Received 31 December 2023; revised 29 February 2024; accepted 22 March 2024. Date of publication 29 March 2024; date of current version 18 April 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.3383046

Incentive-Vacation Queueing in Extreme Edge Computing: An Analytical Reward-Based Framework

SHERIF B. AZMY [1] (Graduate Student Member, IEEE), NIZAR ZORBA [2] (Senior Member, IEEE), AND HOSSAM S. HASSANEIN [3] (Fellow, IEEE)

[1]Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada

[2]College of Engineering, Qatar University, Doha, Qatar

[3]School of Computing, Queen's University, Kingston, ON K7L 3N6, Canada

CORRESPONDING AUTHOR: N. ZORBA (e-mail: nizarz@qu.edu.qa)

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant ALLRP 549919-20; in part by the Distributive, Ltd.; and in part by Qatar National Library.

An earlier version of this paper was presented in part at IEEE International Conference on Communications, Rome, Italy, 2023 [1].

ABSTRACT Edge Computing (EC) emerged to address the cloud's shortcomings in meeting demand and latency requirements, leading to a shift in computation closer to the end-user. Extreme Edge Computing (XEC) extends this approach by utilizing nearby user-owned computational resources to support latency-sensitive applications in a distributed manner. In this study, we introduce Reward Edge Computing (REC), a variant of XEC, where service providers recruit user devices for infrastructure support, offering rewards in return. We explore the use of Incentive-Vacation Queueing (IVQ) to manage REC and analyze both its long-term and short-term performance. Our analysis focuses on the choice of an Incentive-Vacation Function (IVF), a contractual function between workers and service providers, proposing a tunable model favoring either party. We provide closed-form expressions for long-term worker behavior under uniform workload pricing and analyze the system's overall short-term operation, including the time a worker spends in the system. REC and IVQ aim to commodify computational resources for edge services, akin to sharing economy models like Uber and Airbnb, utilizing user-owned infrastructure.

INDEX TERMS Extreme edge computing, distributed crowd computing, incentive vacation queueing, incentive mechanisms, performance modeling.

I. INTRODUCTION

EDGE Computing (EC) emerged to alleviate the demand on centralized cloud services by decentralizing computation and bringing it closer to end-users. This approach effectively distributes computational tasks, enhancing performance, reducing latency, and bolstering quality of service and privacy [2].

However, the extent of EC ceases at the network boundaries of enterprise service providers. Consequently, EC fails to leverage the significant resources available on user-owned devices at the Extreme Edge [3], [4]. Exploiting user-owned resources has been challenging for various reasons. Initially, user-owned devices lacked computational power, but the shift over the last two decades from specific-purpose hardware to more capable general-purpose devices, like smartphones, has changed this perception [5]. Although individually user-owned devices may not match the computational power of an edge server, collectively, a multitude of these devices on the *extreme edge* can create a distributed edge server [6]. Realizing such a distributed extreme edge server is complicated, requiring stable interconnectivity between its constituent devices, a factor previously constrained by rigid network and computational architectures [7]. However, with advancements in software-defined networks [8], virtualization [9], and containerization [10], particularly through microcontainers [11] and unikernels [12], these limitations are becoming manageable. As a consequence, a new layer on the edge-cloud continuum is conceivable, positioned beyond

the last mile of current EC, at the *extreme edge*. This layer is known as Extreme Edge Computing (XEC) [1], [13].

XEC is a novel paradigm expanding EC to include Mist Computing and various IoT and crowd computing sub-paradigms. This paper introduces Reward Edge Computing (REC), a sharing economy model within XEC. In REC, service providers deploy services on "worker devices" near the end-user to maintain service continuity, especially when stable connections are unpredictable. This approach is particularly useful in scenarios, where service assets are temporarily deployed on devices in the vicinity to ensure uninterrupted service provision, as the example of mobile game streaming on an intercity train [14]. XEC service providers compensate participation through incentive payments, ensuring seamless user experience despite connectivity challenges [15].

The implementation of XEC and REC, aiming for uninterrupted service provision, encounters significant challenges primarily due to the nature of user-owned devices as multi-purpose and non-dedicated servers [1]. Devices like smartphones serve varied personal needs, making their availability and performance unpredictable. This unpredictability is a direct result of human behavior, which is inherently erratic and variable as it changes over the day [16], [17]. REC addresses this concern by incentivizing device owners to temporarily allocate a portion of their device's capabilities for REC tasks in return for payment. This concept mirrors the sharing economy models of Uber and Airbnb, treating computational resources as a tradable commodity and allowing owners to monetize their device's unused capacities [18].

XEC and REC both have precedents in the literature. The idea of providing computational service in a sharing economy setting has been proposed. The authors in [19] explore forming ad-hoc edge clouds with the purpose of establishing a democratized open market for computation. Their work proposes the distributed alternative to expensive enterprise hardware to provide edge services. Similarly, Microclouds [20] are a prototype that envisions the involvement of user-managed infrastructure in service provision. With regards to the permeance and democratization of service provision, [21] highlights moving from enterprise (private) to citizen-influenced and open-sourced infrastructure. In such systems, the role of smart contracts is paramount as it describes the key elements between the parties involved (metrics, temporality, pricing, operational boundaries, service-level guarantees, and the legal aspects). Such smart contracts are key to enabling seamless XEC and REC-based service provision, especially if they involve user-related metrics such as risk preference [22]. Furthermore, the notion of incentivizing users has been considered before but in the purpose of incentivizing end-users to tolerate delay. The authors in [23] investigate a burdened edge system in which users are incentivized to sacrifice better latency by receiving server from the fog or the cloud. In [24], computational offloading by road-side units to *selfish* service vehicles was investigated. The authors of [24] overcome selfish behaviour by having the RSU lease resources from multiple vehicles, while having the vehicles select contracts that maximize their rewards. Similarly, the authors in [25] study the usage of electric vehicles as computational resource nodes during their charging time. They study the aspect of scheduling computational tasks in exchange for energy with the purpose of maximizing social welfare.

However, the evaluation of the performance of sharing economy systems in light of the application requirements of EC systems is barely treated in the literature. As a result, posing computational offloading as a sharing economy business model in light of the application requirements of EC systems remains a niche. The general trend in the literature is to design incentive mechanisms, starting from a target performance. The measurement and modeling of performance starting from the presence of incentives that are influenced by human whims and their impact on performance is a nascent yet significant topic.

Our study addresses two key challenges in XEC and REC: the multi-tenancy of worker devices and the sporadic availability of these devices. We introduce the Incentive-Vacation Queueing (IVQ) model to manage the dual use of devices, analyzing worker performance and service provision. Additionally, we extend IVQ to a real-time approach for a collective of workers, offering a holistic view of service availability and derive metrics to predict worker churn. This allows for proactive management of XEC services, aiming for consistent and reliable uptime in the dynamic landscape of REC. The contributions of this paper can be summarized as follows:

1) We introduce Reward Edge Computing (REC), a sub-paradigm of XEC, incentivizing user-owned devices' use for distributed service infrastructure.
2) We develop the Incentive-Vacation Queueing (IVQ) model to manage worker performance in XEC, with incentive-based vacation lengths.
3) We propose a tunable Incentive-Vacation Function (IVF) to balance preferences between workers and service providers, optimizing reward and performance dynamics.
4) We extend IVQ to a real-time variant, analyzing its impact on system-wide operations and deriving key metrics for worker time and churn to enhance service reliability and availability.

This paper is organized as follows: Section II presents an overview of the XEC/REC system, detailing its architecture and operation. Section III introduces the IVQ model and discusses relevant metrics. Section IV expands on the real-time variant of IVQ for distributed XEC service provision and models service availability contingent on worker presence. In Section V, we validate our approaches via simulation. Finally, Section VI concludes the paper.

II. EXTREME EDGE COMPUTING

XEC distinguishes itself from traditional EC by targeting computation on devices in close proximity to the end-user,

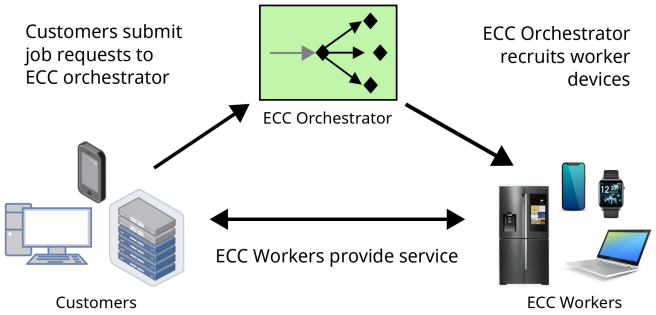

FIGURE 1. Reward Edge Computing System on the Extreme Edge.

focusing on the utilization of user-owned hardware. Unlike EC's reliance on enterprise-owned infrastructure designed for organizational objectives, XEC capitalizes on personal devices intended primarily for individual use. While some extreme edge devices, like those in industrial IoT and Mist Computing, are enterprise-owned, they are typically specialized, embedded, and proprietary, making broader application challenging. Consequently, user-owned devices form the core of XEC's infrastructure.

In this section, we provide an overview of the architecture and operation of a Reward Edge Computing (REC) system, a semi-distributed instance of XEC, acknowledging that while XEC devices are limited in computational capabilities and subject to owner behaviors, they are collectively valuable and best utilized in a distributed or semi-distributed manner.

