Grenze International Journal of Engineering and Technology, January Issue

A Survey of Multicast Network Simulator

Sonia Singhal[1] and Ashwani Kush[2]
[1]Department of Computer Science and Application, Kurukshetra University, Kurukshetra, India
Email: drsoniapnp@gmail.com
[2]IIHS Kurukshetra University, Kurukshetra, India
Email: akush@kuk.ac.in

***Abstract*— A network simulator is a tool that enables users to design, test, and analyze computer networks without having to use physical hardware. They are widely used in various fields such as education, research, network planning, and troubleshooting. This paper discusses two main types of simulators: commercial (like OPNET, QualNet, NetSim), which offer strong support and documentation, and open-source (like NS2, NS3, OMNeT++, J-Sim), which are free and customizable. Simulators include components such as virtual devices, protocols, topologies, and traffic models, and use methods like discrete event or real-time simulation. Popular simulators such as NS-3, OMNeT++, Cisco Packet Tracer, GNS3, and Mininet each serve specific purposes in network testing. Although simulators offer advantages such as cost efficiency, flexibility, and scalability, they also are complicated, require significant hardware, and cannot support newer technologies.**

I. INTRODUCTION

One of the most significant innovations of modern times is simulation. Computer simulations are capable of simulating both real-world and fictional items for analysis. A network simulator is a method for putting a network into practice on a computer. Through this, the network's behavior is determined either by recording and replaying observations from a production network or by employing mathematical formulae to connect network elements. Different types of network simulators can be compared based on the following: range (from very simple to very complicated), specifying the nodes and the links between those nodes and the traffic between the nodes, providing all the protocols used to handle traffic in a network, Text-based applications, which enable more complex forms of customization, graphical applications, which let users see how their simulated environment functions, and programming-oriented tools, which offer a programming framework that can be customized to create an application that mimics the networking environment under test In [1]. A wide range of network simulators are available, each with unique capabilities. OMNeT++, OPNET, NS2, NS3, NetSim, REAL, J-Sim, and QualNet are a few examples of network simulators. Network simulator types can be grouped and described according to certain standards, such as whether they are free or commercial, or whether they are simple or sophisticated.

Certain network simulators are commercial, meaning that the source code for their program or related packages is not freely available to the general public. Every user must pay to obtain a license to use their software or to acquire particular packages based on their own usage needs. The OPNET [OPNET] is a common example. Commercial simulators have both pros and cons. The benefit is that it typically contains updated and comprehensive documentation, which may be regularly maintained by some of the company's qualified employees.

Grenze ID: 01.GIJET.12.1.181

This is a drawback of the opensource network simulator, though, because there are typically not enough experts working on the documentation.
This can become a major issue when the various versions include a lot of new features that make it hard to track down or comprehend the older scripts without the right documentation. In contrast, the opensource network simulator has the benefit of being completely open, allowing any business or individual to contribute and identify any issues.

A. Basic Concept of Simulator

Simulation is a crucial technique in the field of computer and communication networks using mathematical formulas to calculate the interactions between end-hosts, routers, physical links, or packets, or by recording and replaying experimental data from actual networks. Network simulations can be used in conjunction with a variety of applications and services to observe end-to-end or point-to-point performance within the network. Key Components of Network Simulators

- Virtual Devices
- Protocols
- Network Topologies
- Traffic Models
- Metrics Collection

Type of simulator: Different types of network simulators can be categorized (Fig. 1).

