Modeling and Simulation of IEEE 802.11 Wireless LANs: A Case Study of a Network Simulator

Chapter · January 2012

DOI: 10.4018/978-1-4666-0191-8.ch005

Nurul Sarkar
Auckland University of Technology

R. W. McHaney
Kansas State University

Chapter 5
Modeling and Simulation of IEEE 802.11 Wireless LANs:
A Case Study of a Network Simulator

Nurul I. Sarkar
Auckland University of Technology, New Zealand

Roger McHaney
Kansas State University, USA

ABSTRACT

Stochastic discrete event simulation methodology is becoming increasingly popular among network researchers worldwide in recent years. This popularity results from the availability of various sophisticated and powerful simulation software packages, and also because of the flexibility in model construction and validation offered by simulation. In this chapter, the authors describe their experience in using the network simulator 2 (ns-2), a discrete event simulation package, as an aid to modeling and simulation of the IEEE 802.11 Wireless Local Area Networks (WLANs). This chapter provides an overview of ns-2 focusing on simulation environment, architecture, model development and parameter setting, model validation, output data collection and processing, and simulation execution. The strengths and weaknesses of ns-2 are discussed. This chapter also emphasizes that selecting a good simulator is crucial in modeling and performance analysis of wireless networks.

INTRODUCTION

The use of discrete event simulation packages as an aid to modeling and performance evaluation of WLANs has grown in recent years (Bianchi, 2000; Chen, Jian, & Lo, 2002; Das, Castaneda, & Yan, 2000; Fantacci, Pecorella, & Habib, 2004; Tickoo & Sikdar, 2003). This popularity is due to the availability of sophisticated simulation software packages and low cost powerful personal computers (PCs), but also because of the flexibility in rapid model construction and validation offered by simulation.

A detailed discussion of simulation methodology, in general, can be found in numerous literature (Carson, 2004; Law & Kelton, 2000).

DOI: 10.4018/978-1-4666-0191-8.ch005

More specifically, Pawlikowski (1990) in a comprehensive survey of problems and solutions suited for steady-state simulation highlighted the relevance of simulation technique for modeling and performance evaluation of telecommunication networks. This view of simulation and modeling is frequently supported in the wireless communication and networking literature (Hassan & Jain, 2004; Holloway, 2003; Nicopolitidis, Obaidat, Papadimitriou, & Pomportsis, 2003; Sarkar & Pawlikowski, 2002).

A typical WLAN can easily be simulated and its performance evaluated by a network software package (i.e., simulator). It is important for researchers to choose a simulator which is easy to use; more flexible in model development, modification and validation; and incorporates appropriate analysis of simulation output data, pseudo-random number generators, and statistical accuracy of the simulation results (i.e., desired relative precision of errors and confidence interval). These aspects of credible simulation studies are recommended by leading network simulation researchers (Law & Kelton, 2000; Pawlikowski, Jeong, & Lee, 2002; Schmeiser, 2004).

While various simulators exist for building a variety of WLAN models, we briefly describe two popular network simulators namely, ns-2 (Fall & Varadhan, 2011) and OPNET Modeler (www.opnet.com). The ns-2 simulator is one of the most commonly used simulators today and is very popular with researchers, including CS and EE students worldwide. The ns-2 is open-source software and provides an environment for rapid model construction and simulation output data collection.

OPNET, developed by OPNET technologies, is another popular commercial software package commonly used by researchers and practitioners for modeling and performance evaluation of telecommunication networks. It has a robust and flexible wireless node model which consists of process models of the different layers of the network protocol stack. As ns-2, OPNET is an object-oriented simulation package. However, unlike ns-2, it is totally menu-driven with an easy-to-use Graphical User Interface (GUI) for rapid model construction, data collection and other simulation tasks. It is often of interest to study a proposed or existing wireless network to gain insight into its expected behavior. However, since experimentation with the live network is disruptive and not very cost effective, a model is required for this purpose.

This chapter emphasizes that selecting a good network simulator is crucial in modeling and performance analysis of wireless communication networks. Both the ns-2 and OPNET offer flexibility in model construction and validation, and incorporates appropriate analysis of simulation output data, pseudo-random number generators, and statistical accuracy of the simulation results. Without an underlying framework for the model, a valid, verifiable model become much more difficult to develop, particularly in the time-constrained environments found in many wireless application areas.

The remainder of this chapter is organized as follows. We first provide an overview of ns-2 simulator and then describe our experiences in using ns-2 as an aid to modeling and performance evaluation of IEEE 802.11 WLANs. The strengths and weaknesses of ns-2 are discussed, and a brief conclusion ends the chapter.

THE NS-2 SIMULATOR

Environment

Ns-2 is an object-oriented discrete-event network simulator originally developed at Lawrence Berkeley Laboratory at the University of California, Berkeley, as part of the Virtual InterNetwork Testbed (VINT) project. Berkeley released the initial code that made wireless network simulation possible in ns-2. The Monarch project at Carnegie Mellon University has extended the ns-2 with support for node mobility, a realistic physical layer model and an implementation of the IEEE 802.11b Distributed Coordination Function (DCF) Medium Access Control (MAC) protocol (CMU Monarch Project, 2011).

Ns-2 is an open source simulation package which has improved significantly over time through various contributions made by network researchers worldwide. It has three substantial changes from ns version 1: (1) the more complex objects in ns v1 have been decomposed into simpler components for greater flexibility and composability; (2) the configuration interface is now an object oriented version of Tcl (OTcl); and (3) the interface code to the OTcl interpreter is separate from the main simulator (Fall & Varadhan, 2011). This section provides an overview of ns-2 focusing on environment, architecture, model construction and parameter setting, model validation, simulation execution, and output data analysis.

