An Online Mean Field Approach for Hybrid Edge Server Provision

Zhiyuan Wang
The Chinese University of Hong Kong
zhiyuanwang@cuhk.edu.hk

Jiancheng Ye
Network Technology Lab
Huawei Technologies Co., Ltd.
yejiancheng@huawei.com

John C.S. Lui
The Chinese University of Hong Kong
cslui@cse.cuhk.edu.hk

ABSTRACT

The performance of an edge computing system primarily depends on the edge server provision mode, the task migration scheme, and the computing resource configuration. This paper studies how to perform dynamic resource configuration for hybrid edge server provision under two decentralized task migration schemes. We formulate the dynamic resource configuration as a multi-period online cost minimization problem, aiming to jointly minimize the performance degradation (i.e., execution latency) and the operation expenditure. Due to the stochastic nature, one can only observe the system performance for the currently installed configuration, which is also known as the partial feedback. To overcome this challenge, we derive a deterministic mean field model to approximate the large-scale stochastic edge computing system. We then propose an online mean field aided resource configuration policy, and show that the proposed policy performs asymptotically as good as the offline optimal configuration. Numerical results show that the mean field model can significantly improve the convergence speed in the online resource configuration problem. Moreover, our proposed policy under the two decentralized task migration schemes considerably reduces the operating cost (by 23%) and incurs little communication overhead.

CCS CONCEPTS

• **Networks** → **Network management**; • **Theory of computation** → **Random network models**.

KEYWORDS

Edge computing, mean field model, online learning, resource configuration, load balancing

ACM Reference Format:
Zhiyuan Wang, Jiancheng Ye, and John C.S. Lui. 2021. An Online Mean Field Approach for Hybrid Edge Server Provision. In *The Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '21), July 26–29, 2021, Shanghai, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3466772.3467042

This work is supported in part by the GRF 14201819 and CUHK: 6905407.

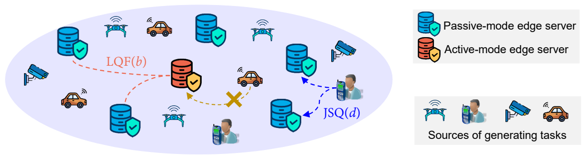


Figure 1: An illustrative scenario with $M = 10$ sources, $N = 5$ passive-mode edge servers, and $K = 1$ active-mode edge server. Here the edge servers can be interconnected via a metropolitan-area-network (MAN) or a local-area-network (LAN).

1 INTRODUCTION

1.1 Background and Motivation

The recent proliferation of smart city and Internet-of-Things (IoT) applications are driving a rapid growth of connected devices (e.g., IoT sensors and mobile users). These devices are the *sources* that repeatedly generate computing tasks of various delay-sensitive services. Edge computing, providing computation resource in close proximity to the sources, is a promising paradigm to reduce the latency for many network applications. The performance of an edge computing system depends primarily on three factors: (1) edge server provision, (2) task migration scheme, and (3) dynamic resource configuration, which are the main focus in this paper.

1.1.1 Edge Server Provision. The computing resource is typically the edge servers in close proximity to the sources. It could be a micro datacenter or servers attached to an access point [1]. In general, edge servers may function in passive mode or active mode:

- The edge servers in *passive mode* will admit and process the tasks offloaded by sources (e.g., IoT sensors), which corresponds to the studies on sources' task offloading problem (e.g., [2][3]).
- The edge servers in *active mode* will not directly admit the tasks offloaded by sources, but will assist passive-mode edge servers to process the waiting tasks. The interaction between the active-mode and passive-mode edge servers corresponds to the studies on collaborative edges (e.g., [6]-[9]).

Fig. 1 shows an illustrative scenario with $M = 10$ sources, $N = 5$ passive-mode edge servers, and $K = 1$ active-mode edge server. In this paradigm, a source's computing task first reaches a passive-mode edge server, and may be extracted by an idle active-mode edge server. The task execution progress is closely related to the adopted task migration scheme, which is discussed next.

1.1.2 Task Migration Scheme. The task migration scheme depends on the aforementioned edge server provision. An appropriate task migration scheme can significantly reduce the task execution latency, thus improves the Quality of Experience (QoE) of sources. There have been studies on task migration under the coordination of the network operator who has the global information (e.g., [8][9]). In practice, however, it is costly to keep track of all the required information globally and persistently, especially when the number of sources M is large. This motivates us to consider a more practical migration scheme under the hybrid edge server provision. Specifically, we focus on two load-balancing policies, i.e., Join-Shortest-Queue (JSQ) and Longest-Queue-First (LQF). As shown in Fig. 1, the two migration schemes work as follows:

- JSQ(d) with $d \in \{1, 2, ..., N\}$: Upon generating a task, the source probes d passive-mode edge servers uniformly at random, and migrates the task to the least loaded one among the d samplings.
- LQF(b) with $b \in \{1, 2, ..., N\}$: Whenever an active-mode edge server has any capacity, it probes b passive-mode edge servers uniformly at random, and extracts a waiting task from the one with the heaviest workload among the b samplings.

Note that as the parameters d and b increase, JSQ(d) and LQF(b) are becoming the classic join-the-shortest-queue and serve-the-longest-queue discipline, respectively. The previous studies (e.g., [17][18]) have shown that a small value can already ensure a good performance in the heavy-demand scenario. However, it is not clear yet whether this still holds when the operator needs to configure the available computing resource in a dynamic fashion (i.e., the third focus in this paper).

1.1.3 Resource Configuration. Despite the extensive studies on the task migration, the computing resource allocation has been overlooked in edge computing. A fixed resource configuration will inevitably result in either a low resource utilization or a poor performance due to the workload fluctuation of the sources. Therefore, it is crucial to configure the computing resource in a dynamic fashion for the edge servers. Note that the *"dynamic configuration"* naturally relies on the underlying computation demand and the expenditure of running the resource. Both of them are priori unknowns and possibly time-varying in practice. This means that it is imperative to study "online resource configuration" for edge servers. The above discussions lead to the following key questions in this paper:

Question 1. *How to optimize the resource configuration under the hybrid edge servers provision in a dynamic fashion?*

Question 2. *Can one harness the benefits from JSQ(d) and LQF(b) even when d and b are small in the dynamic scenario?*

This paper will introduce a mean field model to estimate the large-scale edge network, and proposes an *online mean field aided configuration policy.* We believe the results in this paper could lay the groundwork for using mean field theory to analyze and design dynamic hybrid edge server provision in edge computing.

1.2 Main Results and Key Contributions

This paper studies the multi-period operation of a large-scale edge computing system with hybrid edge server provision, and aims to minimize the total operating cost in an online fashion. At the beginning of each period (e.g., every hour), the network operator determines the resource configuration for the edge servers in two modes. During this period, the passive-mode edge servers receive computation tasks of the sources under JSQ(d). The active-mode edge servers assist the passive-mode ones in executing the waiting tasks according to LQF(b). At the end of each period, the operator observes the cost of operating the network, and then determines the resource configuration for the next period. Due to the stochastic nature of the large-scale edge network, it is difficult to anticipate the cost of other configuration decisions that were not adopted. Therefore, the dynamic resource configuration problem naturally exhibits the *partial feedback* issue.

The main results and key contributions are as follows:

- *Problem Formulation:* We investigate the dynamic resource configuration for hybrid edge server provision under two decentralized task migration schemes. The goal is to jointly minimize the operation expenditure and the performance degradation in an online fashion. To the best of our knowledge, this is the first study on dynamic resource configuration for edge computing.
- *Resolving Partial Feedback via Mean Field:* We introduce a deterministic mean field model to represent the large-scale stochastic edge network. We show that the original stochastic system converges to the deterministic mean field model as the system size and the period duration increase. As far as we know, we are the first to integrate mean field theory and online learning with the partial feedback issue.
- *Online Mean Field aided Configuration Policy:* We devise an online mean field aided configuration policy, which first discretizes the metric configuration space, and then continuously explores and exploits the finite configuration candidates. We show that the cost incurred by our proposed policy is less than the sum of a constant multiple of the offline minimal cost and an extra constant.
- *Performance Evaluation:* We carry out extensive evaluation using real-world electricity market data. Results show that the mean field model significantly improves the convergence speed in the online resource configuration. Moreover, our proposed mean field aided configuration policy under the migration schemes JSQ(2) and LQF(2), considerably reduces the total operating cost (by 23%) and incurs little communication overhead.

