EdgeCloudSim: An Environment for Performance Evaluation of Edge Computing Systems

Cagatay Sonmez
Department of Computer Engineering, Bogazici University
Istanbul 34342, Turkey
Email: cagatay.sonmez@boun.edu.tr

Atay Ozgovde
Department of Computer Engineering, Galatasaray University
Istanbul 34357, Turkey
Email: aozgovde@gsu.edu.tr

Cem Ersoy
Department of Computer Engineering, Bogazici University
Istanbul 34342, Turkey
Email: ersoy@boun.edu.tr

***Abstract*—Edge Computing is a fast growing field of research covering a spectrum of technologies such as Cloudlets, Fog Computing and Mobile Edge Computing (MEC). Edge Computing involves technically more sophisticated setup when compared with the pure Cloud Computing and pure Mobile Computing cases since both computational and network resources should be considered simultaneously. In that respect, it provides a larger design space with many parameters rendering a variety of novel approaches feasible. Given the complexity, Edge Computing designs deserve scientific scrutiny for sound assessment of their feasibility. However, despite increasing research activity, this field lacks a simulation tool compatible with the requirements. Starting from available simulators a significant programming effort is required to obtain a simulation tool meeting the actual needs. To decrease the barriers, a new simulator tool called EdgeCloudSim streamlined for Edge Computing scenarios is proposed in this work. EdgeCloudSim builds upon CloudSim to address the specific demands of Edge Computing research and support necessary functionality in terms of computation and networking abilities. To demonstrate the capabilities of EdgeCloudSim an experiment setup based on different edge architectures is simulated and the effect of the computational and networking system parameters on the results are depicted.**

I. Introduction

Edge Computing covers a wide range of computing concepts such as Mobile Cloud Computing (MCC) [1], Fog Computing [2] and Cloudlet [3]. The common objective of these computing technologies is getting the service from a nearby server located in the geographical proximity of the client. Generally, this paradigm provides a clear advantage for time sensitive applications or when there is no cloud access. Executing the application or a part of application on the edge provides a significant advantage for the mobile devices as well. First of all, the mobile devices can handle the complex operations while consuming less battery by using the edge server. In addition, Edge Computing provides lower transmission delay than cloud computing because the clients do not encounter the wide area network (WAN) delay in the Edge Computing.

It is invaluable to evaluate the performance of different Edge Computing scenarios by considering both network and computational resources. However, it is difficult to find a simulation environment which handles both types of resources in an efficient and an easy way. One of the most popular cloud computing simulation frameworks is CloudSim which allows the modelling of cloud computing infrastructures and the application services [4]. CloudSim is a simulation tool that is streamlined for the standard cloud computing setup and therefore may not contain the necessary functionality for less mature scenarios. Generally, the researchers extend CloudSim by implementing the lacking parts with respect to the needs of their subject matter. For example, Long et al. [5] added the file striping and the data replica management function to CloudSim, and Jung et al. [6] implemented the MapReduce model on CloudSim. The main drawbacks of CloudSim from the point of view of Edge Computing are (i) lack of wireless local area network (WLAN) and WAN communication model, (ii) lack of mobile nodes and mobility support in general, (iii) lack of realistic virtal machine (VM) utilization model. Other popular network simulators such as NS-3, Omnet++ and Opnet can model network resources in detail, but require significant implementation effort for modeling the computational aspects relevant to Edge Computing. Figure 1 depicts the main aspects of Edge Computing simulation modeling in independent axes.

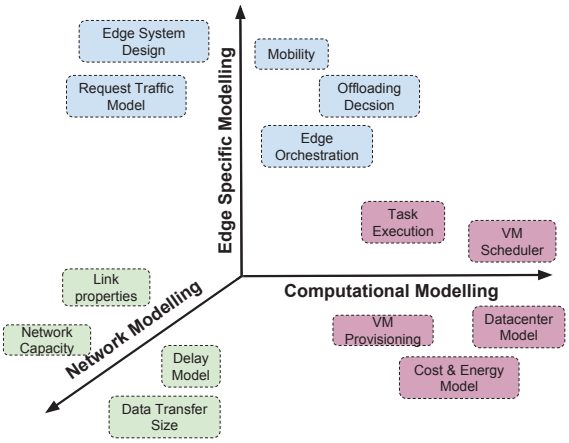


Fig. 1: Aspects of Edge Computing simulation modeling.