Architecture

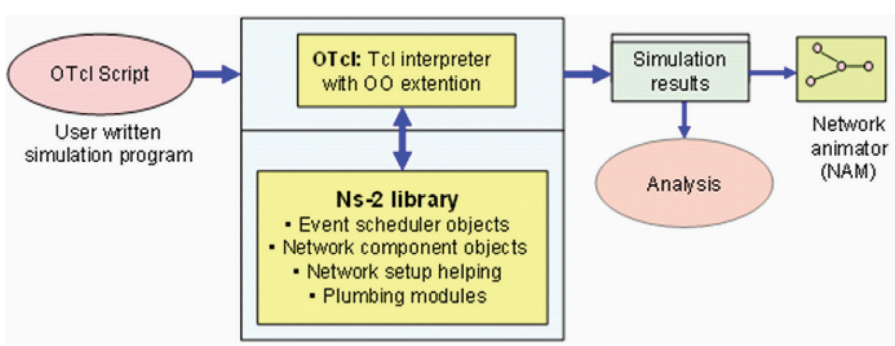
Ns-2 is written in C++ and uses OTcl as a command and configuration interface. The OTcl scripts are used to set up simulation scenarios in the simulator. The main benefit of using OTcl scripts is that there is no need to recompile the simulator between different simulation scenarios. This feature is particularly useful (in terms of saving recompilation time) to study the impact of various influencing factors on the network performance.

Figure 1 shows the architecture of ns-2 illustrating the interaction between a user written program and ns-2 library components. The event scheduler is one of the main components of ns-2. An event in ns-2 is a packet ID that is unique for a packet with scheduled time and the pointer to an object that handles the event. The event scheduler keeps track of simulation time and fires all events in the event queue scheduled for the current time by invoking appropriate network components or objects.

The ns-2 simulator is written in C++ and uses OTcl as a command and configuration interface. The OTcl scripts are used to set up simulation scenarios in the simulator. One of the main benefits of OTcl scripts is that there is no need to recompile the simulator between different simulation scenarios. This feature is particularly useful (in terms of saving recompilation time) to study the impact of various influencing factors on the network performance. By using OTcl scripts, one can easily set up network topologies, specific protocols, link bandwidths, traffic sources and applications to be simulated (these behaviors are already defined in the compiled hierarchy) and the form of the output required.

The ns-2 has a rich library of network and protocol objects called ns objects. These objects

Figure 1. An overview of ns-2 architecture



include nodes, classifiers, links, queues, etc. All objects are derived from a class called NsObject, which is the base for all classes. There are two class hierarchies: the compiled C++ hierarchy and the interpreted OTcl, with one to one correspondence between them. However, the compiled C++ hierarchy provides a greater efficiency in simulation runs in terms of faster execution times. This is particularly useful for detailed analysis of network protocol's behavior.

In ns-2, the timing of events is determined by a scheduler. The scheduler keeps track of simulation time and fires all events in the event queue scheduled for the current time. The influence of network traffic load distribution on network performance is an important observation that interests many researchers. For this task, a variety of traffic generators is needed for automatic traffic creation according to a desired pattern and load. The ns-2 supports several traffic generators, e.g., Exponential ON-OFF, Pareto ON-OFF, and Constant Bit Rate (CBR). More information about the capabilities of ns-2 can be found in the ns manual (Fall & Varadhan, 2011).

Model Construction and Parameter Setting

A network model in ns-2 is constructed by interconnecting several objects, such as nodes, queues, links, and classifiers. The ns-2 objects are built from a hierarchical C++ class structure as illustrated in Figure 2.

Table 1 lists frequently used ns-2 commands and objects for network simulation. More details about ns-2 objects and commands can be found in the ns-2 user manual (Fall & Varadhan, 2011). A user can easily set up and run a simulation by writing an OTcl script using the available simulator objects in the OTcl library. In the script, one defines the network topology, routing algorithm, protocols and applications, initiates the scheduler by instructing traffic sources when to start and stop sending packets, length of the simulation runs, and the form of the output required.

