Journal Title
XX(X):1–26
©The Author(s) 2016
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

A Survey of Interconnection Networks Simulators

Atiyeh Gheibi-Fetrat[1], Fatemeh Serajeh Hassan[1], Negar Akbarzadeh [2], Amir Mirzaei [1], Mahmoud Reza Kheyrati-Fard [1], Mohammad Hosseini [1], Ahmad Javadi Nezhad [1], Jeong-A Lee [3], and Hamid Sarbazi-azad [1, 2, *]

Abstract

Interconnection networks are essential for facilitating communication and resource sharing among multiple entities. These networks can be physical, connecting tangible hardware, or conceptual, linking abstract entities like social networks. The evaluation of interconnection networks is crucial for optimizing performance and ensuring performability, for which simulation is the most versatile and widely adopted approach. It provides flexibility for researchers to model various scenarios and configurations without the constraints of physical experimentation. This flexibility is particularly important as systems become increasingly complex and dynamic.

By utilizing simulation, researchers can identify potential bottlenecks and inefficiencies, leading to informed design decisions and enhancements. Various simulators have been developed for interconnection networks, each with specific strengths and limitations, but choosing the right simulator for a given scenario is challenging. In this paper, we survey notable interconnection network simulators and outline their advantages and disadvantages. We also provide detailed information about the most popular and famous interconnection network simulators. It helps researchers and developers to quickly review available interconnection network simulators and choose a suitable simulator for their specific purpose.

Keywords

Interconnection Networks; Performance Evaluation; Simulation; Network Simulators.

Introduction

Any system involving multiple entities to accomplish common tasks, such as a computer system with various components, requires an inter-entity communication medium known as an interconnection network. This network facilitates communication, data sharing, and resource management across diverse devices and locations, which may include computers, smartphones, sensors, storage units, and other hardware.

Interconnection networks can also be conceptual, connecting two abstract entities, such as social networks that define relationships among individuals in different contexts, like social media and applications. Whether physical or conceptual, a network is best modeled as a graph, where the connection between two nodes is represented as an arc, and each node is considered a vertex. This creates a significant overlap between the fields of interconnection networks and graph theory.

Research in network-related fields often necessitates evaluation tools. Among the three primary evaluation methods—measurement, simulation, and analytical modeling—simulation is the most widely utilized by researchers and developers. Various simulators have been developed, each with specific strengths and limitations. Choosing the most appropriate simulator for a particular study can be challenging and requires a thorough understanding of the available options.

In this paper, we survey a wide range of notable network simulators, categorizing them and outlining their advantages and disadvantages, while providing detailed information about the most popular ones. This document serves as a resource for researchers and developers to efficiently review available network simulators and select the ones that best meet their needs.

In what follows, in Section 2, we first review the basics of interconnection networks and talk about different evaluation means and their advantages and disadvantages. In Section 3, we delve into simulation tools and categorize them. Notable network simulators at micro and macro levels and their characteristics are reported in Section 4 and Section 5, respectively. Finally, we conclude this study in Section 5.

Interconnection networks and their evaluation

Interconnection Networks

An interconnection network is a crucial component of any networked system because the overall efficiency of the

[1] Sharif University of Technology , Tehran, Iran
[2] Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
[3] Chosun University, Gwangju, South Korea
[*] He is now a Visiting Research Professor at Chosun University under the Brain Pool (BP) Program of the National Research Foundation (NRF) of Korea.
Email: atiye.gheibi@gmail.com, fatemeh.serajeh@ce.sharif.edu, akbarzadeh@ce.sharif.edu, amir.mirzaei79@sharif.edu, mahmoud.kheyrati222@sharif.edu, smh.hosseini992@sharif.edu, ahmadjavadi17@gmail.com, jalee@chosun.ac.kr, azad@sharif.edu

system is very sensitive to network latency, throughput, and power consumption[171]. In addition to the technology, several key factors influence network efficiency: topology, switching method, and routing algorithm.

Topology Network topology defines the way nodes are connected and can be described using a graph[9 35 38]. The vertices of this graph are the nodes, and the edges are the physical channels that connect the nodes. The network diameter is the maximum value (in hops) of the minimum distances between any two nodes in the network. The number of links adjacent to a node is called node degree, and network degree is the maximum node degree in the network[34]. A network is regular if all nodes have the same degree. Finally, a network is symmetric if it is isomorphic to itself with any node labelled as origin[11 30]. The best topology is one that is regular and symmetric with a small diameter and node degree[8].

Many topologies have been proposed for interconnection networks, including the star, cube-connected cycles, generalised hypercube, pyramid, and k-ary n-cube. Figure 1 illustrates some of the most commonly used direct networks, the ring, 3-dimensional torus, and hypercube. These all belong to a major family of networks, called k-ary n-cubes, which have many desirable topological properties, including ease of implementation, modularity, symmetry, low diameter, and node degree, plus an ability to exploit locality exhibited by many parallel applications[172].

Switching Method Switching method determines the way messages visit intermediate nodes[40]. Several methods have been described in the literature, of which the two most important are Circuit Switching and Packet Switching. Popular switching methods used in current networked systems are of packet switching type[7 40]. Famous packet switching methods are store-and-forward[29] and virtual cut-through (or VCT for short)[10 25 28] and its buffer limited variation wormhole switching[26 39]. In store-and-forward switching a node will not forward an incoming message till it has the entire message stored in its channel buffer. While most first-generation parallel computers employed store-and-forward switching VCT and wormhole switching have been widely used later due to its low buffering requirement and good performance[176]. Here, a message is divided into flits (flow control unit) for transmission and flow control and each channel buffer needs to be only one flit in length. The first flit of a message, the header flit, includes the routing information and is followed by the data flits in pipelined fashion. If the header cannot be routed in the network due to contention for resources (buffers and channels), the data flits are also blocked in situ, keeping all the allocated buffers occupied[176]. Since wormhole routing uses pipelining, it can perform well even in a high diameter network[34]. Figure 2 illustrates the transmission of an 8-flit message from a source node to a destination node destined 5 hops away in a 4x4 mesh using store-and-forward and wormhole switching methods.

Because wormhole flits can be blocked in the network and then occupy switching resources, the method requires careful deadlock control. One solution to this problem is the use of virtual channel flow control[150 174]. Flow control concerns techniques for dealing with contention by multiple messages for the same channels and buffers. A good flow control policy should reduce congestion, be fair, and retain low latency. Flow control techniques are very dependent on the switching scheme employed; in wormhole routing, the commonest flow control strategy is the use of virtual channels[174].

The preceding switching techniques assume that messages or parts of messages are buffered at the input and output of each buffer channel. Buffers are commonly operated as first-in-first-out (FIFO) queues. Therefore, once a message occupies a buffer for a channel, no other message can access the physical channel even if the message is blocked[176]. Due to the chained blocking property of wormhole switching, the bandwidth of interconnection networks is then limited to a fraction (20%-50%) of the total available physical bandwidth[31 34]. However, it is also possible to multiplex several communications on a flit-by-flit basis by decoupling the allocation of buffers and physical channels, thus dividing a physical channel into several logical or virtual sub-channels[6]. A virtual channel consists of a buffer—together with associated state information—capable of holding one or more flits of a message. Virtual channels were first introduced to prevent deadlocks in wormhole networks based on the torus routing chip. However, it has been shown that virtual channels can also be used to improve network performance and latency by relieving contention[36].

Virtual channels dedicated to a physical channel may be organized in different ways. Figure 3 shows several organizations of virtual channels for a physical channel with a 12-flit buffer. The architectures differ in terms of performance, hardware requirements, and particularly arbiter complexity. Performance is dependent on network parameters, and there is an optimal number of virtual channels where the network performance is maximised[32 33].

Routing Algorithm Most interconnection networks provide multiple physical paths for routing a message between two given nodes. This introduces the problem of choosing the best route among many possible alternatives. Routing is a means used to achieve this. An important requirement for any routing algorithm is to ensure deadlock freedom; deadlock situations occur when no message can advance towards its destination because of occupied channels and buffers[27]. Many practical networked systems have used deterministic routing with virtual channels to ensure deadlock avoidance if needed[33 88]. This is achieved by forcing messages to visit the channels or virtual channels in a strict order[34]. Consequently, messages always take the same path between a given pair of nodes. This form of routing has the advantage of being simple but is unable to adapt to conditions such as congestion or failures. Dimension-ordered routing (DOR) is a typical example of deterministic routing where messages visit network dimensions in a pre-defined order. However, if any channel along the message path is heavily loaded, the message experiences large delays and if any channel along the path is faulty the message cannot be delivered at all. Adaptive routing improves both the performance and fault tolerance of an interconnection network and, more importantly, it has the ability to provide performance which is less sensitive to the communication pattern[67]. Figure 4 shows the channels that have available paths that can be taken by a message from node (0,0) to node (5,5) in a 6x6

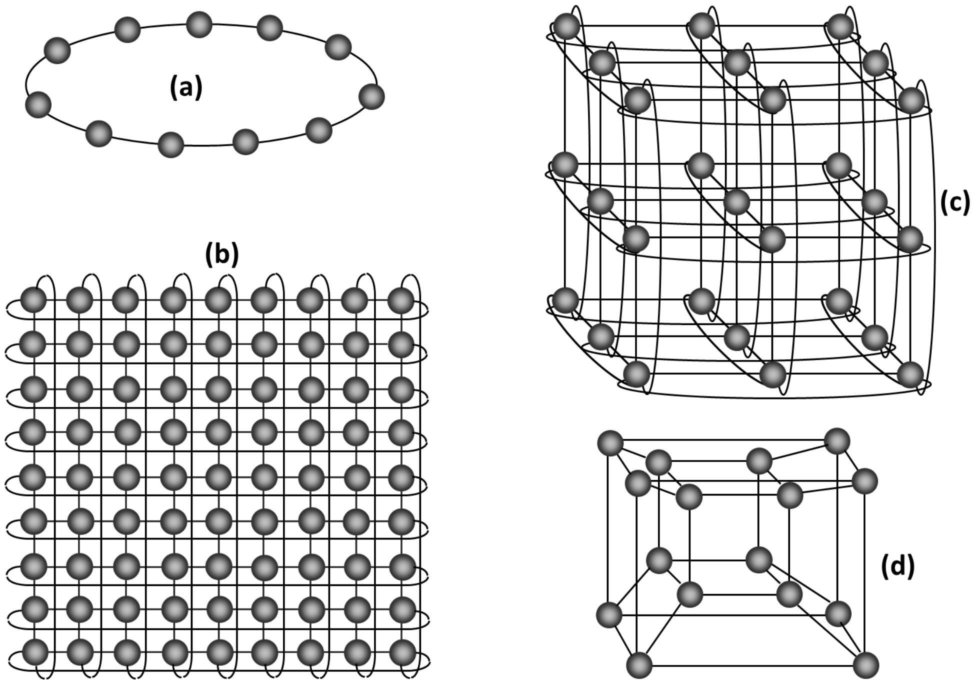

Figure 1. Some popular topologies. (a) A ring (with 11 nodes), (b) 2-D torus (9x9 nodes), (c) 3-D torus (3x3x3 nodes), and (d) 4-D hypercube or 4-cube.

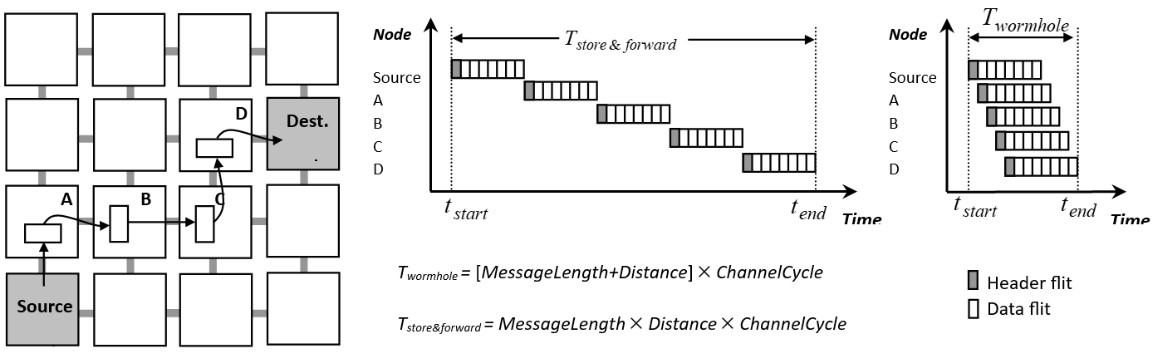

Figure 2. Transmission of an 8-flit message from the source node to the destination node destined 5 hops away in a 4x4 mesh (left) using store-and-forward (middle) and wormhole switching methods (right) via intermediate nodes A, B, C and D.

mesh with a deterministic XY routing algorithm and a fully adaptive routing algorithm.

Performance Evaluation and Simulators

Performance evaluation in various fields can be conducted through three primary means: measurement, simulation, and analytical modeling, each with its own advantages and disadvantages. Proper use of these three methods can lead to a more robust performance evaluation strategy. For instance, measurement can provide the foundational data needed for both simulation and analytical modeling. Simulations can test various scenarios informed by real-world measurements, while analytical models can offer theoretical insights that enhance understanding and interpretation of the measured and simulated data.

Measurement This involves collecting data from real systems or processes to assess performance metrics directly. Measurement is often the first step in performance evaluation, especially in environments where direct observation is feasible. This could involve using sensors, logging tools, or manual data collection techniques to gather performance metrics such as response times, throughput, error rates, and resource utilization. The advantages of measurement-based evaluation include:

– *Accuracy*: Provides real-world data and insights.

– *Relevance*: Reflects actual performance under current conditions.

– *Immediate Feedback*: Allows for quick identification of issues.

However, disadvantages are:

– *Costly and Time-Consuming*: Setting up measurement systems can be expensive and require significant time.

– *Limited Scope*: May not capture all variables or scenarios.

– *Interference*: The act of measuring can sometimes alter system performance.

Simulation This method uses models to replicate the behavior of systems over time, allowing for the testing of different scenarios without impacting real operations. Simulation allows for the exploration of complex systems where direct measurement might be impractical or impossible. By creating a virtual model of a system, various scenarios can be

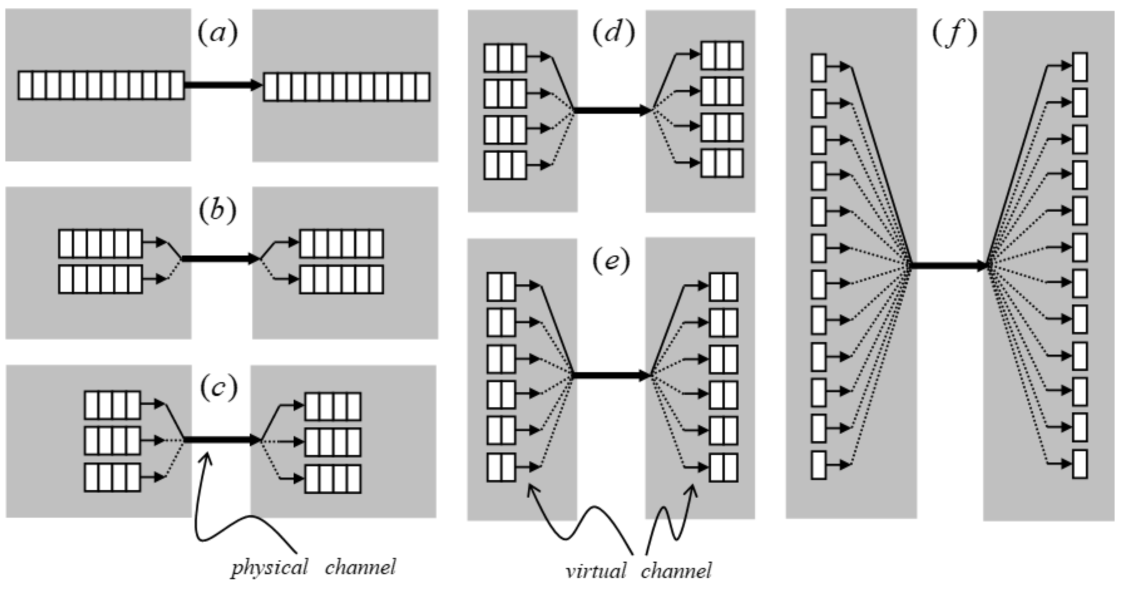

Figure 3. Organizing a 12-flit buffer dedicated to a physical channel; (a) conventional routers organize it into one FIFO queue, while a network using virtual channels may organize it into several independent lanes, resulting in different numbers of virtual channels each with different queue length, namely (b) two 6-flit, (c) three 4-flit, (d) four 3-flit, (e) six 2-flit, and (f) 12 1-flit buffers.