The remainder of this paper is as follows. Section 2 reviews related literature. Section 3 introduces system model and problem formulation. Section 4 derives a mean field model. Section 5 proposes the online resource configuration policy. Section 6 provides the numerical results. We conclude this paper in Section 7.

2 LITERATURE REVIEW

We will review three streams of studies related to this paper, including edge computing, load balancing, and online learning.

2.1 Edge Computing

There have been many excellent studies on edge computing [1]. We will focus on the recent studies that are mostly related to ours.

2.1.1 *Edge Server Provision & Task Migration.*

The passive-mode edge server provision in this paper is related to the previous studies on the task offloading problems. Chen *et al.* in [2] study this problem as a potential game. The later studies further take into account the energy-efficiency (e.g., [3]) and the service caching (e.g., [4][5]). The active-mode edge server provision is related to edge collaboration. Sahni *et al.* in [6] study how to jointly schedule the tasks and the network flows in the collaborative edge computing. Galanopoulos *et al.* in [7] consider the cooperative IoT data analytics on the edge nodes. Tang *et al.* in [8] propose a general 3C resource sharing framework. Poularakis *et al.* in [9] formulate a static service placement and request routing problem, which can generalize several previous studies on edge computing.

2.1.2 *Resource Configuration.*

The resource configuration problem has not been widely studied in edge computing. A few studies (e.g., [10–12]) consider a static scenario. Zhang *et al.* in [10] investigate how to allocate the computing resource of edge and cloud servers, and propose a distributed optimization framework. Chen *et al.* in [11] focus on energy consumption and study the optimal allocation of both computation and communication resources. Meskar *et al.* [12] focus on the fairness issue of multi-resource allocation for edge servers. Zhou *et al.* in [13] study the dynamic server provision in IoT data streaming, but do not consider the resource configuration.

This paper differs from the above works in two aspects. First, we focus on the dynamic resource configuration with demand uncertainty, which is seldom studied in edge computing. Second, we consider the hybrid edge server provision together with two decentralized task migration schemes, which can generalize some previous studies into a unified framework. The two aspects above are mutually connected and render the direct analysis of this problem highly non-trivial.

2.2 Load Balancing in Multi-Server System

The task migration scheme in this paper is related to the load-balancing policies in multi-server systems, which studies how the dispatcher routes the incoming jobs to different servers [14]. In this problem, mean field approximation is widely used to investigate the steady state of the limiting system. Mitzenmacher in [15] characterizes the average response time under JSQ(2) for the M/M/1 system, which unveils an exponential improvement compared to the random dispatching. Later works extend this study from the perspective of batch-job arrival (e.g., [16]), general serving time distribution (e.g., [17]), resource budget (e.g., [18]), and resource pooling (e.g., [19]). Furthermore, Ying in [20] explicitly derives the approximation error in terms of the system size.

In this paper, the task migration scheme from sources to the passive-mode edge severs share a similar idea with those in above studies (e.g., [15]). The migration scheme for the active-mode edge servers is a generalization of the resource pooling study in [19].

2.3 Online Learning Algorithm

To solve the dynamic resource configuration, this paper proposes an online mean field aided policy, which consists of the discretization and learning phases. Specifically, the learning phase is based on the multiplicative weight update (MWU) method [21], which originates from the classic problem "prediction with expert advices" [22]. However, the presence of discretization phase renders the regret analysis of the proposed algorithm substantially different from the standard procedure of the classic MWU method.

3 SYSTEM MODEL

We consider a set $\mathcal{M} = \{1, 2, ..., M\}$ of sources (e.g., IoT sensors or mobile users), which repeatedly generate computing tasks. The edge computing operator adopts a hybrid edge server provision in proximity to the sources. Specifically, there are a set $\mathcal{N} = \{1, 2, ..., N\}$ of passive-mode edge servers and a set $\mathcal{K} = \{1, 2, ..., K\}$ of active-mode edge servers. The system works as follows:

- The sources can offload their computing tasks to the N passive-mode edge servers, but cannot access the active-mode edge servers on their own.
- The K active-mode edge servers will assist the N passive-mode edge servers to process the waiting tasks.

We define $\theta \triangleq N/M$ and $\eta \triangleq K/M$. Accordingly, the tuple (θ, η) represents the operator's hybrid edge server provision, which depends on operator's infrastructure deployment. For example, if the operator decides to deploy 10 passive-mode edge servers and 1 active-mode edge server for every 100 IoT sensors within a region, then we have $(\theta, \eta) = (0.1, 0.01)$. We will study how the system scales as M increases, which captures the rapid growth of the delay-sensitive applications in the future network.

Furthermore, the task generation rate of each source $m \in \mathcal{M}$ could be time-varying and unpredictable in practice. Therefore, given the hybrid edge server provision (θ, η), the operator will dynamically configure the available resource of the edge servers. We consider an operation horizon with a set $\mathcal{T} = \{1, 2, ..., T\}$ of periods (e.g., 1000 hours). Each period $t \in \mathcal{T}$ has the equal time duration δ (e.g., 1 hour), and we let $\tau \in [0, \delta]$ be the time index within each period t. The operator configures the available resource of edge servers at the beginning of each period t, and then the system runs under this configuration until the end of this period.

Next we introduce the network model and characterize the network state in Section 3.1 and Section 3.2, respectively. We then formulate the operator's problem in Section 3.3.

3.1 Network Model

We model the network based on the sources' computation tasks, the migration schemes, and the computing resource.

3.1.1 *Computation Task.*

In period t, source $m \in \mathcal{M}$ will generate multiple computation tasks. We model the stochastic nature of the tasks based on the arriving time and the computing intensity.

- *Arriving Time:* We follow the previous studies (e.g., [4]) and model the task arriving of source $m \in \mathcal{M}$ as a Poisson process at rate λ_t^m in period t. The rate λ_t^m may vary over different periods, capturing the demand fluctuation of source m. Accordingly, we let $\tau_{t[i]}^m \in [0, \delta]$ denote the arriving time of the i-th task of source m in period t. To facilitate our later discussion, we let $\bar{\lambda}_t^{[M]} \triangleq \sum_{m=1}^{M} \lambda_t^m / M$ denote the average rate in period t.
- *Computing Intensity:* The computing intensity of a task represents its complexity and can be roughly measured by the required CPU cycles [2]. We let $l_{t[i]}^m$ denote the computing

intensity of the i-th task of source $m \in \mathcal{M}$ in period t. We follow the previous empirical studies (e.g., [23][24]) and model $l^m_{t[i]}$ as an exponentially distributed random variable with a normalized mean value. Our later analysis can be extended to the general distributions, which will be elaborated at the end of Section 4.

Based on the above discussions, $\phi^m_{t[i]} \triangleq \{\tau^m_{t[i]}, l^m_{t[i]}\}$ represents the i-th computation task of source $m \in \mathcal{M}$ in period t. We let $\boldsymbol{\phi}^m_t \triangleq \{\phi^m_{t[1]}, \phi^m_{t[2]}, ...\}$ denote the task profile of source m in period t. Accordingly, $\boldsymbol{\Phi}_t \triangleq (\boldsymbol{\phi}^m_t : \forall m \in \mathcal{M})$ represents the task profile of the entire system in period t.

3.1.2 Passive-Mode Edge Server Provision. Each source $m \in \mathcal{M}$ can offload its tasks to the N passive-mode edge servers. Different from the previous studies on task offloading (e.g., [2][3]), we focus on a decentralized migration scheme JSQ(d), where $d \in \{1, 2, ..., N\}$. It works as follows:

- Upon generating a task (e.g., $\phi^m_{t[i]}$ at time $\tau^m_{t[i]} \in [0, \delta]$), the source $m \in \mathcal{M}$ inquires about the number of tasks in d passive-mode edge servers, which are uniformly selected at random.
- The source m migrates the task $\phi^m_{t[i]}$ to the one holding the least tasks among the d samplings.

Note that a larger d leads to a better performance, but also increases the communication overhead. Previous studies (e.g., [15][16]) have shown that a small parameter (e.g., $d = 2$) can already ensure a good performance in heavy-demand case. We will further investigate the impact of the parameter d in Section 6.2.