In this work, we introduce a new simulator called EdgeCloudSim which covers the whole 3D simulation space in Figure 1. EdgeCloudSim provides a modular architecture to provide support for a variety of crucial functionality such as network modeling specific to WLAN and WAN, device mobility model, realistic and tunable load generator. EdgeCloudSim

978-1-5386-2859-1/17/$31.00 ©2017 IEEE

by design supports simulating multi-tier scenarios where multiple Edge servers are running in coordination with upper layer cloud solutions. Specific to this need, EdgeCloudSim provides an Edge Orchestrator module to enable its users to model the orchestration actions which typically arise in Edge Computing scenarios. For modeling the computational tasks such as VM creation with a given capacity, EdgeCloudSim relies on the capabilities of CloudSim. CloudSim has a long known and reliable code base for the simulation of computational actions. EdgeCloudSim is publicly available as an open source project [7]. Modular design and open-source code base of EdgeCloudSim allow its users to incorporate their own needs in the tool. To demonstrate the capabilities of the EdgeCloudSim, a simulation scenario is designed. Our edge scenario includes mobility and realistic load generation in terms of both packet size and computational load. Different architectures with different internal organizations are examined in the simulations experiments. The results clearly depict the effect of networking and computational parameters on the performance of the Edge Computing system at hand.

The organization of this paper is as follows: In Section II, the background of the Edge Computing and the related works are detailed. In Section III, the design and the architecture of the EdgeCloudSim is explained. In Section IV and Section V, an example use case is summarized and the results of the simulation are evaluated. Section VI concludes our study and provides possible directions for future research.

II. Edge Computing Background

The philosophy of Edge Computing is basically performing computation at the *edge* which refers to a resource in the proximity. The resource in the proximity used in Edge Computing may be a network resource or a computational resource in between the client and the cloud data centers. Shi et al. [8] and Jararweh et al. [9] explain some opportunities the Edge Computing provides. The Edge Computing, Cloudlet Computing, and Fog Computing concepts are in essence similar concepts. In our study, we focus on the term Edge Computing to maintain consistency. Among the various Edge Computing approaches, the cloudlet concept formed in the light of Satyanarayanans definition deserves further attention [3]. The cloudlets are extensively discussed in the literature and they can be considered as the micro-clouds in the proximity. For instance, Tawalbeh et al. propose a multi-tier architecture where the cloudlets are positioned in the middle tier for the big data applications [10]. Similarly, Routaib et al. propose a cloudlet-based architecture where a centralized multi-tiered tree architecture is used [11]. An Edge Computing infrastructure can be composed of many edge servers and deciding where to offload in such a setup is a challenging task. Earlier works considered computational resources only when formulating the offloading problem. However, since local edge computational units can be accessed by network resources the formulation should consider both resource types.

The network resources are shown to have a significant effect on the results for edge scenarios [12]. This fact has been acknowledged by recent work, for instance, Chen et al. optimizes the offloading decision in a multi-tier environment where mobile users offload computing access point and remote cloud [13]. The authors use a heuristic algorithm by considering both the communication and computation resources. Liu et al. highlight the high latency of the WAN and it is concluded that the cloud communication is not sufficient for the resource intensive and the latency sensitive mobile applications [14]. They propose a Markov decision process for the multi-resource allocation in the cloudlet based multi-tier cloud computing environments. The wireless bandwidth and the computing resources are taken into consideration in this study as well. In this context, with its capabilities directed at both computational and networking aspects, we believe that EdgeCloudSim can satisfy the need for realistic simulation of Edge Computing scenarios.

CloudSim is commonly used for the conventional cloud computing scenarios which is mainly designed for evaluating the performance of datacenters [4]. In conventional cloud computing, the clients request from the datacenter to create a list of VMs and run a list of tasks on the related VM. When the datacenter receives such a request, it tries to create the VMs and executes related tasks. The tasks are considered as a bunch of operations which keep the VMs busy for hours. This flow of operation is similar to renting a VM from Amazon Web Services for a certain time. However, Edge Computing scenarios differ from the conventional cloud computing scenarios in the sense that the tasks do not require too much time to be executed. We assume that the mobile clients offload small tasks which can be processed in sub-second intervals. Due to the size of the tasks, creating a new VM for each request is not efficient.

