Network Simulator for Efficient Performance Parameter Testing & Evaluation

Dr. Atul M Gosai, Bhargavi H Goswami

Abstract— In this paper we present the analysis and performance evaluation of ns2 to facilitate the simulation and upcoming wired & wireless networks. We have briefly explained in the table the comparison between widely used network simulators. Here we have presented the framework for ns2 to facilitate the simulation and also explained the implementation steps. We have shown the performance evaluation matrix. We have also shown the steps to bring improvement in the specified factors affecting the performance of network.

Keywords- Network Simulator, Network Tools, Performance Evaluation, Throughput, Packet Loss, Retransmission, Queue Type and Queue Size.

1. INTRODUCTION

Networks continue to grow more complex as industry deploys a mix of wired and wireless technologies into large-scale heterogeneous network architectures and as user applications and traffic continues to evolve [9]. In order to evaluate the behavior and performance of protocols for wired and wireless networks, simulations are a good compromise between cost and complexity, on the one hand, and accuracy of the results, on the other hand[10]. Since there are many simulators for wireless networks, it is often difficult to decide which simulator to choose [10].

After our literature review we can suggest that to begin our study for the enhancement of any research one should be fundamentally thorough with the existing algorithm. Modeling of existing algorithm over wired & wireless would be the first step towards many upcoming positive development and enhancement with improvements in current technology.

Simulators can provide full control when simulating wired & wireless network behaviors [12]. In the past decade, the test bed based evaluation approach in wireless networking research has emerged to be a popular means due to the emphasis researchers on practicality [12][14]. Surely the results obtained by simulators can never beat the results obtained by practical approach but as far as feasibility is concerned, practical approach is not possible to be implemented in each situation because of the few factors like cost of equipments, area covered, feasibility criteria, network traffic availability in even and odd hours, etc.

But, Simulators enjoy little unique advantage over test-beds. In simulations, network scenario scan be conveniently constructed and modified, and statistical data can be easily collected. Simulations can model a large scale network with a variety of topologies, which would be expensive to build with test-beds. In addition, test-bed results are often heavily affected by the surrounding environment, in which many factors are unpredictable and uncontrollable [12].

It is believe that this survey will aid researchers, application and tool developers while selecting an appropriate tool for their implementation [11] [14].

2. TYPES OF PERFORMANCE ANALYSIS

There are three techniques that are observed for analyzing the performance of wired and wireless network. Our observations are stated below.

2.1 Analytic Modeling:

In this model we perform analysis using numerical calculations with mathematics using probability, calculus, operation research, queuing networks, etc.

2.2 Computer Simulation:

Here, modeling is developed on simulator where realization of physical behavior of developed network model is set and developed using probability, statistics and queuing theory.

2.3 Real Time Physical Measurement:

Actual test are performed over the network under this test. Generation of actual situation is done and test data are fed to the network situations. Here, actual data are used for analysis.

Analytic model is pure mathematics based and so very tedious which if not computed with accuracy, may give us inaccurate results. Real Time Physical Measurement is not feasible approach especially when network includes hundred of nodes and costly routers with costly other network components. This do not permit us to go with the option of costly Real Time Physical Measurement nor tedious and error pron Analytic Modeling. Thus, we have limited our study methodology up to network simulator as there are a number of advantages to this approach: lower cost, ease of implementation, and practicality of testing large-scale networks [8].

3. TYPES OF NETWORK SIMULATORS

Network Simulator is a pure event based simulator and can be of two types [13].

1. Discrete Event Simulator and
2. Continuous Event Simulator.

3.1 Discrete Event Simulator:

In discrete event simulator, the representation of time is quantified and the system state changes only when an event occurs. For example, arrival of person in queue of railway reservation or departure of person from ticket booth after taking ticket. Here, state values are always integer.

3.2 Continuous Event Simulator:

In continuous event simulator, models time as a continuous progression. Here, state values are always real values. For example, snake covering distance, water flowing through the mountain. Generally computer simulators are discrete event simulators.

