Realistic Support For IEEE802.11b MAC in NS

CS499 Project Report

Ashwini Kumar(Y0084) and Sabyasachi Roy(Y0288)

Supervisor : Dr. Bhaskaran Raman

Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur, India 208016
{ashw,sroy}@cse.iitk.ac.in

Abstract

The emergence of IEEE802.11b, as a high data-rate MAC standard for wireless LANs, has led to widespread deployment of WLANs and has also influenced areas of both research and industry. A significant impact of the emergence of this protocol was the opening up of research in this area. Network Simulator (NS) is a discrete event simulator targeted at networking research, and has emerged over the years as the most popular research tool in networks. IEEE802.11b MAC has been implemented in NS; however the implementation is not quite realistic or complete. In this project, we improve upon the implementation of IEEE802.11b in NS, and also add additional novel features to improve its modeling capabilities. We call the improved Network Simulator as *The Enhanced Network Simulator (TeNs)*.

Keywords : IEEE802.11b, Network Simulator (NS), Gray Region, Binary Phase Shift Keying (BPSK) Modulation, Direct Sequence Spread Spectrum (DSSS), Adaptive Data-rate Transitions, Signal-to-Noise Ratio (SNR), Bit Error Rate (BER), Multiple Interfaces, Multihop, Static Routing, Antenna Gain, Propagation Models.

1 Introduction

The area of wireless communication is witnessing a lot of research and development, after the standardization in the form of IEEE802.11 MAC (Media Access Control – its a sublayer of Data Link Layer in the TCP/IP protocol stack) protocol for wireless Local Area Networks (LANs) [1], in 1997. This protocol provides specifications for the physical layer (PHY) as well, which can support and provide services for the requirements of the MAC sublayer. In particular the IEEE802.11 Standard provides specifications for three kinds of physical media :

1. Frequency-Hopping Spread Spectrum (FHSS) PHY for 2.4 GHz ISM (Industrial, Scientific, Medical) band. This frequency band is unlicensed in most countries.
2. Direct Sequence Spread Spectrum (DSSS) PHY for 2.4 GHz ISM band.
3. Infrared (IR) PHY.

All of these physical layer mediums supported data rates of 1 and 2 Mbps(Mega-bits per second). IEEE further improved upon the standard, and introduced IEEE802.11b protocol [2], which supports higher data-rates (5.5Mbps and 11Mbps) in the Direct Sequence Spread Spectrum PHY. The key idea behind this was use of a new modulation technique called Complementary Code Keying (CCK) for higher data-rates. This improvement and standardization has led to the rapid development and widespread deployment of wireless

devices, which are inter-operable among different vendors. This has also influenced the networking research community. Researchers are experimenting with the protocol, trying to improve it, or studying its feasibility for usage in many scenarios. Apart from standardization, another important reason for this is that overall deployment cost (including equipments), is comparatively much lesser than other wireless communication technologies, together with the advantage of easy availability. An example of experimentation and research with this protocol, is using it for long range multi hop wireless networks, rather than only for short range infrastructure networks, which are being studied in the *Digital Gangetic Plains (DGP) Project* [3], at IIT Kanpur. This is an attempt to provide rural connectivity at a low cost to the villages in India.

A reliable simulation tool is a necessary requirement to gauge the feasibility and effectiveness of any new practical research idea. Its here that Network Simulator (NS) [4] comes into the picture. It has become the standard tool for simulation studies in computer networks research community. It was developed by UCB (University of California, Berkeley, USA), in collaboration with researchers from USC (University of Southern California, LA, USA), Xerox PARC, etc. It has support for protocols at various layers, which include FTP, versions of TCP, routing and also wireless (satellite) networks as well. It is an open source and extensible tool, and functionalities are being continuously added to it by the researchers to extend its simulation capabilities, or testing out new ideas. Recently, support for simulation of IEEE802.11b has been added to it. However, the existing implementation is much simplified and isn't realistic. Our aim in this project was to refine this MAC layer protocol implementation of NS (specifically, version 2.1b9a of NS) as well as provide enhanced modeling at the PHY layer. Apart from refining the existing implementation of 802.11b MAC layer, we have added features likes multiple interfaces and multi hop routing to NS, which are necessary for full-scale simulation of a wireless mesh network like the DGP [5].

The new and refined version of NS with the addition of various features is called as **The Enhanced Network Simulator - TeNs**, and we have released TeNS for public usage (see [6]). All the modifications in original NS, that are explained and referred to in the following sections, are now present in TeNs.

The organization of this report is as follows. After the section on motivation of our work, we give a detailed description of the problem statement. We then describe concisely the project progress, status and work done. Further, we discuss in detail the various experiments, simulations, and results that we have obtained. In this section we also describe the changes made in NS and also additional features added to it. In the next section, we briefly describe some typical examples to show where TeNs can be used, followed by a section on future work. The appendix gives some additional information regarding the changes to code in NS, and a brief comparison of TeNs with other popular simulation tools. We also include a brief tutorial on how to use TeNs.

2 Motivation

There are many benefits to a simulation tool which closely models the real world. The most obvious one, being the correctness and reliability of the results observed on running the simulations. NS is often used to simulate scenarios for wireless LANs in networking research, and to get realistic results, the point mentioned before is very important. Another important reason for improving upon the current implementation, is that, we would like to provide a reliable simulation tool to study the behavior of different protocols over the DGP mesh [3].

3 Problem Statement

NS, along with support for various networking protocols, also incorporates the IEEE802.11b MAC layer protocol, the MAC layer standard used in WLANs (Wireless Local Area Networks). However, the current implementation of the MAC and the PHY layers are very simplified compared to the actual standard. There are many features and functionalities, that are either not implemented, or not adequately modeled in the NS implementation. Some of those absent features are as follows.

- **Adjacent and co-channel interference.** The frequency range specified for use by IEEE802.11 standard, is 2.4000 Ghz to 2.4835 Ghz (with minor variations among different countries), and is divided into 11 overlapping channels. This invariably leads to adjacent and co-channel interference, and this is not modeled in NS.
- **Directional antennae.** NS has support only for omni-directional antenna, and with the use of directional antenna in real life, there is an increasing need for directional-antenna support in NS.
- **Gray region and temporal variation.** Gray Region refers to a region, which is far enough from the signal source, so that the signal strength received is near to, but above the *receive threshold* of the receiver. It is observed that the signal strength (which determines the throughput) doesn't go to zero abruptly, rather it shows a steady decline accompanied by minor fluctuations in that region. Wireless communication also shows the phenomena of *random temporal variations*. These refer to the random variations in signal strength (and throughput) at a point, which is observed in real life WLANs. This is natural to expect, because of wireless mode of communication, which is subject to interferences like multipath, diffraction, etc. Gray Region has been found not to be modeled adequately in NS (see section 5).
- **Adaptive data-rate transitions (through 11, 5.5, 2, 1 Mbps)**. Adaptive data-rate transitions, is a mechanism by which a signal sender (like an access point), can switch between standard data-rates (11, 5.5, 2, 1 Mbps) to reduce bit error, and hence improve performance. This feature is currently not present in NS.
- **Multiple interfaces in mobile nodes**. The current NS implementation allows only one network interface per node. In fact, the node entity and the interface are tightly coupled in the existing implementation. This feature is also required as IEEE802.11 is being tested for use in long range networks [3].
- **Multihop support in infrastructure WLAN**. Multihop support is also required for wireless nodes if networks like DGP [5] are to be simulated in NS. This requires forwarding of packets from a node to another in a mesh kind of network. Note that current ad-hoc routing protocols incorporate this kind of forwarding, but it involves lots of overhead, which is not what is encountered in a static wireless multi hop mesh network like DGP.
- **Improper placement of the LLC[1] implementation**. Currently, it is tightly coupled with the MAC layer, rather than being a separate logical entity. This implies that support for different link layer protocols can't be added.

