Integrate Learning and Control in Queueing Systems with Uncertain Payoffs

Wei-Kang Hsu, Jiaming Xu, Xiaojun Lin, Mark R. Bell

Purdue University

February 16, 2018

Abstract

We study task assignment in online service platforms where un-labeled clients arrive according to a stochastic process and each client brings a random number of tasks. As tasks are assigned to servers, they produce client/server-dependent random payoffs. The goal of the system operator is to maximize the expected payoff per unit time subject to the servers' capacity constraints. However, both the statistics of the dynamic client population and the client-specific payoff vectors are unknown to the operator. Thus, the operator must design task-assignment policies that integrate adaptive control (of the queueing system) with online learning (of the clients' payoff vectors). A key challenge in such integration is how to account for the non-trivial closed-loop interactions between the queueing process and the learning process, which may significantly degrade system performance. We propose a new utility-guided online learning and task assignment algorithm that seamlessly integrates learning with control to address such difficulty. Our analysis shows that, compared to an oracle that knows all client dynamics and payoff vectors beforehand, the gap of the expected payoff per unit time of our proposed algorithm in a finite T horizon is bounded by $\beta_1/V+\beta_2\sqrt{\log N/N}+\beta_3 N(V+1)/T$, where V is a tuning parameter of the algorithm, and β_1,β_2,β_3 only depend on arrival/service rates and the number of client classes/servers. Through simulations, we show that our proposed algorithm significantly outperforms a myopic matching policy and a standard queue-length based policy that does not explicitly address the closed-loop interactions between queueing and learning.

1 Introduction

Thanks to advances in communication and computing, new generations of online service platforms have transformed the business models in many domains, from online labor market of freelance work (e.g., Upwork [Upw]), online hotel rental (e.g., Airbnb [Air]), online education (e.g., Coursera [Cou]), crowd-sourcing (e.g., Amazon Mechanical Turk [Ipe10]), to online advertising (e.g., AdWords [AdW]) and many more. By bringing unprecedented number of clients and service providers together, these online service platforms greatly increase access to service, lower the barrier-to-entry for competition, improve resource utilization, reduce cost and delay, and thus enhance the overall well-being of society and our daily life.

Motivated by these online service platforms, in this paper we are interested in learning and control problems in queueing systems with uncertain agent dynamics and uncertain payoffs. Note that a key control decision in operating such online service platforms is how to assign clients to servers to attain the maximum system benefit. Such decisions are challenging because there often exist significant uncertainty in both agent payoffs and agent dynamics. First, there is significant uncertainty in the quality, i.e., payoff, of a particular assignment between a client (needing service) and a server (providing service). For example, in online ad

platforms, when an ad from a particular ad campaign is displayed in response to a search request for a given keyword, the click-throughput rate is unknown in advance [CJS15]. Similarly, in crowd-sourcing, the efficiency of completing a particular task by a given worker is unknown a priori [KOS11a, KOS13, KOS11b, KO16]. Second, the population of agents (i.e., clients or servers) is often highly dynamic. For example, ad campaigns arrive and depart constantly in online ad platforms [TS12]; so do requestors in crowd-sourcing. The statistics of such arrivals and departures are often unknown beforehand, resulting in uncertain queueing dynamics. Therefore, the operator of these platforms must not only continuously learn the payoffs associated with each new agent, but also adaptively control the assignment and resource allocation in response to the uncertain arrival/departure dynamics. Thus, it is imperative to study both online learning (of uncertain payoffs) and adaptive control (of uncertain queueing dynamics) in a unified framework to achieve optimal performance in such a dynamic and uncertain environment.