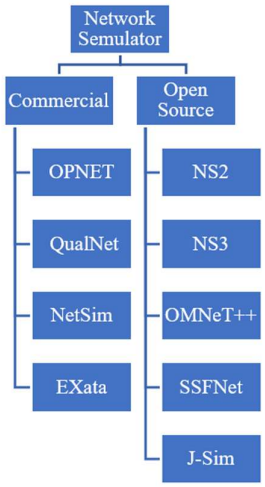

Figure 1. Types of Network Simulator

- Commercial network simulator: Commercial network simulators do not provide the free source code of their software or associated packages to the end users. Licenses for their software must be purchased or specific packages must be ordered for the specific usage requirements of the users.
  - OPNET (Optimized Network Engineering Tools): A tool used to design and test networks by simulating their behavior. It helps analyze network performance and identify ways to improve efficiency without needing physical hardware [2]. It is a vast and robust simulation program that offers numerous of options for simulating whole heterogeneous networks with different protocols [3].
  - QualNet: A simulator designed for large and complex networks. It allows users to test how networks perform and handle different conditions in real-time.
  - NetSim (Network Based Environment for Modelling and Simulation): Software that lets users build and test different types of networks, including wired, wireless, and IoT setups. It's great for learning about networking and solving issues like security problems.
  - EXata: A simulation tool that creates a virtual version of a network. It is used for testing network security and performance in critical areas like defense or aerospace.
- Open-Source network simulator: This type of network simulators allows anyone or organization to contribute and find bugs. The interface can be improved in the future. In addition, it can update quickly with the latest developments of new technologies compared to commercial network simulators.

- NS2 (Network Simulator version2): The Network Simulator 2 (NS2) is the second version of the Network Simulator (NS). It was originally developed in 1989 and evolved over the years to be supported by DARPA today. Many nonprofit organizations contribute to its use in academic research. A discrete event-driven, object-oriented simulation originally developed at UC Berkeley, NS2. C++ is used for protocol implementation and reducing packet/event processing time, while OTcl (Object Tcl) is used to schedule events and control scenarios, modify parameters, and visually assemble components easily. It manages simulation time and coordinates events in a network simulation.

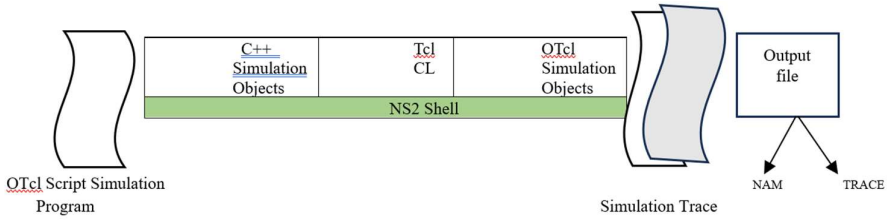

Figure 2. Basic Architecture of NS2

- NS3 (Network Simulator version3): NS3 is not an extension of NS2, but rather a brand-new network simulator. For flexibility, it is written in C++ and has a Python scripting interface. It simulates using lightweight virtual machines. Provides a framework for output statistics and tracing that may be customized without requiring a core rewrite. Active maintainers assist users with testing, validation, bug reporting, and inquiries [4].

