VFogSim: A Data-Driven Platform for Simulating Vehicular Fog Computing Environment

Özgür Umut Akgül, Wencan Mao, Byungjin Cho, and Yu Xiao, *Member, IEEE*

***Abstract*—Edge/fog computing is a key enabling technology in 5G and beyond for fulfilling the tight latency requirements of emerging vehicle applications, such as cooperative and autonomous driving. Vehicular fog computing (VFC) is a cost-efficient deployment option that complements stationary fog nodes with mobile ones carried by moving vehicles. To plan the deployment and manage the VFC resources in the real world, it is essential to consider the spatiotemporal variations in both demand and supply of fog computing capacity and the tradeoffs between achievable quality-of-services and potential deployment and operating costs. The existing edge/fog computing simulators, such as IFogSim, IoTSim, and EdgeCloudSim, cannot provide a realistic technoeconomic investigation to analyze the implications of VFC deployment options due to the simplified network models in use, the lack of support for fog node mobility, and limited testing scenarios. In this article, we propose an open-source simulator VFogSim that allows real-world data as input for simulating the supply and demand of VFC in urban areas. It follows a modular design to evaluate the performance and cost efficiency of deployment scenarios under various vehicular traffic models, and the effectiveness of the diverse network and computation schedulers and prioritization mechanisms under user-defined scenarios. To the best of our knowledge, our platform is the first one that supports the mobility of fog nodes and provides realistic modeling of vehicle-to-everything in 5G and beyond networks in the urban environment. Furthermore, we validate the accuracy of the platform using a real-world 5G measurement and demonstrate the functionality of the platform taking VFC capacity planning as an example.**

***Index Terms*—Capacity planning, cellular networks, mobile computing, systems simulation, vehicular and wireless technologies.**

I. Introduction

EDGE/FOG computing brings cloud-like computing services closer to where the data are generated in order to reduce the network latency. In the case of fog computing for vehicular applications, or vehicular fog computing (VFC), fog computing nodes can be deployed in stationary infrastructures or on moving vehicles. According to Yousefpour et al. [1], VFC enables more flexible and cost-efficient deployment of computing resources by complementing stationary fog nodes colocated with cellular base stations [i.e., cellular fog nodes (CFNs)] with vehicular fog nodes (VFNs) carried by moving vehicles. Specifically, it utilizes the mobility of VFNs to respond to excess demand at peak hours of the day and during special events [2].

Despite being a promising solution, developing and evaluating the capacity and resource management strategies for VFC remains a challenging topic. Minimizing costs while increasing the quality of service (QoS) requires tackling various research questions, including where and how much capacity to deploy [3], [4], whether and where to offload tasks [5], [6], or how to schedule the radio and computing resources jointly [7]. One common challenge of these research questions is how to assess the system performance for various conditions. Real-world experiments provide the accurate evaluation by demonstrating the advantages and risks of the proposed systems. However, real-world test environments are usually quite expensive and difficult to set up. Moreover, in most cases, it is not possible to obtain isolated and dedicated test environments to understand the value of VFC in different scenarios. The need to have a deeper understanding of the deployment decisions under extreme circumstances (e.g., ultradense, fast moving, and unbalanced loads), as well as the low economic and time requirements, has increased the attention on the simulators over the research community.

There are already several simulators for dynamic edge computing, e.g., iFogSim [8], EdgeCloudSim [10], and FogNetSim++ [11]. While these simulators can capture the general attributes of networking (e.g., QoS measurement or the mobility of client vehicles), they are missing several fundamental features, such as mobility of computing nodes [8], [10] or service differentiation [11]. This article aims at solving this challenge by developing an open platform that provides a realistic simulation of dynamic VFC environments and supports the simulation of various application scenarios.

The platform proposed in this article uses a data-driven approach where real-world or synthetic data can be used. The input data consist of the signal-to-interference-plus-noise ratio (SINR) and vehicular traffic locations. Unlike the other simulators, VFogSim supports both stationary and mobile fog nodes and integrates vehicular traffic and network simulation covering physical and upper-layer protocols. It is embedded with both spectral and computational resource allocation policies and can be customized by user-defined algorithms. Finally, our simulator can be used for evaluating various aspects, including the QoS metrics (e.g., data rate and average delay), the technoeconomic performance of the different network, and task allocation

Manuscript received 19 September 2022; revised 9 April 2023; accepted 7 June 2023. Date of publication 29 June 2023; date of current version 30 August 2023. This work was supported by the Academy of Finland under Grant 317432 and Grant 318937. *(Corresponding author: Byungjin Cho.)*

The authors are with the Department of Information and Communications Engineering, Aalto University, 02150 Espoo, Finland (e-mail: ozgur.akgul@aalto.fi; wencan.mao@aalto.fi; byungjin.cho@aalto.fi; yu.xiao@aalto.fi).

Digital Object Identifier 10.1109/JSYST.2023.3286329

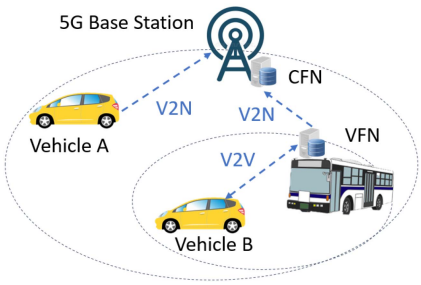

Fig. 1. Application scenario of VFC.

policies, and the impacts of various factors, such as interservice prioritization, traffic load, and pricing strategy.

To the best of our knowledge, it is the first data-driven VFC simulation platform. Our key contributions are given as follows.

1) We develop a data-driven system-level simulation platform, VFogSim, for evaluating different VFC deployment options and resource schedulers. The developed platform is made open source[1] to invite the research community for future developments.
2) We build VFogSim following the modular design principles, which allow it to be easily customized for particular use cases or input data. We assess this modularity by testing our simulator with both real and synthetic data as well as considering different deployment options.
3) We demonstrate the functionality of VFogSim through a case study on VFC deployment in an urban area, the results provide insights into the technoeconomic implications of different capacity planning options.

The rest of this article is organized as follows. Section II reviews the related works. Section III presents the platform architecture and an example scenario. Section IV details the implementation of the platform; Section V evaluates the accuracy and the functionality of our simulator. Section VI discusses the key features, limitations of the platform, and future work. Finally, Section VII concludes this article.

II. Background and Related Works

In this section, we first introduce an exemplary scenario of VFC, then we compare our simulation platform with the state-of-the-art works.

A. Vehicular Fog Computing

Fig. 1 presents an application scenario of VFC, where Vehicles A and B can either offload their tasks to a CFN colocated with a 5G base station or a VFN carried by a bus within a one-hop vehicle-to-everything (V2X) communication range. In this scenario, the CFN serves as a local coordinator for spectral and computational resource allocation. VFC can be used for emerging vehicular applications involving time-critical and data-intensive computational tasks. Table II lists some exemplary applications and their corresponding latency requirements.

[1]Current version of the simulation platform is available at https://mobilecloud.aalto.fi/?page_id=1441

B. Related Works

Current edge/fog computing simulators are mainly built on the existing cloud computing simulators or network simulators. Table I presents a comparison of VFogSim with the state-of-the-art platforms. Three well-known edge computing simulators are IFogSim [8], IoTSim [9], and EdgeCloudSim [10], all of which are built on CloudSim [19]. IFogSim models the fog environment where the fog nodes follow a hierarchical arrangement from the sensors to the cloud and measure the impacts of resource management policies in terms of latency, network congestion, energy consumption, and cost [8]. IoTSim supports the simulation of Internet of Things (IoTs) big data processing using the MapReduce model [9]. However, they simplify the network model and do not consider the communication channel attributes (e.g., SINR). EdgeCloudSim integrates multiple modules into an edge computing system, including the core simulation (i.e., the module responsible for loading and running the edge computing scenarios from the configuration files), networking, load generator, mobility, and edge orchestrator modules [10]. Despite considering the mobility of client vehicles, it does not support the mobility of fog nodes.