3.1.3 Active-Mode Edge Server Provision. The K active-mode edge servers will assist the passive-mode edge servers to execute the waiting tasks offloaded by sources. Different from the previous studies on edge collaboration (e.g., [6][8]), we focus on a decentralized scheme LQF(b), where $b \in \{1, 2, ..., N\}$. It works as follows:

- Upon being idle, an active-mode edge server $k \in \mathcal{K}$ selects b passive-mode edge servers uniformly at random, and inquires about the number of tasks at the b selected servers.
- The active-mode edge server k will extract a task from the most loaded one among the b samplings according to the FIFO rule.

Note that a larger b leads to a better performance, but increases the communication overhead. Section 6 will show that a small parameter (e.g., $b = 2$) can already ensure a good performance.

3.1.4 Computing Resource. Each edge server is equipped with a certain amount of computing resources. The operator needs to configure the available computing resource (e.g., the number of VMs) at the beginning of each period t without the knowledge of the upcoming tasks $\boldsymbol{\Phi}_t$. We let x_t denotes the CPU frequency (in cycles per second) of each passive-mode edge server. Hence the total available computing resource at the passive-mode edge servers is Nx_t. We let y_t denote the computing resource at each active-mode edge server. Hence the total computing resource at the active-mode edge server is Ky_t. We let $z_t = (x_t, y_t)$ denote the resource configuration in period t. The operator chooses z_t in the metric space $\mathcal{Z}$, which is defined as

$$\mathcal{Z} \triangleq \Big\{(x, y) \Big| \, x_L \le x \le x_H, \; y_L \le y \le y_H \Big\}, \tag{1}$$

where the feasible ranges $[x_L, x_H]$ and $[y_L, y_H]$ depend on the hardware in practice.

3.2 Network Characterization

We characterize the network state and introduce the performance metric based on the above network model.

3.2.1 Network State. In each period t, we let $Q^n_t(\tau) \in \mathcal{B}$ denote the number of tasks in the passive-mode edge server $n \in \mathcal{N}$ at time $\tau \in [0, \delta]$, where $\mathcal{B} \triangleq \{0, 1, ..., \mathrm{B}\}$ and B is the buffer size. Accordingly, $\boldsymbol{Q}_t(\tau) = \{Q^n_t(\tau) \in \mathcal{B} : \forall n \in \mathcal{N}\}$ is the network state at time τ in period t. Note that $\{\boldsymbol{Q}_t(\tau) \in \mathcal{B}^N : \forall \tau \in [0, \delta]\}$ is an N-dimensional continuous time Markov chain (CTMC). We have two-fold elaboration on it:

- Given the migration scheme JSQ(d), $Q^n_t(\tau)$ depends on the task profile of the passive-mode edge server n, as well as the task profiles of other passive-mode edge servers.
- Given the migration scheme LQF(b), $Q^n_t(\tau)$ depends on the resource configuration in both the passive-mode and active-mode edge servers.

To emphasize these dependencies, we will often use $Q^n_t(\tau, z_t, \boldsymbol{\Phi}_t)$ to denote the number of tasks in the passive-mode edge server n at time τ. Accordingly, the *average workload* among the N passive-mode edge servers at time $\tau \in [0, \delta]$ is

$$L^{[\mathrm{N}]}_t(\tau, z_t, \boldsymbol{\Phi}_t) \triangleq \frac{1}{N} \sum_{n=1}^{N} Q^n_t(\tau, z_t, \boldsymbol{\Phi}_t), \tag{2}$$

where the superscript [N] denotes the average is taken over the N passive-mode edge servers. Moreover, the *time-average workload* is

$$L^{[\mathrm{N}][\delta]}_t(z_t, \boldsymbol{\Phi}_t) \triangleq \frac{1}{\delta} \int_0^{\delta} L^{[\mathrm{N}]}_t(\tau, z_t, \boldsymbol{\Phi}_t) \mathrm{d}\tau, \tag{3}$$

where the superscript $[\delta]$ represents that the average is taken over the period duration δ.

Next we elaborate why the time-average workload (3) is a crucial performance metric for the operator to optimize.

3.2.2 Performance Metric. The operator aims to expedite the task execution and improve the Quality of Experience (QoE) of the M sources. Hence the execution latency (i.e., the time that a task spends in the system) is a crucial performance metric for the operator to optimize. Specifically, (3) implies that the time-average number of tasks in all the passive-mode edge servers is $NL^{[\mathrm{N}][\delta]}_t(z_t, \boldsymbol{\Phi}_t)$. The sources' total task arrival rate is $M\bar{\lambda}^{[\mathrm{M}]}_t$. By Little's Law [25], the average execution latency of the tasks $\boldsymbol{\Phi}_t$ in period t is

$$\frac{N \cdot L^{[\mathrm{N}][\delta]}_t(z_t, \boldsymbol{\Phi}_t)}{M \cdot \bar{\lambda}^{[\mathrm{M}]}_t} = \frac{\theta \cdot L^{[\mathrm{N}][\delta]}_t(z_t, \boldsymbol{\Phi}_t)}{\bar{\lambda}^{[\mathrm{M}]}_t}, \tag{4}$$

where $\theta = N/M$ is the passive-mode edge server provision ratio. Therefore, the time-average workload (3) is proportional to the task execution latency. In Section 3.3, we will model the network performance degradation based on it.

3.3 Problem Formulation

In the following, we define the operator's cost and formulate the dynamic resource configuration problem.

3.3.1 Operator's Cost. We characterize the operator's cost based on the operation expenditure and the performance degradation.

Operation expenditure is the monetary cost for the operator. We let $\xi_t^{\mathsf{P}} \in [0, \xi_{max}^{\mathsf{P}}]$ denote the operation expenditure per unit computing resource at the passive-mode edge server in period t. Similarly, we let $\xi_t^{\mathsf{A}} \in [0, \xi_{max}^{\mathsf{A}}]$ denote the operation expenditure per unit computing resource at the active-mode edge server in period t. In practice, ξ_t^{P} and ξ_t^{A} depend on many factors such as the infrastructure management, the energy consumption, and the electricity price. The operator usually does not know $(\xi_t^{\mathsf{P}}, \xi_t^{\mathsf{A}})$ until the end of period t. Mathematically, the configuration $z_t = (x_t, y_t)$ incurs the operation expenditure $\xi_t^{\mathsf{P}} N x_t + \xi_t^{\mathsf{A}} K y_t$ in period t. As we will see later, it is important to investigate how the system changes when the number of sources M increases. Hence the effective metric is the following *average operation expenditure* in period t

$$\frac{\xi_t^{\mathsf{P}} N x_t + \xi_t^{\mathsf{A}} K y_t}{M} = \xi_t^{\mathsf{P}} \theta x_t + \xi_t^{\mathsf{A}} \eta y_t, \tag{5}$$

where $\theta = N/M$ and $\eta = K/M$ represent the operator's hybrid edge server provision.

Performance degradation measures the sources' QoE reduction due to the increase in latency. In each period t, we measure the QoE reduction based on the time-average workload $L_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t)$. Mathematically, we adopt a general formulation and let $G(L)$ denote the degradation given the time-average workload L. Specifically, $G(L)$ is continuous and increasing in L with $G(0) = 0$.

Based on the discussion above, we define the operator's cost in period t as follows:

$$C_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t) \triangleq G\left(L_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t)\right) + \xi_t^{\mathsf{P}} \theta x_t + \xi_t^{\mathsf{A}} \eta y_t, \tag{6}$$

which comprises the performance degradation and the average operation expenditure. The operator can flexibly choose the function $G(\cdot)$ to balance how much it prioritizes the system performance over the monetary expenditure. That is, given an appropriate function $G(\cdot)$, the operator achieves its desired outcome by minimizing the cost (6) over $z_t \in \mathcal{Z}$.

3.3.2 Operator's Problem. The operator determines the resource configuration z_t sequentially at the beginning of period t, aiming to minimize the total cost during the T periods. However, the operator cannot observe the task profile Φ_t and the per-unit operation expenditure $(\xi_t^{\mathsf{P}}, \xi_t^{\mathsf{A}})$ until the end of period t. That is, the operator needs to solve the following online cost minimization problem:

Problem 1 (Online Cost Minimization Problem).