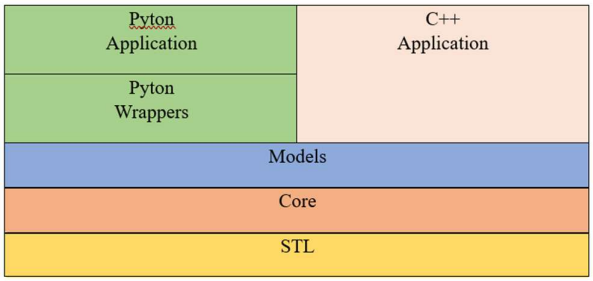

Figure 3. NS-3 architecture

- OMNet++ (Optical Micro-Networks Plus Plus): Communication networks are the main application for the component-based discrete event simulator OMNeT++. It makes it possible to use high-level languages like Python (NS3) or OTcl (NS2) to combine C++-programmed modules into bigger models. In addition to GUI support for simulation running, OMNeT++ provides a command-line interface, simulation kernel library, NED topology editor, and graphical output tools (Plove, Scalars). Because of its modular design, it can be customized to meet different simulation requirements and works with Windows, Linux, and Unix-like systems [5]. With open-source models for IP, IPv6, MPLS, mobility, and ad-hoc networks, OMNeT++ provides a framework for building bespoke simulations rather than a stand-alone simulator. It is extensively utilized in both academia and industry.
- SSFNet: For HS object tracking, a spatial-spectral fusion network with spectral angle awareness (SST-Net) combines the information of the RGB and HS modalities to create a robust fusion feature representation, which improves tracking performance; for hyperspectral object tracking, a spatial-spectral fusion network with spectral angle awareness (SSF-Net); for extracting the more robust spatial and spectral features, a spatial-spectral feature backbone (SSFB); for incorporating the visual information from the RGB modality into the HS modality to create robust HS fusion features; and for predicting the more accurate position of objects, a spectral angle awareness module (SAAM).
- J-Sim (Java-based simulation): With funding from the National Science Foundation (NSF), DARPA's Information Technology Office, the Air Force Office of Scientific Research's Multidisciplinary University Research Initiative, Ohio State University, and the University of Illinois at Urbana-Champaign, the Distributed Real-time Computing Laboratory (DRCL) created the simulation tool known as JSim. The source code for JSim is accessible, and it is free to use [6].

TABLE I. SHOWS THE SIMULATOR NAME ALSO THEIR AVAILABILITY OF SOFTWARE AND THE URL ADDRESS

Simula or Name	License, Available Sites	Language
GloMoSim	Open source https://networksimulationtools.com/glomosim/	C
Ns-2	Open source http://www.isi.edu/nsnam/ns/ns-build.html	OTCL and C++
Ns-3	Open source http://www.nsnam.org/ns-3-13/download/	C++ and optional Python bindings
Omnet++	Academic public license http://www.omnetpp.org/component/docman/cat_view/17-downloads/1-omnet-releases	C++ also JAVA (IDE) and C#.
Opnet	Commercial http://www.opnet.com/university_program/itguru_academic_edition/	C and C++
Netsim	Commercial/Proprietary http://www.ssfnet.org/download/license.html	C and Java
Real	Open source http://www.cs.cornell.edu/skeshav/real/overview.html	C
Qualnet	Commercial also provides separate licenses for academic and other purposes http://www.it.iitb.ac.in/~qualnet/	C++
j-sim	Open source https://sites.google.com/site/jsimofficial/downloads	Java, tcl

II. PURPOSE OF NETWORK SIMULATOR

- Education and Training: Simulators are a convenient, cost-effective way to learn about networking concepts. Students can experiment with different topologies, protocols, and configurations to learn more about how they work. Predefined labs are provided with Cisco Packet Tracer to teach networking and real-world troubleshooting.
- Research and Development: Researchers use simulators as virtual testing grounds to explore how new ideas in technology or network design might work. Imagine developing a new method for managing internet traffic—using a simulator, you can see how well it handles different levels of activity without affecting real-world systems. For example, you could test a new way for data to find the fastest path through a network and observe how it performs under heavy or light traffic. Tools like NS-3 are popular because they let researchers dig into the details of how these systems behave, making them invaluable for academic studies.
- Network Planning: Before building a real network, engineers can use simulators to predict how it will perform. These tools let them experiment with different designs, such as where to place devices or how to route traffic, to find the best setup. By testing various configurations, engineers can ensure the network uses resources efficiently and avoids potential traffic jams or slowdowns.
- Troubleshooting: Simulators mimic real-world network problems, like congestion or hardware failures, in a controlled environment. This is incredibly useful because it allows engineers to investigate and solve complex issues without risking disruptions to actual, live networks.