FogNetSim++ [11] is an edge computing simulator based on OMNet++ [20], focusing on simulating the network characteristics of distributed edge computing devices and enabling users' customization of mobility models and fog node scheduling algorithms [11]. However, it cannot be used to estimate the network metrics (e.g., throughput) due to a lack of physical-layer protocols. Similarly, EmuFog [12] and Fogbed [13] are two edge computing simulators based on the network simulator Mininet [21] and its extended version MaxiNet [22]. EmuFog enables users to design the network topology with embedded fog nodes and run docker-based applications on those nodes connected by an emulated network [12]. Fogbed enables the dynamic adding, connecting, and removing of virtual nodes via docker containers and supports to perform real-world protocols and services [13]. Although they support the evaluation of cost and latency, they do not support other advanced functions (e.g., customizable scheduler, pricing strategy, and interservice prioritization).

FogTorch is a simulation tool that supports QoS-aware deployment of IoT applications to fog infrastructures [14]. Extending FogTorch, FogTorchII exploits Monte Carlo simulations to take into account the variations of the QoS and classifies deployments in terms of both QoS assurance and fog resource consumption [15]. Nevertheless, neither of the works has integrated the mobility model of client vehicles or taken the mobility of the fog nodes into account. To simulate the vehicular network, Veins [16] couple OMNeT++ [20] with the mobility simulator SUMO [23] with the implementation of IEEE 802.11p. It can also be used together with SimuLTE [24] and Simu5G [25], which offer a detailed model of the long-term evolution based and 5G new radio (NR)-based V2X, respectively. However, it does not contain a computational scheduler, advanced network scheduler, or economic model.

In this work, we propose a data-driven platform for VFC that contain the above-mentioned modules and functions. In the

TABLE I
COMPARISON OF OUR WORK WITH THE EXISTING EDGE/FOG SIMULATORS

Simulator	CP	RA	QoS metrics	CM	FM
IFogSim [8]	No	Yes	Latency, network congestion	No	No
IoTSim [9]	Yes	No	Latency	No	No
EdgeCloudSim [10]	Yes	No	Latency, acceptance ratio	Yes	No
FogNetSim++ [11]	Yes	Yes	Latency, packet loss ratio, handover	Yes	No
EmuFog [12]	Yes	No	Latency	No	No
Fogbed [13]	Yes	No	Latency	No	No
FogTorch [14]	Yes	No	Latency, bandwidth	No	No
FogTorchII [15]	Yes	No	Latency, bandwidth	No	No
Veins [16]	No	Yes	Latency, packet loss ratio, throughput	Yes	Yes
VFogSim	**Yes**	**Yes**	**Latency, network congestion, acceptance ratio, economic aspects**	**Yes**	**Yes**
CP: Evaluation of capacity planning strategies. RA: Evaluation of resource allocation strategies.			CM: Integration of mobility model of client vehicles. FM: Support of mobility of fog nodes.		

The bold entities highlight the proposed work in this paper.

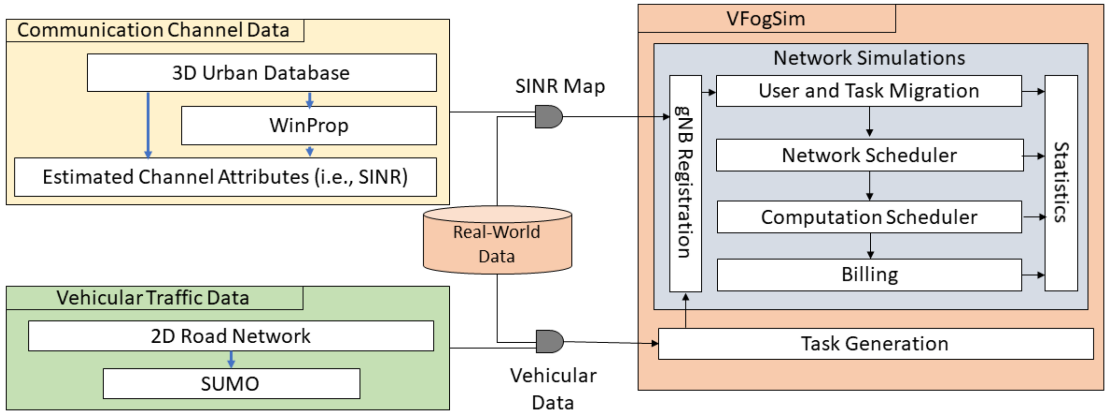

Fig. 2. System architecture of the VFogSim platform.

current version, the inputs can be the synthetic data generated by SUMO and WinProp [26] or real-world measurements. In the formal case, while SUMO enables a fine-grained simulation of vehicular traffic, WinProp supports a variety of air interfaces for 5G and beyond networks. Moreover, WinProp takes various real-world information, such as the locations of the base stations, buildings, and trees into consideration, thus offering a more realistic and widely applicable vehicular network simulation. In the future version, we are considering integrating our platform with Simu5G [25] and Veins [16].

III. System Architecture and Example Scenario

This section gives an overview of the modular and customizable platform, VFogSim. We introduce the system architecture followed by an example use case.

A. System Architecture

As illustrated in Fig. 2, the VFogSim platform requires the GPS coordinates of the vehicles and an SINR map for the given region. Therefore, the first step of running a simulation is to provide these input data. We design the VFogSim platform as a discrete-time optimizer, so the input data need to be discretized into transmission time intervals (TTIs). Once these data are inputted, the vehicles are registered to different base stations, i.e., gNBs. Note that in synthetic data, it is possible to associate the users with specific base stations automatically. At each base station, we first update the active user information to determine the task migrations, i.e., when the vehicle moves to another base station before completing the active task. Once the active tasks are determined, each base station performs the network and computation scheduling and determines resource allocations to vehicles. The resulting system behaviors, such as resource allocation, delay, and billing information, are stored in statics, respectively, to be analyzed at the end of the simulation.

Input Data of VFogSim: VFogSim can take as input either real-world measurements or synthetic data of SINR maps and vehicular locations. To generate the synthetic data, in this article, we use WinProp [26] and SUMO [23]. We consider the 3-D map of the considered area from OpenStreetMap, real-world cellular bases stations, and propagation model and air interface configuration in WinProp [26] to estimate the SINR map of the region. For tractability, we omit the secondary reflections (e.g., other vehicles) on the SINR map. Meanwhile, based on the 2-D road network from OpenStreetMap, SUMO generates a fine-grained microscopic traffic simulation that includes vehicular movement and the traffic infrastructure (e.g., traffic lights). Alternatively, if we use the vehicular GPS data and the SINR data collected from the real-world experiments, both information needs to be associated with the TTIs. Particularly, we need the SINR information of the vehicle at every TTI. Apart from this association and time discretization, both synthetic and real-world

TABLE II
EXEMPLARY VEHICULAR APPLICATIONS AND THEIR DELAY REQUIREMENTS [17], [18]

Application	Delay requirements	Type
Collision avoidance	Bound, 10 ms	Safety
Vehicle platooning	Bound, 25 ms	Safety
Collective perception	Bound, 100 ms	Safety
Information sharing	Average, 250ms–500 ms	Safety/Nonsafety
Vehicle scheduling	Average, 1s	Nonsafety
AR/VR	Bound, 10 ms	Entertainment
Cloud gaming	Average, 100ms-1s	Entertainment

data can directly be used. Table III details the input data used in VFogSim.

Task Generation: The task generation block associates the vehicular traces with specific task requests and creates the network load. We discuss the specific association strategy in Section IV-B. It is possible to customize the tasks in terms of their CPU/GPU/NPU computing units, battery consumption, memory requirements, or interarrival times. The generated tasks, the vehicle ID, and the location of the vehicle are passed to the gNB registration block.

