Queueing Analysis of Incentive-based Extreme Edge Service Systems

Sherif B. Azmy*, Nizar Zorba♦, Hossam S. Hassanein‡
* Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada
♦ Electrical Engineering Department, Qatar University, Doha, Qatar
‡ School of Computing, Queen's University, Kingston, ON, Canada
Email: sherif.azmy@queensu.ca, nizarz@qu.edu.qa, hossam@cs.queensu.ca

GLOBECOM 2023 - 2023 IEEE Global Communications Conference | 979-8-3503-1090-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/GLOBECOM54140.2023.10436999

***Abstract*—In Edge Computing, computation is pushed towards the end-user to reduce backhaul load, address nascent privacy issues, and enable a range of low latency applications. Extreme Edge Service systems (EES) are a subset of Edge Computing in which services are deployed on user-owned devices in the proximity of the end-user. In this work, we model and analyze an orchestrator-based EES in which users' devices are recruited in exchange for an incentive. We propose to model the incentives' impact on performance using Incentive-Vacation Queueing (IVQ), a vacation queueing model in which server vacations are a proxy for incentives. Moreover, we derive closed-form expressions to evaluate the performance and directly link the performance to incentives, showing the impact of each one of the system parameters.**

***Index Terms*—Incentive; Vacation; Queueing; Extreme Edge; Edge Computing.**

I. Introduction

The number of cloud-connected devices is growing rapidly, and is expected to reach a scale that surpasses the capacity of the centralized cloud [1]. Different variants of cloud computing were proposed to help alleviate the burden of the centralized cloud. Generally, these variants involve a degree of decentralization by pushing the computation away from the centralized cloud towards the end-user [2]. This led to the development of the Fog and Edge computing paradigms. In Fog computing, services are deployed at a semi-centralized *distance* from the end-user, whereas in Edge computing, service is deployed as close as possible to the end-user by the service provider [3]. In addition to alleviating the burden on the cloud, this *proximity* has enabled a myriad of applications that were once impossible with Cloud computing due to their stringent latency and privacy requirements, namely applications such as virtual and augmented reality (VR/AR), autonomous vehicles and on-site data analytics.

However, there is still untapped potential beyond edge computing lying on the Extreme Edge (EE), by exploiting the users' own devices such as mobile devices, vehicles, and home appliances [4]–[6]. This potential is possible to tap due to the recent trend in shifting from embedded and specialized computing to general-purpose computing [1]. Not only are user-owned devices plentiful and idle most of the time, they are also computationally capable and connected. This led to the nascence of Pervasive Edge Computing, or alternatively *Extreme Edge Computing* (EEC), as a variant of edge computing where an edge service providers deploy their service on user-owned devices in the proximity of the end-user [7], [8]. In such Extreme Edge Service Systems (EESs), user-owned Extreme Edge Devices (EEDs) can function as customers, service providers, or workers.

EEC, however, comes with a few challenges. Mainly summarized into two main challenges: EEDs become multi-tenant devices, meaning that they serve their own owners (main purpose of the device) in addition to providing the computation service; and EEDs are heterogeneous devices with different usage patterns, connectivity, and capabilities. Thus, leveraging EEDs is often difficult due to the risk posed by uncertainty and unreliability.

Various works attempted to solve the problem of uncertainty on the extreme edge. A mechanism to mitigate uncertainty is suggested in [9] by approximating demand and optimizing payment for service offloading in edge-based Internet of Things (IoT) networks. Similarly, [10] uses an online learning policy to learn offloading success probabilities for edge devices based on their observed service quality. These probabilities are then employed to enhance offloading decisions. From the perspective of incentive payment, [11] proposes a user-agnostic pricing policy for service-provider edge computing. Incentive and uncertainty on the extreme edge is an area that is poorly addressed for EEC. An area that overlaps with EEC is Mobile Crowd Sensing (MCS), where sensing tasks require human involvement with incentives [12]. The main difference between MCS and EEC is that in MCS incentives are meant to compensate the effort put by the human, whereas in EEC they are meant to compensate the device's owner for a temporary reduction in the device's performance or availability. As such, the role of incentives in EEC is to make computation not only as a service, but as a commodity.

