

Intelligent task processing using mobile edge computing: processing time optimization

Sara Maftah¹, Mohamed El Ghmary², Hamid El Bouabidi¹, Mohamed Amnai¹, Ali Ouacha³

¹Department of Computer Science, Faculty of Sciences, Ibn Tofaïl University, Kenitra, Morocco

²Department of Computer Science, FSDM, Sidi Mohamed Ben Abdellah University, Fez, Morocco

³Department of Computer Science, Faculty of Sciences, Mohammed V University, Rabat, Morocco

Article Info

Article history:

Received Jul 27, 2022

Revised Jan 18, 2023

Accepted Jan 30, 2023

Keywords:

Computation offloading

EdgeCloudSim

Edge orchestration

Internet of things

Mobile edge computing

Offloading decision

Processing optimization

ABSTRACT

The fast-paced development of the internet of things led to the increase of computing resource services that could provide a fast response time, which is an unsatisfied feature when using cloud infrastructures due to network latency. Therefore, mobile edge computing became an emerging model by extending computation and storage resources to the network edge, to meet the demands of delay-sensitive and heavy computing applications. Computation offloading is the main feature that makes Edge computing surpass the existing cloud-based technologies to break limitations such as computing capabilities, battery resources, and storage availability, it enhances the durability and performance of mobile devices by offloading local intensive computation tasks to edge servers. However, the optimal solution is not always guaranteed by offloading computation, therefore, the offloading decision is a crucial step depending on many parameters that should be taken in consideration. In this paper, we use a simulator to compare a two-tier edge orchestrator architecture with the results obtained by implementing a system model that aims to minimize a task's processing time constrained by time delay and the limited device's computational resource and usage based on a modified version.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Sara Maftah

Department of Computer Science, Faculty of Sciences, Ibn Tofaïl University

University Campus, BP 133, Kenitra, Morocco

Email: sara.maftah@uit.ac.ma

1. INTRODUCTION

Our world today is dominated by cloud computing which is an efficient computing platform that grew rapidly over the last few decades. Driven by the endless connected devices which represent the internet of things and their massive real-time computing and processing demands, as well as the astonishing evolution of communication and networking technologies, cloud computing infrastructures became inappropriate to perform with an acceptable level in face of these high quality requirements. Motivated by these challenges and the need to make a move towards the 5th Generation of cellular network, mobile edge computing appeared with capabilities such as storage and computational capacities that offer low latency, high bandwidth and real-time access [1]. The concept of edge computing is regarded as the mechanism that allows the computation to be performed at the edge of the network [2]. However, the main challenge is that the internet of things connects many heterogeneous devices with limited local computing resources, which will require a strategy that enables these devices to offload their heavy tasks to a processing environment represented by the deployed virtual machines on the nearby edge servers [3]. Therefore, computation offloading comes in useful with the

ability to overcome the resource constraints on user devices, especially for the computation intensive tasks [2]. Computation offloading is then considered as a way to offer powerful infrastructure resources to augment the computing capabilities of mobile devices [4].

In computation offloading, a mobile device can adopt one of the following three modes: local execution, partial offloading and full offloading [5]. This process, made to achieve a minimal task completion and the least energy consumption, involves application partitioning, task execution and offloading decision on which, according to [6], the edge server depends to calculate the execution time of each mobile devices' request. In this regard, since most of the existing literature concerns only one or two of the said metrics, a variety of solutions have been proposed, and the related publications according to [7] has increased in the recent years to make joint optimization a promising research area. Furthermore, Shakarami *et al.* [8] produced a well organized review on computation offloading mechanisms that are based on game theory methodology. In recent surveys [9]-[10], many existing offloading methods were examined, such as machine learning and artificial intelligence methodologies, which proved to have an important impact on the subject. Lin *et al.* [9] found that machine learning methods can solve the scalability issue in large scale computation offloading based on a centralized mode to achieve an intelligent decision. Moreover, in [11] a detailed taxonomy of offloading mechanisms based on machine learning was proposed.

In order to improve the offloading rate and reduce the energy consumption of the device in a single user multitask scenario, in [12] dynamically adjusted the transmitting power and local central processing unit (CPU) frequency. Adopting the same single multitask device scenario, where the task has an execution time deadline and the device is constrained by limited energy, El Ghmary *et al.* [13], [14] aimed to minimize the energy consumption by using simulated annealing that serves to decide the tasks' offloading and the resource allocation. The offloading decision is obtained based on a multitask scenario, since a single application running on a mobile device is generally divided into multiple tasks which means the offloading decision concerns each one of these tasks [15]. In a more complex environment consisting of an edge server and multiple mobile devices each having a set of tasks, Huang *et al.* [16] discussed the computation offloading and considered its optimization as a mixed integer non linear programming problem, to which they proposed an algorithm based on time and energy consumption to optimize the computation offloading process. Moreover, when it comes to time consumption resources, it includes local and edge processing time as well as communication time consumption in which uploading the input data and downloading the output result. However, most studies ignore the latter, assuming that the output data size is insignificant when compared with the input data size [5], [12], [15]-[17].

