Simulation Modelling Practice and Theory 121 (2022) 102657

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

# End-to-end simulation environment for mobile edge computing

Katja Gilly [a,*], Cristina Bernad [a], Pedro J. Roig [a], Salvador Alcaraz [a], Sonja Filiposka [b]

[a] *Department of Computer Engineering, Miguel Hernandez University, Elche, Spain*
[b] *Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University, 1000 Skopje, North Macedonia*

ARTICLE INFO

*Keywords:*
5G mobile communication
Edge computing
End-to-end delay analysis
Simulation
Workflow

ABSTRACT

Performances analysis of resource management techniques and algorithms for edge computing is a crucial step that reveals the effectiveness of edge service placement and migration strategies in terms of delay and usage. To be able to gauge how different optimisations affect the end user quality of experience a holistic approach is needed, so that the end-to-end delay can be measured in a combined setup where the edge infrastructure is integrated in the 5G system architecture. For these purposes we put forward an open source workflow based on the integration of proven simulators as SUMO, OMNeT++ and CloudSim, that will provide the means to create large scale urban mobile simulation environments. This paper discusses the benefits and challenges of such an approach, and provides example results that showcase a few aspects of the potential end-to-end delay analysis by processing the gathered simulation output.

## 1. Introduction

Today's staggering proliferation of new smart applications used by inherently mobile users are putting a tremendous strain on portable devices. Although the Moore's law keeps increasing the computing capacity of our mobile devices, the requirements for computing and storage that are imposed by this new generation of intelligent applications is insatiable [1]. To alleviate this problem many applications have been designed in such a way that the application "heavy lifting" is done away from the user equipment, i.e. in the cloud. Hence, while the seemingly never-ending computing and storage resources in the cloud are an advantageous solution to the problem, unfortunately they are not always as efficient if the results need to be obtained in a timely manner. Namely, for real- and near real-time applications, using the resources in the cloud is not an option due to the imposed communication delay because of the distance between the application data producer/consumer and the computing resources that process the data [2].

Edge computing is aiming to solve this problem by bringing the cloud closer to the user [3], as it is depicted in Fig. 1. With the computing resources located right next to the mobile base station, the incurred delay when offloading processing to the edge hosts drops down to the relative performances of the wireless communication technology. In essence, the edge computing strategy of expanding the mobile service provider infrastructure to encompass additional edge computing resources in the form of interconnected distributed micro data centres is only one part of the whole. The second part is the mobile communication that now needs to be able to support the additional requirements made by smart applications in terms of bandwidth and delay. For these purposes, edge computing has become an intrinsic partner of the 5G communication technology [4]. 5G provides the freedom of mobility without sacrificing performances (high bandwidth, low latency, low battery consumption, increased number of devices), while edge computing provides the answer to resource hungry applications. Together as a team they provide the means for development of killer apps. This symbiotic relationship will only continue to enhance as we move towards 6G and the development of edge intelligence [5].

* Corresponding author.
*E-mail address:* katya@umh.es (K. Gilly).

https://doi.org/10.1016/j.simpat.2022.102657
Received 16 November 2021; Received in revised form 16 September 2022; Accepted 17 September 2022
Available online 24 September 2022

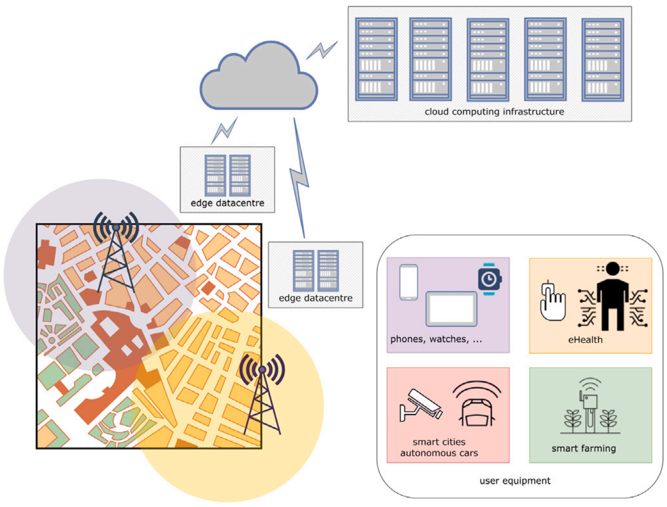


**Fig. 1.** Edge computing services and scenarios overview.

Because of this convergence of technologies where both edge computing and mobile communications play a vital part in the overall user perceived performances and experience, it is essential that they are analysed in unison. In other words, when analysing the performances of edge computing resource management solutions or scheduling algorithms, to gauge the effect on the end user, one must consider how these will perform in a combined environment, measuring the overall end-to-end performances, with delay being one of the main performance indicators [6]. Sometimes this becomes a difficult task since these parts inherently come from two different fields of research, one from computer engineering, and another from telecommunications. Thus, a multi-disciplinary approach is needed for a comprehensive understanding of the constraints and possibilities. Taking this holistic approach, it is evident that when aiming to design an efficient overall system there are two aspects that need to be considered: the networking aspect comprised of the wireless and the optical backbone network communication, and the overall virtualised computing resource management that needs to solve the problem of placement and migration of user services using virtual machines or containers.

The simulation of this system in an environment where all relevant aspects concerning edge computing are addressed is far from trivial [7]. With the aim to be able to approach edge computing and 5G as a combined problem that needs to be tackled as a whole in order to perform an encompassing end-to-end analysis, the goal of this paper is to present an approach that enables researchers to create complex simulation scenarios that incorporate all aspects of both systems such as user mobility, wireless communication, and distributed edge computing resources that work together to provide a full application service management lifecycle. Focusing on the problem of determining the end-to-end latency, we have decided to combine together simulation tools that are used as de-facto standard tools when analysing different isolated aspects of the environment: SUMO [8] for urban mobility scenarios, OMNeT++ with a 5G extension [9] for the wireless channel, and CloudSim with Edge Computing [10] for resource management. The description of the proposed workflow and all related software including pre and post processing scripts are made publicly available on GitHub [11] under the Creative Commons licence.

The major contributions of this paper are as follows:

- A workflow that integrates several popular simulators for studying mobile edge computing environments and their performance is provided. The workflow input is defined using a large number of parameters that enable users to setup a vast variety of simulation scenarios, including dense and/or urban 5G-MEC scenarios. The proposed workflow is formally validated and its performances regarding the use in large scale simulation scenarios are discussed.
- Open source end-to-end delay analysis tools and scripts are provided to practically implement the proposed workflow by linking all simulators and producing graphical output that enables in-depth observation of the obtained results and performance metrics. All code and scripts are made available in a public repository and they can be re-used and extended by the research community.
- A vehicular MEC case study is described in detail as an example of our workflow implementation. The case study results showcase the use of the delay analysis tools in a dense urban environment.

The remainder of this paper is organised as follows: Section 2 discusses the related work in terms of existing research on the problem of latency and efficiency analysis in 5G edge computing scenarios and related simulation tools. Section 3 introduces the concept of developing a holistic end-to-end delay analysis simulation environment covering all relevant aspects and parameters. Section 4 provides an example of using the proposed suite and discusses the sample results. Finally, Section 5 concludes the paper and tackles open issues and possible future work directions.

## 2. Related work

An essential part of the design process of any complex system is to define the theoretical foundations and employ first order theoretical analysis that will validate the model and provide first order insight into its potential performances. When it comes to the stated problem of combining 5G and MEC, given the complexity of the two when considered independently, it becomes exponentially difficult to perform a theoretical analyse of a combined 5G-MEC system that will take into account the variabilities and parameters that can affect the overall performance. The most prominent theoretical formal approaches include the use of Markov processes, however their use is focused on one particular aspect of the problem, not taking into consideration other components and their influence [12]. When applying theoretical approaches to large scale distributed systems one must deal with problems such as state explosion and compacting and aggregation. Thus, the theoretical models are always presented with a very limited set of carefully chosen parameters, making assumptions or declaring other parameters negligible [13]. The initial analysis is made for a very simple small scenario: single base station, single MEC service, less than 10 users [14]. Of course, no matter if it is Markov processes, graph theory, game theory or optimisation theory, the initial theoretical design when devising a new aspect or a new algorithm for a system is essential. The analysis of these models is very fast and can minimise the time needed to conclude if the chosen approach has any restrictions, as well as what are its best expected performances. However, due to the existing limitations, when working with large scale distributed systems, further analysis using simulation is needed to fully understand how the components works together with the rest of the system.

Considering previous efforts of mobile edge computing related work, we observe that the topic of end-to-end delay performance analysis is one of the most important when it comes to the implementation of an efficient edge computing and/or 5G system. Hence, there are many research papers that focus on different aspects of the matter.

End-to-end latency analysis in pure 5G networks implemented using Network Function Virtualisation (NFV) is the topic of [15]. The authors aim to model multiple flows in one NFV chain focusing on delay queues and packet processing. They use OMNeT++ to verify their analytical delay models. End-to-end congestion control algorithms that aim for high throughput and low latency over 4G/5G network links are surveyed in [16]. The reported simulation tools used in the papers are NS2 or NS3 for validation and analysis. In [17] the trade-off between energy efficiency and low latency requirements is analysed. Architectural solutions and procedures are discussed to help achieve low latency while minimising energy expenditure. Reliable communication with guaranteed delay is considered in [18] where eRESERV is proposed as a way to enhance the reliability of a service function chain (SFC). The proposed solution is analysed using queueing theory and is simulated in MATLAB Simulink. A study with similar goal is presented in [19] where a subchaining method is proposed for a reliable SFC with guaranteed delay. The performances of the proposal are analysed using MATLAB Simulink.

While MATLAB Simulink seems to be extensively used for validating analytical approaches that involve queueing, in [20] the authors analyse the network simulators that are used to create 4G and 5G network simulation scenarios. They analyse the features of NS3, NetSim, OMNeT++, and Riverbed and conclude that the choice depends on the study motive. Although the authors state that OMNeT++ does not have support for 5G, at the same time, in [21] the authors present an extension of OMNeT++ for 5G networks.

When it comes to delay analysis and performances in edge computing, the research literature becomes more complicated. Many papers are focusing on the application service placement problem only while analysing static scenarios without any user mobility. Such is the example of [22] where the authors propose a game approach to allocating resources in a hierarchical architecture for Smart Homes. The proposed algorithms are evaluated using MATLAB. Similarly in [23] the author proposes a quality of service aware placement of tasks in an edge computing environment and compares the result with several basic scheduling algorithms using Cloud Analyst. The main constraints in these examples is that the simulation scenarios are overly simplistic when taking into account that the edge computing services are mainly intended to be used in environments with mobile users. Also, when the authors propose the placement algorithms they do not analyse their impact on the rest of the 5G edge system, analysing only the performances of the edge servers and the latency within the backbone edge network.