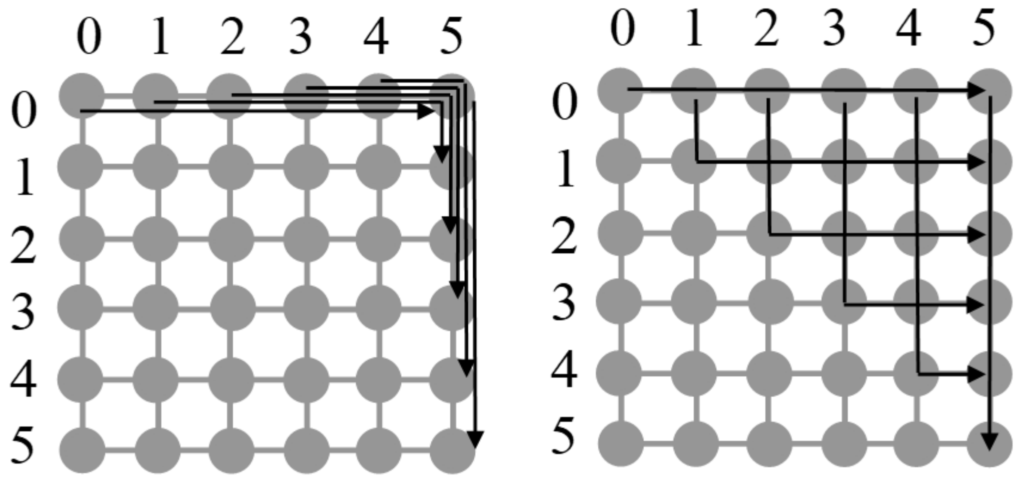

Figure 4. Routing messages in an 6x6 mesh from node (0,i) to node (i,5) (for i=0,1,. . . ,5); (left) using dimension order routing, five messages must traverse the channel from (0,4) to (0,5); (right) using adaptive routing, all messages proceed simultaneously.

tested, including extreme conditions that may not be encountered in regular operations. Advantages of the simulation method are:

– *Flexibility*: Can easily modify parameters and test various scenarios.

– *Risk-Free Testing*: No impact on actual systems, allowing for exploration of ”what-if” situations.

– *Visualization*: Often provides visual representations of performance metrics.

Disadvantages include:

– *Complexity*: Developing accurate simulations can be complicated and require expertise.

– *Data Dependency*: The quality of results is highly dependent on the input data and assumptions.

– *Computational Resources*: May require significant computational power and time.

Analytical Modelling This involves using mathematical formulas and theories to represent system behavior and performance. It employs mathematical techniques to derive performance metrics based on theoretical constructs. This method is particularly effective in systems where relationships between variables can be clearly defined. Analytical models are desirable due to their important advantages, including:

– *Efficiency*: Can provide quick results with relatively low computational resources.

– *Insightful*: Can offer deep insights into system performance and underlying mechanisms.

– *Scalability*: Easily scalable to larger systems or different configurations.

However, advantages come with some disadvantages, namely:

– *Simplifications*: Often relies on assumptions that may not hold true in real-world scenarios, leading to inaccuracies.

– *Limited Applicability*: May not be suitable for highly complex or dynamic systems.

– *Expertise Requirement*: Developing accurate models requires a strong understanding of both the system and mathematical techniques.

Use of the Evaluation Methods Note that 1) measurement and simulation results can be used to validate analytical models; 2) simulations can be enhanced using measured data to ensure that the virtual models accurately reflect real-world conditions; 3) analytical models can be used to validate simulation results, ensuring that they align with theoretical expectations, and 4) analytical models can guide the development of simulations by identifying key variables and relationships that need to be included. Conversely, results from simulations can be used to refine analytical models, ensuring they are more representative of real-world conditions.

Considering all advantages and disadvantages of the three evaluation means, simulation has been the one most widely employed. Many simulators of networked systems have been developed, introduced, and used by the researchers. In what follows, we will try to introduce, categorize, and explain some important ones in more details. Such a survey can be very useful for researchers, engineers, and students to properly select the simulator that best suits their purpose.

Simulators

The growing complexity of contemporary systems, which includes integrated circuits and global communication networks, necessitates the utilization of a potent tool called the simulator. They enable engineers and researchers to perform thorough analyses and improvements on network systems before they are physically created or deployed in the real world.

To best understand the various options available, we can categorize simulators into three main groups based on their scale and specialization: 1) Micro-scale network simulators which focus on intricate details of individual network components like routers and switches, and excel at modeling low-level protocols and hardware interactions within a small network, 2) Macro-scale network simulators which deal with larger network scenarios, typically focusing on internetworking and routing protocols across geographically dispersed areas, and finally, 3) specialized network simulators that caters to specific network types, such as wireless sensor networks or mobile ad-hoc networks. Each category offers unique advantages depending on the specific network behavior you want to analyze.

In the following subsections, we will delve into each category to better understand their unique capabilities and applications.

Micro-Scale Network Simulator

These simulators focus on interconnection networks in small-sized on-chip or on-board systems and for a limited number of interconnected devices. They are particularly useful for examining detailed interactions within networks with small geometry, including the performance and behavior of individual components.

One of the most important and popular micro-scale networks is on-chip networks or network-on-chips. Simulators for network-on-chip are essential in analyzing and enhancing the efficiency of on-chip interconnections in multi-processor systems. Various simulators have been developed to facilitate the design and testing of NoCs [4 5]. These simulators allow for the evaluation of various topologies, e.g., mesh and torus, taking into account factors such as latency, power consumption, and resource utilization. Efforts have been undertaken to migrate CPU-based simulators, like NEST, to diverse hardware platforms such as GPUs and FPGAs. This is aimed at enhancing the performance and expandability for conducting large-scale simulations[3]. These advancements allow researchers to efficiently investigate and enhance on-chip network configurations.

Macro-Scale Network Simulator

These simulators handle larger-scale networks, focusing on the interactions and performance of extensive network infrastructures. They are essential for studying the behavior of complex networks that span large geographical areas, including local, metropolitan, and wide-area networks. The following subsections explore different types of macro-scale network simulators.

Local Area Network Simulators Local Area Network (LAN) simulators model and mimic the operation of network designs to determine performance metrics like throughput, delay, and packet loss [92 93 94 95]. These simulators aid in optimizing network performance before actual implementation, crucial for both wired and wireless LANs. Examples include OPNET Modeller for wireless LANs and special-purpose languages like GPSS and SIMSCRIPT for general LAN simulations. Simulation parameters commonly analyzed in such simulators encompass throughput, delay, packet loss, processing time, and network capacity, among others. By utilizing these simulators and evaluation parameters, network engineers can assess and enhance LAN performance, identify bottlenecks, and plan for future network capacity requirements effectively.

Metropolitan Area Network Simulators Metropolitan Area Networks (MANs) are a type of network designed to cover larger geographical areas than LANs, but within the boundaries of a city or metropolitan region. One of their primary functions is to interconnect various local access networks, even when those networks use different technologies. MANs offer significant advantages, including high bandwidth, low latency, and reliable transmission quality. As a backbone network, a MAN can connect many LANs, facilitating efficient communication across the city. However, MANs are optimized for metropolitan-scale deployments and would need substantial modifications to be effective for Wide Area Networks (WANs) that span across greater distances [173].

To model and evaluate MAN performance, simulators employ various techniques to replicate the behavior of network traffic and infrastructure. One common approach is to use discrete event simulators for optical Wavelength

Division Multiplexing (WDM) networks, which enable detailed analysis of metrics such as network delay, packet loss, and bandwidth utilization, particularly in WDM unidirectional slotted ring networks[96 97]. These simulators take into account several key factors, including network topology, traffic patterns, latency-sensitive protocols, and redundancy strategies, to ensure high performance and reliability in MAN environments[98 99].

Wide Area Network Simulators Wide Area Network (WAN) simulators emulate the behavior of large-scale networks to evaluate performance and test various scenarios. Examples include NS2, NS3, OMNET++, QUALNET, and GUTS [100 101]. These simulators offer various simulation parameters such as network mobility support, realistic internet-scale topologies, artificial latency, bandwidth limitation, and throughput statistics display [102 201]. By utilizing these simulators, researchers and practitioners can analyze network behavior, assess the impact of different configurations, and optimize network performance without the need for costly real-world implementations, thus facilitating cost-effective and efficient network research and development [104].

Specialized Network Simulators

These simulators focus on specific network types or technologies, allowing for a more targeted analysis and optimization. They are particularly useful for studying unique network behaviors and performance metrics in specialized environments. The following subsections examine different types of specialized network simulators.

Software-Defined Networking (SDN) Simulator By decoupling the control plane from the data plane, Software-defined networking (SDN) allows for centralized control, automation of network tasks, and improved network design, management, and optimization[59]. SDN simulators are crucial tools used to validate proposed solutions before deployment in real networks, especially in complex and costly environments[56]. These simulators help in replicating network scenarios, measuring traffic flow, improving packet loss rates, and enhancing bandwidth utilization[57]. SDN simulators, such as Mininet and Miniedit, enable the simulation of SDN networks to evaluate performance under various operational conditions and security threats[58]. SDN simulators play a vital role in testing and validating SDN solutions, contributing to the advancement and adoption of this innovative networking technology[60].

Mobile Network Simulators A Mobile Ad-hoc Network (MANET) is a key focus in wireless communication research. It comprises some nodes that communicate via wireless links, without relying on a fixed infrastructure and backbone, or centralized control. To evaluate the behavior and performance of communication protocols in MANETs, researchers employ various methods. MANET simulators, in particular, play a crucial role in helping network developers verify the functionality and efficiency of MANETs in real-time scenarios [175]. Several platforms are available for simulating and testing MANET routing protocols, each with its own strengths. While no single simulator comprehensively covers all aspects of MANETs, some tools stand out. NS-3 is widely regarded for its broad protocol support across multiple layers, making it a top choice for MANET simulations. GloMoSim is highly scalable and particularly well-suited for simulating networks with large numbers of nodes. OMNET++ is another valuable option, known for its powerful graphical user interface (GUI), which enhances user experience and simulation visualization [175].

Network Security Simulators A network security simulator functions by integrating attack simulation, risk simulation, and security defense simulation subsystems to model various attack scenarios, adjust security defense strategies, and evaluate network security measures. Examples of network security simulators include Nessi2, which focuses on distributed and security simulations [85], and the Graphical Network Simulator (GNS3), which utilizes Kali Linux for penetration testing and security audits, allowing for the emulation of diverse network components and configurations to assess network vulnerabilities and performance [86]. Various simulators like NS2, NS3, OMNeT++, NetSim, REAL, OPNET, and QualNet offer different features, advantages, and disadvantages, catering to specific simulation requirements in wired and wireless networks [87]. These simulators enable users to simulate network topologies, analyze performance, and test security measures efficiently and cost-effectively.

Internet of Things Simulators Internet of Things (IoT) network simulators play a crucial role in analyzing and predicting the performance of IoT deployments [82]. These simulators must accurately replicate the behavior of network components at all levels of the system stack [83]. Various simulators have been developed for IoT and wireless sensor networks (WSNs), such as NS3, OMNeT++ (FLoRa), and LoRaSim, focusing on technologies like LoRa (Long Range) communication [84]. These simulators are evaluated based on parameters like Packet Delivery Ratio (PDR), utilization, memory usage, execution time, and collision rates to assess their performance in simulating IoT networks effectively. By utilizing these simulators and corresponding parameters, researchers can simulate IoT systems and applications before actual deployment, ensuring better performance and reliability.

Notable Network Simulators at Micro-Scale Level

In the previous sections, we categorized network simulators into three main groups: Micro-Scale Network Simulators, Macro-Scale Network Simulators, and Specialized Network Simulators. Each category serves unique purposes and offers distinct advantages based on the scale and specific network behaviors they aim to analyze.

This section introduces some of the most renowned simulators dedicated to micro-scale environments, providing a comprehensive view of their key features, strengths, and weaknesses. Furthermore, we will conduct a comparative analysis of these simulators to underline where each stands out. This assessment aims to equip researchers and industry professionals with the insights needed to choose the most effective simulation tools for their granular network research and development activities.

Supersim

SuperSim[2] is an open-source, flit-level interconnection network simulator designed specifically for evaluating the complexities of large-scale high-performance networks. It excels at providing a detailed model of network architectural intricacies, which is crucial for examining the performance implications of various router microarchitectures and adaptive routing algorithms. SuperSim offers a flexible and extensible simulation framework, allowing for quick modeling of systems through a programmer-centric approach. Its hierarchical abstract interface facilitates easy integration of component models, and it is complemented by a comprehensive suite of tools for efficient simulation management, analysis, and visualization through Python scripting.

SuperSim excels in offering in-depth modeling of interconnection network architectures, making it particularly effective for performance analysis of router microarchitectures and adaptive routing. It enables quick and straightforward system modeling through a flexible simulation framework designed to enhance programmer productivity. This is achieved by allowing the easy integration of new component models and facilitating the addition of new components and configurations via its hierarchical abstract interface. Furthermore, SuperSim includes tools for efficient management, analysis, and visualization of simulations using Python scripting. It supports multiple levels of simulation accuracy, from cycle-accurate to higher-level abstractions, adjustable according to the study's needs.

However, the detailed and flexible nature of SuperSim may pose challenges for users who are not well-versed in simulation frameworks. Additionally, its primary focus on large-scale network simulations might limit its applicability to smaller system designs. Certain idealized assumptions, such as those concerning congestion credit accounting and flow control techniques, may also lead to inaccurate performance predictions.

Booksim

BookSim[2 20] is a cycle-accurate simulator designed for Network-on-Chip (NoC) architectures and other interconnection networks[12 13]. It emphasizes simulation flexibility and accurate modeling of network components, including configurable parameters like topology, routing algorithm, flow control, and router microarchitecture. BookSim2[14] enhances these capabilities by introducing detailed modeling of router microarchitecture, supporting additional traffic models, and accurately simulating inter-router channel delays. The enhancements in BookSim2 focus on providing a more realistic and detailed simulation environment, validated through comparisons with RTL implementations, offering insights into various networking aspects.

BookSim provides a high degree of flexibility and modeling fidelity, making it ideal for evaluating novel network designs with cycle-accurate precision. Validated against RTL implementations of NoC routers, BookSim ensures the accuracy of simulation results. By using this detailed network simulator, researchers can understand the impact of accurately modeling the router pipeline on network performance. The updated version, BookSim2, enhances the simulation capabilities further by including detailed router microarchitecture modeling and supporting additional traffic models.

However, compared to some event-driven simulators, BookSim may run slower, especially under low to medium network load scenarios. Moreover, configuring and utilizing advanced features like dynamic buffer management and traffic class differentiation requires substantial effort and familiarity with the system, which may pose challenges for some users.

Noxim

The Noxim simulator[15] is a cycle-accurate Network-on-Chip (NoC) simulator that allows for the analysis of performance and power statistics of both wired and wireless NoC designs[16]. It is Developed using SystemC, a system description library written in C++. It offers the flexibility to modify network size, packet size, routing algorithms, and packet injection rates, enabling the evaluation of various parameters such as throughput, delay, energy consumption, and network traffic distributions [17]. Noxim is widely used due to its open-source nature, support for wireless networks, and the ability to simulate mesh wireless NoC architectures[23]. However, the original Noxim simulator lacks accuracy in power and thermal estimations due to the absence of temperature/power cross-impact considerations, which is addressed by the PAT-Noxim simulator[24], providing more precise power and temperature estimations for a wide range of VLSI technologies. Additionally, the original Noxim does not support trace-driven workloads. Pires et al.[17] enhance Noxim to run real trace-driven networks, thereby expanding its applicability.

Noxim is a versatile simulator capable of modeling both heterogeneous wired and wireless NoC architectures, making it an effective framework for exploring the design space of NoC-based systems. Its modular design allows researchers to easily add or modify components such as routing algorithms and traffic models, which significantly enhances research flexibility. Additionally, PAT-Noxim provides detailed statistics for evaluating performance and power metrics and supports the inclusion of new features and algorithms using standard C++ code. This flexibility is further augmented by its support for both cycle-accurate simulation and Transaction-Level Modeling, catering to varying simulation needs.

However, users unfamiliar with SystemC and C++ may need some acclimatization to effectively utilize Noxim's features. Additionally, while cycle-accurate simulation offers precision, it can be computationally intensive and time-consuming. These aspects may pose challenges, particularly in scenarios requiring rapid prototyping or large-scale simulations. Nevertheless, Noxim's comprehensive capabilities make it a powerful tool for in-depth NoC research and development.

Orion V3

ORION3[43] is a comprehensive tool designed for Network-on-Chip (NoC) router estimation, focusing on accurate power and area estimation methodologies within many-core architectures. The simulator employs both parametric and non-parametric modeling techniques, enabling users to derive models directly from actual physical implementation data, achieving average estimation errors of no more than 9.3% across various parameters. Developed as an open-source tool, ORION3.0 supports extensive microarchitecture configurations and multiple router RTL (Register Transfer Level) generators. It enhances flexibility by allowing customization of key parameters such as the number of ports and virtual channels. Additionally, ORION3.0 integrates advanced non-parametric regression techniques like radial basis functions and support vector machine regression, which provide robust estimations. While achieving high accuracy for power and area modeling, ORION3.0 also maintains backward compatibility with its predecessor, ORION2.0[18], by including logic template-based models.