Besides, knowing that internet of things sensors are generally powered by batteries with a limited capacity, Khan *et al.* [18] added energy harvesting devices that are being highly considered to improve energy efficiency along with computation offloading in a scenario consisting of multiple mobile devices and edge servers, the authors then proposed an improved strategy based on integer linear programming to solve the energy consumption issue. Likewise, aiming an energy optimized model, Bi *et al.* [19] proposed an offloading decision optimization based on a genetic and simulated annealing method. Meanwhile, Zhang *et al.* [20] combined a simulated annealing method with the genetic algorithm for the purpose of selecting mobile edge servers automatically. Furthermore, in order to minimize the costs, Kuang *et al.* [21] solved the offloading decision problem by comparing the genetic algorithm, the greedy strategy and backtracking to find that the greedy strategy is the suitable one for their problem resolution.

There are many tools available for researchers to simulate and obtain real world experiment results, such as EdgeCloudSim [22], in which the authors implemented various samples using different architectures in each one of them, to prove the benefits of deploying the edge computing paradigm [23]. The existing simulation of computation offloading uses the virtual machine capacity to decide whether to offload or process locally. Due to the constraint of the mobile device's energy consumption and the limited time that a task should take to be completed, we propose a new offloading decision making mechanism in which local processing time will be taken in consideration as well as the mobile device's energy consumption and compare it with the existing two-tier with an edge orchestrator architecture. After introducing the theme and its literature, the paper is structured as follows. Section 2 is a description of the system model. The problem is formulated in section 3 as well as the elaboration of its resolution. The simulation environment is then described in section 4 and the results are presented and discussed in section 5. Finally, section 6 represents the paper's conclusion.

2. SYSTEM MODEL

Computation offloading in mobile edge computing was introduced to support the interconnection of resource-limited devices with the internet, it enables the mobile device to offload a part of the computation to a nearby remote server in order to increase its capabilities and prolong its battery lifespan. However, computation offloading is a complicated process based on three key components: task partition, offloading decision and resource allocation [9]. It is also divided into two distinct modes: binary and partial offloading [9]. For the purpose of studying this technique, we will consider an architecture where a single mobile device needs to process N independent tasks, either by executing the task locally or offloading it to the mobile edge server according to a certain decision making mechanism. Figure 1 is an illustration of the general deployed architecture.

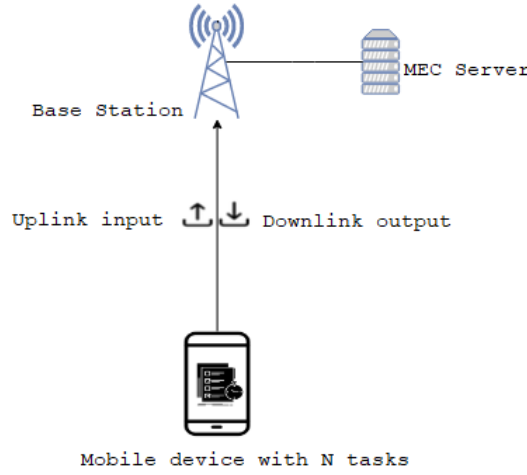


Figure 1. System model topology

The adopted system model involves, as said before, a Single Mobile device that contains a set of N independent tasks ready to be processed, either by the mobile device itself or by offloading it to a nearby edge server in case the local resources were not enough. The computation offloading decision can minimize the processing time [24], hence the response time will be drastically improved [25] as well as energy consumption [26], [27]. Each task is represented by $i \in N$ where $N = \{1, 2, \dots, N\}$ and identified by its data size D_i^{size} , data input size D_i^{in} and data output size D_i^{out} . In the context of studying computation offloading in a mobile edge computing server, we will focus on the communication model while considering the usage of the mobile device. Assuming that tasks are independent, communication and computation costs will be calculated separately, on both the mobile device and the mobile edge computing server.

2.1. Local processing

Given D_i^{size} as the data size of a task $i \in N = \{1, 2, \dots, N\}$, and S^{md} as the processing speed of the mobile device, the processing time will be calculated as shown in (1). The processing time T_i^{md} represents the time cost that will be required for a mobile device to process and execute a certain task i . T_i^{md} will be used later to calculate local energy consumption.

$$T_i^{md} = \frac{D_i^{size}}{S^{md}} \quad (1)$$

2.2. Edge processing

When the computation resources in the mobile device are not enough to process a certain task, it gets offloaded to the mobile edge computing server. Thus, response time in this case involves a transmission delay which represents both uploading the input and downloading the output as well as the time for processing the task on the mobile edge computing server. In order to get the cost of response time, we will elaborate both transmission delay and the processing time.