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} C_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t) \\ s.t., \quad & z_t \in \mathcal{Z}, \ \forall t \in \mathcal{T}. \end{aligned} \tag{7}$$

Problem 1 is an online optimization problem. The key challenges for the operator to solve it are two-fold:

- First, Problem 1 exhibits the partial feedback issue in terms of the performance degradation. Specifically, the operator can observe the performance degradation after adopting the configuration z_t, but the operator does not know the performance of other configuration due to the stochastic nature of the system.
- Second, the operator does not know the explicit gradient of $C_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t)$ with respect to z_t, let alone its convexity. Hence the gradient-based online algorithms (e.g., online gradient decent) do not work in Problem 1.

To overcome the challenges, we first introduce how to tackle the partial feedback issue via the *mean field theory* in Section 4. We then propose an *online mean field aided policy* in Section 5.

4 MEAN FIELD MODEL

In Section 3, we characterize the hybrid edge network as an N-dimensional stochastic CTMC. This section introduces a *deterministic* mean field model that approximates the N-dimensional stochastic CTMC. Mathematically, we want to estimate the time-average workload $L_t^{[\mathrm{N}][\delta]}(z, \Phi_t)$ for any configuration $z \in \mathcal{Z}$ based on the mean field model. For notation simplicity, we will focus on a generic period and suppress the period index t in this section.

We will first introduce the density-based state and the mean field model in Section 4.1 and Section 4.2, respectively. We then study the relationship between the deterministic mean field model and the original stochastic system in Section 4.3.

4.1 Density-Based State

Recall that $Q^n(\tau)$ represents the number of tasks in passive-mode edge server $n \in \mathcal{N}$ at time $\tau \in [0, \delta]$. That is, $Q(\tau) = \{Q^n(\tau) \in \mathcal{B} : \forall n \in \mathcal{N}\}$ is a *quantity-based* state characterization. Now we introduce a *density-based* state and let $S_i^{[\mathrm{N}]}(\tau)$ denote the "fraction" of passive-mode edge servers with at least i tasks at time τ, i.e.,

$$S_i^{[\mathrm{N}]}(\tau) \triangleq \frac{1}{N} \sum_{n=1}^{N} \mathbb{I}\big(Q^n(\tau) \geq i\big), \ \forall i \in \mathcal{B}, \tag{8}$$

where $\mathbb{I}(\cdot)$ is an indicator function and the superscript [N] represents that there are N passive-mode edge servers. Note that we have $S_0^{[\mathrm{N}]}(\tau) = 1$ for any τ by definition. Moreover, $S^{[\mathrm{N}]}(\tau) = \{S_i^{[\mathrm{N}]}(\tau) : \forall i \in \mathcal{B}\}$ is the *density-based* state characterization for the system. Proposition 1 demonstrates the relationship between $Q(\tau)$ and $S^{[\mathrm{N}]}(\tau)$. The proof follows directly from the definition (8).

Proposition 1. *The quantity-based state $Q(\tau)$ and the density-based state $S^{[\mathrm{N}]}(\tau)$ satisfy the following equality*

$$\frac{1}{N} \sum_{n=1}^{N} Q^n(\tau) = \sum_{i=1}^{\mathrm{B}} S_i^{[\mathrm{N}]}(\tau), \quad \forall \tau \in [0, \delta], \tag{9}$$

which represents the average workload $L^{[\mathrm{N}]}(\tau, z, \Phi)$.

Proposition 1 essentially introduces two ways of calculating the average workload $L^{[\mathrm{N}]}(\tau, z, \Phi)$. As we will see later, it is more convenient to depict the system dynamics based on the density-based state $S^{[\mathrm{N}]}(\tau)$. Therefore, we will derive the mean field model based on the density-based state next.

4.2 Derivation of Mean Field Model

We first introduce the state transition and the expected drift. Afterwards, we formally define our mean field model.

4.2.1 State Transition. The density-based state is associated with two types of transitions, i.e., task admission and task execution. We will introduce the two transitions based on the vector $\boldsymbol{e}^a = \{e_i^a : \forall i \in \mathcal{B}\}$ defined as $e_i^a \triangleq \mathbb{I}(i = a)/N$ for any $i \in \mathcal{B}$.

- Task admission transition occurs whenever a passive-mode edge server admits a new task from the sources under JSQ(d). If the admission transition occurs to a passive-mode edge server holding $a - 1$ tasks at time τ, then $S_a^{[\text{N}]}(\tau)$ increases by $1/N$, while the other elements of $\boldsymbol{S}^{[\text{N}]}(\tau)$ do not change. That is, the density-based state becomes $\boldsymbol{S}^{[\text{N}]}(\tau) + \boldsymbol{e}^a$.
- Task execution transition occurs whenever a task in a passive-mode edge server is completed or migrated to one of the active-mode edge servers. If an execution transition occurs at a passive-mode edge server holding a tasks at time τ, then $S_a^{[\text{N}]}(\tau)$ decreases by $1/N$, while the other elements of $\boldsymbol{S}^{[\text{N}]}(\tau)$ do not change. That is, the density-based state becomes $\boldsymbol{S}^{[\text{N}]}(\tau) - \boldsymbol{e}^a$.

Next we introduce the expected drift of the density-based state $\boldsymbol{S}^{[\text{N}]}(\tau)$ based on the two types of state transitions above.

4.2.2 Expected Drift. The expected drift of the density-based state $\boldsymbol{S}^{[\text{N}]}(\tau)$ at time τ is defined as

$$F_i\left(\boldsymbol{S}^{[\text{N}]}(\tau)\right) \triangleq \lim_{\Delta \to 0} \frac{\mathbb{E}\left[S_i^{[\text{N}]}(\tau + \Delta) - S_i^{[\text{N}]}(\tau)\right]}{\Delta}, \ \forall i \in \mathcal{B}, \tag{10}$$

which measures the increasing rate of $S_i^{[\text{N}]}(\tau)$. We present the expression of $F_i(\cdot)$ in Proposition 2 and elaborate the rationale in the following proof sketch.

Proposition 2. *Given the resource configuration (x, y) and the average task arrival rate $\bar{\lambda}^{[\text{M}]}$, the expected drift at the density-based state $\boldsymbol{s} = \{s_i \in [0, 1] : \forall i \in \mathcal{B}\}$ is given by $F_0(\boldsymbol{s}) = 0$ and*

$$\begin{aligned} F_i(\boldsymbol{s}) = & \frac{\bar{\lambda}^{[\text{M}]}}{\theta}\left(s_{i-1}^d - s_i^d\right) - x\left(s_i - s_{i+1}\right) \\ & - \frac{\eta}{\theta} \cdot y\left[(1 - s_{i+1})^b - (1 - s_i)^b\right], \ \forall 1 \le i \le \text{B}, \end{aligned} \tag{11}$$

where the tuple (θ, η) represents the hybrid edge server provision and we let $s_{\text{B}+1} = 0$ for consistency.

Proof Sketch of Proposition 2. To derive (11), we consider a time interval $[\tau, \tau + \Delta]$ and compute the following term

$$\mathbb{E}\left[S_i^{[\text{N}]}(\tau + \Delta) - S_i^{[\text{N}]}(\tau)\right]. \tag{12}$$

Based on the previous discussion, the task admission transition $\boldsymbol{e}^i$ and the task execution transition $-\boldsymbol{e}^i$ lead to the increment $1/N$ and the decrement $-1/N$ for $S_i^{[\text{N}]}(\tau)$, respectively. To calculate (12), we consider the expected times that the transitions $\pm\boldsymbol{e}^i$ occur during the interval $[\tau, \tau + \Delta]$. Next we introduce an event (i.e., SA_i) leading to the task admission transition $\boldsymbol{e}^i$ and another two events (i.e., PE_i and AE_i) leading to the task execution transition $-\boldsymbol{e}^i$.

