Marginal Cost of Computation as a Collaborative Strategy for Resource Management at the Edge

Emanuele Carlini[2][0000−0003−3643−5404], Patrizio Dazzi[1][0000−0001−8504−1503], Luca Ferrucci[1][0000−0003−4159−0644], Jacopo Massa[1][0000−0002−5255−537X], and Matteo Mordacchini[3][0000−0002−1406−828X]

[1] Department of Computer Science, University of Pisa, Pisa, 56126 Italy
(e-mail: luca.ferrucci@gmail.com, patrizio.dazzi@unipi.it, jacopo.massa@phd.unipi.it)
[2] ISTI-CNR, National Research Council, Pisa, 56124 Italy
(e-mail: emanuele.carlini@isti.cnr.it)
[3] IIT-CNR, National Research Council, Pisa, 56124 Italy
(e-mail: matteo.mordacchini@iit.cnr.it)

Abstract. This paper explores an extended applications' cost function to model the willingness of Edge data centres to accommodate additional users in decentralized edge computing environments. By enhancing the Marginal Computing Cost per User (MCU) concept, we introduce a dynamic cost factor influenced by the number of users currently served. Through extensive simulations conducted on the PureEdgeSim platform, we evaluate the impact of this variable MCU on system performance across various configurations. The results reveal a critical trade-off between cost sensitivity (i.e., collaboration willingness) of Edge data centres and optimization potential. This work offers insights into user allocation strategies in heterogeneous edge systems and sets the stage for future research into non-linear MCU configurations and diverse application workloads.

Keywords: Self-Organization · Cloud-Edge Continuum · Marginal Cost.

1 Introduction

Edge computing holds the promise of enhancing the performance of applications that require high interactivity and low latency by bringing computation and storage closer to end-users. This paradigm shift provides significant infrastructural support for applications that may struggle with the inherent delays of traditional cloud-based solutions. Edge computing spans a diverse range of infrastructure, from widespread small data centres to localized computational units, from general-purpose hardware to specialized GPUs and various accelerators, and from traditional virtual machines to lightweight WebAssembly (WASM, a complementary technology to Linux containers) containers. This heterogeneity necessitates sophisticated strategies for managing the interactions among all involved entities. The allocation of applications to resources within an edge

computing environment involves complex decisions. These decisions require consensus among various stakeholders and often involve factors beyond straightforward resource matching, such as political, economic, or other considerations. To manage these allocations effectively and ensure scalability, many state-of-the-art solutions avoid relying on a centralized authority [3, 12]. Instead, they employ distributed or decentralized approaches [11, 1, 2, 7, 4]. However, existing solutions often assume equal participation from all stakeholders, which can overlook different actors' unique needs and constraints. The decision to allocate resources is not solely based on matching resource availability with user requests; it also involves evaluating various factors that may not be directly related to monetary considerations. For instance, two parties with identical resources might have different criteria for hosting decisions based on non-technical factors such as political or economic interests. To address these complexities, we propose a new cost function that extends the Marginal Computational Cost of a User (MCU) concept introduced by Ferrucci, Mordacchini, and Dazzi [5]. This extended cost function incorporates the "willingness" of a party to accept additional users. This willingness is modelled as a function concerning the number of users served by a particular "instance" of a given application in a real-world scenario, as explained in Section 4. Employing extensive simulations, we show that the MCU is a good metric to express the "willingness" of an entity to serve additional users. The experimentation is performed by changing the parameters of the function representing such cost, studying the behaviour of a system under variation of such parameters. Specifically, we demonstrate that our approach effectively balances the need to minimize resource usage – such as memory, CPUs, storage, and bandwidth – to reduce the overall cost represented by the MCU function.

The remainder of this paper is subdivided as follows. Section 2 presents the formal definition and concept of the MCU and the associated propositions. Section 3 presents the formal definition of the problem and illustrates the approach we propose. Section 4 describes the experimental methodology, objectives and datasets adopted for assessing the proposed solution, the experimental results achieved, their description and analysis. Finally, Section 5 draws concluding remarks and highlights future work directions.