Figure 2. A partial tree of ns-2 class hierarchy

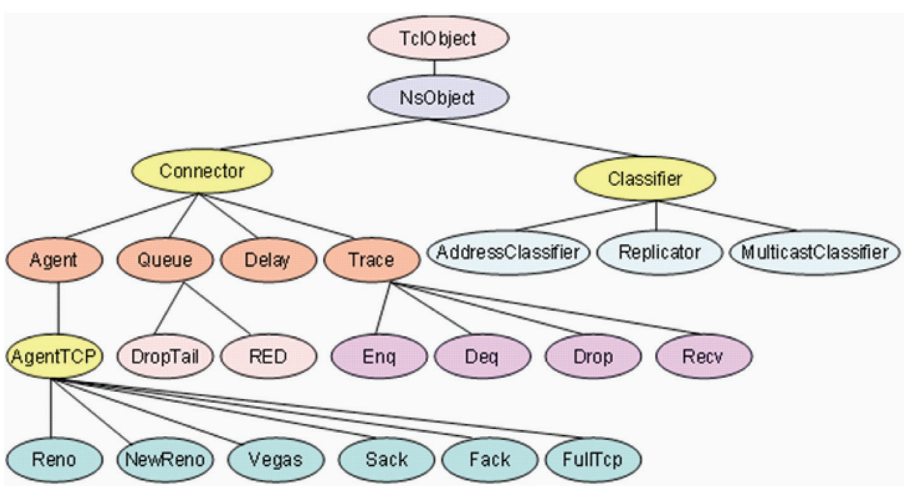


Table 1. Frequently used ns-2 commands and objects

Ns-2 command	Object	Description
set n0 [$ns node]	node	Create a node called n0 on the network.
$ns duplex-link $n0 $n1 6Mb 5ms DropTail	link	Create a duplex link with a drop-tail queue from node n0 to n1 with a capacity of 6 Mbps and a propagation delay of 5 ms.
set udp [new Agent/UDP]	UDP agent	Setup a UDP connection
set tcp [new Agent/TCP/Newreno]	TCP agent	Setup a TCP connection
set ftp [new Application/FTP] $ftp attach-agent $tcp	Application	Setup a FTP application over TCP connection
set cbr [new Application/Traffic/CBR]	Traffic generator	Create a constant bit rate (CBR) traffic generator
set e [new Application/Traffic/exponential]	Traffic generator	Create an exponential on-off traffic generator
set p [new Application/Traffic/Pareto]	Traffic generator	Create a Pareto on-off traffic generator

Model Validation

A main concern in wireless network simulations or any simulation efforts is to ensure a model is credible and represents reality. If this can't be guaranteed, the model has no real value and can't be used to answer desired questions (McHaney, 1991; Sargent, 2004). The ns-2 models described in this chapter are no exception and were subjected to rigorous validation.

Validation is the process of determining the real-world system being studied is accurately represented by the simulation model. Not only does this process provide assurance that the conceptual approach is correct, it establishes an acceptable level of confidence in the conclusions drawn from running the simulation and provides an insight into the true operating characteristics of the modeled system (McHaney, 2009). As with any form of simulation, a simulator plays an important role in building a credible model for the system under study. Therefore, it is important for researchers to use the right simulator which offers flexibility in model construction and validation along with appropriate analysis of simulation output data, and statistical accuracy of the simulation results. More details about building credible simulation models can be found in (Law & Kelton, 2000; Pawlikowski, et al., 2002).

The simulation models were validated using one or more of the following methods: (1) we ran the same simulation experiment several times and found that ns-2 produced repeatable results ensuring the credibility of the final results of our simulation studies; (2) results obtained from ns-2 are compared and validated with empirical measurements using wireless laptops and access points for WLANs; (3) we compared our results with that of published results (both academic and industry white papers) by other researchers and found a good match ensuring the credibility of our simulation studies; and (4) discussion with experienced colleagues and researchers in the areas of network simulation.

Output Data Collection and Processing

When a simulation experiment is finished, ns-2 produces one or more output data files (e.g., trace file "out.tr" and network animator "out.nam") that contained detailed simulation output data. The trace file (out.tr) records each packet as it arrives at a node, departs a node, or is dropped at a link or queue. The trace file is useful for debugging or verification of the simulation program, but it is very difficult to obtain a specific performance metric directly from this file. We have written a perl script for output data processing and to obtain the desired performance metrics, such as delay, throughput, fairness and packet drop ratio, from a trace file.

An example of a trace file containing simulation output data for five trace entries is shown in Table 2. The first column shows the type of operation performed on the packet: enqueue (represented by +), dequeue (-), receive (r) and drop (d). The second column shows the time at which the event occurs. The source and destination nodes are shown in columns 3 and 4, respectively. The packet type, IP packet size, flags, and IP flow identifier are shown in columns 5 to 8, respectively. The source and destination IP addresses (in the form of "node.port") are shown in columns 9 and 10, respectively. The packet sequence number and unique packet identifier are shown in columns 11 and 12, respectively.

The out.nam can be used as an input to a network animator (NAM) tool for graphical simulation display. More details about NAM can be found at http://www.isi.edu/nsnam/nam/.

Simulation Execution

After a network model is constructed and various simulation parameters and performance measures are set, the next step is to execute a simulation. To achieve this in ns-2, an object of class Simulator needs to be created first as follows: set ns [new Simulator]. The simulation can then be executed by the following command: $ns run.

CASE STUDY

We have been using ns-2 for several years and our experiences are generally favorable. The ns-2 provides an excellent environment for easy simulation model development and performance evaluation of wireless communication networks. Figure 3 shows a simple framework in which we develop and execute various simulation models under ns-2 to study the performance of the IEEE 802.11 WLAN. In addition to modeling wireless network protocols, ns-2 also supports various propagation modeling.

Our current research focuses on developing a framework for estimating as well as improving the capacity of WLANs by integrating wireless network protocols and propagation modeling (Sarkar, 2004). We believe that our work contributes substantial extensions to ns-2 and provides insights into the simulator.

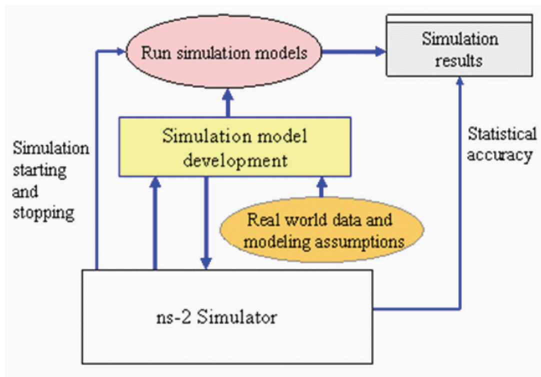
One of the main issues in network simulation is the statistical accuracy of simulation results. A model must be validated and used in a 'valid experiment,' which requires suitable sources of 'randomness' as well as appropriate means of analyzing simulation output data.