- *Source-task-admission* event SA_i means that a new task from the sources arrives at one of the passive-mode edge servers holding exactly $i - 1$ tasks under JSQ(d). First, source $m \in \mathcal{M}$ generates new tasks at rate λ^m, thus there are $\Delta \sum_{m=1}^{M} \lambda^m$ arrival tasks during the interval $[\tau, \tau + \Delta]$ on average. Second, the migration scheme JSQ(d) indicates that a new task is migrated to a passive-mode edge server holding exactly $i - 1$ tasks with the probability $[S_{i-1}^{[\text{N}]}(\tau)]^d - [S_i^{[\text{N}]}(t)]^d$. The event SA_i leads to the following increment for (12):
$$\frac{1}{N}\Delta M \bar{\lambda}^{[\text{M}]}\left(\left[S_{i-1}^{[\text{N}]}(t)\right]^d - \left[S_i^{[\text{N}]}(t)\right]^d\right). \tag{13}$$
- *Passive-execution* event PE_i means that one of the passive-mode edge servers holding i tasks completes a task. First, there are $[S_i^{[\text{N}]}(t) - S_{i+1}^{[\text{N}]}(t)]N$ passive-mode edge servers holding i tasks. Second, the exponentially distributed computing intensity (i.e., $l_{t[i]}^m \sim \text{Exp}(1)$) and the CPU frequency x in a passive-mode edge server indicate that event PE_i occurs at rate x. That is, there are Δx tasks completed during the interval $[\tau, \tau + \Delta]$ on average. Hence event PE_i leads to the following decrement for (12):
$$-\frac{1}{N}\Delta x\left[S_i^{[\text{N}]}(t) - S_{i+1}^{[\text{N}]}(t)\right]N. \tag{14}$$
- *Active-execution* event AE_i means that the active-mode edge server completes a task for a passive-mode edge server holding i tasks under LQF(b). First, the LQF(b) scheme implies that the heaviest load among the b samplings is i with probability $[1-S_{i+1}^{[\text{N}]}(t)]^b - [1-S_i^{[\text{N}]}(t)]^b$. Second, the exponentially distributed computing intensity (i.e., $l_{t[i]}^m \sim \text{Exp}(1)$) and the CPU frequency Ky of all the active-mode edge servers indicate that event AE_i occurs at rate Ky. Hence event AE_i leads to the following decrement for (12)
$$-\frac{1}{N}\Delta K y\left(\left[1 - S_{i+1}^{[\text{N}]}(t)\right]^b - \left[1 - S_i^{[\text{N}]}(t)\right]^b\right). \tag{15}$$

Finally, (12) equals to the summation of (13)~(15). Substituting it into (10) leads to the expression in (11). □

4.2.3 Mean Field Model. Next we will formally define our mean filed model using lowercase notations $\{s_i(\tau) : \forall i \in \mathcal{B}\}$ for clarity.

Definition 1. *The mean field model $\boldsymbol{s}(\tau) = \{s_i(\tau) \in [0, 1] : \forall i \in \mathcal{B}\}$, is defined by the following three types of conditions:*

- *Initial condition $\boldsymbol{s}(0) = \boldsymbol{s}^{\mathbf{0}}$.*
- *Boundary condition $s_0(\tau) = 1$, $\forall \tau \ge 0$.*
- *Drift condition $\frac{\text{d}\boldsymbol{s}(\tau)}{\text{d}\tau} = \boldsymbol{F}(\boldsymbol{s}(\tau))$ where the set of functions $\boldsymbol{F}(\boldsymbol{s}) = \{F_i(\boldsymbol{s}) : \forall i \in \mathcal{B}\}$ are given in (11).*

Note that the above mean field model is a set of ordinary differential equations, which are deterministic. Specifically, the initial condition specifies where the mean field starts to evolve. The boundary condition coincides with the definition of the density-based state in (8). The drift condition is the same as the expected drift of the original stochastic system $\boldsymbol{S}^{[\text{N}]}(\tau)$. Next we introduce the connection between the deterministic mean field model $\boldsymbol{s}(\tau)$ and the original stochastic system $\boldsymbol{S}^{[\text{N}]}(\tau)$.

4.3 Fixed Point and Convergence

We first present the fixed point of the mean field model. We then introduce the convergence relation between the deterministic mean field model and the original stochastic system.

4.3.1 Fixed Point. The fixed point of the mean field model is a state $\tilde{s}$, at which the mean field model does not change. That is, we have $s(\tau') = \tilde{s}$ for any $\tau' \geq \tau$ if $s(\tau) = \tilde{s}$. Theorem 1 presents the fixed point. The proof is given in our technical report [26].

Theorem 1. *The fixed point $\tilde{s} = \{\tilde{s}_i : \forall i \in \mathcal{B}\}$ of the mean field model in Definition 1 is given by*

$$\tilde{s}_i = \begin{cases} 1, & \text{if } i = 0, \\ g(\tilde{s}_{i+1}; \gamma), & \text{if } 1 \leq i < \mathrm{B}, \\ g(0; \gamma), & \text{if } i = \mathrm{B}, \end{cases} \tag{16}$$

where the function $g(s;\gamma)$ is given by

$$g(s;\gamma) \triangleq \left[\frac{\theta x s - \eta y (1-s)^b + \gamma}{\bar{\lambda}^{[\mathrm{M}]}}\right]^{\frac{1}{a}}. \tag{17}$$

Moreover, the constant γ solves $g^{(1+\mathrm{B})}(0;\gamma) = 1$, where the function $g^{(1+\mathrm{B})}(\cdot,\gamma)$ is the $(1+\mathrm{B})$-th iterate of $g(\cdot;\gamma)$.

Based on Theorem 1, one can efficiently compute the fixed point $\tilde{s}$ given the configuration (x, y) and the average arrival rate $\bar{\lambda}^{[\mathrm{M}]}$. Accordingly, we often use $\tilde{s}(x, y, \bar{\lambda}^{[\mathrm{M}]})$ to emphasize the dependency. Recall that this section aims to estimate the time-average workload $L^{[\mathrm{N}][\delta]}(x, y, \Phi)$ based on the mean field model. For this goal, we define $l(x, y, \bar{\lambda}^{[\mathrm{M}]})$ based on the fixed point as follows

$$l\left(x, y, \bar{\lambda}^{[\mathrm{M}]}\right) \triangleq \sum_{i=1}^{\mathrm{B}} \tilde{s}_i\left(x, y, \bar{\lambda}^{[\mathrm{M}]}\right), \tag{18}$$

which is an accurate estimation of $L^{[\mathrm{N}][\delta]}(x, y, \Phi)$. The following convergence result ensures the accuracy.

4.3.2 Convergence. In Theorem 2, we introduce the relationship between the original stochastic system and the deterministic mean field model. The proof is given in our technical report [26].

Theorem 2. *Given the edge server provision (θ, η) for the M sources, the limiting system (i.e., $M \to \infty$) satisfies*

$$\lim_{\substack{M\to\infty \\ N=M\theta}} \lim_{\delta\to\infty} \left\| L^{[\mathrm{N}][\delta]}(x, y, \Phi) - l\left(x, y, \bar{\lambda}^{[\mathrm{M}]}\right) \right\| = 0, \tag{19}$$

where δ is the duration of a single period.

Theorem 2 shows that the mean field model is an accurate approximation of the time-average workload if the system size M and the period duration δ are large. The intuitions are two-fold:

- The stochastic system is characterized by a CTMC $\{S^{[\mathrm{N}]}(\tau) : \tau \in [0, \delta]\}$ in each period. As the period duration δ increases, the CTMC is approaching to its steady state.
- The drift condition of the mean field model is defined by the expected drift of the stochastic system. As N increases, the CTMC $\{S^{[\mathrm{N}]}(\tau) : \tau \in [0, \delta]\}$ behaves closer to its expectation (i.e., the mean field model) by the Law of large number [20].

So far, we have introduced the connection between the deterministic mean field model and the stochastic edge network. Although the above analysis assumes that the computing intensity follows the exponential distribution, one can obtain similar results under the general distributions with decreasing hazard rate based on *asymptotic independence* or *propagation of chaos* [27]. We refer interested readers to Section 9.1 of [14] for more details.

5 AN ONLINE MEAN FIELD POLICY

This section proposes an *online mean field aided configuration policy* $\mathfrak{A}$, which leverages the mean field model to address the partial feedback issue in Problem 1. To proceed, we first define the *mean field cost* in period t as follows

$$C_t\left(z_t, \bar{\lambda}_t^{[\mathrm{M}]}\right) \triangleq G\left(l\left(z_t, \bar{\lambda}_t^{[\mathrm{M}]}\right)\right) + \xi_t^{\mathrm{P}} \theta x_t + \xi_t^{\mathrm{A}} \eta y_t, \tag{20}$$

which replaces the time-average workload $L_t^{[\mathrm{N}][\delta]}(z_t, \Phi_t)$ with the deterministic formula $l(z_t, \bar{\lambda}_t^{[\mathrm{M}]})$. Moreover, the mean field cost has the following features.