- Transmission delay: it includes both time to upload the task's input data denoted as T_i^{up} as well as time to download its output data denoted as T_i^{down} , those metrics are related to the transmission bandwidth bw . The representation of each one is shown in the (2) and (3):

$$T_i^{up} = \frac{D_i^{in}}{bw} \quad (2)$$

$$T_i^{down} = \frac{D_i^{out}}{bw} \quad (3)$$

- Processing time: time for the mobile edge computing server to process a certain task, denoted as T_i^{mec} where S^{mec} is the processing speed of the mobile edge computing server.

$$T_i^{mec} = \frac{D_i^{size}}{S^{mec}} \quad (4)$$

- Total time cost: in order to get the total time cost in case a task i was offloaded to the mobile edge computing server, we will be using the (2)-(4) to calculate T_i^{off} .

$$T_i^{off} = \frac{D_i^{in}}{bw} + \frac{D_i^{size}}{S^{mec}} + \frac{D_i^{out}}{bw} \quad (5)$$

2.3. Local energy consumption

In the previous section we identified and denoted the various equations to calculate time consumption at each level. Similarly, with each operation, whether it's processing, transmitting or waiting for a response, both the mobile device and the mobile edge computing server consume energy. In this paper, we focus on the energy consumed by the mobile device only, this consumption is composed of three parts. In case the task is processed locally we calculate the energy cost of local processing, otherwise, the task gets offloaded and we calculate the energy cost while transmitting the data which includes uploading the data and downloading the results and the idle energy, which represents the consumption while the mobile device is on hold and waiting for the task to be processed on the mobile edge computing server. The energy consumption model for the mobile device is then divided into three modes: local processing, transmission and idle which is a state, according to [28], where Mobile devices are often on a connected-standby operation mode, which means they are idle and connected to the communication channels as well for any eventual usability. This mode allows the mobile devices to run on low power which increases the battery life. Energy consumption on the mobile device while processing a task is denoted as E_i^{local} and calculated by multiplying time cost T_i^{md} by the consumed power by the mobile device P_{proc}^{md} while processing the task.

$$E_i^{local} = T_i^{md} * P_{proc}^{md} \quad (6)$$

2.4. Computation offloading decision

The decision, in computation offloading, to whether process a certain task locally or send it to a more powerful device, the mobile edge server in our case, is crucial. This process consists of migrating computing tasks to higher resourceful servers that are located at the edge of the network. Jaddoa *et al.* [25] proposed a dynamic offloading decision based on response time and energy consumption. Likewise, Liao *et al.* [29] implemented a joint decision making based on both communication and computation resources. Meanwhile, in [24], [30], [31] the decision to offload was based on a fuzzy logic using minimum and maximum functions to determine the result of multiple combined rules within a set. In our paper, knowing that mobile devices have limited computing resources, the decision mechanism is based on setting a limited processing time in the mobile device in order to deliver an output result within a reasonable response time, as well as balancing the usage of these resources by setting a CPU usage threshold. The CPU usage in a utilization-based approach affects directly the consumed power [32], and it is considered that energy consumption and CPU utilization increase linearly [27], [33], [34].

3. PROBLEM FORMULATION AND RESOLUTION

3.1. Problem formulation

Given $x_i \in \{0, 1\}$ the offloading decision of a certain task i , represented by two possibilities. If $x_i = 0$, the task is processed locally, however, if $x_i = 1$ the computation is then offloaded to the mobile edge server. In this paper, we aim to reduce the processing time of tasks considering a processing deadline and energy consumption that should not be exceeded by the mobile device. Hence, the optimization problem can be written as follows:

$$\begin{aligned} \min_{x_i, S^{md}, P_{proc}^{md}, S^{req}} \quad & \sum_{i=1}^N (1 - x_i) * \frac{D_i^{size}}{S^{md}} + \sum_{i=1}^N x_i * \left(\frac{D_i^{in}}{bw} + \frac{D_i^{size}}{S^{mec}} + \frac{D_i^{out}}{bw} \right) \quad (P1) \\ \text{s.t.} \quad & C1: \quad x_i \in (0, 1) \quad \forall \quad i \in \{1, 2, \dots, N\} \\ & C2: \quad \forall \quad i \in \{1, 2, \dots, N\} \quad (1 - x_i) * \frac{D_i^{size}}{S^{md}} < T_i^{max} \\ & C3: \quad \forall \quad i \in \{1, 2, \dots, N\} \quad \sum_{i=1}^N (1 - x_i) * \frac{D_i^{size}}{S^{md}} * P_{proc}^{md} < E^{max} \\ & C4: \quad \forall \quad i \in \{1, 2, \dots, N\} \quad x_i * S_i^{req} < S^{mec} \end{aligned}$$

where the first constraint $C1$ implies that the offloading decision x_i of a task i is a binary variable and should be equal to 0 or 1. The second constraint $C2$ was set to limit the processing time locally T_i^{md} , in case the processing time of a task i exceeds the maximum value T_i^{max} , which represents the maximum time cost for a single task to be processed. Therefore, if $T_i^{md} > T_i^{max}$ the task i gets offloaded to an edge server. The third constraint $C3$ makes sure, in case of local processing, that the energy consumed while processing the set of tasks of a certain application does not exceed a maximum value E^{max} which represents a maximum energy consumption by the mobile device.

