Multitiered Worker-Oriented Resource Allocation at the Extreme Edge

Marah De'bas, Sara A. Elsayed, Hossam S. Hassanein
School of Computing, Queen's University, Kingston, ON, Canada
marah.debas@queensu.ca, selsayed@cs.queensu.ca, hossam@cs.queensu.ca

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10000855

***Abstract*—Fostering Edge Computing (EC) by recycling prolific yet underutilized computational resources of the Internet of Things (IoT) devices, also referred to as Extreme Edge Devices (EEDs), has gained significant momentum lately. Fair resource allocation is a primary concern in such computing paradigms. However, fairness is typically considered from the requester's perspective, whereas fairness for workers (i.e., EEDs) is mostly overlooked. In this context, we propose the Multitiered Worker-Oriented Resource Allocation (MWORA) scheme. In MWORA, the resource allocation problem is formulated as an Integer Linear Program (ILP). MWORA aims to maximize service capacity and minimize the task response delay while enabling fair resource allocation that maintains a specific satisfactory profit for workers. Such a satisfactory level is maintained to prevent the workers from leaving the system and ensure their recurrent subscription to the service. This is done while abiding by the deadline demanded by each requester and without exceeding a certain budget. MWORA also accounts for the fact that EEDs are user-owned devices and are thus subject to a dynamic user access behavior, which can affect the level of computational resources that workers are willing to offer. In particular, MWORA enables multitiered computational resources to be granted by each worker depending on the price of the allocated task. Extensive simulations have shown that MWORA outperforms other baseline resource allocation schemes regarding average response delay, service capacity, worker satisfaction ratio, and fairness.**

***Index Terms*—Edge Computing, Extreme Edge, EEDs, Fair Resource Allocation, Optimization**

I. Introduction

With the promulgation of the Internet of Things (IoT), it is estimated that 23.3 billion IoT devices will be connected to the Internet by 2025 [1]. This unprecedented surge is expected to impose severe demands on computing resources to meet the rigorous Quality of Service (QoS) requirements associated with latency-sensitive and/or data-intensive IoT applications [2]. Satisfying such requirements can be challenging in cloud computing due to the substantial amount of data that must be fully transmitted to distant data centers, which increases the delay and creates a heavy traffic load at backhaul links [3].

Edge Computing (EC) has paved the way toward mitigating the issues mentioned above by moving the computing service closer to end users [2]. However, most existing EC platforms and models rely on infrastructure-based edge nodes exclusively controlled by cloud service providers and/or network operators [4]. Breaking this monopoly by recycling prolific yet underutilized computational resources of Extreme Edge Devices (EEDs), such as tablets, smartphones, and autonomous vehicles, can democratize the edge and enable more players to develop and manage their own edge cloud. This can foster a new tech market that is people-retained, democratically controlled, and accessible/rewarding. In addition, parallel processing at EEDs can bring the computing service closer to end-users, significantly reducing the delay [5]. Considering its promising impact, fair resource allocation is crucial in such computing paradigms [5].

Aside from load balancing approaches [6], most existing resource allocation schemes in EC have considered fairness from the requesters' perspective [7]–[11] while overlooking workers fairness. In EED-enabled computing environments, overlooking workers fairness regarding achieving a satisfactory profit can cause such workers to leave the system, avoid recurrent subscription to the service, and join other competing service providers (i.e., service facilitators acting as mediators between requesters and workers, such as [12]). In the long run, this can affect the yielded QoS since the available resources might not be able to cope with the load imposed by incoming requests.

In this paper, we propose the Multitiered Worker-Oriented Resource Allocation (MWORA) scheme to ensure workers fairness. MWORA aims to minimize the task response delay and maximize service capacity while sustaining a certain level of satisfactory profit demanded by each worker. This is while abiding by a certain budget specified by the service provider and a certain deadline indicated by each requester. In addition, in contrast to existing schemes, the effect of offloading the tasks to user-owned devices that have a dynamic user access behavior is accounted for in MWORA.