ORION3 discussed offers significant strengths in improving accuracy for power and area predictions in microarchitectural designs. Its extensible and configurable architecture allows customization of key parameters, such as the number of ports and virtual channels, facilitating a wide range of applications. By leveraging advanced non-parametric regression techniques like radial basis functions and support vector machine regression, the model provides robust estimations grounded in actual physical implementation data. This enhances the reliability of the predictions and enables more precise design optimizations.

However, achieving maximum estimation accuracy requires users to supply high-quality training and testing datasets, which can be challenging without access to precise and comprehensive data. Additionally, the integration of advanced modeling techniques can be complex and may require a sophisticated understanding of statistical modeling and regression. This can be a barrier for users who lack expertise in these areas, potentially limiting the tool's accessibility and ease of use. Nevertheless, the strengths of ORION3 make it a powerful resource for those seeking detailed and accurate predictions in microarchitectural design.

SunFloor–3D

SunFloor-3D[45] is a design tool for synthesizing application-specific 3D NoCs, addressing the challenges of integrating efficient interconnects in 3D SoCs [44]. It determines optimal topologies, paths for communication flows, and component placement across 3D layers, showcasing significant improvements in power consumption (average of 38%) and delay (average of 13%) compared to 2D implementations [46 45]. The tool offers advantages such as enhanced performance, reduced power consumption, and minimized delays, making it a valuable asset for designing efficient 3D NoCs. However, challenges may include complexity in design optimization, potential resource constraints, and the need for specialized expertise in utilizing the tool effectively. Overall, SunFloor 3D proves to be a beneficial solution for optimizing 3D NoC designs, balancing performance gains with potential design complexities and resource requirements.

SunFloor-3D distinguishes itself by integrating various design methods specifically tailored for synthesizing custom NoCs that address the unique challenges posed by 3D integration technologies. This simulator allows for extensive customization, enabling users to tailor designs to specific application needs, which can lead to optimized performance metrics. The ability to adapt and fine-tune network parameters ensures that the resulting NoC architectures are highly efficient and performance-driven.

However, designing for 3D ICs introduces additional complexity, particularly in managing the maximum number of Through-Silicon Vias (TSVs) and handling vertical interconnections. These factors can create significant challenges in achieving effective implementation, as careful consideration and management of vertical links are critical to maintaining feasible and high-performing 3D NoC designs.

HNOCS

HNOCS (Heterogeneous Network-on-Chip Simulator)[89] is a modular, open-source NoC simulator based on OMNeT++[126]. It supports modeling heterogeneous NoCs with variable link capacities and varying numbers of virtual channels per port, addressing diverse traffic needs. The simulator includes three types of routers—synchronous, synchronous virtual output queue (VoQ), and asynchronous and offers detailed statistical measurements at both the flit and packet levels. Built on OMNeT++, HNOCS benefits from easy traceability, debugging utilities, and support for arbitrary topologies, making it highly extendible and scalable. It allows flexible traffic configuration and comprehensive performance evaluation, making it a versatile tool for advanced NoC research and design exploration.

One of primary strengths of HNOCS is its efficient simulation management. Leveraging the robust features of OMNeT++ for parallelism and flexible topology definition, HNOCS significantly reduces simulation run-time and complexity. This makes it an excellent tool for researchers looking to investigate the diverse aspects of heterogeneous NoC designs. However, any limitations or bugs within the OMNeT++ framework can directly impact HNOCS. Users might need to rely on the OMNeT++ community for fixes and updates, which can be a constraint in terms of timely resolution and support.

Nirgam

NIRGAM[155] is a cycle-accurate, discrete event simulator based on SystemC, designed for research in the field of Network on Chip (NoC). It provides a platform to experiment with NoC design, including routing algorithms and applications on various topologies. The simulator supports mesh and torus topologies, and can simulate different routing algorithms, buffer depths and configurable traffic patterns.

Nirgam simulator is highly regarded for its ability to provide detailed performance metrics, including latency and throughput, which are essential for effectively evaluating NoC designs. The simulator offers users the flexibility to configure various NoC parameters such as topology size, clock frequency, buffer depth, flit size, and the number of virtual channels. This allows for in-depth customization and fine-tuning to meet specific project needs, thereby enhancing the overall evaluation and optimization process.

However, Nirgam suffers from insufficient documentation, which can make it difficult for users to fully understand and leverage its features. This lack of comprehensive guides and support materials may hinder new users or those unfamiliar with the simulator. Additionally, Nirgam does not support a wide range of network topologies, which may limit its applicability for diverse research projects that require exploring various NoC configurations.

ATLAS

ATLAS[91] is an automated environment designed for the generation and evaluation of Networks-on-Chip (NoCs). It supports various NoC architectures, including those proposed by the GAPH Group[221]. The environment is implemented in Java, allowing execution across different hardware and software platforms. Key functionalities include tools for NoC generation, traffic generation, simulation, traffic evaluation, and power evaluation. Specifically, the NoC Generation tool allows users to configure network dimensions, bandwidth, and various routing algorithms, while the Traffic Generation tool enables the creation of different traffic patterns for simulation purposes. The evaluation tools help analyze performance metrics such as latency, throughput, and power consumption, providing comprehensive feedback for optimization.

S simulator stands out due to its ability to automate NoC generation and evaluation, significantly reducing design time and effort for researchers and engineers. It encompasses a comprehensive suite of tools for NoC generation, traffic generation, simulation, and power evaluation, making it an all-in-one solution for NoC prototyping and analysis. One of its key strengths is the capability to prototype NoCs on Xilinx FPGAs, allowing for practical testing in real-world scenarios. Additionally, ATLAS provides detailed reports on performance and power consumption, aiding in thorough evaluation and optimization. Its versatility makes it suitable for both academic research and industrial applications, especially in the telecommunications sector. However, it requires external VHDL/SystemC simulators, which can involve additional costs and setup efforts. Moreover, the simulator's extensive features and numerous configurable parameters can pose challenges for new users, necessitating a steep initial effort to become proficient with the tool.

MC-Sim

MC-Sim[137] is an efficient simulation framework designed to model heterogeneous multi-core systems. It integrates diverse components like processor cores, memories, custom hardware blocks, and network-on-chip (NoC) architectures on a single chip. MC-Sim provides rapid and accurate performance estimation, supporting various processor, memory, and NoC configurations. A key feature is its ability to generate fast, cycle-true behavioral C-based simulators for coprocessors using high-level synthesis, achieving significant speed improvements over RTL descriptions. MC-Sim allows for the simulation of complex real-life applications, enabling designers to explore and optimize MPSoC designs efficiently.

The MC-Sim simulator excels in enhancing simulation speed, achieving an impressive average 45x speedup relative to RTL descriptions. This performance leap allows for more efficient modeling of complex systems. It provides cycle-true simulation, ensuring accurate performance modeling and is capable of modeling real-life applications with a minimal deviation of just 7% from actual implementations. MC-Sim offers extensive configurability for processors, memory systems, and NoC architectures, supporting the integration of both loosely-coupled and tightly-coupled coprocessor models. The simulator can handle systems with up to 64 or more active superscalar cores and supports multi-tasking, making it adept at managing workloads composed of multiple single-threaded or multi-threaded applications.

However, leveraging MC-Sim to its fullest potential requires users to have a strong understanding of multi-core processor architecture, memory systems, and NoC configurations. Additionally, the simulator lacks an energy model and does not calculate energy consumption, which limits its comprehensiveness in terms of evaluating energy efficiency.

VNOC

VNOC 2.0[222] is a versatile Network-on-Chip (NoC) simulator developed from its predecessor VNOC 1.0. It is primarily implemented in C++, featuring a straightforward and clean codebase with abundant comments for ease of understanding. Key features include a simple graphical user interface (GUI), an ability to simulate various traffic types (such as uniform random, transpose, hotspot, and self-similar traffic using Glen Kramer's generator[223]), and integrated Orion 2 and 3 power models for power consumption estimates. VNOC 2.0 supports Dynamic Voltage and Frequency Scaling (DVFS) strategies, allowing users to implement and study their own DVFS ideas easily.

The VNOC simulator is distinguished by several strengths that make it a powerful tool for NoC research and development. Its graphical user interface (GUI) significantly enhances usability, allowing users to visualize simulations more effectively, although this feature may slightly extend runtime. VNOC 2.0's capability to simulate a variety of traffic patterns adds to its versatility, making it highly suitable for diverse research scenarios. The integration of embedded power models, specifically Orion 2 and 3, allows for precise power consumption estimates, which is critical for studies focused on energy efficiency. Additionally, VNOC's support for dynamic voltage and frequency scaling (DVFS) offers users the flexibility to experiment with different voltage and frequency scaling techniques, further broadening its applicability.

On the downside, the VNOC simulator requires users to manually edit source files to adjust buffer sizes for accurate power estimations, adding complexity to the setup process. The integration process for Orion power models is not automated, demanding manual coding changes and recompilation, which can be laborious and time-consuming. Furthermore, while the simulator is available for free for non-commercial use, it necessitates permission for commercial applications, potentially limiting its accessibility based on the users' intentions.

AcENoCs

The AcENoCs (Accelerated Emulation platform for Network-on-Chips) [139] is a configurable hardware/software platform designed for FPGA-accelerated emulation of Network-on-Chip (NoC) architectures. It aims to validate synchronous and Globally Asynchronous Locally Synchronous (GALS) based NoC architectures[224] with high speed and accuracy. The platform consists of reconfigurable network components, traffic generators and ejectors, and modules for statistics collection and analysis. It supports the exploration of various router features and network configurations, allowing users to perform quick design validations and optimizations. AcENoCs achieves significant performance improvements over traditional HDL and software simulators, with speedups ranging from 10000-12000X and 14-47X, respectively.

AcENoCs simulator boasts several key strengths, such as its ability to maintain cycle-accurate emulation, which is critical for validating timing-sensitive designs reliably. The platform supports the easy reconfiguration of emulation parameters and integrates a variety of router architectures and network-on-chip topologies. Additionally, it leverages a co-design approach that efficiently uses FPGA resources: hardware manages the network components while software oversees traffic generation, statistics analysis, and configuration. However, the simulator does have some limitations. The size of the network-on-chip (NoC) that can be emulated is restricted by the available FPGA resources, posing a challenge for larger network designs. The initial setup and integration of hardware and software components can be complex, demanding significant expertise from users. Furthermore, reconfiguring hardware components can introduce overhead and require time-consuming modifications, which can hinder rapid prototyping.

DART

The DART simulator[140] is a flexible and efficient FPGA based architecture designed for simulating Networks-on-Chip (NoCs) to meet the growing demand for on-chip communication bandwidth in multi-core systems. It virtualizes NoC components, mapping them to a generic simulation engine on an FPGA, allowing any NoC design to be simulated without resynthesizing the FPGA. DART achieves over 100 times speedup compared to traditional cycle-based software simulators while maintaining accuracy. Its architecture includes configurable components like traffic generators, routers, and flit queues, synchronized by a global time counter, optimizing for area efficiency and enabling larger NoC simulations within FPGA constraints. This eliminates the need for time-consuming FPGA resynthesis, providing a powerful, high-speed, and accurate simulation tool.

The DART simulator offers several significant advantages for Network-on-Chip (NoC) design simulation. Notably, it allows any NoC design to be simulated without the need to resynthesize the FPGA, saving considerable time and effort compared to traditional FPGA-based emulators. DART achieves more than 100 times speedup over cycle-based software simulators, ensuring rapid simulation without compromising accuracy. Its architecture is optimized for area efficiency, enabling the simulation of larger NoC designs within the FPGA's capacity constraints. Moreover, DART features configurable components, such as traffic generators, routers, and flit queues, which can be adjusted at runtime to simulate various NoC designs and behaviors. Despite these strengths, there are some limitations. The size of the NoC that can be simulated remains bound by the physical capacity of the FPGA. The setup and configuration process of the DART simulator, while flexible, can be complex and requires a deep understanding of NoC architectures and FPGA programming. Additionally, users must have access to suitable FPGA hardware, which may not be feasible for all users or institutions. Interacting with the simulator can introduce latency due to Input/Output overhead, particularly under high traffic conditions, affecting the overall simulation efficiency.

NoCTweak

NoCTweak[70] is an adaptable and open-source simulator, specifically crafted for analyzing the performance and energy efficiency of Network-on-Chip (NoC) architectures. Utilizing SystemC, NoCTweak ensures cycle-level precision and supports various settings, including diverse router types (such as wormhole, virtual-channel, and bufferless), network dimensions, routing protocols, and traffic schemes. It provides extensive statistical outputs, offering insights on metrics like average network latency, throughput, and router power consumption. The simulator leverages post-layout power data obtained from commercial CMOS standard-cell libraries to ensure accurate energy evaluations. Designed for the initial phases of many-core chip designs, NoCTweak facilitates rapid and flexible modeling of concurrent hardware modules.

The NoCweak simulator stands out by offering extensive customization capabilities for network and router parameters, which allows for highly tailored simulations suited to specific design needs. It provides thorough statistical outputs that enable detailed analysis of network performance and energy efficiency. Additionally, the simulator leverages accurate power and timing data from standard-cell libraries, ensuring the reliability of its simulation results. On the downside, NoCweak primarily supports 2-D mesh networks, which might pose a limitation for users interested in simulating more complex or varied network topologies. This focus can restrict its applicability to a broader range of designs and simulations.

DARSIM

DARSIM[106] is a highly configurable, parallel cycle-level network-on-chip (NoC) simulator designed to evaluate routing and virtual channel (VC) allocation algorithms. It supports extensive customization of hardware parameters and various traffic generation methods, including trace-based injectors and a built-in MIPS simulator. DARSIM utilizes a multithreaded simulation engine to distribute tasks across multiple cores, offering both cycle-accurate and performance-optimized simulations. Its open-source nature under the MIT license promotes accessibility. The simulator can operate in network-only mode or integrate with a MIPS-based multicore system, supporting MPI-style applications. DARSIM's design helps researchers explore NoC design space and assess performance impacts, aiding in the development of efficient multicore processors.

The DARSIM simulator offers several significant strengths, particularly its detailed customization of hardware parameters, routing, and virtual channel (VC) allocation algorithms, making it highly adaptable for various research and design needs. Its ability to leverage multiple cores for simulation tasks enhances performance, enabling faster simulations. It supports both trace-based traffic injection and cycle-level MIPS simulation, accommodating multiple traffic patterns and application scenarios. Additionally, DARSIM injects randomness to break arbitration ties, thereby reducing biases in traffic patterns and improving the fairness of simulations.

That said, DARSIM also has its drawbacks. Its multithreaded simulation, while providing speed, can be quite resource-intensive, necessitating powerful multicore processors to fully benefit from its capabilities. This high demand for processing power could be limiting for users without access to such robust hardware.

AdapNoC

AdapNoC[142] is a configurable, cycle-accurate FPGA-based Network-on-Chip (NoC) simulator designed to evaluate various NoC architectures and configurations. It supports a wide range of configurable parameters via software, operates with a dual-clock architecture, and includes a traffic aggregator for adaptive routing algorithms. By offloading some components, such as Traffic Generators (TGs) and Traffic Receptors (TRs), to software, AdapNoC enhances simulation speed and optimizes FPGA resource usage. The architecture allows for the simulation of both virtualized and non-virtualized networks, making it particularly beneficial for exploring many-core system designs. AdapNoC achieves significant speed improvements, demonstrating a 53x to 180x speed-up compared to traditional software simulators like Booksim.

The AdapNoC simulator offers notable strengths, including extensive configurability of NoC parameters such as topology, routing protocols, flow control mechanisms, and buffer sizes. It supports both non-virtualized and virtualized configurations, enabling the simulation of networks with up to 1024 nodes. Additionally, AdapNoC incorporates a centralized traffic aggregator and supports table-based adaptive routing algorithms, which enhances its routing simulation capabilities. By balancing hardware and software aspects, AdapNoC effectively leverages the strengths of both, contributing to improved overall simulation performance.

Nevertheless, AdapNoC has some weaknesses. The hardware/software co-design and extensive configurability can make it more complex to set up and use compared to simpler simulators. Despite various optimizations, it still demands significant FPGA resources, which poses a limitation for very large-scale simulations or for users with constrained hardware capabilities. Lastly, the virtualization process can introduce additional complexity in synchronizing intra-cluster and inter-cluster transmissions, potentially affecting performance and increasing the difficulty of maintaining simulation accuracy.

Garnet

The GARNET simulator[163] is a detailed, cycle-accurate model of an on-chip interconnection network integrated into the GEM5 full-system simulation framework. It accurately models a five-stage pipelined router with virtual channel flow control and includes detailed microarchitectural components like flit-level input buffers, routing logic, allocators, and the crossbar switch. This precision allows GARNET to capture realistic interactions between the memory hierarchy and the interconnection network, essential for accurate CMP design simulations. GARNET enables the evaluation of different network configurations and optimization techniques on full-system performance, such as comparing shared and private L2 cache configurations and assessing the benefits of Express Virtual Channels (EVCs). By providing detailed system-level insights, GARNET is a valuable tool for CMP system design and research.