In this work, we discuss an orchestrator-based extreme edge system where the EEDs are not owned by the system, and therefore there is uncertainty about their participation and commitment. Incentives will be used to mitigate the uncertainty. In such a system, users can allow their devices to be *rent* as workers in exchange for a worthy benefit; for example, a monetary incentive or a future service. We propose Incentive-Vacation Queueing (IVQ), a model for a worker EED in an

EES with incentives. We analyze the proposed model using queueing theory, in particular the M/M/1 queue with vacations, where server vacation time represents the impact of incentives. We consider the case in which a worker's total incentive from its current jobs is uniform and derive closed-form expressions for the worker's performance.

This paper is organized as follows: in Section II, we provide an overview of extreme edge systems and the system model; in Section III we describe vacation queueing and link it to incentives in Incentive-Vacation Queueing (IVQ). In Section IV, we provide an analysis of the closed form results obtained using a numerical example. Finally, in Section V, we conclude and provide a brief outlook on the future directions for this work.

II. Extreme Edge System Overview

Unlike Edge and Fog systems, extreme edge systems rely on user-owned devices. Such devices have a heterogeneous and uncertain nature albeit their abundant idle resources. In this paper, we consider a scenario in which an extreme edge service provider, also called an orchestrator, aims to recruit workers to host their edge service. In such an extreme edge system, the orchestrator's job is to assign tasks (jobs) - and their respective payment - to the recruited workers with the objective of getting them done, ultimately providing service to its customers. In this section, we provide a description of a model for an orchestrator-based EES that this work considers, as well as provide a brief overview of the traditional M/M/1 queue without vacation.

A. System Description

The system in question comprises two main entities: i) an EE orchestrator whose job is to provide some edge service to its customers. This orchestrator is a service that could be deployed on user-owned hardware or proprietary service-provider hardware; and ii) EE workers recruited by the orchestrator to service the orchestrator's customers. One way to look at this system is to view it as a server farm [13], in which the EE orchestrator represents a scheduler with a queue in which customers' jobs - with an attached payment - arrive, and are then distributed to EE workers. Figure 1 provides an illustration of this EE scenario.

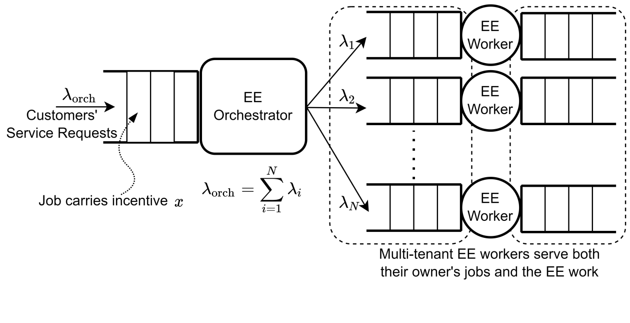
EE Queue
User Queue
λ_{orch}
Customers' Service Requests
EE Orchestrator
λ_1
λ_2
λ_N
EE Worker
EE Worker
EE Worker
Job carries incentive x
$\lambda_{\text{orch}} = \sum_{i=1}^{N} \lambda_i$
Multi-tenant EE workers serve both their owner's jobs and the EE work

Fig. 1: Extreme Edge System Job Distribution Model

Generally, customer requests can be abstracted in the form of an EE orchestrator arrival rate, λ_{orch}, which is split according to some scheduling mechanism into EE worker arrival rates $\lambda_{\text{orch}} = \sum_{i=1}^{N} \lambda_i$, assuming a total of N EE workers. Nevertheless, these workers are not entirely dedicated to servicing the edge, as they are also user-devices. This makes them *multi-tenant devices* that provide service for more than an entity. In this case, EE workers serve both the EE orchestrator - in exchange for a benefit - and serve their owners. For example, a device could be a personal computer that is *renting* its idle computational resources to the EE orchestrator but is also being used by its owner. As such, EE workers need to manage their resources in a manner that allows them to serve both the extreme edge and its owner.

In this paper, we zoom into the perspective of the EE worker, modeling the service time spent on serving the owner's jobs as a vacation whose length is dictated by the amount of incentive. This shall be described in more detail in section III, after we provide a brief overview of the $M/M/1$ queue (subsection II-B) and vacation queueing (subsection III-A).