2 Marginal Computing Cost

In economics, *marginal cost* refers to the change in total cost resulting from the production of one additional unit of a product, essentially the cost of producing one more item [10]. Unlike the *average cost*, which is the total cost divided by the number of units produced, the marginal cost pertains specifically to the cost of the last (marginal) unit produced.

When the cost function C is continuous and differentiable, the marginal cost MC is given by the first derivative of the cost function concerning the quantity of output Q:

$$MC(Q) = \frac{\partial C}{\partial Q} \tag{1}$$

For non-differentiable cost functions, the marginal cost can be defined as:

$$\overline{MC}(Q) = \frac{\Delta C}{\Delta Q} \tag{2}$$

where Δ signifies a change of one unit.

Marginal Computing Cost of a User In this paper, we apply the concept of marginal cost to define the *marginal computing cost per user* (MCU). In a previous study [5], we defined MCU simply as the additional computational resources required by a service to accommodate one more user. In this study, MCU is designed to represent the willingness of an Edge data centre to accept additional users rather than the actual computational cost incurred. This approach allows for a more flexible and realistic resource allocation modelling in edge computing environments.

Definition 1. *The marginal computing cost per user for an application A, currently serving k users, measures the willingness to serve x more users in the form of additional resources needed. It is denoted as* $MCU_{A,k,x}$.

This metric aids in analyzing the resource dynamics of applications. If $TC_{A,k}$ is the total cost for application A serving k users, the average cost per user is defined as:

$$AvgU_{A,k} = \frac{TC_{A,k}}{k} \tag{3}$$

Based on these definitions, we can assess whether serving additional users (and, thus, the willingness to do it) with the existing application instances or to create new instances is beneficial. This depends on the relationship between $AvgU_{A,k}$ and $MCU_{A,k,1}$. We define $\Delta_{cost(A,k,1)}$ as the difference between the average cost and the marginal computing cost when adding one user:

$$\Delta_{cost(A,k,1)} = AvgU_{A,k} - MCU_{A,k,1} \tag{4}$$

$\Delta_{cost(A,k,1)}$ indicates the advantage of serving one more user with an existing application instance. Therefore, it drives the decision-making process of an Edge data centre. In fact, if this value is positive, allocating additional users to the existing instance is beneficial. In case of a request to add x additional users to k already being served, the decision process can be described as a rule of thumb:

$$\Delta_{cost(A,k,x)} \implies \begin{cases} \text{Allocate additional users} & \text{if } > 0, \\ \text{Reject the users} & \text{if } <= 0, \end{cases} \tag{5}$$

3 Problem Definition and Proposed Solution

This paper explores the trade-offs that arise from different collaboration schemes in a decentralized Edge system. Specifically, we analyze the impact on system

performance when Edge data centres exhibit varying degrees of collaboration. We assume that Edge data centres host a heterogeneous set of application instances. Users are initially assigned to their nearest Edge data centre to minimize latency, which is the primary factor influencing users' Quality of Experience (QoE). Each application has specific latency limits that constrain the eligible Edge data centres for serving its users.