Fortunately, ns-2 simulator takes care of simulation output-data analysis and statistical accuracy of simulation results. Therefore, researchers can focus on developing and validating simulation models for various performance measures without worrying about controlling the simulation itself e.g., length of simulation runs (to get steady-state

Table 2. An example of a trace output showing first five trace entries

+	1.94375	0	2	tcp	1000	-------	0	0.0	3.1	235	610
-	1.94375	0	2	tcp	1000	-------	0	0.0	3.1	235	610
r	1.94471	2	1	tcp	1000	-------	1	3.0	1.0	195	600
r	1.94566	2	0	ack	40	-------	2	3.2	0.1	84	602
d	1.94609	2	3	tcp	1000	-------	0	0.0	3.1	235	610

Figure 3. A framework for developing and executing simulation models under ns-2



analysis), number of independent simulation runs, and use of appropriate random number generators.

Simulation Scenarios

In this section we present two simulation scenarios based on the ns-2, namely, IEEE 802.11 ad hoc and infrastructure networks. Figure 4 illustrates the basic concept of a simulated ad hoc wireless network with 50 mobile stations. These stations communicate without any infrastructure or centralized control.

Figure 4 shows the simulated infrastructure-based wireless LAN with 50 mobile stations and one wireless access point (AP) linked to wired backbone with 50 fixed stations. In the infrastructure network, data traffic travels from mobile stations to wired stations via the AP. Both Figures 4 and 5 are captured from animation output using ns-2's Network Animator (NAM) utility (ISI, 2010).

Figure 4. Simulation scenario of the IEEE 802.11 ad hoc network with 50 mobile stations (176x176 m grid)

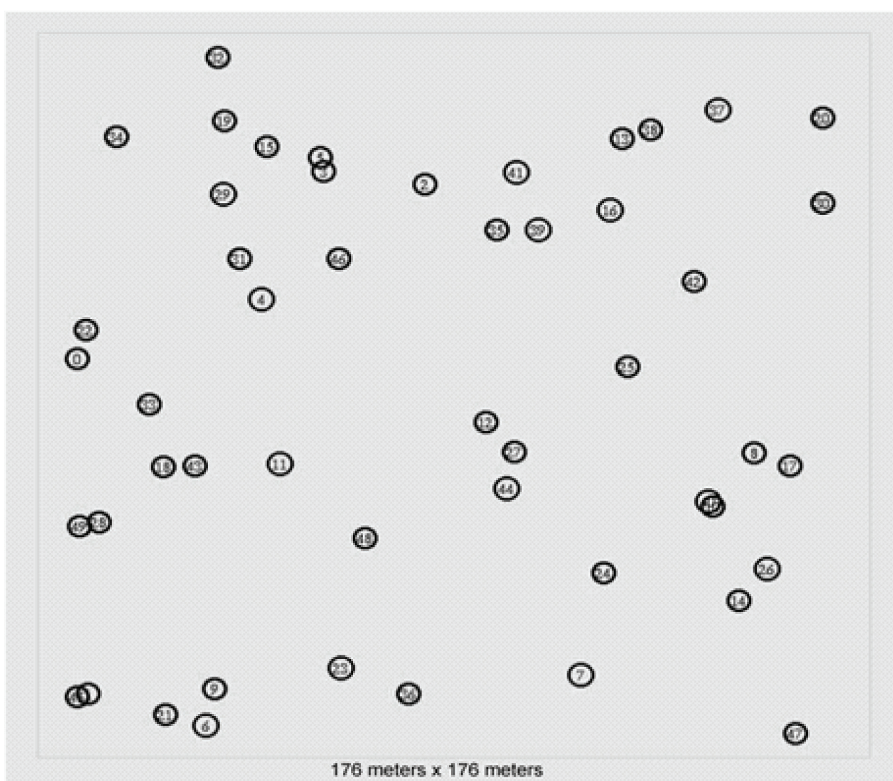
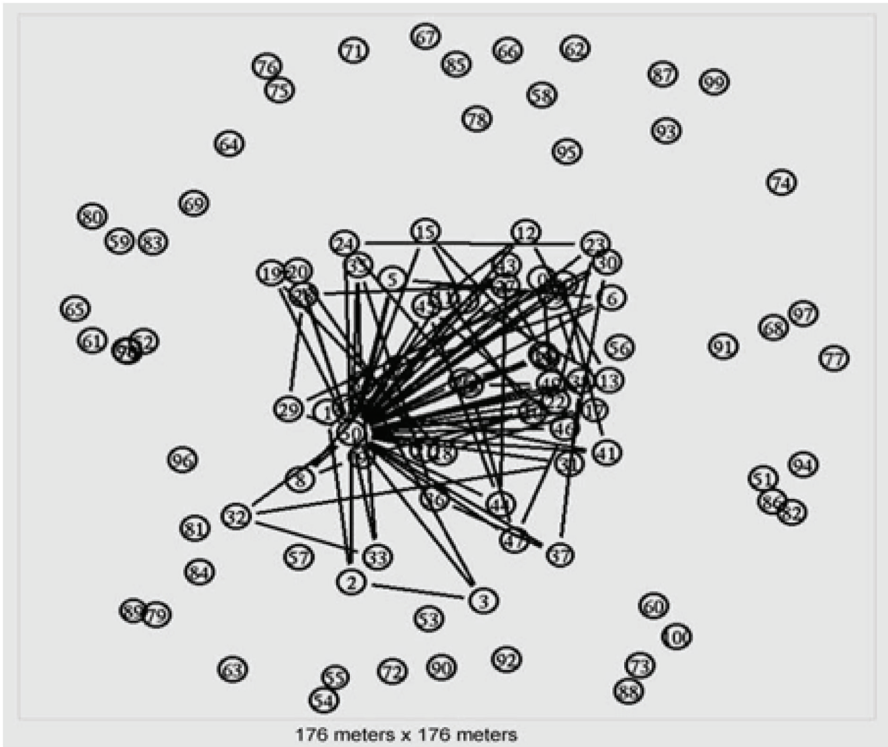


Figure 5. Simulation scenario of IEEE 802.11 infrastructure network with 50 mobile stations, one AP and 50 fixed stations (176x176 m grid)



Both the ad hoc and the infrastructure network are based on IEEE 802.11 with a maximum data rate of 11 Mbps. Mobile stations are simulated by setting up a grid of size 176m x 176m in which the longest distance between any two stations is 250m. This grid size is also the maximum transmission range of two simulated stations.