More complex scenarios are introduced in studies that incorporate user mobility. In this case, the research usually focuses not only on service placement, i.e. resource allocation, but also on service migration, i.e. resource migration for a continuously optimised use of the available edge resources. Many examples that incorporate high mobility and migration of services are focusing on vehicular networks such as [24] where the authors propose an intelligent edge computing resource management system based on deep reinforcement learning and cooperation. Other example is focusing on multimedia services where using a binary particle swarm optimisation algorithm they achieve high quality of service as presented in the simulation results obtained from CloudSim [25].

However, many edge computing resource management focused papers are not including the wireless channel modelling when analysing the performances of their proposal, as is the case with [25]. There are many simulators that are used in edge computing related research such as IoTSim-Edge, EdgeCloudSim, iFogSim, CloudSimSDN, all of which are based on the CloudSim simulator and incorporate some additional features, such as a wireless channel but only up to 4G [26]. As the joint approaches to modelling 5G and edge computing are now becoming the focus of the research [27,28], it is ever more important to be able to create an extensive mobile urban simulation that will be used to analyse the end-to-end latency performances of newly proposed approaches in a holistic manner that reflects the 5G and edge computing symbiosis.

We have compiled the most popular tools for simulating 5G and edge scenarios in Table 1 providing a description of them, the simulation tool they are based on, if any, and detailing the features that are not contemplated in their implementation. The intention of providing the information of the features not covered is to remark that none of the tools implements all the needed features required for simulating an end-to-end edge system that operates with 5G protocols, that can be summed up in 4 generic characteristics: the network stack implementation, 5G protocols implementation, compute virtualisation implementation and mobility models implementation.

**Table 1**
Comparative chart of simulation tools for 5G and edge scenarios.

| Simulation tool | Based on | Description | Lack of | Perform. analysis |
|---|---|---|---|---|
| CloudSim | | Popular discrete event cloud management simulation tool | 🟢, 🟠, 🟣 | [29] |
| OMNeT++ | | Popular discrete event network simulation platform that implement Internet stacks and other network protocols. Some extensions implement different technologies as 4G, 5G, vehicular networks, etc. | 🔵, 🟣 | [30] |
| SUMO | | Popular traffic modelling simulator that models intermodal traffic systems over road networks that can be imported from OpenStreetMap (OSM) | 🟢, 🔵, 🟠 | [8] |
| NS2 - NS3 | | Discrete event simulator for network protocol modelling and simulation | 🔵, 🟠 | [31] |
| MATLAB Simulink | | Programming environment for modelling, simulating and analysing multi-domain systems from a theoretical point of view | 🟢, 🔵 | [32] |
| NetSim | | Discrete event simulator for network protocol modelling and simulation | 🔵, 🟠 | [33] |
| Riverbed Modeller Academic edition | OPNET Modeller | Discrete event simulator for network protocol modelling and simulation | 🔵, 🟠, 🟣 | [34] |
| IoTSim-Edge | CloudSim | IoT and edge computing extension that also models some network protocols | 🟢, 🟠 | [35] |
| EdgeCloudSim | CloudSim | Fog and edge computing extension that implements nomadic mobility and dynamic network modelling | 🟢, 🟠 | [36] |
| iFogSim | CloudSim | Fog and edge computing extension that models the behaviour of sensors and actuators that represent IoT devices | 🟢, 🟠, 🟣 | [37] |
| CloudSimSDN | CloudSim | SDN and NFV extension | 🟢, 🟣 | [38] |
| IOTSim | CloudSim | Application processing extension using the MapReduce model for concurrent processing of data collected from IoT device | 🟢, 🟠, 🟣 | [39] |
| IoTSim-Osmosis | CloudSim | IoT and edge computing extension that implements SDNs and SDWANs. | 🟠, 🟣 | [40] |
| FogNetSim++ | OMNeT++ | Fog extension that implements several mobility models, handoff management and fog communication protocols | 🔵, 🟠 | [41] |

🟢 Network stack implementation modules, 🔵 Compute virtualisation modules,
🟠 5G modules, 🟣 Mobility models.

## 3. E2E delay analysis simulation environment

The end-to-end delay for a given 5G-MEC scenario refers to the total service delay experienced in the communication between the MEC application and the user equipment. The framework to perform an end-to-end network connection delay analysis by simulation has to necessarily encompass configuration attributes for the backbone infrastructure of the cloud datacentre, the edge layer infrastructure and the wired and/or wireless access network from where the MEC service is demanded. In this setup the end-to-end delay can be obtained by defining several points of measurement: the user equipment, the base station, the virtual machine (VM) where the MEC application is running. At each of these measurement points we are interested in the timestamps of the following events: user equipment is handed over to a new base station, user equipment sends a packet to its MEC service, user equipment receives a packet from its MEC service, base station receives a packet from user equipment and forwards to the MEC infrastructure, base station receives a packet from MEC infrastructure and forwards to user equipment; MEC service receives a packet and MEC service sends a packet. When analysing these events it is evident that the base station is the point where a network technology changes with 5G wireless on one side and optics infrastructure on the other. Thus, we can use the base station as a pivot point to calculate the end-to-end delay as a sum of the delay on the wireless channel between the user equipment and the base station and the delay on the wired side between the base station and the host, or in particular the VM, where the MEC service is running. If at some point the capacity of the MEC hosts is filled and a new MEC service cannot be placed on the edge, the corresponding VM is placed in the backbone cloud in which case the delay from the base station to the VM encompasses the delay in the edge layer and the delay in the backbone on the path the packets travel to reach the cloud VM.

Using this approach, it is therefore needed to combine at least two simulation tools to completely define a 5G/edge environment and generate urban mobile scenarios with the aim to obtain the perceived end-to-end delays as described. To successfully combine the simulator tools additional set of pre and post processing routines is needed. In this section we describe in detail how we have configured and connected a 5G and edge computing simulators by developing a customised workflow for this purpose. We also discuss the difficulties we have encountered and the open challenges that need to be overcome.

### 3.1. 5G enabled edge computing simulation workflow

It can be observed in Table 1 that the only two simulators that include 5G modules are OMNeT++ and MATLAB Simulink (in this case, 5G implementation is restricted to the physical layer). We have therefore selected OMNeT++ as the discrete event simulation tool to simulate the 5G access network in our simulation environment scenario. Namely, 5G based vehicular networks can be easily configured combining OMNeT++ with the 5G-Sim-V2I/N libraries [9] that includes the INET suite for wireless mobile networks, Veins to run vehicular networks and SUMO to define the urban area and create realistic mobility patterns using the input from real life OpenStreetMaps. With this combination of tools, performance results of all implemented protocols of the network protocol stack in the 5G access network can be obtained in the form of scalar and vector metric files when running OMNeT++ simulations. This OMNeT++ combination of libraries supports the design of complex simulation sets representing scenarios where different kinds of networking use cases of interest can be used such as vehicular networks, autonomous cars, geographical map based scenarios, smart cities, etc.

The OMNeT++ framework includes implementation libraries for mobile networks (LTE and 5G), that combined with SUMO, INET and Veins, can be used to compose a global coverage area of several base stations in urban dense scenarios for testing. Multiple variations of simulation scenarios can be run in parallel while obtaining a whole set of micro scale performance metrics. Simulation results are obtained as plain text files, and can be either post processed with externally developed tools and scripts, or can be initially analysed using the OMNeT++ result analysis tool. The size of the result files can easily be in the magnitude order of gigabytes.

While OMNeT++ is used for the 5G access network, the MEC part together with the wired network infrastructure is simulated using a combination of original and customised libraries in CloudSim [29] that models the edge and the cloud network topology and their associated computing facilities being CloudSim the most appropriate simulator for modelling computing virtualisation entities as it can be observed in Table 1. The CloudSim simulation scenario details the network topology that represents the edge computing backbone network and defines which policies are going to be used for the edge computing resource management during the simulation. Researchers focusing on edge computing resource management need to develop their own custom versions of these policies which can then be cross compared with the ones already available. The goal of the optimisation oriented resource management policies that need to be implemented in CloudSim is how to select the computing facilities where each edge services will run. CloudSim represents each edge service as a cloudlet that is run into a VM instantiated on a given edge server or host. Good policies need to make sure that the location where the edge service will be placed is as close as possible to the end user so that the delay is minimised even in the cases of mobile end users where conditions change over time. Thus, two fundamental operations are considered in the resource management policies: on one hand, allocation of edge services at the infrastructure network that is performed based on the location of the user equipment that asks for the service focusing on minimising the possible network delay. On the other hand, migration operations to minimise delays are considered when the user equipment changes from a 5G cell to another and there are enough available resources in the MEC hosts that are attached to the base station that serves the new user location. Aiming to optimise the placement and migration policies for MEC, custom resource management algorithms are used. The details of the resource management strategy employed by the algorithms together with their performance analysis are provided in [10]. The defined edge computing extensions for CloudSim to enable researchers to simulate the role of the MEC services orchestrator as defined in the ETSI MEC reference architecture [42]. Implementation information of the functionalities of the MEC orchestrator in CloudSim and some basics of vehicle to infrastructure (V2I) communication specifications are described in [43].

The simulation results in the form of output logs obtained from both simulators are used as input for custom developed scripts that aggregate and analyse the data. The output of these scripts is the end to end delay information and related parameters which can be then properly visualised. The whole workflow can be described through the following steps:

(i) choose a simulation area from OpenStreetMaps and define the coordinates of the 5G base stations and co-located edge servers;
(ii) import the map characteristics into SUMO that will model the vehicular (or pedestrian or other) nodes mobility pattern;
(iii) define the 5G base stations locations together with the channel and traffic characteristics for the 5G OMNeT++ library;
(iv) run the OMNeT++ simulation (you can run multiple simulation scenarios in parallel);
(v) analyse the OMNeT++ massive log files through parallel post-processing routines that extract the time-based access network delay and handover events for every mobile node;
(vi) use the output from the previous step that defines the edge services and their migration pattern as input to CloudSim where the edge servers and edge network are defined in the simulation configuration file that defines the chosen placement and migration policies;
(vii) analyse the CloudSim output logs to obtain network delays associated with the backbone network infrastructure that represents the cloud and the extension infrastructure that represents the network edge. Bash and R post-processing jobs are performed in this case in order to obtain all details related to the location of the virtualised edge services in the network topology and the geographical reference of the coverage area;
viii) combine the service delay times per mobile node from both OMNeT++ and CloudSim to obtain the total end-to-end delay per user/edge service communication by running customised post-processing code based on vaex and pandas data analytics libraries.