Network Simulations: The network simulations start with associating the vehicular traces with the SINR values at the gNB registration block. This block is also responsible for determining the load at different gNBs. If a client vehicle changes gNBs, the gNB registration block triggers the user and task migration block where the task migration tasks are handled and stored. Otherwise, the tasks are moved to the schedulers.

Network and Computation Schedulers: The network scheduler and computation scheduler focus on the allocation of spectral and computational resources, respectively. As we perform the resource allocations, we assume that the priority of a service is determined by the requested resources, the price of the service, and the remaining service execution time. We have separated the network and computation schedulers to ensure that different scheduling strategies can be covered using VFogSim. The information regarding the user and task migration, the resource (both spectral and computing) allocation decisions, and the billing information are stored in the statistics block for evaluation purposes, e.g., efficiency in network orchestration and resource management [27].

B. Example Scenario

The modular and customizable structure of VFogSim enables simulating a large variety of testing scenarios. The blocks, as presented in Fig. 2, can be either used as the default mode (i.e., presented in the article) or can be customized to test specific scenarios. Two testing scenarios for VFC are capacity planning and resource allocation. Capacity planning aims to determine where to deploy the fog nodes and how much should be deployed with cost efficiency and the QoS guarantee. For example, Mao et al. [4] proposed a capacity planning framework for VFC in order to minimize the costs while meeting the latency requirements. Resource allocation aims to find the matching strategies between the fog nodes and the client vehicles in order to maximize the QoS. For example, Zhu et al. [5] proposed a resource allocation algorithm for video crowd-sourcing tasks in a VFC environment in order to jointly minimize video quality and latency. To test different capacity planning strategies, the computation scheduler could be changed. On the other hand, to test the technoeconomic performances of different resource allocation strategies, both the network scheduler and the computation scheduler could be replaced.

To demonstrate an example scenario in this article, we consider the capacity planning for VFC in an urban area, as detailed in Section V. For the sake of simplicity, each user is assumed to be associated with one active service at each TTI. The SINR of user k is estimated based on the vehicle location and the SINR map. At every TTI, the simulator will first generate the active tasks and then perform the network scheduling per cell. Among the active services, the task scheduling algorithm runs for the computational resources. The user is queued if it fails to receive the computational resources. To ensure tractability, this work assumes that the respective containerized vehicular applications are active at all the fog nodes (including CFNs and VFNs). Consequently, the migration delay is the time it takes to move the user data from one fog node to another. This work assumes that the client vehicles are always connected to the cells with the highest SINR values.

IV. SIMULATOR DESIGN

In this section, we detail the data-driven simulator design and the key attributes of the default mode.

A. Generating Vehicular Traffic

We generate the synthetic vehicular trajectories in a region in two steps. In the first step, we feed the 2-D map of the region into SUMO. The road network generated by SUMO from the 2-D map contains the information on the nodes (e.g., road intersections), the edges (e.g., road segments), and the relationship between the edges (e.g., junctions). While generating the road network, we include the road information (e.g., the number of lanes), road type (e.g., motorway), and traffic regulations (e.g., one way or both). Additionally, the road network contains information on the traffic lights at each intersection and generates the corresponding time schedules.

In the second step, the trips of the client vehicles are generated using the built-in function randomTrips. The number of vehicles simulating at each time is controlled by the arrival and departure rates. More specifically, if there are n users in the setting, then the arrival rate is n vehicles/s. We control the arrival period to be very small (0.01 s) so that all the n vehicles appear at the same time. The randomTrips function will randomly choose the road segments as the origin and destination of each vehicle's trip and route the vehicles according to the shortest path. The vehicle will disappear after it has arrived at its destination. The generation of the trips also considers the number of lanes as weight so that the traffic density on the primary roads is generally higher than on the secondary roads. Moreover, we use different arrival and departure rates to simulate various traffic conditions, such as peak hours, off-peak hours, and extreme scenarios (e.g., when the request arrival rate is extremely high). In this work, VFNs

TABLE III
DESCRIPTION OF INPUT DATA USED IN VFOGSIM

Input data	To module	Data type (x and y are in UTM)	Potential sources
2-D map	SUMO	Road network with traffic infrastructure	OpenStreetMap, HERE map
3-D map	WinProp	Urban database with buildings, trees, etc.	OpenStreetMap, CADmapper
Cellular map	WinProp	List (base station ID, x, y)	CellMapper, OpenCellID
SINR map	VFogSim	Matrix (x, y, SINR)	WinProp or real-world measurements
Input.cvs	VFogSim	Time series (time step, user ID, x, y, SINR)	SUMO and WinProp or real-world measurement
bus.cvs	VFogSim	Time series (time step, VFN ID, x, y)	SUMO or onboard GPS data

are assumed to have regular routes and time schedules (e.g., carried by buses), where their routes are generated using the origin–destination matrix. More specifically, we set the origins and destinations of all the buses and generate their routes by duarouter, which calculates the shortest paths.

B. Modeling Communication Channel

We built an SINR map of the region in question using Altair WinProp [26], taking the 3-D map of the area as input. Note that the SINR map can also be extracted from real-world measurement or simulated by other open-source platforms (e.g., Veins [16], cf., Section VI). The 3-D map (i.e., from OpenStreetMap) is used for estimating the transmission loss caused by reflections and shadowing. For instance, we consider the location and height of buildings and trees when calculating transmission loss using the propagation model (e.g., the dominant path model) for urban areas with dense buildings and trees. We place the base stations according to their real-world locations. For example, the locations of the base stations are extracted from the street addresses in CellMapper [28] and exported to universal transverse mercator (UTM) coordinate system using geocoding. The air interface is configured according to the 5G configuration in [26]. We use 5G NR vehicle-to-network (V2N) for communication between the vehicles and with the infrastructure, i.e., the VFNs interact with the client vehicles with the aid of the cellular base stations. The parameters to model the communication channel are listed in Table V. According to the location of the VFNs, the capacity of VFNs is assigned to each cell. Based on the 3-D city database, base station information, and communication channel parameters, we calculate the SINR map in WinProp. The SINR map is a matrix of SINR values in the area with a spatial granularity of $1\,\mathrm{m}^2$. When the vehicle's position changes, the SINR value will change accordingly. Combing the SINR map with the vehicular traces, we calculate the achievable rate of each vehicle at each time slot (i.e., TTI) using the Shannon formula, i.e., presented in (2). When the client vehicle moves out of the cell coverage, the task migration is carried out, and we calculate the SINR value in the new cell.

To represent the relationship between the transmission data rate expectations R and utility values U, we consider a piecewise linear utility function proposed in [7] (cf., Fig. 3). The utility values reflect the QoS received by the client vehicles. This utility function is determined by six parameters, namely R_1, R_2, R_3, U_1, U_2, and U_3. The region between R_1 and R_2 is considered to be a standard quality region to which the QoS is strictly tied. After R_2, the increase in the data rate has a slower effect on the achieved utility. We assume that R_3 is the saturation point, namely further increasing the achievable rate above this value would not impact the QoS. The service is expected to reach the average data rate of R_3 over a time interval of W_s. If a service fails to reach the average rate of R_3 at the end of W_s, it is considered to be dropped. Therefore, W_s sets an indirect delay requirement for the system. The proposed utility function in Fig. 3 covers a multitude of services and can be customized to reflect a specific characteristic. For example, for a critical service where the QoS is not scaled with the data rate and a continuous data rate of R is needed, it is possible to set $R_1 = R_2 = R_3 = R$. This way the service is not considered to be active unless it achieves R, and once it becomes active, it directly moves to the saturation region.

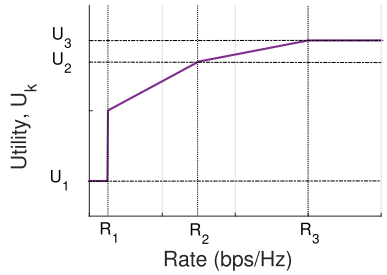

Fig. 3. Generic utility function from the article presented in [7].

