`nextGSIM`: Towards Simulating Network Resource Management for Beyond 5G Networks

Alba Jano[†*], Mehmet Mert Bese[‡*], Nitinder Mohan[‡], Wolfgang Kellerer[†], Jörg Ott[‡]
[†] Chair of Communication Networks, Technical University of Munich, Germany
[‡] Chair of Connected Mobility, Technical University of Munich, Germany
{alba.jano, mehmetmert.bese, wolfgang.kellerer}@tum.de, {mohan, ott}@in.tum.de

***Abstract*—Researchers have already begun experimenting with next-generation cellular technologies and algorithms to enable use cases that lie beyond the scope of the current 5G standard, e.g. XR, smart factories, AI networks ops, etc. The common denominator requirement of such scenarios is the joint (coupled) operation of radio channel and edge computing resources within the core network. While there are numerous tools that allow experimenting with various aspects of radio resource management and computing resource management individually, there is a lack of solutions that enable researchers to prototype and evaluate applications and technologies dependent on both aspects simultaneously. In this work, we present `nextGSIM`, a 5G and beyond network simulator that realistically models the radio access network and edge network jointly to provide an end-to-end service to various user devices running microservice-based application workloads. We detail our design decisions and modular architecture of `nextGSIM` which resembles real-world setup of cellular networks, enabling effective and detailed simulations of resource management algorithms. We demonstrate the effectiveness and capabilities of `nextGSIM` through indoor factory case study wherein we evaluate widely regarded radio and edge resource management algorithms. We compare these against a joint radio-compute scheduler which emphasizes the need and benefits of joint resource allocation decision making, which is only possible through tools such as `nextGSIM`.**

I. Introduction

As majority of studies within 5G are focusing on understanding the real-world operations and performance of the cellular network [1], [2], simultaneously, researchers and standardization bodies alike have started examining the next-generation use-cases that cannot be fulfilled by current 5G specifications [3], [4]. In addition to the enabling latency and compute critical applications [5] such as extended reality (XR) [6], [7], autonomous drone swarms [8], smart industry and cities [9], etc., 5G++[1] networks are expected to embody autonomous compute and radio resource management using machine learning (ML)/ artificial intelligence (AI) and deeply integrate compute-capabilities within network operations [4], [10], [11]. Within this scope, AI integrated under the digital twin concept enables dynamic, context-aware, and demand-aware network control and application orchestration [12]. While core functionalities and technologies enabling 5G++ are still being speculated, the general conception hints at a network that will likely to imbibe different frequency radio channels and different configurations of edge servers for enabling user-facing applications and intelligent core network management [13], [14]. In this scope, O-RAN and MEC [15]–[17] architectures are envisioned to enable the development of dynamic radio resource allocation and service orchestration algorithms that work hand-in-hand in the future.

Despite the growing interest and research potential, we argue that there are no suitable simulators available that allow interested researchers to "realistically" examine their end-to-end resource management algorithms and create rich simulated models for their use cases. Majority of existing simulators provide detailed modeling of either the wireless radio access [18]–[20] or the edge computing and service deployment aspects [21]–[24]. There have been a few efforts in recent past that intend to combine both network and edge layers together [25]–[27], albeit at an expense of simplifying either the radio access networks (RAN) and channel modeling or the computing aspects. While both NS-3 and OMNET++ extensively model the radio network, they lack the capability to jointly allocate radio and edge resources through algorithms. Simu5G [28], an OMNET++ library for simulating 5G and mobile edge computing, does not implement microservices, service orchestration and joint operation. Instead, it implements ETSI compliant interfaces between monolithic applications, edge servers and the core network. The primary reason being the richness and model complexity inherent to both RAN and edge computing operations which can bloats the design, implementation and usage of the simulator if accurately incorporated [29]–[31].

In this work, we plug this outstanding gap via `nextGSIM`, a modular and extensible simulator for experimenting (and designing) next-generation of communication networks. The core contribution of `nextGSIM` is enabling coupled simulations incorporating both (radio) network and edge computing parameters of 5G++, which enables researchers to play with several new problem areas such as autonomous network control, resource management and service orchestration. We achieve this by modeling the RAN in detail, incorporating different use case scenarios, channel models, and network protocols defined by 3GPP [29]–[31] (see §III-A). `nextGSIM` utilizes a software defined RAN (SD-RAN) controller architecture [32]–[34] to separate the RAN control and data plane, enabling flexibility and programmability for efficient control of the dynamic RAN environment. In parallel, the edge com-

[*]Authors with equal contribution.
[1]We take a broad practical stance on next generation cellular technologies and refer to them as 5G++ instead of 6G.

puting infrastructure is modeled as heterogeneous servers with different resource capacities hosting distributed microservice-based applications (modeled by directed acyclic graphs) with strict networking and processing needs catering to end-user devices (see §III-B). The deployment of microservices on edge servers is controlled by an orchestrator, which allows for dynamic scaling and migration operations as user load in the system increases/shifts. Additionally, the RAN and edge simulator blocks are inherently decoupled and simulate in-step by passing control messages via defined APIs at fine-grained time-step intervals. This allows researchers to accurately simulate complex RAN and edge computing scenarios over different physical machines, which (i) avoids shared server resource contention for large-scale simulations, (ii) aligns with realistic deployment of communication networks, where the edge servers are separated by the underlying RAN network and (iii) simplifies collection and analysis of data for ML/AI resource management, especially due to the simulator focus on the network resources.