MWORA considers the impact of the dynamic user access behavior on the level of computational resources that the workers are willing to offer, which can differ based on the cost of compromising their own convenience. Take, for instance, the scenario in which a user is streaming a video on their device. In this scenario, the worker (user's device) could be willing to forego the video by pausing and devoting their maximum computational capability to an offloaded task if it is worth a high financial reward. Alternatively, if the reward is moderate, the worker could continue streaming the video and devote less computational capability to the offloaded task. Note that the worker here is sacrificing a smaller portion of their capabilities to preserve some of its own convenience. If the task's reward is too low, the worker could refrain from

giving up any of their resources. Thus, in contrast to existing schemes, MWORA fosters granting multitiered computational capabilities by each worker based on the financial reward associated with the offloaded task.

To the best of our knowledge, MWORA is the first scheme that solicits multitiered computational capabilities and aims to achieve fair resource allocation for workers by ensuring that each worker receives a certain demanded reward. The resource allocation problem is formulated as an Integer Linear Program (ILP) and is solved using the Gurobi optimizer [13].

The remainder of the paper is structured as follows. Section II provides an overview of some related work. Section III presents our proposed scheme (MWORA). Section IV discusses the performance evaluation and simulation results. Finally, Section V presents our conclusions and future work.

II. Related Work

Resource allocation is a technique used to allocate computing resources to various processes, threads, data flows, and programs/applications [14]. Fair resource allocation is required to balance the load on the system, ensure fair distribution of resources, and give priority according to the set of defined rules in the implemented scheduling algorithm while allocating the available resources [6], [14]. In general, fair resource allocation strives to guarantee the capability of a system to serve all requests and achieve specific QoS measures [6]. In [15], it has been shown that 23% of categorized resource allocation schemes aim to improve the response delay, as the case in [16], 5% aim to improve the execution delay. The remaining 72% are distributed among the remaining measures of QoS, including the recruitment cost [17].

Aside from load balancing schemes [6], existing research efforts have mostly considered fair resource allocation from the requester's perspective [7]–[11]. In [7], the authors aim to minimize the maximum server delay in a multi-user and multi-server system in Multi-access Edge Computing (MEC) by fairly selecting the appropriate server on which the requester can offload tasks, which minimizes the overall delay of the system. In [8], a multi-objective optimization problem formulation aims to minimize the delay, energy consumption, and cost paid to the MEC service provider. A study on minimizing delay and energy consumption is also proposed in [9] by partitioning the task and offloading it to multiple devices in MEC. The work in [10] partitions the task into portions and aims to minimize the sum difference between the actual delay taken and the desired delay of each portion. In [11], the authors present a resource allocation scheme that maximizes the number of tasks executed within their allowed deadline while simultaneously ensuring fairness by prioritizing the tasks and maintaining high network stability.

Contrary to our work, the existing schemes do not consider that workers may be user-owned devices, which can dynamically impact their available computational capability. In other words, they do not account for the associated dynamic user access behavior and its implications on the user's convenience and the level of computational capability that the worker is willing to share. They also fail to consider the impact of workers' fairness on their availability and willingness to cooperate with the service provider if they are not receiving a satisfactory monetary reward.

III. Multitiered Worker-Oriented Resource Allocation (MWORA)

In MWORA, the system consists of user-owned devices operating as workers, a set of requesters with tasks that need to be offloaded for execution, and a Service Provider (SP) functioning as the centralized entity responsible for making resource allocation decisions in the system. The SP has a coverage area, a set of workers operating within this coverage area, and a set of tasks received from the requesters. The SP is responsible for allocating the tasks to the participating workers within given constraints and limitations.

A. System Model:

Given a set of workers $\mathrm{W} = \{w_1, w_2, \dots w_m\}$, a set of requesters $\mathrm{U} = \{u_1, u_2, \dots u_n\}$, and a set of tasks $\mathrm{T} = \{t_1, t_2, \dots t_n\}$ at any given time. Each task t_i has a certain computation workload or intensity q_i (in CPU cycles/bit) and involves a certain amount of data in bits, denoted L_i and should be executed within a specified deadline, denoted by ϵ_i. The distance between the requester u_i and worker w_j is denoted by d_{ij}, R_{ij} denotes the data rate of the link between u_i and w_j, and the propagation speed is denoted by v. The SP has a specific budget β that is used to recruit the available workers. This budget should not be exceeded.