The Garnet simulator excels in providing a cycle-accurate model of on-chip interconnection networks, capturing detailed aspects of router microarchitecture such as flit-level buffers, routing logic, and virtual channel (VC) flow control. By integrating with the GEM5 full-system simulator, Garnet allows for accurate simulations of CMP systems, reflecting realistic interactions between the memory hierarchy and interconnection network. It supports comprehensive evaluations of various network configurations and optimization techniques, aiding in decisions about shared vs. private L2 cache configurations and the impact of techniques like Express Virtual Channels (EVCs). Additionally, it includes performance counters for detailed performance analysis, which are crucial for optimizing system efficiency. Garnet can model networks with diverse topologies and configurations, making it suitable for studying future CMP systems with dozens to hundreds of nodes.

However, Garnet has its downsides. Its detailed modeling can lead to increased simulation overhead, slowing down simulations, especially for large-scale systems or realistic workloads. The complexity and depth of Garnet's features can be challenging for new users to learn and effectively utilize. Researchers focusing on low-level interconnection issues might find trace-driven simulation methods more appropriate, as Garnet's full-system approach may be unnecessarily complex for their needs. Since Garnet is integrated with GEM5, any limitations or issues within GEM5 can potentially affect Garnet's performance and accuracy. Additionally, the detailed simulations require significant computational resources, which can be a limitation for some research groups or applications.

Xmulator

XMulator is a listener-based, integrated simulation platform designed to evaluate multicomputer interconnection networks. It provides a flexible, object-oriented, and multilayered environment, allowing easy extension and independent development of different network components. XMulator supports various network topologies, routing algorithms, and switching techniques, and it is the first simulator capable of simulating any arbitrary interconnection topology, including those with faults. The simulator uses XML for defining topologies, parameters, and outputs, enhancing flexibility and interoperability. It is implemented in C# and employs modern software architecture paradigms to facilitate the modeling and performance evaluation of multicomputer networks [203].

Xmulator boasts several strengths, including its flexibility and extensibility, thanks to its object-oriented design and listener-based integration, which allow for easy extension and modification of the simulation environment without altering existing models. It offers a comprehensive toolbox with a wide range of network topologies, routing algorithms, switching techniques, and router models, making it versatile for various network designs. The XML-based configuration provides high flexibility and user-friendliness, simplifying the definition of network topologies, input parameters, and outputs. XMulator enhances performance accuracy by decoupling individual parts of its code, enabling accurate performance evaluation under different conditions, including the presence of faults. Its mixed-mode event processing efficiently handles both simulation events and programming language events, boosting simulation performance. The modular architecture, with its multi-layered design, ensures ease of maintenance, while extensive logging capabilities through log4net allow detailed tracing and monitoring of the simulation process.

However, the simulator's advanced features and flexible architecture can result in a steep learning curve for new users, requiring significant time and effort to master. The modular design and mixed-mode event processing, while enhancing flexibility, might introduce some performance overhead compared to specialized or optimized simulation tools. Being implemented in C#, Xmulator may not be as accessible to users familiar with other programming languages and may face limitations on platforms where C# is less commonly used. Detailed simulations and comprehensive features can demand significant computational resources, limiting large-scale or highly detailed simulations. Continuous development and maintenance are required to keep the simulator up-to-date with the latest advancements in interconnection network design and simulation methodologies, which can be resource-intensive.

CINSim

CINSim[48] is an advanced simulation tool designed for the detailed modeling of component-based interconnection networks. It stands out due to its ability to handle dynamic reconfiguration, allowing users to observe network behavior during changes in topology or component specifications. The simulator is capable of both steady-state and terminating simulations, which are crucial for analyzing transient behaviors in complex network systems. An additional notable feature of CINSim is its support for irregular multistage interconnection networks, enhancing its applicability beyond traditional tools that are typically limited to regular topologies. The simulator also incorporates stochastic events, facilitating more accurate and realistic performance evaluations.

The strengths of CINSim lie in its flexibility, dynamic reconfiguration, and support for both steady-state and terminating simulations. It is capable of modeling irregular multistage interconnection networks and incorporates stochastic events for more realistic performance evaluations. On the downside, CINSim have some limitations concerning the complexity associated with its extensive configurability, potentially making it challenging for new users to navigate effectively. Furthermore, while the ability to model dynamic reconfiguration is a significant advantage, it implies increased computational demands, which could constrain performance, especially for larger network models.

The table 1 provides a detailed comparison of well-known micro-scale network simulators, highlighting their core features and outputs. This comparison includes simulators across different programming languages, years of release, and accessibility, reflecting the range of tools available for network simulation research. The table categorizes each simulator's output capabilities, such as throughput, latency, area, and power, allowing for a view of which simulators are best suited for specific evaluation needs. By consolidating these details, this comparison aids in identifying the most appropriate simulators based on project requirements and specific micro-scale network performance aspects.

Other Simulators

There are also other notable simulators that are not widely used compared to the major simulators but offer valuable features and functionalities for specific applications. BigNetSim[160], for example, is tailored for large-scale interconnection networks and utilizes optimistic parallel discrete event simulation techniques to model packet-level communication efficiently. Its high scalability and detailed network modeling make it suitable for comprehensive studies of network performance and behavior. However, BigNetSim faces challenges such as synchronization overheads during parallel execution and the potential for rollback operations that can affect performance.

NoC_DEVS[143] provides a flexible and detailed component-based modeling environment based on the DEVS formalism, supporting all NoC topologies and emphasizing timing and performance metrics like delay and throughput. It offers advanced visualization capabilities and configurability, which can be beneficial for educational and detailed simulation purposes. Nonetheless, NoC_DEVS might struggle with simulation speed for large-scale systems due to the intricate nature of DEVS modeling and could face compatibility issues when integrating with other simulation tools or models.

Other notable tools include NNSE[105] and OCIN_TSIM[144], which offer specialized functionalities such as configurable simulation kernels and dynamic voltage and frequency scaling (DVFS), respectively. NNSE features a user-friendly GUI for easy network and traffic configuration and supports both regular and application-specific traffic patterns, although it is limited to 2D topologies and can be less effective for three-dimensional network architectures. OCIN_TSIM emphasizes detailed power modeling and visualization capabilities but requires specific versions of libraries and software environments, which can make setup cumbersome. GPNOCSIM[107], a Java-based general-purpose NoC simulator, supports multiple topologies and provides flexible configuration options. However, its use of Java may introduce performance bottlenecks, and the uniform traffic distribution assumptions may limit its applicability for more complex traffic scenarios. Additionally, simulators like FOLCS[141] and PANE[145] provide lightweight and versatile platforms for simulating NoC architectures with

Table 1. Feature Comparison of various Micro-Scale Simulators.

Name	Year	Language	Availability	Outputs			
				Throughput	Latency	Area	Power
Supersim	2018	C++	✓	✓	✓	✗	✗
Booksim2	2010	C++	✓	✓	✓	✗	✗
Noxim	2015	SystemC	✓	✓	✓	✗	✓
Orion3	2015	C and MATLAB	✓	✗	✗	✓	✓
SunFloor-3D	2010	SystemC	✗	✓	✓	✗	✓
HNOCS	2012	C++	✓	✓	✓	✗	✗
Nirgam	2007	SystemC	✓	✓	✗	✓	✗
ATLAS	2010	Java	✓	✓	✓	✗	✗
MC-SIM	2008	C	✗	✓	✓	✓	✗
VNOC	2014	C++	✓	✗	✓	✗	✗
AcENoCs	2011	HDL	✗	✓	✓	✗	✗
DART	2011	Verilog HDL	✗	✗	✓	✗	✗
NoCTweak	2012	SystemC	✓	✓	✓	✓	✓
DARSIM	2010	C++	✗	✓	✓	✗	✗
AdapNoC	2016	Verilog HDL	✗	✗	✓	✗	✗
Garnet	2009	C++	✓	✗	✓	✗	✗
Xmulator	2007	C#	✓	✗	✓	✗	✓
CINSim	2005	C++	✗	✓	✓	✗	✗

a focus on FPGA implementation and asynchronous communication, respectively. FOLCS may face difficulties in accurately simulating higher-level network behaviors due to its low-level focus, and PANE may encounter issues in validating asynchronous designs against real-world performance benchmarks.

Notable Simulators for Macro-Scale Networks

In this section, we delve into the simulators designed for larger scale and broader network environments, such as data centers, wide-area networks (WANs), and other large-scale distributed systems. These simulators provide critical insights into traffic flow management, routing efficiency, congestion control, and the scalability of network infrastructures. We will present an overview of several key macro-scale network simulators, detailing their distinctive features, strengths, and limitations. Additionally, a comparative analysis will be included to highlight the specific scenarios and conditions where each simulator excels, offering valuable guidance for researchers and engineers aiming to optimize large-scale network performance and reliability in varied contexts.

NS-2

The Network Simulator version 2 (NS-2) was developed in 1996 as part of the VINT (Virtual Inter Network Testbed) project through a collaborative effort by the University of California at Berkeley, the University of Southern California's Information Sciences Institute, Lawrence Berkeley National Laboratory, and Xerox Palo Alto Research Center[50]. NS-2 operates on event-driven principles, making it highly beneficial for researching communication networks. Leveraging OTcl for scripting flexibility and C++ for high-performance protocol implementation, NS-2 provides a robust environment with capabilities such as mobility models, energy analysis, and distinct separations between control and data planes. This combination of features has made NS-2 a widely utilized platform for evaluating network protocols and performance, benefiting from continuous improvements and adaptations over the years.

NS-2's strengths lie in its rich set of analysis tools and comprehensive network visualization capabilities, which facilitate in-depth studies and evaluations of network behaviors and performance metrics[52]. It supports an extensive range of protocols across all network layers, offering a vast model library that simplifies the creation of complex scenarios[130]. This flexibility in modeling various network environments, along with strong community support and extensive documentation, has positioned NS-2 as a cornerstone tool in network research.

However, NS-2 is not without its weaknesses. Creating simulations that strictly adhere to specific methodologies can be challenging, which may affect accuracy and reproducibility. The simulator's tracing system can be cumbersome, making debugging and analysis particularly difficult for large-scale network models. The intricate code structure further complicates this, presenting a steep barrier to new users and hindering customization efforts without a deep understanding of its architecture. Moreover, studies have highlighted issues with NS-2's TCP simulation accuracy, particularly when the first-hop link acts as a bottleneck in lower bandwidth networks like wireless and mobile environments, potentially limiting its applicability in these scenarios.

NS-3

Network Simulator version 3 (NS-3) is a widely used open-source network simulation tool, especially popular within the Internet of Things (IoT) domain. NS-3 is versatile, functioning both as a simulator and an emulator, allowing researchers and engineers to create operational network environments [49 53 54]. It supports extensive modeling capabilities for various network technologies, including TCP/IP, Wi-Fi, and wireless sensor networks, making it a powerful tool for analyzing and optimizing network performance in diverse scenarios[55].

One of NS-3's key strengths is its high modularity, which facilitates easy customization and extension. This is complemented by its unique emulation mode, enabling integration with real-world network environments, which is particularly useful for testing and validation of network protocols in realistic settings. Additionally, NS-3's purely C+±based model, as opposed to NS-2's mix of object-oriented Tcl and C++, enhances performance and simplifies debugging. The provision of a Python-based scripting API further adds flexibility, allowing for easier and more diverse simulation settings.

Despite these advantages, NS-3 has some notable weaknesses. The steep learning curve associated with the programming and networking skills required for effective usage has limited its adoption in the industrial community. However, the introduction of frameworks like SIFRAN aims to mitigate this challenge by providing a no-code simulation environment for IoT networks. These frameworks enable non-programmers to leverage NS-3's capabilities through simple web interfaces, broadening its accessibility.

Moreover, the development of specialized modules, such as the Sigfox protocol simulation module, has enhanced NS-3's utility by allowing detailed modeling of energy consumption and scenario planning, critical for IoT applications. Such modules enrich NS-3's feature set, making it a more compelling choice for researchers focused on energy-efficient network design and deployment planning.

While NS-3 benefits from an active and responsive community, which is crucial for ongoing development, bug fixes, and enhancements, it does face concerns regarding reliability and limited Python scripting support. Scalability can also be an issue when simulating large networks[52], which might pose challenges for extensive research projects or industrial applications.

ns-3 LENA

The ns-3 5G-LENA (Long-Term Evolution New Radio Access) module[166] is an advanced, open-source NR system-level simulator designed to facilitate research and evaluate new solutions in mobile communication networks[168].. This extension of the ns-3 simulator is particularly adept at modeling the complexities of 5G networks, including features like Dual-Polarized MIMO (DP-MIMO) for enhanced data rates and realistic propagation modeling, as well as Beamforming (BF) methods using Sounding Reference Signals (SRS) for precise channel estimation and improved network coverage in millimeter-wave and sub-6 GHz bands[169]. Researchers use ns-3 LENA to assess key network performance metrics such as throughput, latency, and energy efficiency across various network configurations[177].

One of the module's main strengths lies in its calibration based on 3GPP specifications, ensuring that its performance aligns with industrial standards and real-world network behaviors, particularly in outdoor environments[191]. This alignment provides confidence in the reliability and accuracy of simulation results, bridging the gap between theoretical research and practical application. ns-3 LENA also supports a wide range of frequency bands, offering flexibility for diverse research scenarios. Additionally, it incorporates advanced protocols and mechanisms, such as mobility management and Quality of Service (QoS), allowing for comprehensive and precise simulations that reflect real-world conditions[177]. These capabilities enable researchers to verify and optimize network configurations effectively.

However, ns-3 5G-LENA does present certain challenges. The extensive networking and programming expertise required to use the simulator effectively can limit its adoption outside of specialized research groups. Moreover, while the integration of advanced features like DP-MIMO and BF significantly boosts its capabilities, these sophisticated models and methods demand considerable computational resources, potentially impacting the scalability of simulations for very large networks.

COOJA

The COOJA simulator[193], closely integrated with the Contiki OS, is a prominent tool for simulating sensor networks, particularly focusing on resource-constrained devices. With its event-driven approach, COOJA facilitates efficient execution and detailed system behavior tracking. It supports pre-emptive multithreading on a per-process basis, allowing for the simulation of concurrent processes within each sensor node. This makes it particularly effective for studying complex, concurrent operations in sensor networks. Additionally, it features a comprehensive implementation of the TCP/IP protocol stack, which can be extended using custom interfaces to represent various sensor node properties, such as location, user interactions, and radio transceivers, thus offering a high degree of modularity for researchers[177].

One of COOJA's significant strengths is its seamless integration with real application code. Through a dedicated Java interface, the simulator allows researchers to execute existing application code directly within simulated sensor nodes, eliminating the need for code modifications and simplifying development and testing processes. Furthermore, COOJA's user-friendly graphical user interface (GUI) enables researchers to visually define network topologies and configure simulation parameters conveniently, enhancing the platform's usability. These features make COOJA a powerful tool for gaining insights into network performance and energy efficiency in a visual and accessible manner, thus accelerating sensor network research and development.

However, COOJA has certain limitations. It lacks support for sensing coverage modeling[182], which necessitates the use of additional tools like WSN-Maintain to address coverage issues and improve energy efficiency by putting redundant nodes to sleep. Furthermore, in low data rate systems, idle listening can dominate the energy costs, presenting challenges for optimizing power consumption[183].

QualNet

The QualNet simulator[63] is a powerful tool designed for the testing and simulation of network protocols, known for its high accuracy, speed, and portability[61]. It supports the analysis of various routing protocols, such as AODV, DSR, and OLSR, across different traffic loads, allowing for comprehensive comparisons based on metrics like Packet Delivery Ratio and End-to-End Delay[62]. This versatility makes it particularly useful for large-scale network simulations where detailed performance analysis is crucial. Additionally, QualNet can be enhanced with features like a programmable dynamic simulation interface that enables online configuration of static routes during simulations. This reduces the need for extensive pre-setting work, streamlining the configuration of complex, large-scale networks.

One of QualNet's primary strengths is its ability to perform high-powered simulations efficiently, leveraging multiprocessor systems and distributed computing. This capacity for handling complex networks allows researchers and engineers to achieve faster simulation results without sacrificing accuracy. The simulator's flexibility in modeling both wired and wireless technologies within the same network further enhances its utility, enabling more realistic and comprehensive performance analyses. Coupled with a user-friendly graphical user interface (GUI), QualNet simplifies model creation and interaction, allowing users to focus more on their research and less on the intricacies of the simulation setup.

However, there are certain challenges associated with using QualNet. As a commercial extension of GloMoSim, it requires licensing fees, which might limit accessibility for some users, particularly those in academia or smaller research institutions. Additionally, installing QualNet on Linux systems can be complex, and the Java-based user interface may suffer from performance issues, potentially impacting the workflow efficiency of some users.