We demonstrate the simulation capabilities and scalability of the `nextGSIM` by simulating several scenarios, using standardized channel models, mobile devices and load generation (see §IV). Additionally, we dig in deeper to the resource management problems in one of the leading use case for 5G++, indoor factory floor operations, and experiment with several separate and joint radio and edge resource scheduling algorithms. Our investigation showing that joint RAN and edge resource management algorithms can lead to significant performance benefits and satisfying strict quality of service (QoS) requirements of target 5G++ use cases, highlights the utility of `nextGSIM` which allows realistic experimentation of RAN and edge components. NextGSim is publically available under MIT license[1], along with a documentation and simulation scripts to reproduce results in this work.

II. Related Work

The continuous advancement of communication networks is accompanied by the development of simulators, which serve as testing tools for novel algorithms. These simulators can be classified based on the network layer they simulate, namely, computing layer [21]–[24], [35], network layer [18]–[20], or end-to-end simulators [25]–[27], providing support for both the network and computing layer of a communication network. CloudSim [35] is a widely used cloud computing simulator that evaluates resource provisioning algorithms by simulating large-scale cloud computing systems in a physical node. To support a wide range of IoT applications, IoTSim [21] models cloud computing and application provisioning through the MapReduce framework. The emergence of fog and edge computing triggered the expansion of CloudSim to iFogSim [22] and EdgeCloudSim [23]. iFogSim [22] leverages CloudSim to simulate internet of things (IoT) applications in fog environments, incorporating a simplistic network model with fixed delays and static device locations. On the other hand, EdgeCloudSim [23] introduces aspects lacking in CloudSim, such as edge orchestrators, mobile nodes with mobility support, and WLAN/WAN communication models. YAFS [24] serves as a baseline for our edge computing layer, offering simplicity in utilization and addressing similar edge computing problems, such as dynamic application placement, load balancing, and service scheduling. YAFS has limited extensibility to realistic scenarios and simplifies communication modeling in the network, making it unsuitable for mobile edge computing scenarios. While these simulators comprehensively model cloud and edge computing processes, they lack the networking layer required for realistic communication and adopt a multi-tier fog-cloud architecture rather than the operator-enabled edge.

The networking layer is extensively modeled by NS-3 [18], OMNET++ [19], and Opnet [20] simulators. However, these simulators exhibit a complex architecture for simulating the communication networks and require significant efforts to implement the edge computing layer, relevant to establish end-to-end resource management. In the context of end-to-end IoT services and networking, IoTNetSim [25] inherits from CloudSim, however, models different types of network connections with signal types, capacity, and power models, including 3G, Wi-Fi, Bluetooth, LoRa, and Zigbee. IoTSim-Framework [26] extends the simulated network protocols to include 4G. The authors enhance the features of IoT devices with battery, synchronization, and security protocols. Energy modules of mobile devices are included also at FogNetSim++ [27] which extends OMNET++ simulator. It utilizes a simple state diagram as a baseline and considers queuing traffic at the fog computing layer.

Despite the considerable efforts devoted to achieving realistic, scalable, and flexible simulation of the edge and network layers, the evolution towards 6G introduces new challenges with the new protocol procedures, applications, scenarios, and requirements. Emergence of low-latency applications that 6G will enable require efficient and intelligent resource management algorithms. Consequently, there is a growing demand for more realistic simulations, aligned with the 3GPP standardization, supporting scalable scenarios, flexible configuration of applications with different QoS requirements and controllers that can efficiently manage network resources.

III. `nextGSIM` Architecture

We believe that there are two unique aspects of `nextGSIM` that makes it suitable for simulating beyond-5G networks. Firstly, `nextGSIM` couples the RAN and edge computing aspects together, allowing researchers to envision complex simulation interactions that not only incorporate mobile user/IoT devices utilizing novel wireless mediums, but also compute applications processing the data generated by those devices in edge servers. Secondly, we design and implement `nextGSIM` as modular and extensible such that it can cope with the growing requirements of the field. Fig. 1 shows its architecture. Generation of packets/tasks are simulated by **Devices** (shown in red) and the communication between base stations and

[1]https://github.com/6G-Future-Lab-Bavaria/NextGSim

TABLE I
SUPPORTED FEATURES IN THE SIMULATOR.