Finally, using EdgeCloudSim, the load generator module generates a certain task i , and the orchestrator predicts the required virtual machine capacity and compares it with the available resources at that moment whether on the mobile device or the edge server. Therefore, the required processing speed for a certain task S_i^{req} is compared to the virtual machine processing speed S^{mec} accordingly to satisfy the fourth constraint $C4$. The value of S^{md} is defined before starting the simulation, along with other metrics. Once the simulation begins, the load generator generates the tasks and predicts its required processing speed S_i^{req} in a sequential order based on their start time, and each task has the necessary characteristics such as data size D_i^{size} , input D_i^{in} and output D_i^{out} size and the required computing resources. The objective function aims processing time minimization based on a deadline and a maximum energy consumption on the mobile device as well as the required processing speed.

3.2. Offloading decision mechanism

Assuming that the offloading decision is known, resolving the problem P1 is achieved by satisfying the processing time in a condition of a deadline and maximum energy consumption in case of local processing. Otherwise, in case of offloading the task, the required processing speed will be considered for the task's completion. S^{md} is given as a constant which is used to compute the local processing time, and therefore T_i^{md} will be subjected to $C2$. The CPU usage is calculated by EdgeCloudSim using a class that provides a realistic utilization model based on some metrics, mainly the usage percentage and processing speed required by the task. Therefore, in case $x_i = 0$ the utilization model predicts the task's requirements and its CPU usage in the mobile device, and calculates afterwards the consumed energy to satisfy the third constraint $C3$. However, in case $x_i = 1$, P1 is achieved by satisfying the task's processing time in a condition of processing speed requirement only, hence P1 is subjected to $C4$.

The latest version of EdgeCloudSim is provided with five different samples and each one of them functions different than the other according to a certain processing scenario and orchestration mechanism. The purpose of this paper is on one hand to show the utility of computation offloading and its added value in terms of processing time. On the other hand, implementing an algorithm which is an initiation to experiment different decision making mechanisms in order to obtain better results. The algorithm is an implementation of the objective function aiming to minimize processing time considering a local processing deadline, a mobile device energy consumption threshold and a processing speed requirements.

3.3. Modified two_tier with edge orchestator algorithm

Our approach to tackle the problem we formulated in 3.1. consists on a simple implementation of the objective function. The objective function aims to minimize the processing time considering a local processing deadline, a mobile device energy consumption threshold and a processing speed requirement. The modified two_tier with edge orchestrator algorithm shown in Algorithm 1.

Algorithm 1 : Modified two_tier with edge orchestrator algorithm

```

1:  $D_i^{size}, S^{md}, S^{mec}, S_i^{req}, P_{proc}^{md}, T_i^{max}, E^{max}$ 
2: for  $i \in N$  do
3:   Estimate  $S_i^{req}$ 
4:    $T_i^{md} = \frac{D_i^{size}}{S^{md}}$ 
5:    $T_i^{mec} = \frac{D_i^{size}}{S^{mec}}$ 
6:    $E_i^{local} = T_i^{md} * P_{proc}^{md}$ 
7:   for  $j \in N$  do
8:      $E_j^{local} \leftarrow E_i^{local}$ 
9:      $E^{local} += E_j^{local}$ 
10:  end for
11:  if  $T_i^{md} < T_i^{max}$  and  $E^{local} < E^{max}$  then
12:     $tmp \leftarrow 0$ 
13:  else
14:    if  $S_i^{req} < S^{mec}$  then
15:       $tmp \leftarrow 1$ 
16:    end if
17:  end if
18:   $x_i \leftarrow tmp$ 
19: end for

```

4. SIMULATION ENVIRONMENT

A number of simulation experiments have been conducted using the mentioned simulating tool, which is an extended version of CloudSim [35], developed to match the edge computing environment and provide accurate results. A simulation starts with an initialization phase by loading the configuration files needed to run the scenario and by generating a random set of tasks that will be processed sequentially according to their start time. For each task, there will be a decision on whether to process it on the mobile device or offload it to an edge server. This decision will be based on multiple factors.

5. RESULTS AND DISCUSSION

In order to compare the processing time of the generated tasks, the results were obtained by running each scenario multiple times. In each experiment, we adopted three different scenarios: Mobile, Edge and Hybrid. The first one is an only mobile processing scenario in which the tasks are executed locally in the mobile device. Meanwhile the second scenario depends on a coarse-grained offloading [36] and offloads the entire set of tasks to an edge server. Finally, the third scenario depends on partial offloading or what we can call fine-grained offloading [36]. In this case, only the tasks with a heavy computation are offloaded according to a decision making strategy in order to optimize the response time. Moreover, in order to evaluate the adopted offloading decision making mechanism, we compare the results of a modified approach of the original Two-tier with Edge Offloading architecture. Figure 2 and Figure 3 are graphical representations of processing a load of independent tasks sequentially, in terms of response time using the three different scenarios and opting two different offloading decision making mechanisms. In Figure 2 we use the Two-tier with Edge Offloading mechanism which depends on comparing the required capacity to process the task and the available resources. We accumulate afterwards the time cost throughout the experiment, which resulted a total response time cost of **1272.5716**, **544.3834** and **1055.8426** seconds adopting respectively the Mobile, Edge and Hybrid scenario. It is obvious that response time when the task is processed in the edge server is better due to the available computing resources, meanwhile, in the hybrid scenario, the mobile device depends mainly on its resources until they are no longer enough, however it offloads at the same time the tasks which percentage usage was predicted to be demanding which explains the high time consumption at first (local processing and computation

offloading transmission delays). Meanwhile, in Figure 3 the benefits of a full or partial offloading to an edge server are obvious in terms of processing time.