Some of the features mentioned above, are not directly a part of MAC layer (like multi hop support), but their requirements were felt in correctly modeling scenarios like the DGP mesh.

An important initiative, in providing the missing features discussed above, was taken in the Bachelor's Project by Ankur Khandelwal last year (see [7]), and support for two of the above deficiencies were provided – channel interference, and directional antennae.

Our main aims in this project are the following.

1. Implementation and validation of gray region, and adaptive data-rate transitions.
2. Validation of the channel interference implementation, and improvement in its modeling, if required.
3. Verifying the current directional antennae implementation, and to augment the implementation to account for *back and side lobes*, as well as radiation pattern for common directional antenna.
4. Implementation of multiple interfaces in mobile nodes.
5. Adding multi hop support for wireless simulations.
6. To place the LLC code at the proper place (currently, because of its interlinking with the MAC layer, the LLC may not work, if some Link Layer protocol other than "stop and wait" is used).

[1]Logical Link Control Layer.

4 Project Progress and Work Done

Table 1, given below gives concisely the status of work done and objectives achieved.

S. no.	Topic	Task	Status
1	Gray Region	Implementation+Validation	done
2	Random Temporal Variations	Validation	done
3	Multiple Interfaces	Implementation	done
4	Adaptive data-rate transitions	Implementation+Validation	done
5	Multihop Support	Implementation	done
6	Channel Interference	Validation	done
7	Directional Antennae	Implementation+Validation	done
8	LLC Code	Relocation	to be done

Table 1: Project - current status.

The work consisted of both implementation and actual experiments. NS is quite a big software, and before actually doing anything, the structure of NS and the enhancements made in a previous project [7] were studied. Also, the relevant features of IEEE802.11 protocol, were considered before adding any new code to the NS. The above mentioned study was a major part of the project in previous semester. Following gives a detailed description of work done.

1. A comprehensive study of NS (version 2.1b9a) implementation and its basic structure, together with the important features of IEEE802.11 protocol, was done.

2. The enhancements made in an earlier project [7] were studied, in the process of adding the features of channel interference, and directional antennae. Following are some details of the previous work. The channel interference implementation in this is quite simple. Depending upon the channel difference of the sending interface and the receiving interface, the power of the transmitted packet is decreased by some value. Greater the difference greater the reduction of power and thus greater the chance of the packet being dropped. Implementation of the directional antenna is that, once the packet is received, it is dropped or passed on to higher layers, depending on whether the straight line between the receiver and the sender lies within the cone formed by the antenna radiation patterns. This implementation is discrete, i.e., the packets are either dropped or correctly received.

3. Validation of existing *Gray Region* modeling in NS was performed. Its modeling in NS was found to be inadequate, and hence features like *Bit Error Rate (BER)* were added, to make it more realistic. BER gives the probability of a bit being received in error, and depends on received signal strength as well as the modulation. IEEE802.11b uses DSSS physical layer in 2.4 GHz band. The standard error function for BPSK modulation was used for determining the BER function to use (see section 5 for details).

4. Validation of *random temporal variations*, that is observed in real life scenarios (see section 5) was done. The modeling for this case was found to be adequate in NS, as it is incorporated in the propagation model (*Shadowing Model* to be precise).

5. Implementation of the adaptive data-rate transition feature was also done. Though, this feature is implemented by almost all vendors, the protocol standard [2] doesn't specify the actual numbers. We modeled this feature in NS according to the data given by the Cisco data-sheets [8].

6. Implementation of multiple interfaces support on a single mobile node was completed and verified. Original NS supports only a single interface mobile node. This restricted the scenarios which could be simulated in NS. The support for this functionality required changes in certain modules, especially the *add-interface* function of a mobile node.

7. Implementation of multi hop support, in wireless scenarios was also done. Multihop forwarding of packets is a necessary requirement for wireless mesh networks. To provide this support, a new wireless

routing protocol *WLStatic* was implemented. It does static routing of packets, and doesn't involve any route maintenance or other overheads as in ad-hoc routing protocols. Some results showing its implementation to be successful, are presented in section 5.

8. The directional antenna implementation was validated for correctness through various NS simulations (see point 2, for description of original implementation). In addition, we also improved further the directional antenna implementation by adding support for realistic antennas, complete with lobes. An example of antenna-pattern of such an antenna is shown in Fig. 11. In the implementation, the antenna radiation pattern file is input and propagation decisions are based on the directional gain values encoded in that file.

9. Validation of channel interference implementation was done. For this we performed an actual experiment consisting of two parallel 802.11b links running simultaneously. The channel of the links were varied to produce the interference in the communication. See section 5 for more details.

5 Experiments and Results

In this section we describe in detail the following *validation experiments/feature additions* together with results.

1. Gray Region Validation
2. Random Temporal Variations
3. Multiple Interfaces
4. Multihop in Wireless Nodes
5. Adjacent Channel and Co-channel Interference Validation
6. Directional Antennae Implementation

5.1 Gray Region Validation

We performed simple simulations and experiments to verify for the Gray Region implementation (see section 3 for description of the issue).

5.1.1 Setup

- **Simulation :** The simulation scenario consisted of 2 nodes, each having single interface and separated by some distance. A CBR[2] connection was started between the two, with one node as the sender and the other as receiver. Total simulation time was 30 seconds, with the CBR connection starting at 10 seconds. Hence, total run-time for the connection or traffic was 20 seconds. The size of each packet was kept at 1000 bytes, and the packets were sent every 0.0005 seconds. Thus the traffic was overloaded (the data rate capacity of the link was kept at its maximum – 11 Mbps). RTS/CTS[3] was kept on. The *Shadowing* propagation model was used, as is deemed to be the most realistic propagation model. Also, the transmission power level was kept at 5 mw. The throughput for the CBR traffic was then calculated from the trace file generated by NS, at different distances between the source and the sink nodes.

- **Experiment :** The experimental scenario was arranged in manner, that it is as close to simulation scenario described above. An Access Point (AP) was kept fixed at a location, that was also connected to the IITK LAN. A laptop was used as the second node, which connected to the AP. The distance of the laptop was varied from the AP, and the readings were taken. To generate traffic on the link, *Netperf*

[2]Constant Bit Rate

[3]RTS or "Request To Send", and CTS "Clear To Send" are mechanisms used for tackling the *Hidden Node Problem*. Please refer to [1] for more details.

software [9] was used. Its a client-server mechanism based software, used for network benchmarking. We ran the server, called *netserver*, on one of the PCs in the LAN. Then at various distances from the AP, the netperf client was run on the laptop, which connected to the netserver. UDP traffic was used, with packet sized kept at 1000 bytes, and the throughput reading was obtained for a period of 10 seconds of traffic. As in the simulation scenario above, RTS/CTS mechanism was kept on and the power was kept at 5 mw.

We used a Cisco AP, and PCMCIA[4] cards, and used its client utility to record SNR, etc, at various distances. Three similar experiments were performed, and we averaged over the readings obtained in these cases, to obtain a benchmark reading for comparison with the simulation results. Our objective was to see if the general trend of readings from simulations match those from the experiments, and to modify or augment NS if they didn't quite match. Note that, no attempt was made to tune NS , in order to make the experimental readings and simulation readings match exactly, which certainly would be an overkill in the probabilistic phenomenon of wireless communication.