Prior Work That Studies Learning and Queueing Seperately: Prior studies on online service platforms often treat the optimal learning problem and the adaptive control problem separately, and as a result may produce significantly sub-optimal results when there is uncertainty in both agent dynamics and agent payoffs. From the learning side, multi-armed bandits (MAB) [LR85, ACBF02, GGW11, BCB+12] have been extensively used to model online decision problems with uncertain payoffs. In a classical MAB formulation, a client has a sequence of jobs that need to be assigned, one at a time, to servers with uncertain payoffs (corresponding to "arms" in the MAB literature). The key question is how to balance exploitation (using the currently-known best server) and exploration (finding out which server is the best). However, *these studies usually focus on a static setting, without considering the uncertainty in queueing dynamics.* For example, most stochastic bandit [LR85, ACBF02], Markovian bandit [Git79, Whi88] and adversarial bandit problems [ACBFS02] assume only one client. Although bandits with dynamic arms (e.g., the open-bandit processes) [LY88, WZ13, Whi81] allow the number of servers/arms to follow a Markov process, the statistics of the Markov chain must be known in advance. Similarly, although contextual bandits (as in the studies of online ad [CJS15, JS13, PO07, BLS14, LZ08] and crowd-sourcing [JNN14, TTSRJ14]) may be interpreted as allowing arriving jobs of multiple types, the label (i.e., context) of each job must be known in advance. Thus, they cannot be used when there is a dynamic population of clients with *unknown labels*. More recently, the work in [KNJ14, NKJ16, GKJ12, AMTS11, LEGJP11, LPT13, CP15, LCS15, SRJ17, BTZ15] considers a more-general multi-player setting and combines learning (of agent features) with matching, combinatorial optimization, and/or consensus decisions. However, such results still assume a static setting with a fixed number of agents, and thus do not account for the queueing dynamics due to agent arrivals and departures in practical online service platforms. In this sense, they can at best be viewed as a myopic solution for a snapshot of the system in time. As we will elaborate in Section 3, using such a myopic solution from a static setting could lead to substantially lower levels of system performance when the agent dynamics are taken into account.

At the other end of the spectrum, the operation of online service platforms has been studied as a stochastic or adversarial matching/queueing problem. Adaptive control policies under uncertain dynamics have been provided, e.g., for online ad [TS12, FKM+09, CZL08, AHL13, ZCXL13] and/or queueing networks in general [TE92, NML05, Chi04, ES05, LS05, LSS06, LSS08], to maintain system stability, maximize payoff, and/or minimize cost and delay. However, *these studies usually assume that the payoff vectors of the clients are known.* In practice, the payoff vectors of new clients are often unknown. These adaptive control policies will break down without this crucial piece of information. Recently, there have been studies that combine the notion of learning with queueing [HLH14, HZCL16]. However, the focus of these studies is in learning the arrival rates. Thus, they have not considered uncertain payoffs.

Our Contributions: The main contribution of this paper is therefore to develop new schemes that integrate online learning with adaptive control to address uncertainty in both payoff parameters and client dynamics. We propose a new utility-guided online learning and task assignment algorithm that can achieve near-optimal system payoff even compared to an "oracle" that knows the statistics of client dynamics and all

clients' payoff vectors beforehand. Specifically, we show that the gap between the expected payoff per unit time achieved by our proposed algorithm and that achieved by the "oracle" in a finite T horizon is at most the sum of three terms: the first one is of the order $1/V$ where V is a parameter of the proposed algorithm that can be taken to a large value; the second one is of the order $\sqrt{\log N/N}$ where N is the average number of tasks per client; the third one is of the order $N(V+1)/T$ which decreases as the time T increases. These three terms have the following natural physical interpretations. The first $1/V$ term is due to the uncertainty in client arrival process; the second $\sqrt{\log N/N}$ term is because of learning from noisy payoff feedback; the third $N(V+1)/T$ term characterizes the payoff loss incurred by the backlogged clients.

Our proof builds upon the standard Lyapunov drift technique [NML05, LSS08, DLK01, BM01] and finite-time regret analysis [ACBF02]; however, we make two major innovations, which could be of independent interest. First, to obtain $\sqrt{\log N/N}$ loss in payoff learning, we combine a martingale argument and a comparison argument to capture both the impact of the payoff estimation errors on the queueing dynamics, and the impact of congestion on the rate of payoff learning. Second, to derive the NV/T loss in finite time horizon, we upper bound the mean number of backlogged clients in the system by establishing a coupling between the current system and a Geom/Geom/m queue.

