An Approach to Evaluate Network Simulators: An Experience with Packet Tracer

Michel Bakni, Yudith Cardinale, Luis Manuel Moreno

Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 5, No. 1, pp. 29-36, Junio 2018
Fecha recepción: 21/05/2018, Fecha aceptación: 25/07/2018

An Approach to Evaluate Network Simulators: An Experience with Packet Tracer

Michel Bakni[1], Yudith Cardinale[2], Luis Manuel Moreno[3]
bakni@estia.fr, ycardinale@usb.ve, 13-10934@usb.ve

[1] Ecole Suprieure des Technologies Industrielles Avances, ESTIA, Bidart, France
[2] Departamento de Computación, Universidad Simón Bolívar, Caracas, Venezuela
[2] Coordinación de Telecomunicaciones, Universidad Simón Bolívar, Caracas, Venezuela

Abstract: Besides the measurement and the mathematical analysis, network simulation is a widespread methodology that is used to study computer systems and display their different aspects. However, a simulator is only an approximate model of the desired setting, leading to the need of establishing guidelines that support researchers in the tasks of selecting and customizing a simulator to suit their preferences and needs. In this paper, we propose a simple approach, based on a set of criteria for the evaluation and selection of network simulators, the implementation of the approach leads to results that are measurable and comparable. Then, the proposed approach was put to the test on Packet Tracer. The obtained results give a comprehensive overview of the simulator's features, its advantages, and disadvantages. This paper does not propose a method for selecting the best simulator, but it provides researchers with an evaluation tool that can be used to describe and compare network simulators in order to select the most appropriate one for a given scenario.

Keywords: Network Simulators; Evaluation Criteria; Comparison Approach; Packet Tracer.

I. Introduction

Network simulation is one of the most powerful and predominant evaluation methodologies in the area of computer networks. It is widely used for the development of new communication architectures and network protocols, as well as for verifying, managing, and predicting their behavior. Network simulators have grown in maturity since they first appeared and they have become an essential tool of the research domain, for both wired [1] and wireless networks [2].

Simulators are easy to control, save efforts in terms of time and cost, and allow repeating the same experiment with input changes. However, they are only an approximate model of the desired setting. Although the simulator is capable of simulating the whole network model, it is not possible to cover all of its aspects with the same level of details. Instead, the simulator focuses on one or two of the following aspects [3]: algorithms, application protocols, network protocols, and hardware. Then, the simulator fills the gaps in the other aspects using assumptions [4]. Hence, more studies are needed to establish guidelines that support researchers in the tasks of selecting and customizing a simulator to suit their preferences and needs.

A set of simulator design principles [5] were developed in the mid and late nineties. They meant to address main performance topics such as the balance issues between the execution time and accuracy, and how this issues can affect the simulation of nodes, links, and loads. In addition to that, simulators developers emphasize the concept of Validation, Verification, and Accreditation (VVA) [6], which was adopted by United States Department of defense as a principle method for modeling and simulation [7]. These principles and concepts described the design of simulators accurately, they led to the emergence of dozens of simulators in the next few years. Nowadays, there is a considerable number of simulations tools, in the market and freely available, that are distinguished, either for quality characteristics, such as accuracy, speed, ease of use, and monetary expense or by the capacity of modeling. As a result, selecting the simulator, that fits a given scenario the most, becomes a complex problem due to the variety of parameters to consider and the lack of clear methodology to follow.

One of the main motivation of this paper is to address this lack of guidelines. We propose a simple approach, based on a set of criteria to cover aspects related to the simulation process, as well as aspects related to the evaluation of the network simulator. Our criteria include ten items that can be applied to different network simulators in order to obtain a measurable and comparable assessment. We do not pretend that our approach is a methodology that identifies the best network simulator, as there are varieties of parameters and different possible network scenarios to adequately address that. Instead, this paper demonstrates how the suitability of simulators can be validated for particular needs, following an approach comprised by simple steps and based on a set of

criteria.

To illustrate the applicability of our proposed approach, we evaluate Packet Tracer, it is a simulation tool for both wired and wireless networks. Moreover, it can be used to build complex topologies that simultaneously run different protocols, thus, it is a powerful tool to implement complex and inter-protocols scenarios [8]. Packet Tracer allows the simulation of Cisco's IOS with a high degree of accuracy. It also allows simulating other information systems, such as servers and terminals, as well as some concepts of Internet Of Things (IoT), but with a high level of abstractions, [9]. The simulator has an attractive customizable graphical user interface (GUI) and allows contribution for multi-users activities [10].

The initial studies that mentioned Packet Tracer were limited to the educational aspects, but with the development of the simulator, new studies start taking place in many technical fields. However, there is no specific or comprehensive study for the simulator itself or for its features. In total, the simulator is not a considerable option for researchers in network domains. Hence, we demonstrate that if it is properly evaluated, it becomes an available option for researchers to pursue in their studies.