5.1.2 Results and Discussion :

When we compared the experimental results with the simulation results obtained from *original* NS, the decay trends obtained were far apart (see Fig. 1). It can be noticed that throughput obtained from simulations didn't decay appreciably over large distances, as against the values obtained from actual experiments. Hence, the current modeling was found to be inadequate. NS already incorporated propagation models, but

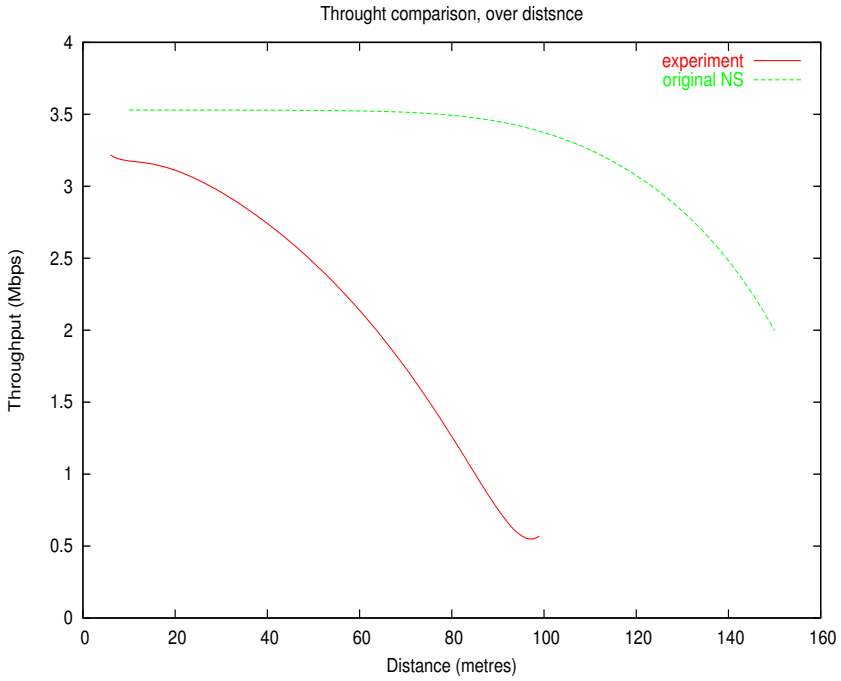


Figure 1: Trend comparison for throughput decay with distance. It can be seen that original NS implementation gives markedly different trends as compared to experimental results.

no model for *BER*, or the Bit Error Rate (probability of a bit being received in error) was implemented. Since IEEE802.11b uses DSSS[5] Physical Layer and BPSK[6] modulation, we implemented a BER scheme for the same, which is given by,

[4]Personal Computer Memory Card International Association. A standard laptop expansion cards. Also called PC cards
[5]Direct Sequence Spread Spectrum.
[6]Binary Phase Shift Keying

$P_e = erfc\left(\sqrt{\frac{E_b}{N_0}}\right)$, where,

$erfc$ is the standard complementary error function,
E_b is transmitted signal energy per bit,
N_0 is noise spectral density.

Basically E_b/N_0 is the SNR (Signal to Noise Ratio). Expanding the $erfc$ function, we get BER as given below [10].

$$P_e = \int_{\sqrt{SNR}}^{\infty} e^{-z^2}\, dz$$

Since, above is non-integrable, a table lookup mechanism has to be used. The code for implementing above was added[7]. So the basic mechanism is as follows. When a packet is received, its SNR is calculated and then the BER for the entire packet is deduced depending on the modulation scheme used (currently only BPSK is implemented). BER is not calculated for each and every bit of the packet, rather the entire packet is considered as a single unit (because all the bits are received with same SNR due to deterministic nature of NS). If BER is greater than a random number generated, then the packet is dropped. Currently, in order to keep the use of modulation optional for a simulation, a variable *modulationscheme_* has been defined which can be set to 1, in the simulation script by the user for invoking the BPSK modulation scheme implementation. Otherwise the default is not to use any modulation and BER.

In addition to BER, an implementation of *adaptive data-rate transitions* was also done. It was observed that most of the vendors implement this feature in their APs and PC cards. Adaptive data-rate transition is a mechanism to reduce the BER, at the expense of a high data-rate by changing the modulation scheme. However, our implementation currently doesn't change the modulation scheme, but simply reduces the data-rate. Though the standard ([2]) allows this mechanism, the actual numbers are not specified. We have used the numbers from Cisco data-sheets [8] in our implementation, which are reproduced in Table 2.

S. no.	Data Rate	Transition Boundary
1	1 Mbps	<-94 dbm
2	2 Mbps	-91 dbm
3	5.5 Mbps	-89 dbm
4	11 Mbps	>-85 dbm

Table 2: Data rate transition table for Cisco Aironet Clients and APs. For Orinoco cards, the thresholds differ slightly by 1 or 2 dbms. **Note :** *dbm* is a logarithmic measure of power, and 0 dbm equals 1 milliwatt.

Note, that the effect of data-rate transitions is observed only when SNR decreases to a very low value, which is often very close to receive sensitivity. In those extreme cases, the behavior is difficult to quantify. This is because from experiments we observed that the SNR varies erratically under those circumstances. As in the case of BER, adaptive rate transition is not invoked by default. The user can force its use by setting a variable *isAdaptive_* as 1, in the simulation script.

Excellent results were obtained from simulations after using the above mentioned additions to the existing implementation of the protocol (See Fig. 2). Slight tuning of calculated BER made the new curve approach closer to experimental results. The graph in Fig. 2 clearly shows that these new additions have improved the existing model.

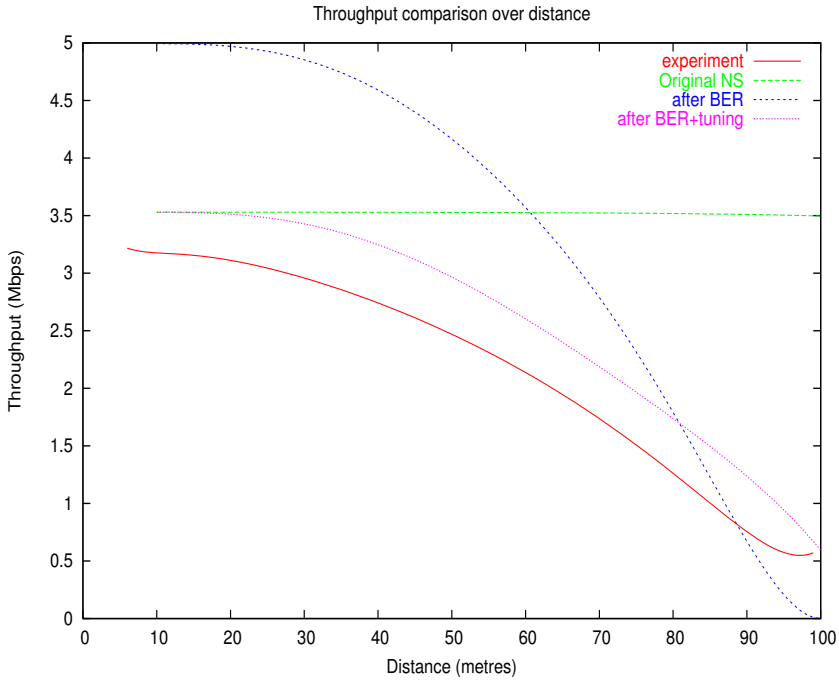


Figure 2: Trend comparison for throughput decay with distance. Incorporation of BER improved the trends for the simulation. Further tuning made the trends almost alike.