iFogSim is another simulator running on top of CloudSim [15]. It is developed for simulating IoT and Fog environments by modeling components like the sensor, actuator, fog devices and cloud. iFogSim allows the definition of the location of devices taking service from the fog servers. However, this information is static and it is not updated by any mobility model. The network model in iFogSim is over simplistic where a fixed delay is assigned to links ignoring the effect of network load on the operation. As discussed above, for the realistic assessment of Edge Computing proposals, network resources as well as computational resources should be considered with satisfactory accuracy. To the best of our knowledge, an Edge Computing simulator covering aforementioned concerns has not yet been proposed. Therefore, we propose an open to the research community EdgeCloudSim as a sophisticated tool based on CloudSim for the seamless performance evaluation of Edge Computing proposals [7].

III. EdgeCloudSim Architecture

EdgeCloudSim provides a modular architecture where each module addresses a specific aspect of Edge Computing with clearly defined interfaces to other modules. As depicted in Figure 2, in the current EdgeCloudSim version there are

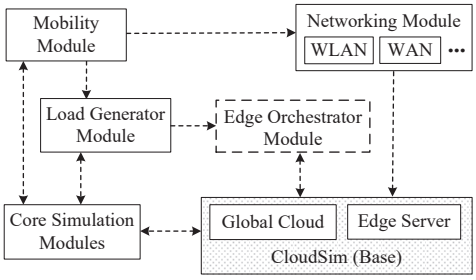


Fig. 2: Relationship between EdgeCloudSim modules.

five main modules available, namely: Core Simulation, Networking, Load Generator, Mobility and Edge Orchestrator. To ease fast prototyping efforts, each module contains a default implementation that can be tuned via simulation parameters. As explained in the previous sections, EdgeCloudSim relies on the capabilities of CloudSim for modeling computational tasks. Since CloudSim uses a discrete event based simulation engine, the time complexity of the EdgeCloudSim depends on the number of te events created in the simulation. When we simulate 75 minutes real life scenario with 100 devices in our simulation, around 100K tasks are generated and the simulation takes average of 10 minutes on a Linux host using Intel core i7-5600U processor.

Core Simulation Module: The core simulation module is responsible for loading and running the Edge Computing scenarios from the configuration files. In addition, it offers a logging mechanism to save the simulation results into the files. The results are saved in comma-separated value (CSV) data format by default, but it can be changed to any format.

Networking Module: The networking module particularly handles the transmission delay in the WLAN and WAN by considering both upload and download data. The mobile clients may use the cellular network or the WLAN to access the cloud services. In CloudSim, it is possible to add link delays between the network components but these delays are static and fixed for all users. However, network link quality may vary from time to time if we consider the mobile devices. In addition, if many mobile devices are using the same Wi-Fi access point, high number of devices would inevitably increase the link delay for either the local area network (LAN) or WLAN communication. In order to model LAN, metropolitan area network (MAN) and WAN communication more accurately, a networking module is provided in EdgeCloudSim. Before sending a task to a VM or downloading the result of the task from the VM, the transmission delay for the upload/download operation is calculated by our network link model. The default implementation of the networking module is based on a single server queue model. Users of EdgeCloudSim can incorporate their own network behavior models into the networking module. Currently, we are working on a large scale WAN delay characterization crowdsource project and plan to incorporate a realistic WAN delay distribution model into the Networking Module.

Edge Orchestrator Module: The edge orchestrator module is the decision maker of the system. It uses the information collected from the other modules to decide how and where to handle incoming client requests. In the first version, we simply use a probabilistic approach to decide where to handle incoming tasks but more realistic edge orchestrator can be added by extending abstract Edge Orchestrator class. The edge orchestrator has a strong relationship with the edge server layer which is shown in Figure 3. In EdgeCloudSim, the hosts deploy a WLAN which is located at a fixed physical location with predefined wireless coverage. The VMs served by the near-by hosts are basically the standard VMs provided by CloudSim with lower capacity and a more realistic CPU utilization model. In CloudSim, the bandwidth, RAM, CPU and the storage resources are limited resources considered while creating a VM. When it comes to the tasks, there is no limitation for the number of tasks running on VMs. If there are many tasks to run, simply, the execution of tasks takes longer time. However, this phenomenon is not reasonable for our scenario, because we expect the cloudlet to handle incoming tasks in a very short time interval. Therefore, we implement a new CPU utilization model for the VMs where the number of maximum tasks running in parallel on the VMs is limited. The CPU utilization of a single task is static and it is defined in the configuration file. A dynamic utilization model can be also implemented by re-writing the related CPU utilization class.