In summary the contribution of this work is twofold:

1) Propose an approach and a set of criteria to evaluate network simulators; and
2) Evaluate Packet Tracer features, performance, advantages, and disadvantages based on the criteria previously proposed, to show its suitability for researchers in network domains.

The remainder of this paper is organized as follows. In Section II, we survey recent works focused on proposing criteria or methodologies to evaluate network simulators and studies that have evaluated Packet Tracer simulator. Our proposed approach is described in Section III. How the methodology works, is illustrated in Section IV by evaluating Packet Tracer. We draw some recommendations based on the results. Finally, Section V highlights conclusions and perspectives.

II. Related Work

In this section we present two separate surveys, first, we survey studies focused on proposing methods and criteria to evaluate network simulators, and then we provide works that have evaluated Packet Tracer. We highlight their limitations and differences compared with our proposal.

A. Network Simulators Evaluation

The Virtual InterNetwork Testbed (VINT) Project [11] intended to develop methods and tools to address the scale and heterogeneity of the Internet protocols. One important result of the work was adding definitions related to the simulation issues, including the type and the nature of simulators, in addition to highlighting different interactions of the simulated protocols. In [12], there was another attempt to address the issues that concern the simulators developers concluded that there are four of them, namely the type of problem, the level of abstraction, the extensibility, and the diagnosis of existing codes. Later, a detailed and comprehensive study [13] recognized modeling as a foundation stone in the choices of simulators.

In [14], nine evaluation criteria are proposed to evaluate wireless sensor networks. Some of them have been incorporated in our set of criteria. Some other works propose the evaluation of simulators in terms of computational run time, memory usage, and scalability [3][15][16][17][18].

Even though these works propose some aspects that should be taken into account to evaluate simulators, none of them propose a coherent and complete method to do the evaluation, neither evaluate Packet Tracer, as we do.

B. Packet Tracer Evaluation

A variety of studies evaluated one or more different aspects of Packet Tracer. Authors in [19] used Packet Tracer, as well as another network simulator called GNS, to study the traffic in networks that support both IPv4 and IPv6, either using the dual stack technique or the tunneling. As a result, the article concludes that Packet Tracer is "easy to use", but it does not simulate all services and functions like tunneling. On the other hand, in [20], the problem of support for tunneling in Packet Tracer was addressed. In their study, GRE tunnels were properly simulated in addition to many IPsec features. This is a good example of the problem of lack of comprehensive studies. In fact, the tunneling feature was supported since the version 5.3, which was released in 2010.

In [21], a detailed study of the dynamic routing used Packet Tracer as a simulator. Four routing protocols were evaluated, they are Routing Information Protocol (RIP) (version 1 and 2), Open shortest path first (OSPF), and Enhanced Interior Gateway Routing Protocol (EIGRP). The article does not highlight on the simulator itself, thus, the simulation results were presented and discussed based on only the technical side of the network. A similar study that covers only RIPv2 and EIGRP can also be found in [22].

In [23], a performance study is presented based on a scenario implemented using Packet Tracer, The scenario covers both IPv4 and IPv6 networks. The study focuses on the delay, routing traffic, and convergence when OSPF and EIGRP are used. In the end, the authors concluded that Packet Tracer is a useful tool for routing studies, especially to select a routing protocol and to design the optimal routing topology based on that.

A comprehensive study of the Link Layer technologies and protocols can be found in [24]. Trunk ports, static Virtual Local Area Networks (VLANs), Dynamic VLANs, Inter-Switch Link (ISL), and IEEE 802.1Q were tested and verified. In addition to that, the authors implement a scenario using both OSPF as a routing protocol, Dynamic Host Configuration Protocol (DHCP) as a client/service protocol, and access lists as a security application. Packet Tracer was able to simulate the network and trace the packets when different-layers protocols were simultaneously used.

In [25], the use of the Packet Tracer as an assessment tool is discussed. The application has an advantage that it allows the user to stop the simulation at a given moment and check all the messages exchanged among different network nodes. The author concludes that although the simulator was not primarily designed as an assessment and measurement tool, it can use to aid certain educational purpose. The use of the Packet Tracer as an assessment tool is related to the nature of the study, while it does not appear to be used in performance studies, such a tool can add a benefit in the studies of the routing protocols.