Each worker w_j specifies a certain price τ_j that it should be paid per executing one CPU cycle. Also, each worker w_j specifies a minimum profit that it should gain to be deemed satisfied (i.e., a satisfactory reward), denoted s_j. Such a satisfactory reward acts as an indicator of workers satisfaction by the SP. One of SP's primary functions when assigning tasks to workers is to assign one or more tasks to a worker capable of completing each assigned task within the task deadline ϵ_i. At the same time, the assigned tasks must render each worker's required satisfactory profit s_j without exceeding the budget β.

The cost of executing each task t_i on worker w_j is denoted p_{ij} and is given by Eq. 1:

$$p_{ij} = \tau_j q_i L_i \tag{1}$$

Each worker w_j has a maximum CPU clock speed denoted by $c_j^{\max}$ (in CPU cycles/sec). Each worker w_j is willing to dedicate one of three possible levels of its $c_j^{\max}$ according to high and low-profit thresholds that each worker specifies, denoted by Ω_{jH} and Ω_{jL}, respectively, the CPU cycle frequency that each worker w_j is willing to dedicate to execute task t_i is denoted by c_{ij}, and is given by Eq.2, where $0 \leq \delta \leq 1$.

$$c_{ij} = \begin{cases} \delta c_j^{\max} & \text{if } p_{ij} \geq \Omega_{jH} \\ \alpha_j \delta c_j^{\max} & \text{if } \Omega_{jL} \leq p_{ij} < \Omega_{jH} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Note that $\delta c_j^{\max}$ is the maximum load capacity that can be purchased from worker w_j that ensures that it is not completely

overwhelmed. The capacity level $\alpha_j \delta c_j^{\max}$ reduces the capacity that can be purchased from w_j by a defined factor α_j , where $0 \leq \alpha_j \leq 1$. If the price of executing task t_i on worker w_j, p_{ij}, does not correspond to any of the preceding levels, the worker refrains from devoting any capacity level to task t_i. Therefore, Ω_{jL} of w_j acts as a cut-off point below which the worker does not find it feasible to perform the task.

The response delay associated with executing task t_i on worker w_j, denoted γ_{ij}, is composed of the execution delay, transmission delay, propagation delay, and is given by Eq. 3.

$$\gamma_{ij} = \frac{q_i L_i}{c_{ij}} + \frac{L_i}{R_{ij}} + \frac{d_{ij}}{v} \tag{3}$$

B. Problem Formulation:

The objective is to maximize the service capacity (i.e., the number of executed tasks) and minimize the total response delay. We formulate the problem as an ILP multi-objective problem, where the binary decision variable x_{ij} is set to 1 if task t_i is assigned to w_j, and 0 otherwise, as given by Eq. 4.

$$x_{ij} = \begin{cases} 1 & \text{if } t_i \text{ is assigned to } w_j \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

The problem formulation is shown below:

$$\max_{x_{ij}} \sum_{j \in W} \sum_{i \in T} x_{ij}$$

$$\min_{x_{ij}} \sum_{j \in W} \sum_{i \in T} x_{ij} \gamma_{ij}$$

subject to:

C1: $\sum_{j \in W} x_{ij} \leq 1 \quad \forall i \in T$

C2: $\sum_{j \in W} \sum_{i \in T} x_{ij} p_{ij} \leq \beta$

C3: $\sum_{j \in N_i} x_{ij} = 0 \quad \forall i \in T$

C4: $\sum_{i \in T} x_{ij} c_{ij} \leq \delta c_j^{\max} \quad \forall j \in W$

C5: $\sum_{j \in W} \sum_{i \in T} x_{ij} \gamma_{ij} \leq \epsilon_i$

C6: $\sum_{i \in T} x_{ij} p_{ij} \geq \psi s_j \quad \forall j \in W_j$

Constraint C1 specifies that each task must be assigned to at most one worker; this ensures that each task is assigned only once. Constraint C2 ensures that the total reward gained by all workers from the tasks assigned to them does not exceed the budget specified by the SP. Constraint C3 ensures that no task is assigned to a worker from the set N_i, which is the set of tasks with which the worker has a zero c_{ij}. Constraint C4 ensures that for each worker, the CPU cycle frequency used to execute all the tasks assigned to it does not exceed the maximum load capacity that can be purchased from the worker. Constraint C5 ensures that when assigning a task to a worker, is be executed within the task deadline ϵ_i. Constraint C6 ensures that for each worker, the total reward gained from all the tasks assigned to it is no less than a certain threshold.