Mobility Module: The location of the mobile devices is updated according to the mobility module. Since CloudSim focuses on the conventional cloud computing principles, the mobility is not considered in the framework. In our design, each mobile device has x and y coordinates which are updated according to the dynamically managed hash table. In addition to the mobile devices, we also define locations where dedicated Wi-Fi access points are utilized. As the default radio behavior, it is assumed that the coverage of the Wi-Fi access points is fixed and the link quality is the same for all of the covered clients regardless of their distance to the related access point.

Load Generator Module: The load generator module is responsible for generating tasks for the given configuration.

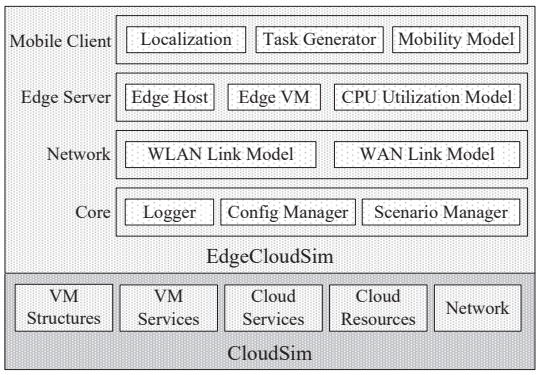


Fig. 3: EdgeCloudSim block diagram.

The mobility and load generator modules are the main components which provide input to other components. They are handled within the same (Mobile Client) layer as shown in Figure 3. They enable collecting additional results which cannot be provided by CloudSim alone, such as the average LAN delay, average number of failed tasks due to mobility and average number of mobile clients in a specific location. By default, the tasks are generated according to a poisson distribution, and the mobile devices move according to the nomadic mobility model. If other task generation or mobility models are required, the mobile device manager module should be modified. It should be noted that, the data size and the length of the tasks should be decided with a proper distribution with respect to the selected task generation distribution. These variables are exponentially distributed random numbers by default because the tasks arrive according to a poisson distribution.

IV. EdgeCloudSim In Operation: the Face Recognition Use-Case

In order to evaluate the performance of the different Edge Computing architectures on EdgeCloudSim, we set up a virtual environment which resembles a university campus where the students walk around and request services from the edge server located at the buildings on the specific territories. In our example scenario, a face recognition application, which could be part of an augmented reality application, is considered, hence the service to be mentioned hereinafter refers to the face recognition service. We compare three different architectures which are (i) single-tier, (ii) two-tier, and (iii) two-tier with edge orchestrator (EO). The single-tier architecture allows the mobile devices to utilize the edge server located in the same building with the student. When it comes to the two-tier architectures, the mobile devices can send their tasks to the global cloud by using the WAN connection provided by the connected access point. The two-tier with EO architecture has a considerable advantage, because for the tasks which are executed on the first tier, only the two-tier with EO architecture

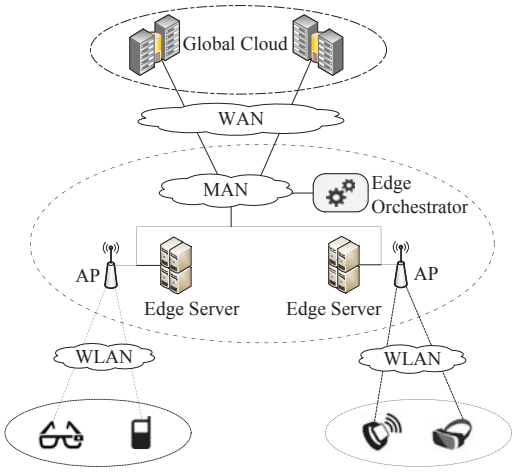


Fig. 4: Generic Edge Computing architecture.

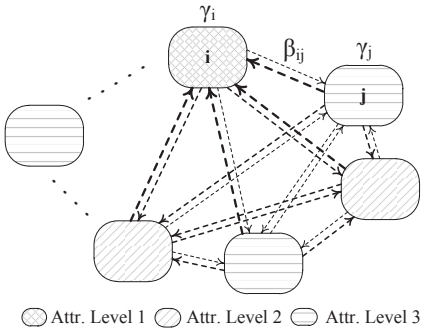


Fig. 5: Mobility model used in the simulation scenario.

can offload the tasks to any edge server located in different buildings. It is assumed that, the edge servers and the edge orchestrator are connected to the same network. A generic Edge Computing architecture is shown in Figure 4.