Algorithm 1 Actions performed by a generic Edge E_i at a time step t

Input: $\mathcal{N}$ = set of neighbors of E_i

Randomly choose a neighbor E_j from $\mathcal{N}$
Let $\tilde{A}$ be the set of apps having instances in both E_i and E_j
if $\tilde{A} \neq \emptyset$ **then**
  if $W_i \geq W_j$ **then**
    Let $\tilde{A}_l = \{A_k \in \tilde{A} \mid l(u_{i,k}, E_j) \leq l_k\}$
    Let $\mathcal{A} = \{A_k \in \tilde{A}_l \mid MCU_{A_k,u_{jk},u_{ik}} + w_j \leq W_j\}$
    if $\mathcal{A} \neq \emptyset$ **then**
      $A_m = \max_{A_k \in \mathcal{A}} \Delta_{cost(A_k,u_{jk},u_{ik})}$
      Direct the users of u_{ik} to use the instance on E_j
      Turn off the instance on E_i
    end if
  else
    Let $\tilde{A}_l = \{A_k \in \tilde{A} \mid l(u_{j,k}, E_i) \leq l_k\}$
    Let $\mathcal{A} = \{A_k \in \tilde{A}_l \mid MCU_{A_k,u_{ik},u_{jk}} + w_j \leq W_j\}$
    if $\mathcal{A} \neq \emptyset$ **then**
      $A_m = \max_{A_k \in \mathcal{A}} \Delta_{cost(A_k,u_{ik},u_{jk})}$
      Direct the users of u_{jk} to use the instance on E_i
      Turn off the instance on E_j
    end if
  end if
end if

Neighboring Edge data centres collaborate by exchanging groups of users of their running applications, reducing the number of active instances. This approach reduces overall energy consumption and frees resources to host new users and applications. Our proposed algorithm dynamically adjusts user allocation based on real-time resource consumption evaluations at each Edge data centre. However, each Edge data centre can limit its level of collaboration by setting constraints on its willingness to accept new users, measured by the MCU function. An Edge data centre will accept additional users from a neighbour only if the resulting MCU does not exceed its resource limits. In doing this, we do not use the MCU as a direct measure of the computational cost but rather as an indicator of the Edge data centre's willingness to accommodate additional users. This perspective enables dynamic adjustments based on resource availability and collaboration policies among Edge data centres.

This self-organizing behaviour is implemented using the steps described in Algorithm 1. We define a function $l(u_{hk}, E_a)$ that checks if the latency limits l_k of an application A_k are met for all the members of its subset h of users u_{hk} when assigned to Edge data centre E_a; w_a represents the current resource occupancy of E_a, while W_a is its maximum capacity. In Algorithm 1, an Edge E_i randomly selects a neighbor E_j (line 2 of Algorithm1). They exchange information to identify the set $\tilde{A}$ of applications with active instances on both. They compare their occupancies, w_i and w_j. The Edge with lower occupancy checks if it can host additional users. If E_j has lower occupancy, it calculates the MCU for all applications on E_i whose users can be moved to E_j without violating latency limits (lines 6-7). If any application in $\tilde{A}$ satisfies $MCU_{A_k,u_{jk},u_{ik}} + w_j \leq W_j$, the users on E_i of $A_k \in \tilde{A}$ that maximises the Δ_{cost} function are redirected to E_j (lines 9-10), and the instance on E_i is turned off (line 11). The roles are reversed if E_i has lower occupancy.

4 Experimental setup and evaluation

In this section, we present the results of our study with different MCUs using PureEdgeSim [9], a simulator based on CloudSimPlus [6] that supports various Edge-Cloud scenarios. PureEdgeSim allows the simulation of Edge devices, their characteristics, and the modelling of user-generated requests submitted to these devices.

For the geographic placement of Edge data centres, we used the EUA Dataset[4] from Lai *et al.* [8]. This dataset includes location information for Edge resources from the Australian Communications and Media Authority (ACMA), precisely the positions of cellular base stations in Melbourne. The dataset fixes the number of Edge data centres at 125, each covering a range of 450 to 750 meters in a simulation area of about 1.71 km^2.

To ensure users are initially served by an application instance meeting their QoE requirements, we distributed applications and users uniformly. Each Edge data centre has the same resources, allowing us to study various scenarios by changing application types and MCU functions, extending the study by Ferrucci, Mordacchini, and Dazzi [5].