Parameter	**Example values**
Number of gNBs	1, 2, ...N*
Number of UEs	1, 2, ...M*
Simulation Time	1, 2, ...T*
TTI Duration [ms]	1, 0.5, 0.25, 0.125, 0.0625
Scenario Area Dimensions [m]	Length: 120, Width: 80
Communication Type	Uplink, Downlink
Operating Frequency	FR1, FR2
Bandwidth [MHz]	5 - 400
Subcarrier Spacing [kHz]	15, 39, 60, 120
Scheduler Type	Proportional-Fair, Max-Rate, Round-Robin
Scheduling Granularity [ms]	1, 10, 100
Applications	Smart Factory, Smart City, Autonomous Drone
Application Delay Tolerance [ms]	10, 100, 500
Application Data Size [kbit]	1, 20, 50 ...D*
Mobility	Random Waypoint
Traffic Model	Full Buffer, FTP3, Web Browsing, Video Streaming
Device Activation model	Uniform, Beta distribution
Channel Model	Indoor Factory, Outdoor
Interference	True
Task Offloading	True
Task Complexity [cycle/bit]	50, 100
RRC State Switching	True
Scenario Visualization (GUI)	True
Number of MEC Servers	1, 2, ...S*
Backhaul Topology	Flexible
Service CPU Share Requirement [cores]	0.3, 1
Service Memory Requirement [GB]	4, 8
Latency Awareness of Microservices	True, False
CPU Clock Speed of a Server [MIPS]	412000
Memory of a Server [GB]	32

* Parameter that spans to large values depending on the scenario.

edge servers are simulated by **Backhaul Network** (shown in pink). The RAN aspects are jointly simulated by **RAN data plane** entity (shown in blue) and **RAN control plane** implemented in SD-RAN controllers [36] (shown in green). On the other hand, the application deployment and operation aspects are simulated by **edge computing orchestrator** (which acts as compute control plane) and the **edge servers** (read compute data plane) shown in violet. Joint management server is denoted with orange. We implement `nextGSIM` in `Python` language spanning ≈12000 LOC. The simulator is coordinated using `SimPy` [37] library, which is a well established library for discrete-event based simulations like in `nextGSIM`. Routing in backhaul network is implemented using `networkX` library [38]. Users can submit simulations using a configuration file with dynamic features as shown in Table I. Radio and edge environments, RAN and edge algorithms in use, applications users are running and parameters related to them, granularity of the simulation and visualisation options can be configured through this file. The RAN and the compute blocks of the simulator are also decoupled as two separate services which can be hosted on different physical machines. The two components synchronize with each other at every simulation time step over the network (see §III-C). Our design decision allows `nextGSIM` to (i) simulate complex RAN and edge computing scenarios without requiring a large compute-capable server to house the entire simulator, and (ii)

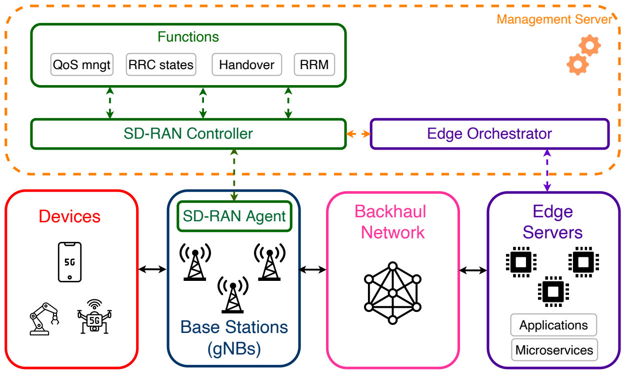

Fig. 1. `nextGSIM` simulator architecture.

independently extend the RAN and edge components with additional features without impacting the system at-large. We next explain the make-up of different `nextGSIM` components.

A. RAN Simulation

RAN Data Plane: The **RAN data plane entity** follows a time based approach, with time granularity defined by the transmission time interval (TTI). The latter depends on the configuration of radio resource grid with the respective bandwidth and subcarrier spacing. The default value of the subcarrier spacing is 15 kHz, which maps to a TTI of 1 ms. The data plane is simulated based on the scenario chosen in the configuration file. Currently, `nextGSIM` can simulate outdoor, indoor, and multiple indoor factory scenarios defined in 3GPP specifications [29], where the respective parameters are specified in the *Scenario* module. The data plane is further composed of the following modules.

▷ Device module: The module is responsible for creating mobile devices and managing their mobility, uplink data generation and radio resource control (RRC) state. Each device has x and y coordinates which are updated by the user-defined mobility model, e.g. random waypoint [39]. The mobile devices can also generate uplink traffic (through packet inter-arrival rates and sizes) based on the traffic models described by 3GPP standardization in [40]. At simulation time $t = 0$, all devices are in "power-off" mode and are activated following uniform or beta distribution functions [41]. The activation switches the devices to RRC Connected state. The device state, RRC Connected, RRC Inactive or RRC Idle is dependent on the generated traffic and the protocol timers [30].

▷ gNB module: gNB module handles the positions of the gNBs, to provide a full coverage for the connected devices. The gNBs have a static x and y coordinates calculated in the initial scenario configuration [29]. Additionally, they contain the radio resource pool defined by the selected bandwidth and subcarrier spacing to be allocated to the users [42].

▷ Channel model module: The module simulates the quality of signaling between the devices and the gNBs, which is highly scenario-dependent. This module calculates the channel quality based on the outdoor, indoor and factory indoor channel models defined by 3GPP [31], and accounts for the small-scale fading and large-scale fading components.