The students are moving within the buildings according to a nomadic mobility model in which they move to another place after a random amount of time has passed [16]. In our model, there are three location types with different attractiveness levels. If the attractiveness level of the location is high, the student is likely to spend more time in that place. For example, the students spend more time in a library or a cafe than a kiosk. The mobility model is illustrated in Figure 5 where γ is the dwell time and β_{ij} refers the probability of moving from the location i to the location j. In our simulation, the attractiveness level of the location directly affects the dwell time that the student spends in the related place. In fact, the probability of moving to any location is the same for all locations.

It is assumed that each place is covered by a dedicated Wi-Fi access point and the mobile devices join related WLAN when they move to the related location. After joining the WLAN, the mobile devices start sending tasks to the edge server. If a task is decided to be offloaded to the global cloud, the WAN connection provided by the Wi-Fi access point is used. The tasks have random input/output file sizes to upload/download and have random lengths in terms of the number of the instructions. In real life, there may be different types of services which the clients want to use. In our simulation, we model a face recognition application where high CPU computation, small amount of data to download, and bigger amount of data to upload are required. We use 1500 KB upload-data size considering a typical jpeg image, and 15 KB download-data size referring recognized person's metadata. In order to simulate the real life more realistically, we decide to use on/off model as a task generation pattern. According to the on/off model, the users create tasks during on time, and then they wait during the off time. We can say that, there is a state machine for each client where the client can be either in the active or idle state. In Table I, the important simulation parameters are listed.

TABLE I: Simulation parameters.

Parameter	Value
Poisson Interarrival Time of Tasks (second)	3
Simulation Time (hours)	4
Number of repetitions	10
User Mobility Model	Nomadic M. M.
Number of Mobile Devices	50 to 250
Number of Place Type (Attractiveness Level)	3
Probability of selecting a place type	equal (1/3)
Number of places Type1/Type2/Type3	2/4/8
Dwell time of Type1/Type2/Type3 place (minute)	60/30/15
Active/Idle period of the user (second)	45/15
Number of Edge Server per Place	1
Number of VMs per Edge Server/Cloud	4/∞
CPU Utilization of Face Rec. per Edge Server/Cloud (%)	10/$\sim$0
VM Processor Speed (MIPS) per Edge Server/Cloud	1000/20000
Probability of Offloading to Cloud (%)	10
Average Data Size for Upload/Download (KB)	1500/15
Average Task Size (MI)	1500
WAN/WLAN Bandwidth (Mbps)	20/300
WAN Propagation Delay (ms)	100