Table 3 lists the parameter values that we used in the simulation of the IEEE 802.11. Each simulation run lasted for 50 seconds simulated time in which the first 10 seconds is the transient period. The observations collected during transient period are not included in the final simulation results.

Simulation Results

We consider two important network performance metrics, namely, mean packet delay and throughput performance, for both individual stations and overall network. The mean packet delay at station i (i = 1, 2,..., N) is defined as the average time (measured in seconds or time slots) from the moment the packet is generated until the packet is fully dispatched from that station. A packet arriving at station i experiences several components of delay including queuing delay, channel access delay (i.e., contention time) and packet transmission time. The throughput (measured in bits per second) is defined as the fraction of the total channel capacity that is used for data transmission. We extracted both throughput and mean packet delay from the simulation output trace file generated by ns-2 simulator.

The simulation results were obtained from simulation runs for IEEE 802.11b infrastructure network. We consider Poisson packet arrivals and a packet length of 1,500 bytes. The simulation results report the steady-state behavior of the network.

Table 3. Simulation parameters

Parameter	Value
Data rate	11 Mbps
Basic rate	2 Mbps
Wireless cards	802.11b
Slot duration	20 µs
SIFS	10 µs
DIFS	50 µs
MAC header	30 bytes
CRC	4 bytes
PHY header	96 µs
Packet/Traffic type	UDP
Application	CBR
RTS-CTS	Off
PHY modulation	DSSS
Propagation model	Two ray ground
CWmin	31
CWmax	1023
Simulation time	50 minutes

The effect of increasing the number of active users on network throughput and mean packet delay performance of the IEEE 802.11b infrastructure network under various offered loads are shown in Figure 6 and 7, respectively. These results were validated through radio propagation measurements from wireless laptops and access points for 802.11b (Sarkar & Sowerby, 2006; Siringoringo & Sarkar, 2009). A good match between ns-2 simulation and real measurement results for N = 2 to 4 stations validates the simulation model.

Figure 6. Offered load versus network throughput of IEEE 802.11b infrastructure network

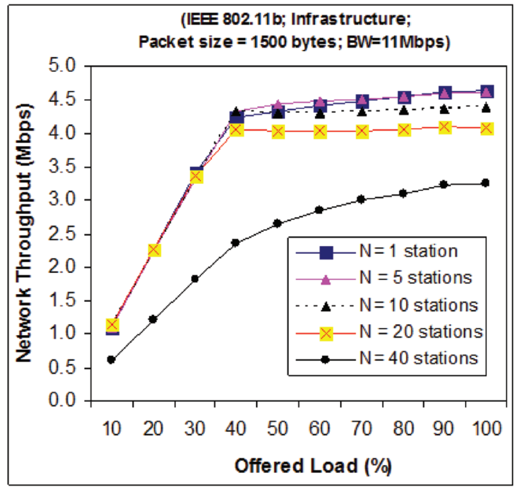
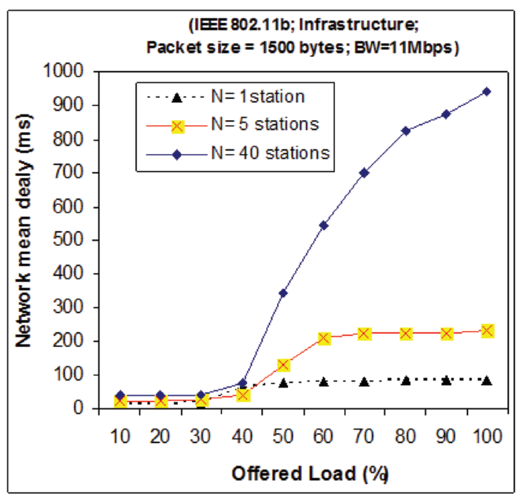


Figure 7. Offered load versus packet delay of IEEE 802.11b infrastructure network



STRENGTHS AND WEAKNESSES OF NS-2

Although the use of ns-2 as a network modeling and simulation tool has many advantages over other competing options, the simulation package has some limitations. The relative strengths and weaknesses of ns-2 are discussed next.

Strengths

Ns-2 is available for download on a variety of operating systems at no cost, including Red Hat Linux, FreeBSD, and MS Windows. Another strength of ns-2 is that many network researchers are contributing towards further extension of ns-2 since it is an open source software package (ISI, 2011). Authors of research papers often publish ns-2 code that they used, allowing other researchers to build upon their work using the original code. This is particularly useful to academia, specifically Master's and Doctoral students who are looking for a tool for network modeling and performance comparison.