There are a few recent studies in the literature that also attempt to integrate learning and queueing [JKK16, MX16, JKK17, BM15, SGMV17]. Compared to these related studies, our work makes a number of new contributions. First, note that a key challenge in integrating learning with queueing is the non-trivial closed-loop interactions between the queueing process and the learning process. Specifically, excessive *queueing*, which can be caused by both agent arrivals/departures and task assignments/completion, leads to both delay in observing the payoff feedback and a slow-down of the *learning* process of the uncertain payoffs. Ineffective *learning*, in turn, leads to suboptimal assignment decisions based on inaccurate payoff estimates, which increases *queueing* delay and lowers system payoff under uncertain agent dynamics. Such closed-loop interactions, if not properly managed, can produce a vicious cycle that severely degrades the overall system performance (see further discussions in Section 3). Second, while prior work in [JKK16, MX16, JKK17] attempts to deal with this closed-loop interactions, their policies explicitly divide exploration (where learning occurs) and exploitation (where queueing and control occur) into two stages, by assuming perfect knowledge of various uncertainty, such as the class-dependent payoff vectors, the number of tasks per client, and/or the "shadow prices" (which in turn require knowledge of the statistics of the client arrival dynamics). On the contrary, our algorithm requires no such prior knowledge and seamlessly integrates exploration, exploitation, and dynamic assignment at all times. Third, the analyses in [JKK16, JKK17] focus exclusively on the stationary setting. In contrast, our work carefully characterizes the transient behavior of the system and obtains a payoff gap which holds for any finite time horizon, providing important design insight when real systems either operate at early phases or experience recent changes of statistics. Finally, our proposed algorithm can scale to large systems with many clients and servers, which is in sharp contrast to the previous work in both [BM15] (which only deals with two types of clients) and [SGMV17] (which uses an exponential number of virtual queues).

The rest of the paper is organized as follows. The system model is defined in Section 2, and related approaches are discussed in Section 3. We then present our new algorithm and main analytical results in Section 4. Simulation results are shown in Section 5, and key proofs are presented in Section 6. Finally, we conclude and discuss possible future directions.

2 System Model

We model an online service platform as a multi-server queueing system that can process tasks from a dynamic population of clients, as shown in Fig. 1. Time is slotted and the system is empty at time 0. At the beginning of each time-slot $t \geq 1$, a new client arrives with probability $\lambda/N < 1$, independently of other

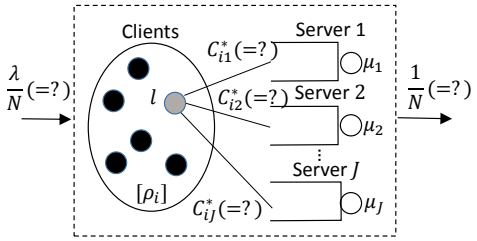

Figure 1: Uncertain client dynamics and payoffs.

time-slots. Upon arrival, the client carries a random number of tasks that is geometrically distributed with mean N. Note that in this way the total rate of arriving tasks is λ. A client leaves the system once all of her tasks have been served. We assume that each client must belong to one of I classes. There is a probability distribution $[\rho_i, i = 1, ..., I]$ with $\sum_{i=1}^{I} \rho_i = 1$, such that a newly-arriving client belongs to class i with probability ρ_i, independently of other clients. Let $\mathcal{S} = \{1, 2, \ldots, J\}$ denote the fixed set of servers. Each server $j \in \mathcal{S}$ can serve exactly μ_j tasks in a time-slot. Let $\mu = \sum_{j=1}^{J} \mu_j$.

Each class i is associated with a payoff vector $[C^*_{ij}, j = 1, ..., J]$, where $C^*_{ij} \in [0, 1]$ is the expected payoff when a task from a class-i client is served by server j. Note that all tasks associated with a client have the same payoff vector. *However, the arrival rate λ, the class distribution ρ_i, the expected number N of tasks per client, the class label of a new client, her total number of tasks, and her payoff vector are all unknown to the system.* The system does know the identity of the servers and their service rates μ_j. Another important quantity that the system can learn from is that, after a task from a client is served by server j, the system can observe a noisy payoff of the task. Conditioned on the task being from a class-i client, this noisy payoff is assumed to be a Bernoulli random variable with mean C^*_{ij}, (conditionally) independently of other tasks and servers.