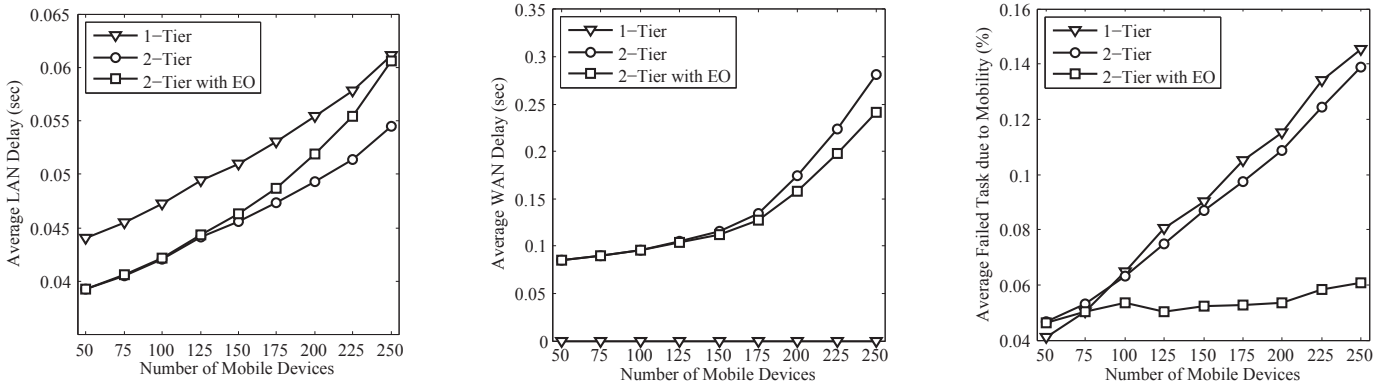
V. Simulation Results and Discussions

We can compare the performance of the sample topologies in many aspects, such as the utilization of the VMs, the service time, and the cost. However, we want to highlight the results which can be only provided by EdgeCloudSim. In Figure 6, the average WAN delay, average LAN delay and the percentage of the failed tasks due to the mobility are given. The total network delay includes both the LAN and the WAN delay. As expected, the amount of time spent on WAN dominates the network delay. Since the single-tier architecture does not send tasks to the global cloud, it has no WAN delay. If the number of the mobile devices in the campus is low, which means there is no congestion, the two-tier topologies provide better LAN delay performance, because they can pass some request to the global cloud. The LAN delay in the two-tier with EO architecture is growing faster than the other architectures. It is misleading to evaluate this graph alone, since the results are also related to the task losses. The average task failure is very low in the two-tier with EO architecture, thus the number of the mobile devices sending/receiving task simultaneously is getting higher. As a result, in this architecture the LAN delay increases faster as the number of mobile device increases.

The task failure due to mobility occurs when the mobile device offloads a task and leaves the coverage area of the WLAN before getting the response. In fact, the average task failure due to the mobility is very low as shown in Figure 6c. The average task failure due to the mobility on the two-tier with EO architecture seems not increasing, because the tasks are distributed to all edge servers and they are executed in a fast manner. As a result, very limited number of tasks fail due to the mobility for all cases. However, a considerable amount of congestion occurs especially in the attractive places on the other topologies. Therefore, the service time increases, and the average task failure due to the mobility increases as well.

We also investigate the effect of the simulation parameters on the results to find out which parameters have an important role in our scenario. For each simulation parameter, we select two exaggerated values and rerun simulations without changing the other parameters. In this paper, only the average task failure and the average service time analysis is explained. In Figure 7, the impact of WAN bandwidth value on the average task failure is shown. 40 Mbps and 4 Mbps bandwidth values are used in Figure 7a and Figure 7b respectively. The singe-tier architecture does not offload tasks to the cloud, as a result the WAN bandwidth has no effect for this architecture. However, it can be clearly seen that the task failure dramatically increases for the topologies which uses the global cloud. In the worst case, the task failure of two-tier architecture increases by 25% due to the WAN congestion.

In Figure 8, the average service time performance is shown with respect to the average task size parameter. In this simulation, the average task size is selected as 250 million instructions (MI) and 4000 MI and the results are given in Figure 8a and Figure 8b respectively. If the average task size is small the average service time performance difference between the single-tier and the other architectures occurs due to the WAN delay because almost no task failure is observed in this case. Since the two-tier and the two-tier with EO architectures has WAN delay, they provides worse performance than the

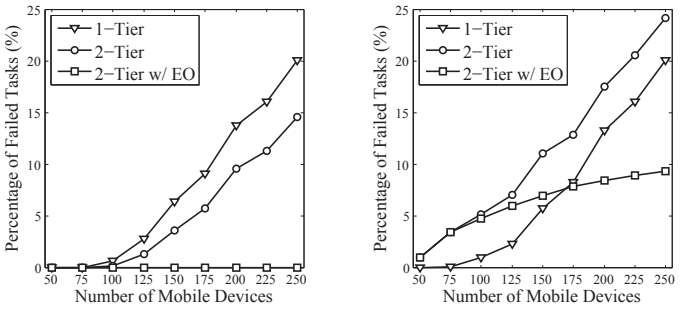


(a) Average LAN delay

(b) Average WAN delay

(c) Average failed task due to mobility

Fig. 6: Important simulation results.



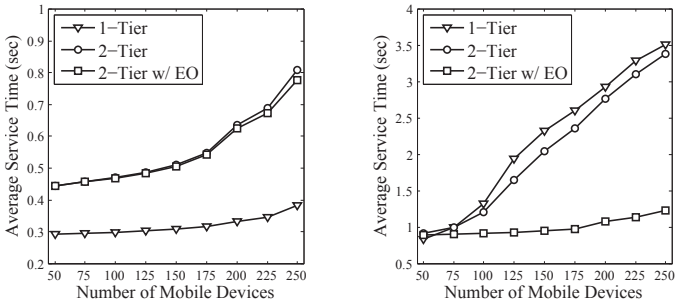
(a) 4 Mbps WAN bandwidth (b) 40 Mbps WAN bandwidth

Fig. 7: Average failed task for different ASDL bandwidths.

single-tier architecture. On the other hand, if the average task size is large, the single-tier and the two-tier architectures encounter with a computational resource congestion on VMs. Therefore the average service time is increasing due to the congestion as the number of mobile devices increases on the single-tier and the two-tier architectures. When it comes to the two-tier with EO architecture, it can distribute the client request to the edge servers, so the service time is not increased as its competitors.