IV. Performance Evaluation

This section evaluates the performance of MWORA compared to a baseline resource allocation approach, referred to as Requesters-Oriented Fair Allocation (ROFA). ROFA is a representative of existing resource allocation schemes that focus on fairness in terms of the QoS provided to requesters without considering the satisfaction of workers [7]. Note that ROFA does not support multitiered computational capabilities. In order to show the effect of the multitiered computational capabilities fostered by MWORA, we implement ROFA twice, once with multitiered computational capabilities and once where each worker grants only a single level of computational capabilities. The former approach is referred to as ROFA-MC, and the latter is referred to as ROFA-SC.

In order to show the effect of satisfying workers, we implement MWORA, ROFA-MC, and ROFA-SC over multiple rounds, where only the satisfied workers remain in the system for the next round to execute the incoming tasks. To demonstrate this effect, we show the results of each scheme for the first and last rounds, referred to as MWORA-F and MWORA-L, ROFA-MCF, ROFA-MCL, and ROFA-SCF and ROFA-SCL, respectively.

We use the following performance metrics: 1) the average response delay which is the average time taken to offload and execute the tasks (i.e, the average γ_{ij}, where γ_{ij} is given by Eq. 3), 2) the service capacity ratio, which is the ratio of the number of successfully executed tasks to the total number of tasks, 3) the workers satisfaction ratio, which is the ratio of the number of workers that acquire their desired satisfactory reward s_j to the total number of workers, 4) the workers fairness, which is calculated using Jain's fairness index [18], denoted by F, and is given by Eq. 5, where π_j denotes the actual reward obtained by worker w_j, and F ranges from 0 (best case scenario) to 1 (worst case scenario), and 5) the total energy consumption of workers, which is the sum of the energy consumed by each worker due to task execution, as given by Eq. 6, where μ_j is the on-board CPU capacitance of worker w_j, and λ = 2.

$$F = \frac{\left(\sum_{j=1}^{n} \pi_j - s_j\right)^2}{n \cdot \sum_{j=1}^{n} \left(\pi_j^2 - s_j^2\right)} \tag{5}$$

$$\sum_{j \in W} \sum_{i \in T} x_{ij} \mu_j q_i L_i c_{ij}^{\lambda - 1} \tag{6}$$

A. Simulation Setup

MWORA, ROFA-MC, and ROFA-SC are all implemented using Gurobi [13]. Simulations are performed over $500m \times 500m$ area, where all requesters and workers are uniformly distributed. The total number of tasks is set to 250, and the number of available workers is set to 100. The data size of tasks is uniformly distributed in the range of [5,10] bits. Unless otherwise specified, the computation intensity of tasks ranges between [10, 20] cycles/bit. The propagation speed is set to 120 m/s, and the data rate is uniformly distributed in the range

of [15,30] bits/sec. The maximum computation capability of workers is set in the range of [300, 800] CPU cycles/sec. The price per one CPU cycle that is specified by workers ranges between [0.1, 0.4]. The factor δ is set to 0.9, whereas α_j ranges between [0.1, 0.4]. The satisfactory profit of workers ranges between [100, 250], and Ω_{jH} and Ω_{jL} are set in the range of [90, 250] and [5, 89], respectively.

The budget is set to 2500. The value of ψ was set to 80%. The deadline of tasks ranges between [0.5, 10] seconds. The on-board capacitance of all workers μ_j is set to 10^{-11} , and λ is set to 2.