A number of recent contributions enhancing the ns-2 are reported in network simulation literature. For example, Kurkowski et al. (2005) developed a new tool called iNSpect for visualization and analysis of wireless networks simulation under ns-2. This tool can be used to animate both a single stand-alone mobility file (ns-2 input file) and a trace file (ns-2 output file). iNSpect also provides various additional features than the ns-2's NAM, including analysis and validation of mobility models, debugging ns-2 scripts, and optimization of network protocol performance.

Mahrenholz and Ivanov (2004) report on a real-time network emulator that can be used with ns-2 for simulating live networks. Other enhancements of ns-2, including GPRS network simulation (Qiu, Zhang, & J., 2004), ant mobility platform for network simulator (Liao, Ting, Yen, & Yang, 2004), on-demand multipath routing protocol for a mobile ad hoc network (Sakurai & Katto, 2004), and ns-2 based simulation environment for Network-On-Chip (NOC) traffic analysis (Hegedus, Maggio, & Kocarev, 2005).

Ns-2 has a rich library of network and protocol objects, including nodes, links, and queues. In ns-2, there are two class hierarchies, namely the compiled C++ hierarchy and interpreted OTcl, with one to one correspondence between them. The compiled C++ hierarchy provides a greater efficiency in simulation runs in terms of faster simulation execution, which is particularly useful for detailed analysis of network protocol's behavior.

Ns-2 is a multi-protocol simulator that supports a wide range of network protocols, including unicast and multicast routing algorithms, network and transport layer protocols, TCP congestion control, router queuing policies, reservation and integrated services, and application layer protocols such as HTTP (Fall & Varadhan, 2011). In addition, ns-2 incorporates a range of link-layer protocols and scheduling algorithms, including wireless and mobile networks (Breslau, et al., 2000). Ns-2's in-built Network Animation (NAM) tool allows graphical display of links, packet flows, network node position and movement which is very useful for debugging and testing network models.

While various network simulators exist for simulation and modeling of link-layer protocols, very few of them can support a realistic physical layer modeling. Ns-2, however, supports three radio propagation models: free-space, two-ray ground, and shadowing, for wireless channel modeling (Dricot & De Doncker, 2004; Fall & Varadhan, 2011).

The influence of traffic load distribution on network performance is an important observation that interests many network researchers. For this task, a variety of traffic generators is needed for automatic traffic creation according to a desired pattern and load. The ns-2 supports several traffic generators including Poisson, Pareto, Exponential, CBR, and Hyperexponential.

In stochastic discrete-event simulation, the Pseudorandom Number Generators (PRNGs) are used in generating various random variables, including traffic generators, random movement of wireless nodes, and link error models. Therefore, it is important for network researchers to select a network simulator that has a good PRNG. However, Ns-2 (versions 2.1b9 and later) implements a combined Multiple Recursive Generator (MRG32k3a) proposed by L'Ecuyer (2001), which is one of the well established generators that has been tested thoroughly for its robustness. This generator replaces the previous implementation of PRNG, which was a rather weak generator based on the minimal standard multiplicative linear congruential generator of Park and Miller (Park & Miller, 1988). More details about PRNGs and their strengths and weaknesses can be found in computer simulation and modelling literature (Hechenleitner & Entacher, 2002; L'Ecuyer, 2001; L'Ecuyer, Richard, Chen, & Kelton, 2001; Law & Kelton, 2000; Matsumoto & Nishimura, 1999; Park & Miller, 1988; Pawlikowski, et al., 2002).

The implementation of MRG32k3a in ns-2 provides 1.8x1019 independent streams of random numbers, each of which consists of 2.3x1015 sub-streams, and each sub-stream has a period (i.e., the number of random numbers before overlap) of 7.6x1022 with a total period for the entire generator is 3.1x1057. With these features of PRNG, ns-2 can produce acceptable (un-biased) simulation results, and consequently users of ns-2 do not need to worry about the credibility of their simulation results as long as they use valid simulation models. The source code for ns-2 PRNG can be found in the ns-2 package under tools/rng.h and tools/rng.cc (Fall & Varadhan, 2011).

Weaknesses

In spite of possessing strengths, ns-2 has several limitations. Firstly, it does not provide any support for creating sophisticated graphical presentations of simulation output data. The raw data must be processed using scripting languages such as 'awk' or 'perl' to produce data in a suitable format for tools like Xgraph or Gnuplot (Fall & Varadhan, 2011; Sarkar & McHaney, 2006).

Another disadvantage of ns-2 is that it is not a user-friendly package because of its text-based interface, and many student researchers point out that ns-2 has a steep learning curve. A tutorial contributed by Marc Greis (2011) and the continuing evolution of ns documentation have improved the situation, but ns-2's split-programming model remains a barrier to many developers.

Although ns-2 supports various radio propagation and error models, the software neither supports Bit-Error Rate (BER) nor Signal-to-Noise-and-Interference Ratio (SNIR), which are important parameters that need to be considered when simulating real-world wireless network scenarios. In addition, ns-2 does not support simulation of co-channel interference, which is one of the major limitations of ns-2's wireless network simulation engine. These limitations of ns-2 are also highlighted by leading network researchers (Hegedus, et al., 2005; Wellens, Petrova, Riihijarvi, & Mahonen, 2005).

In summary, it is quite difficult to implement a new channel access protocol in ns-2. A sound knowledge and understanding of both the high-level programming and operating systems is required in implementing a new protocol under ns-2. We experienced similar difficulty in implementing the BUMA protocol in ns-2 (Sarkar & Sowerby, 2005).