III. POPULAR NETWORK SIMULATORS

- NS-3: ns-3 is an open-source network simulator widely used in research and education. It is written in C++ and Python, allowing flexibility for users to create, simulate, and analyze various network scenarios [4].
  Key features of ns-3 include:
  Support for wired and wireless networks: Users can model both types of networks, enabling a wide range of simulation scenarios.
  Academic focus: It is extensively used in academia for studying and improving networking protocols, analyzing mobility models, and optimizing network performance.
  Flexibility: With its dual-language support, ns-3 makes it easier for researchers and students to customize simulations and perform advanced analyses.
- OMNeT++: OMNeT++ is a powerful, modular, and extensible simulation framework widely used for research and development in communication networks [5].
  Key features of OMNeT++:
  Modular Design: Its modular architecture allows users to build and reuse simulation components, making it suitable for complex network simulations.
  Extensibility: The framework is highly flexible and can be adapted for a wide range of network types and simulation scenarios.

Academic and Industrial Use: OMNeT++ is extensively used in both academia and industry for studying network behavior, testing protocols, and analyzing communication systems.
Custom Model Development: Users can create custom simulation models using C++, offering precise control over the simulation logic.

- Cisco Packet Tracer: Network simulation software like cisco Packet Tracer is realistic and easy to use, which helps students better understand how networks work. It allows them to create, configure, and test different network setups without needing expensive hardware [1]. By using it, they can practice tasks like setting up local networks (LAN) or wide-area networks (WAN), solving network problems, and designing different network layouts. It allows students to identify and fix common configuration mistakes, like assigning the wrong IP addresses or setting up routing protocols incorrectly [7]. Cisco Packet Tracer (CPT) is also a multi-tasking network simulation software that can be used to perform and analyze various network activities [8]. It is widely used for learning and testing networking concepts, especially for Cisco certifications.
- GNS3: Graphical Network Simulator-3 (GNS3) is an open-source network software emulator that was first made publicly accessible in 2008. It has an easy-to-use graphical interface that makes it possible to quickly create simulated topologies and combine virtual and real devices to simulate complex networks. It uses Dynamics emulation software to simulate Cisco IOS, it also supports devices from other network vendors like Juniper and others. If a network device IOS image is introduced into GNS3 then we may select allocated hardware resources, a number of network interfaces and their type. When the simulated device is added into the topology, we can access it with a ssh remote control. One of GNS3 important advantages is the possibility to connect the simulated network topology to the real network environment. This can be done using the cloud virtual device. We may select there a real or virtual network interface available on PC running GNS3. GNS3 is used by many large companies including Exxon, Walmart, AT&T and NASA, and is also popular while preparing for network professional certification exams.
- Mininet: Mini-Net is a lightweight model designed for segmenting medical images, making it easier to identify specific structures or regions, such as anatomical parts or diseased areas [9]. The model uses an encoder-decoder structure where the encoder (input side) focuses on extracting features, and the decoder (output side) reconstructs the segmented image[10]. Its architecture is built around two key components:

Dual Multi-Residual Block (DMRes): This block helps the model extract and refine features from the input image at multiple scales, ensuring both global context and fine details are captured. The DMRes blocks are central to the encoder, enhancing its ability to process complex details while keeping the model efficient. This is especially important in medical imaging, where accurate and detailed segmentation of structures is critical.
Expand-Squeeze Block: Inspired by modern feature extraction techniques, this block optimizes the way features are processed, improving the model's efficiency.
Mini-Net strikes a balance between identifying broad patterns and preserving small details, making it highly effective for tasks that require precision, such as identifying anatomical features or pathological regions in medical images.

IV. SIMULATION METHOD

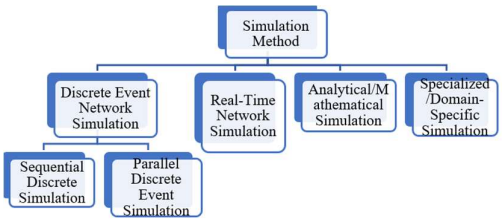

Figure 4. Types of Simulation Method

- Discrete Event Simulation (DES) is a strategy that uses virtual objects and events to simulate real-world systems, such as networks or traffic. Events illustrate how things change or interact, objects depict actual things, and the simulation centers on significant activities throughout time. It is employed to research and enhance systems without really interfering with them. There are two types sequential discrete simulation and parallel discrete event simulation. There are the characteristics of discrete event simulations: events occur at specific times, state changes are the focus of DES, time skipping reduces complexity and resource intensity, queueing theory is heavily used.