As shown in Figure 1, REC involves service providers delivering services to end-users without owning the necessary infrastructure for deployment. To overcome this shortage, they rent computational resources by enlisting worker devices located near the end-user, effectively utilizing the immediate vicinity's computational assets. As such, the REC system is comprised of three main entities:

- End-users (or customer devices): Individuals or entities seeking services from the service provider.
- Worker devices: General-purpose, user-owned devices rented out for service provision to end-users in exchange for a reward. Owners may optimize device availability for increased rewards.
- Service Provider (or orchestrator): Manages and deploys services on worker devices, compensating with rewards while aiming for profitability by balancing reward payouts and revenue.

It's noteworthy that while reward computing systems can function independently of XEC, within the XEC context they resemble sharing economy models like Uber or Airbnb with the added challenge of ensuring low latency and privacy guarantees at the extreme edge. This alignment with sharing economy principles in the demanding context of XEC highlights both the potential and the challenges of implementing REC systems effectively.

The operation of the REC system, illustrated in Figure 2, unfolds as follows: A customer requests a service from the service provider, who then recruits worker devices to

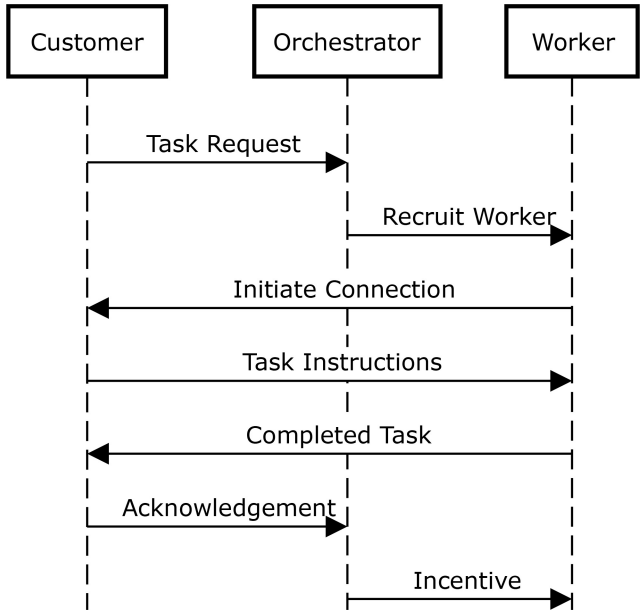

FIGURE 2. Operation of an REC System: service requested by end-user is deployed by the service provider on REC worker devices.

distribute and compute the service's load. Suitable workers in proximity to the end-user facilitate timely service provision through direct communication. Meanwhile, the service provider oversees the service quality remotely and through a local daemon on the customer's device. Workers are rewarded for each successfully computed service slice, drawing parallels to established distributed computation models.[1]

In a more general sense, in an REC a service provider has a pool of workers that are ready to be recruited, or are proactively expected to be available for recruitment. When a customer requests an edge service, that is a service that has stringent latency requirements and requires up-time for a certain duration, the service provider orchestrates the provision of that service by deploying that service on user-owned XEC worker devices. The service is deployed in a distributed setting on multiple XEC worker devices to guarantee service reliability in case a worker churns. The service provider is also ready to handover the deployed service to either a parallel set of XEC workers (or equivalently, an XEC server) in case more than a worker churns, or to elevate service deployment to an EC server. As such, the service provider seeks to guarantee the Service Level Agreement (SLA) specifications promised to the end-user. However, elevating to an EC server might violate the stringent latency requirements if the end-user is in an area of weak coverage. Proactive handing over to another XEC server (which is a collective of XEC workers) can help guarantee such service provision.

[1]Similar commercial precedents include Distributive Inc. and the Distributed Compute Protocol (DCP) [26].

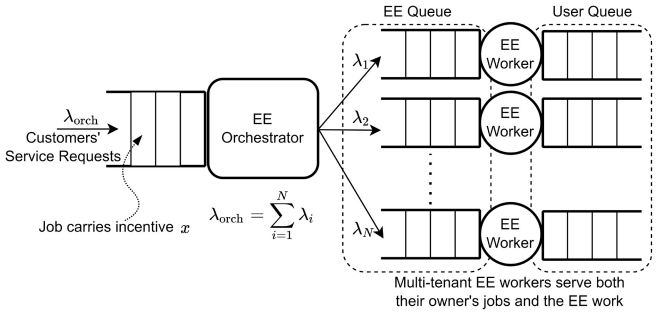

FIGURE 3. REC Server Farm Model: the service provider acts as a scheduler that assigns slices to multi-tenant worker devices.

To model this system, we adopt a server farm approach with the service provider acting as a scheduler, distributing service slices to REC workers, as depicted in Figure 3. The XEC service provider manages worker arrival rates and incentive distribution, assuming sufficient revenue to cover worker rewards. An REC system may engage up to K workers, but as these are multi-tenant devices, owner use of his/her device may impact availability. The subsequent section details how we integrate incentive payments with vacation queueing to formulate a policy managing this multi-tenancy, influenced by the level of incentives provided.

III. INCENTIVE-VACATION QUEUEING

In this section, we address the need for a new queueing model, the Incentive-Vacation Queueing (IVQ), essential in XEC. Driven by the unpredictable availability of user-owned devices, we seek a model that offers flexibility for service providers and fair compensation for workers. We turn to traditional P-limited vacation queueing models, integrating incentives with vacation durations to create a direct link between worker pay and service availability. Our introduction of IVQ delves into its performance, analyzing how it affects the length of the queue and waiting times. We discuss the incentive-vacation function and its adaptability through the parameter α, offering a system that can be fine-tuned for different service and worker scenarios. This section provides a comprehensive analysis, showing how incentive structures can significantly affect both service efficiency and worker engagement, aiming for an operational balance that benefits all parties involved.

A. PRELIMINARIES: VACATION AND P-LIMITED VACATION QUEUEING

Vacation Queueing refers to a family of queueing disciplines in which the service is unavailable for periods of time, or *vacations*. A vacation policy introduces a degree of flexibility in the modeling of real-time systems as it allows abstracting a wide range of activities into a single random variable, V, which is the duration of the server's vacation [27], [28], [29]. The random variable V ends up having statistical properties that stem from the activities that constitutes it. As a consequence, the duration ultimately follows some distribution of V. In the context of XEC

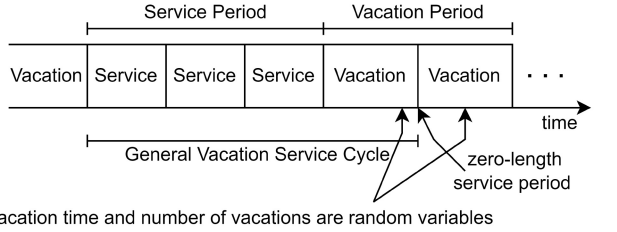

FIGURE 4. General Vacation Queueing: in VQ, a service cycle consists of a period of consecutive service followed by a period of consecutive vacations.

and uncertainties induced to user behaviour, the distribution of V stems from two sources: the user's behaviour, and how influenced they are by how much incentives they are promised.

Figure 4 illustrates the activity of a server following a vacation queueing discipline. A server's server cycle consists of a one or more services followed by one or more vacations. In a loose sense, a vacation can be used to perform maintenance or setup, or do another task (such as serving another queue, for example) [28]. The total duration of service within a service cycle is denoted the *service period*. Similarly, the total duration of vacations within a service cycle is referred to as the vacation period [27], [28].

Vacation queueing is promising for the modeling of XEC systems, particularly REC systems, for a number of reasons. They are capable of capturing the dynamics of the workers and abstracting them in the variable V. In contrast to processor-sharing queueing models, vacation queuing does not require knowledge of what the worker device intends to do; an estimate of the vacation duration is sufficient for a vacation model. Additionally, some vacation models can be mapped to a standard queue with modified parameters. This versatility allows using vacation models as a measuring tool for dynamical systems, particularly those involving incentives whose amounts impact performance. They are also capable of capturing the aspect of worker churn if they were to go into a vacation indefinitely. These characteristics are even more important in a highly dynamic environment such as the extreme edge. Proactive techniques that require data and historical analysis of not as efficient as reactive techniques on the extreme edge due to the transient and varying nature of workers. Vacation queueing models provide a closed-form performance model that allows quick reactive analysis with minimal complexity. Such closed-form analysis and monitoring help the service provider make quick decisions. This is key to ensure the performance of ephemeral edge servers, formed by user-owned XEC devices.

In this work we utilize a type of vacation queueing models called the P-Limited Vacation Queueing (PVQ) to model the performance of a worker in an XEC scenario. As illustrated in Figure 5, a server following PVQ would always take a vacation after every service. If it happens that there is no service available, then the server keeps repeating vacations until a job arrives.

An interesting aspect of PVQ lies in its compatibility with multi-tenancy, which is the natural mode of operation

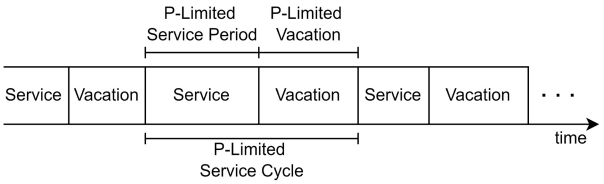

FIGURE 5. P-Limited Vacation Queueing: in PVQ, a service cycle is always a single service followed by a single vacation. It is analogous to an $M/G/1$ queue with a modified service period.