B. Preliminaries: $M/M/1$ Queue

We prelude our analysis by a brief description of the $M/M/1$ Queue. In the $M/M/1$ queue, jobs arrive to a queue according to a Poisson process with rate λ. Similarly, these jobs that line up the queue are serviced (i.e., jobs depart) according a poisson process with rate μ (or equivalently, mean service time, $1/\mu$). The load, or utilization, denoted by ρ, is the ratio between arrivals and departures, $\rho = \lambda/\mu$. A system is said to be stable for $\rho < 1$, while unstable for values of $\rho > 1$ as jobs arrive at a rate beyond the server's ability to serve them, ultimately causing queue length to increase indefinitely.

There are a few random variables in an $M/M/1$ queue that help describe its performance. The number of customers in the system, L, has a mean of $\mathbb{E}[L] = \rho/(1-\rho)$ and a variance of $\mathbb{V}[L] = \rho/(1-\rho)^2$. Similarly, the time in the system, T, has a mean of:

$$\mathbb{E}[T] = \frac{\mathbb{E}[L]}{\lambda} = \frac{1}{\mu - \lambda} \tag{1}$$

and the time in the queue, also called waiting time, denoted T_Q, has a mean of:

$$\mathbb{E}[T_Q] = \mathbb{E}[T] - \frac{1}{\mu} = \frac{\rho}{\mu - \lambda} \tag{2}$$

Similarly, the duration for which the server is busy, B, has a mean of

$$\mathbb{E}[B] = \frac{1}{\mu(1-\rho)} \tag{3}$$

This provides a brief summary of the $M/M/1$ queue and the associated performance metrics.

III. Incentive-Vacation Queueing (IVQ)

In the EES described in section II, the EE worker has to serve both queues where the portion of service each queue receives varies according to the amount of incentive received. In this section, we propose and detail the usage of vacation queueing to model the effect of incentives on a queue such as the EE worker's. By linking vacation queueing to incentives, stakeholders and developers will have at their disposal performance metrics that would allow exploiting the extreme edge.

A. P-Limited Vacation Queueing

Vacation queueing is a type of queueing in which the server becomes unavailable for a period of time called a vacation. Implementing a vacation policy introduces a degree of flexibility in the study of real systems, as vacations abstract the server's other duties into a single random variable, V, that represents the length of the vacation. Generally, vacation queueing is a wide category of queueing disciplines that encompasses different classes [14], [15]. For example, single server vacation queueing could be classified according to the vacation policy. The vacation policy can be *exhaustive* or *non-exhaustive*, with regards to whether the server starts its vacation only after having finished the queue or not. There are different types of vacation queues as well as to whether there is a threshold (i.e., a specific number of vacations has occurred or not), whether it is preemptive or not, or whether the service is gated or not [15].

Figure 2 depicts the general vacation server's activity over time. If the type of vacation model allows consecutive service with no vacation in between, then the *service period* is the total period for which the server was busy. Similarly, if consecutive vacations have no service in between (i.e., zero-length vacation), then *vacation period* is the sum of the consecutive vacations' length. Generally, the *service cycle* spans the service period and a single vacation. This is the case for general vacation models [14]. In this work, we use a certain type of vacation queueing, P-Limited Vacation Queueing (PVQ) to model EE workers in an extreme edge scenario.

P-Limited, or *pure limited*, Vacation Queueing is a type of non-exhaustive vacation queueing in which the server takes a vacation after each departure, thus having a limited service period to only one job [14], [15]. If it happens that at a vacation completion instant there were no jobs queued for service, the server returns to repeating vacations until a job arrives. Figure 3 illustrates the server's activity over time in PVQ.

PVQ is an interesting model for vacation as it allows *polling* between the EE queue and the user queue without the need to involve the details of the non-EE queue, i.e., the other activities that the EE worker does, including the owner's jobs, are all abstracted in the vacation random variable, V. Moreover, PVQ's analysis is also simple, as it modified the service time by introducing the length of a vacation to it. In other words, an $M/M/1$ queue changes to have a modified service time, becoming an $M/\widetilde{M}/1$ queue, in which the new service time, $\widetilde{S}$, becomes a sum of the the $M/M/1$ service time, S, and the vacation time, V, i.e., $\widetilde{S} = S + V$. This is possible due to the stochastic decomposition property [15].