CONCLUSION AND FUTURE WORK

Stochastic discrete event simulation methodology has become popular as a network modeling and performance analysis tool. In this chapter we described the use of a network simulator, ns-2 and wireless networking issues addressed by simulation. The models built under ns-2 simulator were validated using empirical measurements from wireless laptops and access points for an IEEE 802.11b WLAN. A good match between ns-2 simulation results and empirical measurements were reported. The chapter provides a general discussion on techniques for building valid and credible models, and on the strengths and weaknesses of simulation methodology. Although our case study focused on the use of ns-2 for modeling and performance evaluation of IEEE 802.11 wireless LANs, this case study can be used in other fields, such as management, and engineering.

In summary, we want to stress the importance of using a good simulator for modeling and performance analysis of wireless communication networks. The ns-2 offers more flexibility in model construction and validation, and incorporates appropriate analysis of simulation output data, pseudo-random number generators, and statistical accuracy of the simulation results. Without these features, a simulation model would be useless since it will produce invalid results. As Kleinjen (1979) pointed out that "…*instead of an expensive simulation model, a toss of the coin had better be used*."

There are several interesting research problems in the emerging area of network simulation and modeling. Some of these research issues include use and misuse of simulation according to a survey of recent IEEE publications. We are currently addressing some of these research problems, and research results will be presented in future articles.

ACKNOWLEDGMENT

The first author was supported in part by the AUT University's School of Computing and Mathematical Sciences Contestable Research Grant. An earlier version of the chapter appears as: Sarkar, N. I, & McHaney, R. Modeling and Simulation of IEEE 802.11 WLANs: A Case Study of a Network Simulator, *Proceedings of the 2006 Information Resources Management Association International Conference, May 21-24, Washington DC, USA.*

REFERENCES

Bianchi, G. (2000). Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, *18*(3), 535–547. doi:10.1109/49.840210

Breslau, L., Estrin, D., Fall, K., Floyd, S., Heimann, J., & Helmy, A. (2000). Advances in network simulation. *Computer*, *33*(5), 59–67. doi:10.1109/2.841785

Carson, J. S., II. (2004). Introduction to modeling and simulation. In *Proceedings of 2004 Winter Simulation Conference*, (pp. 1283-1289). IEEE Press.

Chen, W.-T., Jian, B.-B., & Lo, S.-C. (2002). An adaptive retransmission scheme with QoS support for the IEEE802.11 MAC enhancement. In *Proceedings of the 55th IEEE Vehicular Technology Conference*, (pp. 70-74). IEEE Press.

CMU Monarch Project. (2011). *Webpage*. Retrieved January 5, 2011 from http://www.monarch.cs.cmu.edu.

Das, S. R., Castaneda, R., & Yan, J. (2000). Simulation-based performance evaluation of routing protocols for mobile ad hoc networks. *Mobile Networks and Applications*, *5*(1), 179–189. doi:10.1023/A:1019108612308

Dricot, J.-M., & De Doncker, P. (2004). High-accuracy physical layer model for wireless network simulations in ns-2. In *Proceedings of 2004 International Workshop on Wireless Ad-Hoc Networks*, (pp. 249-253). IEEE Press.

Fall, K., & Varadhan, K. (2011). The ns manual. *The VINT Project*. Retrieved January 5, 2011 from http://www.isi.edu/nsnam/ns/doc/.

Fantacci, R., Pecorella, T., & Habib, I. (2004). Proposal and performance evaluation of an efficient multiple-access protocol for LEO satellite packet networks. *IEEE Journal on Selected Areas in Communications*, *22*(3), 538–545. doi:10.1109/JSAC.2004.823437

Greis, M. (2011). *Ns-2 tutorial*. Retrieved January 5, 2011 from http://www.isi.edu/nsnam/ns/tutorial/index.html.

Hassan, M., & Jain, R. (Eds.). (2004). *High performance TCP/IP networking - Concepts, issues, and solutions*. New York: Pearson Prentice Hall.

Hechenleitner, B., & Entacher, K. (2002). On shortcomings of the ns-2 random number generator. In Znati, T., & McDonald, B. (Eds.), *Communication Networks and Distributed Systems Modeling and Simulation* (pp. 71–77). New York: The Society for Modeling and Simulation International.

Hegedus, A., Maggio, G. M., & Kocarev, L. (2005). A ns-2 simulator utilizing chaotic maps for network-on-chip traffic analysis. In *Proceedings of the IEEE International Conference on Circuits and Systems (ISCAS 2005)*, (pp. 3375-3378). IEEE Press.

Holloway, D. (2003). *Wireless networking: A guide to wireless networking and deployment*. Retrieved September 25, 2003 from http://www.hill.com/archive/pub/papers/2003/01/paper.pdf.

ISI. (2010). *Network animator.* Retrieved June 5, 2010 from http://www.isi.edu/nsnam/nam/.

ISI. (2011). *ns-2 contributed code*. Retrieved January 5, 2011 from www.isi.edu/nsnam/ns/ns-contributed.html.

Kleijnen, J. P. C. (1979). The role of statistical methodology in simulation. In Zeigler, B. P. (Ed.), *Methodology in Systems Modelling and Simulation. North Holland*. The Netherlands: Academic Press. doi:10.1145/1102786.1102793

Kurkowski, S., Camp, T., Mushell, N., & Colagrosso, M. (2005). A visualization and analysis tool for ns-2 wireless simulations: iNSpect. In *Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, (pp. 503-506). IEEE Press.

L'Ecuyer, P. (2001). Software for uniform random number generation: Distinguishing the good and the bad. In *Proceedings of the 2001 Winter Simulation Conference*, (pp. 95-105). Arlington, VA: IEEE Press.

L'Ecuyer, P., Richard, S., Chen, E. J., & Kelton, W. D. (2001). An object-oriented random number package with many long streams and substreams. *Operations Research*, *1*(1).