For application footprints, we used the Alibaba cluster-trace-microservices-v2021 dataset[5], containing runtime metrics of microservices. We applied the k-means algorithm to cluster these microservices into eight reference applications. This process ensures the selected applications represent different resource footprints in a modern IDC. The following assumptions apply to our model:

1. Any application type can run on any Edge data centre.
2. There is at most one instance of each application type per Edge data centre.
3. All Edge data centres can communicate and be reached by any user.

[4] https://github.com/swinedge/eua-dataset
[5] https://github.com/alibaba/clusterdata

The resources simulated include $VCPU$s and RAM, representing the number of Virtual CPUs and memory required by an application type. Each Edge data centre has a fixed capacity of 36 VCPUs and 48 Gbytes of RAM, based on a ratio of the capacity used in Alibaba's cluster traces. In our experiments, we extracted eight application types from the Alibaba dataset. Each application type A_i is characterised by a fixed cost $C^{fix}_{A_i}$ (resources needed to start an instance for a single user) and a variable cost C^{var} (for each additional user), which corresponds to the $MCU_{A_i,k,1}$. We used a uniform distribution of applications and users to ensure that each user is initially served by an application instance that meets their QoE requirements. This approach randomly places the same number of users within the range of each Edge data centre, which then serves them. These users are further uniformly distributed across different application types. Each Edge data centre has the same amount of resources, allowing us to study the behaviour of different application types and MCU functions in various scenarios, building on the study in [5].

Starting from the work in [5], in this paper we have defined a new different variable Marginal Computing cost function, in addition to the constant functions evaluated in the previous work, giving the function $MCU_{A_i,1,1} = 1/8 * C^{fix}_{A_i}(R)$, 12.5% of application startup cost, where R is the resource type (VCPU or RAM). Such function is a **polynomial** Marginal computing cost function concerning the number of yet served users on a specific Edge data centre and is defined by the following formula:

$$MCU_{A_i,k,1} = MCU_{A_i,1,1} + MCU_{A_i,1,1} * c * k^{Exp} \tag{6}$$

where c is a predefined coefficient, which represents the growth rate of the Marginal cost function. In our experiments, c is in $\{0, 0.05, 0.1\}$, where $c = 0$ represents a constant MCU function. Exp represents a predefined constant exponent that allows the study of non-linear Marginal cost functions in future works.

Table 1 shows the different application types and the corresponding fixed costs. Our experiments simulated a scenario with three users generating tasks of a given application type on each Edge data centre. Given eight application types, we have $8 * 125 * 3 = 3000$ users. Given the different MCU functions studied and defined for the tests, we considered a total of 3 different scenarios: a scenario with a constant MCU function and two scenarios with linear MCU functions, with respectively $c = 0.05$ and $c = 0.1$.

Another parameter considered in our simulations is the function used to simulate and measure the real-time latency, that is, the simulated time, in milliseconds, to send a message over the communication channel, either between a user (its mobile device, actually) and an Edge data centre or between two Edge data centres. Such function is the following:

$$f_{latency}(d, Edgedatacenter) = ChanLat_{fix} + dist(d, Edgedatacenter) * C_{Lat}$$

The function is composed of two parts:

Table 1: $C_{A_i}^{fix}(R)$ for each application type, for 8 apps tests

Type	VCPU	Ram (Mbyte)
A_1	0.318	898
A_2	0.08	763
A_3	0.174	833
A_4	0.057	466
A_5	0.153	1023
A_6	0.195	646
A_7	0.077	609
A_8	0.082	909

- a fixed part, $ChanLat_{fix}$, which is dependent on the communication channel type; in our experiments, it is fixed at $ChanLat_{fix} = 0.02$ seconds and also includes the component of the latency that depends on the bandwidth of the channel and the dimension of the packet sent;
- a linear part, proportional to the Euclidean distance $dist(d,\ Edgedatacentre)$ between the Edge data centre hosting the instance of the serving application and the user's device or Edge data centre d. The C_{Lat} predefined constant represents the latency cost for each unit of distance and is fixed in our tests at $C_{Lat} = 0.00006$ seconds.