Finally, there is a comparison study that mentioned 12 comparative items between GNS3 and Packet Tracer [26]. The items are: the GUI design, the memory requirement, the hardware models supported, the protocol supported, the commands supported, the computer systems supported, the ability to analyse traffic, the ability to exchange the topology, the types of connection supported, the certifications that use the simulators, the license, and the support for the instructor. Although [26] covers many aspects of Packet Tracer, it addresses the aspects from the comparison point of view, without considering the simulator's own capabilities or its maximum limits. In addition, this study does not include items for the performance of the simulator. Instead, it only mentions the minimum memory requirements. Finally, the authors insist on providing results rather than developing a coherent methodology, that, in turn, makes it intended for students and teachers more than researchers.

There are other studies that are interested in the simulator as an e-learning tool [27][28], but that is out of the scope of this article.

III. Evaluation Approach and Criteria

In this section, we explain how we address the problem of evaluating network simulators. First, we describe the proposed evaluation approach, then, we provide, in detail, a list of ten criteria to be used as measurements for the evaluation.

A. Evaluation Approach

As far as we know, there is no fixed approach or methodology to evaluate network simulators. As long as the developing of simulators continues, any methodology will remain subject to modernization and modification [29]. Thus, we do not pretend to establish a methodology, instead, we propose a single approach based on few steps and a set of criteria to demonstrate how the suitability of simulators can be validated for particular needs. The primary objective of the development of this approach is to evaluate qualitative aspects, as well as to obtain measurable or comparable values after applying the approach to a network simulator to describe its behavior, capacity, and performance.

Hence, to evaluate simulators, we propose to follow the following steps:

1) Establish a set of criteria. The evaluation of the simulator requires clear and accurate criteria to assess the different aspects of the simulator. Qualitative criteria can be described by a word or number, while quantitative criteria need to be measured. Moreover, there can exit composite parameters, that are composed of multiple sub-parameters. In the next section, we provide precise and specific definitions of ten parameters that describe and evaluate simulators from different qualitative and quantitative aspects.
2) Establish the experiment setup. It is worthy to install the selected simulator(s) on different systems (e.g., Windows, Linux, MacOS) under the same architecture. The way that operating systems manage system resources and the produced overhead have an important impact on the behavior of applications.
3) Evaluate the qualitative criteria of the simulator(s). Revise the available documentation of simulator(s) and elaborate a table highlighting their characteristics.
4) Design a test scenario to evaluate the measurable criteria. Decide the network elements that will be simulated according to the protocols that are intended to evaluate. Define the number and type of experiments, as well as the time of the simulation, taking into account the criteria to be evaluated.
5) Evaluate the measurable criteria of the simulator(s) by executing the designed experiments. Elaborate tables and graphics to show the results in order to facilitate the analysis and comparison (if there is a case).
6) Elaborate a discussion by analyzing the results.

These steps can be applied to evaluate a single simulator or to compare several of them.

B. Criteria

The following parameters will be used to evaluate the simulator, a detailed and precise definition is provided for each of them.

1) **Nature of the software:** The simulation consists of a number of models that are executed to interact with each other. The nature of the simulation is an assessment of how the simulation is performed. Precisely, the use of the word *simulation* means that the entire process is programmed, it is a software. But if the word *emulation* is used, the hardware is involved in the process [30].
2) **Type of the software:** It is a characterization of the philosophy underlying the simulator's work. Network simulations are based on two philosophies, a simulator is either a discrete-event simulator or trace-driven one. In the first, an initial set of events is generated, it represents the initial conditions. Those conditions, in turn, generate another set of events, the process continues like that, until the end of the simulation.
   In the trace-driven simulation, all events to be simulated are added to the simulator in the form of inputs, thus, it can simulate it and trace the outputs [31].
3) **License:** An evaluation of the capability to use the simulator from a legal aspect, simulators can be private property or they can be developed under a free or public agreement.

4) **User interface:** An evaluation of how can a user interact with the simulator, this includes two aspects:
   - Graphical User Interface (GUI): an evaluation of the support for the graphic interface. Is it an integral part of the simulator? what are the level of details it can show ?
   - Supported programing languages.
5) **Supported platforms:** It is the characterization of the usability of the simulators source code on different platforms and operating systems [32].
6) **Heterogeneity:** An evaluation of the ability to simulate heterogeneous systems where different types of nodes can exist in the same scenario [33].
7) **Modeling:** An evaluation of the ability to modify existing models or to implement and test new ones.
8) **Level of details:** An evaluation of the level of aspects that are being simulated. Those aspects, sorted in descending order, are: abstract algorithms, high level protocols, low-level protocols, and hardware. The lower the level, the less the assumptions and the more the constraints [14].
9) **Supported technology and protocols:** In order to evaluate the support provided for the protocols, TCP/IP model is used [34]. It is a 4-layer stack model, that classifies the network protocols, features, and services according to the function. Starting from the top, these layers are: application, transport, Internet, and link layers.
   We have excluded the routing protocols from this stack and combined them into a single item. The reason behind this is the distribution of the routing protocols in the layers of the model, this does not serve the primary purpose of this item, namely the assessment of support to the protocols.
10) **Performance:** The main purpose of the study of performance is to provide a general idea of the effectiveness of the simulator in terms of implementation time and the consumption of available resources. However, the proposed approach includes three factors for the performance study:
    - CPU Utilization: it is a measure of the application performance [35], it is the percentage of time spent performing the applications processes of the total processing time [36], i.e., the percentage of the processor cycles that are consumed by the applications processes.
    - Execution time: it is the time needed to complete a simulated scenario; measured in seconds.
    - Memory usage: it is the amount of memory used by the application, measured in bytes.