- Theorem 2 implies that the mean field cost (20) is an accurate estimation for the operator's real cost (6).
- At the end of period t, the operator can efficiently compute the mean field cost for any configuration $z \in \mathcal{Z}$ after observing the average arrival rate $\bar{\lambda}_t^{[\mathrm{M}]}$ and the per-unit expenditure $(\xi_t^{\mathrm{P}}, \xi_t^{\mathrm{A}})$.

Although the mean field cost exhibits an explicit expression, one can check that it is not convex in z_t. Therefore, the classic online convex optimization algorithms (e.g., OGD) cannot preserve a good performance in our problem. Next we will introduce our approach in Section 5.1 and analyze its performance in Section 5.2.

5.1 Online Mean Field Aided Policy

5.1.1 Basic Idea. The proposed policy $\mathfrak{A}$ works in two phases: (a) *discretization phase* and (b) *learning phase*. In the discretization phase, we discretize the metric space $\mathcal{Z}$ into a finite set $\mathcal{A}$ of resource candidates. In the learning phase, we learn about the optimal resource candidate based on the past observations and the mean field model. Note that the mean field model enables the learning phase to obtain a full feedback. Moreover, we will show that the discretization phase only incurs a bounded performance loss. We summarize the proposed policy $\mathfrak{A}$ in Algorithm 1 and elaborate it in the following.

5.1.2 Discretization Phase. We discretize the metric space $\mathcal{Z}$ based on a parameter $\beta > 0$, i.e., Line 1 and Line 2. Specifically, for the passive-mode edge servers, we define $x_{[i]}$ as follows

$$x_{[i]} \triangleq \min\left(x_H, x_L(1+\beta)^i\right), \ \forall i \in \{1, 2, ..., A_x\}, \tag{21}$$

where $A_x \triangleq \left\lceil \log_{1+\beta} \frac{x_H}{x_L} \right\rceil$ depends on the parameter β and $\lceil\cdot\rceil$ is the ceil function. Note that the set $\{x_{[i]} : \forall 1 \leq i \leq A_x\}$ includes all powers of $1+\beta$ in the range $[x_L, x_H]$. Similarly, for the active-mode edge servers, we define $y_{[j]}$ as follows:

$$y_{[j]} \triangleq \min\left(y_H, y_L(1+\beta)^j\right), \ \forall j \in \{1, 2, ..., A_y\}, \tag{22}$$

where $A_y \triangleq \left\lceil \log_{1+\beta} \frac{y_H}{y_L} \right\rceil$.

Algorithm 1: Mean Field aided Configuration Policy $\mathfrak{A}$

```
  Output: Decisions $(x_{[i_t]}, y_{[j_t]})$ for each period $t$.
1 Initial $\epsilon \in (0,1)$ and $\beta > 0$.
2 Define the resource candidate $(x_{[i]}, y_{[j]})$ for any tuple
  $(i,j) \in \mathcal{A}$ based on (21) and (22).
3 Initial $w_1(i,j) = 1$ for any tuple $(i,j) \in \mathcal{A}$
4 for $t = 1$ to $T$ do
5     Calculate $\boldsymbol{p}_t = \{p_t(i,j) : \forall (i,j) \in \mathcal{A}\}$ based on (23)
6     Determine $(x_{[i_t]}, y_{[j_t]})$ by drawing a tuple $(i_t, j_t) \in \mathcal{A}$
      randomly according to the probability distribution $\boldsymbol{p}_t$
7     Calculate the normalized mean-field cost $c_t(i,j)$ for any
      $(i,j) \in \mathcal{A}$ based on (24)
8     Update weight matrix $\boldsymbol{w}_{t+1}$ according to (25)
9 end
```

Based on the discussions above, the set $\mathcal{A} \triangleq \{(i,j) : \forall 1 \le i \le A_x, 1 \le j \le A_y\}$ contains all the resource candidates after discretizing $\mathcal{Z}$ based on the parameter β. Note that a smaller parameter β leads to a more precise discretization, thus a larger set $\mathcal{A}$. We will show how the parameter β affects the theoretical performance of the proposed policy $\mathfrak{A}$ in Section 5.2.

Next we introduce the learning phase based on the set $\mathcal{A}$.

5.1.3 Learning Phase. The learning phase of policy $\mathfrak{A}$ follows the multiplicative weight update (MWU) method [21]. Specifically, we maintain a weight matrix $\boldsymbol{w}_t = \{w_t(i,j) \in [0,1] : \forall (i,j) \in \mathcal{A}\}$ in each period t. As we will see later, the weight $w_t(i,j)$ in period t is negatively related to the total mean field cost incurred by the candidate $(x_{[i]}, y_{[j]})$ before period t. Hence a larger weight corresponds to a better resource configuration. Moreover, policy $\mathfrak{A}$ uses the weight matrix $\boldsymbol{w}_t$ to calculate the probabilistic selection matrix $\boldsymbol{p}_t = \{p_t(i,j) : \forall (i,j) \in \mathcal{A}\}$, and determines the resource configuration probabilistically based on $\boldsymbol{p}_t$. Overall, the resource candidate with a larger weight is selected with a higher probability.

As shown in Algorithm 1, the learning phase includes Lines 3∼8. Specifically, policy $\mathfrak{A}$ will initialize the weight matrix equally and repeat the following two steps:

- Line 5 to Line 6: We calculate the probabilistic selection matrix $\boldsymbol{p}_t$ based on the weight matrix $\boldsymbol{w}_t$ as follows:
$$p_t(i,j) \triangleq \frac{w_t(i,i)}{\sum_{(i',j')\in\mathcal{A}} w_t(i',j')}, \ \forall (i,j) \in \mathcal{A}. \tag{23}$$
We then determine the resource configuration $(x_{[i_t]}, y_{[j_t]})$ in period t by randomly drawing a tuple $(i_t, j_t) \in \mathcal{A}$ according to the probability distribution $\boldsymbol{p}_t$.
- Line 7 to Line 8: At the end of period t, the operator observes the average arrival rate $\bar{\lambda}_t^{[\mathrm{M}]}$, and calculates the normalized mean field cost $c_t(i,j)$ based on the mean field model as follows:
$$c_t(i,j) \triangleq \frac{C_t(x_{[i]}, y_{[j]}, \bar{\lambda}_t^{[\mathrm{M}]})}{\bar{C}}, \quad \forall (i,j) \in \mathcal{A}. \tag{24}$$
where $\bar{C} \triangleq G(\mathrm{B}) + x_H \xi_{max}^{\mathsf{P}} + y_H \xi_{max}^{\mathsf{A}}$ represents the maximal mean field cost. Finally, we update the weight $\boldsymbol{w}_{t+1}$ for the next period according to
$$w_{t+1}(i,j) = w_t(i,j) \cdot (1-\epsilon)^{c_t(i,j)}, \ \forall (i,j) \in \mathcal{A}. \tag{25}$$
where $\epsilon \in (0,1)$ is a parameter initialized in Line 1.

So far we have introduced the proposed policy $\mathfrak{A}$ in Algorithm 1. Next we move on to the performance analysis.

5.2 Performance Analysis

Recall that Theorem 2 shows the convergence relationship between the original stochastic edge network and the mean field model. In this section, we will focus on the mean field cost in performance analysis. Specifically, the total mean field cost incurred by policy $\mathfrak{A}$ is
$$C_T^{\mathfrak{A}} = \sum_{t\in\mathcal{T}} \mathbb{E}\left[C_t\left(x_{[i_t]}, y_{[j_t]}, \bar{\lambda}_t^{[\mathrm{M}]}\right)\Big|\, \boldsymbol{p}_t\right], \tag{26}$$
where the expectation is taken over the randomness in Line 6 of Algorithm 1. We will analyze the performance gap between the proposed policy $\mathfrak{A}$ and the offline optimal solution (x^*, y^*), i.e.,
$$(x^*, y^*) \triangleq \underset{(x,y)\in\mathcal{Z}}{\arg\min} \sum_{t\in\mathcal{T}} C_t\left(x, y, \bar{\lambda}_t^{[\mathrm{M}]}\right). \tag{27}$$
Accordingly, we let $C_T^* \triangleq \sum_{t\in\mathcal{T}} C_t(x^*, y^*, \bar{\lambda}_t^{[\mathrm{M}]})$ denote the offline minimal mean field cost.