Control Plane: The **SD-RAN entity** acts as RAN control plane and is responsible for managing radio resources, devices RRC states, handover procedures, and fulfilling the RAN QoS requirements. The controller decisions are transmitted to the user plane by SD-RAN agents via APIs.

▷ Radio resource management (RRM) module: The simulator allocates the available radio resources to the UEs using the classical scheduling techniques, such as round robin, throughput proportional fair and max-rate allocation. The round robin technique assigns the available radio resources recursively to the connected users, without accounting for the channel conditions. Meanwhile, the throughput proportional fair allocates the resource to achieve fairness on the throughput over time and the max-rate scheduler allocates the resources to the users with the best channel quality. The modular structure of the module allows extending to sub-optimal and optimal resource allocation algorithms.

▷ Radio resource connection (RRC) module: The module implements the transition of the device between RRC Connected, RRC Inactive and RRC Idle states, depending on parameters such as RRC inactivity timer, discontinuous reception cycle length, and traffic patterns of the devices [30]. As a result, we avoid scenarios where all the devices remain in the connected state and create the baseline for sustainability studies at the device side. The devices can transition to more energy-efficient states for reducing battery drainage.

▷ Handover module: The module enables mobility scenarios in `nextGSIM` as it allows devices to switch the serving gNBs offering better signal quality. The handover module is responsible for initial connection of the devices and maintaining the connection of mobile devices with the best serving gNB.

B. Edge Computing & Backhaul Simulation

`nextGSIM` also allows researchers to simulate an extensible edge infrastructure which processes compute-requests of applications catering to the RAN-connected UEs. In addition to the server capacity configuration, `nextGSIM` also incorporates "realistic" resource management, application deployment and migration resembling real-world counterparts. For example, the *orchestrator* class enables novel application orchestration methods, such as horizontal/vertical auto-scaling and service migration algorithms, whereas *microservice* class can model distributed applications composed of several interconnected microservices [43]. Such simulations are supported by the following modules.

▷ Application module: The module consists of three classes: *application*, *microservice* and *message*. Applications are modeled as directed acyclic graphs (DAG) of microservices, which is a widely accepted representation [44]. Microservices are characterized by their required CPU share and memory, input and output messages to other connected services in the graph. We take inspiration from state-of-the-art [45], [46] for designing the *orchestrator* which utilizes dynamic performance metrics of the microservices, such as experienced latency, queue length, etc. for scheduling decisions. Similarly, microservices can also utilize control information from the orchestrator to modify their packet processing behavior. Messages are characterized by their size (in bytes) and complexity (in number of instructions). Their size affects latency, whereas their complexity can affect the processing time.

▷ Entity module: The module consists of the three entity (read nodes) classes: *computing*, *non-computing* and *auxiliary*. Computing entities are processing-capable, e.g. mobile devices and edge servers. Non-computing entities are gNBs and orchestrator. Note that orchestrator is a logical entity that gathers information from servers, microservices and SD-RAN controller for optimal control decisions. CPUs and VMs are auxiliary entities to be placed within the computing entities to model computation and partition available resources.

▷ Network module: Network module is used to model the wired part of the network, namely the backhaul. Wired links are modeled by their bandwidth and latency in the Link class. Topology class is used to store the underlying network topology and as an extension, microservice topology as an overlay topology. Routing class is used to model routing algorithms in the topologies.

C. `nextGSIM` Workflow

Researchers can configure their simulation scenarios using easy-to-manage JSON `conf` files, which allows them to define fine-grained parameters to be considered by `nextGSIM`. At each TTI, the RAN and edge blocks of `nextGSIM` exchange information with each other, mimicking the communication between SD-RAN controllers and edge orchestrators in real-world systems [32]. Information from the RAN includes packets received by the gNB along with other wireless connection statistics such as, sender devices, receiving base station, wireless packet latencies and throughput, packet size (in kB), complexity of the task (in cycles/bit) and the tolerated delay for the processing of the packet (in ms). Information shared by the edge side consists of the load of the edge servers (available CPU and memory), processing status of user tasks in percentage and the respective application instance responsible for processing. Radio control and service orchestration algorithms can make use of the radio/edge side information with pre-defined intervals mimicking the near-real time applications (xApps) that are present in the O-RAN systems [15]. We also offer a graphical user interface (GUI) which provides a real-time visualization of the simulated scenario, including placement of gNBs (base stations), mobility of users, etc.

IV. Evaluation

We now demonstrate the functionality of `nextGSIM` and showcase its flexibility, fidelity (compared to real-world deployments), and its ability to meet QoS (latency) requirements under varying numbers of UEs and MEC servers (Fig. 6). We focus on an indoor factory scenario depicted in Fig. 2, which is a leading use case for beyond 5G communications [29].