In the network site, the service attributes are coupled with a utility value, which is proportional to the average data rate, R_k (bps/Hz). The average data rate is calculated by

$$R_k[n] = \sum_{i=n-W_s}^{n} x_k[i] r_k[i] / W_s. \tag{1}$$

In this equation, continuous variable $x_k[i] \in [0, 1]$ is a decision variable that represents the assigned wireless resource to a client vehicle k at time slot i. $r_k[i]$ (bps/Hz) is the achievable transmission data rate of client vehicle k at time slot i, which is calculated by

$$r_k[i] = B\log_2(1 + \mathrm{SINR}_k[i]). \tag{2}$$

B (Hz) represents the all available network bandwidths. The actual data rate of client vehicle k at time slot i is then given by $x_k[i]r_k[i]$.

C. VFogSim Simulation

The vehicular traffic traces and channel simulation data as well as the service attributes (e.g., application profiles) are inputted to VFogSim. In practice, the information that the cellular base station needs to collect from the client vehicles includes the

TABLE IV
TRANSMISSION DATA RATE EXPECTATIONS $R_1 - R_3$ (BPS/HZ), UTILITY VALUES $U_1 - U_3$, AVERAGE TASK SIZE C_{AVG} (KB/TASK), AND PRICE p (MU/TASK) FOR FOUR EXEMPLARY SERVICE TYPES WITH DELAY EXPECTATIONS W_s (TTI) EQUAL TO ONE

Service type	R_1	R_2	R_3	U_1	U_2	U_3	C_{avg}	p
A_1: Background	0.05	0.07	0.07	0	1	1	0	1
A_2: Object detection	0.01	0.08	0.4	-1	0.7	1	500	2
A_3: Lane detection	0.1	0.23	0.55	-0.5	0.7	1	1000	5
A_4: Video transcoding	0	1.08	∞	0	1	∞	1500	10

When the service differentiation is turned OFF, the *R* and *U* values for all the services are set according to A_1. When the pricing is turned OFF, the *p* values for all the services are set as 1 MU.

vehicle ID, vehicle dynamics per second (e.g., location, speed, and direction), service type, and latency requirement, which is collected every TTI.

Task Generation: The computing tasks can be classified based on their degree of demand for computation resources. Such resource demand is determined by average task size C_{avg} (kB/task) and computation intensities γ (cycles/kB). In our simulation platform, we consider each task as a basic unit for offloading with an average demand size C_{avg}. The computation intensity represents how many CPU cycles are required to process one-bit input data for a task mainly depending on the nature of the applications. This yields the mean number of CPU cycles per processed task, D (cycles/task), i.e., $D = C_{\text{avg}} \cdot \gamma$. We use four exemplary services in Table IV to model the service heterogeneity.

Network Simulations: On the network side, the service demand per cell is pooled. We assume that if the client vehicles cannot react to a certain threshold in their transmission data rate, they failed to transmit their tasks, and they need to retransmit their requests in the next TTI. The vehicles that reach this threshold are considered to be active. We design the spectral resource allocation and the resource management of fog nodes as two separate problems.

Network Scheduler: The objective of the network scheduler is to maximize the total transmission data rate of the overall client vehicles at each TTI. The different slopes in the piecewise linear utility function reflect the service priorities in the max-rate scheduler. More specifically, the scheduler would give the resources to the services, which can create the highest utility increase with a unit resource. Consequently, the applied utility function also determines the interservice priorities. In addition to this interservice priority, due to the definition of R_k in (1), the achievable rates of the client vehicles also affect the achieved utility and the network resource allocation. We focus on a simple scheduler at the spectrum side formulated as follows:

$$\max_{x_k[n]} \sum_{k \in K} U_k(R_k[n]) \tag{3a}$$

$$\text{s.t} \sum_{k \in K} x_k[n] \leq 1. \tag{3b}$$

The objective function in (3a) maximizes the total achieved utility over all the vehicles, K. We calculate the achieved utility based on the piecewise linear utility function and R_k in (1). The utility function in Fig. 3 is linearized using standardized methods. Note that as a real-time scheduler, the optimizer only considers $x_k[n]$ as a variable, while the rest of the allocated resources before n are considered as parameters. The constraint in (3b) limits the assigned spectral resources to the maximum available resources.

Computation Scheduler: We assume that the client vehicles are associated with the colocated fog node of the gNB to which they are connected. Therefore, in this work, we focus on fog node selection from a set of available ones rather than using an explicit task allocation algorithm and consider the following task scheduling problem as follows:

$$\max_{q_k} \sum_{k \in K} p_k D_k \log^{-1} t_{\text{remaining}} q_k \tag{4a}$$

$$\text{s.t}\, q_k \in \{0, 1\} \tag{4b}$$

$$\sum_{k \in K} q_k D_k \leq AR_c \;\; \forall c \in C. \tag{4c}$$

The objective function (4a) models the interservice prioritization, based on the price of a service p_k, the total demanded resources of service k, D_k, and the remaining time to finish the execution of the task $t_{\text{remaining}}$. The remaining time calculated based on the delay expectations W_s progressively decreases during the simulation.

The scheduler gives a higher priority to the services with lower remaining execution times, ensures the continuous execution of the services, and creates possibilities to interrupt the service execution if a higher priority service request is received (e.g., in case of emergency). As a design metric, the service requests with higher resource demand are prioritized over the smaller ones to minimize fragmentation. The binary variable q_k set in (4b) presents whether the client vehicle k is chosen for execution or not (i.e., 1 or 0). Finally, (4c) ensures that the total demand is not exceeding the available computing resource at the cth cell in a set of cells C (i.e., AR_c).

The task scheduling algorithm in (4a)–(4c) is run at every TTI. This way, the computation scheduler can dynamically update the resource usage based on the active services at each TTI. The current version of VFogSim supports only 5G V2N [29]. It can be extended to support vehicle-to-vehicle (V2V) in the future.

D. Accuracy and Scalability

In this section, we evaluate how close the wireless network simulation results are to the real-world measurements. The evaluation includes two parts, namely accuracy and scalability.

Experimental Setup: To test the accuracy of our network simulations, we set up a driving test with a Xiaomi Mi 10 T Pro 5G phone placed inside of a car. The phone is equipped with an Elisa commercial subscriber identification module card with

Fig. 4. Measurement setup in Otaniemi, a university campus in Helsinki. The figure shows the white car used for the measurement (on the top) and the test trail (at the bottom).

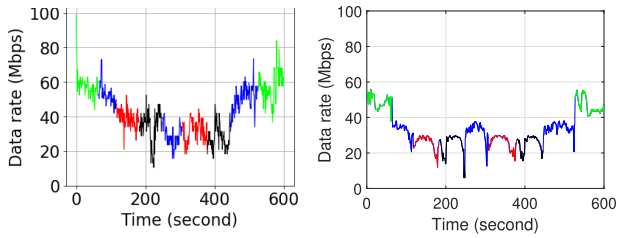

Fig. 5. Comparing the real-world measurements (on the left) and simulation results (on the right) in terms of data rate (i.e., throughput). The four cellular base stations with cell IDs #1 (red), #2 (black), #3 (green), and #4 (blue) have 8, 8, 4, and 6 client vehicles, respectively.

