An Extensive Analysis of Open Source Wireless Network Simulators

Koushik Banerjee, Hemant Sharma

Research Scholar, ITM University, Gwalior. Research Scholar, ITM University, Gwalior

E-mail: koushik_banerjee21@yahoo.in, sharmahemant1289@gmail.com

Abstract

Now a day it has been observed that most of the researchers are busy in building new simulators or extending the existing simulation tools so that analysis of protocols can be done and conclusions can be drawn whether the protocols is workable in the real world. To evaluate the performance of protocols on real network is too much hectic in terms of cost and complexity. So simulators are needed for reducing cost as well as complexity. Network simulations are widely used by researchers to test their own hypothesis and protocols on networks without implementing in the real world. This approach saves the combination of cost, complexity and time. There are various types of simulators available in market among which some are general purpose simulator others are special purpose simulators. Each simulator has its own pros and cons therefore the researcher uses different simulators or emulators for specific research. In this paper, explanations and comparison of some available network simulators like NS-2, NS-3, GloMoSim, Avrora, COOJA, J-SIM and SENSE on the basis of multiple parameters is presented. The aim of this paper is to identify an effective simulator for research in Wireless sensor network.

1. Introduction

Simulation of a proposed technology is a matter of prime importance, in order to study the impact of proposed work a real environment is required, but testing of the proposed technique in a real environment is too costly and restricted in terms of flexibility, scalability etc.. Simulators provide researchers a platform where they can design, configure, run, analyze and draw a conclusion without implementing their methodology on the real environment. Therefore, researchers can easily draw a conclusion after testing their concerned area of research. The main focus of this research paper is to focus on parameters such as open-source, scalability, platforms, GUI etc. For this we have designed a comparison table which will aid researchers to easily choose a simulator as per their requirement. The network simulators are available in a wide range which can be compared on various parameters specifying the nodes in term of power capacity and the links among the nodes, traffic in the network, specifying even every tiny detail regarding protocols, graphical user interface allows user to visualize the movement of data packets, connectivity etc..

On the basis of simulation result one can easily analyze their methodology.

2. RELATED WORK

Researchers need a profound and analytical study for the selection of a simulator through which they can visualize and simulate their methodology and techniques. Different researchers are busy testing different simulators in terms of routing protocols [1]. Each simulator has its own merits and demerits in terms of cost, scalability, execution processing to evaluate the performance of network on different parameters [2]. There are many available surveys in the field of wireless network simulators. We have categorized them under comparative and descriptive papers. Some of these are provided in table-I as short overview.

<table>
<thead>
<tr>
<th>Ref Number</th>
<th>Type</th>
<th>Simulators</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>comparison</td>
<td>Opnet, NS-2, OMNeT++, SSFNet, QualNet, J-SIM, Totem</td>
</tr>
<tr>
<td>[5]</td>
<td>description</td>
<td>OMNeT++, REAL, NS-2, C++Sim, cnet, SSFNet, CLASS, SMURPH</td>
</tr>
<tr>
<td>[6]</td>
<td>comparison</td>
<td>SSF, SWANS, J-SIM, NCTUns, NS-2, OMNeT++, Ptolemy, ATEMU, Em-Star, SNAP, TOSSIM</td>
</tr>
<tr>
<td>[8]</td>
<td>comparison</td>
<td>NS-2, TOSSIM</td>
</tr>
<tr>
<td>[9]</td>
<td>comparison</td>
<td>NS-2, Avrora, Opnet, GloMoSim</td>
</tr>
<tr>
<td>[10]</td>
<td>comparison</td>
<td>NS-2, cnet, JNS, Opnet, AdventNet, NCTUns</td>
</tr>
</tbody>
</table>

3. SIMULATORS

3.1. NS-2

NS-2[12] is a discrete event network simulator, designed at the University of California, Berkeley, which supports wired and wireless network many network objects such as protocols, applications and traffic source behavior. NS-2 is open source and is a part of the software of the VINT
project that is supported by DARPA since 1995. NS-2 is an object oriented simulator and more popular among researchers. It is written in C++ and it has an OTcl interpreter which serves as a front end. NS-2 supports a class hierarchy (compiled hierarchy) in C++ as well as a very much alike class hierarchy in the OTcl interpreter. Those hierarchies are interrelated to each other. NS-2 uses two languages for doing two different kinds of things. NS-2 provides, detailed simulations of protocols using a systems programming language which provide user to do manipulation of the packet, header and bytes as well as implementation of algorithms that run over huge data. NS-2 have reduced OSI layer model in which the presentation and session layers are omitted. NS-2 offers a wide number of features. Graphs can be easily generated using Xgraph or Tracegraph. It has movement and traffic patterns and an energy model which can be easily generated. However traffic and mobility are typically produced before the actual simulation start and not an integral part of NS-2’s architecture. The main disadvantage of NS-2 is scalability and time taking analysis. Again addition or implementation of any new protocol is complex [11, 13]