- Real-time simulation runs in fixed time steps, mimicking the real system's timing. It must complete calculations within each step to stay synchronized with real time. If calculations take too long, it falls behind, causing a lag called an overrun.
- Analytical/Mathematical Simulators are tools that rely on mathematical models to estimate how a network will perform. Instead of simulating every detail, like individual data packets, they use equations and algorithms to predict overall performance metrics, making them faster and more efficient for certain types of analysis.
- Specialized/Domain-Specific Simulators are designed to focus on particular types of networks or technologies. For example, they might cater specifically to wireless sensor networks, mobile ad hoc networks (MANETs), or Internet of Things (IoT) systems. These simulators are tailored to address the unique characteristics and challenges of these domains, making them highly effective for targeted research and development.

V. Advantages of Network Simulators

- Cost-Effective: Does not require costly physical hardware, making it beneficial for businesses or individuals on limited resources.
- Flexible: To test various situations, quickly alter network topologies, protocols, or environments. Perfect for experimenting with "what-if" analysis.
- Scalable: Designed to simulate networks of any dimension, ranging from a massive ISP-level architecture to a little office LAN.
- Risk-Free Testing: Mimic assaults or interruptions in networks without affecting real systems. Optimizes cybersecurity and disaster recovery methods.

A. Applications

- To maximize coverage, minimize interference, and verify device compatibility, simulate Wi-Fi, 4G/5G, and IoT deployments.
- Examine how new or pre-existing protocols function in a range of situations, such as overbearing latency or congestion.
- Examine the impact of cyberattacks (such as DDoS and man-in-the-middle) and examine the way successfully countermeasures work.
- For massive data centers, improve broadband links, load adjusting, and server deployment.
- For effective operation and safety, model interconnected infrastructures such as traffic lights, automobiles, and essential services.

B. Challenges

- Due to the high computation and storage requirements, realistic simulation sometimes not feasible for big networks.
- The majority of network simulators depict network activity using computational models, which could not accurately reflect actual circumstances.
- Various network simulators, like NS-3, OMNeT++, and OPNET, need a thorough comprehension of protocol design and coding.
- It might be time-consuming to write scripts, set up network configurations, and troubleshoot simulation failures.
- Numerous classical simulation devices were created for traditional networks, such as old IP networks and wired LANs.
- Existing simulators frequently don't directly handle newer technologies like SDN (Software-Defined Networking), cloud-based Networks, IoT (Internet of things), Edge computing, 6G.
- Numerous network simulation tools are freely available endeavors with minimal upkeep and assist.
- Abnormal activity or failures may result from bugs in the simulator's programming.

VI. Conclusion

Network simulators are crucial resources for designing, testing, and researching computer networks in both industry and education without the need for actual hardware. Several well-known simulators were described in this study, including open-source (NS2, NS3, OMNeT++, GNS3) and commercial (OPNET, NetSim).

Commercial simulators are less versatile and more expensive, but they have good features and support. Although open-source technologies are free and adaptable, they might not have the necessary support and documentation. Simulators have some drawbacks, such being complicated, requiring a lot of processing power, and not necessarily supporting emerging technologies like IoT or 6G. However, they also save money, are simple to scale, and enable safe testing. To make them more effective and user-friendly, additional advancements are required. Simulators should improve in the future in terms of intelligence, usability, and compatibility with contemporary network technology.