GloMoSim

The Global Mobile Information System Simulator (GloMoSim)[108] is a prominent open-source parallel discrete-event simulator designed to analyze and study large-scale hybrid networks, encompassing wireless, wired, and satellite communications[109]. Known for its versatility, GloMoSim allows for simulations in both sequential and parallel environments, making it highly effective in handling extensive networks with numerous nodes[110]. This capability is particularly valuable for researchers and developers working on Mobile Ad-Hoc Networks (MANETs), where GloMoSim enables the evaluation of various network protocols, such as AODV, and aids in gaining insights into network behavior, security issues, and the performance of infrastructure-less wireless nodes.

One of GloMoSim's key strengths lies in its ability to model complex network scenarios and simulate diverse network environments. This enables researchers to analyze protocol performance under different conditions, providing a robust platform for testing and development. However, despite these advantages, GloMoSim does have some limitations. It may face challenges in accurately representing real-world network dynamics due to the need for ongoing updates to stay aligned with the rapid evolution of network technologies and protocols. Additionally, GloMoSim has certain constraints when it comes to sensor network routing and energy modeling, which can limit its effectiveness in specific applications.

NetSim

NetSim is a stochastic discrete event simulator that finds application across various domains such as robotics[111], IoT[112], network modeling[113], underwater vehicle operations, and blockchain systems[115]. It is highly adaptable and extensible, capable of real-time operation, and offers high-fidelity simulation. NetSim is particularly valued for its ability to model complex systems with a diverse range of structures. It supports a wide array of protocols, including Aloha, Ethernet variations, Wi-Fi standards, the TCP/IP suite, various routing protocols, and cellular technologies. Additionally, it can simulate Cisco routers and switches, providing a realistic environment that is beneficial for network design, troubleshooting, and protocol analysis.

One of NetSim's significant strengths is its ability to provide detailed and accurate representations of systems, support research in complex environments, and allow flexibility in adjusting model parameters and network sizes. This extensiveness makes it an effective tool for simulating diverse and intricate systems. NetSim's detailed simulations are crucial for ensuring that models closely mirror real-world scenarios, thereby enhancing the reliability of the research findings. Its real-time operation capability further increases its utility, particularly in fields like robotics and IoT, where real-time data processing and response are critical.

However, NetSim does have some limitations. It lacks appropriate tools for simulating certain specific aspects, which may require additional customization or supplementary tools to achieve the desired outcomes. Furthermore, as a proprietary tool, it is not open-source, potentially limiting its accessibility for some users or institutions. Another drawback is its limited distribution capabilities, particularly when advanced and realistic distributed simulations are necessary for large-scale network scenarios.

OPNET

OPNET (Optimized Network Engineering Tools)[119] is one of the leading fast discrete-event simulators widely used for comprehensive network simulation. It supports a variety of attributes, such as the ability to model different routing protocols like AODV and DSR, offering robust capabilities in the simulation of network behaviors. OPNET facilitates energy modeling for Wireless Sensor Networks (WSN) using RM battery models, enabling detailed analysis of energy consumption in sensor networks[120]. Additionally, it excels in simulating multimedia traffic scenarios to analyze Quality of Service (QoS) in WLAN networks, and it is adept at creating simulation models for communication transmission networks like Synchronous Digital Hierarchy (SDH) in power systems[121].

One of the key strengths of OPNET is its capability to accurately model network behaviors, leveraging three powerful technologies that enable high-fidelity simulations. This accuracy is crucial for researchers and engineers who need reliable data to base their analyses and decisions. OPNET's ability to simulate diverse network scenarios and evaluate performance metrics further enhances its utility, making it an indispensable tool for a wide range of applications, from academic research to industrial network design. Moreover, OPNET's integrated graphical user interface (GUI) simplifies debugging and model creation, offering an intuitive platform for users to interact with and refine their simulations.

However, OPNET does present some weaknesses. Despite its powerful modeling capabilities, there can be potential limitations in accurately modeling complex network behaviors, which may require meticulous adjustments and expert knowledge to overcome. Additionally, OPNET's connectivity options might be restricted compared to other tools, potentially limiting its integration with certain systems or networks.

OMNeT++

OMNeT++ (Objective Modular Network Tested in C++)[126 128] is a powerful network simulator distinguished by its comprehensive infrastructure and modular component architecture[122 123]. It is especially suitable for simulating communication networks and distributed systems, leveraging its component-based discrete event simulation capabilities. OMNeT++ offers extensive features for building, analyzing, and demonstrating computer networks, making it a popular choice in both research and teaching environments[124]. The flexibility of OMNeT++ permits the implementation and verification of various network algorithms, including centralized and distributed approaches for multi-hop networks, such as shortest path and flooding-based asynchronous spanning tree algorithms[125].

One of the primary strengths of OMNeT++ is its ease of tracing and debugging, which simplifies the development and fine-tuning of network simulations. Its modular and extensible architecture supports a wide range of network technologies, providing a versatile platform for simulating diverse scenarios. OMNeT++ also excels in the comprehensive simulation of hardware and physical effects, enabling researchers to conduct detailed and realistic analyses of network systems. This versatility and depth make OMNeT++ a valuable tool for a wide array of simulation needs, from academic research to industrial applications.

However, OMNeT++ does present some challenges. The complexity of its interfaces can distract from the main focus of the simulation, making it difficult to influence a running simulation directly. This has necessitated the development of a remote interface for live modifications and event monitoring, which adds an additional layer of complexity. Furthermore, while OMNeT++ is effective for most time-dependent algorithms, ensuring the accuracy of real-time data is crucial for certain scenarios, such as testing multi-hop communication in Vehicular Ad-hoc Networks (VANETs).

GrooveNet

GrooveNet[132] is a modular, event-based simulator specifically designed for vehicular networks, enabling seamless communication between simulated and real vehicles[133]. It incorporates various communication models and supports multiple network interfaces, GPS, and events triggered by vehicles' on-board computers. This versatility allows GrooveNet to simulate thousands of vehicles in any US city, making it an excellent tool for advanced vehicular network research. Its modular architecture also facilitates the addition of new models for networking, security, applications, and vehicle interaction, which makes it possible to tailor simulations to specific research needs.

One of the significant strengths of GrooveNet is its ability to run large-scale simulations that provide insights into market penetration requirements through extensive on-road testing[135]. This capability is crucial for understanding how vehicular networks perform under real-world conditions, covering significant distances and various traffic scenarios. GrooveNet supports hybrid simulation, combining real and simulated vehicles, which enhances its utility for protocol design and in-vehicle deployment. This feature is particularly important for studying message latency and coverage in inter-vehicular communication within real street map-based topographies[134].

However, GrooveNet does present some challenges. The complexity associated with incorporating both real and simulated vehicles can require additional effort for setup and ongoing maintenance, potentially posing a significant barrier for some users[136]. This complexity may necessitate a higher level of expertise and resources to manage effectively. Additionally, GrooveNet has been criticized for its poor documentation, which can further complicate the process of setting up and utilizing the simulator to its full potential.

TOSSIM

TOSSIM[148] is a discrete event simulator designed for TinyOS sensor networks, widely used to simulate wireless sensor networks (WSNs)[147]. It offers a range of features such as mobility support, energy consumption modeling, and realistic propagation components. By allowing for better comparison between experimental and simulation results, TOSSIM aids significantly in software testing for WSN-based applications[151]. Researchers and developers benefit from its ability to closely emulate real-world conditions, which helps streamline the testing and development processes for WSN applications.

One of the notable strengths of TOSSIM is its scalability, which allows it to handle large network sizes and high node speeds with acceptable additional execution time. This scalability is crucial for accurately modeling and estimating power consumption in extensive WSN deployments [152 153]. TOSSIM supports hardware-in-the-loop (HIL) integration, which enhances its utility by allowing physical nodes to be included in simulations[154]. This integration, as seen in extensions like H-TOSSIM, improves error detection and provides more accurate power consumption estimation. These capabilities make TOSSIM a valuable tool for simulating wireless sensor networks and conducting comprehensive software testing.

However, TOSSIM has its limitations. It may struggle to reveal detailed communication issues or timing errors, which can be critical for certain applications. Additionally, despite its energy consumption modeling capabilities, it lacks an entirely accurate power consumption model and provides limited support for power consumption estimation. These weaknesses can compromise its effectiveness in scenarios where precise power management and timing accuracy are crucial.

NCTUns

NCTUns (National Chiao Tung University Network Simulator) is a powerful network simulator specialized for vehicular networks, with the latest version, NCTUns 6.0[204], released in 2010, and the initial public release in 2002. This simulation tool supports both wired and wireless networks, making it versatile for the development, testing, and evaluation of network protocols[156 158]. NCTUns is particularly effective for modeling network behavior, analyzing packet flow in TCP networks within Vehicular Ad-hoc Networks (VANETs), and testing various VANET topologies[157]. Its robust graphical user interface (GUI) includes four primary components: the Topology Editor, Node Editor, Performance Monitor, and Packet Animation Player, with the Node Editor being particularly useful for module developers[130].

One of NCTUns's significant strengths is its ability to generate realistic traffic that reflects real-life applications, which enhances the accuracy and applicability of simulation results[130]. It leverages the kernel TCP/IP stack for faster simulations and integrates network and traffic simulations, making it particularly suitable for Intelligent Transportation System (ITS) applications[159]. This integration allows researchers to evaluate the performance of network protocols under realistic traffic conditions, providing valuable insights for the development of effective VANET solutions. Additionally, NCTUns's comprehensive toolset, including the GUI components, offers a user-friendly experience while maintaining the flexibility required for detailed network simulations.

However, NCTUns has some limitations. Operating the simulator effectively requires a certain level of expertise, which may pose a barrier for users without a strong background in network simulation. Additionally, accurately representing real-world conditions, especially in wireless multi-hop transmission models, remains a challenge. This can impact the fidelity of the simulation outcomes in complex network scenarios. Furthermore, the granular node manipulation in NCTUns, while providing detailed control, might hinder efficient configuration and scalability for larger, more complex network simulations.

SSFNet

SSFNet (Scalable Simulation Framework Network Models)[127 164] is a Java-based, discrete-event network simulator designed for comprehensive network simulation applications, including various network elements, topologies, protocols, and traffic scenarios. It is particularly adept at supporting large-scale network simulations, such as those representing the Internet, owing to its high-performance Java foundation. SSFNet efficiently manages global traffic patterns and offers robust support for many network protocols like BGP, TCP, UDP, HTTP, FTP, and others. Furthermore, it provides users with a clear and standardized Domain-Specific Modeling Language (DML) for model creation, which simplifies high-level network descriptions and enhances the flexibility of network design.

One of the notable strengths of SSFNet is its ability to tackle large-scale network simulations effectively. Its high-performance Java foundation allows for efficient management of global traffic patterns, making it suitable for simulating vast and complex networks[130]. The Domain-Specific Modeling Language (DML) offered by SSFNet simplifies the process of defining network models, enabling users to create detailed and extensive network simulations with relative ease. This support for a wide range of protocols adds to its flexibility and applicability across various research and development scenarios, allowing comprehensive testing and modeling of different network behaviors.

Despite these advantages, SSFNet has significant limitations, particularly concerning usability. It lacks supplementary tools for designing network elements and analyzing simulation results, which poses a challenge for general users[164]. Without these tools, users must rely on manual intervention to fine-tune simulations, increasing the likelihood of errors and making it difficult to achieve reliable results[165]. Additionally, the manual network modeling process required by SSFNet can be cumbersome and error-prone, especially when constructing complex network simulation models[165].

GNS3

The Graphical Network Simulator (GNS3)[208] is an open-source network emulation tool designed for emulating, configuring, testing, and troubleshooting both virtual and real networks. GNS3 supports a wide range of physical and virtual devices from various vendors, making it an invaluable resource for network engineers. Its ability to create diverse network topologies, ranging from small multi-node setups to large-scale networks akin to the Internet, enhances its versatility and practicality in educational and professional settings.

One of the key strengths of GNS3 is its open-source nature, which not only provides accessibility but also encourages community-driven development and customization. This accessibility allows network engineers to build and test network topologies quickly, making fast predictions and modeling complex scenarios with relative ease. GNS3's reputation for speed, accuracy, and flexibility makes it a powerful simulation platform that is valuable for both physical and network applications. These capabilities allow users to explore various network configurations and scenarios, facilitating a deeper understanding of networking principles and testing setups before deployment in real-world environments[161 162]. However, GNS3's complexity poses challenges, particularly during the setup and configuration phases, which can be time-consuming and detail-oriented.

This complexity might be daunting for newcomers or those without extensive experience in network simulation and emulation.

Cisco Packet Tracer

Cisco Packet Tracer is a proprietary commercial network simulator primarily used to emulate networks of various scales composed of Cisco network components. As a valuable educational tool, it enables users to visualize abstract networking concepts[189], thus enhancing the learning experience. By facilitating the implementation of Wide Area Networks (WANs) for data transfer and communication, Cisco Packet Tracer proves instrumental in teaching and evaluating key aspects such as traffic transmission, bandwidth usage, and overall network efficiency[190]. Its utility extends to providing practical hands-on experience, making it a critical resource in networking education and training.

A major strength of the Cisco Packet Tracer lies in its ability to serve as an effective tool for practical learning in networking. It allows students and professionals to build and test network configurations and scenarios in a simulated environment that closely mimics real-world network operations. This practical application helps users grasp complex networking concepts thoroughly. For instance, Gwangwava et al.[197] employed Cisco Packet Tracer to simulate an IoT control system for monitoring environmental variables in smart city and industrial applications, demonstrating its versatility and predictive capabilities in complex simulations.

Despite its benefits, Cisco Packet Tracer has certain limitations. One significant challenge is the need for highly skilled operators to simulate complex networks accurately, which can impact its accessibility and usability for beginners[189]. Potential errors during simulations can affect learning outcomes, necessitating a strong foundational knowledge to troubleshoot and resolve issues effectively. Additionally, being limited to Cisco hardware restricts its applicability in scenarios where diverse networking equipment is used.

The table 2 offers a comprehensive comparison of prominent macro-scale network simulators, focusing on their release year, programming languages, accessibility, graphical user interface (GUI) support, and parallelism capabilities. This comparison underscores the diversity of available simulators, each designed to address specific aspects of large-scale network modeling and analysis. By examining these key features, researchers and practitioners can better assess which simulators meet their project requirements, whether for educational, research, or commercial applications, facilitating informed decisions on tool selection for macro-scale network simulation tasks.

Other Simulators

In addition to the widely used macro-scale network simulators several other notable simulators have made significant contributions to the field. The Realistic And Large (REAL) simulator[210], developed at UC Berkeley in 1988, was designed to explore the dynamic behavior of flow and congestion control mechanisms in packet-switched data networks. It allows researchers to define network topologies and protocols, simulating their performance under various conditions[201]. Another significant simulator is Shunra Virtual Enterprise (Shunra VE) 5.0, a hardware-based simulation environment known for its speed, which was later discontinued in 2015.

SWANS[199], developed as a scalable alternative to NS-2, is another important tool for Wireless Network simulations, leveraging the JiST framework to assemble modular software components into complete wireless networks. ShoX[200], an object-oriented network simulator developed in Java, follows the OSI seven-layer model and offers a comprehensive platform for network simulations. A distributed version of ShoX was later developed to extend its capabilities[19]. J-Sim[22] is another open-source, component-based network simulation environment developed entirely in Java, primarily for simulating wireless sensor networks (WSNs). Meanwhile, the Harvard network simulator[20], introduced in 1999, is a TCP/IP simulator in BSD UNIX environments that simulates multiple network nodes by re-entering the UNIX kernel of the host system.

Further, EVE-NG[21] and Cisco Modeling Labs, previously known as Cisco Virtual Internet Routing Lab (VIRL)[216], are virtual environments for network emulation similar to Cisco Packet Tracer and GNS3, with the latter offering more advanced features. DRMSim[217], a Dynamic Routing Model simulator, and RDMSim[117], which focuses on decision-making under uncertainty, both provide platforms for testing new routing protocols and self-adaptation techniques. GTNetS[213], developed at Georgia Tech, is designed for large-scale simulations with a focus on scalability and ease of extension, implemented in object-oriented C++. Akaroa[209 212], a simulation package, automates processes during parallel stochastic simulations, while P2PRealm[211] specializes in simulating peer-to-peer networks with a focus on the efficiency of neural network-based algorithms. Each of these simulators offers unique features and strengths, contributing to the diverse landscape of network simulation tools.

Conclusion

In the paper, we presented a comprehensive survey of various interconnection networks simulators, at micro-scale and macro-scale level. Starting with an overview of notable micro-scale network simulators, we explored their unique features, strengths, and limitations in modeling fine-grained network architectures. These tools are crucial for researchers focusing on the intricate behaviors and performance characteristics of network components at a smaller scale.