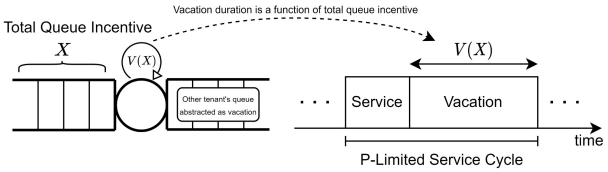

FIGURE 6. Incentive-Vacation Queueing: In IVQ, vacation duration V is influenced by how much incentives the worker is receiving. This allows capturing the multi-tenancy, as well as the idleness duration.

of user-owned XEC devices; users expect to be capable of using their personal and smart devices when they desire to. The other benefit of using PVQ lies in the fact that it allows the simplification of the analysis to that of an $M/G/1$ queue with a modified service time, $\widetilde{S}$. This is due to the fact that the stochastic decomposition property allows expressing the vacation influenced service time as the sum of the traditional $M/G/1$ service time S and the PVQ vacation time, V, i.e., $\widetilde{S} = S + V$ [28]. This property of the PVQ system is powerful as it allows the analysis of any $M/G/1$-analogous system - whether it has vacation or not - as a vacation queue, as the PVQ server utilization, $\widetilde{\rho}$ then becomes:

$$\begin{aligned}\widetilde{\rho} &= \rho + \lambda\mathbb{E}[V] \\ &= \lambda(\mathbb{E}[S] + \mathbb{E}[V]) < 1,\end{aligned} \tag{1}$$

where λ is the arrival rate of tasks to the queue, $\rho = \lambda\mathbb{E}[S]$ is the $M/G/1$ utilization, and $\mathbb{E}[V]$ is the expected vacation duration. The PVQ utilization is the ratio of service done during the service cycle. The presence of the vacation reduces the $M/G/1$ utilization by $\lambda\mathbb{E}[V]$ as $\mathbb{E}[V]/(1/\lambda)$ represents the proportion of the service cycle spent in vacation.

The mean number of slices present in the system (in queue and one in service) becomes

$$\begin{aligned}\mathbb{E}[L_v] = \tilde{\rho} + \lambda^2 \frac{\mathbb{E}\left[S^2\right] + 2\mathbb{E}[S]\mathbb{E}[V] + \mathbb{E}\left[V^2\right]}{2(1-\tilde{\rho})} \\ + \lambda \frac{\mathbb{E}\left[V^2\right]}{2\mathbb{E}[V]}\end{aligned} \tag{2}$$

where $\mathbb{E}[V^2]$ is the vacation distribution's second moment. In case the vacation duration is null, i.e., $\mathbb{E}[V] = \mathbb{E}[V^2] = 0$, this expression reduces to the fundamental $M/G/1$ number of slices expression, $\mathbb{E}[L_v|V=0] = \lambda\mathbb{E}[S] + \lambda^2\mathbb{E}[S^2]/(2(1-\rho))$. As such, the PVQ model is inclusive of modeling its fundamental $M/G/1$ queue.

The PVQ waiting time can then be computed

$$\mathbb{E}\left[T_{Qv}\right] = \frac{\mathbb{E}[L_v] - \tilde{\rho}}{\lambda} \tag{3}$$

where $\frac{1}{\lambda}$ is the interarrival time that also corresponds to the average duration of the service cycle $\widetilde{S} = S + V$.

PVQ's relationship to the $M/G/1$ queue is especially useful when considering the economic aspect of incentive payments and treating computational resources as a commodity in a market [30]. In the following subsection, we describe the Incentive-Vacation Queueing, a modification of the PVQ that links incentives to performance through the vacation variable.

B. INCENTIVE-VACATION QUEUEING: A SINGLE SERVER PERSPECTIVE

In a Reward Edge Computing (REC) system, participants who are renting their resources to the service provider do so in exchange for a monetary reward. This reward is in exchange for the deployment of a service to serve other users using their hardware during durations of idleness, and in exchange for deliberately increasing this idleness duration in exchange for more incentives. But it is also the case that these devices might be essential to their owners, thus they might need to use them. As such, multi-tenancy must be respected. By defining the vacation duration as a function of the incentives, we can capture both: the duration for which a device is idle and its resources are available for use, as well as the user's deliberation to stay idle in exchange for making more profit. The vacation duration, V, in this sense, refers to the duration in which the device's owner needs to use it, while the service duration, S, refers to the idleness duration in which the service provider can provide their service to their end-user.

Incentive-Vacation Queueing is a form of PVQ in which the vacation duration, V, is a function of amount of incentives X the worker expects to receive. In an REC system, the orchestrator receives the fee for service provision from the customers. It would then utilize the resources of the worker devices and give them an incentive amount X to rent their resources. At a specific time instance, a worker device sees a incentive total of X for the jobs it has in its queue. As a consequence, we refer to the amount of incentive X as the *Total Queue Incentive (TQI)*. The TQI is crucial for sharing economy systems because it indicates the potential revenue a worker can receive in return for providing the service on behalf of the service provider. In IVQ, V becomes an *Incentive-Vacation Function (IVF)* of X, denoted $r(X)$ that captures a contractual relationship between the worker's performance and the amount of incentives they receive *if they choose to put more time* servicing for the edge, giving them a tangible measure of the opportunity cost of their vacation. Figure 6 illustrates this link between incentives and vacations.

In the following subsection, we discuss the properties of a valid $r(X)$, as well as provide a tunable variant of it with a

parameter α that would represent how the orchestrator trusts or values the worker.

C. THE INCENTIVE-VACATION FUNCTION, R(X), AND ITS IMPACT

In IVQ, the IVF needs to be chosen in a manner that would guarantee the system's stability and performance. Thus, the proper selection of $r(X)$ is of crucial importance and significant impact on the performance of the worker (from the perspective of the service provider), as it would abstract the behaviour of the primary tenant (the worker device's owner). There are numerous aspects that can be captured by the proper choice of $r(X)$ that extend beyond just multi-tenancy. For example, aspects related to the worker device's capabilities, performance, reliability, as well as the uncertainties stemming from behaviour. However, for the current work, we only include aspects relating to the presence of incentives and the idleness duration in the IVF. Furthermore, we capture the aspect of how trustworthy a worker device is deemed in a tunable parameter α that reflects the IVF's *fairness* in the pricing of marginal vacation durations.

Initially, there are specific properties regarding the general choice of the IVF, $r : X \mapsto V$. It needs to be

- $r(X)$ has to be monotonically decreasing as longer incentives should yield shorter vacations as device owners would be more deliberate in increasing the idle time.
- $r(X)$ needs to be chosen in a manner that guarantees the stability of the system, i.e.,

$$\mathbb{E}[V] = \mathbb{E}[r(X)] < \frac{1}{\lambda} - \mathbb{E}[S], \tag{4}$$

  where $\mu = 1/S$ is the $M/G/1$ service rate; $\mathbb{E}[r(X)]$ refers to the average vacation being smaller than the difference between interarrival time and service time. If this is not the case, jobs would pile up in the worker's queue infinitely; this particularly undesirable in a service provision scenario such as an XEC scenario in which latency is paramount for the quality of service and experience.
- $r(X)$ is bounded by a minimum and a maximum vacation duration, $V_{\min}$ and $V_{\max}$, respectively. Its domain is also bounded by the TQI equivalents, $V_{\min} = r(X_{\max})$ and $V_{\max} = r(X_{\min})$. This choice of bounds guarantees stability as well as a minimum threshold for the idleness duration, $S_{\min} = 1/\lambda - V_{\min}$ that the REC system guarantees. Thus, $r : X \in [X_{\min}, X_{\max}] \mapsto V \in [V_{\min}, V_{\max}]$.

A vacation function, $r(X)$, is *admissible* if and only if it satisfies these three conditions.

It is important to note, however, that these restrictions on $r(X)$ do not restrict the convexity of $r(X)$ as both concave and convex functions can be monotonically decreasing. The convexity of $r(X)$ is an important factor as it affects the marginal pricing of each second spent in vacation. A convex $r(X)$ would price marginal vacations at a lower price than a concave $r(x)$. A consequence of this is that the IVF would be favoring the orchestrator if it is convex, and favoring the worker device if it is concave. Figure 7 illustrates the relationship between convexity and preference.

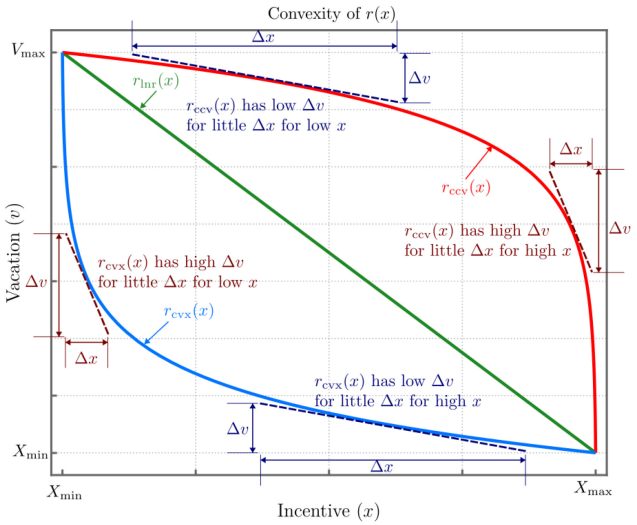

FIGURE 7. Convexity of the Incentive-Vacation Function: A convex $r(X)$ (in blue) values multi-tenancy at a less price, which is more favorable to the service provider. On the contrary, a concave $r(X)$ values the multi-tenancy at a high price. A linear $r(X)$ (in green) is fair.