100 Mbps 5G mobile broadband for business. With regards to real-world performance measurement, we collect two metrics: cell ID representing which cell the vehicle is connected to (measured by Cellular-Z [30]); and data rate (i.e., throughput, measured by iperf3 [31]), and the data rate values are averaged over ten repeated measurements. The above metrics are associated with the GPS coordinates (measured by both Cellular-Z and iperf3, which are used for cross validation), collected every second over a sample duration of 10 min. During the sample collection, the car drives two cycles at an average speed of 18.57 km/h on the test trail, as shown in Fig. 4, with a speed limit of 30 km/h. On a commercial network, determining the active number of client vehicles and their demand is not trivial from the client side. Therefore, we used the measured data rate during our experiment to determine the number of clients. In particular, we tried various numbers of client vehicles in our simulations and selected the client number that minimizes the distance between the simulated result and the real-world measurement. For simplicity, we assumed that through the simulation horizon (i.e., 10 min in this case), the number of active client vehicles per cell and the active background services is constant, while the client vehicle's connected cell ID and the SINR value are changing in real time.

Simulator Accuracy: Fig. 5 shows the comparison between the measured throughput values in the real-world setup on the left and the simulator on the right. The actual throughput of a client vehicle depends not only on the SINR values but also on the scheduling dynamics, such as the number of active client vehicles, the service mix at the given cell, or the scheduling policy. This dynamic nature of resource scheduling causes the high fluctuations observed in real-world measurements. Fig. 5 shows that our network simulator performs close to the real-world measurements. In particular, both the simulation results and the real-world measurements have similar peak and minimum data rates. Although our assumption on the static number of client vehicles and their active applications in our simulation causes a more stable characteristic in Fig. 5, the result shows a similar trend as the real-world measurements. Therefore, the network simulator can provide a realistic estimation of the data rates. Note that the key objective of this work is to perform real-time resource allocation in VFN. Thus, the available computing capacity as well as the vehicular routes are parameters. The capacity planning or vehicular traffic routing problems are not considered in this work.

Simulator Scalability: From the execution time perspective, as the number of client vehicles increases from $|K| = 50$ to $|K| = 100$, we measure approximately 64% computation time increase, from 89.205 to 139.705 s, in a commercially available computer with an I7 processor and 16 GB random access memory. With 150 users in the peak scenario, the total network data overhead from the client vehicles to the cellular base stations are around 3500 kB.

V. Case Study

We evaluate the functionality of VFogSim by demonstrating how to use it for analyzing the feasibility of different VFC deployment plans from a technoeconomic perspective. In our analysis, we measure the data rate, delay, and cost of different deployment options. Moreover, we evaluate the impacts of interservice prioritization, traffic loads, and pricing on the capacity plan, which can be used either in the default mode presented in this section or substituted by the user-defined ones. *Experimental Setup:* In our simulations, we consider the 1 km^2 area in downtown Helsinki city, as shown in Fig. 6. Table V lists the simulation parameters. To generate the SINR values using WinProp, we use a digital map of the area and locations of cellular cells from OpenStreetMap. Using SUMO, we generate the vehicular traces through this region for various numbers of client vehicles (i.e., $|K|$). Using the GPS coordinates of the client vehicles and the generated SINR map, we determine the SINR values of client vehicles at each TTI. These data are inputted into the task generation block of our simulator. To test the functionality of our simulator, we consider two task generation cases, i.e., high and low task generation. During the high task generation, the client vehicles generate a new task at each TTI. Therefore, this case is considered to be a worst-case scenario on the network. On the other hand, during the low task generation case, the client vehicles generate a task during a fraction of the simulation duration (i.e., 80% during the simulation), which is

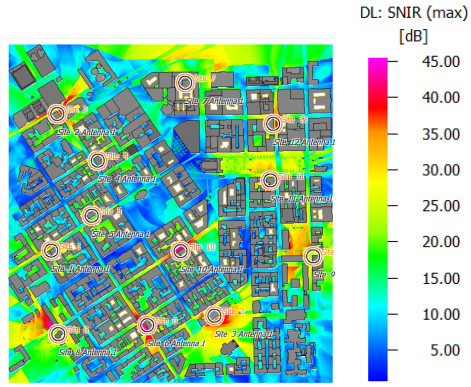

Fig. 6. SINR map generated in WinProp.

TABLE V
SIMULATION PARAMETERS IN THE CASE STUDY [32]

Parameter	Value
Air interface	5G NR n78
Multiple access	OFDM/OFDMA
Duplex separation	TDD
Frequency band	3500MHz
Channel bandwidth	100MHz
Minimum required SINR	−10dB
Number of base stations	12
Height of base stations	30m
Transmission power	46dB
Propagation model	Dominant path model
Simulation area	1km × 1km
Simulation granularity	1m × 1m
Simulation horizon	200s
Simulation time slot	1s

randomly distributed to the simulation horizon. In this article, we consider a relatively short-time horizon (i.e., in the order of hours); therefore, long-term aspects, such as the number of VFNs, communication technology, and network or edge capacity, are considered to be constant. However, it is possible to test different communication technologies, changing bus routes, or capacity expansion strategies.

To map the service diversity in 5G networks, we model the four services with different QoS expectations, interservice priorities, and prices. The different services are modeled by changing the R_1, R_2, R_3, U_1, U_2, U_3, C_{avg}, W_s and p parameters.

1) *Background (A_1):* This service runs in the background and requires a relatively low data rate (e.g., connectivity service). It does not have any computational demand.
2) *Object detection (A_2):* This service is latency sensitive and rate sensitive (i.e., nonelastic). We implement it through YOLOv5s [33] trained on COCO [34] dataset.
3) *Lane detection (A_3):* This service is latency sensitive and rate demanding (i.e., nonelastic). We implement it through OpenCV [35] in a Python environment.
4) *Video transcoding (A_4):* This service has relatively soft latency and rate constraints (i.e., elastic). We implement it through HandBrake video transcoder [36] with an x265 video encoder and mp4 container.

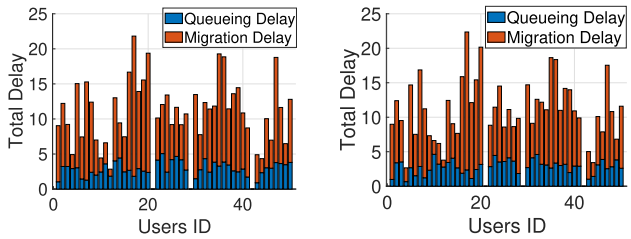

Fig. 7. Estimated total delay for deployment options NB (on the left) and WB (on the right) for $|K| = 50$ and high task generation rate.

We profile the CPU resource demands of each service using an Intel Core i7-7700 K eight-thread CPU with 4.2 GHz frequency and detailed in Table IV. For this scenario, we assume that the network operator only supports one service.

Finally, we compare the following four deployment options based on the simulation results, i.e., estimated network connection and acceptance ratio.

1) *NB Scenario:* In order to provide a base-level performance, we consider the scenario where the CFNs are used to serve all the demands.
2) *WB Scenario:* To present our hybrid deployment solution, we consider the scenario where the VFNs are used to complement the CFNs.
3) *InNB Scenario:* As an alternative to the WB scenario, we consider the case where the provider chooses to double the available CFNs instead of investing in VFNs.
4) *B Scenario:* To provide another base-level performance, we consider the case no CFNs are deployed and all the computational tasks rely on VFNs.

The different scenarios are modeled by changing the available computing capacity at the buses and the base stations. To simulate NB and INB scenarios, we set the computing capacity of the buses equal to zero, whereas, for the B scenario, we set the CFN capacity equal to zero. For the WB, we consider both the CFN and VFN; therefore, both computing capacities are nonzero. Due to the V2N scenario, from the network scheduler perspective, there are no differences among these scenarios. The VFogSim platform stores the resource allocations per user, the network data rates, the experienced delays per user (i.e., migration delay and queuing delay), and the total cost. In the rest of this section, we detail how these results can be utilized to analyze the technoeconomic dynamics.