4. NETWORK SIMULATORS COMPARATIVE STUDY

Widely Used simulators are Ns-2, GloMoSim, OMNet++, OPNet, and QualNet. While for physical layer2, very few simulator tools are used. The theoretical, numerical, statistical analyses are vastly used for physical layer. As a simulator, NS2, GloMoSim, Qualnet, OPNET, OMNET++ is used for the implementation of the error model at the physical layer [6]. In this section, we will give a short overview and a comparative study about the five widely used network and system simulators, respectively.

In this sub section we summarize the most interesting capabilities, advantages, base language and type of existing simulating tools for wired and wireless networks in given table. Table 1 has all simulators considered in the previous section listed in the consecutive columns and main features and its base [7] in the context of all simulators in the consecutive rows, respectively [3].

Wireless Networks simulators exhibit different features and models. Each has advantages and disadvantages, and each is appropriate in different situations. In choosing a simulator from the available tools, the choice of a simulator should be driven by the requirements. Developers must consider the pros and cons of different programming languages, the means in which simulation is driven (event vs. time based), component- based or object oriented architecture, the level of complexity of the simulator, features to include and not include, use of implementation, and practicality of testing large-scale networks [8].

TABLE I: CHARACTERISTICS OF DIFFERENT NETWORK SIMULATOR TOOLS

	Protocols Supportted	Types	Based On
OPNET	ATM, TCP, FDDI, IP, Ethernet, Frame Relay, 802.11	Commercial	Finite State Machines
OMNET++	Wireless Oriented	Open Source for Academia	Object Oriented Concepts C++
GlomoSim	Wireless Networks	Open Source	Object oriented Concepts JAVA and PARSEC
Qual Net	WAN, Wired, Wireless Networks	Commercial	Object Oriented Concepts C++
NS-2	TCP/IP, Multicast, Routing, TCP Protocols, Wired & Wireless Networks	Open Source	Object Oriented Concepts of C++ and OTcl

5. NETWORK SIMULATORS COMPARATIVE STUDY

Widely Used simulators are Ns-2, GloMoSim, OMNet++, OPNet, and QualNet. While for physical layer2, very few simulator tools are used. The theoretical, numerical, statistical analyses are vastly used for physical layer. As a simulator, NS2, GloMoSim, Qualnet, OPNET, OMNET++ is used for the implementation of the error model at the physical layer [6]. In this section, we will give a short overview and a comparative study about the five widely used network and system simulators, respectively.

In this sub section we summarize the most interesting capabilities, advantages, base language and type of existing simulating tools for wired and wireless networks in given table. Table 1 has all simulators considered in the previous section listed in the consecutive columns and main features and its base [7] in the context of all simulators in the consecutive rows, respectively [3].

Wireless Networks simulators exhibit different features and models. Each has advantages and disadvantages, and each is appropriate in different situations. In choosing a simulator from the available tools, the choice of a simulator should be driven by the requirements. Developers must consider the pros and cons of different programming languages, the means in which simulation is driven (event vs. time based), component-based or object oriented architecture, the level of complexity of the simulator, features to include and not include, use of parallel execution, ability to interact with real nodes, and other design choices.

There are several popular network simulators such as Tiny OS Simulator (TOSSIM), National Chao Tung University Network Simulator (NCTUns), NetSim, Java Sim (Jsim), etc. Out of all above stated simulators, NS2 is most widely used network simulator [1][6]. NS2 is open source and easily available on internet. It is easy to download and install on any linux (open source) based platform like Ubuntu, Fedora, Suse. Thus, big cost benefits especially for those whose research projects are not sponsored by large amount and large rich organization and university. NS2 support for applications, protocols, network types, network elements, topologies, traffic modes, etc [1] [6].

Above stated all simulators are not implementing protocols according to RFC (Request for Comment) standardization. Only NS2 provides all its protocol implementation according to RFC standardization specification [13].

In addition to above stated features, NS2 do not create problems for errors like SYN, END, ACK, URGENT [13].

6. NETWORK SIMULATOR-2 ARCHITECTURE

NS2 is a discrete event and packet level simulator [5] developed at US Berkeley to analyse the performance of wired and wireless networks [2][13]. It is object oriented simulator written in C++ and Otcl (Object oriented TCL Tool Command Language) are object oriented then the question arise in mind that why NS2 does uses two languages? Answering these question because the simulator needs to fulfill two different purposes.