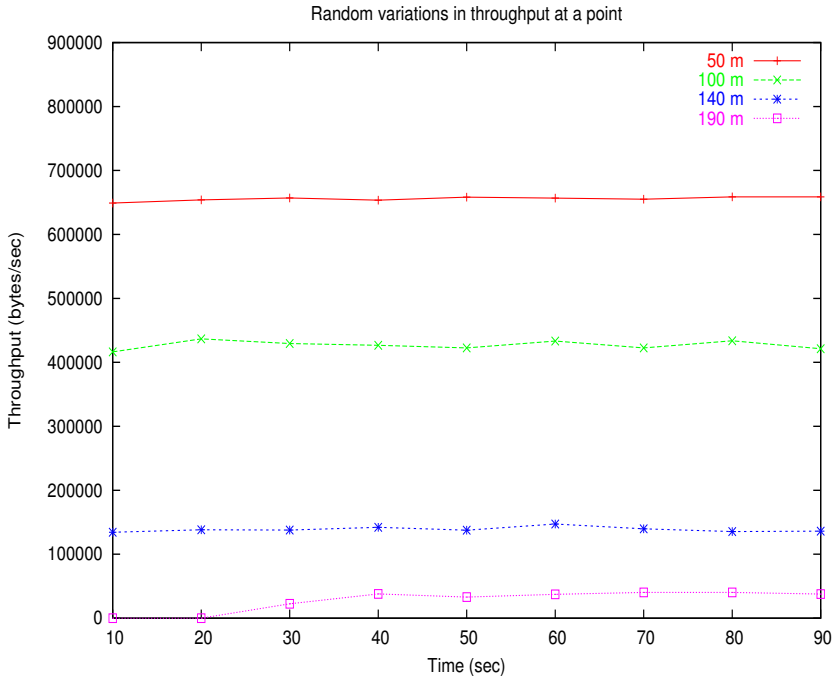


Figure 3: Graph showing random temporal variations in simulations. Each bold point represents the throughput observed over last 10 second interval. Variations of throughput for 5 different distances are plotted.

5.2 Random Temporal Variation

5.2.1 Setup

This issue was briefly described in section 3. On conducting our experiments (as described in subsection 5.1.1), it was noticed that there were minor, random fluctuations in signal strength and hence throughput at same point. These arise because of the non-deterministic nature of interference, due to various sources in real life. To check whether this phenomena is captured in NS in some way, we performed a single pair simulation, almost identical to the simulation scenario described in 5.1.1. However the duration for CBR connection was 100 seconds, and the distance between the two nodes was fixed.

5.2.2 Results and Discussion

For the setup described above, the plot of throughput obtained, every 10 seconds is given in Fig. 3. It can be seen that throughput randomly over the 100 second period, and the amount of variation is low. Hence, it can be concluded from these results that NS does adequately model random temporal variations. The reason for this variation, is the randomness incorporated in the propagation model (Shadowing model) implemented in NS.

5.3 Multiple Interfaces

As discussed briefly in section 3, currently NS supports only single interface mobile nodes. This is a big restriction on simulations, when a scenario where a wireless node supports multiple interfaces on different channels using different antennas is required. Example of such a scenario is the DGP mesh network [5]. Note that this feature has really nothing to do with MAC protocol. However, its was felt that such a feature is necessary for realistic simulation of scenarios like the one mentioned above, apart from being a significant value addition to NS.

To add this feature, modifications were made in an already existing *add-interface* function in pertaining to a mobile node. Provision for adding multiple interfaces was added by introducing a new variable *numifs*, to denote the number of multiple interfaces that can be added. Some other minor modifications in other files were also done. Please refer to the TeNs package [6] for further information regarding code changes and hacks.

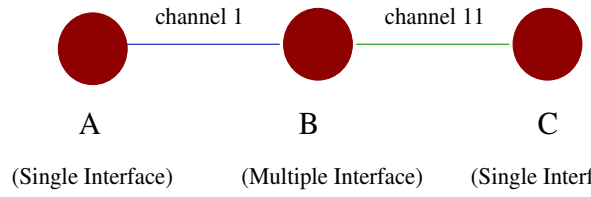


Figure 4: Scenario consisting of 3 nodes. The middle node has 2 interfaces.

To show, that multiple interfaces was indeed successfully implemented, the following simulation was run (see Fig. 4). Three nodes A, B, C positioned along a straight line were created as shown in Fig. 4. Node B had two interfaces – one at channel 1 and other at channel 11. Node A was at channel 1 and Node C at channel 11. Two separate connections were started : B to A and B to C. Non-zero throughput, at both nodes A and C, were observed at the end of the simulation, showing that the multiple interface at the middle node was used, as C cannot receive packets meant for A and vice-versa. The reason for this is the channel interference implementation, which was briefly mentioned in 4. A network interface configured to receive packets at a particular channel, will not receive any packet transmitted on a different channel, such that difference between the channels is greater than 5. Hence, clearly the two interfaces at node B must have been used for non-zero throughput observed at A and C.

[7] Specifically the files mobile/modulation.cc,h and mac/wireless-phy.cc,h were modified/added in the core NS directory.

5.4 Multihop in Wireless Nodes

Implementation of this feature was also found to be essential, given the experiments of multi hop scenarios as in the case of DGP [5]. However, implementing this feature was found to be quite non-trivial. It required changes at various levels in NS. Changes at levels of MAC layer, Data Link Layer, Classifier, Packet Queues, etc, were required (see the TeNs code [6]).

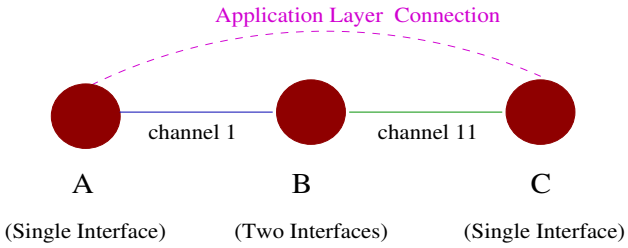


Figure 5: Scenario to show multi hop implementation. The middle node has 2 interfaces.

To show, that multi hop was implemented the following simulation was run (see Fig. 5). It is quite similar to the simulation scenario as in case of multiple interfaces above. However this time there was a single connection between A and C. Still the some non-zero throughput was observed at C. The only way this could have been possible is when the Node B which had multiple interfaces, did the routing of packets from A to C, leading to multi hop transmission. Besides, the throughput observed was as expected. See section 6 for the actual throughput values obtained, and further discussion.

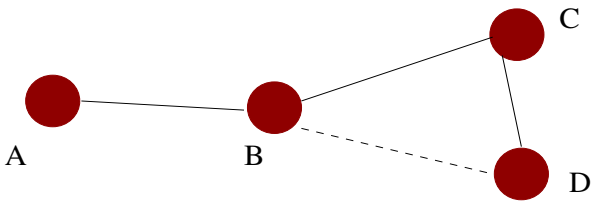


Figure 6: An example network topology. Node B and C have 2 interfaces each.