3.2. NS-3

NS-3 is also an open source discrete-event network simulator designed primarily for researchers to assist in research and education. It is licensed under the GNU GPLv2 license and freely available. The ns-3 project [14] was started in mid of 2006 and is yet under massive development. NS-3 is prepended as a replacement of NS-2, not an extension [15]. NS-3 does not have an OTcl API. It is written in C++ and python languages. In order to simulate a methodology or technology NS-3 supports C++ and python. NS-3.20 is the latest version of NS-3. NS-3 has enhanced characteristic and inbuilt support for parallel simulation. NS-3 is currently supports only IPv4, which can be considered as a limitation.

3.3. GloMoSIM

GloMoSIM [16] is widely known as Global Mobile Information System Simulator. It is a simulation environment typically used for large scale wireless networks. GloMoSim is based on Parsec. GloMoSiM uses parallel discrete-event simulation [17]. In addition, Parsec compiler is used in GloMoSiM for compiling the simulation of protocols [16]. The node aggregation technique is introduced, which actually became one of its plus points, into GloMoSim to produce significant benefits to the simulation performance. In GloMoSim, each node depicts a geographical area of the simulation. GloMoSim has various choices based on its assets for CSMA MAC protocols, wireless routing protocols and implementations of TCP and UDP. GloMoSiM supports only wireless network protocol.

3.4. Avrora

Avrora, is an open-source, platform independent, accurate simulator for embedded sensing programs developed by UCLA Compilers Group. It is implemented in Java, which helps flexibility and portability. One of the distinguished characters of Avrora is its accuracy and scalability. It is indeed a very powerful simulator for the real hardware platform on which actual and real time sensor programs run. Avrora is operating system as well as language independent. Avrora gives an infrastructure for analytical research as well as a framework which allows static checking of embedded software. Avrora can simulates a network of motes, runs the real time microcontroller and performs actual simulations of the devices [18]. Mica2 and MicaZ are two typical platforms which can be easily emulated and it is also capable of running AVR elf-assembly or binary codes. Avrora is efficient in simulating real sensor network with accuracy. It uses the software stack provided in TinyOS, which allow nodes to communicate via the radio. It provides an extension which help to simulate different sensor network with different orientation and different number of nodes. Developers claim that Avrora can simulate a network of up to ten thousand nodes and its performance is very faster than most of the other simulators [19]. On the other hand clock drift is not modelled here, which is the major drawback.

3.5. COOJA

COOJA [19] is a simulator which runs in the Contiki Operating System [19] and particularly developed for wireless networks which comprises sensors running this particular OS. Instant Contiki [20] (a virtual machine containing Contiki development environment) also utilize this tool in every aspect. First things first, the simulator operates at three layers of abstraction. The top layer is called the networking layer, at which users not only implement various applications but also routing protocols. The program code which was developed and tested on COOJA at the beginning of its formation at the operating system level can run without any modifications directly on Contiki OS. Layer two which is the machine code instruction layer uses MSPSim so that the emulation at a bit level is possible, therefore the platform is able to emulate ESB nodes with TI MSP430 microcontrollers [21]. After simulation, nodes can be parted in different types, both in the software as well as hardware aspects. One more great feature of COOJA that it also adds those sensors which do not use Contiki. The only drawback of this tool is that one single node is simulated entirely as one of the layers. It is seen as a disadvantage, but the other way around, a single network of sensors simulated at particular levels of detail can give a broader look at the structure with an acceptable amount of time. But there are some limitations. COOJA has a lot of calculations and not very much efficient.

3.6. J-SIM

Java Sim or so we called J-SIM [22] is a network simulator and built in accordance to component based software epitome. In J-SIM culture, this is called autonomous component architecture (ACA). J-SIM is actually component based, and yet the compositional simulation environment which had been developed by a
team at the Distributed Real Time Computing Laboratory of the Ohio State University and Illinois University. J-SIM is a network simulator written in Java [22]. Every dot in J-SIM is a component. It is either a node, a link or protocol. Work of each component is depicted using ports. Adversely speaking, there are three possible ways to connect ports which are one to one, one to many, and last but many to many is more frequently used. On a more intend level, J-SIM can be represented in two layers, The Lower layer Core Service Layer (CSL) holds every aspect OSI layer from network to physical where the higher layer represents the rest of the OSI layers [23]. J-SIM is relatively complex (not more than NS-2), inefficient and it has some overheads in intercommunication model.