1) First, the detailed simulation of protocols require a system programming language which can efficiently manipulate bytes, packet headers and implement algorithm than run over large data set. For such task, run time (execution time) is more important than turn around time. C++ is fast to run but slow to change, making it suitable for detailed protocol implementation [13].

2) Second, research in networks involves slightly varying parameters, setting different configuration or quickly exploring different scenario. For such task, iteration time is more important than run time. OTcl runs slow but can be changed quickly, making it suitable for varying parameters, configuration settings, etc. Thus, OTcl can be used to set up different topologies, varying the number of nodes in topology, etc [13].

C. NS2 Architecture is composed for 5 parts:

a. Event Schedular

b. Network Components

c. TclCl (Tool Command Language with C++ and OTcl Language)

d. OTcl Library

e. Tcl 8.0 scripting language

Advantage of using Tcl language is, it does not need special editors, nor does it has any complex structure that need to be followed during coding [2]. The sequence of statements is also not necessary to be maintained, as it automatically fetches the required instructions for topology generation and other task as and when required.

To develop a network model research needs to follow following broad steps:

Step 1: User writes simulation script using Tcl.

Step 2: Make the entry of each event occurred during simulation execution into trace file.

Step 3: Display graphical form of simulator model during execution using nam trace.

Step 4: Generate graphs based on trace file analysis and implementing c++ codes for analysis & graph generation based on dot tr file

7. NETWORK MODEL GENERATION

Now moving towards 1st step we go for detailed study of generation of simulation scripts. As stated before, for generation of simulation script we use Tcl scripting language [2]. We generate a network simulator having 8 node structures with one router which when congested due to heavy traffic drops the packets. Fig 1 shows the Network Topology providing us the overall idea for our test bed. We start our simulation by generating object of the simulator. This can be done using following syntax:

set <object name> [new Simulator]

One example for the above specified syntax:

set ns [new Simulator]

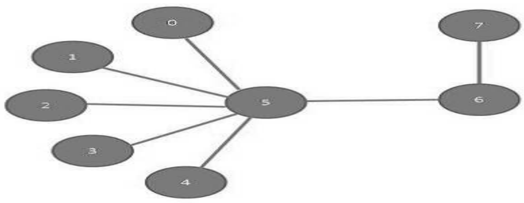

Fig. 1 The architecture of nodes

We can use routing or we can omit this activity under simulator.

Syntax: **$<object>rtproto <protocol>**

Eg: **$ns rtproto DV**

Before moving forward, we should generate the trace file to store the output or log of the execution which is further used for analysis of experiment after the experiment ends. We can do this as stated below.

Syntax:

Set <file instance> [open <filename>.tr w] Eg: set f [open out.tr w]

This creates the file with file name and open that file in write mode. After opening the file, we need to set our simulation to store all events to that log file in .tr format. This can be done by:

$ns trace-all $f

Thus, we have set all settings for trace file where entry to file is made when simulator is in run mode. Once the task of trace file generation is over, we move forward for generation of nam file to display the graphical format of events occurring in trace file. This can be done as follows:

First create object for nam file and then attach that object to operation.

eg. **set nf [open out.nam w]**
$ns namtrace-all $nf

Now next step for us would be the generation of topology. Topology consists of nodes and edges.

Talking network point of view, we generate senders and receivers and intermediate gateways and then we set TCP connections between nodes. Generation of nodes would be the first step to topology generation.

eg. # 0, 1, 2, 3, and 4 are sender nodes.
set n1 [$ns node]

5th node named as G is acting as gateway or intermediate node

set n5 [$ns node]

7th node is named R is acting as receiver connected through gateway node

set n7 [$ns node]

Hash indicates comments here.

Here we named G and R as gateway and receiver respectively. Till we do not set the parameters and link specifications between the nodes, each sender, receiver or intermediate nodes are treated as same i.e a simple node or terminal. We can see that definition has no difference between them whether a sender, receiver or gateway.

As there are many senders sending to single receiver in our topology, we prefer to distinguish between the traffic flows from each sender to single receiver. For this purpose, we would like to suggest following syntax.

Define different color for data flow.