The main goal of the proposed approach is to define a simulation workflow that is based on proven and popular open source simulators augmented with pre and post processing scripts that are also made available as open source on GitHub together with a step by step description on how to use all of the tools to create different simulation scenarios [11]. By providing the full open

| Step number | Scrip name in GitHub repository [11] |
|---|---|
| 8 | mobileHandover_and_Delay |
| 13 | cloudsim_postprocess<br>cloudsim_postpro_R |
| 14 | PostProcess_E2Edelay |
| 17 | e2eDelay_plots<br>e2eDelay_bins_plot |

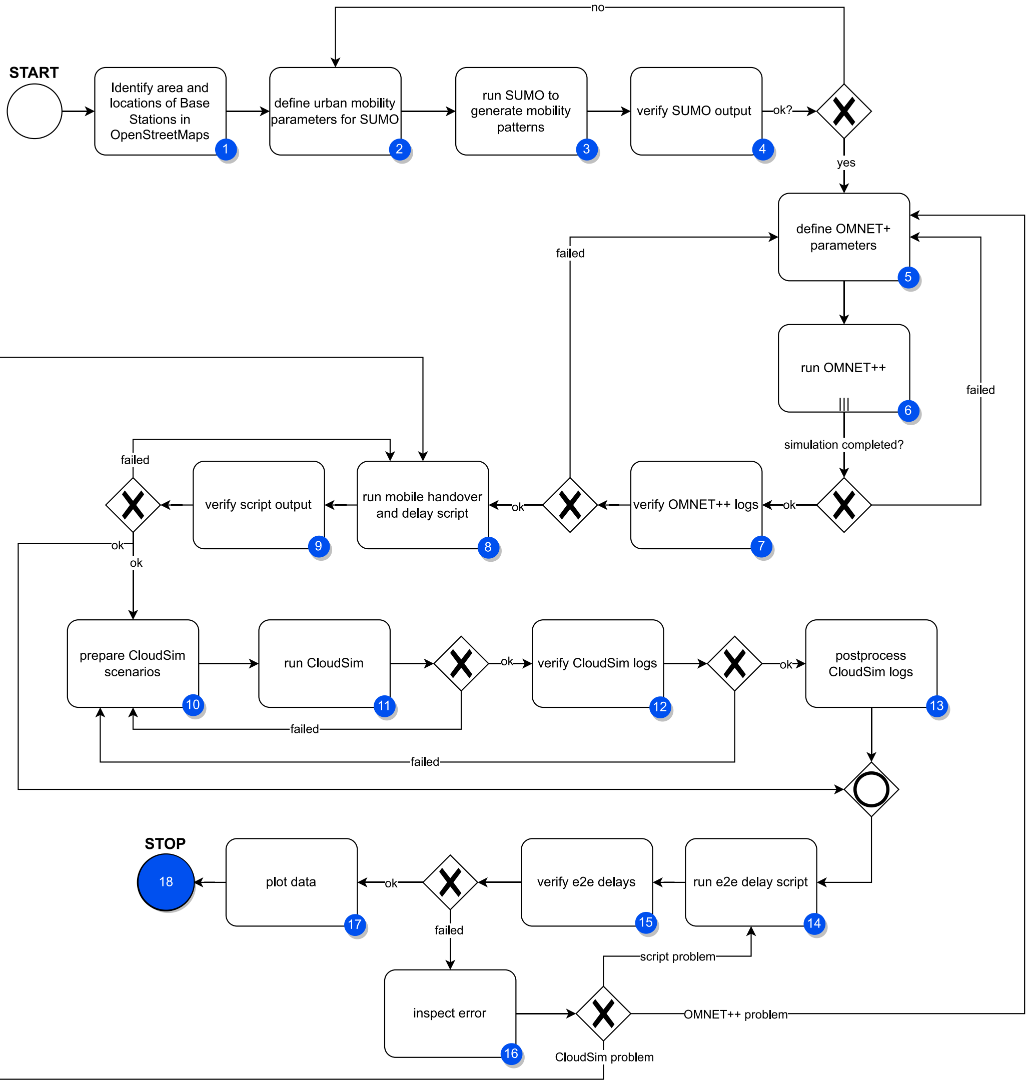


**Fig. 2.** Workflow validation process.

source code for the complete workflow, reproducibility of the research data is supported facilitating cross comparison of different algorithms and optimisation techniques.

The proposed workflow consists of readily used components: SUMO, OMNeT++ and CloudSim, customised extension for 5G support in OMNeT++, custom extension for placement and migration policies in CloudSim, and additional scripts and Jupyter notebooks that are used to bind the components together. Thus, the correctness of the approach depends on the validity of its components, extensions and any additional processing. All three simulators: SUMO, OMNeT++ and CloudSim are extensively used by the research community and their validity is widely accepted. The proposed extensions on the other hand, have already been theoretically and practically validated elsewhere. Namely, the 5G extension for OMNeT++ including its modelling based on the ITU Guidelines, the 3GPP specification and the well-known SimuLTE/Simu5G and the implementation for OMNeT++ is presented in [44]. The CloudSim extension for community based MEC services is based on multi-objective optimisation and graph theory and has been theoretically introduced and validated in [10]. Thus, in this work we have focused on verifying the processing scripts and validating the workflow itself.

We have configured a validation process of the workflow presented in this paper by using a formal validation technique. We firstly present in Fig. 2 a business process notation diagram that details the steps taken during the design of the end-to-end 5G-based edge computing simulation workflow. As described in the figure, the output of each separate component used in the workflow has been analysed and verified before proceeding to the next step of the workflow. The verifications performed in these steps are first sanity checks that the output corresponds to the expected given the defined simulation parameters. This is done by checking the ranges of values for the output parameters of interest (i.e. for SUMO we check that all coordinates of nodes are within the defined area at all times and their velocity is never above the max limit; for OMNeT++ we check that the simulation is successfully completed and there is log information for the whole simulation duration, that all nodes were able to send and receive information; for CloudSim the checks include the verification of simulation scenario completeness with all services provisioned on existing MEC hosts or in the cloud). The postprocessing scripts are also verifying that all expected input is present before continuing with the analysis, and they perform a final check of the output by proceeding to the next step. All pre- and post-processing task scripts and Jupyter notebooks [11] position in the related steps of the workflow are also indicated in Fig. 2. The whole workflow has been tested first with small scale scenarios so that the step by step outputs can be double checked and cross-compared with other tools available for analysis such as the built-in tools for statistical analysis in OMNeT++. After this initial verification, large scale simulation scenarios were run with the workflow and several iterations of script optimisations have been done in order to improve the performances and reduce the processing time.

Listing 1: Pseudocode of Promela implementation of the concurrent processes defined in the workflow of Fig. 2

```

#define In(Ti) = Set of input places Pi1 , ..., Pin of Ti (i = 1, ..., m)
#define Out(To) = Set of output places Po1 , ..., Pon of To (o = 1, ..., m)
#define Ex(In(Ti)) = (Pi1 && ... && Pin )
#define Clear(In(Ti)) = Pi1=0; ...; Pin=0
#define Set(Out(To)) = Po1=1; ...; Pon=1

bool P1=0;
...
bool Pn=0;
bool deadlock=0;

proctype Workflow()
{
        do
        :: atomic { Ex(In(T1)) -> Clear(In(T1)); Set(Out(T1)); }
        ...
        :: atomic { Ex(In(Tm)) -> Clear(In(Tm)); Set(Out(Tm)); }
        :: else -> goto dead
        od;

        dead: deadlock=1
}

init {
        run Workflow();
        If (deadlock == 0)
                exit with success
        else
                exit with error
        fi
}
```

Finally, the presented workflow has been validated in Promela language. The Promela language provides the modelling of asynchronous, concurrent processes and non-deterministic behaviour. In addition, it allows the modelling of inter-process communication through asynchronous FIFO channels. Each workflow process has been implemented as an asynchronous Promela process that runs concurrently with the rest of the processes. The pseudocode of the Promela model is detailed in Listing 1. The process synchronisation has been modelled by sending a token through the communication channel, so that process $P_i$ receives a token $T$ through channel $In(Ti)$. The numbering of processes is equivalent to steps of Fig. 2. Processes that receive the awaiting token(s) then send the token over the output channel $Out(Ti)$ to the destination process. In the behaviour of process $P_{14}$, a *JOIN* operation is implemented, as the process depends on both tokens coming from $P_9$ and from $P_{13}$.

The evaluation of the validation condition in Promela has been modelled using LTL (Linear Temporal Logic) using the "$\langle\rangle f$" operation, represented in the pseudocode Listing 1 as the deadlock flag that will be set if the execution does not end properly. By parallelising all processes, the Promela code checks if the LTL condition is reached for all possible event sequences guaranteeing that the workflow ends for all cases and it is, therefore, validated. For the evaluation of the model the Promela interpreter called Spin (Simple Promela Interpreter) [45] was used, which, through an exhaustive search, evaluated the condition in the entire space of states and obtained no error. The model has 43 states and the validation process took 51 transitions.

### 3.2. Opportunities to be gained

The described approach offers a number of benefits when considering the end-to-end analysis in a 5G enabled edge computing scenario. By moving away from purely analytical approaches and introducing a simulation environment that can model all relevant actors, we are able to understand how different settings affect the performances. The chosen combination of simulators provides the necessary modules for modelling massive simulations that include a vast number of users and services spread over a wide area.

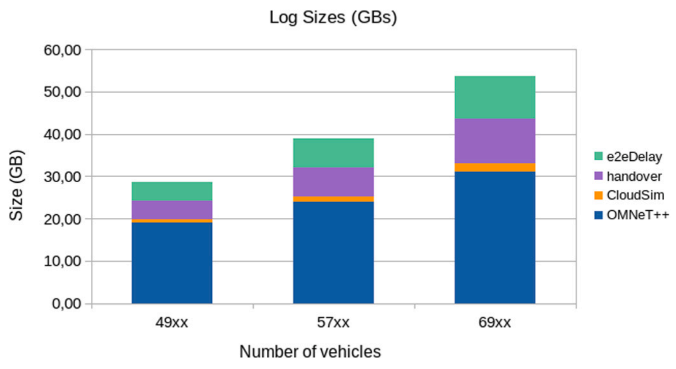


**Fig. 3.** Simulation and post-process log sizes obtained during the workflow.

Typically, the initial research results provided are considering a small rectangular area ranging in several hundred meters with a few base stations and up to a dozen edge servers, with maximum few hundred services. However, we believe that the real performances of proposed algorithms, enhancements and upgrades must be tested in large scale scenarios. Furthermore, when considering 5G, this scenario should be an urban complex scenario that is as close to real life conditions as possible.