Law, A. M., & Kelton, W. D. (2000). *Simulation modelling and analysis* (3rd ed.). New York: McGraw-Hill.

Liao, H.-C., Ting, Y.-W., Yen, S.-H., & Yang, C.-C. (2004). Ant mobility model platform for network simulator. In *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2004)*, (pp. 380-384). IEEE Press.

Mahrenholz, D., & Ivanov, S. (2004). Real-time network emulation with ns-2. In *Proceedings of the Eighth IEEE International Symposium on Distributed Simulation and Real-time Applications (DS-RT 2004)*, (pp. 29-36). IEEE Press.

Matsumoto, M., & Nishimura, T. (1999). Mersenne twister: A 623-dimentionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulation*, *7*(1), 3–30.

McHaney, R. (1991). *Computer simulation: A practical perspective*. San Diego, CA: Academic Press.

McHaney, R. (2009). *Understanding computer simulation*. Frederiksberg, Denmark: Ventus Publishing APS. Retrieved December 20, 2010 from http://bookboon.com/us/textbooks/it/understanding.

Nicopolitidis, P., Obaidat, M. S., Papadimitriou, G. I., & Pomportsis, A. S. (Eds.). (2003). *Wireless networks*. New York: John Wiley & Sons, Ltd.

Park, S. K., & Miller, R. W. (1988). Random number generators: Good ones are hard to find. *Communications of the ACM*, *31*(10), 1192–1201. doi:10.1145/63039.63042

Pawlikowski, K. (1990). Steady-state simulation of queuing processes: A survey of problems and solutions. *ACM Computing Surveys*, *1*(2), 123–170. doi:10.1145/78919.78921

Pawlikowski, K., Jeong, H.-D. J., & Lee, J.-S. R. (2002). On credibility of simulation studies of telecommunication networks. *IEEE Communications Magazine*, *40*(1), 132–139. doi:10.1109/35.978060

Qiu, Q., Zhang, D., & J., M. (2004). GPRS network simulation in ns-2. In *Proceedings of the 10th Asia-Pacific Conference on Communications and 5th International Symposium on Multi-Dimentional Mobile Communications*, (pp. 700-704). APCC Press.

Sakurai, Y., & Katto, J. (2004). AODV multipath extension using source route lists with optimized route establishment. In *Proceedings of the 2004 International Workshop on Wireless Ad-Hoc Networks*, (pp. 63-67). IEEE Press.

Sargent, R. G. (2004). Validation and verification of simulation models. In *Proceedings of the 2004 Winter Simulation Conference*, (pp. 17-28). IEEE Press.

Sarkar, N., & Pawlikowski, K. (2002). A delay-throughput performance improvement to the pi-persistent protocol. *Pakistan Journal of Applied Sciences*, *2*(3), 390–399.

Sarkar, N. I. (2010). *Capacity estimation of wireless LANs*. Retrieved March 15, 2010 from http://elena.aut.ac.nz/homepages/staff/Nurul-Sarkar/capacity/.

Sarkar, N. I., & McHaney, R. (2006). Modeling and simulation of IEEE 802.11 WLANs: A case study of a network simulator. In *Proceedings of the 2006 Information Resources Management Association International Conference*, (pp. 715-718). Washington, DC: IEEE Press.

Sarkar, N. I., & Sowerby, K. W. (2005). Buffer unit multiple access (BUMA) protocol: an enhancement to IEEE 802.11b DCF. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM 2005)*, (pp. 2584-2588). St. Louis, MO: IEEE Press.

Sarkar, N. I., & Sowerby, K. W. (2006). Wi-Fi performance measurements in the crowded office environment: A case study. In *Proceedings of the 10th IEEE International Conference on Communication Technology (ICCT 2006)*, (pp. 37-40). Guilin, China: IEEE Press.

Schmeiser, B. (2004). Simulation output analysis: A tutorial based on one research thread. In *Proceedings of the 2004 Winter Simulation Conference*, (pp. 162-170). IEEE Press.

Siringoringo, W., & Sarkar, N. I. (2009). Teaching and learning Wi-Fi networking fundamentals using limited resources. In Gutierrez, J. (Ed.), *Selected Readings on Telecommunications and Networking* (pp. 22–40). Hershey, PA: IGI Global. doi:10.4018/9781605660943.ch003

Tickoo, O., & Sikdar, B. (2003). On the impact of IEEE 802.11 MAC on traffic characteristics. *IEEE Journal on Selected Areas in Communications*, *21*(2), 189–203. doi:10.1109/JSAC.2002.807346

Wellens, M., Petrova, M., Riihijarvi, J., & Mahonen, P. (2005). Building a better wireless mousetrap: Need for more realism in simulations. In *Proceedings of the Second Annual Conference on Wireless On-Demand Network Systems and Services (WONS 2005)*, (pp. 150-157). IEEE Press.

KEY TERMS AND DEFINITIONS

Access Point (AP): A device which acts as a bridge between wireless LANs and a wired backbone network. An AP coordinates communication among the mobile stations.

Ad-Hoc Network: A class of wireless LAN where nodes communicate without wireless access points. A wireless network operating in ad-hoc mode is also called an Independent Basic Service Set (IBSS).

IEEE 802.11: A family of wireless LAN standards.

Infrastructure Network: A class of wireless network in which mobile stations are connected to the wired backbone network through wireless access points.

Ns-2: An open source discrete event network simulation package.

OPNET: A commercial network simulation package which is available to many educational institutions under OPNET academic program worldwide.

Simulator: A software package used for modeling and simulation tasks.