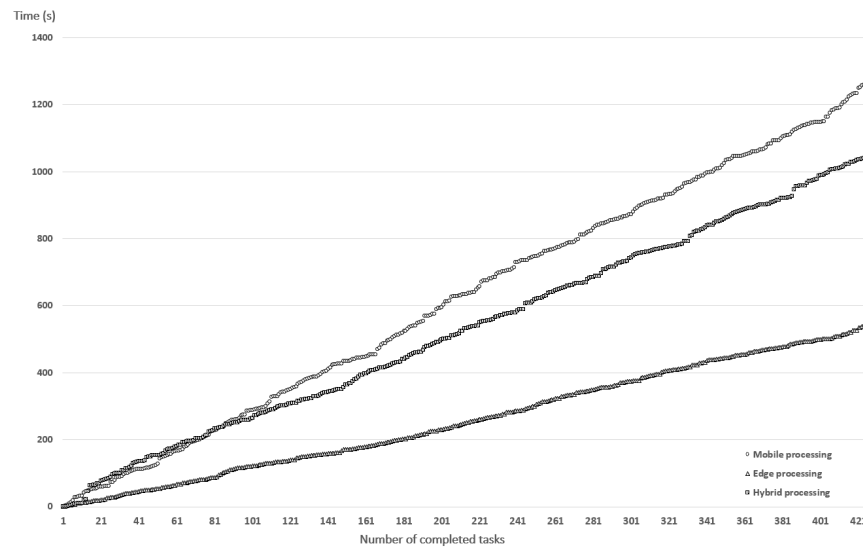


Figure 2. Time consumption while processing a set of heavy tasks using a Two-tier with edge orchestrator architecture

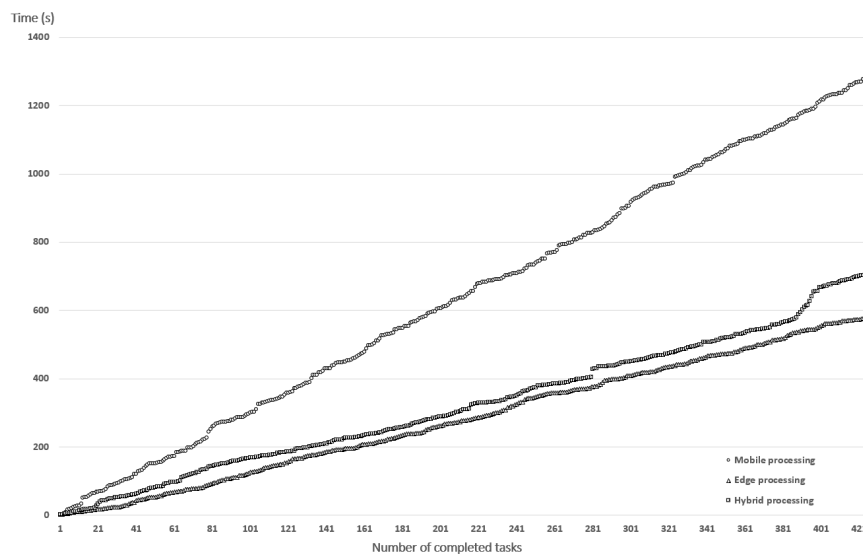


Figure 3. Time consumption while processing a set of heavy tasks using a modified Two-tier with edge orchestrator architecture

Moreover, in case of partial offloading, as said before, the decision making is the crucial part. In this experiment, beside comparing the required capacity to process a task and the available resources, we depend on three metrics to decide whether to offload or not, which are the local processing time and the CPU usage, hence energy consumption. We obtained a total response time cost of **1289.2961**, **581.3376** and **711.8558** seconds adopting respectively the Mobile, Edge and Hybrid scenario. We compare the results obtained from the Two-tier with edge orchestrator architecture and the modified version to find that **343.9868** seconds were saved when processing the tasks based on the customized offloading decision. Figure 4 represents the total processing cost in terms of time consumption and shows the enhancement made when we adopted the customized offloading decision making in the hybrid processing scenario. Hence, whether opting for the Two-tier

with edge orchestrator architecture or a modified version of this strategy, response time is optimized. However, the average response time was reduced by approximately **32.5%** compared with the initial results in a hybrid scenario.