Simulations are conducted over 5 rounds to show the effect of workers satisfaction. Each round is a new time interval during which the SP assigns incoming tasks to operating workers. Workers who are unsatisfied with their reward leave the system, so the number of operating workers can decrease between successive rounds. To demonstrate this effect, we present the first and last rounds of each scheme.

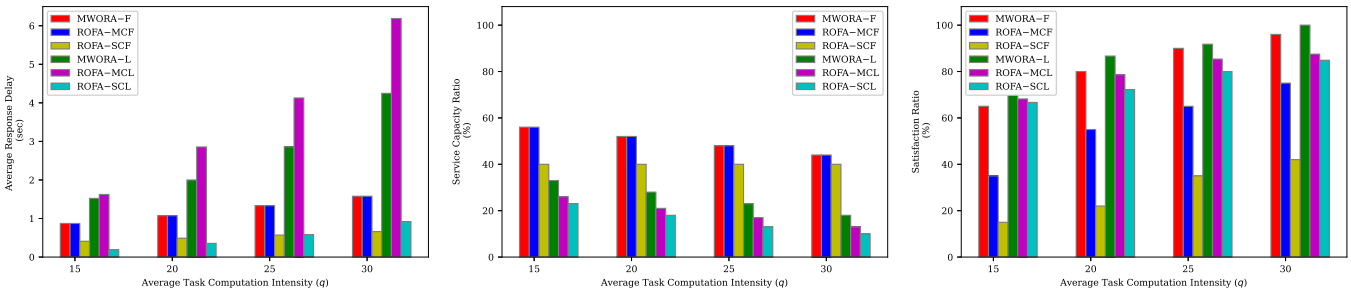
B. Simulation Results and Analysis

Initially, we evaluate the performance of MWORA, ROFA-MC, and ROFA-SC in terms of the average response delay, as shown in Fig. (1a). For the first round, MWORA and ROFA-MC have identical values of the average response delay for the same values of q. This is because the workers in MWORA and ROFA-MC dedicate the same amount of computational capability c_{ij}. For ROFA-SC, as mentioned earlier in section IV, all the workers in the system have a single level of computational capability, dedicating their maximum $\delta c_j^{\max}$ capability. As a result of constraint C4, the workers are assigned a single task at a time. Hence, ROFA-SC provides the lower bound on the achievable average response delay since all workers are exploited at their maximum computational capability. On the other hand, it is seen in Fig. (1a) that MWORA and ROFA-MC do not exhibit identical values for the average response delay in the last round. In fact, MWORA consistently outperforms ROFA-MC in the last round, even though the workers dedicate the same computational capability to these two schemes. This is because MWORA will retain a higher number of workers cooperating with the SP in the long run since it specifically considers workers' satisfaction, unlike ROFA-MC. For ROFA-SC in the last round, it is noted that it provides the lower bound on the achievable average response delay because the workers in ROFA-SC operate at their maximum available computational capability, as mentioned earlier.

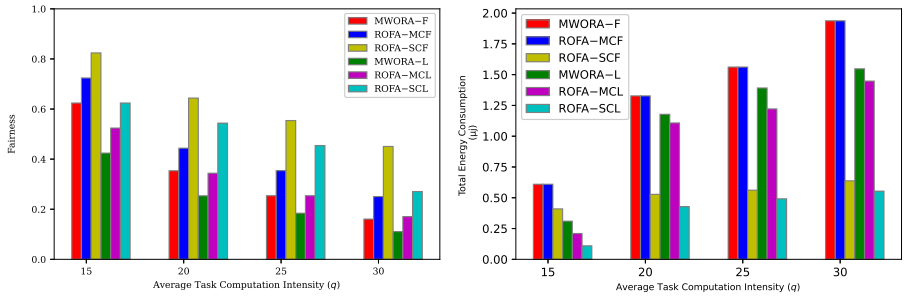
Since the average response delay accounts for the time needed to execute a task, we observe that schemes show an upward trend as the average computational intensity q increases in Fig. (1a). Moreover, MWORA, ROFA-MC, and ROFA-SC exhibit an increase in the average response delay between the first and last rounds. This is because, in the last round, fewer workers are subscribed to the service, while all the workers unsatisfied with their received rewards unsubscribed, leaving fewer workers to handle the incoming task requests and thus incurring higher response delays. Note that the advantage of our MWORA scheme and its consideration of workers' satisfaction is clearly shown by comparing its performance in the last round to ROFA-MC for increasing values of q. Particularly, the performance gap regarding the average response delay grows significantly larger as q increases, with MWORA consistently outperforming ROFA-MC, and this is because, again, MWORA retains a more significant number of workers that are satisfied with their rewards and hence, has more workers to take on the incoming task requests. For q = 30, MWORA shows a 45.7% decrease in the average response delay compared to ROFA-MC.