Given the predefined values for $ChanLat_{fix}$ and C_{Lat} and the minimum and maximum distance of a user from the closest base station, we can obtain a range for the latency belonging to the interval $Lat = [20,\ 42.5]$ milliseconds inside a 5G cell, which is typical for a cell of such cellular network generation. For each type of application, we also specified a maximum latency of the network link that has been chosen *a priori* randomly, in the range $MaxLinkLat = [0.07,\ 0.1]$ seconds.

Our simulation is divided into iterations that occur at discrete time intervals. Each Edge data centre behaves like an agent that initiates the algorithm once per iteration. In our scenario, an iteration is started every 30 seconds, and every experiment has a simulated duration time of 15 minutes, so the number of iterations is fixed to 29. All the various experiment parameters (number of users, applications, datasets, latency calculation, users and Edge data center distributions, duration of the experiment, etc.) are chosen to allow a direct comparison with similar experiments in our previous paper [5].

4.1 Experimental Evaluation: Results

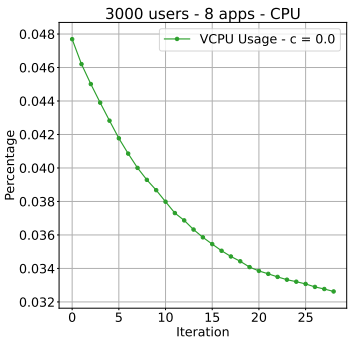
In this section, we present the experimental evaluation results of our proposed approach, which focuses on the variable MCU presented in 6. This approach introduces variability in the cost associated with exchanging users, which impacts the overall resource management and latency within the system. Unlike the fixed MCU used in previous works, this dynamic approach provides a more realistic simulation of the varying costs in real-world scenarios: such *costs* transcend the

simple monetary value or occupancy of resources and represent the willingness of the party to add new users to a computation on a certain Edge data centre, exchanging them from another Edge data centre as explained in Section 3. If the *costs* of hosting such users on a unique Edge data centre are more significant than the *costs* to maintain such users on the other Edge data centre (expressed by the MCU of the exchanged users), the exchange is avoided, leading to a reduction in the total costs of the system and, consequently, also a better load balancing in the workload of the Edge data centres.

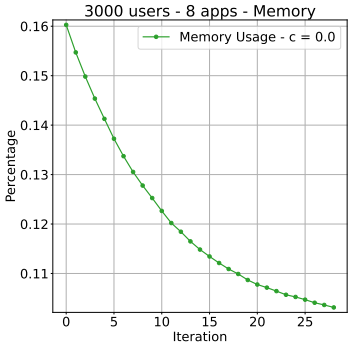
This section comprises three parts: the dynamics of active instances, resource footprint, and latency. For the baseline comparison, we refer to the previous work [5] when $c = 0$ and $Exp = 1$. Our experiments will consider 3000 users, 8 applications, and varying values of c , while Exp is fixed to 1, leading to different linear Marginal computing cost functions. In future work, further tests with a value of $Exp > 1$ will be performed. We limited the number of users to 3000 due to the computational complexity of the simulator.

Dynamics of the Active Instances The dynamics of the active instances of the applications hosted by the edge infrastructure are depicted in subfigures (c) of Figures 1, 2, and 3. When $c = 0$ and $Exp = 1$ (linear MCU), the graph demonstrates a gradual reduction in the number of active instances from an initial value of 1.0 to approximately 0.5 by iteration 25, consistent with the findings of the baseline. This indicates that the system efficiently consolidates users into fewer instances, optimising resource usage. In contrast, for $c = 0.05$ and $Exp = 1$, the number of active instances starts at 1.0 and decreases to around 0.95 by iteration 25. The higher c value results in a less significant reduction as the cost of moving users increases. When $c = 0.1$ and $Exp = 1$, the active instances remain almost stable around 0.98, showing minimal consolidation due to the increased cost sensitivity. These results suggest that while our approach effectively reduces the number of active instances, the impact of the cost factor c plays a significant role in determining the extent of optimising the total costs, in accordance to the willingness expressed by the operator in the MCU function.