The goal of the system is then to use the noisy payoff feedback to learn the payoff vectors of the clients and to assign their tasks to servers in order to maximize the total system payoff. Such decisions must be adaptively made without knowing the statistics of the client arrivals and departures. At each time-slot t, let $n(t)$ be the total number of clients in the system (including any new arrival at the beginning of the time-slot). (Note that $n(0) = 0$.) Let $p^l_j(t)$ be the expected number of tasks from the l-th client that are assigned to server j in this time-slot, $l = 1, ..., n(t)$. Then, the decision at time t of a policy Π maps from the current state of the system (including all payoff feedback observed before time t) to the quantities $p^l_j(t)$. Such decisions lead to the evolution of the following two types of queues. First, let $q_j(t)$ denote the number of tasks waiting to be served at server j at the beginning of time-slot t. Let $Y^l_j(t)$ be the actual number of tasks from the l-th client that are assigned to server j at time-slot t, with mean given by $p^l_j(t)$. The dynamics of the task-queue $q_j(t)$ can then be described as

$$q_j(t+1) = \max\left[q_j(t) + \sum_{l=1}^{n(t)} Y^l_j(t) - \mu_j, 0\right]. \tag{1}$$

Second, let $n_i(t)$ be the number of class-i clients in the system at the beginning of time t. (We caution that, while the operator sees $n(t)$, $n_i(t)$ cannot be directly observed because the class labels of the clients are unknown.) The dynamics of the client-queue $n_i(t)$ can be described as

$$n_i(t+1) = n_i(t) + U_i(t+1) - D_i(t), \tag{2}$$

where $U_i(t+1)$ is the number of client arrivals of class i at the beginning of time $t+1$ and is Bernoulli with mean λ_i/N, and $D_i(t)$ is the number of client departures of class i at time-slot t. Because the total number

of tasks per client is assumed to be geometrically distributed, conditional on $[Y_j^l(t)]$ and $[n_i(t), i \in I]$, the departure $[D_i(t), i \in I]$ is independent of everything else.

The expected payoff per unit time of such a policy Π for a given time horizon T is defined as:

$$R_T(\Pi) = \frac{1}{T} \sum_{t=1}^{T} \sum_{j=1}^{J} \mathbb{E}\left[\sum_{l=1}^{n(t)} p_j^l(t) C^*_{i(l),j} \right], \tag{3}$$

where $i(l)$ denotes the underlying (but unknown) class of the l-th client. In contrast, if an "oracle" knew in advance the statistics of the arrival and departure dynamics (i.e., λ, ρ_i and N), as well as the class label and payoff vector of each client, one can formulate the following linear program:

$$R^* = \max_{[p_{ij}] \geq 0} \quad \lambda \sum_{i=1}^{I} \rho_i \sum_{j=1}^{J} p_{ij} C^*_{ij} \tag{4}$$

$$\text{subject to} \quad \lambda \sum_{i=1}^{I} \rho_i p_{ij} \leq \mu_j \text{ for all servers } j, \tag{5}$$

$$\sum_{j=1}^{J} p_{ij} = 1 \text{ for all classes } i, \tag{6}$$

where p_{ij} is the probability that a task from a class-i client is assigned to server j. It is not difficult to show that R^* provides an upper bound for the expected payoff per unit time $R_T(\Pi)$ under any policy Π [JKK16, JKK17]. Thus, our goal is to achieve a provably small gap $R^* - R_T(\Pi)$ for any given time T.