Second, we examine the service capacity in Fig. (1b). Similar to Fig. (1a), MWORA and ROFA-MC exhibit identical values of the service capacity in the first round since the workers offer identical amounts of computational capability for task execution. Additionally, ROFA-SC consistently underachieves regarding the service capacity in the first round compared to MWORA and ROFA-MC. This is because the workers in ROFA-SC are constrained to perform a single task at a time, which effectively reduces the overall service capacity. On the other hand, MWORA surpasses both ROFA-MC and ROFA-SC in the last round. Again, this is due to the consideration of workers' satisfaction, which increases the number of satisfied workers available to execute more tasks in the long run. The workers in MWORA continue to operate with SP. More importantly, in MWORA, a primary goal is to maximize the service capacity. Hence, workers are assigned more tasks as long as their computational capabilities are not exceeded, as conditioned by constraint C4. Furthermore, by examining Fig. (1b), we note that all schemes display a decrease in the service capacity for increasing values of the computational intensity q in both the first and last rounds of each scheme. This is because when the computational intensity increases, the p_{ij} of task t_i increases, and the budget of the SP will not be enough to execute the same number of tasks in the preceding experiment.

Third, we evaluate the performance of MWORA, ROFA-MC, and ROFA-SC in terms of workers' satisfaction ratio in Fig. (1c). The workers' satisfaction ratio metric highlights the core advantages of our MWORA scheme. As depicted, MWORA offers superior performance in the first round compared to ROFA-MC and ROFA-SC. MWORA exhibits a 50% increase in the workers' satisfaction ratio compared to ROFA-MC and a 76% increase compared to ROFA-SC for q=15. Similar behavior is seen in the last round as well. ROFA-SC offers the lowest worker's satisfaction ratio in the first and last rounds compared to MWORA and ROFA-MC since ROFA-SC workers operate at a single level of computational capabilities. In addition, the workers in ROFA-MC and ROFA-SC execute the assigned task without checking if it renders the worker's required satisfactory profit s_j. Note that the workers' satisfaction ratio is calculated by considering the remaining workers in the system at each round. Consequently, the workers' satisfaction ratio will be higher for all three schemes in the next round than in the preceding round. In addition, the workers' satisfaction ratio increases as the computational intensity increase for the same scheme in the same



(a) Average response delay (b) Service capacity ratio (c) Workers satisfaction ratio



(d) Workers fairness (e) Total energy consumption of workers

Fig. 1: Performance results of MWORA, ROFA-MC and ROFA-SC over varying average task computation intensity q

round, and this is because p_{ij} increases for tasks with higher computational intensity. Hence, workers are rewarded more when performing tasks with higher computational intensity, and their satisfaction levels consequently increase.

Fourth, we investigate the performance of MWORA, ROFA-MC, and ROFA-SC in terms of fairness. In MWORA, workers are assigned tasks in one of two ways. First, the SP assigns the worker a single task by purchasing the higher amount of computational capability $\delta c_j^{\max}$, causing the requested s_j to be met. Alternatively, the SP assigns the worker multiple tasks at the $\alpha_j \delta c_j^{\max}$ level, and in total, the assigned tasks will meet the required s_j. This assignment process depends on the available budget β, p_{ij} of each task and the task's deadline ϵ_i. Besides the fairness from the workers' perspective, the SP also considers the fairness from the requesters' perspective. Constraint C5 guarantees that the provided c_{ij} level by the worker for the task will ensure it is finished within its deadline. Therefore, we should consider both of the above cases to measure fairness. Specifically, we take into consideration whether worker w_j received a reward that meets their specified threshold s_j or an amount that exceeded s_j.