In the next section, we apply the approach to evaluate Packet Tracer.

IV. Applying the Approach

This section is dedicated to the practical aspect, in which we apply the proposed approach to evaluate Packet Tracer. In the following, we describe how the proposed steps and set of evaluation criteria are considered to evaluate Packet Tracer simulator. At the end, we discuss about the suitability of our proposed approach.

A. Step 1: Establish a Set of Criteria

Following the proposed approach leads to a 10-items description for the simulator. The considered set of evaluation criteria is the one presented in Section III-B.

B. Step 2: Establish the Experiment Setup

In order to apply the proposed criteria, we installed the simulator on two different systems, namely Linux Ubuntu 16.04 LTS and Microsoft Windows 10 version 10.0.14393. Both were installed on the same computer with the following characteristics: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz with 16 GB for the RAM, 915 GB of the hard disk is allocated for Linux while 909 GB is allocated for Windows.

C. Step 3: Evaluate the Qualitative Criteria

After the installation, nine of the evaluation criteria can be pointed out, according to the documentation and general knowledge about Packet Tracer. Only the performance criterion requires special scenario preparation. Table I shows the result of this step.

Some of the information presented in Table I was directly obtained from the official website of Packet Tracer, such as supported platforms. Others, like the supported technologies and protocols, required running the simulator to test and verify whether the support exists.

D. Step 4: Design the Test Scenario

We designed a scenario involving several experiments, in which we used the Spanning Tree Protocol (STP) to measure performance determinants. Originally, the STP is used in a layer 2 switched environment to create a loop-free path to data traffic. By default, the protocol convergence time is between 20 to 55 seconds. Several factors can affect the exact value, including the network complexity and the timers values. To consider that, we established the duration of each experiment in 60 seconds, while the convergence time is the time needed for the protocol to converge.

The scenario is built in a way that reflects the CPU utilization and memory usage. To achieve that, we adopted a meshed topology, whose size is increasing exponentially every time we are repeating the test. The basic component of the topology consists of four 2960 Cisco Catalyst switches arranged in a ring topology. Figure 1(a) shows the ring topology of the basic component, which is the scenario of the first test. Then, the second test is done with two basic components, i.e., eight switches, as shown in Figure 1(b). The third one is composed by four basic components, with 16 switches (see Figure 1(c)), and so on increasing the number of basic components exponentially with base 2, until 64 basic components, with 256 switches. In total, we conducted seven tests with 1, 2, 4, 8, 16, 32, and 64 basic components, on each system (Linux and Windows).

Table I: Nine Qualitative Criteria of Packet Tracer

Criteria	Packet Tracer Characteristic
Nature of the software	Simulator
Type of the software	Discrete-event
License	Proprietary, but an End User License Agreement (EULA) exists
User Interface	**GUI:** Yes a built-in GUI interface is supported, with a possibility to trace and store all events. Different languages are supported for the GUI including: English, Russian, German, Portuguese, Spanish and French. **Supported programing language**: Non, it is private property, but scripting is allowed using the Cisco IOS Syntax.
Platform	Linux, Android 4.1+, iOS 8+ and Microsoft Windows.
Heterogeneity	It is supported, different types of real **routers**, such as: Cisco 1941, Cisco 2901, Cisco 2911, and others are supported, as well as different types of real **switches** like: Cisco Catalyst 2950, Cisco Catalyst 2960, Cisco Catalyst 3560-24PS are supported. In addition to that, Linksys WRT300N **wireless router**, Cisco 2504 **wireless controller**, and Cisco Aironet 3700 **access point** are supported. Cisco ASA 5505 **firewall** is supported as well. Variety of IoT devices are supported.
Modeling	It is not supported.
Level of details	Packet level.
Supported technology and protocols	**Application Layer:** Protocols: DHCP, DHCPv6, FTP, HTTP, HTTPS, RADIUS, POP3, SMTP, SNMP, SSH, Telnet, TACACS. Technology: Access Lists, DNS, IoT, IoT TCP, SYSLOG. **Transport Layer:** Protocols: SCCP, TCP, UDP. **Network Layer:** Protocols: ARP, CAPWAP, HSRP, HSRPv6, ICMP, ICMPv6, IP, IPv6, NDP. Technology: IPSec, Cisco NetFlow. **Link Layer**: Protocols: Bluetooth, CDP, CTP, H.323, LACP, LLDP, PAgP, STP, USB, VTP. **Routing Protocols:** BGP, EIGRP, EIGRPv6, OSPF, OSPFv6, RIP, RIPng,