In this work, we use a specific α-parametrized $r(X)$ defined as:

$$r(x,\alpha) = \begin{cases} (1+\alpha)r_{\text{lnr}}(x) - \alpha r_{\text{cvx}}(x) & -1 \le \alpha \le 0 \\ (1-\alpha)r_{\text{lnr}}(x) + \alpha r_{\text{ccv}}(x) & 0 < \alpha \le 1 \end{cases} \tag{5}$$

where the term $r_{\text{cvx}}(x)$ a convex component that is exclusively present when $\alpha = -1$; $r_{\text{lnr}}(x)$ is a linear component that is exclusively present when $\alpha = 0$; and $r_{\text{ccv}}(x)$ is a concave component that is exclusively present when $\alpha = 1$. In this work, we select the components $r_{\text{cvx}}(x)$, and $r_{\text{ccv}}(x)$ to be:

$$r_{\text{cvx}}(x) = \frac{V_{\max}V_{\min}(X_{\min} - X_{\max})}{(V_{\min} - V_{\max})x - X_{\max}V_{\min} + V_{\max}X_{\min}}, \tag{6}$$

and

$$r_{\text{ccv}}(x) = \frac{V_{\max}^2(x - X_{\max}) + V_{\min}^2(X_{\min} - x)}{V_{\max}(x - X_{\max}) + V_{\min}(X_{\min} - x)}. \tag{7}$$

Finally, the linear component is

$$r_{\text{lnr}}(x) = \frac{V_{\min} - V_{\max}}{X_{\max} - X_{\min}}x + V_{\max}\left(1 - \frac{X_{\min}(V_{\min} - V_{\max})}{V_{\max}(X_{\max} - X_{\min})}\right). \tag{8}$$

The reason for this choice that having a rational convex function as the kernel of the IVF, a concave anti-convex can be created by rotating $r_{\text{cvx}}(x)$ around the midpoint $(\frac{1}{2}(X_{\min} + X_{\max}), \frac{1}{2}(V_{\min} + V_{\max}))$. The rational function provides a symmetry in the curvature of the IVF, which makes it a fair function to represent a contrctual agreement between both service providers and workers. Of course, this symmetry has to be violated if the service being provided requires

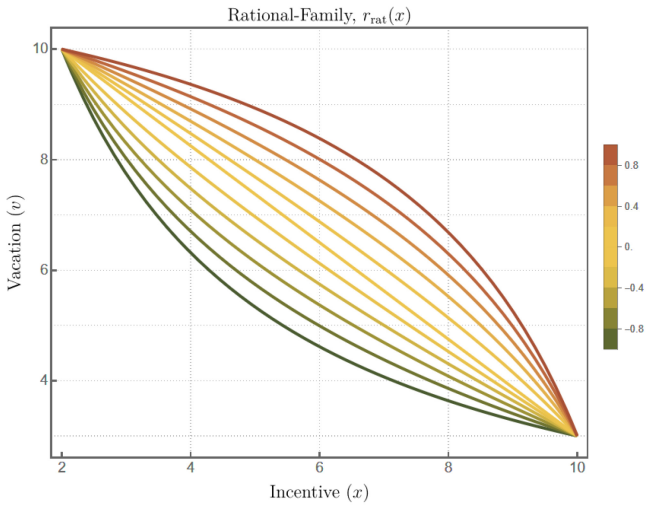

FIGURE 8. A Rational α-parameterized Incentive-Vacation Function: changing α moves the curve from the most convex to the most concave, changing the marginal pricing of vacation durations. High α represents high worker preference, while low α prefers the service provider.

a degree of reliability or criticality. The definition of the IVF in that manner, in combination with the parameter α, adds an aspect of negotiation and trust to the contractual relationship between the service provider and the worker. For instance, changing α from -1 to -0.25 reduces the drop in the marginal pricing of the vacation, increasing the net profit the worker can make from the contract. Increasing α further to 1 increases such profit, but puts a greater burden on the service provider's budget.

The rational α-parametrized IVF is illustrated in Figure 8. The choice of parameter α reflects the degree of preference to either the service provider or the worker in the pricing of the vacation duration, as previously illustrated in 7. A choice of $\alpha = -1$ is an extreme choice that favors the service provider over the worker, while a choice of $\alpha + 1$ is vice-versa. A consequence of defining the IVF is that it provides the worker with a measure of their opportunity cost, allowing them to decide as to whether it would be profitable for them to put in more time towards the edge work or not. This degree of versatility offered by the IVF is also useful to the service provider as the retention of workers is crucial for sustainable and reliable service provision on the extreme edge whose infrastructure is highly volatile and dynamic. Tuning α allows a contractual negotiation between both parties, the service provider and the worker, to ensure fairness along service provision.

D. PERFORMANCE ANALYSIS UNDER UNIFORM JOB INCENTIVES

The incentive variable in IVQ, X, represents the amount of incentives the worker is expecting for a collection of slices at a time. However, this definition is not restrictive, but it serves for a long-term analysis of a worker under IVQ. It is more practical for IVQ to adjust the duration of the vacation based on each slice; performance at this level of detail is discussed in the following section. However, in the current section, we look at a TQI that stems from a set of slices whose incentive are uniformly distributed. In other words,

$$X = \sum_{i=1}^{n} x_i \text{ for } x_i \sim \text{Uniform}(x_{\min}, x_{\max}) \tag{9}$$

where x_i is the incentive attached to the i^{th} slice, which is uniformly distributed between $x_{\min}$ and $x_{\max}$, while n is the number of slices a worker observes in its queue.

A natural outcome of this relationship is that the total queue incentive follows an Irwin-Hall distribution [31], with a minimum possible value of $X_{\min} = nx_{\min}$ and a maximum possible value of $X_{\max} = nx_{\max}$. In other words,

$$f_X(x; x_{\min}, x_{\max}, n) = \frac{1}{x_{\max} - x_{\min}} \left(\frac{\sum_{k=0}^{\left\lfloor \frac{x - nx_{\min}}{x_{\max} - x_{\min}} \right\rfloor} (-1)^k \binom{n}{k} \left(\frac{x - nx_{\min}}{x_{\max} - x_{\min}} - k \right)^{n-1}}{(n-1)!} \right) \tag{10}$$

or equivalently in terms of the unit step function

$$f_X(x; x_{\min}, x_{\max}, n) = \frac{1}{x_{\max} - x_{\min}} \cdot \sum_{k=0}^{n} (-1)^k \binom{n}{k} \frac{\left(\frac{x - nx_{\min}}{x_{\max} - x_{\min}} \right)^{n-1}}{(n-1)!} u\left(\frac{x - nx_{\min}}{x_{\max} - x_{\min}} - k \right) \tag{11}$$

illustrating an n-fold convolution.

This expression can be approximated using the normal distribution for values of $n > 2$ with an error bounded by the Berry-Esseen bound [32], and converges to a normal distribution for large n due to the central limit theorem. In particular,

$$f_X(x; x_{\min}, x_{\max}, n) \cong \sqrt{\frac{6}{\pi n (x_{\max} - x_{\min})^2}} \exp\left(- \frac{6\left(x - \frac{1}{2} n (x_{\min} + x_{\max}) \right)^2}{n (x_{\max} - x_{\min})^2} \right) \tag{12}$$

As such, computing performance metrics based on Eq. (1) can be done by utilizing the Law of the Unconscious Statistician (LOTUS) where

$$\begin{aligned} \mathbb{E}[V] &= \mathbb{E}[r(X, \alpha)] \\ &= \int_{nx_{\min}}^{nx_{\max}} r(x, \alpha) f_X(x; x_{\min}, x_{\max}, n) dx \\ \mathbb{E}\left[V^2\right] &= \int_{nx_{\min}}^{nx_{\max}} (r(x, \alpha))^2 f_X(x; x_{\min}, x_{\max}, n) dx \end{aligned} \tag{13}$$

where both integrals are transcendental and cannot be simplified further. This is mainly due to the similarity of these expressions to the Gaussian error function, $\text{erf}(\cdot)$. The average IVQ metrics for a single worker derived in Section III-A can be employed to evaluate the long-term performance of a single worker. What remains is a direct substitution in Eq. (1).

IV. WORKER SLOT MODEL: XEC SERVICE MODEL UNDER IVQ

In XEC-REC, our goal is to utilize distributed infrastructure of consumer and user-owned devices for service provision. While voluntary resource contribution is valuable, the reliability of such crowd computing varies widely. Introducing incentives can enhance this reliability. IVQ, discussed earlier, provides a framework for understanding average system performance over time. Yet, to effectively manage and control these systems, especially for low-latency applications characteristic of XEC, it's necessary to focus on real-time analysis and operation, narrowing down the time horizon for more immediate responsiveness, i.e., short term analysis.