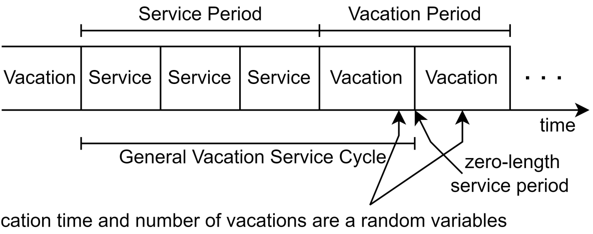

Fig. 2: General Vacation Model

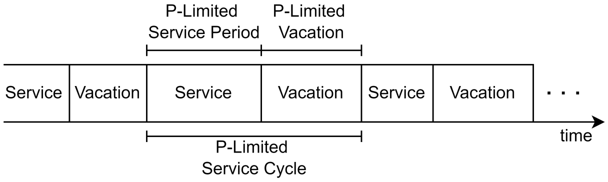

Fig. 3: P-Limited Vacation Model

As a consequence, under an $M/M/1$ PVQ, the condition for stability becomes

$$\widetilde{\rho} = \rho + \lambda \mathbb{E}[V] < 1, \tag{4}$$

while the mean number of customers in the system, $\mathbb{E}[L_v]$, becomes:

$$\mathbb{E}[L_v] = \widetilde{\rho} + \lambda^2 \frac{\frac{1}{\mu}(1 + 2\mathbb{E}[V]) + \mathbb{E}[V^2]}{2(1 - \widetilde{\rho})} + \lambda \frac{\mathbb{E}[V^2]}{2\mathbb{E}[V]} \tag{5}$$

and the waiting time, T_{Qv}, has a mean:

$$\mathbb{E}[T_{Qv}] = \frac{\mathbb{E}[L_v] - \widetilde{\rho}}{\lambda} \tag{6}$$

B. IVQ: Vacation as a Proxy for Incentives

Incentives, which are a form of credit that can be monetary or in the form of reward points, are an important part of service systems [16]. On the Extreme Edge, while the EE orchestrator is the service provider, it delegates the execution of the service on the EE worker device. The EE orchestrator matches the EE customer with an EE worker, receiving a portion, or a commission, of the incentive paid by the EE customer to the EE worker. The EE worker adjusts its service based on the amount of incentive they are receiving from the jobs in their queue.

Since each job comes associated with an incentive, x - as previously depicted in Figure 1, - the EE worker effectively has a total incentive of $X = \sum_{i=1}^{Q} x_i$. We reflect the effect of the incentive by defining the vacation as *the reciprocal of the total incentive*, i.e., $V = \frac{1}{X}$; that is, the vacation's length is a function of the lump sum of incentive present in a queue.

Such a definition has an implication on the range of values vacations and incentives can take. We define X_{min} as the minimum amount of vacation possible that translates to a maximum vacation length, V_{max}. Similarly, X_{max} translates to a minimum vacation length, V_{min}. The total incentive can then be regarded as a random variable $X \in [X_{\text{min}}, X_{\text{max}}]$, and consequently the vacation follows a random variable $V \in [V_{\text{min}}, V_{\text{max}}]$ where $V_{\text{min}} = X_{\text{max}}^{-1}$ and $V_{\text{max}} = X_{\text{min}}^{-1}$.

For simplicity of analysis and to show the effectiveness of this model, we shall assume that the total incentive follows a uniform distribution over the interval $[X_{\text{min}}, X_{\text{max}}]$. This results in the vacation following a reciprocal uniform distribution. In other words,

$$f_X = \frac{1}{X_{\text{max}} - X_{\text{min}}} \rightarrow f_V = \frac{1}{v^2} f_X \tag{7}$$

where $x \in [X_{\text{min}}, X_{\text{max}}]$. We define $X_{\text{avg}} = (X_{\text{max}} + X_{\text{min}})/2$ as the mean incentive and $X_{\text{range}} = X_{\text{max}} - X_{\text{min}}$ as the incentive range. These definitions help us rewrite X_{max} and X_{min} in terms of the incentive's mean and range, i.e.,:

$$X_{\text{max}} = X_{\text{avg}} + \frac{1}{2} X_{\text{range}}, \quad X_{\text{min}} = X_{\text{avg}} - \frac{1}{2} X_{\text{range}} \tag{8}$$