A problem that arises at this point is that, in this case the exact hop sequence is not under the control of the user, as it is determined by the topology and the scenario simulated. For example, consider the topology shown in Fig. 6, in which both C and D are in range of B, and also in range of each other. With simple forwarding as above, the packets meant for D can't be forced to go through C, as it will be directly received by D and rejected by C. This calls for explicit routing of packets, and hence for this purpose we implemented a new routing protocol which we name as **WLStatic** or *Wireless Static* Routing protocol. WLStatic is a static routing protocol for wireless networks. In other words, the routes are manually specified, and it contains no route management and maintenance stuff. So if a mobile node goes out of range and *statically* specified route is broken, still no update of routing tables, etc, is attempted. Therefore, it really caters only to scenarios where nodes are not moving, unlike the case of mobile ad-hoc networks and its routing protocols (like DSR[8]). Such a routing protocol is required to simulate the wireless networks like DGP [5], in which the nodes are fixed and the routing is static. Now using routing support provided by WLStatic, explicit forwarding of packets from node B to D via C, as required in above topology (Fig. 6) can be easily done.

For implementing WLStatic, we started with a copy of existing mobile ad-hoc routing called AODV[9], and modified it extensively. Basically a number of additional features of AODV were removed and only static

[8] Dynamic Source Routing
[9] Ad-hoc On Demand Routing Protocol

specification of routes was kept. See TeNs code [6] for more details. In addition, changes were incorporated so that routes can be specified from the tcl simulation script itself (see Appendix A for a brief manual on TeNs).

5.5 Adjacent Channel Interference Validation

We performed simple simulations and experiments to verify for the Channel Interference implementation (see section 3 and section 4 for additional details) in NS. To state the channel interference issue in more detail, the DSSS PHY (the physical layer for 802.11b) specifies 11 channels in the 2.4Ghz spectrum for communication, each of which is 22 MHz wide. However, these channels overlap and adjacent channels have overlap of 5 MHz (see Fig. 7). Only channels with a channel difference of five or more are completely independent of each other for eg. channels 1,6 and 11. This leads to adjacent channel interference when communication goes on channels that are close by, leading to lower throughput than normal.

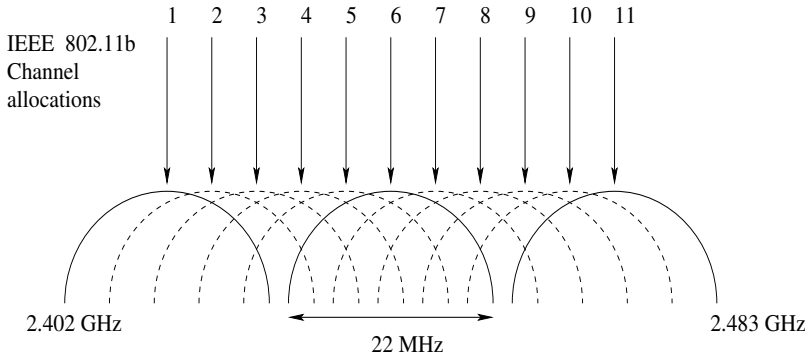


Figure 7: Channels in IEEE802.11b DSSS PHY.

Though channel interference, in reality, is a non-deterministic phenomena, we have attempted to capture its effects through the following implementation, based on the difference in channel at which the packet was sent and the channel at which the particular listening network interface was configured. The idea is to reduce the power based on channel interference, in order to increase the probability of bit error in the packet, which may lead to packet drop in later stages of packet processing.

$c1$, is the channel at which packet is sent.
$c2$, is the channel at which the receiving interface was configured to receive.
$txPr$, is power at which a packet was transmitted.
calcChangedPower(double txPr), returns new power, based on channel difference.

```
double calcChangedPower(double txPr) {
        diff = c1 − c2;
        if(diff >= 5) return 0.0;
        if(diff == 4)return txPr ∗ 1.0/5.0;
        if(diff == 3)return txPr ∗ 2.0/5.0;
        if(diff == 2)return txPr ∗ 3.0/5.0;
        if(diff == 1)return txPr ∗ 4.0/5.0;
        return txPr;
}
```

To see how closely the above mentioned implementation models actual adjacent channel interference in real

life, we performed the following experiment.

5.5.1 Setup

- **Simulation :** The simulation scenario, consisted of 4 nodes, each having single interface. All the nodes were in range and two parallel CBR connections are present, as shown in Fig. 8. Simultaneously the traffic was started for 20 seconds for each of the connections, and throughput was noted. Size of each packet was kept at 1000 bytes, and the packets were sent every 0.0005 second. Thus the traffic was full (the data rate capacity of the link was kept at its maximum – 11 Mbps). RTS/CTS (as discussed in 5.1.1) was kept on. The *Shadowing* propagation model was used at maximum possible (100 mw), for nodes to be always in range, and no gray region phenomena crops up. The throughput for the two CBR traffics were calculated from the trace file generated by NS, at different channels for the two connections.

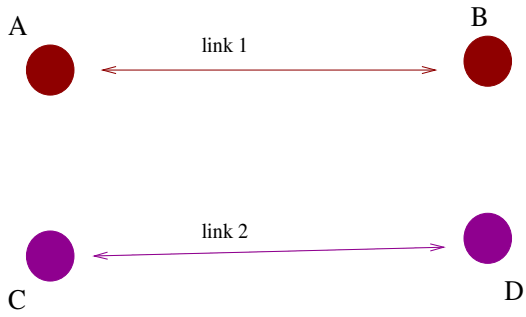


Figure 8: The scenario created for channel interference validation.

- **Experiment :** The experimental scenario was arranged in manner, that it is as close to simulation scenario described above. Nodes A and C (see Fig. 8) were Access Points in the experiment, while, B and D were laptops. The APs were connected to the IITK LAN. To generate the traffic on the link, *Netperf* [9] software was used. We ran the server, called *netserver*, on one of the PCs in the LAN. Then at various channel differences between the two connections, the netperf client was run on the laptop, which connected to the netserver. UDP traffic was used, with packet sized kept at 1000 bytes, and the throughput reading was obtained for a period of 20 seconds of traffic. As in the simulation scenario above, RTS/CTS mechanism was kept on and the power was kept at 100 mw. Care was taken, using the concept of SSID[10], to prevent association of the laptops to an AP other than we intended to.

  We used Cisco AP, and PCMCIA cards. For each value of channel difference (varying between 0 to 10), the experiment was performed and we averaged over the readings obtained in these cases, to obtain a benchmark reading for comparison with the simulation results. As mentioned before, our objective was to see if the general trend of readings from simulations match those from the experiments, and to modify or augment NS if they didn't quite match.