In this section, we delve into the real-time dynamics of the IVQ model to estimate worker system presence and assess the availability of a "live service" considering system capacity and worker requirements. We introduce a "worker slot perspective" for the service provider, combining detailed worker activity and abstract slot activity to model worker time in the system. Using Continuous-time Markov Chains (CTMCs) for state holding times, we derive a closed-form expression reflecting worker and service availability. Our goal is to predict system time based on service provider parameters and incentive structures, contrasting the long-term behavior from Section III with the immediate, slice-incentive-based dynamics here. This analysis is crucial for developing real-time scheduling techniques, optimizing system efficiency, and selecting appropriate preference parameters, α, for each worker.

A. WORKER SLOT MODEL

In XEC-REC, due to its remote and distributed nature on the edge and extreme edge, the service provider divides the workload for service provision into slices of minimal computational requirement that would allow them to run on microcontainers and unikernels, and thus to be deployed on smart and mobile devices with minimal impact on their performance. As a consequence, the payment is processed on a per-slice basis, where each slice is priced in a manner similar to how Application Programming Interface (API) tokens are priced in today's SaaS models. This allows slices to be offloaded to a worker device and allows the compensation of the worker device's owner.

However, having a single worker might not be sufficient for the service provision. This is due to issues in reliability and persistence, which are expected on the extreme edge due to unpredictable user behaviour. This is due to the fact that ad-hoc architectures are heavily impacted by node churns [10]. Thus, it is rational for the service provider to recruit a number of devices for the service provision. We assume that the service provider has some required number of workers, $k_{\min}$, necessary for its service to be alive at a specific time, where $k_{\min} < K$, the maximum number of workers the service provider can handle. This *global view* extends to each of the K *worker slots*, where each of them has its own activity.

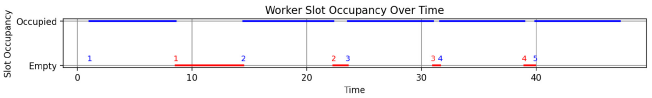

FIGURE 9. Example Worker Slot Activity Over Time. Slot is occupied by the w^{th} worker until it churns, then it remains empty until the $(w+1)^{\text{th}}$ worker arrives and so on.

Suppose that we have K worker slots. A worker slot is a virtual construction for the sake of modeling that does not have to exist in reality. Generally, each slot is empty until it's occupied by a worker. The XEC service is said to be live when there are at least $k_{\min}$ workers occupying $k_{\min}$ worker slots out of the total K slots. A worker arrives to a worker slot, provides service for the duration it spends occupying the slot while taking a brief vacation after each service, then it ultimately churns for some reason. In this work, we restrict the causes of its churning to only achieving some target profit. After the worker churns, the worker slot remains empty until another worker arrives. The service discipline, according to which this worker functions during its time in the worker slot, follows the IVQ queueing discipline discussed in Section III, but instead of designating the duration of the vacation on the bulk of services, the vacation duration is a function of only the current slice's incentive.

We then proceed to perceive the worker slot's activity over time as a binary variable status of the slot. It is initially empty until the first worker arrives, then it remains active until the first worker churns, then it is empty until the second worker arrives and so on. As such, the availability of the k^{th} slot over time, $A_k(t)$ can be expressed as

$$A_k(t) = \begin{cases} 1 \text{ slot } k \text{ is occuppied by a worker,} \\ 0 \text{ slot } k \text{ is empty.} \end{cases} \tag{14}$$

Figure 9 illustrates an example of the worker slot activity with $N_k(t_{\max}) = 5$.

The total number of workers who have visited the k^{th} slot up to time t, is related to $A_k(t)$ through the derivative

$$N_k(t) = \int_0^t \max\left(\frac{dA_k(s)}{ds}, 0\right) ds \tag{15}$$

Naturally, the status of a worker slot extends to the XEC service availability: at a specific time instant, t, the number of available workers, denoted $N_A(t)$ is the sum of all of the K slots' availabilities, i.e.,

$$N_A(t) = \sum_{k=1}^{K} A_k(t), \tag{16}$$

and thus the service availability, $A(t)$, can be expressed as

$$A(t) = \begin{cases} 1 \; N_A(t) \geq K_{\min} \\ 0 \; N_A(t) \leq K_{\min}. \end{cases} \tag{17}$$

Given the worker slot model, we could view the worker slot as *a server with a zero-length queue* to which workers

arrive with a rate $\lambda_k^{\texttt{slot}}$ and churn with a rate $\mu_{k,w}^{\texttt{slot}}$. The w^{th} worker's arrival time to the slot can be expressed as

$$t_{k,w}^{\texttt{slot}} = wD_k + \sum_{i=1}^{w-1} T_{k,i}^{\texttt{slot}} \tag{18}$$

where D_k is the delay between the w^{th} and $(w+1)^{\text{th}}$ $\forall w \in \mathbb{N}$, $T_{k,i}^{\texttt{slot}}$ is the time spent in slot k by the i^{th} worker. Since the w^{th} worker spends $T_{k,w}^{\texttt{slot}}$ in the slot after its arrival, the churn time is simply

$$t_{k,w}^{\texttt{churn}} = wD_k + \sum_{i=1}^{w} T_{k,i}^{\texttt{slot}}. \tag{19}$$

The last two equations are interesting because they allow the characterization of the distributions of the arrival times if the workers behaved similarly. If the workers behaved in an i.i.d. fashion, then the expressions would just be the w-fold convolution of the delay distribution D_k and T_k. But there is no guarantee that this sort of ergodicity is static across workers. However, if our objective is to *predict* the time which is spent by the w^{th} worker in the kth slot, i.e., predict $T_{k,w}^{\texttt{slot}}$, then we only need information about that worker, and thus, we care about only the time a single worker spends. In the following subsection, we use this piece of information, in combination with the IVQ model to derive a closed form expression for the time spent by a worker in the slot.

B. WORKER'S TIME IN SLOT: IVQ, CTMCS AND MEMORYLESSNESS

In this subsection, we look at the behaviour of the worker slot from two perspectives: 1) the perspective of the worker occupying the slot itself and how they perform under the IVQ discipline and; 2) the perspective of the worker slot as a server with zero-length queue mentioned in the previous subsection. The benefit of using CTMCs lies in the Markovian property that the time spent in each state is independent from the time spent in any other state. The nature of the system itself: the fact that workers process slices that are computationally miniscule and independent from each other, combined with the fact that workers themselves are independent from each other allow us to estimate - given knowledge of the target profit of the workers - the time to be spent by each worker. The expression for the expected time to be spent is meant to be used as a preliminary analysis step once a worker arrives. If parameters change over time, then so does the prediction.

In the first CTMC, or IVQ-CTMC, we know that the IVQ service cycle consists of a service period followed by a vacation period, thus the duration of the n^{th} service cycle can be expressed as

$$\widetilde{S}_{k,w}[n] = S_{k,w}^{\texttt{job}}[n] + \underbrace{r\left(x_{k,w}^{\texttt{job}}[n], \alpha_{k,w}[n]\right)}_{V_{k,w}^{\texttt{job}}[n]} \tag{20}$$

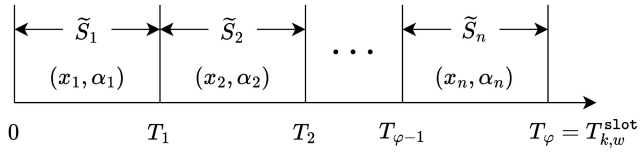

FIGURE 10. First CTMC, IVQ-CTMC, Perspective of Time Spent in Slot: each slice arrives with a specific incentive and preference that induce a service cycle of duration $\widetilde{S}_n$.

where $S_{k,w}^{\texttt{job}}[n] = \frac{1}{\mu_{k,w}^{\texttt{job}}[n]}$ is the service time; $V_{k,w}^{\texttt{job}}[n] = r(x_{k,w}^{\texttt{job}}[n], \alpha_{k,w}[n])$ is the vacation duration; $x_{k,w}^{\texttt{job}}[n]$ is the incentive received for successfully processing the n^{th} slice; and $\alpha_{k,w}[n]$ is the preference parameter at the time of the n^{th} slice. We define the states of the IVQ-CTMC as the combination of incentive attached to the slice and the preference parameter, i.e., the stochastic process is the tuple $\{(x_{k,w}^{\texttt{job}}[n], \alpha_{k,w}[n])\}_{n\geq 1}$, and the time spent in each state is $\widetilde{S}_{k,w}[n]$. We look at the temporal progression of the worker's activity in Figure 10. We see that after some number of successful service cycles, φ, the worker churns. The time spent in the system after the n^{th} slice has been processed is equivalent to the sum of the durations of the service cycles spent. In other words, we have that

$$T_{k,w}[n] = \sum_{i=1}^{n} \widetilde{S}_{k,w}[i] \tag{21}$$

which allows us to define the time spent after the φ^{th} slice, which is the time spent in the system as,

$$T_{k,w}^{\texttt{slot}} = T_{k,w}[\varphi] = \sum_{n=1}^{\varphi} \widetilde{S}_{k,w}[n] \tag{22}$$