3 Related Approaches in the Literature and Their Weakness

Before we present our solution to this problem, we briefly overview approaches to related problems in the literature and discuss why they cannot be used to solve the problem that we introduced in Section 2. First, the work in [JKK16, JKK17] also studies how to maximize the average payoff for a system with unknown client types. (The notion of "type" in [JKK16, JKK17] is comparable to "class" in this paper, while "worker" is comparable to "client" in this paper.) However, there are a number of crucial differences. In [JKK16, JKK17], the expected payoff vector of each type is assumed to be known in advance. Further, the theoretical results in [JKK16, JKK17] assume that the policy knows the "shadow prices," which in turn requires that the statistics of the client dynamics (i.e., λ, ρ_i and N) are known in advance. Moreover, the policies in [JKK16, JKK17] explicitly divide exploration (where learning occurs) and exploitation (where queueing and control occur) into two stages. In order to control the length of the exploration stage, these policies require prior knowledge of the total number of tasks for each client. In contrast, in our model neither the payoff vector for each type (i.e., each "class" in this paper) nor the client arrival/departure statistics are known to the system. As a result, the theoretic policies in [JKK16, JKK17] are not applicable in our setting. Although [JKK17] also provides a heuristic policy that uses queue length to replace the shadow price, no theoretical performance guarantee is proved. Moreover, this heuristic policy requires that the unused service of the servers can be queued and utilized later. Since our model does not allow such queueing of unused service, we cannot use this heuristic policy either. One could argue that, after the system has been operated in a stationary setting for a long time, the operator may eventually be able to learn the arrival/departure statistics and the class-dependent payoff vectors, in which case the theoretic policies in [JKK16, JKK17] could then be used (although acquiring such knowledge would require highly non-trivial learning procedures for estimating both the clients' payoff vectors and their underlying classes.) Still, the setting studied in this

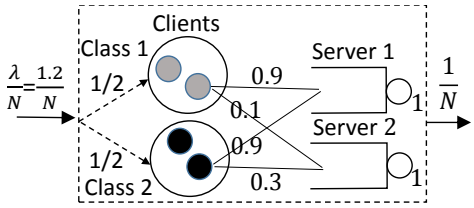

Figure 2: Myopic matching is suboptimal.

paper, as well as the adaptive algorithm that we will develop, will be important both for the early phase of the system operation, and when the class composition or arrival/departure statistics experience recent changes. Finally, note that the work in [MX16] also assumes the knowledge of the payoff vectors for all types in advance and divides exploration and exploitation into distinct stages, although [MX16] does not aim to maximize the system payoff as in this paper.

Myopic Matching: A second line of related work is the multi-player MAB formulation in [KNJ14, NKJ16, GKJ12, AMTS11, LEGJP11, LPT13, CP15, SRJ17, BTZ15], which is also a common way in the literature to study online learning in a two-sided system, where clients and servers with unknown payoffs need to be matched to maximize the system payoff. However, this class of work assumes that both the client population and the server population are fixed. In contrast, in our model the client population is constantly changing. Thus, the multi-player MAB solution can at best be viewed as a *myopic* solution for a snapshot of our system in time. It is not difficult to envision that such a myopic matching strategy (for each snapshot in time) is sub-optimal in the long run with client dynamics. Indeed, as illustrated below, this sub-optimality exists even when the payoffs vectors are known in advance!

Example (Myopic matching is suboptimal): Assume two servers, each of service rate $\mu_j = 1$ as shown in Fig. 2. There are two classes, and a new client belongs to one of the two classes with equal probability 0.5. The payoff vectors of the two classes are $[0.9\ 0.1]$ and $[0.9\ 0.3]$, respectively. Assume $N = 100$ and $\lambda/N = 1.2/N$. It is easy to see that the optimal solution should assign all class-1 tasks to server 1, $2/3$ of class-2 tasks to server 1, and the rest $1/3$ of class-2 tasks to server 2. The resulting optimal payoff is $0.6 * 0.9 + 0 * 0.1 + 0.4 * 0.9 + 0.2 * 0.3 = 0.96$. However, if a myopic matching strategy is used, with close-to-1 probability two tasks will be assigned at each time, one of which will have to be assigned to server 2. Thus, the overall payoff is approximately upper-bounded by $1.2 * (0.9 + 0.3)/2 = 0.72$. As we can see, the myopic matching strategy focuses too much on maximizing the current payoff, and as a result sacrifices future payoff opportunities. Naturally, if the class labels and payoff vectors are unknown and need to be learned, a similar inefficiency will incur.