$ns color 1 Red;

$ns color 2 Sea Green; $ns color 3 Blue;

This gives different colour to traffic generated between source n1 to receiver R, from n2 to R, n3 to R and n4 to R. This research specifies the colours between different senders sending traffic to single receiver and then next step is defining the edges between created nodes.

 # create links between nodes

$ns duplex-link $n1 $n5 6mb 1ms Droptail
$ns duplex-link $G $n7 6mb 1ms Droptail

In the congestion situation, our router would try to accommodate maximum of the packets limited to size of queue defined. To define this, we can use following code.

Eg. $ns queue-limit $n5 $n7 5

The example in this research indicates that buffer size to accommodate unsent packets is set to 5 packets. After 5 packets are added to queue, further congestion results to packet dropping.

This research also defines the layout of the proposed topology. This is done using following code for each node and goes on as per topology requirement.

$ns duplex-link-op $n1 $n5 orient right-up
$ns duplex-link-op $n2 $n5 orient right
$ns duplex-link-op $n3 $n5 orient right-down
$ns duplex-link-op $n4 $n5 orient right

We can start monitoring queue at particular time. This can be done using following code.

$ns duplex-link-op $n1 $n5 queuePos 0.5.

We can use this code for each connection between nodes with gateway. Next job for us would be creation of traffic. Before execution start, even if we have topology ready, parameters determined, execution of simulation is effective only if we generate test traffic. Even the analysis can be done only after the traffic is generated and executed.

Traffic can be of two types: Connection oriented and Connectionless. To create connection oriented traffic, we need to take help of TCP and SINK agents available with NS2. And to create connectionless traffic, we need UDP and NULL agents. This can be done between each sender and receiver using following code.

```
#Setup a TCP connection
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $n7 $sin

$ns connect $tcp $sink $tcp set window_ 8 $tcp set fid_1

#Setup a FTP over TCP connection set ftp [new Application/FTP]

$ftp attach-agent $tcp

$ftp set type_ FTP

#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n4 $udp
set null [new Agent/Null]
$ns attach-agent $n7 $null
$ns connect $udp $null
$udp set fid_

#Setup a CBR over UDP connection set cbr [new Application/Traffic/CBR] $cbr attach-agent $udp

$cbr set type_ CBR

$cbr set packet_size_ 1000 $cbr set rate_ 1mb

$cbr set random_ false
```

In the above written code, we set TCP connection on source and attach it to sender node. Then we created TCP Sink to receiver. We connected the TCP and TCPSink. Also, we set window size equal to 8. The set up flow of traffic between 2 nodes have been given id 1 in last line. There after we created FTP over TCP connection. Not only data packets also files can be transferred over the same connection between nodes n0 and n7.

Then we created UDP connection between nodes n4 and n7.We generated CBR (constant bit rate) traffic over UDP connection with the packet size equal to 100 and rate set to 1mb with no randomization. These connections set are shown in figure 2.

Same code would be implemented between other nodes with options available for agents are TCP, TCPSink, UDP, Null, CBR, etc. Connection between these nodes gets completed only after connecting these nodes.

We are done with all other task. Moving towards end of the definition of tcl file, we define finish procedure.

```
Proc finish {} {
    global ns

$ns flush-trace

puts "running nam..."
exec nam -a out.nam &
exit 0 }
```

Above stated code sets global instance of ns and clear trace logs stored in trace file. Gives output string indicating that now graphical version of simulator in the form of nam (network animator). Thereafter, it executes NAM file. Finally it exits. Now the only thing remaining in our model is to set the start and stop time for running our simulator model. This can be done as follows.

```
$ns at 0.5 "$ ftp, cbr start" $ns at 150.0 "$ ftp, cbr stop" $ns at 150.05 "finish"

$ns run
```

Thus, after calling the finish procedure, we have called the instance to run the created simulator model.

Before we can proceed with performance evaluation, we must choose the different metrics that would help us in making comparisons. There could be different metrics to determine the performance like throughput, delay, jitter, packet loss. The choice of metric would depend upon the purpose of the network setup. The metrics could be related to the different layers of the network stack. For example, TCP throughput is based on the application layer, whereas IP round trip time is based on the network layer. For example, a network supporting multimedia applications should have minimum delay and jitter. Packet loss might not be a critical issue for such a network. However, packet loss might be a considerable factor for networks supporting textual data oriented applications, something like someone downloading by FTP.