Towards these goals, using SUMO to describe the simulation area and the mobility of the end users enables us to create massive edge computing scenarios with thousands of mobile users of various types such as vehicles, bicycles, public transportation, pedestrians and emergency services. Powered by OpenStreetMaps, SUMO enables us to define a real urban setting considering a real city area while incorporating all traffic rules and restrictions for that area such as traffic lights, one way streets, max speed allowed, etc. Going one step further, by adding the 5G wireless modules in OMNeT++, we are able to define the location of the 5G base stations in the city urban area and choose the modelling for the radio signal propagation. Then, in CloudSim we can define the edge servers connected to each base station and the backbone network that connects them together. Thus, as it is presented in the results section, we are able to simulate scenarios with multi-thousands of mobile services that are active in a kilo-meters wide urban area.

An important benefit of this approach is that all components are popular open source solutions which can be easily extended with additional modules that implement specific algorithms aimed at edge management, mobile communication, or mobility patterns. The provided scripts that setup the simulation and analyse the results work with the existing modules present in CloudSim and OMNeT++ , however any researcher can develop custom extensions that can be used in place of the example choices presented in this paper. Depending on the complexity of the extensions, adjustments of the provided processing notebooks may be required together with a series of verifications of the implementation. On the other hand, the provided workflow and scripts can be used as a blueprint to setup a similar simulation environment where one or multiple components may be changed. This option will require more extensive changes in the provided scripts so that their input/output is adjusted to the new component.

The whole scenario definition includes a multitude of parameters that can be used to tune the scenario to focus the analysis on different aspects. In addition to the definition of the exact urban area and the density of mobile end-nodes, there are many parameters that describe the 5G wireless communication, such as propagation models and antenna types and gain, etc, and there are additional parameters that define the orchestration behaviour of the edge servers network such as the host and VM types and different policies for placement and migration. By performing different series of simulations with varying parameters one can analyse the performances under different circumstances and measure the impact of each factor on the whole end-to-end communication pattern. In addition, the proposed workflow can be seen as a foundation for the possibility to make other types of end-to-end analysis in addition to the provided end-to-end delay. The outputs provided from CloudSim and OMNeT++ can be combined and analysed together to perform an energy efficiency analysis for an example. For these purposes the post-processing scripts need to be extended with the analysis of choice.

### *3.3. Performances and challenges*

However, the benefits that come from designing a complex holistic scenario bring additional problems that need to be addressed to be able to efficiently perform the simulations and the results analysis and obtain conclusions in a timely manner. One of the biggest disadvantages of this approach is two fold:

1. By creating a large complex simulation scenario the produced log files from both simulators become a big data problem with log output size from OMNeT++ being over 20 GB per scenario, plus additional few GB of data from CloudSim that needs to be properly aligned with the OMNeT++ data. In Fig. 3 the increase in log size as a function of the scenario size is presented where the number of vehicles (and corresponding services MEC) increases from 4900 to 6900. While OMNeT++ output log is consistently the largest one, it must be noted that in addition to the output logs from the simulators, the intermediary processed logs obtained with the pre and post processing scripts that are used in the workflow are also significant in size (around 25% of the OMNeT++ logs size). It is also noticeable that all logs are increasing linearly with a similar rate as the number of nodes and services increases.

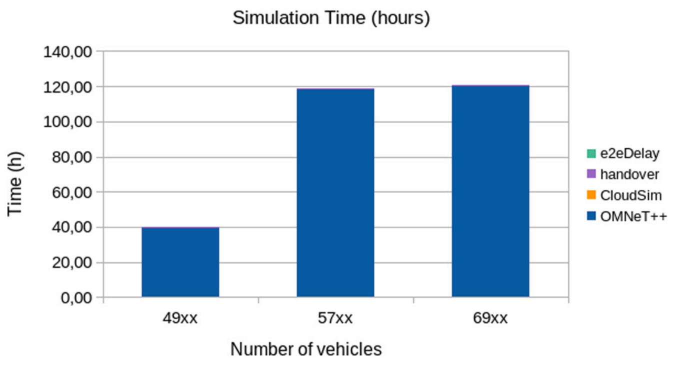


**Fig. 4.** Simulation and post-process duration.

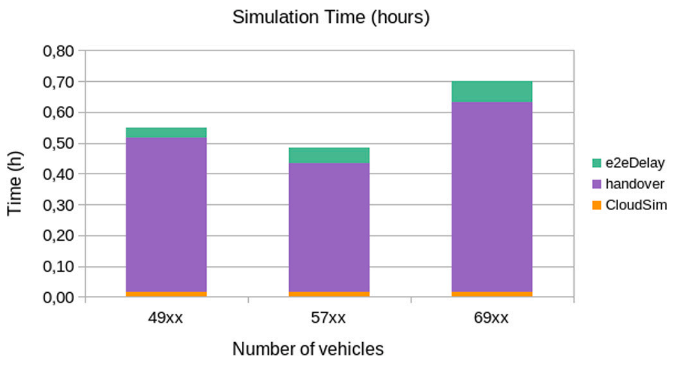


**Fig. 5.** Simulation and post-process duration excluding OMNeT++ processing time.

2. The simulation processing time is extremely long in the case of OMNeT++ that sometimes needs more than 5 days to complete the simulation of only one scenario. While the OMNeT++ output logs file size increases with the increased number of services, the biggest factor that influences the output log file size is the simulation duration time — how many seconds are simulated within the scenario. During our simulation tests we concluded that simulation time longer than 10 000 s yields to situations where OMNeT++ may need up to a week to complete high number of services simulations which becomes difficult and expensive to manage. Finally, the big data logs require a corresponding disk size space so that they can be stored and then post-processed. For instance, in order to be able to gather all outputs and create the post-processing datasets presented in the results section we required over 0.5 TB of storage available.

The increase in simulation time with the increase of nodes in the simulation scenario is presented in Fig. 4. It is evident that the main step that defines the total time needed to implement the proposed workflow is OMNeT++, while the rest of the workflow steps require negligible time in comparison. For smaller scale scenarios of around 4900 nodes, the time required is less than two days, but the time required seems to exponentially rise with the scenario size reaching 120 h (on average) in the case of 6900 nodes. It must be noted however, that the simulation time duration is very volatile when it comes to OMNeT++ and heavily depends on the mobility patterns and the events in the wireless channel such as handovers, loss of connection due to fading signal, number of retransmissions, etc. Thus, even for a static scenario size the difference in simulation time for OMNeT++ can be measured in hours.

Using Fig. 5 one can analyse the performances of the other components of the workflow regarding simulation time duration. It is evident that CloudSim is extremely scalable and fast showing almost no dependency on the increasing simulation size. The post-processing end-to-end delay scripts are also very efficient and handle increasing loads well. As expected, the handover processing which is mainly dependent on the OMNeT++ logs requires more processing time but still manages to analyse the logs in less than an hour coping well with the increasing simulation size.

An additional consequence of the big data logs problem is that the log files post-processing scripts need to be carefully written, using optimised libraries that are able to handle the amount of data that needs to be processed. The first major hurdle that needs to be surpassed in this case is being able to read the massive log files in the cases when their size exceeds the size of the available RAM memory. For these purposes, we used a special vaex function that reads the input in chunks and transforms it into a Hierarchical Data Format version 5 (HDF5) file format that supports large, complex, heterogeneous data. Once the data is ingested, the processing of data must be done using highly optimised functions from specialised high performance libraries that work with big datasets. If care is not taken to ensure data processing optimisation, then the resulting post processing analysis may take hours or even days to produce the required results.

**Table 2**
Simulation scenario parameters.

| Parameters | Considered values |
|---|---|
| **SUMO** | |
| Number of seeds | 3 |
| Car ranges | 49xx cars : [4900,4999] |
| | 57xx cars : [5700,5799] |
| | 69xx cars : [6900,6999] |
| | 85xx cars : [8500,8599] |
| **CloudSim** | |
| Virtual resource usage | Space sharing |
| MEC network | 3 layer 1 Gb fat tree |
| MEC network switches | 9 edge, 3 access, 2 core |
| MEC network latencies | 1.57 ms, 2.45 ms, 2.85 ms |
| Number of server hosts | 45, 63, 81, 99, and 113 hosts |
| Server host type | 2-6 cores, 12 GB RAM, 100 Mbps |
| Initial placement policy | Hierarchical communities |
| Migration policy | Follow-me |
| VM characteristics | 2 cores, 2 GB RAM, 100 Mbps |
| Cloudlet complexity | 100% of CPU time used |
| **OMNeT++** | |
| Type of 5G antenna | UMi_A |
| Geographical area size | $1800 \cdot 2000$ m$^2$ |
| Number of antennas | 9 |
| Antennas connection network | Mesh topology with 10 GB p2p links |

To adopt this approach to end-to-end analysis in a combined 5G edge simulation one must have a good understanding of the possibilities of all simulation tools and libraries that are available in OMNeT++ and CloudSim. This effort initially requires a steep learning curve because both simulators have many options and settings that need to be taken into account. Thus, to exploit the opportunities and benefits of the approach one cannot simply approach the problem with a plug and play attitude. In addition, as the complexity and size of the scenarios grow, we found bugs and limitations in both simulators. For an example, the CloudSim internal events scheduler does not support concurrent events, so one must make sure that these do not happen while defining the simulations scenario paying great attention to the timestamp of service creation, timestamps of service migration events and timestamp of service termination.

Related to the challenge of learning how to use the simulators is the problem of extending the simulators with the proposal in question such as a new resource placement or resource migration algorithm. If the focus is on the edge computing part, and OMNeT++ is used only to be able to analyse how edge computing will perform in the 5G wireless communication environment, then all extensions for new proposals and solutions will need to be done in CloudSim. This will require coding new policies in the simulator that will then be used in the scenario definition. For these purposes one can use the pointers provided in the main CloudSim documentation, or they can use the custom Java classes developed by the authors for the purposes of analysing the community based placement and migration strategies described in [10] which are provided in the GitHub repository as a starting point.

As previously discussed we believe that the benefits outweigh the challenges when it comes to the problem of creating a holistic simulation approach to 5G enables edge computing environments.