Fig. (1d) shows the fairness index against increasing values of computational intensity. Recall that in fairness Eq. 5, the lower the value, the fairer the system. As seen in Fig. (1d), MWORA surpasses ROFA-MC and ROFA-SC in the achieved fairness in the first and last rounds, for varying levels of q, since MWORA takes the required s_j by workers into consideration. Notably, we note that ROFA-SC shows the poorest performance since workers perform a single task at a time.

Fifth, we study the impact of the total energy consumed by the workers in Fig. (1e). Similar to our observations in Fig. (1a) and (1b), MWORA and ROFA-MC show identical energy consumption levels in the first round since the dedicated computational capability levels are similar. In addition, since in ROFA-SC each worker performs a single task at a time, ROFA-SC has a lower energy consumption than MWORA and its ROFA-MC counterpart in the first and last rounds. It has a significantly lower number of workers at each round than MWORA and ROFA-MC. In the last round, MWORA performs a higher number of tasks than ROFA-MC and exhibits higher energy consumption. This represents the cost incurred by MWORA in order to achieve adequate workers' satisfaction.

Finally, we conclude our analysis by plotting the average response delay for the even number of rounds in Fig. (2). Note that in Fig. (1), we plotted our metrics for the first and last (fifth) rounds. On the other hand, Fig. (2) shows the performance of our proposed scheme in the long run when the average computational intensity q is set to 20. The behavior of the investigated schemes is similar to that shown in Fig. (1a), sustaining an upward trend between progressing rounds. Similarly, MWORA surpasses ROFA-MC, while ROFA-SC exhibits the highest response delay for the above-mentioned reasons. More importantly, Fig. (2) reveals that MWORA reaches a steady level regarding the average response delay. As shown in the figure, MWORA experiences a negligible increase in the response delay after the sixth round, whereas

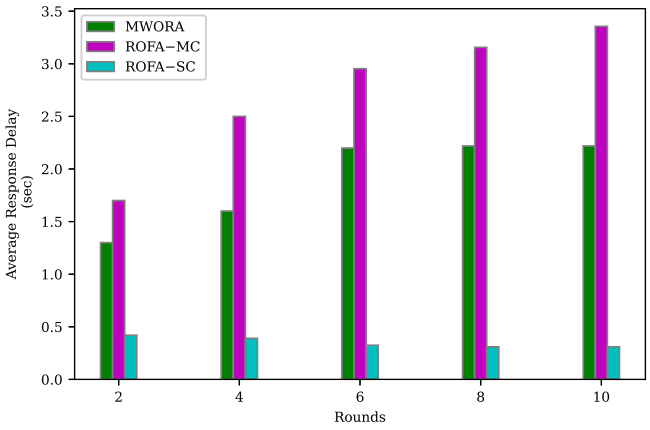


Fig. 2: Average response delay over varying rounds

ROFA-MC continues to show significant growth as rounds progress. Hence, in addition to the demonstrated advantages of our MWORA scheme, an important characteristic is that the system performance stabilizes after a while, and it no longer exhibits an exponentially increasing trend. This is because MWORA sustains a certain number of workers that become guaranteed recurrent subscribers since their satisfaction is always met through the SP's available budget. In other words, considering workers' satisfaction improves workers' retention in the system. In addition, it offers a cap on the average response delay, which benefits the task requesters.

V. Conclusion And Future Work

In this paper, we have proposed the Multitiered Worker-Oriented Resource Allocation (MWORA) scheme. MWORA leverages the prolific yet underutilized computational resources of EEDs. This can help democratize the edge and enable more players to build and manage their own edge cloud. In contrast to existing schemes, MWORA considers that EEDs tend to be user-owned devices, and are thus subject to a dynamic user access behavior, which can dynamically impact their computational capabilities and introduce a human factor related to preserving users' convenience. To address this issue, MWORA fosters multitiered computational capabilities that can be granted by workers, compliant with the level of their own convenience that they are willing to sacrifice based on the corresponding profit of the offloaded task. In addition, MWORA achieves fair resource allocation from the worker's perspective by maintaining the demanded satisfactory profit. This can encourage workers to remain in the system and recurrently donate their computational services, which can significantly impact the QoS in the long run. Extensive simulations have shown that MWORA achieves significant improvements compared to other baseline resource allocation schemes, in terms of average response delay, service capacity, worker satisfaction ratio, and fairness. In future work, we plan on using game theory to solve the resource allocation problem.