The scenario involves several IoT sensors attached to factory machines (a.k.a. user within paper context) that offload strict latency-constrained compute tasks to nearby edge servers via

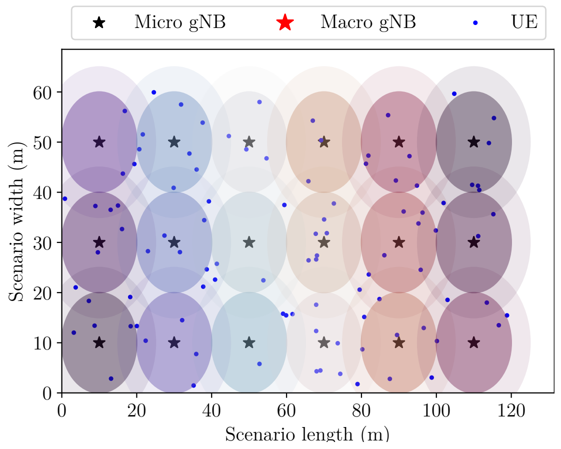

Fig. 2. Indoor factory scenario in a factory hall, where 18 equally distanced gNBs serve in total 100 UEs having UL data to transmit to the network.

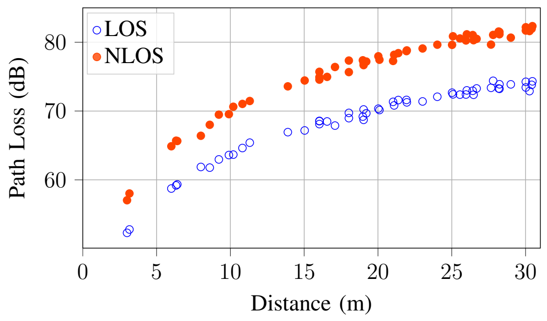

Fig. 3. Indoor factory channel model generated for a single gNB operating at 3.7 GHz frequency, which aligns with real-world measurements in [47].

several available gNBs. To demonstrate the accuracy of the simulated channel, we compare the simulated indoor factory channel with realistic measurements obtained from a factory environment by a previous study [47]. We mimic their setup by placing a single gNB at a height of 185 cm, serving UEs with a height of 144 cm positioned at 75 different positions on the factory floor, with a maximum distance of 30 m. Fig. 3 illustrates the simulated path loss for line-of-sight (LOS) and non-line-of-sight (NLOS), which closely aligns with the measurement results in [47].

For the rest of the study, we assume that all users run the same application and send packets (tasks) every 10 ms for processing, where packet size and task complexities vary over the simulation time. Consequently, we test several algorithms that allocate the limited radio resources and computing resources for the incoming task requests separately and jointly. We employ seeded `nextGSIM` that maintains consistent scenario conditions, including channel conditions, user mobility, and generated tasks when testing various algorithms. We use the value of running TTI as the seed value, to assure the dynamic environment during a single simulation run. Our simulation parameters are described in Table II. Note that our parameter selection is only showcasing an example scenario within the study's context and we leave examination of different settings, such as elaborate mobility algorithms, to future work.

TABLE II
SIMULATION PARAMETERS USED FOR INDOOR FACTORY.

Parameter	**Value**
Area Length	120 m
Area Width	60 m
Simulation time	1000 TTI
Environment	Indoor Factory
Number of Base Stations	18
Operating Band	FR1
Bandwidth	20 MHz
Subcarrier Spacing	15 kHz
Mobility of Users	Random Waypoint
Number of Edge Servers	5
Processor Speed of Edge Servers	412 000 MIPS
Service Instances per Server	10
Service-User Association	Round-Robin
Packet Size	30 kB
Packet Inter-arrival Rate	10 ms
Number of Instructions per Packet	*unif(0.75,1.25) MI*
Delay Tolerance of Application	100 ms

A. *Flexibility*

`nextGSIM` simulator enables a flexible configuration of realistic scenarios, ranging from small-scale indoor factory scenarios (Fig. 2) to large-scale outdoor scenarios (see Fig. 4). To showcase the versatility of `nextGSIM`, besides the indoor factory scenario, we simulate an outdoor scenario as described in [29] and depicted in Fig. 4. This scenario involves the deployment of 36 macro gNBs serving in total 100 users. To enhance coverage, each macro gNB is supported by 9 micro gNBs. The placement of the macro and micro gNBs is automatically defined by `nextGSIM`, taking into account the cell radius and the number of serving gNBs to provide a full coverage of the area. The users' connectivity to the serving gNBs is determined based on their distance, which directly affects the quality of the channel and their overall performance. The users follow a random waypoint mobility [39], with varying speeds specified during the simulation initialization. It is important to note that the users maintain a constant speed throughout the simulation. Moreover, `nextGSIM` allows for scalable reconfiguration of network entities, including number of UEs, gNBs and MEC servers, along with operation frequency, bandwidth, radio resources, etc. Additionally, the simulator supports reconfiguration of mobility models, traffic models, control functionalities of the SD-RAN controller and edge orchestrator [48]. For instance, researchers can switch between various radio resource scheduling algorithms, computing resource allocation algorithms, RRC state control methods for UEs.