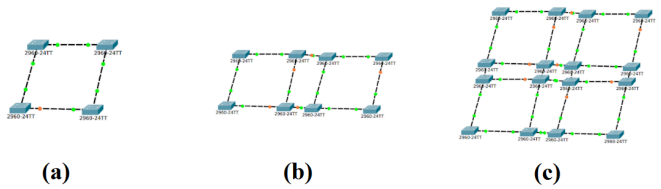

Figure 1: The Different Topologies Used in the Suggested Scenario, (a) A Basic Component Topology, (b) Two Basic Components Topology, (c) Four Basic Components Topology

E. Step 5: Evaluate the Measurable Criteria

Information related to nine of the ten evaluation criteria are shown in Table I, representing the qualitative criteria. The scenario depicted in the previous subsection, was designed to evaluate the performance in terms of CPU utilization, memory usage, and converge time (i.e., the time in which STP converges), which are measurable criteria.

To obtain the performance values in Linux, we used Monit[1], an open source tool for monitoring processes on Unix systems. For the tests in Windows, values were obtained from Task Manager, a built-in monitor of the CPU utilization and memory usage per process.

Figure 2 shows the results of all tests for CPU utilization, when the suggested scenario is implemented on Linux. Figure 3 shows the results for the same tests, when running the scenario on Windows. In both cases, we registered the percentage of CPU utilization every second during the simulation. Comparing both results tells that Windows is more suitable for the simulator in terms of CPU utilization.

Figure 4 displays a comparison of the memory usage for the same previous tests, for both operating system. We measured the percentage of memory usage of Packet Tracer at the beginning of each simulation test, i.e., the memory consumption is constant during the execution, there is no change.

[1]https://mmonit.com/monit

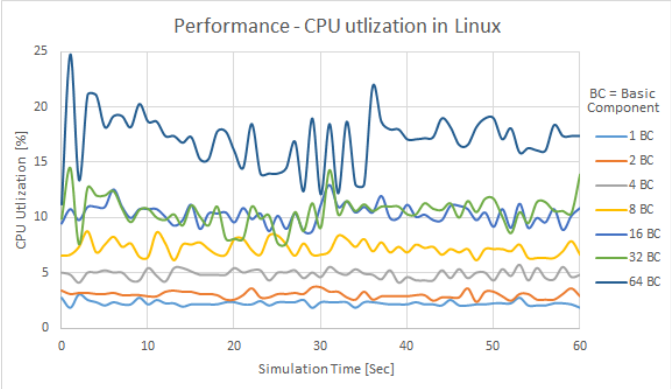

Figure 2: CPU Utilization - Linux

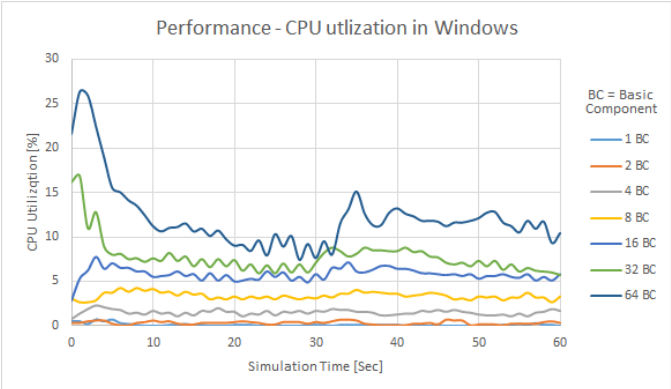

Figure 3: CPU Utilization - Windows

Since Packet Tracer is a discrete-event simulator, it generates a subsequence of events that are gathered in a buffer list, this buffer is overflowed when the number of the basic components is more than eight, We encountered the same problem both on Windows and Linux. Thus, it was not possible to obtain the convergence time of STP from tests whose topologies have more than eight basic components. However, Figure 5 shows the obtained results for the convergence time. As we note in Figure 5, results are similar in both Windows and Linux when