5.5.2 Results and Discussion :

Fig. 9 shows the comparison between the throughput values obtained from actual experiment versus NS simulation. It can be seen that similar trend is present in both readings. However the actual readings do not exactly match, as expected, because channel interference (like almost everything in wireless communication) is a probabilistic phenomena and can't be accurately captured. Also, one may note that the throughput values obtained in actual experiment are slightly lesser than those from the simulation. One factor could be that the experiment was performed in a location where there was high external interference (like trees and walls) as well. Another observation while carrying out the experiment was that the data-rates for the traffic

[10] Service Set Identifier

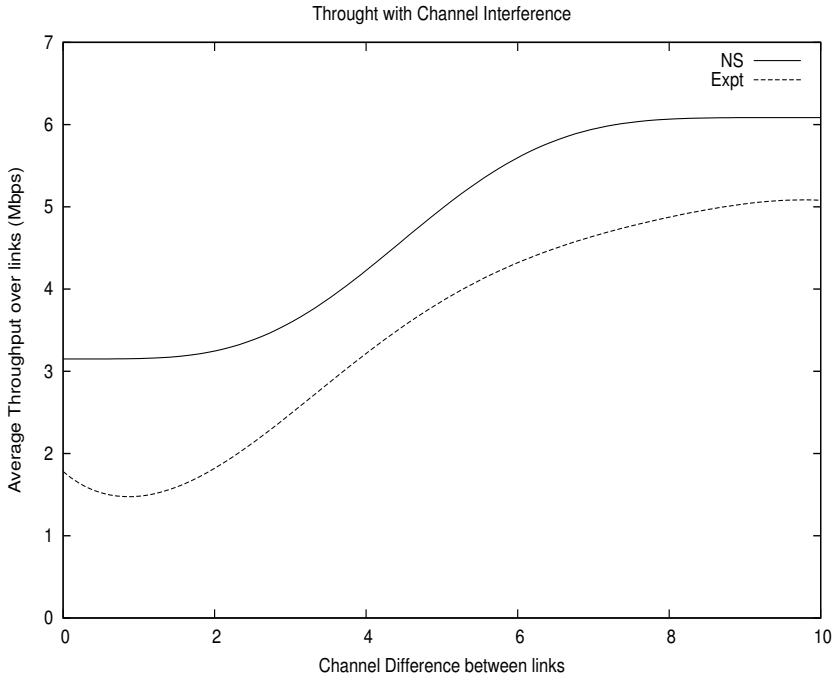


Figure 9: Comparing simulation results with experiment, for validation of channel interference.

frequently switched to lower values (like 2 or 1 Mbps). This indicates that rate-transitions in actual PC cards may also be governed by bit error (in addition to power). However the IEEE802.11b standard doesn't specify any algorithm for data-rate transition and its up to the manufacturer to implement an algorithm of its choice. Hence no attempt was made to capture this in NS, as different manufacturers may come up with different implementations for the same.

Note that there is the small drop in the throughput in case of the experiment, when the channel difference is 1, as compared to channel difference of 0. This is because of carrier sense. In case of pairs communicating on same channel, nodes can sense packets in the channel even though those packets may not be meant for them. In such a case, it senses the channel as busy and thus backs off from the channel contention. However when the channels at which the pairs are communicating are different, nodes are not able to sense packets which are sent on a different channel and thus assume the channel to be free. However, when such nodes try to communicate, their packets interfere with other packets (packets sent at other channel), causing packet drops. Thus, the overall throughput is hampered.

5.6 Directional Antenna Implementation

The feature of directional antenna was added to NS in an earlier project (see [7]). As discussed in section 4, this implementation is quite simple. Basically it ensures that the signal propagation is effective only within a sector (with the source as the center), with included angle and its relative position configurable by the user. As an example, consider the scenario shown in Fig. 10. Due to confining the transmission of A using a directional antenna, only B and C can detect its transmission and no other nodes. We verified this in NS by running a simulation script for the same scenario. It was observed that nodes B and C could receive packet from node A, but not nodes D and E.

However the actual directional antenna has a more complicated radiation pattern, complete with side-lobes. An example of a typical radiation pattern is shown in Fig. 11. We further improved upon the existing directional antenna, to add support for antennas that have more realistic radiation patterns like the one shown in Fig. 11. We have included eight different antenna pattern radiation files , each of which has similar pattern as the one shown (Fig. 11), but the orientations differ. Now the user can specify the kind of

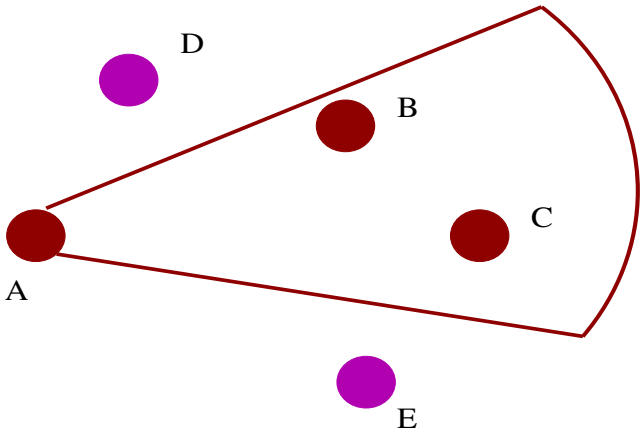


Figure 10: Basic (Ideal) directional antenna. This antenna simply controls the region that is illuminated by the transmission of node A.