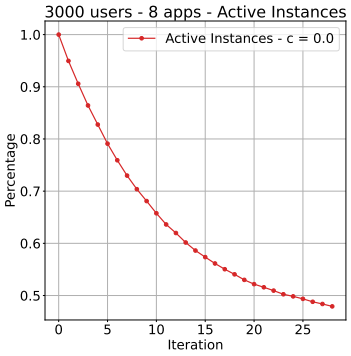
Resource Footprint The impact of varying MCU on the overall resource footprint of the system is shown in subfigures (a) and (b) of Figures 1, 2, and 3. For $c = 0$ and $Exp = 1$, memory usage decreases significantly from 0.16 to 0.12 by iteration 25, consistent with the baseline. This reduction reflects efficient memory consolidation with a linear MCU and no additional cost factor. However, for $c = 0.05$ and $Exp = 1$, memory usage starts at 0.13 and reduces slightly to around 0.125. The presence of a cost factor limits the level of consolidation, as some memory usage remains necessary to avoid excessive costs from exchanging users. When $c = 0.1$ and $Exp = 1$, memory usage remains stable at around 0.126 with minimal reduction, indicating that higher costs associated with exchanging users significantly restrict memory optimisation but lead to an increase in load balancing and, as expressed by the MCU function itself, a reduction in total



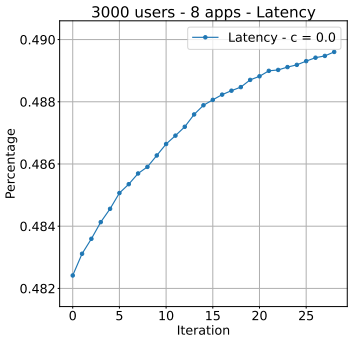
(a) VCPU Usage



(b) Memory Usage



(c) Active Instances

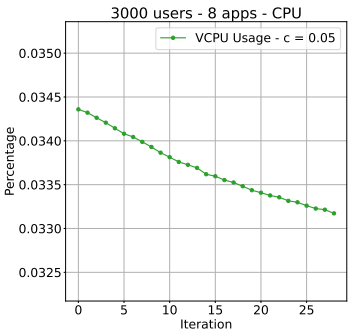


(d) Latency

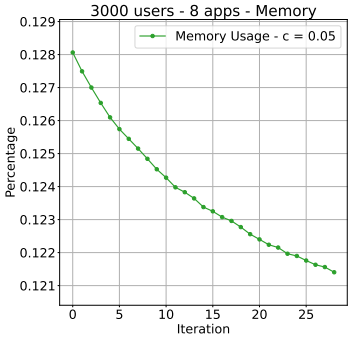
Fig. 1: Dynamics for $c = 0$ and $Exp = 1$ - Baseline from previous work [5]

costs for the party. Notably, the CPU usage trends closely mirror memory usage across the different configurations. As the cost factor increases, the extent of optimisation diminishes similarly for both CPU and memory, highlighting the consistent impact of the dynamic MCU function on resource consolidation. As for the dynamics of the active instances, the presence of a cost factor c crucially determines the given optimisation, helping parties to express concisely the trade-off between merely resource optimisation and the actual costs represented by the MCU function.

