printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
printf("Network Saturation status is checked\n");
printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");

//--------------------------------------------
// Step 1.: Calculating the mobility factor
//--------------------------------------------

int   i;
double mob_factor_recalc = 0.0;

for(i=0; i < num_nodes;i++)
  {
      // Step 1.1 Calculating average movement of node
    // Both arrays used below are used from the mobility process model

      avg_deplacement_per_node[i] = (double)
total_relative_deplacement[i]/op_sim_time();
    //   printf("Avg Relative Mvt for %d =
%f\n",i,avg_deplacement_per_node[i]);
  }

// Step 1.2: compute the avg movement for all nodes

mob_factor_recalc = 0;
for(i=0; i < num_nodes;i++)
```

```
    {
      mob_factor_recalc+= avg_deplacement_per_node[i];
       }

//   printf("Total Relative Mvt for %d =
%f\n",my_node_addr,total_relative_deplacement[my_node_addr]);

// Step 1.3: Recalculate the current mobility factor

mob_factor_recalc = mob_factor_recalc/num_nodes;
printf("Current Mobility Factor:%f\n",mob_factor_recalc);


//-----------------------------------------------------------------------
--------
// Step 2. if the network is operating in
//  hello mode and stationary ==> switch to no hello and extend routes
lifetime
//-----------------------------------------------------------------------
---------


if(HELLO_MODE)
      {
      printf("Network operating in hello mode\n");
    printf("If nodes are stationary, switching to elimination of
hello\n");

      //---------------------------------------
      // Step 2A.2 Canceling hello mode
      //---------------------------------------
      if(mob_factor_recalc<0.1) // stationary nodes
            {
            printf("Network is currently stationary, canceling hello
messages\n");

            HELLO_MODE = 0;
            CONN_SUPPORT_TYPE = NONE;

            //-------------------------------------------------------
            // Step 2A.2 (cont.): canceling waiting hello interrupts
            //-------------------------------------------------------

            if (op_ev_pending (hello_intrvl_evh))
                  op_ev_cancel(hello_intrvl_evh);

            //----------------------------------------------------------
-------
            // Step 2A.3 since nodes are stationary, topology is less
dynamic
            //
            // Should input code here to either
            // 1. Extend all entries in the route table ( can write a
function for that
```

```
            // 2. update a dynamic factor that is always considered for
route table expiration
            // Notice that at this point of time many route table
entries are scheduled

            // for expiration

            // 2A.31: extend DELETE_PERIOD to a higher arbitrary value

            DELETE_PERIOD = DELETE_PERIOD*5;
            }

      }
else
    {
      //----------------------------------------
      // Step 2B: Network is not in Hello mode,
      //          if nodes are highly moving (high mobility factor)==>
    //              switch to hello_mode
      //---------------------------------------
      printf("HELLO Mode is not currently supported\n");
      if(mob_factor_recalc>0.7)
            {
            printf("Network has a higher mobility factor while not in
HELLO Mode\n");
            printf("==> Switching to Hello mode\n");
            // Supporting HELLO of RREQ
            HELLO_MODE = 1;
            CONN_SUPPORT_TYPE=HELLO;

          // Initializing necessary variables to send hello messages
            // conditions not executed in init state

            hello_msg_type=RREQ_PK_TYPE; // only supporting RREQ types
            hello_msg_template = op_pk_create_fmt("AODV_RREQ");
            op_pk_nfd_set(hello_msg_template,"TTL", 1);
            op_pk_nfd_set(hello_msg_template,"G"  , 1);
            // Set size
            op_pk_total_size_set(hello_msg_template, 192);

            // Assign source and dest

            op_pk_nfd_set(hello_msg_template,"Source"
,my_node_addr);
            op_pk_nfd_set(hello_msg_template,"Dest"
,my_node_addr);
            op_pk_nfd_set(hello_msg_template,"HopCount"   ,0);

            // trigger first intrpt for hello broadcast

            hello_dist = op_dist_load ("uniform_double", 0.0,1.0);
            first_hello_intrvl = op_dist_outcome(hello_dist);
```

```
            hello_intrvl_exp =
op_intrpt_schedule_self(op_sim_time()+first_hello_intrvl,HELLO_CODE);
            // Adjusting_parameters that might have changed due this
modification

            DELETE_PERIOD = 5*max_dbl(ACTIVE_ROUTE_TIMEOUT,
ALLOWED_HELLO_LOSS*HELLO_INTERVAL);



            }

      }


//----------------------------------------------------------
// Step last: reschedule for rechecking after SAT_INTRVL_CHECK (50sec)
//----------------------------------------------------------

op_intrpt_schedule_self(op_sim_time()+ SAT_INTRVL_CHECK,
SATURATION_CODE);
```