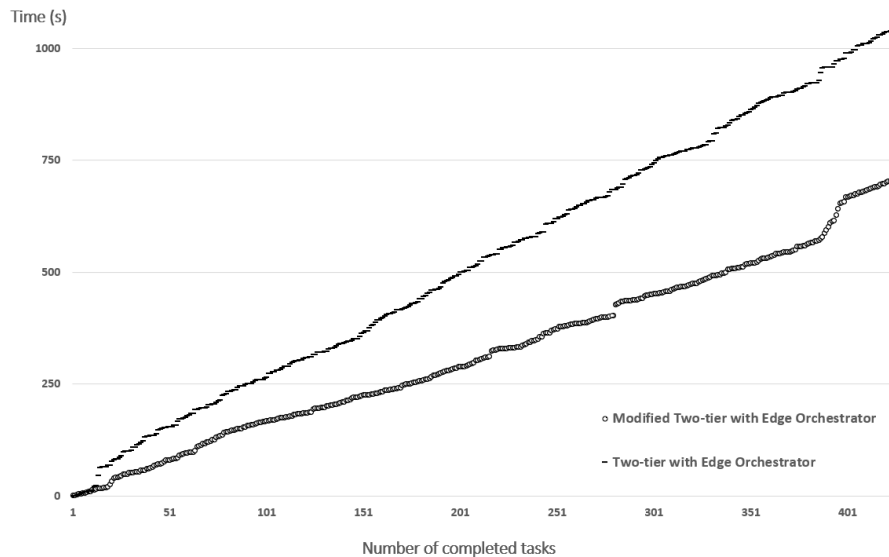


Figure 4. Comparing time consumption using two different offloading decision making mechanism

6. CONCLUSION

The aim of this study is to simulate the computation offloading process between a mobile device and an edge server in order to improve the processing time of completed tasks, by adopting an offloading decision considering the virtual machine capacity, a local processing deadline, as well as a limited CPU usage when it comes to the mobile device. The obtained results are reduced in terms of processing time and energy consumption as well compared to results provided by the simulation tool EdgeCloudSim using an existing Two-tier with edge orchestrator and a modified version of the said architecture. This paper is the first step into computation offloading in edge computing, a field that caught the attention of many researchers, and a first attempt to implement a strategy that could be extended in future work to include an optimization problem where energy consumption whether on the mobile device or the mobile edge computing server is also considered in the process of the computation offloading decision. Including the cloud into our system model is also intended to have a larger ground for further experiments.

REFERENCES




- [1] N. Hassan, K.-L. A. Yau, and C. Wu, "Edge computing in 5g: A review," *IEEE Access*, vol. 7, pp. 127 276–127 289, 2019, doi: 10.1109/ACCESS.2019.2938534.
- [2] D. B. Abdullah and H. H. Mohammed, "Computation offloading in the internet of connected vehicles: A systematic literature survey," in *Journal of Physics: Conference Series*, vol. 1818, no. 1, 2021, p. 012122, doi: 10.1088/1742-6596/1818/1/012122.
- [3] A. Ouacha and M. El Ghmary, "Virtual machine migration in mec based artificial intelligence technique," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 1, p. 244, 2021, doi: 10.11591/ijai.v10.i1.pp244-252.
- [4] L. Lin, X. Liao, H. Jin, and P. Li, "Computation offloading toward edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1584–1607, 2019, doi: 10.1109/JPROC.2019.2922285.
- [5] Y. Deng, Z. Chen, X. Yao, S. Hassan, and A. M. Ibrahim, "Parallel offloading in green and sustainable mobile edge computing for delay-constrained iot system," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 12 202–12 214, 2019, doi: 10.1109/TVT.2019.2944926.
- [6] A. Shahidinejad and M. Ghobaei-Arani, "Joint computation offloading and resource provisioning for edge-cloud computing environment: A machine learning-based approach," *Software: Practice and Experience*, vol. 50, no. 12, pp. 2212–2230, 2020, doi: 10.1002/spe.2888.
- [7] B. Wang, C. Wang, W. Huang, Y. Song, and X. Qin, "A survey and taxonomy on task offloading for edge-cloud computing," *IEEE Access*, vol. 8, pp. 186 080–186 101, 2020, doi: 10.1109/ACCESS.2020.3029649.