For a uniform total incentive, the EE worker's vacation time follows a reciprocal inverse uniform distribution with mean:

$$\mathbb{E}[V] = \frac{\ln(X_{\text{max}}) - \ln(X_{\text{min}})}{X_{\text{range}}} = \frac{\ln\left(\frac{4X_{\text{avg}}}{2X_{\text{avg}} - X_{\text{range}}} - 1\right)}{X_{\text{range}}} \tag{9}$$

and second moment:

$$\mathbb{E}[V^2] = \frac{1}{X_{\text{max}} X_{\text{min}}} = \frac{4}{4X_{\text{avg}}^2 - X_{\text{range}}^2} \tag{10}$$

Substituting in Eq. 5, the number of customers in an IVQ system with P-Limited vacations, L_v^{IVQ}, has a mean of:

$$\begin{aligned} \mathbb{E}[L_v^{\text{IVQ}}] &= \rho + \frac{\lambda}{X_{\text{range}}} \ln\left(\frac{4X_{\text{avg}}}{2X_{\text{avg}} - X_{\text{range}}}\right) \\ &+ \frac{\lambda X_{\text{range}}}{\coth^{-1}\left(\frac{2X_{\text{avg}}}{X_{\text{range}}}\right)\left(X_{\text{avg}}^2 - X_{\text{range}}^2\right)} \\ &+ \frac{\lambda}{2}\left(\frac{\rho + \frac{4\rho}{X_{\text{range}}} \coth^{-1}\left(\frac{2X_{\text{avg}}}{X_{\text{range}}}\right) + \frac{4\lambda}{4X_{\text{avg}}^2 - X_{\text{range}}^2}}{1 - \rho - \frac{2\lambda}{X_{\text{range}}} \coth^{-1}\left(\frac{2X_{\text{avg}}}{X_{\text{range}}}\right)}\right) \end{aligned} \tag{11}$$

Similarly, substituting in Eq. 6, the queue waiting time under P-Limited IVQ, T_{Qv}^{IVQ}, has a mean:

$$\mathbb{E}[T_{Qv}^{\text{IVQ}}] = \frac{\mathbb{E}[L_v^{\text{IVQ}}] - \frac{1}{X_{\text{range}}} \ln\left(\frac{4X_{\text{avg}}}{2X_{\text{avg}} - X_{\text{range}}} - 1\right)}{\lambda} \tag{12}$$

For a uniformly distributed total incentive in a P-Limited IVQ system, i.e., $X \sim \text{Uniform}(X_{\text{max}}, X_{\text{min}})$, the time in the system and the waiting time mainly depend on the choice of X_{max} and X_{min}. In particular, the range $X_{\text{max}} - X_{\text{min}}$, the product $X_{\text{max}} X_{\text{min}}$, the ratio between X_{max} and X_{min}, and the usual queue variables: the arrival λ and the service rate μ

IV. Results and Discussion

In this section, we demonstrate the behaviour of a P-Limited IVQ EE worker, dictated by Eqs. 11 and 12. In particular, we analyze how the system behaves with respect to the incentive's mean and range; which is a scaling of the uniform distribution's variance $\mathbb{V}[X_{\text{range}}] = \frac{1}{12} X_{\text{range}}^2$.

Figure 4 illustrates the mean number of jobs in a P-Limited IVQ system (Eq. 11) versus the mean incentive. It can be seen that the number of jobs decreases as the incentive increases. This is due to less vacation time, allowing the EE worker more time to service the jobs assigned to it, thus increasing the instantaneous service rate and effectively reducing the number of jobs in the system.

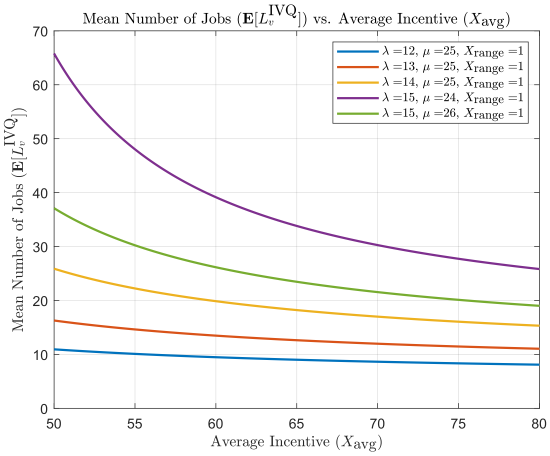

Fig. 4: Number of customers in the system vs. mean incentive $\mathbb{E}[X]$ in a P-Limited IVQ System, where $\lambda = 5, \mu = 6, X_{\text{range}} = 0.8$.