But we have little information about φ. To that end, we resort to the other CTMC, the worker slot CTMC. What it tells is that the time spent by the w^{th} worker to visit slot k is exponentially distributed with parameter $\mu_{k,w}^{\texttt{slot}}$, i.e., the time spent by each worker is $T_{k,w}^{\texttt{slot}} = 1/\mu_{k,w}^{\texttt{slot}}$. This allows us to state that the sum in Eq. (22) exponentially distributed and that the time spent is

$$\begin{aligned}\frac{1}{\mu_{k,w}^{\texttt{slot}}} &= \mathbb{E}\left[T_{k,w}^{\texttt{slot}}\right] = \mathbb{E}\left[\sum_{n=1}^{\varphi} \widetilde{S}_{k,w}[n]\right] \\ &= \mathbb{E}\left[\sum_{n=1}^{\varphi}\left(S_{k,w}^{\texttt{job}}[n] + \underbrace{r\left(x_{k,w}^{\texttt{job}}[n], \alpha_{k,w}[n]\right)}_{V_{k,w}^{\texttt{job}}[n]}\right)\right]\end{aligned} \tag{23}$$

The expression in Eq. (23) can be further simplified. The first simplification relates to the preference rate. Since the expression needs to be evaluated for a single transient worker, we proceed on the assumption that we kep $\alpha_{k,w}[n] = \alpha_{k,w}$ is constant for all slices. We further assume that the current observed slice service rate for the worker remains as it is, i.e., $S_{k,w}^{\texttt{job}}[n] = \frac{1}{\mu_{k,w}^{\texttt{job}}[n]} = S_{k,w}$ $\forall n$ is constant. We also

consider that the incentive payments for each of the slices are identically and independently distributed, as over the coherence duration of an extreme edge system the service provider has no reason to treat each slice differently from the other, due to the principle of indifference. This allows us to establish a uniform ergodicity where

$$\{x_{k,w}^{\texttt{job}}[n]\}_{n\geq 1} = \{x[n]\}_{n\geq 1}, \forall n : x[n] \sim \text{Unif}(x_{\min}, x_{\max}) \implies \text{dist}(x[i]) = \text{dist}(x[j])\ \forall i \neq j. \tag{24}$$

In other words, that the stochastic process comprising the sequence of incentives is ergodic, with the sample at any time instance follows uniform distributed between $x_{\min}$ and $x_{\max}$ and has an average value $\bar{x}$. While this is true in REC systems, it is important to note that this assumption is valid for the transient set of CTMCs used to predict the time spent in the slot.

Finally, as previously mentioned in earlier sections, we assume that the worker churns after achieving some target profit. Thus, we estimate φ, the number of service cycles for which the worker stays, as the number of average incentive payments until the w^{th} worker in the k^{th} slot achieves their target profit, i.e.,

$$\varphi \cong \frac{x_{k,w}^{\text{target}}}{\bar{x}}. \tag{25}$$

Of course, φ could vary significantly if the worker is churning for a reason other than achieving their target profit. However, in this performance model, we look at the time spent in the system in a ceteris paribus manner (keeping all other things constant) with respect to incentives. As such, we can thus simplify Eq. (23)

$$\begin{aligned}\mathbb{E}\big[T_{k,w}^{\texttt{slot}}\big] &= \mathbb{E}\left[\sum_{n=1}^{\varphi}\big(S_{k,w} + r\big(x_{k,w}, \alpha_{k,w}\big)\big)\right] \\ &= \varphi S_{k,w} + \varphi\mathbb{E}\big[r\big(x_{k,w}, \alpha_{k,w}\big)\big] \\ &= \varphi\big(S + \varphi\mathbb{E}\big[r\big(x_{k,w}, \alpha_{k,w}\big)\big]\big).\end{aligned} \tag{26}$$

We then evaluate the expected value of the IVF to obtain

$$\begin{aligned}\mathbb{E}[r(x,\alpha)] &= \int_{-\infty}^{\infty} r(s,\alpha) f_x(s)\, ds \\ &= \begin{cases} \int_{-\infty}^{\infty}[(1+\alpha)r_{\text{lnr}}(s) - \alpha r_{\text{cvx}}(s)]f_x(s)\, ds & -1 \leq \alpha \leq 0 \\ \int_{-\infty}^{\infty}[(1-\alpha)r_{\text{lnr}}(s) + \alpha r_{\text{ccv}}(s)]f_x(s)\, ds & 0 < \alpha \leq 1 \end{cases} \\ &= (1+\alpha)\left(\frac{V_{\min}+V_{\max}}{2}\right) - \alpha\left(\frac{V_{\min}V_{\max}\log\left(\frac{V_{\max}}{V_{\min}}\right)}{V_{\max}-V_{\min}}\right).\end{aligned} \tag{27}$$

It can be seen from Eq. (27) for a rational IVF, the α-parameter controls a mixture of a linear term of $V_{\min}$ and $V_{\max}$ and a non-linear term $V_{\min}V_{\max}\log(V_{\max}/V_{\min})/(V_{\max}-V_{\min})$. This allows us to express the time spent in the k^{th} worker slot by the w^{th} worker as

$$\mathbb{E}\big[T_{k,w}^{\texttt{slot}}\big] = \frac{1}{\mu_{k,w}^{\texttt{slot}}}$$

TABLE 1. Simulation parameters.

Parameter	Description	Value
$S = 1/\mu$	Service time, exponentially distributed	$1/70$
$\widetilde{S} = 1/\lambda$	Duration of service cycle	$1/25$
$V_{\min}$	Minimum vacation duration	$10\% \cdot \widetilde{S}$
$V_{\max}$	Maximum vacation duration	$50\% \cdot \widetilde{S}$
$X_{\min}$	Minimum TQI	5
$X_{\max}$	Maximum TQI	10
n	Number of jobs contributing to TQI	40
$x_{\min}$	Minimum Job Incentive	$X_{\min}/n$
$x_{\max}$	Maximum job incentive	$X_{\max}/n$
α	α-preference parameter of IVF $r(x,\alpha)$	-0.5
$\bar{x}$	Average job incentive	$\frac{x_{\min}+x_{\max}}{2}$
X^{target}	Worker's target incentive	100

$$= \frac{x_{k,w}^{\text{target}}}{\bar{x}}\left[S_{k,w} + (1+\alpha_{k,w})\left(\frac{V_{\min}+V_{\max}}{2}\right) - \alpha_{k,w}\left(\frac{V_{\min}V_{\max}\log\left(\frac{V_{\max}}{V_{\min}}\right)}{V_{\max}-V_{\min}}\right)\right], \tag{28}$$

which is a function of the worker's target profit $x_{k,w}^{\texttt{target}}$, the average incentive payment per slice $\bar{x}$, the worker's service rate $S_{k,w}$, the dynamic preference parameter $\alpha_{k,w}$, and the static IVF parameters $V_{\min}$, $V_{\max}$. It is important to note that the average incentive, $\bar{x}$ is dictated by the distribution of the incentive payments and the incentive bound parameters, $x_{\min}$ and $x_{\max}$.

While such a model might seem reductional, it is quite the contrary, as $T_{k,w}^{\texttt{slot}}[n]$ itself varies after each slice served and so do the CTMCs that allow us to predict it. In other words, each instant in time has its own CTMC and its own prediction of the time spent by the w^{th} worker in the k^{th} slot. This makes this model a power tool that can be used in slotted time optimization techniques.

V. RESULTS AND DISCUSSIONS

A. SIMULATION SETUP

To analyze and study the performance of the REC system, we employ monte-carlo simulations based on the closed-form metrics derived in Section III and validate the real-time dynamics proposed in Section IV. The simulations are designed to capture the behavior of worker slots under various operational scenarios. Detailed parameters, including service demand rates, worker device characteristics, and incentive structures, are outlined in Table 1. These parameters are chosen to reflect realistic REC environments and to explore the performance implications of different system configurations. The following results and analyses are grounded in this simulated environment, providing a comprehensive view of the REC and IVQ model efficacy.

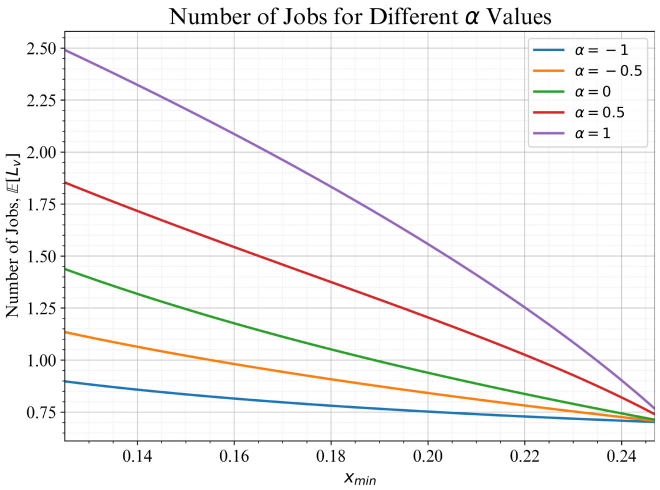

FIGURE 11. Number of slices being processed by the worker for different maximum slice incentive, $x_{\min}$.