- [8] A. Shakarami, A. Shahidinejad, and M. Ghobaei-Arani, "A review on the computation offloading approaches in mobile edge computing: A game-theoretic perspective," *Software: Practice and Experience*, vol. 50, no. 9, pp. 1719–1759, 2020, doi: 10.1002/spe.2839.
- [9] H. Lin, S. Zeadally, Z. Chen, H. Labiod, and L. Wang, "A survey on computation offloading modeling for edge computing," *Journal of Network and Computer Applications*, p. 102781, 2020, doi: 10.1016/j.jnca.2020.102781.
- [10] G. Carvalho, B. Cabral, V. Pereira, and J. Bernardino, "Computation offloading in edge computing environments using artificial intelligence techniques," *Engineering Applications of Artificial Intelligence*, vol. 95, p. 103840, 2020, doi: 10.1016/j.engappai.2020.103840.
- [11] A. Shakarami, M. Ghobaei-Arani, and A. Shahidinejad, "A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective," *Computer Networks*, p. 107496, 2020, doi: 10.1016/j.comnet.2020.107496.
- [12] H. Li, "Multi-task offloading and resource allocation for energy-efficiency in mobile edge computing," *International Journal of Computer Techniques*, vol. 5, no. 1, pp. 5–13, 2018, doi: 10.29126/23942231/IJCT-V5I1P2.
- [13] M. El Ghmary, T. Chanyour, Y. Hmimz, and M. O. C. Malki, "Efficient multi-task offloading with energy and computational resources optimization in a mobile edge computing node," *International Journal of Electrical & Computer Engineering*, vol. 9, no. 6, 2019, doi: 10.11591/ijece.v9i6.pp4908-4919.
- [14] M. El Ghmary, Y. Hmimz, T. Chanyour, and M. O. C. Malki, "Energy and processing time efficiency for an optimal offloading in a mobile edge computing node," *International Journal of Communication Networks and Information Security*, vol. 12, no. 3, pp. 389–393, 2020, doi: 10.17762/ijcnis.v12i3.4750.
- [15] M. El Ghmary, Y. Hmimz, T. Chanyour, and M. O. C. Malki, "Time and resource constrained offloading with multi-task in a mobile edge computing node," *International Journal of Electrical & Computer Engineering*, vol. 10, 2020, doi: 10.11591/ijece.v10i4.pp3757-3766.
- [16] M. Huang, Q. Zhai, Y. Chen, S. Feng, and F. Shu, "Multi-objective whale optimization algorithm for computation offloading optimization in mobile edge computing," *Sensors*, vol. 21, no. 8, p. 2628, 2021, doi: 10.3390/s21082628.
- [17] T. Chanyour, Y. Hmimz, M. El Ghmary, and M. O. C. Malki, "Delay-aware and user-adaptive offloading of computation-intensive applications with per-task delay in mobile edge computing networks," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 1, 2020.
- [18] P. W. Khan, K. Abbas, H. Shaiba, A. Muthanna, A. Abuarqoub, and M. Khayyat, "Energy efficient computation offloading mechanism in multi-server mobile edge computing—an integer linear optimization approach," *Electronics*, vol. 9, no. 6, p. 1010, 2020, doi: 10.3390/electronics9061010.
- [19] J. Bi, H. Yuan, S. Duanmu, M. C. Zhou, and A. Abusorrah, "Energy-optimized partial computation offloading in mobile edge computing with genetic simulated-annealing-based particle swarm optimization," *IEEE Internet of Things Journal*, 2020, doi: 10.1109/IIOT.2020.3024223.
- [20] Y.-w. Zhang, W.-m. Zhang, K. Peng, D.-c. Yan, and Q.-l. Wu, "A novel edge server selection method based on combined genetic algorithm and simulated annealing algorithm," *Automatika*, vol. 62, no. 1, pp. 32–43, 2021, doi: 10.1080/00051144.2020.1837499.
- [21] L. Kuang, T. Gong, S. OuYang, H. Gao, and S. Deng, "Offloading decision methods for multiple users with structured tasks in edge computing for smart cities," *Future Generation Computer Systems*, vol. 105, pp. 717–729, 2020, doi: 10.1016/j.future.2019.12.039.
- [22] C. Sonmez, A. Ozgovde, and C. Ersoy, "Edgecloudsim: An environment for performance evaluation of edge computing systems," *Transactions on Emerging Telecommunications Technologies*, vol. 29, no. 11, p. e3493, 2018, doi: 10.1002/ett.3493.
- [23] C. Sonmez, A. Ozgovde, and C. Ersoy, "Performance evaluation of single-tier and two-tier cloudlet assisted applications," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2017, pp. 302–307, doi: 10.1109/ICCW.2017.7962674.
- [24] V. Nguyen, T. T. Khanh, T. Z. Oo, N. H. Tran, E.-N. Huh, and C. S. Hong, "Latency minimization in a fuzzy-based mobile edge orchestrator for iot applications," *IEEE Communications Letters*, vol. 25, no. 1, pp. 84–88, 2020, doi: 10.1109/LCOMM.2020.3024957.
- [25] A. Jaddoa, G. Sakellari, E. Panaousis, G. Loukas, and P. G. Sarigiannidis, "Dynamic decision support for resource offloading in heterogeneous internet of things environments," *Simulation Modelling Practice and Theory*, vol. 101, p. 102019, 2020, doi: 10.1016/j.simpat.2019.102019.
- [26] M. Rahati-Quchani, S. Abrishami, and M. Feizi, "An efficient mechanism for computation offloading in mobile-edge computing," *arXiv preprint arXiv:1909.06849*, 2019, doi: 10.48550/arXiv.1909.06849.
- [27] M. Daraghme, I. Al Ridhawi, M. Aloqaily, Y. Jararweh, and A. Agarwal, "A power management approach to reduce energy consumption for edge computing servers," in *2019 Fourth International Conference on Fog and Mobile Edge Computing (FMEC)*, 2019, pp. 259–264, doi: 10.1109/FMEC.2019.8795328.
- [28] J. Haj-Yahya, Y. Sazeides, M. Alser, E. Rotem, and O. Mutlu, "Techniques for reducing the connected-standby energy consumption of mobile devices," in *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2020, pp. 623–636, doi: 10.1109/HPCA47549.2020.00057.
- [29] Y. Liao, L. Shou, Q. Yu, Q. Ai, and Q. Liu, "Joint offloading decision and resource allocation for mobile edge computing enabled networks," *Computer Communications*, vol. 154, pp. 361–369, 2020, doi: 10.1016/j.comcom.2020.02.071.
- [30] C. Sonmez, A. Ozgovde, and C. Ersoy, "Fuzzy workload orchestration for edge computing," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 769–782, 2019, doi: 10.1109/TNSM.2019.2901346.
- [31] T. T. Khanh, V. Nguyen, and E.-N. Huh, "Fuzzy-based mobile edge orchestrators in heterogeneous IoT environments: An online workload balancing approach," *Wireless Communications and Mobile Computing*, vol. 2021, 2021, doi: 10.1155/2021/5539186.
- [32] Q.-H. Nguyen and F. Dressler, "A smartphone perspective on computation offloading—a survey," *Computer Communications*, vol. 159, pp. 133–154, 2020, doi: 10.1016/j.comcom.2020.05.001.
- [33] E. Ahvar, A.-C. Orgerie, and A. Lebre, "Estimating energy consumption of cloud, fog and edge computing infrastructures," *IEEE Transactions on Sustainable Computing*, 2019, doi: 10.1109/TSUSC.2019.2905900.
- [34] T. Ciesielczyk et al., "An approach to reduce energy consumption and performance losses on heterogeneous servers using power capping," *Journal of Scheduling*, pp. 1–17, 2020, doi: 10.1007/s10951-020-00649-4.
- [35] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp.