The discussion then transitioned to macro-scale network simulators, where we delved into the detailed functionalities of well-established simulators. These platforms have been instrumental in enabling large-scale network simulations, offering robust environments to model complex network topologies, protocol stacks, and wireless communication systems. Additionally, we briefly highlighted other important macro-scale simulators, such as REAL, Shunra VE, and GTNetS, which further enrich the simulation landscape by

Table 2. Feature Comparison of various Macro-Scale Simulators.

Name	Year	Language	Availability	GUI	Parallelism
NS-2	1996	C++ and OTCL	Open Source	✓	✗
NS-3	2008	C++ and Optional Python Bindings	Open Source	✓	✗
NS-3 LENA	2011	C++ and Optional Python Bindings	Open Source	✗	✓
COOJA	2006	Java	Open Source	✓	✓
QualNet	2000	C++	Commercial	✓	✓
GloMoSim	1998	C	Open Source	✗	✓
NetSim	1997	C and Java	Proprietary	✓	✗
OPNET	1987	C and C++	Commercial	✓	✓
OMNeT++	2001	C++	Open Source for study, Commercial for industrial purposes	✓	✗
GrooveNet	2006	C++	Open Source	✓	✗
TOSSIM	2003	Python, C++ and NesC	Open Source	✓	✗
NCTUns 6.0	2010	C++	Open Source	✓	✗
SSFNet	1999	Java and C++	Open Source	✗	✓
GNS3	2008	Python	Open Source	✓	✗
Cisco Packet Tracer	1999	Python, JavaScript and Visual (Blockly)	Academic	✓	✗

addressing specific needs, from routing protocol validation to distributed simulations.

Overall, our survey underscores the diversity and evolution of network simulation tools, each catering to different research requirements. As network technologies continue to advance, the need for more sophisticated, scalable, and accurate simulators becomes increasingly evident. Future developments in this field will likely focus on enhancing the realism of simulations, improving scalability, and integrating emerging network paradigms like 5G, IoT, and cloud computing. By continuously refining these tools, the research community can better understand and address the challenges posed by modern networks, ultimately driving innovation in network design and optimization.

References

1. Dally, William James, and Brian Patrick Towles. Principles and practices of interconnection networks. Elsevier, 2004.
2. McDonald, Nic, Adriana Flores, Al Davis, Mikhail Isaev, John Kim, and Doug Gibson. ”SuperSim: extensible flit-level simulation of large-scale interconnection networks.” In 2018 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 87-98. IEEE, 2018.
3. Nguyen, Quang Anh Pham, et al. ”Transitioning spiking neural network simulators to heterogeneous hardware.” ACM Transactions on Modeling and Computer Simulation (TOMACS) 31.2 (2021): 1-26.
4. Amerikanov, A. A., and A. S. Ponomarev. ”Universal on-chip network simulator for networks-on-chip development.” 2021 International Russian Automation Conference (RusAutoCon). IEEE, 2021.
5. Prabhu Prasad, B. M., Khyamling Parane, and Basavaraj Talawar. ”An efficient FPGA-based network-on-chip simulation framework utilizing the hard blocks.” *Circuits, Systems, and Signal Processing* 39.10 (2020): 5247-5271.
6. R. Bashizade, and H. Sarbazi-Azad. ”P2R2: Parallel pseudo-round-robin arbiter for high performance NoCs.” Integration 50 (2015): 173-182.
7. N. Teimouri, M. Modarressi, and H. Sarbazi-Azad. ”Power and performance efficient partial circuits in packet-switched networks-on-chip.” *21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing.* IEEE, 2013.
8. S. Razavi, and H. Sarbazi-Azad. ”The triangular pyramid: Routing and topological properties.” *Information Sciences* 180.11 (2010): 2328-2339.
9. D. Rahmati, H. Sarbazi-Azad, S. Hessabi, and A. E. Kiasari. ”Power-efficient deterministic and adaptive routing in torus networks-on-chip.” *Microprocessors and Microsystems*, 36(7), 2012 571-585.
10. P. Lotfi-Kamran, M. Modarressi, and H. Sarbazi-Azad. ”An efficient hybrid-switched network-on-chip for chip multiprocessors.” *IEEE Transactions on Computers* 65.5 (2015): 1656-1662.
11. Guerrier, Pierre, and Alain Greiner. ”A generic architecture for on-chip packet-switched interconnections.” Proceedings of the conference on Design, automation and test in Europe. 2000.
12. Marri, J., Manishankar, S., Radha, D., and Moharir, M. (2019, July). Implementation and analysis of adaptive odd-even routing in booksim 2.0 simulator. In 2019 International Conference on Communication and Electronics Systems (ICCES) (pp. 76-83). IEEE.
13. Pérez, I., Vallejo, E., Moreto, M., and Beivide, R. (2020, August). BST: A BookSim-based toolset to simulate NoCs with single-and multi-hop bypass. In 2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) (pp. 47-57). IEEE.
14. Jiang, Nan, Daniel U. Becker, George Michelogiannakis, James Balfour, Brian Towles, David E. Shaw, John Kim, and William J. Dally. ”A detailed and flexible cycle-accurate network-on-chip simulator.” In 2013 IEEE international

symposium on performance analysis of systems and software (ISPASS), pp. 86-96. IEEE, 2013.

15. Catania, V., Mineo, A., Monteleone, S., Palesi, M., and Patti, D. (2016). Cycle-accurate network on chip simulation with noxim. ACM Transactions on Modeling and Computer Simulation (TOMACS), 27(1), 1-25.
16. Saliu, M. S., Momoh, M. O., Chinedu, P. U., Nwankwo, W., and Daniel, A. (2021). Comparative Performance Analysis of Selected Routing Algorithms by Load Variation of 2-Dimensional Mesh Topology Based Network-On-Chip. ELEKTRIKA-Journal of Electrical Engineering, 20(3), 1-6.
17. Pires, Ivan Luiz Pedroso, Marco Antonio Zanata Alves, and Luiz Carlos Pessoa Albini. ”Trace-driven and processing time extensions for Noxim simulator.” Design Automation for Embedded Systems 23.1 (2019): 41-55.
18. Kahng, A. B., Li, B., Peh, L. S., and Samadi, K. (2009, April). ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration. In 2009 Design, Automation and Test in Europe Conference and Exhibition (pp. 423-428). IEEE.
19. Janacik, Peter, Johannes Lessmann, and Michael Karch. ”Distributed simulation environment for the ShoX network simulator.” 2010 Sixth International Conference on Networking and Services. IEEE, 2010.
20. Wang, Shie Yuan, and H. T. Kung. ”A simple methodology for constructing extensible and high-fidelity TCP/IP network simulators.” IEEE INFOCOM’99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320). Vol. 3. IEEE, 1999.
21. VE-NG Ltd. 2024. EVE-NG Website. https://www.eve-ng.net/
22. Sobeih, Ahmed, Wei-Peng Chen, Jennifer C. Hou, Lu-Chuan Kung, Ning Li, Hyuk Lim, Hung-Ying Tyan, and Honghai Zhang. ”J-sim: A simulation environment for wireless sensor networks.” In 38th Annual simulation symposium, pp. 175-187. IEEE, 2005.
23. Chen, Kun-Chih, and Ting-Yi Wang. ”NN-noxim: High-level cycle-accurate NoC-based neural networks simulator.” 2018 11th International workshop on network on chip architectures (NoCArc). IEEE, 2018.
24. Norollah, A., Derafshi, D., Beitollahi, H., and Patooghy, A. (2018, September). PAT-NOXIM: a precise power and thermal cycle-accurate NoC SImulator. In 2018 31st IEEE International System-on-Chip Conference (SOCC) (pp. 163-168). IEEE.
25. Salminen, Erno, Ari Kulmala, and Timo D. Hamalainen. ”Survey of network-on-chip proposals.” white paper, OCP-IP 1 (2008): 13.
26. Dally, and Seitz. ”Deadlock-free message routing in multiprocessor interconnection networks.” IEEE Transactions on computers 100.5 (1987): 547-553.
27. Duato, Jose, Sudhakar Yalamanchili, and Lionel Ni. Interconnection networks. Elsevier, 2002.
28. De Micheli, G., Seiculescu, C., Murali, S., Benini, L., Angiolini, F., and Pullini, A. (2010, June). Networks on chips: From research to products. In Proceedings of the 47th Design Automation Conference (pp. 300-305).
29. Tanenbaum, Andrew S. ”Computer Networks, Prentice-Hall International.” Upper Saddle River, NJ 2 (1996).
30. Vaidya, Aniruddha S., Anand Sivasubramaniam, and Chita R. Das. ”Impact of virtual channels and adaptive routing on application performance.” IEEE Transactions on Parallel and Distributed Systems 12.2 (2001): 223-237.
31. Dally, William J. ”Performance analysis of k-ary n-cube interconnection networks.” IEEE transactions on Computers 39.6 (1990): 775-785.
32. Dally, William J. ”Virtual-channel flow control.” IEEE Transactions on Parallel and Distributed systems 3.2 (1992): 194-205.
33. Duato, Jose, Sudhakar Yalamanchili, and Lionel Ni. Interconnection networks. Elsevier, 2002.
34. Sarbazi-Azad, Hamid. Performance analysis of wormhole routing in multicomputer interconnection networks. Diss. University of Glasgow, 2001.
35. Gheibi-Fetrat, A., Akbarzadeh, N., Hessabi, S., and Sarbazi-Azad, H. (2023). Tulip: Turn-Free Low-Power Network-on-Chip. IEEE Computer Architecture Letters.
36. A. Mirhosseini, M. Sadrosadati, A. Fakhrzadehgan, M. Modarressi, and H. Sarbazi-Azad (2015, March). ”An energy-efficient virtual channel power-gating mechanism for on-chip networks.” *In 2015 Design, Automation and Test in Europe Conference and Exhibition (DATE)* (pp. 1527-1532). IEEE.
37. D. Rahmati, S. Murali, L. Benini, F. Angiolini, G. De Micheli, and H. Sarbazi-Azad. ”A method for calculating hard qos guarantees for networks-on-chip.” *In Proceedings of the 2009 International Conference on Computer-Aided Design*, pp. 579-586. 2009.
38. H. Hashemi-Najafabadi, H. Sarbazi-Azad, and P. Rajabzadeh. ”An accurate performance model of fully adaptive routing in wormhole-switched two-dimensional mesh multicomputers.” *microprocessors and microsystems* 31.7 (2007): 445-455.
39. H. Sarbazi-Azad, M. Ould-Khaoua, and L. M. Mackenzie. ”An accurate analytical model of adaptive wormhole routing in k-ary n-cubes interconnection networks.” *Performance Evaluation* 43.2-3 (2001): 165-179.
40. M. Modarressi, A. Tavakkol, and H. Sarbazi-Azad. ”Virtual point-to-point connections for NoCs.” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2010): 855–868.
41. V. A. Belyakov, O. G. Filatov, S. A. Grigoriev, S. E. Sytchevsky, and V. N. Tanchuk, “Computer code ‘ORION’ for simulation of heat and mass transfer in materials impacted by high heat fluxes,” *Plasma Devices and Operations*, **12**(2) (2004), 103–121.
42. H. S. Wang, X. Zhu, L. S. Peh, and S. Malik, “Orion: A power-performance simulator for interconnection networks,” *Proc. 35th Annual IEEE/ACM International Symposium on Microarchitecture*, 2002, pp. 294–305.
43. A. B. Kahng, B. Lin, and S. Nath, “ORION3. 0: A comprehensive NoC router estimation tool,” *IEEE Embedded Systems Letters*, **7**(2) (2015), 41–45.
44. J. D’Hoore, P. Bahrebar, and D. Stroobandt, “3D NoC emulation model on a single FPGA,” *Proc. Workshop on System-Level Interconnect: Problems and Pathfinding Workshop*, 2020, pp. 1–8.
45. C. Seiculescu, S. Murali, L. Benini, and G. De Micheli, “SunFloor 3D: A tool for networks on chip topology synthesis for 3-D systems on chips,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **29**(12) (2010), 1987–2000.

46. K. Latif, A. M. Rahmani, K. R. Vaddina, T. Seceleanu, P. Liljeberg, and H. Tenhunen, "Enhancing performance sustainability of fault tolerant routing algorithms in NoC-based architectures," *Proc. 2011 14th Euromicro Conference on Digital System Design*, 2011, pp. 626–633.
47. A. R. Bambhaniya, Y. Chen, R. Banerjee, T. Krishna, and others, "Proteus: HLS-based NoC Generator and Simulator," *Proc. 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–6.
48. D. Tutsch, D. Lüdtke, A. Walter, and M. Kühm, "CINSim-a component-based interconnection network simulator for modeling dynamic reconfiguration," *Proc. 12th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA 2005)*, 2005, pp. 132–137.
49. M. L. Almada and C. A. Talay, "Consideraciones al momento de evaluar la migración del simulador ns-2 al ns-3," *Informes Científicos Técnicos-UNPA*, **12**(2) (2020), 63–83.
50. N. Badini, M. Marchese, and F. Patrone, "Ns-3-based 5g satellite-terrestrial integrated network simulator," *Proc. 2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON)*, 2022, pp. 154–159.
51. M. H. Kabir, S. Islam, M. J. Hossain, and S. Hossain, "Detail comparison of network simulators," *International Journal of Scientific & Engineering Research*, **5**(10) (2014), 203–218.
52. R. L. Patel, M. J. Pathak, and A. J. Nayak, "Survey on network simulators," *International Journal of Computer Applications*, **182**(21) (2018), 23–30.
53. S. Si-Mohammed, M. Janumporn, T. Begin, I. G. Lassous, and P. Vicat-Blanc, "Sifran: Evaluating iot networks with a no-code framework based on ns-3," *Proc. 2022 Latin America Networking Conference*, 2022, pp. 42–49.
54. M. Naeem, M. Albano, D. Magrin, B. Nielsen, and K. Guldstrand, "A sigfox module for the network simulator 3," *Proc. 2022 Workshop on ns-3*, 2022, pp. 81–88.
55. M. Hawa, "A Graphical User Interface for the ns-3 Simulator," *Proc. 12th International Conference on Computer Modeling and Simulation*, 2020, pp. 159–163.
56. S. Buzura, A. Peculea, B. Iancu, E. Cebuc, V. Dadarlat, and R. Kovacs, "A hybrid software and hardware SDN simulation testbed," *Sensors*, 23(1) (2023), 490.
57. D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, 103(1) (2014), 14–76.
58. R. Khalifa and M. El-Aasser, "Network security challenges in SDN environments," *Proc. 2022 5th International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2022, pp. 1–6.
59. T. Aditya, A. D. David, G. Thippanna, M. M. Kousar, and K. Swetha, "The Future of Networking: Embracing Software-Defined Solutions," *Future*, 3(2) (2023).
60. A. Nunez, J. Ayoka, M. Z. Islam, and P. Ruiz, "A brief overview of software-defined networking," *arXiv preprint arXiv:2302.00165*, 2023.
61. M. S. Abood, M. M. Hamdi, A. S. Mustafa, Y. A. Najem, S. A. Rashid, and I. J. Saeed, "Analysis and Simulation for Mobile Ad Hoc Network Using QualNet Simulator," *Proc. International Conference on Intelligent Systems Design and Applications*, 2020, pp. 689–700.
62. L. Chen, K. Xue, J. Li, N. Yu, R. Li, Q. Sun, and J. Lu, "SimQN: a network-layer simulator for the quantum network investigation," *IEEE Network*, 2023.
63. N. H. I. Al-Rekabi and B. Y. M. Al-Sultani, "Qualnet performances of grid-based clustering for WSN's routing protocols," *Indonesian Journal of Electrical Engineering and Computer Science*, **17**(1) (2020), 448–456.
64. H. Li, P. Gong, Y. Liu, G. Li, and D. Shan, "PDSI-based Static Route Online Configuration Method for QualNet Simulator," *Proc. 2019 21st International Conference on Advanced Communication Technology (ICACT)*, 2019, pp. 325–329.
65. X. Shu, H. Wang, Z. Xu, K. Ning, and G. Wu, "A Distributed Simulator of Mobile Ad Hoc Networks," *Proc. International Conference on Wireless Algorithms, Systems, and Applications*, 2022, pp. 165–177.
66. C. Sauer, M. Sliskovic, and M. Schmidt, "Simulating Mobile Networks for Industrial Applications," *Proc. 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 1371–1374.
67. Ould-Khaoua, Mohamed, and Hamid Sarbazi-Azad. "An analytical model of adaptive wormhole routing in hypercubes in the presence of hot spot traffic." IEEE Transactions on Parallel and Distributed Systems 12.3 (2001): 283-292.
68. R. Lerbour and R. Trifan, "Devices and method for the simulation of a mobile telecommunications network," *US Patent 10,764,761*, Sep. 1, 2020.
69. S. Khan, S. Anjum, U. A. Gulzari, and F. S. Torres, "Comparative analysis of network-on-chip simulation tools," *IET Computers & Digital Techniques*, **12**(1) (2018), 30–38.
70. A. T. Tran and B. Baas, "NoCTweak: a highly parameterizable simulator for early exploration of performance and energy of networks on-chip," *VLSI Computation Lab, ECE Department, University of California, Davis, Tech. Rep. ECE-VCL-2012-2*, 2012.
71. I. L. P. Pires, M. A. Z. Alves, and L. C. P. Albini, "Trace-driven extension for noxim simulator," *Proc. 2017 VII Brazilian symposium on computing systems engineering (SBESC)*, 2017, pp. 102–108.
72. K. C. J. Chen, T. Y. G. Wang, and Y. C. A. Yang, "Cycle-accurate noc-based convolutional neural network simulator," *Proc. International Conference on Omni-Layer Intelligent Systems*, 2019, pp. 199–204.
73. M. Raurell-Torredà, I. Zaragoza-García, A. M. Aliberch-Raurell, J. Sánchez-Chillón, M. Torralba-Melero, O. Arrogante, A. Rojo-Rojo, R. Gómez-Ibáñez, M. Lamoglia-Puig, and M. Farrés-Tarafa, "SIMULAZERO: taller de simulación para actualizar conocimientos y habilidades en la prevención de la neumonía asociada a ventilación mecánica y bacteriemia relacionada con catéter (Proyectos Zero)," *Enfermería Intensiva*, **33** (2022), S45–S55.
74. S. Bouter, E. van Weel-Baumgarten, and S. Bolhuis, "Construction and validation of the Nijmegen evaluation of the simulated patient (NESP): assessing simulated patients' ability to role-play and provide feedback to students," *Academic Medicine*, **88**(2) (2013), 253–259.
75. M. Millberg, E. Nilsson, R. Thid, S. Kumar, and A. Jantsch, "The Nostrum backbone-a communication protocol stack for networks on chip," *Proc. 17th International Conference on VLSI Design*, 2004, pp. 693–696.