VI. Conclusions and Future Work

Edge Computing promises an opportunity to provide time-sensitive complex applications on the mobile devices with a limited in computational capacity and battery power by offloading the tasks to the near-by server. Since the Edge Computing concept is not standardized yet, it is difficult to develop different architectures or application scenarios. As a result, a simulation environment is commonly used to evaluate the performance of Edge Computing scenarios. In this paper, we introduce a new Edge Computing simulator called EdgeCloudSim which is based on the CloudSim. EdgeCloudSim provides the mobility model, network link model and edge server model to evaluate various aspects of Edge Computing. In the first version, single server queue model is implemented to calculate network delay. In order to increase accuracy, different access technologies can be implemented by using different delay model. In addition, we increase the usability of the EdgeCloudSim by providing a mechanism to get the configuration of devices and applications from the xml files instead of defining them programatically. We plan to add a hand-off mechanism on EdgeCloudSim as a future work. The hand-off mechanism would decrease the task failures due to mobility by using a VM migration method.



(a) 250 MI task size (b) 4000 MI task size

Fig. 8: Average service time for different task sizes.

Acknowledgment

This work is supported by the Galatasaray University Research Foundation under the Grant No. 16.401.002, the Bogazici University Research Fund under the grant number 12663 and the State Planning Organization of Turkey, under the TAM project with the grant number 2007K120610.

References

[1] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless communications and mobile computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*. ACM, 2012, pp. 13–16.

[3] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing, IEEE*, vol. 8, no. 4, pp. 14–23, Oct 2009.

[4] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw. Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[5] S. Long and Y. Zhao, "A toolkit for modeling and simulating cloud data storage: An extension to cloudsim," in *Proceedings of the 2012 International Conference on Control Engineering and Communication Technology*, ser. ICCECT '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 597–600.

[6] J. Jung and H. Kim, "Mr-cloudsim: Designing and implementing mapreduce computing model on cloudsim," in *2012 International Conference on ICT Convergence (ICTC)*, Oct 2012, pp. 504–509.

[7] "EdgeCloudSim," https://github.com/CagataySonmez/EdgeCloudSim, accessed: 2017-01-15.

[8] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct 2016.

[9] Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, and E. Benkhelifa, "The future of mobile cloud computing: Integrating cloudlets and mobile edge computing," in *2016 23rd International Conference on Telecommunications (ICT)*, May 2016, pp. 1–5.

[10] L. A. Tawalbeh, W. Bakheder, and H. Song, "A mobile cloud computing model using the cloudlet scheme for big data applications," in *2016 IEEE First International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, June 2016, pp. 73–77.

[11] H. Routaib, E. Badidi, M. Elmachkour, E. Sabir, and M. ElKoutbi, "Modeling and evaluating a cloudlet-based architecture for mobile cloud computing," in *2014 9th International Conference on Intelligent Systems: Theories and Applications (SITA-14)*, May 2014, pp. 1–7.

[12] A. Betzler, F. Quer, D. Camps-Mur, I. Demirkol, and E. Garcia-Villegas, "On the benefits of wireless sdn in networks of constrained edge devices," in *2016 European Conference on Networks and Communications (EuCNC)*, June 2016, pp. 37–41.

[13] M. H. Chen, M. Dong, and B. Liang, "Joint offloading decision and resource allocation for mobile cloud with computing access point," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 3516–3520.

[14] Y. Liu, M. J. Lee, and Y. Zheng, "Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system," *IEEE Transactions on Mobile Computing*, vol. 15, no. 10, pp. 2398–2410, Oct 2016.

[15] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments," *arXiv preprint arXiv:1606.02007*, 2016.

[16] A. Ribeiro and R. Sofia, "A survey on mobility models for wireless networks," *SITI, University Lusófona, Tech. Rep. SITI-TR-11-01*, 2011.