Queue-Length Based Control: The above example clearly illustrates the need to cater to uncertain client dynamics in the system. In the literature, a third line of related work uses queue-length based contol to balance current payoff with long-term payoff in the presence of uncertain arrival/departure dynamics [TE92, NML05, Chi04, ES05, LSS06]. This line of work usually assumes that each client's payoff vector is known. By building up queues of unserved tasks at the servers, one can use the queue length q_j at server j as a "shadow price" that captures the congestion level at the server. The system operator then adjusts each client's payoff parameter C^*_{ij} to $C^*_{ij} - q_j/V$ with a proper choice of V, and assigns the next task to the server with the highest adjusted payoff. When the payoff vector of each client is known in advance, this type of queue-length based control can be shown to attain near-optimal system payoff when the parameter V is large [TS12], although the length of the server-side queues also grows with V.

However, when the payoff vector of each client is unknown, such a queue-length based control leads to complicated closed-loop interactions between queueing and learning. In one direction, *queueing* degrades *learning* in at least two ways. First, because the operator only learns the noisy payoff of a task *after* the task is served, excessive server-side queues (esp. when V is large) not only delay the service of the tasks, but also

delay the payoff feedback. As a result, the learning of the clients' uncertain payoff vectors is also delayed. (We will refer to this effect as the "learning delay.") Second, because all clients compete for the limited server capacity, the learning process of some clients will inevitably be slowed down, which also results in poor estimate of their payoff vectors. (We will refer to this second effect as the "learning slow-down.") Then, in the opposite direction, ineffective *learning* in turn affects *queueing* because assignment decisions have to be made based on delayed or inaccurate payoff estimates. Such sub-optimal decisions not only lower system payoff, but also increase queueing delay. Together, the above closed-loop interactions between queueing and learning will produce complex system dynamics which, if not designed and controlled properly, can severely degrade system performance. Indeed, as we will demonstrate in Section 5, a straightforward version of such queue-length based control will lead to lower system payoff when combined with online learning.

Online Matching: There is a fourth line of related work on online matching that also aims to address uncertainty in future client dynamics (albeit in an adversarial setting) [AHL13, CZL08, FKM+09, ZCXL13]. However, similar to queue-length based control, such studies typically do not deal with payoff uncertainty either.

Finally, although [BM15] and [SGMV17] also study the integration of learning and control, they either allow only a small system with two types of clients and two servers [BM15], or have to use an exponential number of virtual queues to approach optimality [SGMV17]. In contrast, in this work we aim to develop computationally-efficient algorithms that can be implemented even for large systems.

In summary, it remains an open question how to seamlessly integrate online learning with adaptive control when there is uncertainty in both client dynamics and payoff vectors, and how to account for the complex closed-loop interactions between queueing and learning. Below, we will propose our new algorithm and analytical techniques that address these difficulties.

4 Dynamic Assignment with Online Learning under Uncertainties

In this section, we present a new algorithm that seamlessly integrates online learning with dynamic task assignment to address the aforementioned closed-loop interactions. Unlike classical queue-length based control policies that rely on the task-queues as the "shadow prices," we instead eliminate the task-queues at the servers, and thus eliminate the "payoff-feedback delay" altogether. This is achieved by controlling the number of tasks assigned to each server j at each time-slot to be always no greater than its service rate μ_j. Therefore, the length of the server-side task-queue $q_j(t)$ given by (1) is trivially zero at all times. However, without the server-side task-queues or "shadow prices", we need another congestion indicator to guide us in the dynamic assignment of tasks. Here, we propose to rely on the number of backlogged clients in the system. Specifically, we use the solution to a utility-maximization problem (that is based on the current number of backlogged clients in the system) to help us trade-off between the current and future payoffs. The parameters of the utility-maximization problem are carefully chosen to also control the *learning slow-down* of all clients in a fair manner. We remark that while the structure of our proposed utility-guided algorithm bears some similarity to that of flow-level congestion control in communication networks [LSS08, DLK01, BM01], the focus therein was only on the system stability (but not the system payoff), and there was no online learning of uncertain parameters. *To the best of our knowledge, our work is the first in the literature that uses utility-guided adaptive control policies with online learning to optimize system payoff.*