- 23–50, 2011, doi: 10.1002/spe.995.
- [36] Z. Wang, W. Hao, L. Yan, Z. Han, and S. Yang, “Cooperative scheduling of multi-core and cloud resources: fine-grained offloading strategy for multithreaded applications,” *IET Communications*, vol. 14, no. 10, pp. 1632–1641, 2020, doi: 10.1049/iet-com.2019.1060.

BIOGRAPHIES OF AUTHORS






Sara Maftah    is a Ph.D. student at the Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco. Affiliated with the Research Laboratory in Computer Science and Telecommunications (LaRIT). She obtained a Research Master’s degree in Cloud and High Performance Computing from the National School for Computer Science and System Analysis (Ensias), Mohammed V University (UM5), Rabat, Morocco. She can be contacted at email: sara.maftah@uit.ac.ma.






Mohamed El Ghmary    is a Professor of Computer Science at the Faculty of Sciences Dhar El Mahraz (FSDM), Sidi Mohamed Ben Abdellah University, Fez, Morocco. He is an associate member of the Intelligent Processing and Security of Systems (IPSS) team of Computer Science Department at the Faculty of Sciences, Mohamed V University (UM5), Rabat Morocco. He is also an associate member of Research Laboratory in Computer Science and Telecommunications (LaRIT), Team Networks and Telecommunications, Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco. His research interests are mobile edge computing (MEC), Cloud computing, machine learning, deep learning, intelligent systems and optimization. He can be contacted at email: mohamed.elghmary@usmba.ac.ma.






Hamid El Bouabidi    is a Ph.D. student at the Faculty of Sciences, Ibn Tofail University, Kenitra, Morocco. Affiliated with the Research Laboratory in Computer Science and Telecommunications (LaRIT), Team Networks and Telecommunications. He obtained a Research Master’s degree in Cloud and High-Performance Computing from the National School for Computer Science and Systems Analysis (ENSIAS), Mohammed V University (UM5), Rabat, Morocco. He can be contacted at email: hamid.elbouabidi@uit.ac.ma.



Mohamed Amnai    has been an Assistant at National School of Applied Sciences Khouribga, Settat University, Morocco. He joined, the Department of Computer Science and Mathematics, Faculty of Sciences Tofail University, Kenitra, Morocco, as an Associate Professor in 2018. He is also an associate member of Research Laboratory in Computer Science and Telecommunications (LaRIT), Team Networks and Telecommunications Faculty of Science, Kenitra, Morocco. He is also an associate member of laboratory IPOS National School of Applied Sciences, Hassan 1 University, Khouribga, Morocco. He can be contacted at email: mohamed.amnai@uit.ac.ma.



Ali Ouacha    is a professor of computer science at the Faculty of Sciences of Rabat (FSR) Mohamed V University (UM5). He received his doctorate degree from the Mohammadia School of Engineering (EMI). Ouacha is a permanent member of the Intelligent Processing and Security of Systems (IPSS) team of Computer Science Department at the FSR. His research is currently focused on optimizing the performance of routing protocols in an internet of things (IoT), mobile edge computing (MEC), and mobile ad-hoc networks environment. He is the author and co-author of several publications in international journals and conferences. Ouacha is a member of the organizing committees of several scientific events (Conferences and Congresses). He is also a member of the Technical Program Committee (TPC) of several international conferences and journals. He can be contacted at email: a.ouacha@um5.ac.ma.