## 4. Example simulation scenario

The defined workflow has been applied to create an end-to-end simulation environment of a dense urban scenario that is geographically based in the city centre of Alicante in the South East of Spain. Aiming to create a realistic example closely related to intelligent transportation, we strived to align the chosen simulation parameters as much as possible to the examples and use cases provided in the white paper produced by the 5G Automotive Association (5GAA) [46]. In this document various user stories are provided with use cases ranging from software updates, to automated parking and speed harmonisation. The service level requirements that are defined for the examples can be described as: 350 m vehicle range, 50 km/h as max velocity in urban areas, average density of 1500 vehicles/km$^2$. Take note that the max latency tolerated is around 120 ms. However there are use cases that require latencies of less than 10 ms. The following simulation description and chosen parameter values are based on the service level requirements as provided by the 5GAA. Detailed information about the parameter setting used in the different simulators can be found in Table 2.

Following the assumption of max 350 m range, we have strategically placed nine 5G base stations across the simulated city area to provide the access infrastructure to end users mobile applications. An Urban Micro (UMi) outdoor scenario is configured in OMNeT++ where the antennas are connected to each other using a mesh topology with 10 GB point to point links following the urban scenario setup as described in [9]. Vehicle's routes generated by SUMO traffic simulator act as the main workload for OMNeT++, that simulates the access network operation between the user equipments and the 5G antennas during 3600 s in the described scenario. The recording of statistics has been customised in order to obtain the radio link control (RLC) delays per vehicle along the simulation time. These delays need to be post-processed in order to get individualised access delay values per vehicle and per simulation time slot, so they can be linked to the edge and cloud delays per virtualised entity obtained by CloudSim simulations, and therefore we can obtain end-to-end delays per vehicle across the simulation time.

Within CloudSim we implemented a fat-tree network topology for the edge infrastructure where the root of the tree located in the 5G provider mobile exchange centre connects to the cloud used whenever there are no resources available at the edge where the tree leaves are located and the interconnection between the base stations and the MEC hosts is achieved. Note that CloudSim supports the definition of any type of network topology that interconnects the computing hosts with an arbitrary number of routers/switches. The fat-tree network topology for the MEC hosts has been chosen as it is typical for datacentres [47] related to cloud, fog and edge deployments. Since the hosts in this logical topology are located in the leaves of the tree and the network topology exhibits natural grouping on the level of edge and aggregation switches, this topology fits very well with the ETSI MEC concepts of hierarchical neighbourhoods [42] by grouping branches of the tree. This enables a one to one mapping of the base station service area to the co-located fat-tree leaf, enabling efficient implementations of geo-tracking. The tree leaves are the edge datacentres associated to each 5G base station in the coverage area. Therefore, we have considered nine homogeneous datacentres connected to the infrastructure network, that are composed of 5, 7, 9, 11, or 13 hosts each, resulting in 45, 63, 81, 99 or 117 MEC hosts in total for the whole MEC system. The number of hosts has been chosen so that the performances of the system can be evaluated under varying capacity levels relative to the number of services requested. Each datacentre is locally attached to one of the 5G base stations installed across the city centre. Two types of MEC hosts have been analysed, on one hand, type 1 MEC host that includes 4 CPU cores, 8 GB of RAM and, on the other hand, type 2 that is configured with 6 CPU cores and 12 GB of RAM. Both host types have a network connection of 1 GB. The host types and profile configurations are specified in the CloudSim configuration library and their use ensures that the power consumption information provided by CloudSim is accurate.

SUMO's vehicular traffic density is configured in four groups (49xx, 57xx, 69xx, 85xx) ranging from light vehicular density (49xx for less than 1500 vehicles/km$^2$) to a high vehicular density (85xx corresponding to around 2400 vehicles/km$^2$) scenario, which brings in traffic jams without causing severe congestions. One MEC service is demanded per vehicle to the access network that establishes a mobile connection (OMNeT++) to the edge and cloud infrastructure (CloudSim). The edge computing orchestrator assigns a virtualised computing facility at the closest MEC datacentre as a response to the user request (CloudSim). If there are no resources available at the edge, the edge service is placed in the Cloud. Fig. 6 shows the performance of the optimisation mechanism for allocating the edge services of the vehicles that enter the coverage area. The number of hops that the service data has to take through the infrastructure network from the user equipment till the allocated virtualised resources in the MEC or cloud hosts are considered in order to evaluate the performance of the allocation mechanism, in case the allocation is not optimal. This is the reason to show in Fig. 6 the number of optimal and not optimal allocation results of four MEC datacentre simulation configurations considering the most dense scenario of 85xx vehicles.

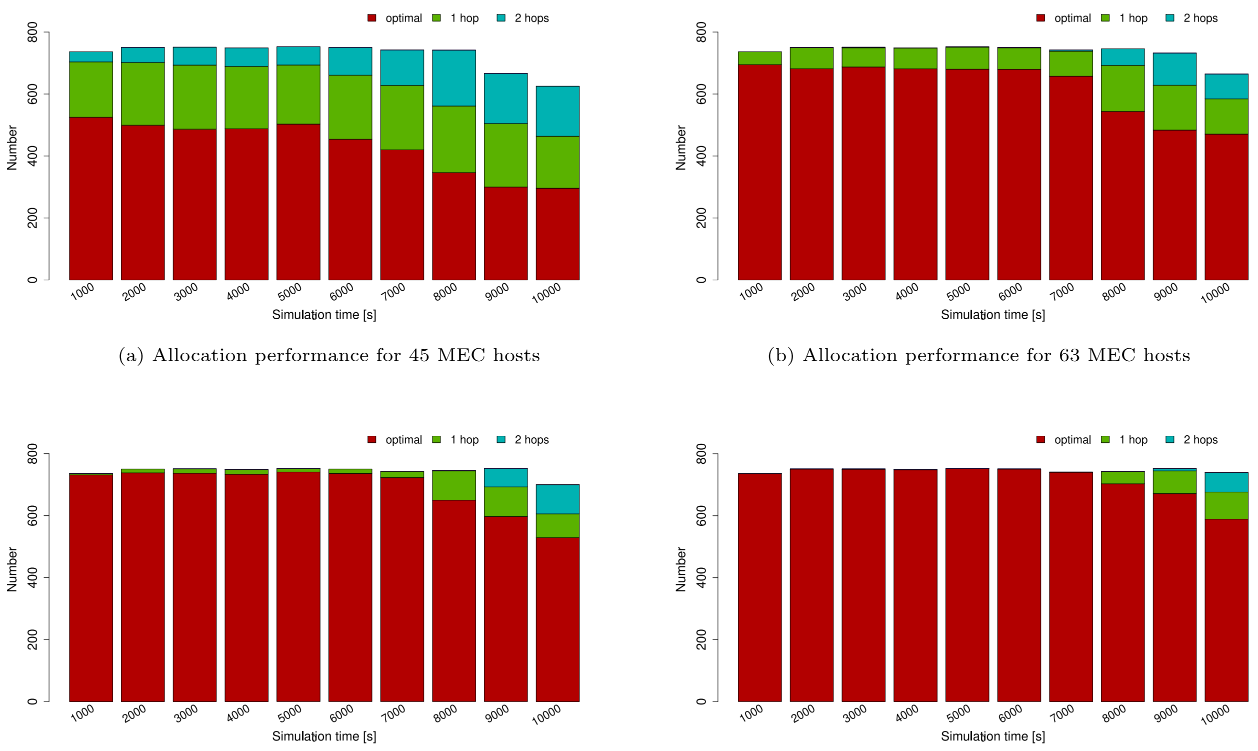
As the vehicle moves around the coverage area, hand-off operations occur when the connection switches to another base station (OMNeT++ ). This intermediately triggers a migration request at the edge infrastructure (CloudSim) in order to achieve minimal service latency. However, we have taken into account that these migration operations not always are successful as there might be no enough computing resources to migrate the virtualised entity at the destination datacentre. The possible results of a migration operation request are the following:

- migration successful (`opt_migr`): the service is successfully migrated to the closest MEC datacentre.
- migration unsuccessful (`failed_migr`): in this case there are no available resources in the closest MEC datacentre, therefore the virtualised instance keeps running at the previously allocated position of the edge infrastructure. However, when this happens, the orchestrator is notified to reschedule the migration attempt after a given migration retry timeout. This timeout is set by default to 0.5s in order to ensure agility, but it can be easily changed to a different value during simulation setup.
- migration reattempt successful (`opt_reatt`): when the timeout expires, a migration reattempt is performed that ends up successfully.
- migration reattempt unsuccessful (`failed_reatt`): in this case, the migration is still not possible due to a lack of idle resources in the destination MEC datacentre and more migration reattempts will follow.

Let us analyse the obtained delay results by observing the performance values gathered at the access and edge network separately, and then continue with a comparison of the end-to-end delay results for some chosen scenarios. The developed Jupyter Notebooks used to plot all delay figures shown in this paper can be found in [11]. Note that three simulation runs have been performed for each presented simulation scenario. The presented results are averaged across the corresponding three runs.

### 4.1. Access and edge network delay results

From OMNeT++ simulations output the vectors can be obtained from where we compute the average delay values of the access network related to the different traffic densities introduced in the traffic simulator. These are shown in Table 3 together with their 95% confidence intervals. It can be observed that the average values are considerably high for the expected service response time (compared to intelligent transport requirements of less than 10 ms [48]), and this is due to the high variability in the access connection delays including nodes in dark and grey coverage areas. Dark coverage areas are areas where the mobile network signal strength is below the minimum required threshold needed to establish communication and may appear due to various phenomena related to wireless signals propagation. Similarly, grey coverage areas are the areas where the wireless signal strength is fluctuating



(a) Allocation performance for 45 MEC hosts
(b) Allocation performance for 63 MEC hosts
(c) Allocation performance for 81 MEC hosts
(d) Allocation performance for 117 MEC hosts

**Fig. 6.** Performance results of the optimisation mechanism (for 85xx vehicles).

**Table 3**
Average and 95% confident interval values of access network delays per car range (in ms).

| | 49xx | 57xx | 69xx | 85xx |
|---|---|---|---|---|
| 95% CI low | 12.69 | 6.22 | 19.20 | 112.10 |
| Average | 29.56 | 23.06 | 36.08 | 230.84 |
| 95% CI high | 46.43 | 39.89 | 52.97 | 349.58 |