4.1 Utility-Guided Dynamic Assignment with Online Learning

We present our algorithm in Algorithm 1. At each time-slot t, the algorithm operates in three steps. Step 1 generates an estimate $[C_j^l(t)]$ of the payoff vector for each client l based on previous payoff feedback (Lines

Algorithm 1: Utility-Guided Dynamic Assignment with Online Learning

For every time slot t:
Update the total number of clients $n(t)$ (including newly arriving clients)
Step 1: Form truncated UCB payoff estimates
for $l = 1 : n(t)$ **do**
  for $j = 1 : J$ **do**
    if $h_j^l(t-1) = 0$ *or client* l *is new* **then** $C_j^l(t) \leftarrow 1$; $h_j^l(t) \leftarrow 0$;
    else Set $C_j^l(t)$ according to (9) ;
Step 2: Solve $[p_j^l(t)]$ *for the optimization problem*

$$\max_{[p_j^l] \geq 0} \quad \sum_{l=1}^{n(t)} \left\{ \frac{1}{V} \log \left(\sum_{j=1}^{J} p_j^l \right) + \sum_{j=1}^{J} p_j^l (C_j^l(t) - \gamma) \right\} \tag{7}$$

$$\text{sub to} \quad \sum_{l=1}^{n(t)} p_j^l \leq \mu_j, \quad \text{for all servers } j \in \mathcal{S}. \tag{8}$$

Step 3: Assign tasks and obtain noisy payoff feedback
Initialize to zero the number of tasks from client l to be assigned to server j, i.e., $Y_j^l(t) = 0$.
for $j = 1 : J$ **do**
  for $\nu = 1 : \mu_j$ **do**
    Choose a client l^* randomly such that the probability of choosing client l is equal to $p_j^l(t)/\mu_j$.
    Assign one task from client l^* to server j and let $Y_j^{l^*}(t) \leftarrow Y_j^{l^*}(t) + 1$;
for $l = 1 : n(t)$ **do**
  for $j = 1 : J$ **do**
    Observe $Y_j^l(t)$ number of Bernoulli noisy payoffs
    $X_j^l(t,1), \ldots, X_j^l\left(t, Y_j^l(t)\right) \overset{\text{i.i.d.}}{\sim} \text{Bern}\left(C_{i(l),j}^*\right)$;
    Update $h_j^l(t)$ and $\overline{C}_j^l(t)$ according to $Y_j^l(t)$ and $X_j^l(t,\cdot)$;
  $h^l(t) = \sum_{j=1}^{J} h_j^l(t)$;
Clients with no remaining tasks leave the system.

4-7). Note that $n(t)$ is the current number of clients in the system, including any newly-arriving client. For each client $l = 1, ..., n(t)$, if no task of this client has been assigned to server j yet, we set $C_j^l(t) = 1$ (Line 6); otherwise, we use a truncated Upper-Confidence-Bound (UCB) estimate [ACBF02] to generate $C_j^l(t)$ (Line 7):

$$C_j^l(t) = \min \left\{ \overline{C}_j^l(t-1) + \sqrt{\frac{2 \log h^l(t-1)}{h_j^l(t-1)}},\ 1 \right\}, \tag{9}$$

where $h_j^l(t-1)$ is the number of tasks from client l that have previously been assigned to server j before the end of the $(t-1)$-th time-slot and $h^l(t-1) = \sum_{j=1}^{J} h_j^l(t-1)$; $\overline{C}_j^l(t-1)$ is the empirical average payoff of client l based on the received noisy payoff feedback for server j until the end of the $(t-1)$-th