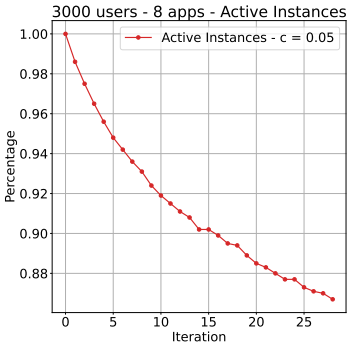
Latency The latency dynamics are depicted in subfigures (d) of Figures 1, 2, and 3, highlighting how latency is affected by the different MCU configurations. For $c = 0$ and $Exp = 1$, latency slightly decreases from 0.49 to around 0.48, showing that latency constraints are well-maintained despite user consolidation, consistent with the baseline. For $c = 0.05$ and $Exp = 1$, latency remains relatively stable, starting at 0.485 and ending at around 0.483. The system effec-



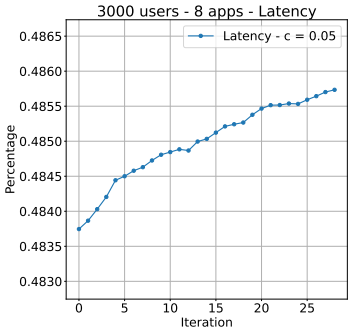
(a) VCPU Usage



(b) Memory Usage



(c) Active Instances



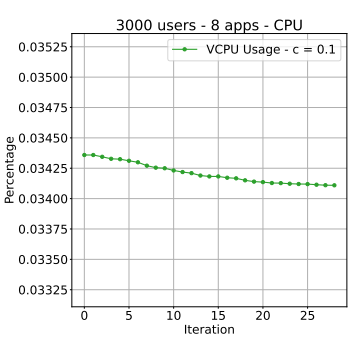
(d) Latency

Fig. 2: Dynamics for $c = 0.05$ and $Exp = 1$

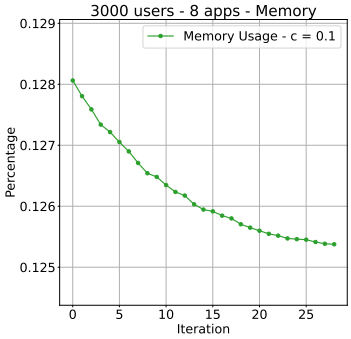
tively manages latency while moderately consolidating users. When $c = 0.1$ and $Exp = 1$, latency shows minimal variation, staying around 0.484. The high cost of exchanging users ensures latency remains stable but limits optimisation benefits. These results demonstrate that our approach effectively maintains latency constraints even when higher costs limit resource consolidation.

5 Conclusions

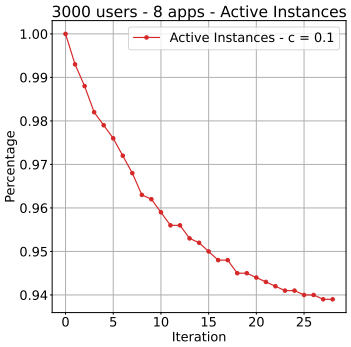
This paper presented extended cost functions to model the willingness of edge data centres to accept additional users in a decentralized edge computing environment. The marginal computing cost per user (MCU) concept was expanded to incorporate a dynamic cost factor based on the number of users served. Extensive simulations were conducted using the PureEdgeSim platform to evaluate the impact of this variable MCU on system performance under different parameter configurations. The results demonstrate that our approach effectively



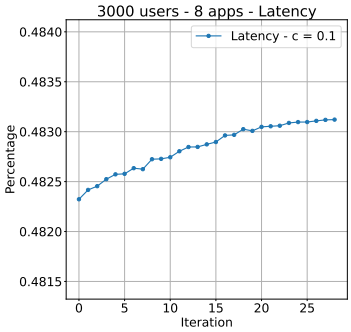
(a) VCPU Usage



(b) Memory Usage



(c) Active Instances



(d) Latency

Fig. 3: Dynamics for $c = 0.1$ and $Exp = 1$

balances the dual objectives of optimizing resource usage through user consolidation, while also reducing the overall cost as represented by the MCU function. When the MCU exhibited linear growth, systems with higher growth coefficients showed less significant reductions in active instances and resource footprint over time. This indicates that a higher cost sensitivity inhibits the extent of optimization. Notably, latency constraints were well-maintained even for scenarios with limited consolidation due to higher costs. This work provides insights into how modelling the flexible willingness of edge data centres via a variable MCU can influence the collaborative user allocation process in decentralized edge systems. It also highlights the trade-off between resource optimization on one hand, and costs expressed by the MCU on the other. Future work will explore non-linear MCU configurations and different types of real-world application workloads and infrastructure topologies to continue assessing this approach under varied conditions. Overall, the proposed solution presents a viable strategy for managing heterogeneous edge systems with self-interested participants.