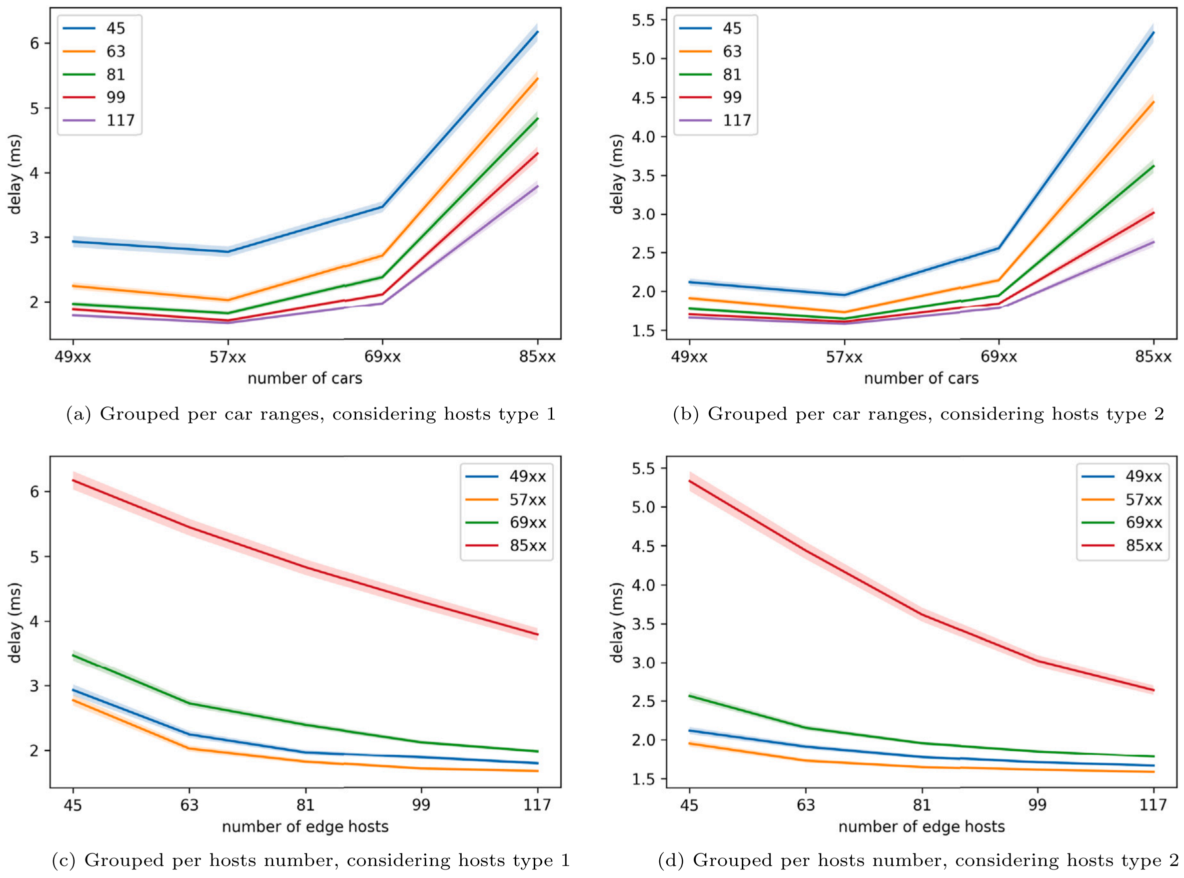
**Table 4**
Median, 25% and 75% percentile values of access network delays per car range (in ms).

| | 49xx | 57xx | 69xx | 85xx |
|---|---|---|---|---|
| 25% perc | 1.97 | 1.97 | 1.97 | 1.96 |
| Median | 2.38 | 2.40 | 2.40 | 2.39 |
| 75% perc | 2.83 | 2.84 | 2.87 | 2.91 |

and thus, the communication is unstable. This point gets clearer if we observe the median values of access network delays that are shown in Table 4, that also include the 25% and 75% percentile values. Therefore, there is work to be done in order to analyse coverage holes in our mobile networks scenario.

Considering now the edge network performance, aggregated results of the three simulation seeds that we have introduced as car ranges in order to determine the workload density are shown in Fig. 7, in the form of average delay values at the edge network and their 95% confidence intervals. Observing Fig. 7(a) and (b) we can analyse that while the delay of the least dense configurations is maintained low, when introducing higher workloads all delays obtained increase accordingly for all sizes of edge datacentres (45 to 117 hosts of type 1 and 2, respectively). When plotting the mean delay against the number of hosts per edge datacentre as it is shown in Fig. 7(c) and (d), we observe that the delay is inversely proportional to the number of available edge hosts. We have also computed median and percentile values for the edge network delay that is shown in Fig. 8. The variability of the generated workload impacts percentiles causing the 75% reach almost 9 ms for the smallest MEC datacentre configuration when the maximum workload is considered (see Fig. 8(a)). In addition, the large type 2 MEC hosts help keep the edge network delay lower as the load increases, with a maximum of 7 ms compared to the 9 ms for host type 1. This effect diminishes for smaller loads.

Therefore, it is worth to analyse the end-to-end behaviour of the tested environment over the simulation time to better understand the delay evolution.



(a) Grouped per car ranges, considering hosts type 1 (b) Grouped per car ranges, considering hosts type 2

(c) Grouped per hosts number, considering hosts type 1 (d) Grouped per hosts number, considering hosts type 2

**Fig. 7.** Average edge network delays and 95% confidence intervals varying MEC host types.

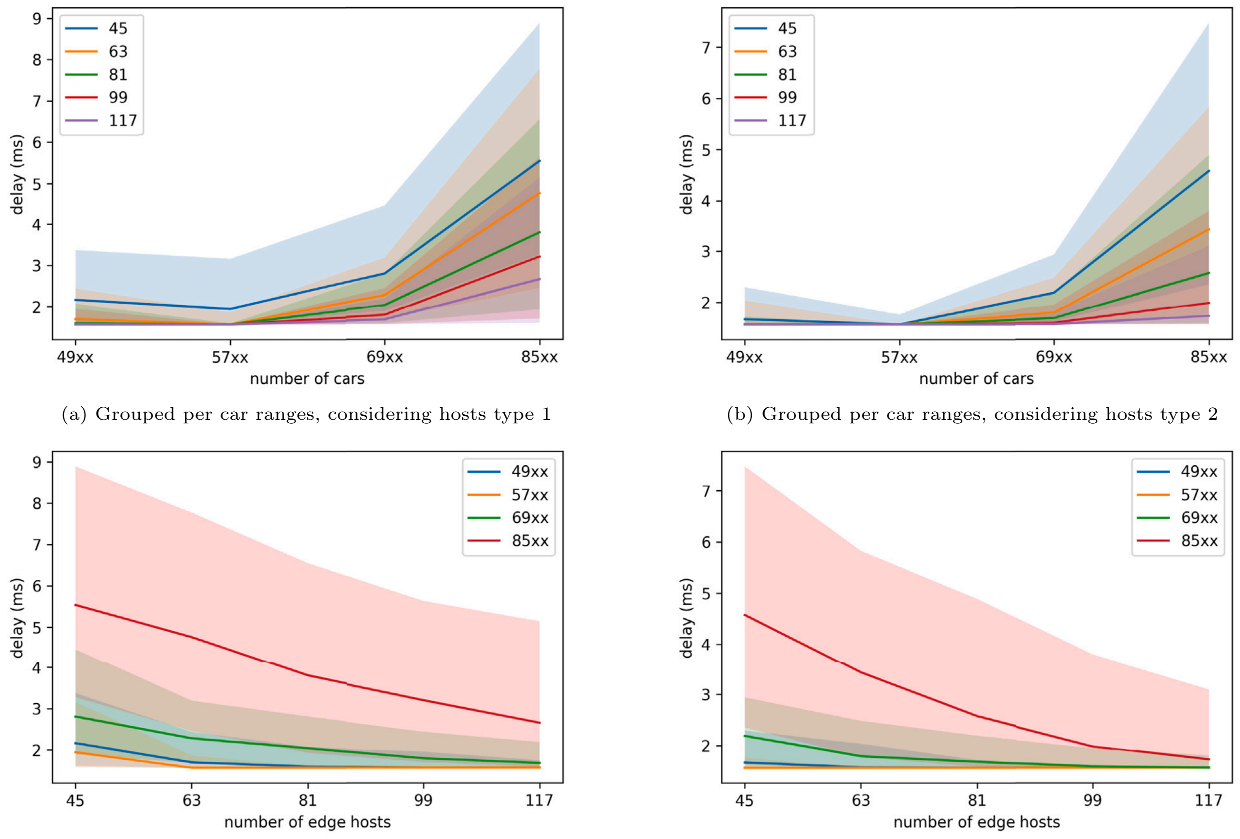
## 4.2. End-to-end delay results

The amount of computing facilities allocated in the pool of available resources at the edge datacentres and at the cloud network infrastructure, and their features, is directly related to the performance of provided edge services. In this section we have chosen to show the end-to-end delays behaviour along the simulation time considering several selected scenarios that are depicted in Fig. 9. The selected scenarios are the ones with the lowest and highest workload that is related to the vehicles density. Boxplot figures have been drawn without outliers in order to obtain a clearer image to better comprehend the evolution of the delay. Simulation time has been slotted and end-to-end delays have been aggregated in 100 s slots, obtaining a total of 36 slots in the X axis.

On one hand, considering that the number of hosts per datacentre is 5 each, that makes a total of 45 hosts at the edge layer, it is interesting to point out that the type of host does affect the delay considerably along the simulation time with 49xx vehicles (and more than 1800 VMs running at the edge). Figs. 9(a) and (b) show this difference. They also show that the maximum delay obtained when considering host type 2 is maintained under 8 ms, while host type 1 values increase to more than 12 ms by the end of the simulation. If we focus on the delay obtained in a dense scenario with 85xx vehicles – more than 3100 VMs demanded – that is shown in Figs. 9(c) and (d), we can appreciate that the delay median reaches a 5 milliseconds time delay much earlier (by slot 10) when considering host type 1 compared to the host type 2 figure. The same behaviour is observed at maximum values. So it is obvious that in the first 4 scenarios shown it the figure, the edge computing infrastructure is congested even with the lighter density vehicular traffic configuration.

On the other hand, we have chosen to show the same traffic configuration but increasing the number of hosts per edge datacentre to 13, that makes a total of 117 hosts at the edge layer. Figs. 9(e) and (f) show an uncongested situation where there are enough resources at the edge hosts to run the workload created, meanwhile figures (g) and (h) do show congestion from slot 15 in the case of host type 1 reaching a median value higher than 4 ms. Considering the case of host type 2, it maintains the median value below 4 ms but congestion is detected from slot 25. This again shows that the host type affects in obtaining earlier increasing end-to-end delays.