The performance gap between C_T^* and $C_T^{\mathfrak{A}}$ depends on the two parameters (ϵ, β) in Algorithm 1. Roughly speaking, the performance loss of policy $\mathfrak{A}$ consists of the *discretization loss* and the *learning loss*, which are presented in Lemma 1 and Lemma 2, respectively. We provide the proof in our technical report [26].

Lemma 1 (Discretization Loss). *There exists a tuple $(i^*, j^*) \in \mathcal{A}$ satisfying the following conditions*
$$\sum_{t\in\mathcal{T}} C_t\left(x_{[i^*]}, y_{[j^*]}, \bar{\lambda}_t^{[\mathrm{M}]}\right) \le (1+\beta)^2 C_T^*. \tag{28}$$

Lemma 1 shows that the discretization scheme in policy $\mathfrak{A}$ increases at most a constant factor of $(1+\beta)^2$ compared to the offline minimal cost C_T^*. To reduce the discretization loss, we need to choose a smaller β. However, as shown by Lemma 2, a smaller β leads to a larger learning loss, since a smaller β enlarges the set $\mathcal{A}$.

Lemma 2 (Learning Loss). *For any tuple $(i,j) \in \mathcal{A}$, we have*
$$C_T^{\mathfrak{A}} \le \Psi(\epsilon, \beta) + \frac{1}{\epsilon}\ln\left(\frac{1}{1-\epsilon}\right) \sum_{t\in\mathcal{T}} C_t(x_{[i]}, y_{[j]}, \bar{\lambda}_t^{[\mathrm{M}]}), \tag{29}$$
where $\sum_{t\in\mathcal{T}} C_t(x_{[i]}, y_{[j]}, \bar{\lambda}_t^{[\mathrm{M}]})$ is the total mean field cost incurred by $(x_{[i]}, y_{[j]})$. Moreover, the constant $\Psi(\epsilon, \beta)$ is given by
$$\Psi(\epsilon, \beta) \triangleq \frac{\bar{C}}{\epsilon} \ln\left(\frac{(1+\beta)\ln\frac{x_H}{x_L}}{\ln(1+\beta)} \cdot \frac{(1+\beta)\ln\frac{y_H}{y_L}}{\ln(1+\beta)}\right). \tag{30}$$

Lemma 2 indicates that the learning loss of policy $\mathfrak{A}$ jointly depends on the two parameters ϵ and β. Note that the constant $\Psi(\epsilon, \beta)$ decreases in ϵ, while the coefficient $\frac{1}{\epsilon}\ln\left(\frac{1}{1-\epsilon}\right)$ increases in ϵ. Hence there is a trade-off in choosing the parameter ϵ. Moreover, the constant $\Psi(\epsilon, \beta)$ decreases in the parameter β, thus a smaller β reduces the discretization loss, but increases the learning loss.

Theorem 3 presents the relationship between $C_T^{\mathfrak{A}}$ and C_T^* by combining Lemma 1 and Lemma 2.

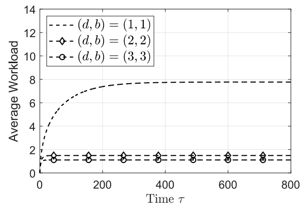
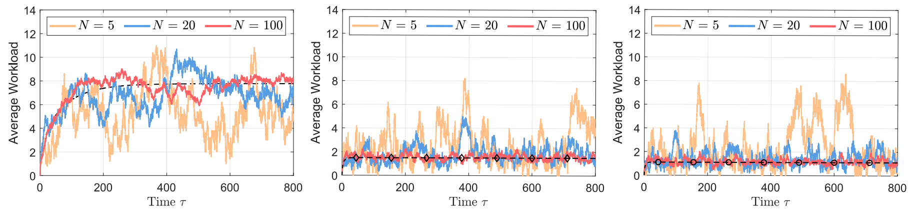


Figure 2: Mean field evolution



(a) $(d,b)=(1,1)$ (b) $(d,b)=(2,2)$ (c) $(d,b)=(3,3)$

Figure 3: Average workload in the stochastic edge network

Theorem 3. *For any $\epsilon \in (0,1)$ and $\beta > 0$, policy $\mathfrak{A}$ can attain the following performance*

$$C_T^{\mathfrak{A}} \leq \Psi(\epsilon,\beta) + \frac{1}{\epsilon}\ln\left(\frac{1}{1-\epsilon}\right)(1+\beta)^2 C_T^*. \tag{31}$$

Theorem 3 shows that the expected cost incurred by the policy $\mathfrak{A}$ is no greater than the sum of a constant multiple of the offline minimal cost C_T^* and an extra constant.

6 NUMERICAL RESULTS

We carry out extensive evaluation on the proposed policy $\mathfrak{A}$ based on empirical data. We will consider a hybrid edge server provision mode $(\theta,\eta) = (0.5, 0.1)$ and investigate the impact of system size M. We start with single-period demonstration in Section 6.1, and then carry out multi-period evaluation in Section 6.2.

6.1 Single-Period Demonstration

We consider a single period and compare the mean field model to the stochastic edge network. Specifically, we will fix the resource configuration $(x,y) = (1,5)$ and the task arrival rate $\lambda^m = 0.9$ for each source $m \in \mathcal{M}$.

Fig. 2 shows how the mean field model $\boldsymbol{s}(\tau)$ evolves from the initial state $\boldsymbol{s}^{\mathbf{0}} = \mathbf{0}$. The vertical axis represents the average workload $\sum_{i=1}^{\mathrm{B}} s_i(\tau)$ and the three curves correspond to different migration parameters (d,b), respectively. As time τ increases, the mean field model converges to the fixed point $\tilde{s}$ defined in Theorem 1.

Fig. 3 shows the average workload $L^{[\mathrm{N}]}(\tau,x,y,\boldsymbol{\Phi})$ of the stochastic edge network. The three sub-figures correspond to different migration parameters, respectively. In each sub-figure, the three solid curves represent different numbers of sources, i.e., $M \in \{10, 40, 200\}$. The black dash curve is the same as that in Fig. 2. Note that the three solid curves are all centered on the black dash curve with some fluctuation. Moreover, a larger system size corresponds to a smaller fluctuation. These observations are consistent with the convergence results in Theorem 2.

Comparing the three sub-figures in Fig. 3, we find it significantly reduces the average workload by changing the parameters (d,b) from $(1,1)$ to $(2,2)$. Also, it merely leads to a tiny reduction by further increasing to $(3,3)$. This means that the migration schemes JSQ(2) and LQF(2) slightly increase the communication overhead, but can reduce the average workload considerably. We will verify this claim in the dynamic setting with multiple periods.

6.2 Multi-Period Evaluation

We evaluate the proposed policy $\mathfrak{A}$ based on the real world electricity market prices in US [28]. Fig. 4 plots the hourly prices of the first three months in 2020. That is, we consider a total of $T = 2208$ periods (i.e., hours). We let p_t denote the price in period t, and quantify the operation expenditure according to $\xi_t^{\mathrm{P}} = 0.1p_t$ and $\xi_t^{\mathrm{A}} = 0.08p_t$. Furthermore, we first generate the sources' task arrival rate in different periods according to a uniform distribution on the support $[0,1]$. We then generate the task profile $\boldsymbol{\Phi}_t$ accordingly. The feasible resource ranges of the edge servers are $[0.2, 2]$ and we use $G(l) = 30l$ to measure the performance degradation. We evaluate the proposed policy $\mathfrak{A}$ with the parameters $(\epsilon,\beta) = (0.2, 0.5)$ and compare the following three cases:

- Case `Mfm` represents the mean field model and measures the mean field cost incurred by Algorithm 1.
- Case `Alg`(N) corresponds to the edge network with $N = \theta M$ passive-mode edge servers, and measures operator's real cost incurred by Algorithm 1.
- Case `Bch`(N) is the benchmark of case `Alg`(N) without the mean field model. That is, `Bch`(N) adopts the same discretization phase as in Algorithm 1, but only relies on the observed partial feedback in the learning phase.