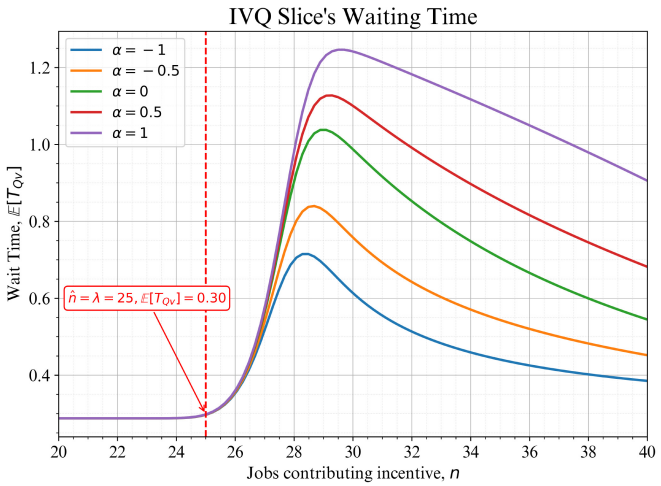

FIGURE 12. Slice's waiting time in queue vs. the number of slices contributing to the TQI.

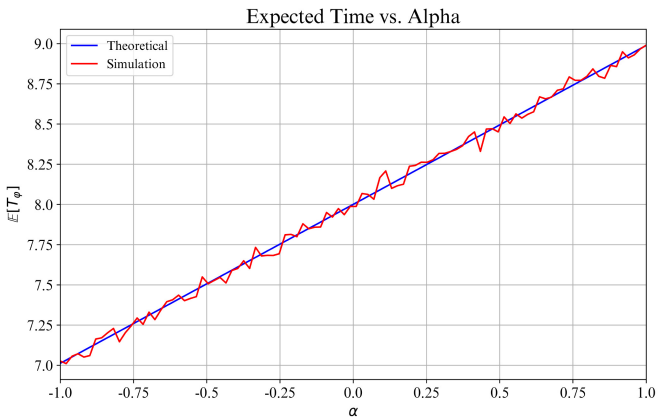

FIGURE 13. Expected time spent by an XEC worker in the system for a varying α.

B. IVQ: SINGLE WORKER PERFORMANCE

As previously discussed in Section III, we analyze the average performance of a worker operating under IVQ discipline. The IVF is a long-term one as it is defined in terms of the TQI, which is the lump sum of incentives contributed by n jobs. The IVF also gives us a one-to-one relationship between incentives and vacation durations. As such, we inspect the influence of incentives through two relationships: 1) how the minimum incentive parameter, $x_{\min}$, influences the average number of slices in the worker's queue, and; 2) how the number of slices to be processed n impacts the waiting time (by impacting the TQI). We look at different α's for both relationships. For both relationships, we do not alter $r(x, \alpha)$.

Figure 11 presents the relationship between $x_{\min}$ (from the distribution function in Eq. (10)) and the average number of slices in queue, $\mathbb{E}[L_v]$, (Eq. (2)), with $\mathbb{E}[V]$ and $\mathbb{E}[V^2]$ being the transcendental expressions in Eq. (2). By increasing the incentive for each slice processed, $x_{\min}$, the number of slices in the system decreases as slices are processed quicker, which is natural: more incentives cause less vacations, and thus more slices processed. Having a low α, however, can keep the number of slices to a minimum, but this impacts the worker's vacations and might influence the retention of workers and cause them to churn sooner, or to never return. In that sense, being lenient and attributing high value to their marginal vacations, increases their long-term productivity over time.

On the other hand, Figure 12 tackles the relationship between $\mathbb{E}[T_{Qv}]$ (from Eq. (3)) and the number of slices contributing the incentive n (from Eq. (10)). This result is interesting because it shows how latency - reflected in the waiting time for each slice - varies with the amount of workload the worker is receiving. Furthermore, the IVF impacts this curve significantly. For a slice arrival rate, $\lambda = 25$, the waiting time remains at n/μ until $n = \lambda$. Afterwards, it keeps increasing to a peak, then it falls down. Prior to the peak, the amount of incentives has not yet influenced the duration of vacations, but as n increases, so does the TQI, and as a consequence around $n = 29$, the wait time decreases as the duration of vacations becomes less. The value of n at which $\mathbb{E}[T_{Qv}]$ is maximum is theoretically possible to obtain in closed-form, but it involves a hypergeometric series stemming from the Irwin-Hall distribution, which can be obtained algorithmically using techniques such as Gosper's and Zeilberger's algorithms [33]. Nevertheless, these bounds show the theoretical possibility and feasibility of an XEC worker to provide service for an XEC service.

C. IVQ: SERVICE PROVIDER PERSPECTIVE

In Figure 13, we show the average time spent in the system versus α for both closed form and simulation. It can be seen that the numerical simulation validates the closed-form result. It is also expected: if XEC workers take vacations with high α, then the net sum of their vacations will increase the time they need to achieve their target profit.

In both Figures 14 and 15, we see the impact of the vacation parameters on the time spent in the system for

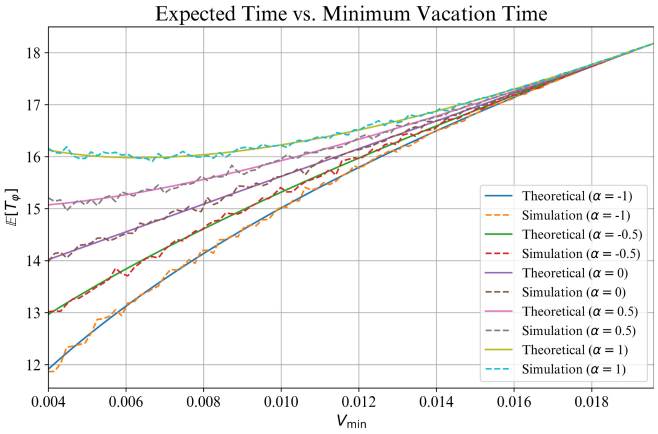

FIGURE 14. Expected time spent by an XEC worker in the system for a varying minimum vacation $V_{\min}$.

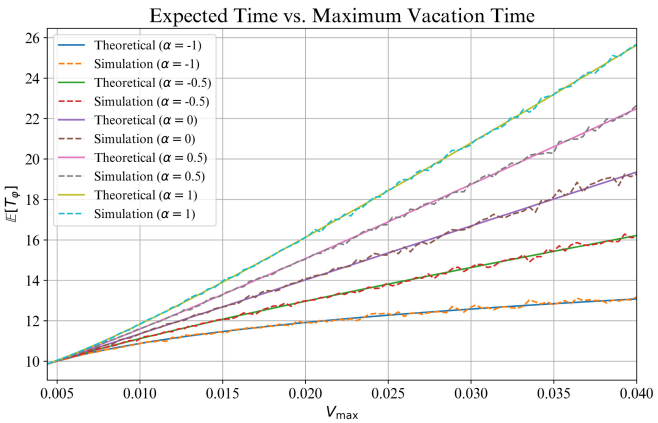

FIGURE 15. Expected time spent by an XEC worker in the system for a varying maximum vacation $V_{\max}$.

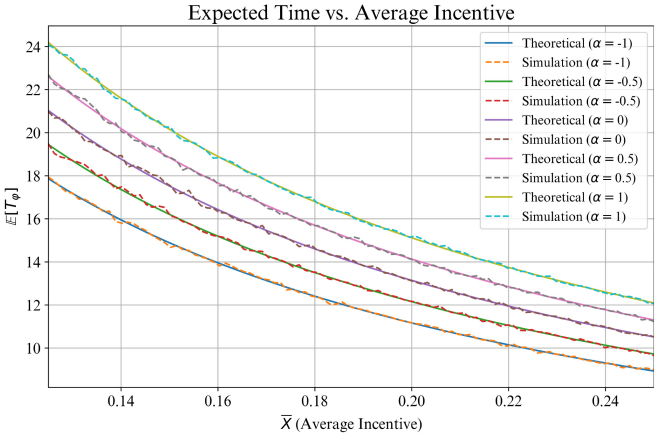

FIGURE 16. Expected time spent by an XEC worker in the system for a varying average slice incentive $\overline{x}$.

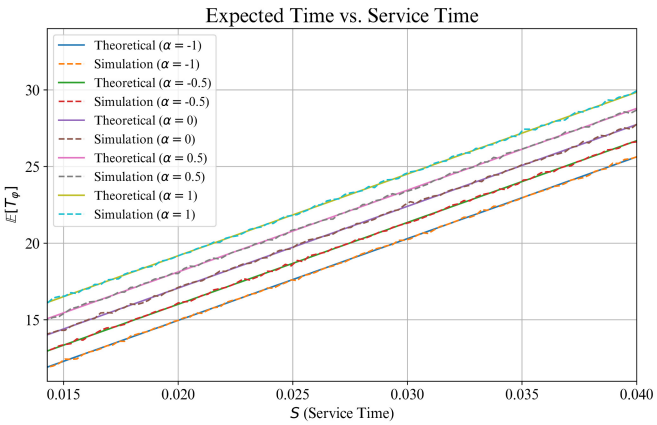

FIGURE 17. Expected time spent by an XEC worker in the system for a varying service rate $\mathbb{E}[S] = 1/\mu$.

different α. Similar to the behaviour observed in Figure 13, increasing the vacation duration increases the time spent in the system by the worker. But it is also noticeable that the time spent in the system is shorter for lower α due to the fact that the XEC workers are stressed by the IVF function to reduce their vacations (or they would lose incentives). It can also be seen from both figures that the wider the difference between $V_{\max}$ and $V_{\min}$, the wider the difference between the expected time in the system for high α and low α. This is due to the amount of variability of vacation duration and how high α's are worker preferring in terms of the amount of load. High α XEC workers can achieve their target profit without having to sacrifice much of their vacation time. In that sense, the value of α relates to the multitenant balance between XEC service work and XEC worker's own work.