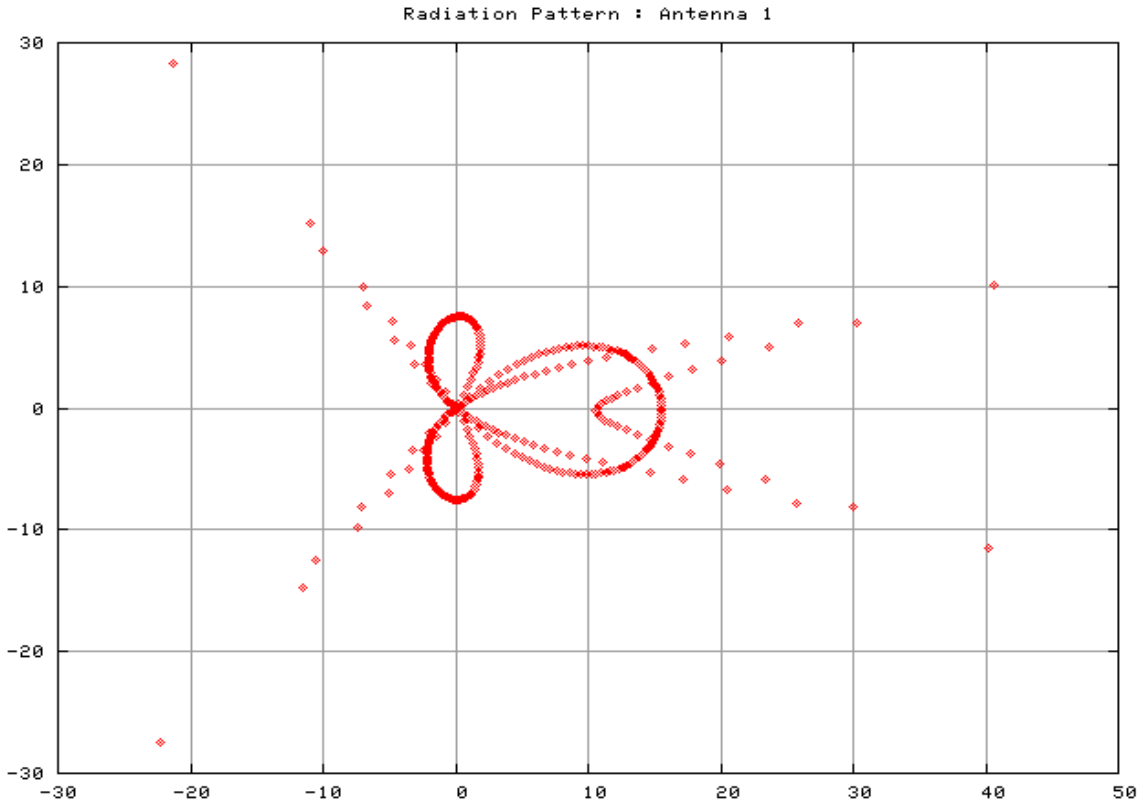


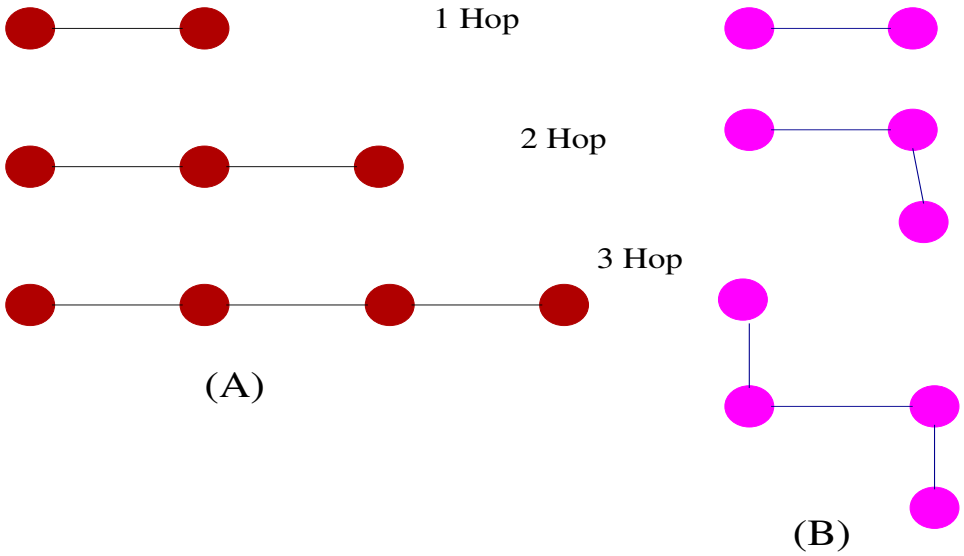
Figure 11: Typical directional antenna radiation pattern.

antenna to use through the simulation script (see Appendix A for details on syntax). This implementation is extensible, in the sense, that support for other specific directional antennas can be added, using the same radiation-pattern file reading approach. See TeNs code [6] for implementation code details.

There are certain issues, however. Currently, there is no standard format for antenna-radiation pattern file which would be required if no change in actual code of NS is required. Also, currently only one of the nine types of directional antenna (discussed above) can be selected.

6 Applications of TeNs

The improved capabilities and additional features present in TeNs, as compared to conventional NS, enable simulation of multihop wireless mesh networks using directional antenna, in addition to the already existing simulation support in NS. In this section, we present some results for simulation of typical multihop wireless links using directional antenna, which could be part of a larger and complete network. The observation and results also confirm the correctness of our implementation.



All the interfaces in scenarios of type (A) above have omni antenna, while those in scenarios of type (B) have directional antenna. The distance between nodes in a link is around 50 metres. Connection (both TCP and UDP) are established between the endpoints in each case.

Figure 12: Some typical multihop scenarios present in wireless mesh networks.

Consider four cases of multihop links, as shown in Fig. 12. We consider UDP and TCP performance over these small topologies. These simple scenarios can be thought of, as subsets of a larger wireless mesh network. There are two cases shown in Fig. 12 - with omni-directional antenna and another with directional antenna. In case of omni-directional antenna, we use a linear network, with all links in a straight line, and only adjacent nodes being able to hear each other. In case of directional antenna, we use a similar configuration, but with adjacent links at 90_o to one another in the 2-hop and 3-hop cases. This change in configuration was done, because, for the same separation as in omni directional case, even nodes other than adjacent nodes can also hear each other. This can be attributed to the higher antenna gain of directional antenna. In all cases, we use our static routing protocol - *WLStatic* with connections established between

the end-points of the nodes. Table 3 shows the throughput obtained in each of the above cases.

Antenna	RTS/CTS	TCP/UDP	1-Hop (Mbps)	2-Hop (Mbps)	3-Hop (Mbps)
Omni	Yes	UDP	4.5	2.2	1.5
Omni	No	UDP	6.1	3.0	2.0
Dir	Yes	UDP	4.5	2.0	1.9
Dir	No	UDP	6.1	2.8	2.7
Omni	Yes	TCP	3.5	1.8	1.1
Omni	No	TCP	4.3	2.1	1.4
Dir	Yes	TCP	3.5	1.3	0.8
Dir	No	TCP	4.3	1.0	0.7

Table 3: Performance of Multihop 802.11

It can be readily seen that the overall throghput in cases when RTS/CTS is used is less than the throughput obtained when RTS/CTS is not used. This can be attributed to the extra round-trip overhead caused by the RTS/CTS packets. In case of omni-antenna, the throughput for 2-hop scenario is much less, as compared to the single hop scenario because two links cannot operate in parallel due to the meachanism of carrier sense. In case of directional antennae the throughput actually drops below than that obtained using omni antenna, which might seem a bit counterintuitive at first, but has a simple explanation. Two adjacent links, using directional antenna, continue with their own transmission in parallel. However, due to the transmission leakages along the side and back lobes (which is a feature of every directional antenna), packets get corrupted and are dropped, causing a dip in the throughput. The values obtained for TCP connections, as expected, are lower. This is attributed to additional extra overheads associated with TCP like congestion control, etc. For additional explanation please refer to [11].

TeNs is being used for simulation studies in research and other projects. See [11], and [12], in which TeNs has been used as part of research work.

7 Future Work

The entire project status has already been presented in Table 1. All the major goals have been completed and we have also released the new improved NS, with the new and fitting name **"The Enhanced Network Simulator"** or **TeNs**. Please see the TeNs website [6] for details.

We enlist the following as the future tasks.

- Support for additional modulation schemes like DBPSK. Also to implement changes in modulation schemes together with adaptive data-rate transitions. Currently only BPSK modulation is supported.
- Support for using user-specified antenna radiation file (of course, there has to be a standard format). Currently we are taking inputs from our own input file, that has support for only 8 basic directional antenna radiation patterns, plus another antenna that confines transmissions to a sector.
- Relocation of LLC code, so that different link layer protocols could be added.

ACKNOWLEDGMENT

We are indebted to Dr. Bhaskaran Raman and Dr. Dheeraj Sanghi, for their constant support, and invaluable suggestions, during the course of our project, without which our work couldn't really have progressed. We are also thankful to *Digital Gangetic Plain, Media Lab Asia Kanpur-Lucknow Lab*, for the infrastructural support they provided for carrying out our experiments for validation.

References

[1] IEEE Std 802.11-1999 [ISO/IEC DIS 8802-11]. Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications. . Technical report, LAN/MAN Standards Committee of the IEEE Computer Society, 3 Park Avenue, New York, NY 10016-5997, USA, 1999.

[2] IEEE Std 802.11b-1999 [ISO/IEC DIS 8802-11]. Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications. Higher-Speed Physical Layer Extension in the 2.4 GHz Band. Technical report, LAN/MAN Standards Committee of the IEEE Computer Society, 3 Park Avenue, New York, NY 10016-5997, USA, September 1999.

[3] Digital Gangetic Plain, Media Lab Asia Kanpur-Lucknow Lab. `http://www.iitk.ac.in/mladgp/`.

[4] S. McCanne and S. Floyd. Network simulator. `http://www.isi.edu/nsnam/ns/`.

[5] Pravin Bhagwat, Bhaskaran Raman, and Dheeraj Sanghi. Turning 802.11 Inside-Out. In *Second Workshop on Hot Topics in Networks (HotNets-II)*, November 2003.

[6] The enhanced network simulator. `http://www.cse.iitk.ac.in/~bhaskar/tens/`.

[7] Ankur Khandelwal. Peformance of 802.11b in a Grid Network. Technical report, Department of CSE, IIT Kanpur, India, April 2003. BTech Project Report.