3.7. SENSE

Sensor Network Simulator and Emulator [24] or SENSE a simulator primarily designed for sensor network simulation. SENSE possesses a template class support which allows the use of different components with different types of data [25]. It can be concluded that SENSE can be classified under component-based simulator for wireless network simulation. SENSE is still under development stage. When trying to use the simulator, we found some issues that were gladly solved by the developers. This simulator provides three user type high level, network designers and component designers. Every component in SENSE communicates via ports this concept free the simulator from dependency and also enables extensibility, reusability and scalability. Extension of components in functionality is possible only if the interface is compatible and no inheritance between components is used. SENSE only supports C++ language with an interface using text, and the results are stored in a text file. This contributes to the efficient use of computational power, but greatly promote adversity in the interface. SENSE is less customizable.

4. Comparison

In this comparative study the focus is given to select a wireless network simulator as per the need so a comparison of simulators on different criteria such as scalability, types of network support, platform, ease of use, etc. Some simulators are general purpose, whereas some are specific to a particular area. Sensor researchers mainly prefer simulators which are designed for WSN areas such as SENSE, COOJA, J-SIM etc. However general purpose simulators are also extending such as Mannasim patch for NS-2 to support sensor network but scholars find learning NS-2 too difficult. The scalability is important if one goes for sensor network as thousands of nodes are deployed in the network, but working on NS-2 do not provide you scalability even GloMoSim does not support GUI for large number of nodes but it is available free. To modify or to test a technology simulators support particular or more than one language so a researcher can go for a simulator for which they have sound knowledge of languages. Table-II provides profound analysis of simulator on a large number of criteria. Abbreviations used in this table are listed below:-

- GUI: Graphical User Interface.
- T/W: Time warp.
- Ad: Ad-Hoc

5. Conclusion

In this paper a comprehensive survey on different network simulators with all background details have been provided the study is done by keeping different parameters in mind and their merits and demerits in particular conditions are also discussed. Those simulators are different, but we can say none are superior or inferior to the other. In order to select a particular simulator for a specific research a sound knowledge of simulators is needed, therefore the authors have given more focus on the selection of specific simulators for particular testing to provide clearer vision regarding Simulators in addition to a short description, a comparative presentation is provided in tabular form which increases more understanding. No particular simulator is universally accepted for all situations, therefore appropriate guidelines are mandatory to choose simulator for a specific research.
Table 2. Result and Comparison among Simulators

<table>
<thead>
<tr>
<th>Sl. No</th>
<th>Tool</th>
<th>Developed By</th>
<th>License</th>
<th>GUI</th>
<th>Scalability</th>
<th>Language</th>
<th>Network Supported</th>
<th>Learning curve</th>
<th>Emulation Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NS-2</td>
<td>University of California, Berkeley</td>
<td>Open source</td>
<td>limited visual aid</td>
<td>Small</td>
<td>C++, OTCL</td>
<td>T/W/A d/WSNA</td>
<td>Steep</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>NS-3</td>
<td>University of Washington, Georgia Institute of Technology</td>
<td>Open source</td>
<td>Better</td>
<td>Large</td>
<td>C++ (core) Python(bindings)</td>
<td>T/W/A d/WSNA</td>
<td>Steep</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>GloMoSiM</td>
<td>Parallel Computing Lab, UCLA</td>
<td>Open source</td>
<td>limited visual aid</td>
<td>Large</td>
<td>Parsc</td>
<td>T/W/A d</td>
<td>Moderate</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Avrora</td>
<td>UCLA Compilers Group</td>
<td>Open source</td>
<td>Originally command-line framework</td>
<td>Very large</td>
<td>Java</td>
<td>Mainly for WSN</td>
<td>Moderate</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>COOJA</td>
<td>Adam Dunkels</td>
<td>Open source</td>
<td>Good</td>
<td>Very large</td>
<td>Java (Simulations in C)</td>
<td>Mainly for WSN</td>
<td>Moderate</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>J-SIM</td>
<td>Ohio State University and Illinois University</td>
<td>Open source</td>
<td>Good visualization</td>
<td>Small</td>
<td>Java/Jacl</td>
<td>T/W/A d/WSNA</td>
<td>Moderate</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>SENSE</td>
<td>Dept. of Computer Science, Rensselaer Polytechnic Institute</td>
<td>Open source</td>
<td>Good(Using G-Sense)</td>
<td>Large</td>
<td>C++</td>
<td>Mainly for WSN</td>
<td>Moderate</td>
<td>Yes</td>
</tr>
</tbody>
</table>

6. References