We run the evaluation for one hundred times and visualize the results in Fig. 5. The two sub-figures correspond to different migration parameters, i.e., $(d,b) = (1,1)$ and $(d,b) = (2,2)$. In each sub-figure, the black dash line represents the time-average offline minimal cost, i.e., C_T^*/T. The black circle curve corresponds to the case `Mfm`. The blue and red curves with markers represent case `Alg`(5) and case `Alg`(10), respectively. The blue and red curves without marker represent `Bch`(5) and `Bch`(10), respectively.

In each sub-figure of Fig. 5, we have the following observations:

- The black circle (i.e., `Mfm`) curve converges to the black dash line, which verifies the performance of policy $\mathfrak{A}$.
- The blue triangle curve (i.e., `Alg`(5)) and the red square curve (i.e., `Alg`(10)) have a slight difference compared to black circle curve (i.e., `Mfm`), which is due to the mean field approximation. The red square curve is closer to the black curve than the blue triangle curve, indicating that the approximation error decreases in the system size.
- Comparing the two blue curves shows that case `Bch`(5) takes much longer time to converge to C_T^*/T than case `Alg`(5). The two red curves also lead to a similar observation. These observations imply that the mean field model can speed up the convergence of online resource configuration.

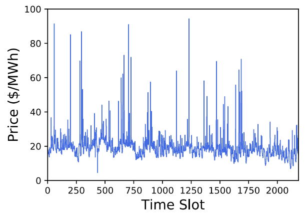
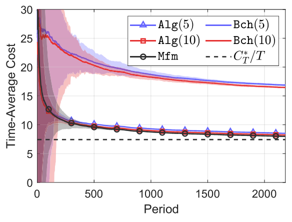
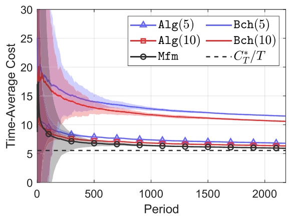


Figure 4: Electricity prices



(a) $(d, b) = (1, 1)$



(b) $(d, b) = (2, 2)$

Figure 5: Time-average cost of the operator

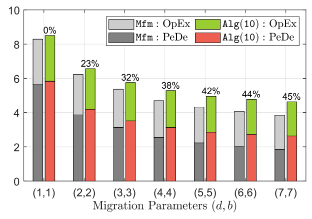


Figure 6: Impact of (d, b)

Comparing the two sub-figures in Fig. 5, we note that the larger migration parameters (d, b) lead to the lower operating cost. This motivates us to investigate the impact of (d, b) in the following.

Fig. 6 investigates the influence of the migration parameters in cases `Mfm` and `Alg(10)`. Specifically, Fig. 6 plots the time-average performance degradation (PeDe) and the operation expenditure (OpEx) over the T periods. At the top of each bar, we label the percentage of the cost reduction compared to $(d, b) = (1, 1)$. We have the following two observations.

- Increasing the migration parameters (d, b) can reduce the operator's total operating cost up to 45%. However, the communication overhead also increases linearly in d and b.
- The cost reduction when $(d, b) = (2, 2)$ is already greater than half of that when $(d, b) = (7, 7)$.

The above observations show that the migration schemes JSQ(2) and LQF(2) under the proposed policy $\mathfrak{A}$ can considerably reduce the operating cost with a small increase in communication overhead.

7 CONCLUSION

This paper considers a hybrid edge server provision under two decentralized task migration schemes, and studies how to configure the computing resource in an online dynamic fashion. Specifically, the dynamic resource configuration of a large-scale stochastic edge network corresponds to an online cost minimization problem with partial feedback. To resolve the partial feedback issue, we derive a deterministic mean field model, and use it to estimate the performance of the stochastic system. Moreover, we propose an online mean field aided configuration policy, and show that the proposed policy can asymptotically perform as good as the optimal offline configuration. As far as we know, we are the first to integrate mean field theory and online learning with partial feedback. We believe that our results in this paper can improve the efficiency of edge server provision and facilitate the large-scale implementation.

REFERENCES

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8):1738–1762, 2019.
[2] X. Chen, L. Jiao, W. Li, and X. Fu. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Transactions on Networking*, 24(5):2795–2808, 2015.
[3] Y. Mao, J. Zhang, and K.B. Letaief. Dynamic computation offloading for mobile-edge computing with energy harvesting devices. *IEEE Journal on Selected Areas in Communications*, 34(12):3590–3605, 2016.
[4] J. Xu, L. Chen, and P. Zhou. Joint service caching and task offloading for mobile edge computing in dense networks. *IEEE INFOCOM*, 2018.
[5] I.H. Hou, T. Zhao, S. Wang, and K. Chan. Asymptotically optimal algorithm for online reconfiguration of edge-clouds. *ACM MobiHoc*, 2016.
[6] Y. Sahni, J. Cao, and L. Yang. Data-aware task allocation for achieving low latency in collaborative edge computing. *IEEE Internet of Things Journal*, 6(2):3512–3524, 2019.
[7] A. Galanopoulos, T. Salonidis, and G. Iosifidis. Cooperative edge computing of data analytics for the internet of things. *IEEE Transactions on Cognitive Communications and Networking*, 2020.
[8] M. Tang, L. Gao, and J. Huang. Enabling edge cooperation in tactile internet via 3C resource sharing. *IEEE Journal on Selected Areas in Communications*, 36(11):2444–2454, 2018.
[9] K. Poularakis, J. Llorca, A.M. Tulino, I. Taylor, and L. Tassiulas. Joint service placement and request routing in multi-cell mobile edge computing networks. *IEEE INFOCOM*, 2019.
[10] H. Zhang, Y. Xiao, S. Bu, D. Niyato, R. Yu, and Z. Han. Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining stackelberg game and matching. *IEEE Internet of Things Journal*, 4(5):1204–1215, 2017.
[11] M. Chen, B. Liang, and M. Dong. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point. *IEEE INFOCOM*, 2017.
[12] E. Meskar and B. Liang. Fair multi-resource allocation in mobile edge computing with multiple access points. *ACM MobiHoc*, 2020.
[13] Z. Zhou, X. Chen, W. Wu, D. Wu, and J. Zhang. Predictive online server provisioning for cost-efficient iot data streaming across collaborative edges. *ACM MobiHoc*, 2019.
[14] M.V. Der Boor, S. Borst, Johan SH Van Leeuwaarden, and D. Mukherjee. Scalable load balancing in networked systems: A survey of recent advances. *arXiv: Probability*, 2018.
[15] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1094–1104, 2001.
[16] L. Ying, R. Srikant, and X. Kang. The power of slightly more than one sample in randomized load balancing. *Mathematics of Operations Research*, 42(3):692–722, 2017.
[17] M. Bramson, Y. Lu, and B. Prabhakar. Randomized load balancing with general service time distributions. *ACM SIGMETRICS performance evaluation review*, 38(1):275–286, 2010.
[18] Q. Xie, X. Dong, Y. Lu, and R. Srikant. Power of d choices for large-scale bin packing: A loss model. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):321–334, 2015.
[19] J. N Tsitsiklis and K. Xu. On the power of (even a little) resource pooling. *Stochastic Systems*, 2(1):1–66, 2013.
[20] L. Ying. On the approximation error of mean-field models. *Stochastic Systems*, 8(2):126–142, 2018.
[21] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.
[22] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, 1998.
[23] K. Lee, M. Lam, R. Pedarsani, D. S Papailiopoulos, and K. Ramchandran. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory*, 64(3):1514–1529, 2018.
[24] A. Reisizadeh, S. Prakash, R. Pedarsani, and A.S. Avestimehr. Coded computation over heterogeneous clusters. *IEEE Transactions on Information Theory*, 65(7):4227–4242, 2019.
[25] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. MIT press, 1965.
[26] Technical report. https://www.dropbox.com/s/0hfyfhlk6bk8h6c/TR.pdf.
[27] M. Bramson, Y. Lu, and B. Prabhakar. Asymptotic independence of queues under randomized load balancing. *Queueing Systems*, 71(3):247–292, 2012.
[28] PJM. https://dataminer2.pjm.com/feed/rt_hrl_lmps/definition.