[8] Cisco Aironet Wireless LAN Client Adapters. `http://www.cisco.com/en/US/products/hw/wireless/ps4555/index.html`.

[9] Netperf. `http://www.netperf.org/netperf/NetperfPage.html`.

[10] S. Haykin. *Communication Systems*. John Wiley and Sons, New York, 2001.

[11] B. Raman. Revisiting network and protocol design for 802.11-based rural internetworking, 2004. `http://www.cse.iitk.ac.in/~bhaskar/morphnet/topo-form.pdf`.

[12] Ashwini Kumar, Manav Ratan Mital, Dheeraj Sanghi, and Bhaskaran Raman. Topology construction for ieee802.11b networks. To be submitted to ICCCN'04.

APPENDIX

A How to make use of additional features in TeNs

We have discussed completely the enhancements contained in TeNs in this report. Here we discuss how a user can invoke these features through the simulation script. Almost all the features that we have added are optional, which means that the user can use them only if he wants to, and has the option to use TeNs just like ordinary NS. Here is a quick guide on howtos. All the normal procedures and methods for wireless simulations still hold. We are specifying only the additions. To get a complete picture and sample code see TeNs release website [6]. We will also make use of acronyms and variable names that should be self-explanatory to someone using NS for simulations.

1. To use the BPSK modulation, add the following statement.

   ```
   Phy/WirelessPhy set modulationscheme_ 1
   ```

   By default its value is set to 0 and is not used.

2. To make enable multiple interfaces in a node, say 3, add the following code.

   ```
   $ns_ node-config -numif 3
   set newnode [ $ns_ node]
   ```

3. To configure the channel and power of any network interface of the node.

   ```
   [ $newnode set netif_(0) ] set channel_number_ 3
   [ $newnode set netif_(0) ] set Pt_ 0.005
   ```

   Note that the power is specified in watts, and various network interfaces of a node can be referred using index starting from 0.

4. To configure directional antenna (say of type 0) for a network interface, first of all the node's antenna type variable should be set as shown.

   ```
   $ns_ node-config -antType Antenna/DirAntenna
   ```

   Then directional antenna can be added to individual network interfaces in the following manner.

   ```
   set ant [ new Antenna/DirAntenna ]
   $ant setType 0
   $ant setAngle 85
   $ant setWidth 10
   ```

   Note that here antenna type is set to 0. For antenna types 1 to 8 (see [6] for details of radiation patterns for each of these antenna types), the additional angle and width parameters are not required. Also note that an omni-directional antenna can be easily be configured using a directional antenna. Now, set the above antenna to the interface.

   ```
   [ $newnode set netif_(0) ] dir-antenna $ant
   ```

5. For making use of **WLStatic** routing, first of all the nodes should be configured for it using the following command.

   ```
   $ns_ node-config -adhocRouting WLSTATIC
   ```

Now, to create the routing tables at a paricular node, the syntax is given below.

```
[ $newnode set ragent_ ] addstaticroute <hopcount> <next hop> <dest> <interface>
```

Note that the nodes are specified using numbers starting from 0, which are assigned by default by TeNs. Similar is the case with specification of interfaces. A sample route specification could be the following.

```
[ $newnode set ragent_ ] addstaticroute 1 1 1 0
[ $newnode set ragent_ ] addstaticroute 2 1 2 0
[ $newnode set ragent_ ] addstaticroute 3 1 3 0
```

Such specifications should be done at each node.

B Brief description of code changes and hacks

NS version 2.1b9a was modified into TeNs. The basic structure of TeNs is same as NS2.1b9a. Modifications were made in already existing files, and a few files were added as well. A new directory *wlstatic* was added in *ns2.1b9a/* directory. Major changes were done in *~ns2.1b9a/mac/* , *~ns2.1b9a/mobile/* , *~ns2.1b9a/common/* , and *~ns2.1b9a/tcl/* directories. Given below are some additional details regarding the modifications.

1. `aodv`
   (a) `aodv_rtable.h` A new variable *link* of type *NsObject* was added to the routing entries to specify the interface through which a packet should be sent.
   (b) `aodv.h,cc` Code segments to transmit packets through one or all of the interfaces were added.
2. `wlstatic` The whole code struture for wlstatic is the same as aodv except for the wlstatic.cc file. An extra code segment to add new routes through a tcl script at the time of initialization was incorporated.
3. `common`
   (a) `packet.h` New packet types were added to take care of the newly added *wlstatic* protocol.
   (b) `packet-stamp.h` New functions to modify packet stamps namely the power and wavelength at which a packet was sent were added.
4. `mac`
   (a) `channel.h,cc` New class named *Channel_802_11* was added.
   (b) `mac-802_11.h,cc` Code for adaptive data rate transition was added here.
   (c) `wireless-phy.h,cc` Code for channel interference and invoking modulation scheme was added here.
5. `mobile`
   (a) `dir-antenna.h,cc` All code related to directional antenna is here.
   (b) `modulation.h,cc` Code for BPSK modulation and calculation of error probability was added here.
   (c) `propagation.c,cc,shadowing.h,cc,tworayground.h,cc` Code was manipulated to take care of cases when the distance between the sender and receiver becomes zero. Otherwise it caused some unpredictable behavior.
6. `queue` Changes in the files *dsr-priqueue.h,cc,priqueue.h,cc* were done to make the whole code consistent with the newly added wlstatic module.

7. `tcl/lib`
   (a) `ns-lib.tcl,ns-mobilenode.tcl,ns-node.tcl,ns-rtmodule.tcl` Changes related to aodv and wlstatic were made in these files.
   (b) `ns-default.tcl` Few more variables that con be bound to values were added.
8. `tcl/mobility`
   (a) `com.tcl` Some changes to ensure consistency with the newly added wlstatic was done.
9. `trace`
   (a) `cmu-trace.h,cc` Code to stamp a packet was changed.

C Comparison of TeNs with other network simulation tools

In general, there are two forms of network simulation. The first is analytical modeling by mathematical analysis, that characterizes a network as a set of equations. But it is usually over simplistic and is unable to simulate complex dynamic systems. Thus, software simulation of real events as discrete events is required. There are several such simulation software packages that are currently available. We discuss the features of some of such simulators in the following subsections.

C.1 REAL

It is a simulator for studying dynamic behavior of flows and congestion control schemes in a packet switched network. Network topology, parameters, protocols etc are described in a simple ASCII representation using a language called the NetLanguage.

C.2 NetSim

It is intended to offer a very detailed simulation of ethernet. It includes realistic modeling of signal propagation, the effect of relative positions of stations on events of the network, collision detection and handling and transmission of deferral mechanism. But is can't be extended to simulate modern networks.

C.3 Maise

It is a C-based simulation tool for parallel discrete event simulation. A logical process is used to model one or more physical processes.

C.4 JavaSim

It is a generic simulation tool. It is meant for simulating all types of real events in varied environments and disciplines, and does not focus exclusively on communication networks.

C.5 Glomosim

It is a package for simulating generic wireless and wired network scenarios. It models all the layers of OSI, with support for almost all the important protocols. It has a very good implementation of the physical layer, with support for various propagation models like Rayleigh Fading Distribution, Ricean Fading distribution etc. It resembles Ns-2 most closely among all the above simulators. However, it does not support channel interference, adaptive data rate control, etc. Some other simulators include *VINT*, *U-Net*, *Harvard Simulator*, *BONeS*, and *COMMET III*.