Figure 16 shows how the expected time spent in the system reduces with higher incentives (for the same $r(x, \alpha)$). This shows that increasing incentives can get more work done in a shorter time period, but it would also mean that the workers would achieve their target quickly and churn.

Finally, Figure 17 shows that the expected time in the server increases as the service time increases, which is a natural result, as it would mean a reduction in the service rate and thus more time elapsing for the service provision and consequently a longer service cycle.

These results show that there is a trade-off between work done and worker retention and return. Assuming that there are other similar XEC service providers in an REC market, XEC workers will gravitate towards the more convenient systems that do not impact the experience of their owners while making their target profits.

VI. CONCLUSION

This paper presented an examination of Reward Edge Computing (REC) within the broader context of Extreme Edge Computing (XEC), utilizing the Incentive-Vacation Queueing (IVQ) model. Our analysis detailed the operational dynamics of REC systems, focusing on individual worker performance and the overall system's ability to maintain service provision. The simulations provided a structured view of how incentives impact worker behavior and system efficiency.

The findings indicate that while REC and IVQ can enhance distributed computing, there are inherent trade-offs

and challenges that need careful consideration, especially regarding incentive distribution and system reliability. Future efforts may explore more refined incentive strategies and robust system designs to better manage these complexities. As EC and XEC environments become more prevalent, understanding and improving upon these models is crucial for developing effective distributed computing solutions.

REFERENCES

[1] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Incentive-vacation queueing for extreme edge computing systems," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 94–99.

[2] H. T. Malazi et al., "Dynamic service placement in multi-access edge computing: A systematic literature review," *IEEE Access*, vol. 10, pp. 32639–32688, 2022.

[3] A. Araldo, A. D. Stefano, and A. D. Stefano, "Resource allocation for edge computing with multiple tenant configurations," in *Proc. 35th Annu. ACM Symp. Appl. Comput.*, 2020, pp. 1190–1199.

[4] K. Abbas, Y. Cho, A. Nauman, P. W. Khan, T. A. Khan, and K. Kondepu, "Convergence of AI and MEC for autonomous IoT service provisioning and assurance in B5G," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 2913–2929, 2023.

[5] D. Milojicic, "The edge-to-cloud continuum," *Computer*, vol. 53, no. 11, pp. 16–25, Nov. 2020.

[6] N. Fernando, C. Arora, S. W. Loke, L. Alam, S. L. Macchia, and H. Graesser, "Towards human-centred crowd computing: Software for better use of computational resources," 2023, *arXiv:2302.05617*.

[7] M. A. Rahman, M. M. Rashid, M. S. Hossain, E. Hassanain, M. F. Alhamid, and M. Guizani, "Blockchain and IoT-based cognitive edge framework for sharing economy services in a smart city," *IEEE Access*, vol. 7, pp. 18611–18621, 2019.

[8] R. S. Alonso, I. Sittón-Candanedo, S. Rodríguez-González, Ó. García, and J. Prieto, "A survey on software-defined networks and edge computing over iot," in *Proc. Int. Conf. Pract. Appl. Agents Multi-Agent Syst.*, 2019, pp. 289–301.

[9] S. Yang, F. Li, S. Trajanovski, R. Yahyapour, and X. Fu, "Recent advances of resource allocation in network function virtualization," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 2, pp. 295–314, Feb. 2021.

[10] S. Becker, F. Schmidt, and O. Kao, "EdgePier: P2P-based container image distribution in edge computing environments," in *Proc. IEEE Int. Perform., Comput., Commun. Conf. (IPCCC)*, 2021, pp. 1–8.

[11] A. J. Ferrer, J. M. Marques, and J. Jorba, "Ad-Hoc edge cloud: A framework for dynamic creation of edge computing infrastructures," in *Proc. 28th Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2019, pp. 1–7.

[12] S. Yazdani, N. Ramzan, and P. Olivier, "Enhancing edge computing with unikernels in 6G networks," in *Proc. IEEE 34th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, 2023, pp. 1–6.

[13] M. S. Allahham, A. Mohamed, A. Erbad, and H. Hassanein, "On the modeling of reliability in extreme edge computing systems," in *Proc. 5th Int. Conf. Commun., Signal Process., Appl. (ICCSPA)*, 2022, pp. 1–6.

[14] A. J. Ferrer, *Beyond Edge Computing: Swarm Computing and Ad-Hoc Edge Clouds*. Cham, Switzerland: Springer, 2023.

[15] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Incentive-vacation queueing for edge crowd computing," *IEEE Internet Things J.*, early access, Dec. 27, 2023, doi: 10.1109/JIOT.2023.3347442.

[16] J. Sun, W. Gan, H.-C. Chao, P. S. Yu, and W. Ding, "Internet of Behaviors: A survey," *IEEE Internet Things J.*, vol. 10, no. 13, pp. 11117–11134, Jul. 2023.

[17] M. T. Dabiri, M. Hasna, N. Zorba, and T. Khattab, "Optimal trajectory and positioning of UAVs for small cell HetNets: Geometrical analysis and reinforcement learning approach," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 2667–2683, 2023.

[18] M. Tenemaza, T. Javier, T. Rodney, and S. Luján-Mora, "Architecture for a services system based on sharing economy," in *Advances in Human Factors and Systems Interaction*. New York, NY, USA: Springer, 2020.

[19] A. Zavodovski, S. Bayhan, N. Mohan, P. Zhou, W. Wong, and J. Kangasharju, "DeCloud: Truthful decentralized double auction for edge clouds," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2019, pp. 2157–2167.

[20] F. Freitag, L. Navarro, M. Selimi, and R. P. Centelles, "End user-managed service deployments in microclouds at the network edge," in *Proc. IEEE 8th Global Conf. Consum. Electron. (GCCE)*, 2019, pp. 549–550.

[21] J. M. García, P. Fernández, A. Ruiz-Cortés, S. Dustdar, and M. Toro, "Edge and cloud pricing for the sharing economy," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 78–84, Mar./Apr. 2017.

[22] D. Li, R. Hao, Z. Wei, and J. Liu, "A budget constraint incentive mechanism based on risk preferences of collaborators in edge computing," *Mathematics*, vol. 12, no. 3, p. 496, 2024.

[23] M. Diamanti, P. Charatsaris, E. E. Tsiropoulou, and S. Papavassiliou, "Incentive mechanism and resource allocation for edge-fog networks driven by multi-dimensional contract and game theories," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 435–452, 2022.

[24] L. Yin, J. Luo, C. Qiu, C. Wang, and Y. Qiao, "Joint task offloading and resources allocation for hybrid vehicle edge computing systems," *IEEE Trans. Intell. Transp. Syst.*, early access, Jan. 25, 2024, doi: 10.1109/TITS.2024.3351635.

[25] L. Wang, L. Hou, S. Liu, Z. Han, and J. Wu, "Reinforcement contract design for vehicular-edge computing scheduling and energy trading via deep Q-network with hybrid action space," *IEEE Trans. Mobile Comput.*, early access, Nov. 2, 2023, doi: 10.1109/TMC.2023.3329643.

[26] "DCP Platform." Distributive. Accessed: Sep. 18, 2023. [Online]. Available: https://distributive.network/platform

[27] H. Takagi, *Queueing Analysis: Discrete-Time Systems*. Amsterdam, The Netherlands: North-Holland, 1991.

[28] N. Tian and Z. G. Zhang, *Vacation Queueing Models: Theory and Applications*. New York, NY, USA: Springer, 2006.

[29] L. Tadj and G. Choudhury, "Optimal design and control of queues," *TOP*, vol. 13, pp. 359–412, Dec. 2005.

[30] S. B. Azmy, N. Zorba, and H. S. Hassanein, "Queueing analysis of incentive-based extreme edge service systems," in *Proc. IEEE Global Commun. Conf. (Globecom)*, 2023, pp. 297–301.

[31] J. E. Marengo, D. L. Farnsworth, and L. Stefanic, "A geometric derivation of the Irwin-Hall distribution," *Int. J. Math. Math. Sci.*, vol. 2017, Sep. 2017, Art. no. 3571419.

[32] L. C. Zhao, C. Q. Wu, and Q. Wang, "Berry-Esseen bound for a sample sum from a finite set of independent random variables," *J. Theor. Probab.*, vol. 17, pp. 557–572, Jul. 2004.

[33] S. Chen, F. Chyzak, R. Feng, G. Fu, and Z. Li, "On the existence of telescopers for mixed hypergeometric terms," *J. Symb. Comput.*, vol. 68, pp. 1–26, May/Jun. 2015.