Acknowledgment

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number: ALLRP 549919-20.

References

[1] GSMA, "The mobile economy 2022." Accessed on 2022, May. [Online]. Available: https://www.gsma.com/mobileeconomy/wp-content/uploads/2022/02/280222-The-Mobile-Economy-2022.pdf

[2] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the internet of things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.

[3] T. H. Noor, S. Zeadally, A. Alfazi, and Q. Z. Sheng, "Mobile cloud computing: Challenges and future research directions," *Journal of network and computer applications*, vol. 115, pp. 70–85, 2018.

[4] L. Peterson, T. Anderson, S. Katti, N. McKeown, G. Parulkar, J. Rexford, M. Satyanarayanan, O. Sunay, and A. Vahdat, "Democratizing the network edge," *ACM SIGCOMM Computer Communication Review*, vol. 49, pp. 31–36, 05 2019.

[5] 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, 2016.

[6] M. Kaur and R. Aron, "A systematic study of load balancing approaches in the fog computing environment," *The Journal of supercomputing*, vol. 77, no. 8, pp. 9202–9247, 2021.

[7] S. Xiao, C. Liu, K. Li, and K. Li, "System delay optimization for mobile edge computing," *Future Generation Computer Systems*, vol. 109, pp. 17–28, 2020.

[8] P. Wang, K. Li, B. Xiao, and K. Li, "Multi-objective optimization for joint task offloading, power assignment, and resource allocation in mobile edge computing," *IEEE internet of things journal*, pp. 1–1, 2021.

[9] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. A. Salinas Monroy, "Multi-objective computation sharing in energy and delay constrained mobile edge computing environments," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 2992–3005, 2021.

[10] S. Zarandi and H. Tabassum, "Delay minimization in sliced multi-cell mobile edge computing (mec) systems," *IEEE Communications Letters*, vol. 25, no. 6, pp. 1964–1968, 2021.

[11] M. Mukherjee, M. Guo, J. Lloret, R. Iqbal, and Q. Zhang, "Deadline-aware fair scheduling for offloaded tasks in fog computing with inter-fog dependency," *IEEE Communications Letters*, vol. 24, no. 2, pp. 307–311, 2020.

[12] KDS, "Kings distributed systems." Accessed on 2022, May. [Online]. Available: https://kingsds.network/

[13] Gurobi, "Gurobi optimizer reference manual." Accessed on 2022, May. [Online]. Available: https://www.gurobi.com/

[14] C.-H. Hong and B. Varghese, "Resource management in fog/edge computing: A survey on architectures, infrastructure, and algorithms," *ACM Comput. Surv.*, vol. 52, no. 5, sep 2019.

[15] M. Haghi Kashani, A. M. Rahmani, and N. Jafari Navimipour, "Quality of service-aware approaches in fog computing," *International Journal of Communication Systems*, vol. 33, no. 8, p. e4340, 2020, e4340 IJCS-19-0006.R2.

[16] R. F. ElKhatib, S. A. ElSayed, N. Zorba, and H. S. Hassanein, "Optimal proactive resource allocation at the extreme edge," *2022 IEEE International Conference on Communications (ICC): IoT and Sensor Networks Symposium (IEEE ICC'22 - IoTSN Symposium)*, 2022.

[17] I. M. Amer and S. Sorour, "Cost-based compute cluster formation in edge computing," *2022 IEEE International Conference on Communications (ICC): IoT and Sensor Networks Symposium (IEEE ICC'22 - IoTSN Symposium)*, 2022.

[18] R. Jain, D. M. Chiu, and H. WR, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *CoRR*, vol. cs.NI/9809099, 01 1998.