B. *Resource Allocation*

We exhibit the utility of `nextGSIM` by comparing several radio and compute resource allocation strategies for proof-of-concept indoor factory use case. We simulate a typical beyond-5G usecase where each client runs a demanding application generating significant data load in the network and requires low-latency computation from nearby edge servers (see Table II for details). We test several radio resource scheduling algorithms, including Round-Robin (RR), Max-Rate (MR),

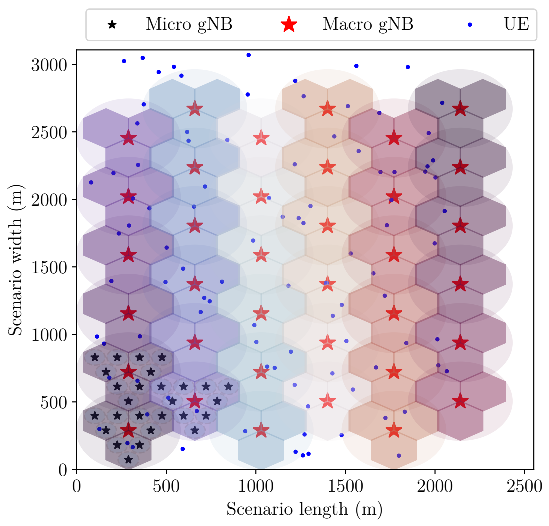

Fig. 4. Outdoor scenario where 36 macro gNBs serve a total of 100 UEs. Each macro gNB contains 9 micro gNB to provide a better service to UEs.

and throughput Proportional Fair (PF), in combination with different edge priority queuing algorithms: First-Come First-Serve (FCFS) and Radio-Aware Selective Queueing (RASQ), which were briefly described in §III-A and §III-B.

While the radio resource scheduling algorithms allocates the resources to improve the users experience, FCFS algorithms remains unaware of the level of QoS satisfaction achieved on the radio side. Fig. 5 depicts this behavior as it shows the rate of tasks that were completed before their latency threshold. For less users, RR slightly outperforms the other scheduling algorithms since the available resources are allocated fairly among UEs, and the utilization of computing resources is not high. However, as the number of users increases, there is higher competition for radio resources, resulting in increased latency. This is especially noticeable for RR-FCFS allocation where the fair distribution of resources, leads to higher transmission latency for tasks. PF and MR perform significantly better since they allocate resources to optimize throughput and the performance of users with the best channel quality, respectively. Note the better performance of RASQ for all radio scheduler variants. In RASQ, the edge orchestrator *jointly* considers both the average radio latency of users assigned to the application instances along with server capacity. As such, the algorithm prioritizes users with poor channel conditions and maximizing the rate of processed packets before hitting the maximum tolerable delay threshold. Notice the steep decrease in round-robin-FCFS plot at >100 users. This is because with FCFS, edge servers compute tasks that have already exceeded the latency threshold due to radio, due to which they miss out on legitimate task requests. Exceeding 250 users, the PF scheduler begins outperforming RR since it selectively and fairly allocates radio resources as they become scarce. The RASQ algorithm does not significantly impact MaxRate as only users with the best radio conditions are scheduled, leaving little room for improvement.

In Fig. 6, the experiment is repeated with 50 and 150 UEs with Round-Robin radio resource scheduling combined with

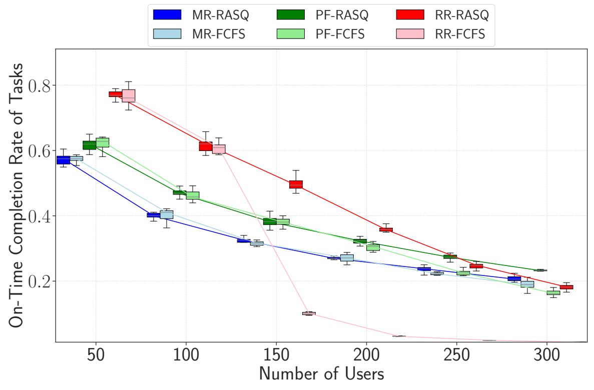

Fig. 5. On-time completion rate of tasks vs number of users with different radio resource allocation and queueing algorithm combinations.

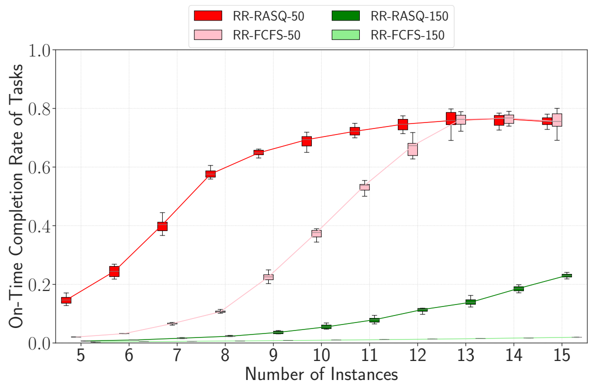

Fig. 6. On-time completion rate of tasks vs number of applications instances deployed per server.