Average Delay: The average delay is outputted from the QoS_Total_Delay() function. Fig. 7 presents the average delay experienced per client vehicle under two different deployment scenarios (i.e., NB and WB). We investigate two major components of delay, namely queuing delays and migration delays. Fig. 7 shows that, in a high mobility environment, the main challenge is the migration delays, which cause approximately 70% of the measured delay. Deployment of the VFN affects mainly the queuing delays (i.e., up to 10% decrease) based on our task allocation strategy. More specifically, we do not reserve or direct any VFN resources to explicitly handle mobility challenges, but instead, we are demonstrating flexible capacity

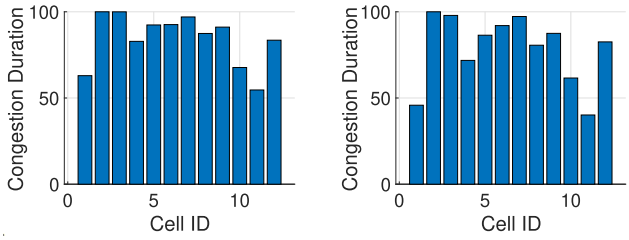

Fig. 8. Estimated cell congestion duration for deployment options NB (on the left) and WB (on the right) for $|K| = 50$ and high task generation rate.

TABLE VI
ESTIMATED NETWORK CONNECTION ($N \cdot C$) AND ACCEPTANCE RATIO ($A \cdot R$) FOR FOUR DIFFERENT SCENARIOS FOR $|K| = 100$

Scenario	$N \cdot C$ (%)	$A \cdot R$ (%)
NB	99.97	82.15
WB	99.97	91.22
InNB	99.97	98, 45
B	99.97	30, 92

management with this periodically available computation capacity. In the NB scenario, the users experience a 10.6026 TTI delay on average, whereas, for the WB scenario, the experienced delay is 10.5144 TTIs since more capacity becomes available with the deployment of VFNs.

Fig. 8 demonstrates the congestion levels of base stations under two different deployment strategies, provided by the QoS_Cell_based() function. Here, we measure the TTIs where the base station has at least one queued client vehicle over the complete simulation horizon. As envisioned, the simulation result demonstrates that the deployment of VFNs would create relaxation on the stationary fog nodes. In particular, in the NB scenario, the average cell congestion over all cells is 84.3422%, whereas, for the WB scenario, the average congestion level is 78.6285%. The deployment of VFNs decreases the overall congestion level 6% due to more sufficient capacity brought by the VFNs. Note that, unlike the InNB scenario, the relaxation in WB is affecting a subset of cells (e.g., cells 1, 4, 5, 8, 11, and 12 in Fig. 8) since the buses which carry the VFNs pass through these cells along their routes. Therefore, the deployment of VFNs needs to be planned according to the objective to achieve the maximum impact.

Technoeconomic Aspects: The applicability of any technical solution in a real-world scenario depends not only on the QoS achievements but also on the economical implications posed by the solution. VFogSim provides the necessary statistics to perform a technoeconomic analysis. We present the effects of various deployment options on the network connection and acceptance ratios in Table VI. The network connection and acceptance ratio are provided by the QoE_Achieved_rate() function, the former indicated from the connectivity service A1 and the latter from the vehicular applications A2, A3, and A4. In this scenario, we assume that the operator has sufficient network resources to guarantee spectral availability for all the client vehicles. Therefore, the application of VFNs does not impact the network connections. Consequently, we observe a connection of

TABLE VII
ESTIMATED ACCEPTANCE RATIOS FOR DIFFERENT SERVICES FOR FOUR DIFFERENT SCENARIOS FOR $|K| = 100$ AND HIGH TASK GENERATION RATE

Scenario	A_1 (%)	A_2 (%)	A_3 (%)	A_4 (%)
NB	99.64	71.7	72.76	38.37
WB	99.64	83.12	84.61	40.78
InNB	99.64	95.7	96.9	43.07
B	99.64	27.45	27.45	11.67

TABLE VIII
ESTIMATED ACCEPTANCE RATIOS FOR DIFFERENT SERVICES FOR FOUR DIFFERENT SCENARIOS FOR $|K| = 100$ AND LOW TASK GENERATION RATE

Scenario	A_1 (%)	A_2 (%)	A_3 (%)	A_4 (%)
NB	99.86	80.88	82.76	53.58
WB	99.86	88.9	90.12	54.78
InNB	99.86	98.95	99.33	56.52
B	99.86	32.19	32.7	19.29

99.97%, indicating that almost all the vehicles are connected to the cells at each time slot.

In terms of the acceptance ratio, we observe that the application of a hybrid platform can increase the efficiency by approximately 11%, while the InNB scenario provides the highest efficiency. On one hand, considering the doubled resources, the observed increase in performance in the InNB scenario can be explained. On the other hand, the applicability of this strategy depends on the additional economical pressure it would create. We assume that the additional cost of deploying fog nodes and the respective operational costs are reflected in a proportional increase in the service price to the end client vehicles. We analyze the client vehicles' acceptance probability of the received service quality for the given price following the relation from [7]:

$$\left(\frac{U_{s1}}{U_{s2}}\right)^{\mu} \leq \left(\frac{P_{s1}}{P_{s2}}\right)^{\epsilon} \tag{5}$$

where the parameters μ and ϵ represent the sensitivities to the utility and the price, respectively, and we assume that they are equal. The left-hand side measures the increase ratio in the utility, while the right-hand side measures the price increase by substituting an existing strategy (e.g., NB) with one with higher capacity. Considering the InNB scenario, doubling the existing resources also doubles the total price, while the increment on the measured utility is relatively too low. Therefore, the inequality in (5) does not hold. However, when we analyze the hybrid computing solution (i.e., WB), as long as we ensure that the price of VFC is equal to or less than 12% of the standalone solutions, the client vehicles would accept the changes in the prices, and WB would become the most cost-efficient solution.

Interservice Prioritization: As detailed in the earlier sections, we map four different service types in our evaluations. In this part, we consider two task generation rates, namely low rate and high rate.

The background traffic is considered to have no computational requirements and is modeled as a live signal of the vehicles. Therefore, this service type has relatively low data rate expectations and strict latency requirements. The background acceptance ratios, as reported in Tables VII and VIII, indicate the connection status of the vehicles. Considering that this service

TABLE IX
ESTIMATED ACCEPTANCE RATIOS FOR DIFFERENT SERVICES FOR FOUR DIFFERENT SCENARIOS AND THE NUMBER OF VEHICLES, $|K|$

Scenario	$\|K\|$	A_1	A_2	A_3	A_4
NB	50	99.60	87.31	88.43	69.38
NB	100	99.64	71.7	72.76	38.37
NB	150	97.27	64.5	63.43	29.29
WB	50	99.6	90.97	91.9	70.24
WB	100	99.64	83.12	84.61	40.78
WB	150	97.27	75.49	76.25	31.53
InNB	50	99.60	99.58	99.76	71.57
InNB	100	99.64	95.7	96.9	43.07
InNB	150	97.27	89.31	91.22	34.48
B	50	99.60	32.23	32.47	25.83
B	100	99.64	27.45	27.45	11.67
B	150	97.27	25.58	25.16	9.66

has very high priority in the network scheduler, and it is well covered in the urban area, more than 99% of the vehicles are always connected to a cell.

The remaining three service types are prioritized based on the previously detailed aspects in function (4a). Among them, object detection and lane detection service types have higher priority than video transcoding services. This interprioritization is visible in the acceptance ratios. Both for WB and InNB scenarios, the prioritized services (i.e., lane detection and object detection) have an increase in their overall acceptance ratio between 10% and 20%, while the nonprioritized services (i.e., video transcoding) report a negligible increase in their overall acceptance between 3% and 5%.

The acceptance ratio difference between the NB scenario and WB scenario indicates that the deployment of VFNs mainly impacts the prioritized services since the VFNs deployed on buses have limited capacity and coverage area. Note that when the capacity and coverage area of VFNs increase, they can also affect the non-prioritized services. Moreover, this occurrence also results from our deployed scheduler in (3a) and (3b). As detailed in the previous sections, it is possible to test different scheduling and task allocation models that can change the results.