Once the metrics have been chosen, one goes for their quantitative evaluation by subjecting the network under diverse conditions. For example, one could make step by step increments in bandwidth of the links, which in turn improve the throughput. However, the throughput might get saturated beyond the certain point. That is, further increase in bandwidth would not improve throughput. Thus, the optimum value of bandwidth has been determined.

It might not be always possible rather feasible to obtain best performance from a network due to various factors like high cost equipments, network and topology complexity, compatibility of available network components. In such cases one would like to obtain optimum performance by balancing different factors which we call trade off between available benefits and inevitable odd situations.

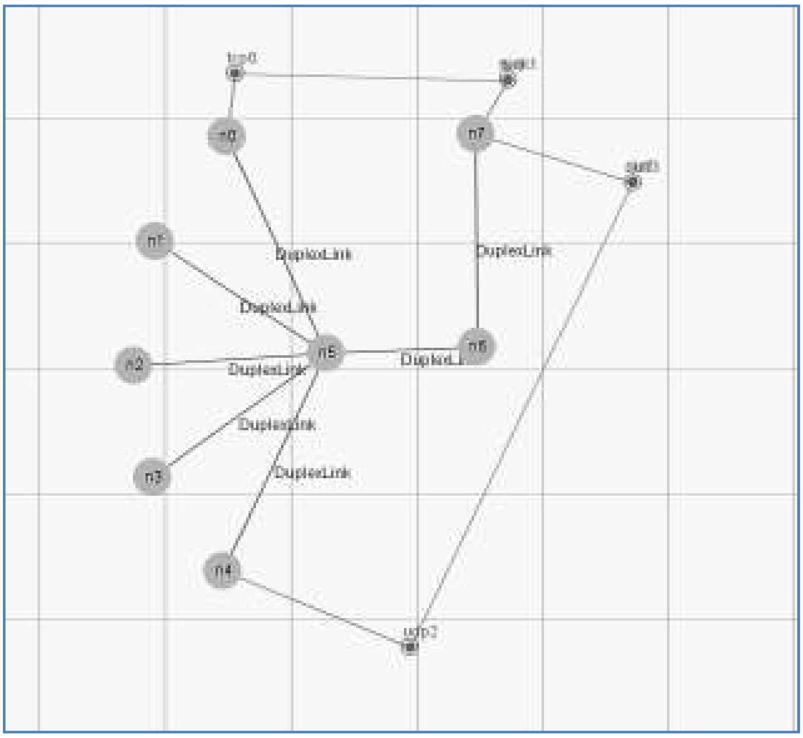

Fig. 2 TCP and FTP are set for communicating nodes

8. PERFORMANCE MEASUREMENT:

Latency: It can take a long time for a packet to be delivered across intervening networks [16]. In reliable protocols where a receiver acknowledges delivery of each frame of data, it is possible to measure this by taking help of round-trip time [3].

Packet loss: In some cases, intermediate devices in a network will lose packets [15]. This may be due to errors, to overloading of the intermediate network, or to intentional discarding of traffic in order to enforce a particular service level. In our case after queue is full, our network start dropping packets

Retransmission: When packets are lost in a reliable network, they are retransmitted. This incurs two delays: First, the delay from re-sending the data; and second, the delay resulting from waiting until the data is received in the correct order before forwarding it up the protocol stack[4][15].

Throughput: The amount of traffic a network can carry is measured as throughput, usually in terms such as kilobits per second. Throughput is analogous to the number of edges between sender and receiver, whereas latency is analogous to its speed limit [15][18].

9. PARAMETERS AFFECTING THE PERFORMANCE OF NETWORK

Different parameters can together or independently affect the output that determines how well a network would perform. Few such parameters are specified below.

Queue type and queue size: The queue of a node is implemented as a part of a link whose input is that node to handle the overflow at the queue. But if the buffer capacity of the output queue is exceeded then the last packet arrived is dropped. We do set the buffer capacity by using queue size [16][17][18].