FCFS and RASQ compute resource scheduling algorithms running and variable number of application instances deployed per server. Number of servers are reduced to 2, to underline the effects of computing resource bottlenecks. It can be seen that by using a radio aware compute resource scheduling algorithm instead of a FCFS scheduling algorithm, same performance can be achieved by deploying less instances per server.

V. Conclusion and Future Work

In this paper we present `nextGSIM`, a comprehensive network simulation tool that allows researchers to experiment with scenarios incorporating both radio networking and edge computing. `nextGSIM` aligns with the 3GPP standardization and incorporates features envisioned for 5G and beyond networks, including the FR2 operating mode, intelligent SD-RAN controllers and mobile edge computing entities. Our objective for designing `nextGSIM` was to reduce the gap between the rapid evolution of the communication networks and evaluating frameworks, especially focusing on the resource management of scarce radio, computing, and memory resources. We showcase `nextGSIM`'s efficacy through our extensive simulations of indoor factory scenario with dense connected devices. We highlight the performance of radio and compute resource schedulers in `nextGSIM` and show that supporting latency-critical applications in beyond-5G networks require a joint optimization of both resources simultaneously, along with effective knowledge transfer between the two layers – which can be experimented using simulators such as `nextGSIM`.

In future, we plan to extend `nextGSIM` with predictive algorithms, explore complex microservice interactions, enhance environment and application context awareness, develop joint resource allocation algorithms and study the energy efficiency of the network entities. We plan to make `nextGSIM` available to the public, to support researchers who want to evaluate their algorithms and contribute to the simulator.

VI. Acknowledgments

This work was supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy project ”6G Future Lab Bavaria” and Federal Ministry of Education and Research of Germany project 6G-Life (16KISK002).

References

[1] A. Narayanan *et al.*, “A first look at commercial 5G performance on smartphones,” in *Proceedings of The Web Conference*, WWW ’20, 2020.

[2] A. Narayanan *et al.*, “A variegated look at 5G in the wild: Performance, power, and QoE implications,” SIGCOMM ’21, 2021.

[3] Z. Zhang *et al.*, “6G wireless networks: Vision, requirements, architecture, and key technologies,” *IEEE Vehicular Technology Mag.*, 2019.

[4] X. You *et al.*, “Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts,” *Science China Information Sciences*, 2021.

[5] N. Mohan, L. Corneo, A. Zavodovski, S. Bayhan, W. Wong, and J. Kangasharju, “Pruning edge research with latency shears,” in *19th ACM Workshop on Hot Topics in Networks (HotNets)*, (New York, NY, USA), p. 182–189, Association for Computing Machinery, 2020.

[6] F. Alriksson *et al.*, “XR and 5G: Extended reality at scale with time-critical communication,” *Ericsson Technology Review*, 2021.

[7] V. Cozzolino, L. Tonetto, N. Mohan, A. Y. Ding, and J. Ott, “Nimbus: Towards latency-energy efficient task offloading for ar services,” *IEEE Transactions on Cloud Computing*, 2023.

[8] H. Ullah *et al.*, “5G communication: An overview of vehicle-to-everything, drones, and healthcare use-cases,” *IEEE Access*, 2019.

[9] A. Slalmi, H. Chaibi, A. Chehri, R. Saadane, and G. Jeon, “Toward 6G: Understanding network requirements and key performance indicators,” *Transactions on Emerging Telecommunications Technologies*, 2021.

[10] Q.-V. Pham *et al.*, “A survey of multi-access edge computing in 5G and beyond: Fundamentals, technology integration, and state-of-the-art,” *IEEE access*, 2020.

[11] X. Lin, “An overview of 5G advanced evolution in 3GPP Release 18,” *IEEE Communications Standards Magazine*, 2022.

[12] H. Liao *et al.*, “Learning-based context-aware resource allocation for edge-computing-empowered industrial IoT,” *IEEE Internet of Things Journal*, 2019.

[13] E. Peltonen, I. Ahmad, A. Aral, M. Capobianco, A. Y. Ding, F. Gil-Castiñeira, E. Gilman, E. Harjula, M. Jurmu, T. Karvonen, M. Kelanti, T. Leppänen, L. Lovén, T. Mikkonen, N. Mohan, P. Nurmi, S. Pirttikangas, P. Sroka, S. Tarkoma, and T. Yang, “The many faces of edge intelligence,” *IEEE Access*, 2022.

[14] A. Y. Ding *et al.*, “Roadmap for Edge AI: A Dagstuhl Perspective,” 2022.

[15] M. Polese, L. Bonati, S. D’Oro, S. Basagni, and T. Melodia, “Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges,” *IEEE Communications Surveys Tutorials*, 2023.

[16] T. Taleb *et al.*, “On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration,” *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[17] N. Mohan and J. Kangasharju, “Edge-fog cloud: A distributed cloud for internet of things computations,” in *Cloudification of the Internet of Things (CIoT)*, 2016.

[18] T. R. Henderson *et al.*, “Network simulations with the ns-3 simulator,” *SIGCOMM demonstration*, 2008.

[19] A. Varga, “Omnet++,” *Modeling and tools for network simulation*, 2010.