Impacts of Traffic Load: In this part, we analyze the implications of increasing the number of vehicles on the service acceptance rate. To reflect the worst-case scenario, we assume the high task generation rate, and the results are presented in Table IX. First of all, the network connection of the vehicles, i.e., the background service, decreases to 97.27% for $|K| = 150$. The loss of approximately 2% is due to the coverage holes in the considered region and shows that the use cases that require high reliability demand new network solutions. Regarding the considered deployment scenarios, we can observe that, for a few numbers of client vehicles, the performance of WB and InNB is comparable with each other. Therefore, when the traffic volume is low, the deployment of VFN provides a more flexible yet efficient network solution.

Although all the deployment options present a decrease in acceptance ratio when the traffic load increases, we can see that the standalone VFN deployments (i.e., deployment case B) present a more than 25% acceptance rate. This result indicates that, although this standalone deployment option is not feasible for urban scenarios, it has the potential to be applied in rural areas.

TABLE X
ESTIMATED ACCEPTANCE RATIOS FOR DIFFERENT SERVICES FOR FOUR DIFFERENT SCENARIOS FOR $|K| = 100$ AND HIGH TASK GENERATION RATE WITH PRICE DIFFERENTIATION

Scenario	A_1	A_2	A_3	A_4
NB	99.64	49.87	74.92	43.33
WB	99.64	69.31	86.76	43.47
InNB	99.64	90.63	99.09	43.36
B	99.64	17.47	28.6	13.96

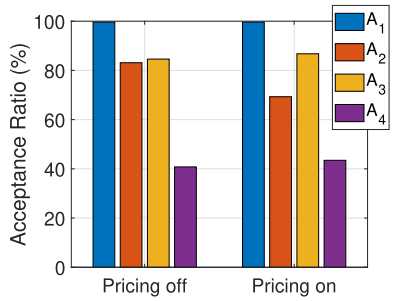

Fig. 9. Acceptance ratios per service type for different pricing strategies for WB deployment scenario and $|K| = 50$.

Impacts of Pricing: Finally, we analyze the implications of different prices for different services. In our task scheduling algorithm, the price of a service proportionally increases the priority of the service. In this part, we demonstrate the impact of different pricing options on resource allocations. In particular, we assume that the service provider charges higher for elastic services and cheaper for nonelastic services. Therefore, despite the underprioritization of video transcoding services, the service provider would have a higher incentive to serve this client vehicle service. To analyze this scenario, we assume that the prices of video transcoding services are ten monetary units (MU), while the object detection and lane detection services are prices of 2 MU and 5 MU, respectively. Because the background services are modeled as live signals, we assume that they are priced the least, i.e., 1 MU. Table X presents the measured acceptance ratios for different deployment options.

Comparing Table VII with Table X, we can see that the background services are not affected by the lower price. This is due to the fact that the prioritization mechanism in the network scheduler is not directly affected by the service price. However, we can observe that this new pricing strategy mainly affects the object detection services, which have the second least price. On the other hand, in terms of the lane detection services, we do not observe any major drop even with the relatively low price. This is because the prioritization mechanism we use relies on a multitude of aspects apart from the service prices. With this new pricing strategy, the video transcoding service gains a higher acceptance rate, showing that the additional incentive produced by the higher price is effective in decision making.

Fig. 9 visually presents the impact of pricing strategy on the acceptance ratio of each service type, where the risk of unstable pricing can be observed. More specifically, the different prices

for services cause an approximately 10% decrease in object detection services, while the other two services report a less than 5% increase. This result demonstrates that, even though the pricing strategy can affect the service-based acceptance rates, the price should reflect the supply and demand aspects of a particular service.

VI. Discussion

In this section, we summarize the key characteristics of our simulation platform and detail our future directions.

Functionality: The results in Section V demonstrate that VFogSim provides the necessary functionality to model and test various deployment options. Furthermore, the simulator output provides both the perceived quality and the total cost to the end-user, which allows us to conduct a deep analysis of the technoeconomic implications of VFC.

Input Flexibility: VFogSim can use either synthetic data or real-world measurements as inputs (cf., Fig. 2 and Table III). This versatile input option enables testing of a large variety of use cases, including real-world implementations and future potential deployment options.

Prioritization: The prioritization mechanism developed in this framework contains both intertenant and interservice dynamics. For a tenant to have a higher priority, e.g., premium service, the $U_{\text{th},m}$ parameter needs to be greater than the other tenants. The interservice prioritization can be active in the network scheduling part, fog scheduling part, or both. The prioritization in the network scheduler is done by changing the R_1, R_2, R_3, U_1, U_2, and U_3 parameters. Being a max-rate scheduler, the network scheduler gives higher priority to the services with a higher gradient value. On the other hand, the fog scheduler is designed to give a higher priority to the services with lower remaining execution time or larger requested resources. Note that this prioritization mechanism can easily be changed by updating (4a).

Modularity: VFogSim is designed and developed in a modular structure to allow customization. It is possible to integrate different schedulers, prioritization mechanisms, or pretrained AI models (see in Section V). Since the task scheduler and network scheduler are separated in our model, they can be customized individually based on the user-defined scenario.

Extendability: VFogSim is extendable to respond to the evolution of the network and edge computing technologies in the intelligent transportation system. Although we performed our analysis for 5G, it is possible to extend it for 6G and beyond networks [37]. For example, we can change base station characteristics by substituting the network configuration file in WinProp and adjusting the traffic behavior by adding more autonomous vehicles in SUMO [38]. Moreover, the simulator can work with different resource schedulers.[2] On the other hand, it is also possible to replace the commercially available tool (i.e., WinProp) with OMNeT++. In particular, VFogSim uses the input of the vehicular traces and their respective achievable rates. It is possible to use Veins (SUMO with OMNeT++) to generate these instances and transfer these input metrics to VFogSim to run the simulation, which will be supported in future versions.

Scalability: Despite the analysis being performed on a relatively small network, VFogSim can accommodate more realistic system scenarios with a larger set of client vehicles, VFNs, and base stations. In particular, the designed framework is not built upon a fixed number of users or base stations. From the input perspective, the increased number of users would not affect the application of the algorithm. Moreover, considering that the scheduler is run per base station, the increased number of cells would not affect the overall execution of the algorithm. On the other hand, the time complexity of the overall simulator depends on the used scheduling algorithms and the simulation scenario. We demonstrated the increase in the time complexity with the number of client vehicles in Section IV.

Limitations and Future Directions: The current version of VFogSim supports 5G NR V2N. In the future, we will extend VFogSim to support both 5G NR V2N and V2V modes. In addition, in the current version, VFogSim generates SINR maps from real-world measurements or using WinProp. In future versions, such data could be provided by open-source simulators, such as Veins with Simu5G [16], [25]. Besides, we deploy VFNs as buses that have fixed routes and timetables. In the future, we will consider other types of vehicles to carry VFNs, such as taxis and passenger cars. Another limitation of the current platform is that different blocks presented earlier in Fig. 2 communicate in an offline manner; thus, the technoeconomic evaluation only starts when capacity plans are provided. An online version is needed in the future. For example, we can extract real-time vehicular location from SUMO using Traffic Control Interface [39] and adjust capacity plans while assessing technoeconomic performance.

VII. Conclusion

This work presented VFogSim, a modular and customizable simulation platform for testing and evaluating technoeconomic implications of different VFC deployment options and use cases. VFogSim supports the mobility of fog nodes and contains realistic modeling of vehicle traffic, resource consumption, and wireless communications. We evaluated the accuracy and scalability of our platform by comparing the network simulation results with real-world measurements. Furthermore, we demonstrated the use of VFogSim by using it for evaluating the technical and financial feasibility of different VFC capacity plans for an urban area. In practice, VFogSim can also be used for testing customized resource and task allocation strategies and prioritization mechanisms in diverse VFC environments and application scenarios.