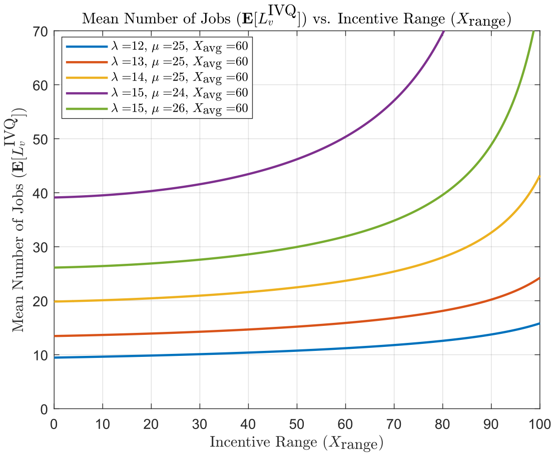

Fig. 5: Number of customers in the system vs. incentive range X_{range} in a P-Limited IVQ System, where $\lambda = 5, \mu = 6, \mathbb{E}[X] = 1$.

Figure 5 shows the relationship between the average number of jobs in the system and the range of possible values, or equivalently, the variance of a uniform X. Figure 5 shows that the number of jobs in the system increases as the range of values becomes wider. This is due to the fact that the wider the pool of incentives is, in terms of breadth, lower incentives become more likely and thus the vacation time becomes a wider distribution. As a consequence, better throughput is closely tied to less variation in incentives; ideally a constant incentive would be best.

Figures 6 and 7 tackle the temporal behaviour of the system, where in Figure 6, the time in the system decreases with more incentive allocated. As previously mentioned, this is due to the server's vacation periods shrinking and thus allowing the

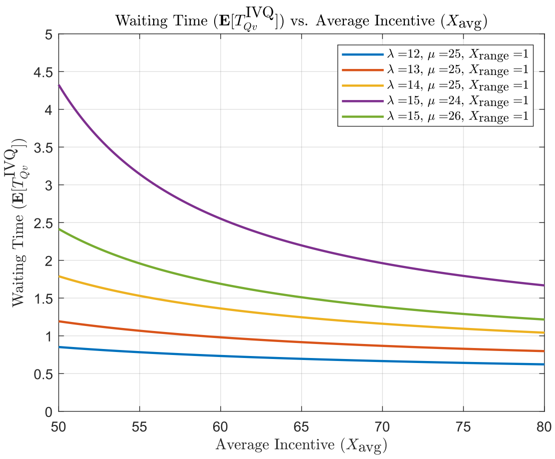

Fig. 6: Time and waiting time in system vs. mean incentive $\mathbb{E}[X]$ in a P-Limited IVQ System, where $\lambda = 5, \mu = 6, X_{\text{range}} = 0.8$.

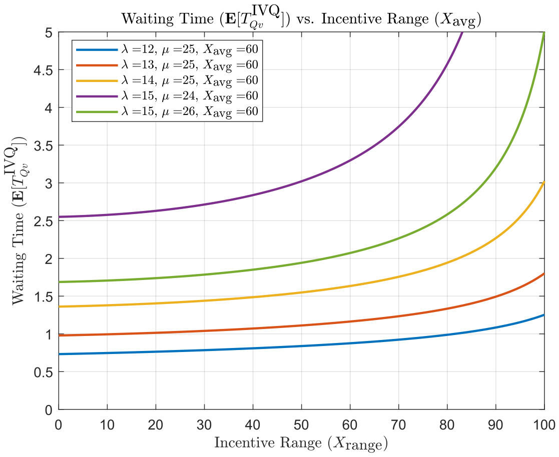

Fig. 7: Time and waiting time in system vs. incentive range X_{range} in a P-Limited IVQ System, where $\lambda = 5, \mu = 6, \mathbb{E}[X] = 1$.

server to provide more service to the jobs in the system. In Figure 7, the time spent by jobs in the system increases as X_{range} increases. This behaviour is similar to that in Figure 5, as it contributes to both the mean of the vacation distribution $\mathbb{E}[V]$ and the variance $\mathbb{V}[V]$, which are both significant terms in Eq. 12.