[20] J. Prokkola, “Opnet-network simulator,” *URL http://www. telecomlab. oulu. fi/kurssit/521365A tietoliikennetekniikan simuloinnit ja tyokalut/Opnet esittely*, 2006.

[21] X. Zeng *et al.*, “IOTSim: A simulator for analysing IoT applications,” *Journal of Systems Architecture*, vol. 72, 2017.

[22] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, “iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments,” *Software: Practice and Experience*, 2017.

[23] C. Sonmez, A. Ozgovde, and C. Ersoy, “Edgecloudsim: An environment for performance evaluation of edge computing systems,” *Transactions on Emerging Telecommunications Technologies*, 2018.

[24] I. Lera, C. Guerrero, and C. Juiz, “YAFS: A simulator for IoT scenarios in fog computing,” *IEEE Access*, 2019.

[25] M. Salama, Y. Elkhatib, and G. Blair, “IoTNetSim: A modelling and simulation platform for end-to-end IoT services and networking,” in *Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing*, 2019.

[26] D. N. Jha *et al.*, “IoTSim-Edge: a simulation framework for modeling the behavior of Internet of Things and edge computing environments,” *Software: Practice and Experience*, 2020.

[27] T. Qayyum, A. W. Malik, M. A. K. Khattak, O. Khalid, and S. U. Khan, “FogNetSim++: A toolkit for modeling and simulation of distributed fog environment,” *IEEE Access*, 2018.

[28] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, “Simu5G–an OMNeT++ library for end-to-end performance evaluation of 5G networks,” *IEEE Access*, 2020.

[29] 3GPP, “ 5G; Study on scenarios and requirements for next generation access technologies ,” Technical Specification (TS) 38.913, 3rd Generation Partnership Project (3GPP), 07 2018. Version 15.0.0.

[30] 3GPP, “ 5G; NR; Radio Resource Control (RRC); Protocol specification,” Technical Specification (TS) 38.913, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.1.0.

[31] 3GPP, “ 5G; Study on channel model for frequencies from 0.5 to 100 GHz ,” Technical Specification (TS) 38.901, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.1.0.

[32] X. Foukas *et al.*, “FlexRAN: A flexible and programmable platform for software-defined radio access networks,” in *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*, 2016.

[33] B. Balasubramanian *et al.*, “RIC: A RAN intelligent controller platform for AI-enabled cellular networks,” *IEEE Internet Computing*, 2021.

[34] W.-H. Ko *et al.*, “EdgeRIC: Empowering realtime intelligent optimization and control in nextG networks,” 2023.

[35] R. N. Calheiros *et al.*, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and Experience*, 2011.

[36] R. Schmidt, M. Irazabal, and N. Nikaein, “FlexRIC: an SDK for next-generation SD-RANs,” in *Proceedings of 17th International Conference on emerging Networking EXperiments and Technologies*, 2021.

[37] N. Matloff, “Introduction to discrete-event simulation and the simpy language,” *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August*, 2008.

[38] A. A. Hagberg, D. A. Schult, and P. J. Swart, “Exploring network structure, dynamics, and function using NetworkX,” in *Proceedings of the 7th Python in Science Conference*, 2008.

[39] E. Hyytiä and J. Virtamo, “Random waypoint mobility model in cellular networks,” *Wireless Networks*, 2007.

[40] J. Navarro-Ortiz *et al.*, “A Survey on 5G Usage Scenarios and Traffic Models,” *IEEE Communications Surveys and Tutorials*, 2020.

[41] M. Laner, P. Svoboda, N. Nikaein, and M. Rupp, “Traffic models for machine type communications,” in *ISWCS 2013; The Tenth International Symposium on Wireless Communication Systems*, VDE, 2013.

[42] 3GPP, “ 5G; NR; User Equipment (UE) radio transmission and reception; Part 1: Range 1 Standalone,” Technical Specification (TS) 38.101-1, 3rd Generation Partnership Project (3GPP), 07 2020. Version 16.4.0.

[43] S. Bäurle and N. Mohan, “ComB: A flexible, application-oriented benchmark for edge computing,” EdgeSys ’22, 2022.

[44] L. Bao, C. Wu, X. Bu, N. Ren, and M. Shen, “Performance modeling and workflow scheduling of microservice-based applications in clouds,” *IEEE Transactions on Parallel and Distributed Systems*, 2019.

[45] CNCF, “Kubeedge.” https://github.com/kubeedge/kubeedge, 2022.

[46] T. L. Foundation, “Kubernetes,” 2022.

[47] M. Schmieder, T. Eichler, S. Wittig, M. Peter, and W. Keusgen, “Measurement and characterization of an indoor industrial environment at 3.7 and 28 ghz,” in *2020 14th EuCAP*, IEEE, 2020.

[48] G. Bartolomeo, M. Yosofie, S. Bäurle, O. Haluszczynski, N. Mohan, and J. Ott, “Oakestra: A lightweight hierarchical orchestration framework for edge computing,” in *USENIX Annual Technical Conference (ATC)*, 2023.