Acknowledgments. This project has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101092950 (EDGELESS project).

References

1. Carlini, E., Coppola, M., Dazzi, P., Ferrucci, L., Kavalionak, H., Korontanis, I., Mordacchini, M., Tserpes, K.: SmartORC: smart orchestration of resources in the compute continuum. Frontiers in High Performance Computing **1**, 1164915 (2023)
2. Carlini, E., Coppola, M., Dazzi, P., Mordacchini, M., Passarella, A.: Self-optimising decentralised service placement in heterogeneous cloud federation. In: 2016 IEEE 10th International Conference on Self-adaptive and Self-organizing Systems (SASO), pp. 110–119 (2016)
3. Ferrer, A.J., Becker, S., Schmidt, F., Thamsen, L., Kao, O.: Towards a cognitive compute continuum: an architecture for ad-hoc self-managed swarms. In: 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), pp. 634–641 (2021)
4. Ferrucci, L., Mordacchini, M., Coppola, M., Carlini, E., Kavalionak, H., Dazzi, P.: Latency Preserving Self-Optimizing Placement at the Edge. In: Proceedings of the 1st Workshop on Flexible Resource and Application Management on the Edge. FRAME '21, pp. 3–8. ACM, Sweden (2021)
5. Ferrucci, L., Mordacchini, M., Dazzi, P.: Decentralized Replica Management in Latency-Bound Edge Environments for Resource Usage Minimization. IEEE Access **12**, 19229–19249 (2024). https://doi.org/10.1109/ACCESS.2024.3359749
6. Filho, M.C.S., Oliveira, R.L., Monteiro, C.C., Inácio, P.R.M., Freire, M.M.: CloudSim Plus: A cloud computing simulation framework pursuing software engineering principles for improved modularity, extensibility and correctness. In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), pp. 400–406 (2017)
7. Korontanis, I., Tserpes, K., Pateraki, M., Blasi, L., Violos, J., Diego, F., Marin, E., Kourtellis, N., Coppola, M., Carlini, E., *et al.*: Inter-operability and orchestration in heterogeneous cloud/edge resources: The accordion vision. In: Proceedings of the 1st Workshop on Flexible Resource and Application Management on the Edge, pp. 9–14 (2020)
8. Lai, P., He, Q., Abdelrazek, M., Chen, F., Hosking, J.G., Grundy, J.C., Yang, Y.: Optimal Edge User Allocation in Edge Computing with Variable Sized Vector Bin Packing. In: ICSOC (2018)
9. Mechalikh, C., Takta, H., Moussa, F.: PureEdgeSim: A Simulation Toolkit for Performance Evaluation of Cloud, Fog, and Pure Edge Computing Environments. In: 2019 International Conference on High Performance Computing Simulation (HPCS), pp. 700–707 (2019)
10. O'Sullivan, S.M.: Economics: Principles in Action. Pearson Prentice Hall, Upper Saddle River, NJ (2003)
11. Russo, G.R., Mannucci, T., Cardellini, V., Presti, F.L.: Serverledge: Decentralized function-as-a-service for the edge-cloud continuum. In: 2023 IEEE Int. Conf. on Pervasive Computing and Communications (PerCom), pp. 131–140 (2023)
12. Surya, K., Rajam, V.M.A.: Novel Approaches for Resource Management Across Edge Servers. International Journal of Networked and Distributed Computing **11**(1), 20–30 (2023). https://doi.org/10.1007/s44227-022-00007-0