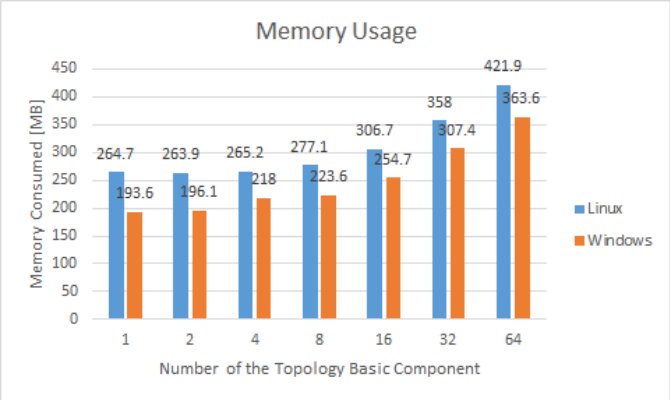

Figure 4: Memory Usage

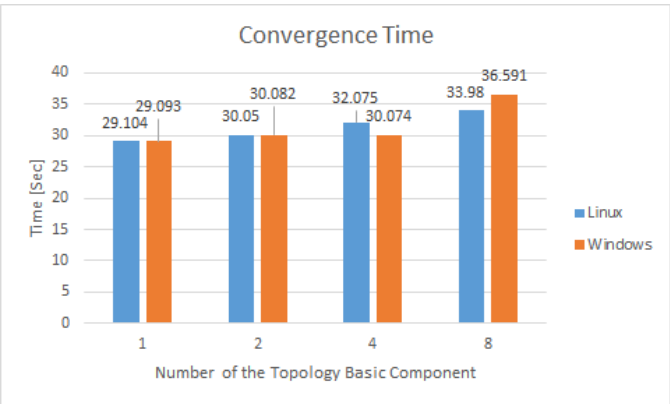

Figure 5: Converge Time

there are eight basic components or less.

F. Step 6: Elaborate a Discussion

In this section we present the analysis and discussion of the evaluation of Packet Tracer, derived from the obtained results.

Analyzing the qualitative criteria in Table I, we can say that Packet Tracer supports a wide variety of protocol in each layer, this gives the researchers multiple choices to create different scenarios. In addition to that, it provides the same GUI and functions on both Windows and Linux platforms.

We follow with the analysis of the measurable criteria. Regarding the CPU utilization, from Figure 2 and Figure 3, we can note that Packet Tracer behaves better in Windows than in Linux. In the beginning, we note that the CPU utilization for the topology that has one or two components is always around 1% for the tests in Windows, while the value rises to around 3% for the same tests done in Linux. For the topology of 16 basic components, the CPU utilization ratio is between 5-6% for Windows and 11-12% for Linux. Finally, the values of 64 basic components curve show a higher variance in the ratio more than all other curves, whether the tests were done in Windows or Linux. Results for all tests in Windows show a regular behavior, i.e., the more complex the topologies become, the higher the utilization of the CPU will be. The Linux results follow the same rule, except when the topology has 32 basic component, then, the behavior is irregular compared to the predictable one.

Concerning the memory usage, from Figure 4 we note that the change of values for memory usage shows an exponential form, both for the values obtained from Windows and Linux, this reflects the exponential change in the number of basic components as we change the input topology. In short, the memory used is directly related to the size of the topology in the scenario. Finally, in all the tests, the memory used by the simulator installed on Windows was better managed than the simulator installed on Linux.

From the evaluation of performance results, it was possible to detect one of the simulator disadvantages: the limit size of the events' buffer. Both, on Windows and Linux there was no way to have the exact converge time when the number of the basic components in the topology is more than eight.

In terms of performance characteristics, on one hand, the simulator uses the hardware more efficiently in Windows environment than in Linux, both on CPU utilization and memory usage parameters. On the other hand, the numbers of the execution time do not give us a clear image because it is limited to only 8 basic components.

Finally, even though Packet Tracer is a private simulator, its available version is good enough for simulating complex topologies from both Wide Area Network (WAN) and Local Area Network (LAN) aspects. It does not allow researchers to test new protocols or algorithms, but, it provides a massive set of protocols that can be used to create a large number of combinations of layered-protocols stacks.

G. Reflections about the Approach

The application of our proposed approach to evaluating Packet Tracer simulator, allow us to point out some reflections:

- A layered-protocols stack model is a powerful tool for categorizing the work done in the network by function, but there are some protocols that do not fit into a particular layer, that is because they perform functions belonging to more than one layer at the same time, examples of those protocols are the Neighbor Discovery Protocol (NDP) and Address Resolution Protocol (ARP), they both work on the Internet and Link layers, in this case, we categorize them in the upper layer, which is the network. Merging the technologies and protocols inside one criterion can become a complex issue if the simulator supports technologies that use more than one protocol, in different layers, this, in turn, will lead to a non-comparative item. In this case, it is better to separate technologies from protocols and by creating a new criterion. Then, the technologies item can have its own independent stack-layered model.
- The heterogeneity criterion needs to be described in more details, sub-criteria can be added based on further studies, the main goal is to enable the item to describe the simulator's ability to emulate different specific models or hardware.