76. Y. Gu, Y. Liu, Y. Wang, H. Li, and H. Yang, "NVPsim: A simulator for architecture explorations of nonvolatile processors," *Proc. 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, pp. 147–152.
77. G. A. F. Rebello, G. F. Camilo, M. Potop-Butucaru, M. E. M. Campista, M. D. de Amorim, and L. H. M. K. Costa, "Pcnsim: A flexible and modular simulator for payment channel networks," *Proc. IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2022, pp. 1–2.
78. S. Panagiotou, H. Sidiropoulos, D. Soudris, M. Negrello, and C. Strydis, "EDEN: A high-performance, general-purpose, NeuroML-based neural simulator," *Frontiers in neuroinformatics*, **16** (2022), 724336.
79. H. Hossain, M. Ahmed, A. Al-Nayeem, T. Z. Islam, and M. M. Akbar, "Gpnocsim-a general purpose simulator for network-on-chip," *Proc. 2007 International Conference on Information and Communication Technology*, 2007, pp. 254–257.
80. J. C. Reijenga, W. J. Kingma, D. Berek, and M. Hutta, "GPCSIM: an instrument simulator of polymer analysis by size exclusion chromatography for demonstration and training purposes," *Acta Chimica Slovenica*, **54**(1) (2007), 79–87.
81. A. W. Stedmon, E. Brickell, M. Hancox, J. Noble, and D. Rice, "MotorcycleSim: a user-centred approach in developing a simulator for motorcycle ergonomics and rider human factors research," *Advances in Transportation Studies*, **27** (2012).
82. R. Van Glabbeek, E. H. Teshome, D. Deac, T. Jemal, J. Tiberghien, and K. Steenhaut, "A Building Block for Internet of Things Prototyping," *Proc. IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*, 2022, pp. 1–6.
83. D. Morato, C. Pérez-Gómara, E. Magaña, and M. Izal, "Network simulation in a TCP-enabled industrial internet of things environment-reproducibility issues for performance evaluation," *IEEE Transactions on Industrial Informatics*, **18**(2) (2021), 807–815.
84. S. Idris, T. Karunathilake, and A. Förster, "Survey and comparative study of LoRa-enabled simulators for internet of things and wireless sensor networks," *Sensors*, **22**(15) (2022), 5546.
85. Q. Zhao, Q. Zeng, X. Liu, and H. Xu, "Simulation-based security of function-hiding inner product encryption," *Science China. Information Sciences*, **61**(4) (2018), 048102.
86. Y. Zhao, Y. Wang, H. Zhang, C. Zhang, and C. Yang, "Agent-based Network Security Simulator Nessi2," *Proc. 2015 3rd International Conference on Machinery, Materials and Information Technology Applications*, 2015, pp. 1634–1637.
87. . . , "Modeling of information security system in computer network," , , **1** (2019), 1.
88. Flich, Jose, Tor Skeie, Andres Mejia, Olav Lysne, Pedro Lopez, Antonio Robles, Jose Duato, Michihiro Koibuchi, Tomas Rokicki, and Jose Carlos Sancho. "A survey and evaluation of topology-agnostic deterministic routing algorithms." IEEE Transactions on Parallel and Distributed Systems 23, no. 3 (2011): 405-425.
89. Y. Ben-Itzhak, E. Zahavi, I. Cidon, and A. Kolodny, "HNOCS: modular open-source simulator for heterogeneous NoCs," *Proc. 2012 international conference on embedded computer systems (SAMOS)*, 2012, pp. 51–57.
90. K. Vyas, N. Choudhary, and D. Singh, "Nc-g-sim: A parameterized generic simulator for 2d-mesh, 3d-mesh & irregular on-chip networks with table-based routing," *Global Journal of Computer Science and Technology*, **13**(14) (2013).
91. A. Mello, N. Calazans, and F. Moraes, "ATLAS-an environment for NoC generation and evaluation," *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2011.
92. M. Iqbal, H. Fitriawan, and D. Kurniawan, "Simulasi Kinerja Web Server Pada Jaringan LAN (Local Area Network) Kampus Menggunakan NS2 (Network Simulator 2)," *Jurnal Komputasi*, **10**(2) (2022), 22–47.
93. M. N. O. Sadiku, *Simulation of local area networks*, CRC Press, 2018.
94. E. N. Ekwem and K. Nisar, "An Experimental Study: Using a Simulator Tool for Modelling Campus Based Wireless Local Area Network," *International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC)*, **6**(3) (2014), 35–53.
95. O. Elechi Onyekachi, "Design and Simulation of Wireless Local Area Network for Administrative Office using OPNET Network Simulator: A Practical Approach," *Information and Knowledge Management*, **4**(10) (2014), 27–34.
96. S. K. Esser, D. S. Modha, and T. M. Wong, "Cortical simulator for object-oriented simulation of a neural network," *US Patent 9,020,867*, Apr. 28, 2015.
97. H. A. Fua'ad, S. Shamala, M. Othman, and Z. Zuriati, "A discrete event modeling and simulation of wave division multiplexing unidirectional slotted ring metropolitan area network," *Journal of Computer Science*, **5**(6) (2009), 456.
98. A. Carranza and C. DeCusatis, "Simulation of Metropolitan Area Wavelength Multiplexing Networks for Data Communication," *Proc. Frontiers in Optics*, 2004, FWH43.
99. R. Fernandes, F. Vieira, and M. Ferreira, "Parallel microscopic simulation of metropolitan-scale traffic," *Proc. 46th Annual Simulation Symposium*, 2013, pp. 1–8.
100. A. H. M. Aman, A. H. A. Hashim, A. Abdullah, H. A. M. Ramli, and S. Islam, "Network Simulators Parametric Comparison for Network Mobility Management," *International Journal of Future Generation Communication and Networking*, 2016, pp. 17–28.
101. S. Gadde, J. Chase, and A. Vahdat, "Coarse-grained network simulation for wide-area distributed systems," *Proc. Communication Networks and Distributed Systems Modeling and Simulation Conference*, 2002.
102. K. Schmidt, J. Cerney, R. Becker, P. Duffy, A. Goulart, and J. Morgan, "The Design of a Low-Cost Wide Area Network Simulator," *Proc. International Workshop on Future Multimedia Networking*, 2009, pp. 200–205.
103. S. Siraj, A. Gupta, and R. Badgujar, "Network simulation tools survey," *International Journal of Advanced Research in Computer and Communication Engineering*, f1(4) (2012), 199–206.
104. J. G. McDaniel, "Simulation studies of a wide area health care network," *Proc. Annual Symposium on Computer Application in Medical Care*, 1994, pp. 438.
105. Z. Lu, R. Thid, M. Millberg, E. Nilsson, and A. Jantsch, "NNSE: Nostrum network-on-chip simulation environment," *Proc. SSoCC*, 2005.
106. M. Lis, K. S. Shim, M. H. Cho, P. Ren, O. Khan, and S. Devadas, "DARSIM: a parallel cycle-level NoC simulator,"

2010.

107. H. Hossain, M. Ahmed, A. Al-Nayeem, T. Z. Islam, and M. M. Akbar, “Gpnocsim-a general purpose simulator for network-on-chip,” *Proc. 2007 International Conference on Information and Communication Technology*, 2007, pp. 254–257.

108. X. Zeng, R. Bagrodia, and M. Gerla, “GloMoSim: a library for parallel simulation of large-scale wireless networks,” *Proc. 12th Workshop on Parallel and Distributed Simulation*, 1998, pp. 154–161.

109. M. Goyal, B. Aydas, H. Ghazaleh, and S. Rajasekharan, “CarbMetSim: A discrete-event simulator for carbohydrate metabolism in humans,” *Plos one*, 2020, 15(3), e0209725.

110. S. V. Mallapur and S. R. Patil, “Survey on simulation tools for mobile ad-hoc networks,” *International Journal of Computer Networks and Wireless Communications (IJCNWC)*, 2012, 2(2), pp. 241–248.

111. M. Calvo-Fullana, D. Mox, A. Pyattaev, J. Fink, V. Kumar, and A. Ribeiro, “Ros-netsim: A framework for the integration of robotic and network simulators,” *IEEE Robotics and Automation Letters*, 2021, 6(2), pp. 1120–1127.

112. M. Salama, Y. Elkhatib, and G. Blair, “IoTNetSim: A modelling and simulation platform for end-to-end IoT services and networking,” *Proc. 12th IEEE/ACM International Conference on Utility and Cloud Computing*, 2019, pp. 251–261.

113. A. Wahid-Ul-Ashraf, M. Budka, and K. Musial, “NetSim–The framework for complex network generator,” *Procedia Computer Science*, 2018, 126, pp. 547–556.

114. T. Schneider and H. Schmidt, “NETSIM: A realtime virtual ocean hardware-in-the-loop acoustic modem network simulator,” *Proc. 2018 Fourth Underwater Communications and Networking Conference (UComms)*, 2018, pp. 1–5.

115. N. Agrawal, R. Prashanthi, O. Biçer, and A. Küpçü, “Blocksim-net: A network based blockchain simulator,” *arXiv preprint arXiv:2011.03241*, 2020.

116. G. Körner, C. Birkenhauer, P. Stief, C. Carlowitz, and M. Vossiek, “Efficient bandwidth enhanced multirate radar target simulation,” *Proc. 2022 IEEE/MTT-S International Microwave Symposium-IMS 2022*, 2022, pp. 534–537.

117. H. Samin, L. H. G. Paucar, N. Bencomo, C. M. Hurtado, and E. M. Fredericks, “RDMSim: an exemplar for evaluation and comparison of decision-making techniques for self-adaptation,” *Proc. 2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, 2021, pp. 238–244.

118. F. Tao, J. Cheng, Y. Cheng, S. Gu, T. Zheng, and H. Yang, “SDMSim: a manufacturing service supply–demand matching simulator under cloud environment,” *Robotics and Computer-Integrated Manufacturing*, 2017, 45, pp. 34–46.

119. M. Zhang and S. Jia, “OPNET-Based Simulation of Wireless Ad Hoc Network Protocol,” *Proc. 2019 International Conference on Mathematics, Big Data Analysis and Simulation and Modelling (MBDASM 2019)*, 2019, pp. 143–146.

120. Y. Qin, “Research on OPNET energy simulation application in wireless sensor network,” *Journal of Physics: Conference Series*, 2021, 1952(4), 042022.

121. S. A. Lafta, A. H. Ali, M. M. Kareem, Y. A. Hussein, and A. H. Ali, “Performance simulation of broadband multimedia wireless networks simulation based on OPNET,” *Indonesian Journal Electrical Engineering and Computer Science (IJEECS)*, 2020, 17(1), pp. 1–9.

122. S. Demirci, S. C. Ileri, and S. Duraki, “OMNeT++ Framework for Simulation of Centralized and Distributed Algorithms in Multi-Hop Networks,” *Handbook of Research on Advances in Data Analytics and Complex Communication Networks*, 2022, pp. 1–33.

123. A. Varga, “A practical introduction to the OMNeT++ simulation framework,” *Recent Advances in Network Simulation: The OMNeT++ Environment and its Ecosystem*, 2019, pp. 3–51.

124. J. Hellwege, M. Köstler, and F. Kauer, “Live Monitoring and Remote Control of OMNeT++ Simulations,” *Recent Advances in Network Simulation: The OMNeT++ Environment and its Ecosystem*, 2019, pp. 301–316.

125. C. Obermaier and C. Facchi, “Observations on OMNeT++ Real-Time Behaviour,” *arXiv preprint arXiv:1709.02207*, 2017.

126. A. Varga, “Omnet++ community site,” *http://www.omnetpp.org/*, 2007.

127. J. H. Cowie, D. M. Nicol, and A. T. Ogielski, “Modeling the global internet,” *Computing in Science & Engineering*, 1999, 1(1), pp. 42–50.

128. A. Varga, “Discrete event simulation system,” *Proc. European Simulation Multiconference (ESM’2001)*, 2001, 17.

129. J. Liu, Y. Zhou, and D. Zhang, “Transim: a simulation framework for cache-enabled transparent computing systems,” *IEEE Transactions on Computers*, 2016, 65(10), pp. 3171–3183.

130. M. H. Kabir, S. Islam, M. J. Hossain, and S. Hossain, “Detail comparison of network simulators,” *International Journal of Scientific & Engineering Research*, 2014, 5(10), pp. 203–218.

131. A. Hassan, “VANET simulation,” *Högskolan i Halmstad/Sektionen för Informationsvetenskap, Data-och . . .* , 2009.

132. R. Mangharam, D. Weller, R. Rajkumar, P. Mudalige, and F. Bai, “Groovenet: A hybrid simulator for vehicle-to-vehicle networks,” *Proc. 2006 Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services*, 2006, pp. 1–8.

133. R. Mangharam, D. S. Weller, D. D. Stancil, R. Rajkumar, and J. S. Parikh, “GrooveSim: a topography-accurate simulator for geographic routing in vehicular networks,” *Proc. 2nd ACM International Workshop on Vehicular Ad Hoc Networks*, 2005, pp. 59–68.

134. A. Rensink, “The GROOVE simulator: A tool for state space generation,” *Applications of Graph Transformations with Industrial Relevance: Second International Workshop, AGTIVE 2003, Charlottesville, VA, USA, September 27-October 1, 2003, Revised Selected and Invited Papers 2*, 2004, pp. 479–485.

135. S. Meghzili, A. Chaoui, M. Strecker, and E. Kerkouche, “Transformation and validation of BPMN models to Petri nets models using GROOVE,” *Proc. 2016 International Conference on Advanced Aspects of Software Engineering (ICAASE)*, 2016, pp. 22–29.

136. S. Teijgeler, “Connecting GROOVE to the world using XMI,” *B.S. Thesis, University of Twente*, 2010.

137. J. Cong, K. Gururaj, G. Han, A. Kaplan, M. Naik, and G. Reinman, “MC-Sim: An efficient simulation tool for MPSoC designs,” *Proc. 2008 IEEE/ACM International Conference on Computer-Aided Design*, 2008, pp. 364–371.