Finally, we have chosen to include some end-to-end figures that show median and percentile values for all configurations in Fig. 10. These four images are the summary of the obtained end-to-end delays for the example simulation scenario. Apart from visualising overlapping of percentiles, higher values can be distinguished and we appreciate that the averaged 75% percentile values



(a) Grouped per car ranges, considering hosts type 1 (b) Grouped per car ranges, considering hosts type 2

(c) Grouped per hosts number, considering hosts type 1 (d) Grouped per hosts number, considering hosts type 2

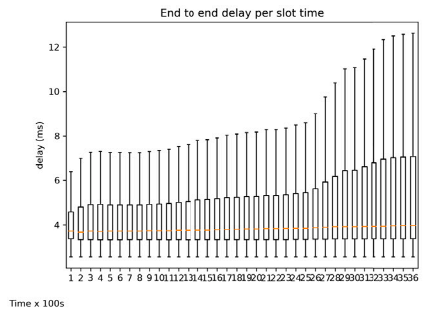
**Fig. 8.** Median edge network delays, 25% and 75% percentile values varying MEC host types.

of 85xx simulations in all plots of Fig. 10 are lower that 13 ms for the smallest MEC datacentres configuration. This confirms that, even though the end-to-end value obtained goes beyond 25 ms as we observed at boxplot figures, on average it does not exceed 13 ms. When we take into consideration the results about the 5G access network delays presented in Tables 3 and 4, where it can be seen that although the overall median is not affected, the average delay values are substantially increased with the simulation load, it must be noted that the high end-to-end delay values are very much influenced not only by the load in the edge servers but also by the load in the 5G network.

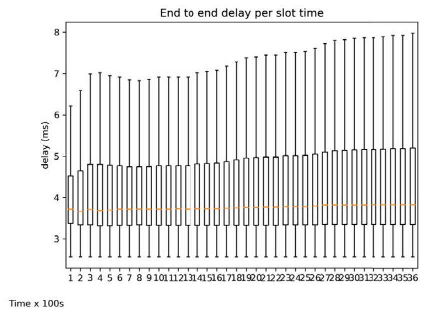
## 5. Conclusion

The promise of low delays for processing near real time data on the edge of the network is the key performance factor for edge computing that has led to a plethora of use cases and applications. By coupling the edge computing virtualised servers with the 5G networking infrastructure a complex symbiotic system is created where the enhanced performances of 5G networks such as low latency, high bandwidth and low energy consumption are augmented with the ability of devices to transfer the burden of processing to the edge of the network. However, the scale of deployment of edge computing servers is far smaller compared to cloud datacentres and thus the available computing resources must be carefully managed by employing optimised resource management techniques. There are many resource management approaches proposed in the research literature and their performances need to be carefully analysed by creating large scale urban complex simulation scenarios that will test the end-to-end performance parameters under different loads. The work presented in this paper aims to help researchers with this problem.

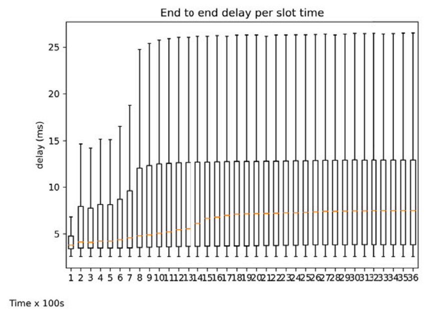
The main contributions of this paper are as follows: a detailed optimised workflow with the accompanying open source code are presented that enables researchers to create 5G-MEC large scale urban mobile simulation scenarios using SUMO, OMNeT++ and CloudSim; the proposed holistic simulation framework can be easily extended with custom modules for the access network in OMNeT++ and/or custom resource management algorithms in CloudSim, as well as easily extended for other types of analysis in addition to end-to-end delay, such as power efficiency for an example; the complexity of large scale 5G-MEC urban simulations is analysed in both time and space domain; the potential and flexibility of the proposed workflow are presented via the end-to-end delay analysis of an example simulation scenario.



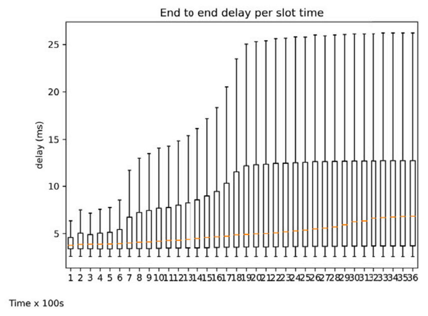
(a) 49xx vehicles density 45 hosts type 1 (18xx VMs)



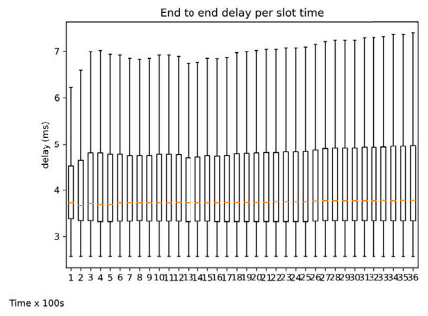
(b) 49xx vehicles density 45 hosts type 2 (18xx VMs)



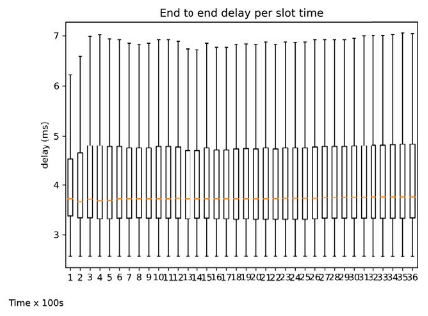
(c) 86xx vehicles density 45 hosts type 1 (31xx VMs)



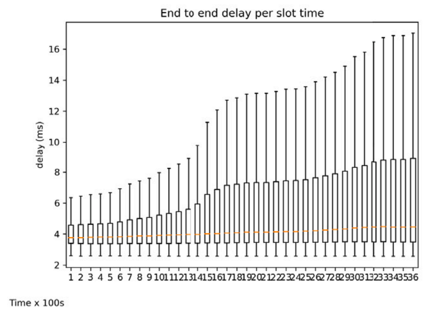
(d) 86xx vehicles density 45 hosts type 2 (31xx VMs)



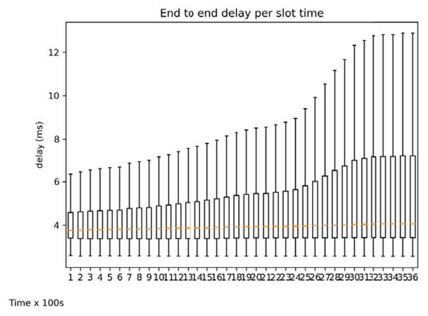
(e) 49xx vehicles density 117 hosts type 1 (18xx VMs)



(f) 49xx vehicles density 117 hosts type 2 (18xx VMs)



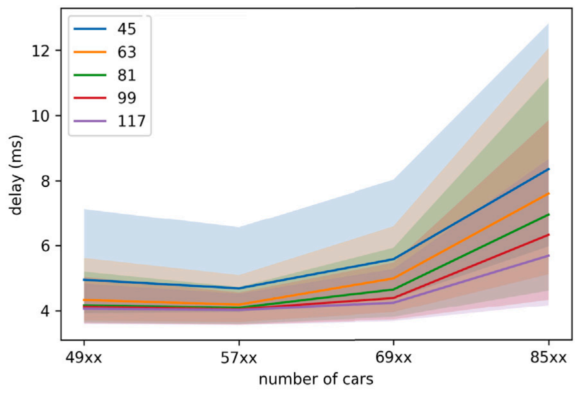
(g) 86xx vehicles density 117 hosts type 1 (31xx VMs)



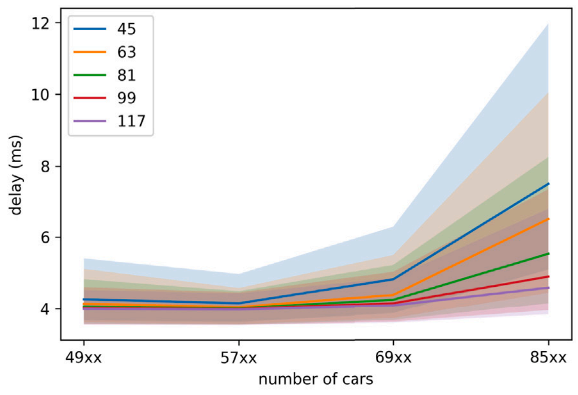
(h) 86xx vehicles density 117 hosts type 2 (31xx VMs)

**Fig. 9.** End-to-end delay boxplot results during 3600 s simulation time.

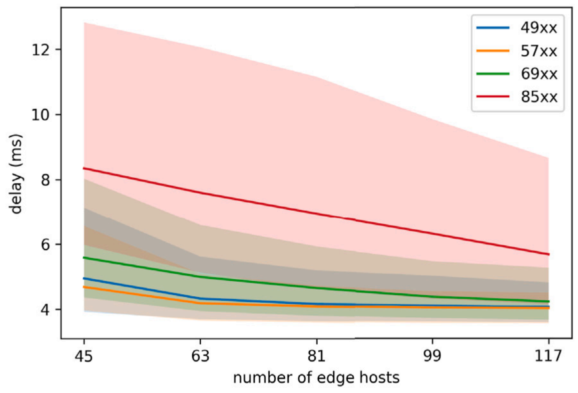
We have discussed the challenges and benefits of creating a simulation environment that combines 5G networking with edge computing infrastructure by linking OMNeT++ and its 5G/LTE extension with CloudSim. Using a series of pre and post processing Jupyter Notebooks the output of the 5G network communication simulated in OMNeT++ is used to define the input for the edge computing infrastructure defined in CloudSim. Both outputs are afterwards combined and analysed together with the aim to make an end-to-end delay analysis that will provide a holistic view of the performances of the system as a whole thus showing how the



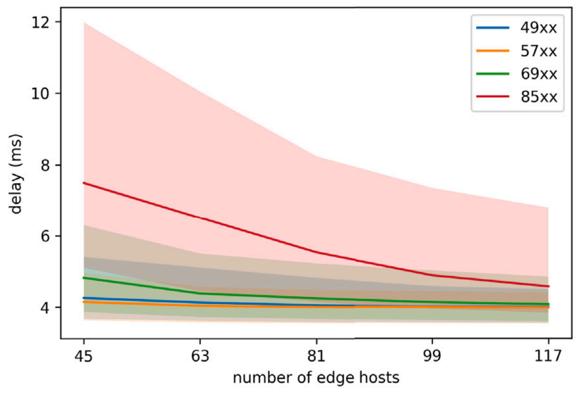
(a) Grouped per car ranges considering hosts type 1



(b) Grouped per car ranges considering hosts type 2



(c) Grouped per hosts number considering hosts type 1



(d) Grouped per hosts number considering hosts type 2

**Fig. 10.** Median end-to-end network delays, 25% and 75% percentile values varying MEC host types.

load in the edge is reflecting in the changes of the delay in the access network communications. The complete workflow and custom simulator code have been made available as open source thus facilitating reproducibility and usage for other simulation scenarios.