- The study of performance characteristics shows that the simulator in Windows handles the hardware better than Linux in term of CPU utilization and memory usage, it is not possible to say that installing the simulator in Windows is better than Linux because there are other aspects of performance that have not been tested in this approach, such as scalability, which highlights different performance parameters that provides a more comprehensive view of The number of nodes that the simulator can simulate.
- We thought about adding a special criterion for the simulator version because it is an important piece of information, but it is related to each simulator itself, thus, it is not comparable among other simulators, that is why we did not consider the version as an item within the suggested approach.
- We are thinking of expanding the approach to include Wireless Sensor Network (WSN) simulators, but this requires further studies to modify the current approach or even developing an independent one. WSN requirements are different from those of wired ones. For example, mobility, environmental, energy consumption, energy harvesting, battery models, and others are specific-purpose concepts that are directly related to the nature of the WSN.

V. Conclusions

In this paper, we have addressed the difficulty of selecting a computer network simulator to fit a given scenario. To achieve that, we proposed an approach of ten criteria that can be applied to the simulator to describe it in a measurable and comparable manner.

In order to test how efficient the suggested approach is, we apply it on Cisco Packet Trace, which is a general-purpose network simulator. The application of the approach proved that it does not only highlight general aspects of the simulator's behavior but it showed its disadvantages as well.

In a future study, we plan to apply the approach to compare several network simulators and include other measurable criteria, such as scalability. We also are working on extending the proposed approach to consider Wireless Sensors Network (WSN) simulators, by involving special items describing the determinants of these networks, such as power constraints, models for energy consumption, and power harvesting.

References

[1] S. Naicken, A. Basu, B. Livingston, and S. Rodhetbhai, *A Survey of Peer-to-peer Network Simulators*, in proceedings of the Seventh Annual Postgraduate Symposium, vol. 2, 2006.

[2] M. Imran, A. Md Said, and H. Hasbullah, *A Survey of Simulators, Emulators and Testbeds for Wireless Sensor Networks*, in proceedings of the International Symposium in Information Technology (ITSim), vol. 2, pp. 897–902, 2010.

[3] H. Sundani, H. Li, V. Devabhaktuni, M. Alam, and P. Bhattacharya, *Wireless Sensor Network Simulators a Survey and Comparisons*, International Journal of Computer Networks, vol. 2, no. 5, pp. 249–265, 2011.

[4] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott, *Experimental Evaluation of Wireless Simulation Assumptions*, in proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 78–82, 2004.

[5] L. Brakmo and L. Peterson, *Experiences with Network Simulation*, ACM SIGMETRICS Performance Evaluation Review, vol. 24, no. 1, pp. 80–90, 1996.

[6] O. Balci, *Validation, Verification, and Testing Techniques throughout the Life Cycle of a Simulation Study*, Annals of Operations Research, vol. 53, no. 1, pp. 121–173, 1994.

[7] J. Heidemann, K. Mills, and S. Kumar, *Expanding Confidence in Network Simulations*, IEEE Network, vol. 15, no. 5, pp. 58–63, 2001.

[8] A. Jesin, *Packet Tracer Network Simulator*, https://docente.ifrn.edu.br/rodrigotertulino/livros/packet-tracer-network-simulator.

[9] Cisco networking Academy, *Introduction to Packet Tracer*, https://www.netacad.com/courses/intro-packet-tracer.

[10] A. Smith and C. Bluck, *Multiuser Collaborative Practical Learning using Packet Tracer*, in proceedings of the Sixth International Conference on Networking and Services (ICNS), pp. 356–362, 2010.

[11] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, P. Haldar, M. Handley, A. Helmy, J. Heidemann, P. Huang, et al., *Virtual Internetwork Testbed: Status and Research Agenda*, Technical Report 98-678, University of Southern California, 1998.

[12] H.-Y. Tyan, *Design, Realization and Evaluation of a Component-based Compositional Software Architecture for Network Simulation*, Ph.D. Thesis, The Ohio State University, 2002.

[13] K. Wehrle, M. Günes, and J. Gross, *Modeling and Tools for Network Simulation*, Sringer Science & Business Media, 2010.

[14] M. Jevtić, N. Zogović, and G. Dimić, *Evaluation of Wireless Sensor Network Simulators*, in proceedings of the 17th Telecommunications Forum (TELFOR), pp. 1303–1306, 2009.

[15] A. Khan, S. Bilal, and M. Othman, *A Performance Comparison of Open Source Network Simulators for Wireless Networks*, in proceedings of the 2012 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), pp. 34–38, 2012.

[16] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus, *Comparative Study of Wireless Network Simulators*, in proceedings of the Seventh International Conference on Networking 2008 (ICN 2008), pp. 517–523, 2008.