138. K.-W. Tseng, “Virtualization Architecture for Reconfigurable Network-on-Chip Systems,” 2015.
139. S. Lotlikar, V. Pai, and P. V. Gratz, “AcENoCs: A configurable HW/SW platform for FPGA accelerated NoC emulation,” *Proc. 2011 24th International Conference on VLSI Design*, 2011, pp. 147–152.
140. D. Wang, N. E. Jerger, and J. G. Steffan, “DART: A programmable architecture for NoC simulation on FPGAs,” *Proc. Fifth ACM/IEEE International Symposium on Networks-on-Chip*, 2011, pp. 145–152.
141. T. Naruko and K. Hiraki, “FOLCS: A lightweight implementation of a cycle-accurate NoC simulator on FPGAs,” *Proc. 3rd International Workshop on Many-core Embedded Systems*, 2015, pp. 25–32.
142. H. M. Kamali and S. Hessabi, “AdapNoC: A fast and flexible FPGA-based NoC simulator,” *Proc. 2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, 2016, pp. 1–8.
143. H. Ahmadinejad, F. Refan, and H. S. Sarjoughian, “NoC-DEVS Simulator.”
144. S. Prabhu, “OCIN_TSIM-a DVFS aware simulator for NoC design space exploration and optimization,” *Ph.D. Thesis, Texas A & M University*, 2010.
145. S. N. Ved, S. Singh, and J. Mekie, “Pane: Pluggable asynchronous network-on-chip simulator,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2019, 15(1), pp. 1–27.
146. X. Zhou, P. Hao, and D. Liu, “PCCNoC: Packet Connected Circuit as Network on Chip for High Throughput and Low Latency SoCs,” *Micromachines*, 2023, 14(3), 501.
147. A. Ochoa, S. González, E. Moriel, J. Arreola, and F. García, “Improving decision-making in a business simulator using TOPSIS methodology for the establishment of reactive stratagems,” *Nature-inspired Design of Hybrid Intelligent Systems*, 2017, pp. 809–819.
148. P. Levis, N. Lee, M. Welsh, and D. Culler, “TOSSIM: Accurate and scalable simulation of entire TinyOS applications,” *Proc. 1st International Conference on Embedded Networked Sensor Systems*, 2003, pp. 126–137.
149. E. A. León, T. D’Hooge, N. Hanford, I. Karlin, R. Pankajakshan, J. Foraker, C. Chambreau, and M. L. Leininger, “TOSS-2020: a commodity software stack for HPC,” *Proc. SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020, pp. 1–15.
150. Rezazad, Mostafa, and Hamid Sarbazi-Azad. ”The effect of virtual channel organization on the performance of interconnection networks.” 19th IEEE international parallel and distributed processing symposium. IEEE, 2005.
151. A. Derhab, F. Ounini, and B. Remli, “MOB-TOSSIM: An extension framework for TOSSIM simulator to support mobility in wireless sensor and actuator networks,” *Proc. 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, 2012, pp. 300–305.
152. J. Bien, “The simulator: An engine to streamline simulations. Submitted,” 2016.
153. W. Li, X. Zhang, W. Tan, X. Zhou, and others, “H-tossim: Extending tossim with physical nodes,” *Wireless Sensor Network*, 2009, 1(4), pp. 324.
154. P. Chhimwal, D. S. Rai, and D. Rawat, “Comparison between different wireless sensor simulation tools,” *IOSR Journal of Electronics and Communication Engineering*, 2013, 5(2), pp. 54–60.
155. L. Jain, B. Al-Hashimi, M. S. Gaur, V. Laxmi, and A. Narayanan, “NIRGAM: a simulator for NoC interconnect routing and application modeling,” *Proc. Design, Automation and Test in Europe Conference*, 2007, pp. 16–20.
156. S. Bhat and K. R. Kamath, “Effective Learning with Usage of Simulators–A Case of NCTUns Simulator in Computer Networks,” *International Journal of Scientific Research and Modern Education*, 2016, 1, pp. 415–420.
157. P. Tyagi and D. Dembla, “Performance analysis and quality-of-service monitoring of protected and unprotected TCP networks using NCTUns simulator,” *Proc. 2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 273–277.
158. S.-Y. Wang and K.-C. Liao, “Innovative network emulations using the NCTUns tool,” *Computer Networking and Networks*, 2006, pp. 157–187.
159. S.-Y. Wang, P.-F. Wang, Y.-W. Li, and L.-C. Lau, “Design and implementation of a more realistic radio propagation model for wireless vehicular networks over the NCTUns network simulator,” *Proc. 2011 IEEE Wireless Communications and Networking Conference*, 2011, pp. 1937–1942.
160. N. Choudhury, T. Mehta, T. L. Wilmarth, E. J. Bohm, and L. V. Kalé, “Scaling an optimistic parallel simulation of large-scale interconnection networks,” *Proc. Winter Simulation Conference, 2005*, 2005, pp. 10–pp.
161. H. Suhendi, M. Fahmi, and I. Saleh, “Analisa dan Perancangan Jaringan WAN dengan Sisco Gateway Load Balancing Protocol (GLBP) Menggunakan Simulator GNS3,” *Jurnal Computech & Bisnis (e-Journal)*, 2022, 16(2), pp. 79–85.
162. J.-I. Castillo-Velázquez and A. Delgado-Villegas, “GNS3 Limitations when Emulating Connectivity and Management for Backbone Networks: A Case Study of CANARIE,” *Proc. 2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2020, pp. 1–4.
163. N. Agarwal, T. Krishna, L.-S. Peh, and N. K. Jha, “GARNET: A detailed on-chip network model inside a full-system simulator,” *Proc. 2009 IEEE International Symposium on Performance Analysis of Systems and Software*, 2009, pp. 33–42.
164. S. Yoon and Y. B. Kim, “A design of network simulation environment using ssfnet,” *Proc. 2009 First International Conference on Advances in System Simulation*, 2009, pp. 73–78.
165. C. Baek, E. Im, E. Park, K. Choi, and G. Jung, “Design and implementation of firewall simulation based on SSFNet,” *Proc. The 6th International Conference on Advanced Communication Technology, 2004*, 2004, 1, pp. 312–316.
166. N. Baldo, M. Miozzo, M. Requena-Esteso, and J. Nin-Guerrero, “An open source product-oriented LTE network simulator based on ns-3,” *Proc. 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, 2011, pp. 293–298.
167. P. Jörke, T. Gebauer, and C. Wietfeld, “From LENA to LENA-NB: Implementation and Performance Evaluation of NB-IoT and Early Data Transmission in ns-3,” *Proc. 2022 Workshop on ns-3*, 2022, pp. 73–80.

168. B. Bojovic, Z. Ali, and S. Lagen, “Ns-3 and 5G-LENA extensions to support dual-polarized MIMO,” *Proc. 2022 Workshop on ns-3*, 2022, pp. 1–9.
169. B. Bojović, S. Lagén, and L. Giupponi, “Realistic beamforming design using SRS-based channel estimate for ns-3 5G-LENA module,” *Proc. 2021 Workshop on Ns-3*, 2021, pp. 81–87.
170. S. Nayak, “ns-3 Simulation Based Exploration of LTE Handover Optimization,” *EAI Endorsed Transactions on Mobile Communications and Applications*, 2022, 7(4), pp. e4–e4.
171. Sadrosadati, M., Mirhosseini, A., Akbarzadeh, N., Aghilinasab, H., & Sarbazi-Azad, H. (2022). An efficient DVS scheme for on-chip networks. In Advances in Computers (Vol. 124, pp. 21-43). Elsevier.
172. Ghozati, S. A., and H. C. Wasserman. ”The k-ary n-cube network: modeling, topological properties and routing strategies.” Computers & electrical engineering 25.3 (1999): 155-168.
173. Stallings, William. Local and metropolitan area networks. Prentice Hall PTR, 2000.
174. Yoon, Y. J., Concer, N., Petracca, M., & Carloni, L. P. (2013). Virtual channels and multiple physical networks: Two alternatives to improve NoC performance. IEEE Transactions on computer-aided design of integrated circuits and systems, 32(12), 1906-1919.
175. Salem, Amer O. Abu, and Hebatallah Awwad. ”Mobile ad-hoc network simulators, a survey and comparisons.” Int. J. P2P Netw. Trends Technol.(IJPTT) 9 (2014): 12-17.
176. A. Burns, J Harbin, and L. S. Indrusiak. ”A wormhole noc protocol for mixed criticality systems.” 2014 IEEE Real-Time Systems Symposium. IEEE, 2014.
177. J. Gomez, E. F. Kfoury, J. Crichigno, and G. Srivastava, “A survey on network simulators, emulators, and testbeds used for research and education,” *Computer Networks*, 2023, 237, pp. 110054.
178. D. Abinoja, R. A. Bedruz, K. L. Jovellanos, M. A. Roque, and M. L. Torregoza, “Double Bi-quadantenna for WiGig applications,” *Proc. 2015 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, 2015, pp. 1–4.
179. I. Ahmed, H. Khammari, M. K. Shahid, and A. Nawaz, “Crowd management using low energy mmwave WiGig with point-to-massive-points communications,” *Proc. 2020 Advances in Science and Engineering Technology International Conferences (ASET)*, 2020, pp. 1–5.
180. W. He, K. Wu, Y. Zou, and Z. Ming, “WiG: WiFi-based gesture recognition system,” *Proc. 2015 24th International Conference on Computer Communication and Networks (ICCCN)*, 2015, pp. 1–7.
181. Y.-Y. Li, C.-Y. Li, W.-H. Chen, C.-J. Yeh, and K. Wang, “Enabling seamless WiGig/WiFi handovers in tri-band wireless systems,” *Proc. 2017 IEEE 25th International Conference on Network Protocols (ICNP)*, 2017, pp. 1–2.
182. L. Sitanayah, C. J. Sreenan, and S. Fedor, “A Cooja-based tool for coverage and lifetime evaluation in an in-building sensor network,” *Journal of Sensor and Actuator Networks*, 2016, 5(1), pp. 4.
183. R. Piyare, T. Istomin, A. L. Murphy, and others, “WaCo: A Wake-Up Radio COOJA Extension for Simulating Ultra Low Power Radios,” *Proc. EWSN*, 2017, pp. 48–53.
184. D. Lee, G. Lee, D. Kwon, S. Lee, Y. Kim, and J. Kim, “Flexon: A flexible digital neuron for efficient spiking neural network simulations,” *Proc. 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, 2018, pp. 275–288.
185. M. Xu, G. Li, W. Yang, and W. Tian, “Flexcloud: A flexible and extendible simulator for performance evaluation of virtual machine allocation,” *Proc. 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, 2015, pp. 649–655.
186. N. Bleier, C. Lee, F. Rodriguez, A. Sou, S. White, and R. Kumar, “FlexiCores: low footprint, high yield, field reprogrammable flexible microprocessors,” *Proc. 49th Annual International Symposium on Computer Architecture*, 2022, pp. 831–846.
187. A. F”orster, K. Garg, D. Puccinelli, and S. Giordano, “Flexor: User friendly wireless sensor network development and deployment,” *Proc. 2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2012, pp. 1–9.
188. J. M. Garrido and J. M. Garrido, “Introduction to flexsim,” *Object Oriented Simulation: A Modeling and Programming Perspective*, 2009, pp. 31–42.
189. J. Allison, “Simulation-based learning via cisco packet tracer to enhance the teaching of computer networks,” *Proc. 27th ACM Conference on Innovation and Technology in Computer Science Education Vol. 1*, 2022, pp. 68–74.
190. A. Bhola, A. Jain, B. D. Lakshmi, T. M. Lakshmi, and C. D. Hari, “A wide area network design and architecture using Cisco packet tracer,” *Proc. 2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, 2022, pp. 1646–1652.
191. K. Koutlia, B. Bojovic, Z. Ali, and S. Lag’en, “Calibration of the 5G-LENA system level simulator in 3GPP reference scenarios,” *Simulation Modelling Practice and Theory*, 2022, 119, pp. 102580.
192. H. Assasa, N. Grosheva, T. Ropitault, S. Blandino, N. Golmie, and J. Widmer, “Implementation and evaluation of a WLAN IEEE 802.11 ay model in network simulator ns-3,” *Proc. 2021 Workshop on ns-3*, 2021, pp. 9–16.
193. F. ”Osterlind, “A sensor network simulator for the Contiki OS,” 2006, *Swedish Institute of Computer Science*.
194. N. Finne, J. Eriksson, T. Voigt, G. Suciu, M.-A. Sachian, J. Ko, and H. Keipour, “Multi-trace: multi-level data trace generation with the Cooja simulator,” *Proc. 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2021, pp. 390–395.
195. D. Jabba and P. Acevedo, “Vitool-BC: Visualization tool based on Cooja simulator for WSN,” *Applied Sciences*, 2021, 11(16), pp. 7665.
196. H. S. Zenalabdin, A. Buhari, and T. E. Nyamasvisva, “Performance analysis of IoT protocol stack over dense and sparse mote network using Cooja simulator,” *Journal of Physics: Conference Series*, vol. 1529, no. 5, 2020, pp. 052007.
197. N. Gwangwava, T. B. Mubvirwi, and others, “Design and simulation of IoT systems using the Cisco packet tracer,” *Advances in Internet of Things*, vol. 11, no. 02, 2021, pp. 59.

198. A. Monika, M. Shekhar, and A. Kaushik, "Network simulators for next generation networks: an overview," *Int J Mob Netw Commun Telemat*, vol. 4, 2014, pp. 39–51.
199. R. Barr, "Swans-scalable wireless ad hoc network simulator," *User Guide*, 2004.
200. J. Lessmann, T. Heimfarth, and P. Janacik, "Shox: An easy to use simulation platform for wireless networks," *Proc. Tenth International Conference on Computer Modeling and Simulation (UKSIM 2008)*, 2008, pp. 410–415.
201. S. Siraj, A. Gupta, and R. Badgujar, "Network simulation tools survey," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 1, no. 4, 2012, pp. 199–206.
202. N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, and others, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, 2011, pp. 1–7.
203. A. Nayebi, S. Meraji, A. Shamaei, and H. Sarbazi-Azad, "Xmulator: a listener-based integrated simulation platform for interconnection networks," *Proc. First Asia International Conference on Modelling & Simulation (AMS'07)*, 2007, pp. 128–132.
204. S.-Y. Wang and C.-C. Lin, "NCTUns 6.0: A Simulator for Advanced Wireless Vehicular Network Research," *Proc. 2010 IEEE 71st Vehicular Technology Conference*, 2010, pp. 1–2.
205. J. Power, J. Hestness, M. S. Orr, M. D. Hill, and D. A. Wood, "gem5-gpu: A heterogeneous CPU-GPU simulator," *IEEE Computer Architecture Letters*, vol. 14, no. 1, 2014, pp. 34–36.
206. S. H. Nikounia and S. Mohammadi, "Gem5v: a modified gem5 for simulating virtualized systems," *The Journal of Supercomputing*, vol. 71, 2015, pp. 1484–1504.
207. A. K. Singh, "Survey of Network-on-Chip simulators."
208. Galaxy Technologies LLC, "GNS3 Documentations," 2024. [Online]. Available: https://docs.gns3.com/. [Accessed: 02-Jul-2024].
209. G. Ewing, K. Pawlikowski, and D. McNickle, "Akaroa-2: Exploiting network computing by distributing stochastic simulation," 1999, *SCSI Press*.
210. S. Keshav, "REAL: A network simulator," 1988, *University of California Berkeley, Calif, USA*.
211. N. Kotilainen, M. Vapa, T. Keltanen, A. Auvinen, and J. Vuori, "P2PRealm-peer-to-peer network simulator," *Proc. 2006 11th International Workshop on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks*, 2006, pp. 93–99.
212. V. Yau and K. Pawlikowski, "Akaroa: a package for automatic generation and process control of parallel stochastic simulation," *Proc. 16th Australian Computer Science Conference*, 1993, pp. 71–82.
213. G. F. Riley, "The Georgia Tech network simulator," *Proc. ACM SIGCOMM Workshop on Models, Methods and Tools for Reproducible Network Research*, 2003, pp. 5–12.
214. EVE-NG Ltd, "EVE-NG Website," 2024. [Online]. Available: https://www.eve-ng.net/. [Accessed: 02-Jul-2024].
215. A. Sobeih, W.-P. Chen, J. C. Hou, L.-C. Kung, N. Li, H. Lim, H.-Y. Tyan, and H. Zhang, "J-Sim: a simulation environment for wireless sensor networks," *Proc. 38th Annual Simulation Symposium*, 2005, pp. 175–187.
216. J. Obstfeld, S. Knight, E. Kern, Q. S. Wang, T. Bryan, and D. Bourque, "VIRL: the virtual internet routing lab," *Proc. 2014 ACM Conference on SIGCOMM*, 2014, pp. 577–578.
217. L. Hogie, I. Tahiri, D. Papadimitriou, and F. Majorczyk, "DRMSim: a network simulator for the investigation of routing schemes User manual," 2010.
218. S.-Y. Wang and H.-T. Kung, "A simple methodology for constructing extensible and high-fidelity TCP/IP network simulators," *Proc. IEEE INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, 1999, pp. 1134–1143.
219. P. Janacik, J. Lessmann, and M. Karch, "Distributed Simulation Environment for the ShoX Network Simulator," *Proc. 2010 Sixth International Conference on Networking and Services*, 2010, pp. 206–211.
220. W. J. Dally and B. P. Towles, "Principles and practices of interconnection networks," 2004, *Elsevier*.
221. GAPH, "Gaph main page," [Online]. Available: http://www.inf.pucrs.br/œgaph/.
222. C. Ababei and N. Mastronarde, "Benefits and costs of prediction-based DVFS for NoCs at router level," *Proc. 2014 27th IEEE International System-on-Chip Conference (SOCC)*, 2014, pp. 255–260.
223. G. Kramer, "Synthetic self-similar traffic generation," 2014. [Online]. Available: http://glenkramer.com/ucdavis/trf_research.html/.
224. D. M. Chapiro, "Globally-asynchronous locally-synchronous systems," *Ph. D. Thesis*, 1984.