V. Conclusions

As Extreme Edge Systems are pushed towards the end-user, there is untapped potential that lies on users' idle and fortuitous devices. In this paper, we describe an orchestrator-based edge system that rents users' multi-tenant devices as workers for the purpose of providing an edge service, in exchange for an incentive. To service both the extreme edge and their own users, we propose the usage of P-Limited vacation queueing to model the EE worker as a server that takes a vacation to do tasks other than the extreme edge's, whereas the effect of incentives is represented in the length of the vacations. In this work, closed-form expressions that relate the performance of such a P-Limited IVQ with uniform incentives were derived, where it was clearly shown that increasing the incentives enhances the performance and reduces the time for jobs in the system. This system is useful in analyzing extreme edge systems in which user-owned devices have the potential to become a major part of the infrastructure, particularly in systems that have an EE orchestrator recruiting and distributing tasks to EE workers.

Acknowledgment

This research is supported by the Natural Sciences and Engineering Research Council of Canada under grant number ALLRP 549919-20, by the Qatar National Research Fund through grant NPRP13S-0130-200200, and by a grant from Distributive Ltd.

References

[1] D. Milojicic, "The Edge-to-Cloud Continuum," *Computer*, vol. 53, no. 11, pp. 16–25, 2020.

[2] M. S. Aslanpour, S. S. Gill, and A. N. Toosi, "Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research," *Internet of Things*, vol. 12, p. 100273, 2020.

[3] A. Yousefpour, C. Fung *et al.*, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, vol. 98, pp. 289–330, 2019.

[4] R. Meneguette, R. De Grande *et al.*, "Vehicular Edge Computing: Architecture, Resource Management, Security, and Challenges," *ACM Computing Surveys (CSUR)*, vol. 55, no. 1, pp. 1–46, 2021.

[5] LF Edge, "The Home Edge Project," 2020. [Online]. Available: https://wiki.lfedge.org/display/HOME/Home+Edge+Project

[6] S. B. Azmy, N. Zorba, and H. S. Hassanein, "CrowdDelegate: An MCS-Based Approach for Improving Retail Labor Cost-Efficiency," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019.

[7] R. Tourani, S. Srikanteswara *et al.*, "Democratizing the Edge: A Pervasive Edge Computing Framework," *ArXiv*, 2020.

[8] A. Kallel, M. Rekik, and M. Khemakhem, "IoT-fog-cloud based architecture for smart systems: Prototypes of autism and COVID-19 monitoring systems," *Software: Practice and Experience*, vol. 51, pp. 116 – 91, 2021.

[9] A. Samanta, F. Esposito, and T. G. Nguyen, "Fault-Tolerant Mechanism for Edge-Based IoT Networks With Demand Uncertainty," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 16 963–16 971, 2021.

[10] C.-S. Yang, R. Pedarsani, and A. S. Avestimehr, "Edge Computing in the Dark: Leveraging Contextual-Combinatorial Bandit and Coded Computing," *IEEE/ACM Transactions on Networking*, vol. 29, no. 3, pp. 1022–1031, 2021.

[11] Z. Wang, L. Gao *et al.*, "Monetizing Edge Service in Mobile Internet Ecosystem," *IEEE Transactions on Mobile Computing*, vol. 21, no. 5, pp. 1751–1765, 2022.

[12] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Quality Estimation for Scarce Scenarios Within Mobile Crowdsensing Systems," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10 955–10 968, 2020.

[13] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, ser. Performance Modeling and Design of Computer Systems: Queueing Theory in Action, 2013.

[14] H. Takagi, *Queueing Analysis: Discrete-time Systems*, ser. Queueing Analysis: A Foundation of Performance Evaluation, 1991.

[15] N. Tian and Z. G. Zhang, *Vacation queueing models: theory and applications*, 2006, vol. 93.

[16] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Incentive-Vacation Queueing for Extreme Edge Computing Systems," in *2023 IEEE International Conference on Communications (ICC)*, 2023.