In our case, queue type is set to Drop Tail and Queue size is 5 packets, after that, packet start dropping instead of en-queueing[18].

Bandwidth: It is the maximum data transfer rate that is allowed by the link. It is measured in unit bits per second (bps)[18].

Propagation Delay: It is the amount of time required for a packet to travel from one node to another [4][18]. If the propagation delay is high then throughput will be low i.e they are inversely proportional to each other. This also inspire high rate of resending if the wait time for ACK is small at sender side.

10. PERFORMANCE EVALUATION TECHNIQUES

Before starting with tuning the performance of a network one must remember that the performance, to some extent, depends on the workload as well as the topology [4][17]. A given topology might give different throughputs under CBR and exponential traffic. Keeping this in mind, one can go for studying an actual network. Otherwise one can simulate its performance using suitable parameters. These simulations would largely depend on queuing theory.

Network Performance Evaluation in NS2

In this section we discuss how to evaluate performance of a network by simulating it with ns2 [14].

Design, develop and generate a network topology to be used within the simulation. This could be a wired network, in which case the topology remains fixed. However, for a wireless network with mobile nodes the topology would change with time, or randomly. This we have kept for future work [4].

Once the topology has been generated, traffic source (sender) and destination (receiver) are fixed. Assign suitable traffic sources to the source nodes like tcp, udp and fpt, and traffic sinks like sink, null to the destination nodes.

Some of the parameters that can be used for comparative study of performance of the network are: link bandwidth, propagation delay, node queue type. For example: In ns2 we do create a link by using this code:

1 $ns simplex-link $n2 $n6 0.3Mb 100ms DropTail 2 $ns duplex-link $n3 $n6 0.3Mb 100ms DropTail

In this code there could be three parameters namely bandwidth, propagation delay and queue type. We can vary these parameters and could possibly obtain different throughputs. From there we can determine the conditions that provide higher throughput values. That is, we can alter different parameters and study their effects on one or more performance metrics and thereby filter out the combination of parameters that gives best performance.

Performance of the network can be determined by considering different metrics for example 'Throughput'[18]. We can vary these parameters and could possibly obtain different throughputs, which can be plotted using x-graph. From there we can determine the conditions that provide higher throughput values. Make suitable combinations with the parameters that will bring some changes in the throughput. Use the best combination of parameters which will bring the improvement in available throughput and implement it giving us best throughput.

This study considers only one performance metric. i.e. throughput in our experiment. Other metrics like packet loss, latency [15] and retransmission [18] can also be measured to evaluate the performance of a network in a more accurate way which will help us to setup the network in a improved manner giving us better result than previous.

11. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed few characteristics of most widely used network simulator tools based on a couple of papers and our survey. Then, with the results of our survey we shown that best suitable network simulator tool for specific case of ours would be NS2 and discussed the criteria of selection of NS2 tool for our requirement. With this paper we documented important issues with implementation of the required topology in selected network simulator tool that is NS2. We suggested increasing the reliability and performance criteria to be considered and to be improved repeatedly of simulation studies.

Based on our work, researchers can take decision to use various simulation tools as per individual's requirement. This paper would help deciding optimum network simulator for the researchers.

Finally, we hope, the result is presented in this paper will motivate the researches to put their efforts in thorough descriptions of the simulation scenarios and taking care of issues in simulation studies of wired and wireless networks.

12. REFERENCES

1. Nicola Baldo, Federico Maguolo, Marco Miozzo, Michele Rossi and Michele Zorzi "ns2-MIRACLE: a Modular Framework for Multi-Technology And Cross-Layer Supporting Network Simulator-2" in Proceeding of the 2nd international conference on Performance evaluation methodologies and tools ICST, Brussels, Belgium, 2007.
2. L. Begg, W. Liu, K. Pawlikowski, S. Perera, and H. Sirisena, "Survey of Simulators of Next Generation Networks for Studying Service Availability and Resilience", Technical Report TR-

COSC 05/06, Department of Computer Sci-ence & Software Engineering, University of Can-terbury, Christchurch, New Zealand, February 2006.