TABLE II: SHOWS THE SIMULATOR ADVANTAGE AND DISADVANTAGE

Simulator name	Advantages	Disadvantages
GloMoSim	Execution of GloMoSim is faster, primary output file generate statistic file also trace file	hard to use, not updated regularly, limited GUI support
Ns-2	Availability of analysis tool and network, support an extensive scope of protocols in all layers, easy for testing of complex scenarios	not for creating methodologically sound simulations, tracing system is difficult, difficult to understand and analyze the code.
Ns-3	high modularity, more flexible, maintaining responsive mailing list, provide realistic environment with well-organized source code	lack of reliability, limited support for Python in scripting and visualization, scalability limited
Omnet++	performed or executed under graphical user interface, debugging and Tracing is easy, supports MAC protocols as well as some localized protocols, highly modular and very well structured	number of the protocols provided by OMNeT++ is less, framework dependent.
Opnet	Fast discrete event simulation engine, Stratified modeling environment, Custom-make wireless modeling	Memory consuming, GUI operations are complex, Costly, Buggy, old and hard to modify ready-made models, Less usability
Netsim	GUI support, good analysis framework for comparisons between intra and inter-protocol, packet animator for analysis of network.	very costly, single process discrete event simulator
Real	very cheap (basically free) and usually fast to set up and run tests on, Flexible	not compatible, GUI support is very slow.
Qualnet	perform is fast, provides high fidelity, loyal and adaptable	modification is not possible, paid and expensive
j-sim	easy to use and learn, generate new code,	Only executed either as applets or beans

REFERENCES

[1] S. J. Runtuwene, A. N. Abdulgani, O. M. L. Pakan, F. C. G. Pinontoan, D. I. Mangaronda, and T. N. Kambey, "Network Simulation Using Cisco Packet Tracer in Computer Network Learning in Higher Education," J. Syntax Admiration, vol. 5, no. 11, pp. 5099–5106, Nov. 2024, doi: 10.46799/jsa.v5i11.1774.

[2] M. Chen, Y. Miao, and I. Humar, "Introduction to OPNET Network Simulation," in OPNET IoT Simulation, Singapore: Springer Singapore, 2019, pp. 77–153. doi: 10.1007/978-981-32-9170-6_2.

[3] A. Singh Toor and A. K. Jain, "A Survey on Wireless Network Simulators," Bull. Electr. Eng. Informatics, vol. 6, no. 1, pp. 62–69, Mar. 2017, doi: 10.11591/eei.v6i1.568.

[4] L. Campanile, M. Gribaudo, M. Iacono, F. Marulli, and M. Mastroianni, "Computer Network Simulation with ns-3: A Systematic Literature Review," Electronics, vol. 9, no. 2, p. 272, Feb. 2020, doi: 10.3390/electronics9020272.

[5] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G–An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks," IEEE Access, vol. 8, pp. 181176–181191, 2020, doi: 10.1109/ACCESS.2020.3028550.

[6] T. L. Veith, J. E. Kobza, and C. P. Koelling, "Netsim: JavaTM-based simulation for the World Wide Web," Comput. Oper. Res., vol. 26, no. 6, pp. 607–621, May 1999, doi: 10.1016/S0305-0548(98)00039-2.

[7] W. Jiang, Y. Zhan, X. Xiao, and G. Sha, "Network Simulators for Satellite-Terrestrial Integrated Networks: A Survey," IEEE Access, vol. 11, pp. 98269–98292, 2023, doi: 10.1109/ACCESS.2023.3313229.

[8] J. Byrne, C. Heavey, and P. J. Byrne, "A review of Web-based simulation and supporting tools," Simul. Model. Pract. Theory, vol. 18, no. 3, pp. 253–276, Mar. 2010, doi: 10.1016/j.simpat.2009.09.013.

[9] F. Keti and S. Askar, "Emulation of Software Defined Networks Using Mininet in Different Simulation Environments," in 2015 6th International Conference on Intelligent Systems, Modelling and Simulation, IEEE, Feb. 2015, pp. 205–210. doi: 10.1109/ISMS.2015.46.

[10] S. Lee, J. Ali, and B. Roh, "Performance Comparison of Software Defined Networking Simulators for Tactical Network: Mininet vs. OPNET," in 2019 International Conference on Computing, Networking and Communications (ICNC), IEEE, Feb. 2019, pp. 197–202. doi: 10.1109/ICCNC.2019.8685572.