[17] G. Lucio, M. Paredes-Farrera, E. Jammeh, M. Fleury, and M. Reed, *OPNET Modeler and ns-2: Comparing the Accuracy of Network Simulators for Packet-level Analysis using a Network Testbed*, WSEAS Transactions on Computers, vol. 2, no. 3, pp. 700–707, 2003.

[18] E. Weingartner, H. Vom Lehn, and K. Wehrle, *A Performance Comparison of Recent Network Simulators*, in proceedings of the IEEE 2009 International Conference on Communications (ICC 2009), pp. 1–5, 2009.

[19] L. Valle-Rosado, L. Narváez-Díaz, C. González-Segura, and V. Chi-Pech, *Design and Simulation of an IPv6 Network Using Two Transition Mechanisms*, International Journal of Computer Science Issues (IJCSI), vol. 9, no. 6, pp. 60–65, 2012.

[20] C. Wang and J.-Y. Chen, *Implementation of GRE Over IPsec VPN Enterprise Network Based on Cisco Packet Tracer*, in proceedings of the 2nd International Conference on Soft Computing in Information Communication Technology, pp. 142–146, 2014.

[21] C. Archana, *Analysis of RIPv2, OSPF, EIGRP Configuration on Router Using Cisco Packet Tracer*, International Journal of Engineering Science and Innovative Technology (IJESIT), vol. 4, no. 2, pp. 215–222, 2015.

[22] K. Dangwal and V. Kumar, *Comparative Study of EIGRP and RIP using Cisco Packet Tracer*, International Journal of Engineering Sciences & Emerging Technologies, vol. 6604, no. 6, pp. 475–480, 2014.

[23] C. Wijaya, *Performance Analysis of Dynamic Routing Protocol EIGRP and OSPF in IPv4 and IPv6 Network*, in proceedings of the First International Conference on Informatics and Computational Intelligence (ICI), pp. 355–360, 2011.

[24] N. H. Prasad, B. K. Reddy, B. Amarnath, and M. Puthanial, *InterVLAN Routing and Various Configurations on VLAN in a Network using Cisco Packet Tracer*, International Journal for Innovative Research in Science and Technology, vol. 2, no. 11, pp. 749–758, 2016.

[25] D. Frezzo, J. Behrens, R. Mislevy, P. West, and K. DiCerbo, *Psychometric and Evidentiary Approaches to Simulation Assessment in Packet Tracer software*, in proceedings of the Fifth International Conference on Networking and Services (ICNS), pp. 555–560, 2009.

[26] T.-S. Chou, S. Baker, and M. Vega-Herrera, *A Comparison of Network Simulation and Emulation Virtualization Tools*, in proceedings of the Annual Conference and Exposition of the American Society for Engineering Education, 2016.

[27] J. Janitor, F. Jakab, and K. Kniewald, *Visual Learning Tools for Teaching/Learning Computer Networks: Cisco Networking Academy and Packet Tracer*, in proceedings of the Sixth International Conference on Networking and Services (ICNS), pp. 351–355, 2010.

[28] W. Makasiranondh, P. Maj, and D. Veal, *Pedagogical Evaluation of Simulation Tools Usage in Network Technology Education*, World Transactions on Engineering and Technology Education, vol. 8, no. 3, pp. 321–326, 2010.

[29] J. Nikoukaran, V. Hlupic, and R. Paul, *A Hierarchical Framework for Evaluating Simulation Software*, Simulation Practice and Theory, vol. 7, no. 3, pp. 219–231, 1999.

[30] S. Robinson, *Conceptual Modelling for Simulation Part I: Definition and Requirements*, Journal of the Operational Research Society, vol. 59, no. 3, pp. 278–290, 2008.

[31] R. Jain, *Art of Computer Systems Performance Analysis: Techniques for Experimental Design Measurements... Simulation and Modeling*, John Wiley, 2015.

[32] M. Mekni and B. Moulin, *A Survey on Sensor Webs Simulation Tools*, in proceedings of the Second International Conference on Sensor Technologies and Applications (SENSORCOMM), pp. 574–579, 2008.

[33] K. Romer and F. Mattern, *The Design Space of Wireless Sensor Networks*, IEEE Wireless Communications, vol. 11, no. 6, pp. 54–61, 2004.

[34] T. Socolofsky and C. Kale, *TCP/IP Tutorial*, RFC 1180, January 1991.

[35] L. Barroso and U. Hölzle, *The Case for Energy-proportional Computing*, Computer, vol. 40, no. 12, 2007.

[36] N. Nurseitov, M. Paulson, R. Reynolds, and C. Izurieta, *Comparison of JSON and XML Data Interchange Formats: A Case Study*, Caine, vol. 9, pp. 157–162, 2009.