3. Nurul I. Sarkar, Syafnidar A. Halim, "A Review of Simulation of Telecommunication Networks: Simulators, Classification, Comparison, Methodologies, and Recommendations"Cyber-Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT), March Edition, 2011.
4. E. K. Bowdon, "Using simulation to evaluate system performance"presented at Proceedings of the 11th workshop on Design automation 1974, pp. 359-365.
5. Harsh Sundani, Haoyue Li, Vijay K. Devabhaktuni, Mansoor Alam, & Prabir Bhattacharya, "Wireless Sensor Network Simulators A Survey and Comparisons" presented in International Journal Of Computer Networks (IJCN), Volume (2) : Issue (5).
6. S. Mehta, Niamat Ullah, Md. Humaun Kabir, Mst. Najnin Sultana, and Kyung Sup Kwak, "A Case Study of Networks Simulation Tools for Wireless Networks", Published in Third Asia International Conference on Modelling & Simulation, 25-29 May, 2009.
7. E. Egea-Lopez, J. Vales-Alons o, A. Martinez-Sala, P. Pavon-Mari, and J. Garcia-Haro, "Simulation scalability issues in wireless sensor networks", IEEE Communications Magazine, 44(7):64–73, July 2006.
8. D. Curren. A survey of simulation in sensor networks. Student project, www.cs.binghamton.edu/˜kang/teaching/cs580s/david.pdf, 2007.
9. J. Heidemann, K. Mills, and S. Kumar, "Expanding Confidence in Network Simulations," IEEE Network, vol. 15, no. 5, 2001, pp. 58-63.
10. J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus. Comparative Study of Wireless Network Simulators. The Seventh International Conference on Networking, pages 517-523, 2008.
11. Imran, M.; Said, A.M.; Hasbullah, H. , "A survey of simulators, emulators and testbeds for wireless sensor networks" published in Information Technology (ITSim), 2010 International Symposium, in Kuala Lumpur, Malaysia. Date:15-17 June 2010 Page(s): 897 – 902 Volume: 2,
12. KefengTan, Daniel Wu, An(Jack) Chan, Prasant Mohapatra, "Comparing simulation tools and experimental test beds for wireless Mesh networks", Pervasive and Mobile Computing 7(2011)434–448.
13. T. Issariyakul and E. Hossain, Introduction to network simulator ns2, Springer, Nov. 2008.

14. M. P. Barcellos, G. Facchini, and H. H. Muhammad, "Bridging the gap between simulation and experimental evaluation in computer networks," presented at the 39th Annual Symposium on Simulation, 2006, pp. 286-293.
15. M. S. Corson. and J. Macker, “Mobile ad hoc networking: routing protocol performance issues and evaluation considerations”. Internet RFC 2501, January-1999,http://www.ietf.org/rfc/rfc2501.txt.
16. Andreas Hanemann, Athanassios Liakopoulos, Maurizio Molina, D. Martin Swany, “A Study on Network Performance Metrics and their Composition”, http://marco.uminho.pt/~dias/MIECOM/GR/Projs/P4/TNC_Metric_Comp_FWork-full-v4.pdf
17. CISCO white paper, "Successful Implementation Strategies for Service-level Managment," CISCO, 2000.
18. http://www.cisco.com/web/about/security/intelligence/network_performance_metrics.html on date 3/3/2011.

AUTHORS PROFILE:

Dr. Atul M Gosai:

Ph.D., MCA, BBA is an Assistant Professor, Department of Computer Science, Saurashtra University, Rajkot. Atul Gosai has received his Ph. D degree in the field of Computer Science from the same University. He has total teaching experience of Twelve years. He has been awarded the “CAREER AWARD FOR YOUNG TEACHERS” from ALL INDIA COUNCIL FOR TECHNICAL EDUCATION (AICTE) New Delhi with amount of Rs. 10.5 lakhs. He is the author of seven books and over 48 research papers related to computer science and computer networking in International and National Journals and conferences.

Bhargavi H Goswami:

MCA,BCA from Saurashtra University, is giving service as Assistant Professor and pursuing Ph.d Simultaneously under Dr. Atul Gosai. Her research interest includes Computer Networks, Wireless Networks and Congestion Control Mechanism. She has written 4 papers in National and International journals and has teaching experience of 3 years.