[2]Further details can be found from supplementary materials at https://mobilecloud.aalto.fi/?page_id=1441

References

[1] A. Yousefpour et al., "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Archit.*, vol. 98, pp. 289–330, 2019.

[2] C. Zhu, G. Pastor, Y. Xiao, and A. Ylajaaski, "Vehicular fog computing for video crowdsourcing: Applications, feasibility, and challenges," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 58–63, Oct. 2018.

[3] M. Noreikis, Y. Xiao, and A. Ylä-Jaäiski, "QoS-oriented capacity planning for edge computing," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.

[4] W. Mao, O. U. Akgul, A. Mehrabi, B. Cho, Y. Xiao, and A. Ylä-Jääski, "Data-driven capacity planning for vehicular fog computing," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13179–13194, Aug. 2022.
[5] C. Zhu et al., "Folo: Latency and quality optimized task allocation in vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4150–4161, Jun. 2019.
[6] B. Cho and Y. Xiao, "Learning-based decentralized offloading decision making in an adversarial environment," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11308–11323, Nov. 2021.
[7] O. U. Akgul, I. Malanchini, and A. Capone, "Dynamic resource trading in sliced mobile networks," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 1, pp. 220–233, Mar. 2019.
[8] H. Gupta, A. V. 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," *Softw., Pract. Experience*, vol. 47, pp. 1275–1296, 2017.
[9] X. Zeng, S. K. Garg, P. Strazdins, P. P. Jayaraman, D. Georgakopoulos, and R. Ranjan, "IOTSim: A simulator for analysing IoT applications," *J. Syst. Archit.*, vol. 72, pp. 93–107, 2017.
[10] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of edge computing systems," in *Proc. 2nd Int. Conf. Fog Mobile Edge Comput.*, 2017, pp. 39–44.
[11] T. Qayyum, A. W. Malik, M. A. Khan Khattak, O. Khalid, and S. U. Khan, "FogNetSim++: A toolkit for modeling and simulation of distributed fog environment," *IEEE Access*, vol. 6, pp. 63570–63583, 2018.
[12] R. Mayer, L. Graser, H. Gupta, E. Saurez, and U. Ramachandran, "EmuFog: Extensible and scalable emulation of large-scale fog computing infrastructures," in *Proc. IEEE Fog World Congr.*, 2017, pp. 1–6.
[13] A. Coutinho, F. Greve, C. Prazeres, and J. Cardoso, "Fogbed: A rapid-prototyping emulation environment for fog computing," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–7.
[14] A. Brogi and S. Forti, "QoS-aware deployment of IoT applications through the fog," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1185–1192, Oct. 2017.
[15] A. Brogi, S. Forti, and A. Ibrahim, "How to best deploy your fog applications, probably," in *Proc. IEEE 1st Int. Conf. Fog Edge Comput.*, 2017, pp. 105–114.
[16] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Trans. Mobile Comput.*, vol. 10, no. 1, pp. 3–15, Jan. 2011.
[17] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2322–2358, Oct./Dec. 2017.
[18] "Study on enhancement of 3GPP support for 5G V2X services," 3GPP 22.886 Release 15, 2018.
[19] R. Calheiros, R. Ranjan, C. De Rose, and R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," 2009, *arXiv:0903.2525*.
[20] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. 1st Int. Conf. Simul. Tools Techn. Commun., Netw. Syst. Workshops*, 2008, pp. 1–10.
[21] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw.*, 2010, pp. 1–6.
[22] P. Wette, M. Draxler, A. Schwabe, F. Wallaschek, M. H. Zahraee, and H. Karl, "MaxiNet: Distributed emulation of software-defined networks," in *Proc. IFIP Netw. Conf.*, 2014, pp. 1–9.
[23] P. A. Lopez et al., "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 2575–2582.
[24] A. Virdis, G. Stea, and G. Nardini, "SimuLTE—A modular system-level simulator for LTE/LTE-A networks based on OMNeT++," in *Proc. 4th Int. Conf. Simul. Model. Methodol., Technol., Appl.*, 2014, pp. 59–70.
[25] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G—An OMNeT++ library for end-to-end performance evaluation of 5G networks," *IEEE Access*, vol. 8, pp. 181176–181191, 2020.
[26] R. Hoppe, G. Wölfle, and U. Jakobus, "Wave propagation and radio network planning software WinProp added to the electromagnetic solver package FEKO," in *Proc. Int. Appl. Comput. Electromagn. Soc. Symp.*, 2017, pp. 1–2.
[27] "Zero-touch network and service management (ZSM)," ETSI GS ZSM 003 V0.22.0, 2020. [Online]. Available: https://www.etsi.org/technologies/zero-touch-network-service-management
[28] "Cellmapper, cellular tower and signal map," Accessed on: Aug. 31, 2021. [Online]. Available: https://www.cellmapper.net/map
[29] G. Naik, B. Choudhury, and J.-M. Park, "IEEE 802.11bd & 5G NR V2X: Evolution of radio access technologies for V2X communications," *IEEE Access*, vol. 7, pp. 70169–70184, 2019.
[30] "Cellular-Z," Accessed on: Jan. 11, 2022. [Online]. Available: https://hzx0910.github.io/cellular-z/introduce.htm
[31] "iperf3," Accessed on: Jan. 11, 2022. [Online]. Available: https://software.es.net/iperf/
[32] U. Saeed, J. Hämäläinen, E. Mutafungwa, R. Wichman, D. González G., and M. Garcia-Lozano, "Route-based radio coverage analysis of cellular network deployments for V2N communication," in *Proc. Int. Conf. Wireless Mobile Comput., Netw. Commun.*, 2019, pp. 1–6.
[33] G. Jocher et al., "Ultralytics/YOLOv5: v3.1—Bug fixes and performance improvements," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4154370
[34] "COCO—Common objects in context," Accessed on: Aug. 31, 2021. [Online]. Available: https://cocodataset.org/
[35] "Opencv," Accessed on: Aug. 31, 2021. [Online]. Available: https://opencv.org/
[36] "Handbrake," Accessed on: Aug. 31, 2021. [Online]. Available: https://handbrake.fr/
[37] X. You et al., "Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts," *Sci. China Inf. Sci.*, vol. 64, no. 1, 2021, Art. no. 110301.
[38] M. Guériau and I. Dusparic, "Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst.*, 2020, pp. 1–8.
[39] "Traci," Accessed on: Mar. 13, 2023. [Online]. Available: https://sumo.dlr.de/docs/TraCI.html

Özgür Umut Akgül received the B.S. degree in electronics engineering and electrical engineering and the M.S degree in computer engineering from Istanbul Technical University, Istanbul, Turkey, in 2011 and 2014, respectively, and the Ph.D. degree in information technology from Politecnico di Milano, Milan, Italy, in 2019.

He is currently a Postdoctoral Researcher with the Department of Information and Communications Engineering, Aalto University, Espoo, Finland. His research interests focus on optimization models, mathematical programming, and machine learning.

Wencan Mao received the B.E. degree in vehicle engineering from the Wuhan University of Technology, Wuhan, China, in 2017, and the M.S. degree in mechanical engineering in 2019 from Aalto University, Espoo, Finland, where he is currently working toward the Ph.D. degree with the Department of Information and Communications Engineering

Her current research interests include edge computing, resource allocation, and capacity planning.

Byungjin Cho received the doctoral degree in communications engineering from Aalto University, Espoo, Finland, in 2016.

He is currently a Postdoctoral Researcher with the Department of Information and Communications Engineering, Aalto University. His research interests include resource managements in networked systems using algorithmic decision theory.

Yu Xiao (Member, IEEE) received the doctoral degree in computer science from Aalto University, Espoo, Finland, in 2012.

She is currently an Associate Professor with the Department of Information and Communications Engineering, Aalto University. Her current research interests include edge computing, wearable sensing, and extended reality.