Although using complex simulators leads to big data challenges when it comes to processing the output logs, the level of detail provided by the simulation framework outpasses other available simulation approaches leading to a diverse set of possible analysis of the effects of changes in both the 5G and the edge computing modelling. The integration of SUMO provides means to define large scale urban mobile scenarios with different actors that are very close to real life vehicular and pedestrian mobility patterns. The example simulation results show that the load of increased number of edge services in the system takes its toll on both the edge computing and the 5G network resources and careful optimisation is needed to provide optimal performances for the end user and avoid the high variation in end-to-end delay that is experienced during high loads.

## Data availability

Data will be made available on request.

## References

[1] S. Abolfazli, Z. Sanaei, A. Gani, F. Xia, L. Yang, Rich mobile applications: genesis, taxonomy, and open issues, J. Netw. Comput. Appl. 40 (2014) 345–362.

[2] X. Ma, Y. Zhao, L. Zhang, H. Wang, L. Peng, When mobile terminals meet the cloud: computation offloading as the bridge, IEEE Netw. 27 (5) (2013) 28–33.

[3] M.M. Satyanarayanan, The emergence of edge computing, Computer 50 (1) (2017) 30–39.

[4] M. Shafi, A. Molisch, P. Smith, T. Haustein, P. Zhu, P. Silva, F. Tufvesson, A. Benjebbour, G. Wunder, 5G: A tutorial overview of standards, trials, challenges, deployment, and practice, IEEE J. Sel. Areas Commun. 35 (6) (2017) 1201–1221.

[5] Y. Xiao, G. Shi, Y. Li, W. Saad, H.V. Poor, Toward self-learning edge intelligence in 6G, IEEE Commun. Mag. 58 (12) (2020) 825 34–40.

[6] H. Chien, Y. Lin, C. Lai, C. Wang, End-to-end slicing as a service with computing and communication resource allocation for multi-tenant 5G systems, IEEE Wirel. Commun. 26 (5) (2019) 104–112.

[7] S. Svorobej, P. Endo, M. Bendechache, C. Filelis-Papadopoulos, K. Giannoutakis, G. Gravvanis, D. Tzovaras, J. Byrne, T. Lynn, Simulating fog and edge computing scenarios: An overview and research challenges, Future Internet 11 (3) (2019) 55.

[8] P. Lopez Alvarez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E. Wießner, Microscopic traffic simulation using SUMO, in: IEEE Intelligent Transportation Systems Conference (ITSC), 2018.
[9] T. Deinlein, R. German, A. Djanatliev, 5G-sim-V2I/N: towards a simulation framework for the evaluation of 5G V2i/V2N use cases, in: 2020 European Conference on Networks and Communications (EuCNC), Dubrovnik, Croatia, 2020, pp. 353–357.
[10] S. Filiposka, A. Mishev, K. Gilly, Mobile-aware dynamic resource management for edge computing, Trans. Emerg. Telecommun. Technol. 30 (6) (2019) 3626.
[11] Edge simulation GitHub repository. Mobile edge computing simulation in 5G environment, 2022, (accessed the 1st of February, 2022). URL https://github.com/EdgeSimulation.
[12] G. Faraci, C. Grasso, G. Schembra, Design of a 5G network slice extension with MEC UAVs managed with reinforcement learning, IEEE J. Sel. Areas Commun. 38 (10) (2020) 2356–2371.
[13] T. Subramanya, D. Harutyunyan, R. Riggio, Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks, Comput. Netw. 166 (2020) 106980.
[14] J. Li, H. Gao, T. Lv, Y. Lu, Deep reinforcement learning based computation offloading and resource allocation for MEC, in: 2018 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2018, pp. 1–6.
[15] Q. Ye, W. Zhuang, X. Li, J. Rao, End-to-end delay modeling for embedded VNF chains in 5G core networks, IEEE Internet Things J. 6 (1) (2018) 692–704.
[16] H. Haile, K. Grinnemo, S. Ferlin, P. Hurtig, A. Brunstrom, End-to-end congestion control approaches for high throughput and low delay in 4G/5G cellular networks, Comput. Netw. 186 (2021) 107692.
[17] A. Mukherjee, Energy efficiency and delay in 5G ultra-reliable low-latency communications system architectures, IEEE Netw. 32 (2) (2018) 55–61.
[18] P. Thiruvasagam, V. Kotagi, C. Murthy, The more the merrier: enhancing reliability of 5G communication services with guaranteed delay, IEEE Netw. Lett. 1 (2) (2019) 52–55.
[19] P. Thiruvasagam, V. Kotagi, S. Murthy, A reliability-aware, delay guaranteed, and resource efficient placement of service function chains in softwarized 5G networks, IEEE Trans. Cloud Comput. (2020).
[20] C. Bouras, A. Gkamas, G. Diles, Z. Andreas, A comparative study of 4G and 5G network simulators, Int. J. Adv. Netw. Serv. 13 (1) (2020).
[21] G. Nardini, D. Sabella, G. Stea, P. Thakkar, A. Virdis, Simu5G: An OMNeT++ library for end-to-end performance evaluation of 5G networks, IEEE Access 8 (2020) 181176–181191.
[22] S. Guo, X. Hu, G. Dong, W. Li, X. Qiu, Mobile edge computing resource allocation: A joint Stackelberg game and matching strategy, Int. J. Distrib. Sens. Netw. 15 (7) (2019) 1550147719861556.
[23] E. Badidi, Qos-aware placement of tasks on a fog cluster in an edge computing environment, J. Ubiquitous Syst. Pervasive Netw. 13 (1) (2020) 11–19.
[24] G. Wang, F.F. Xu, Regional intelligent resource allocation in mobile edge computing based vehicular network, IEEE Access 8 (2020) 7173–7182.
[25] P. Roy, S. Sarker, M. Razzaque, M. Hassan, S. AlQahtani, G. Aloi, G. Fortino, AI-enabled mobile multimedia service instance placement scheme in mobile edge computing, Comput. Netw. 182 (2020) 107573.
[26] T. Le, N. Ioini, C. Pahl, H. Barzegar, Edge computing simulation platforms: A technology survey, in: Z. C. (Ed.), Advances in Service-Oriented and Cloud Computing. ESOCC 2020. Communications in Computer and Information Science, Vol. 1360, Springer, Cham, 2021.
[27] Q. Sun, L. Tian, J. Shi, Y. Zhou, L. Liu, Y. Wang, F. Wang, Joint management of communicating and computing resources in sliced 5G networks, in: GLOBECOM 2020-2020 IEEE Global Communications Conference, IEEE, 2020, pp. 1–6.
[28] M. Laha, S. Kamble, R. Datta, Edge nodes placement in 5G enabled urban vehicular networks: A centrality-based approach, in: 2020 National Conference on Communications (NCC), IEEE, 2020, pp. 1–6.
[29] R. Calheiros, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw. - Pract. Exp. 41 (1) (2011) 23–50.
[30] A. Varga, R. Hornig, An overview of the omnet++ simulation environment, in: Simutools '08: Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems, 2008, Article No.: 60. Pages 1-10.
[31] T.R. Henderson, S. Roy, S. Floyd, G.F. Riley, Ns-3 project goals, in: Proceedings of the Workshop on Ns-2: The IP Network Simulator (WNS2'06), 2006.
[32] Matlab web page, 2022, (accessed the 1st of February, 2022). URL https://mathworks.com/index.html.
[33] NetSim web page, 2022, (accessed the 1st of February, 2022). URL https://www.tetcos.com/index.html.
[34] Riverbed web page, 2022, (accessed the 1st of February, 2022). URL https://cms-api.riverbed.com/portal/community_home.
[35] D.N. Jha, K. Alwasel, A. Alshoshan, X. Huang, R.K. Naha, S.K. Battula, S. Garg, D. Puthal, P. James, A. Zomaya, S. Dustdar, R. Ranjan, IoTsim-edge: A simulation framework for modeling the behavior of internet of things and edge computing environments, Softw. Pract. Exp. Wiley 50 (2020) 844–867.
[36] C. Sonmez, A. Ozgovde, C. Ersoy, EdgeCloudSim: An environment forperformance evaluation of edge computing systems, Trans. Emerging TelTech. 29 (2018) e3493.
[37] H. Gupta, A. Dastjerdi, S. Ghosh, 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. Exper. 47 (2017) 1275–1296.
[38] J. Son, A.V. Dastjerdi, R.N. Calheiros, X. Ji, Y. Yoon, R. Buyy, CloudSimSDN: Modeling and simulation of software-defined cloud data centers, in: Proceedings of the 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing.
[39] X. Zeng, S. Garg, P. Strazdins, P. Jayaraman, D. Georgakopoulos, R. Ranjan, IOTSim: A simulator for analysing IoT applications, J. Syst. Archit. 72 (2017) 93–107.
[40] K. Alwasel, D.N. Jha, F. Habeeb, U. Demirbaga, O. Rana, T. Baker, S. Dustdar, M. Villari, P. James, E. Solaiman, R. Ranjan, IoTSim-osmosis: A framework for modeling and simulating IoT applications over an edge-cloud continuum, J. Syst. Archit. 116 (2021).
[41] T. Qayyum, A. Malik, M. Khattak, O. Khalid, S. Khan, FogNetSim++: A toolkit for modeling and simulation of distributed fog environment, IEEE Access 6 (2018) 63570–63583 10 1109 2018 2877696.
[42] MEC, ETSIGS, Multi-Access Edge Computing (MEC, Vol. V2, Framework and Reference Architecture, 2020.
[43] K. Gilly, A. Mishev, S. Filiposka, S. Alcaraz, Offloading edge vehicular services in realistic urban environments, IEEE Access 8 (2020) 97511491–97511502.
[44] T. Deinlein, A. Brummer, R. German, A. Djanatliev, On the impact of buildings on the LoS evaluation in system-level V2I/N simulations, in: 2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall, IEEE, 2021, pp. 1–5.
[45] M. Ben-Ari, Principles of the Spin Model Checker, Springer-Verlag London, 2008.
[46] 5G Automotive Association, C-V2X use cases methodology, examples, and service level requirements, White Paper, 2019, p. 38850.
[47] Y. Liu, J. Muppala, M. Veeraraghavan, D. Lin, M. Hamdi, Data center networks: Topologies, architectures and fault-tolerance characteristics, Springer Science and Business Media, 2013.
[48] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. Mitschele-Thiel, M. Muller, T. Elste, M. Windisch, Latency critical IoT applications in 5G: Perspective on the design of radio interface and network architecture, IEEE Commun. Mag. 55 (2) (2017) 70–78.