time-slot. Since the true payoffs C_{ij}^*'s are within $[0,1]$, we truncate the UCB estimate at threshold 1 in (9). Then, Step 2 solves a maximization problem (Line 9), subject to the capacity constraint of each server, to obtain the values of $p_j^l(t)$, which corresponds to the expected number of new tasks of client l to be assigned to server j in time-slot t. The objective (7) can be viewed as the sum of some utility functions over all clients. The parameter γ in (7) is chosen to be strictly larger than 1, and V is a positive parameter. Finally, Step 3 determines the exact number $Y_j^l(t)$ of tasks from client l that are assigned to server j (Lines 12-14). The values of $Y_j^l(t)$ are randomly chosen in such a way that (i) each server j receives at most μ_j tasks, i.e., $\sum_{l=1}^{n(t)} Y_j^l(t) \leq \mu_j$; and (ii) the expected value of $Y_j^l(t)$ is equal to $p_j^l(t)$. The tasks are then sent to the servers and new noisy payoffs are received (Lines 15-19).

Before we present our analytical results, we make several remarks on the design of our proposed algorithm.

Seamless Integration of Learning and Control: First, recall that the policies in [JKK16, MX16, JKK17] separate exploration and exploitation into distinct stages by assuming perfect knowledge of class-dependent payoff vectors, the total number of tasks per client, and/or "shadow prices." In contrast, our algorithm does not require such prior knowledge at all and seamlessly integrates exploration, exploitation, and dynamic assignment at all times.

Zero Payoff-Feedback Delay: Second, note that by Step 3 of the algorithm, the number of assigned tasks to each server j is no more than the service rate μ_j. Since server j can serve exactly μ_j tasks per time-slot, there is no longer any "payoff-feedback delay," i.e., all payoff feedback will be immediately revealed at the end of the time-slot.

The Importance of Fairness: Third, if we removed the logarithmic term in (7) and set $\gamma = 0$, then the maximization problem in (7) would have become a myopic matching policy that maximizes the total payoff in time-slot t based on the current payoff estimates. However, such a myopic matching policy focuses too much on the current payoff, and as a result underperforms in terms of the long-term average payoff. Instead, the logarithmic term in (7) serves as a concave utility function that promotes fairness [LSS08, DLK01, BM01], so that even clients with low payoff estimates can still receive some service (i.e., $\sum_{j=1}^{J} p_j^l(t)$ is always strictly positive). This fairness property is desirable in two ways. First, it has an eye for the future (which is somewhat related to the fact that fair congestion-control ensures long-term system stability [BM01]), so that we can strike the right balance between current and future payoffs. Second, it also controls the *learning slow-down* of all clients in a fair manner. Specifically, thanks to this fairness property, we can show that the "learning rate" of each client based on imprecise payoff estimates will not be too far off from the "learning rate" of the client based on its true payoff vector, which constitutes a crucial step in our analysis (see Section 6.3).

Parameter Choice: Fourth, the choice $\gamma > 1$ in (7) and the truncation of the UCB estimates in (9) also play a crucial role in achieving near-optimal system payoff. If $\gamma = 0$, then every client l will have the incentive to increase $\sum_{j=1}^{J} p_j^l(t)$ by utilizing as many available servers as possible, *even when the server payoff is very close to zero*. Such an aggressive approach typically leads to suboptimal performance because in many settings the optimal policy must restrict some clients to only use the high-payoff servers. Instead, by choosing $\gamma > 1 \geq C_j^l(t)$, the second term inside the summation in (7) becomes negative. As a result, the operator assigns clients to low-payoff servers only if the derivative of the logarithmic term, which is equal to $1/[V \sum_{j=1}^{J} p_j^l(t)]$, is sufficiently large. This feature leads to an inherent "conservativeness" of our algorithm in choosing low-payoff servers. As the parameter V increases, our algorithm becomes more and more conservative in choosing low-payoff servers, which benefits the long-term payoff. On the other hand, it also increases the number of clients backlogged in the system, leading to extra payoff loss for a finite time horizon T. Our theoretical results below will capture this tradeoff.

Complexity: Last but not least, the maximization problem in Step 2 is a convex program and can be effectively solved. Thus, our proposed algorithm can scale to large systems with many clients and servers,

which is in contrast to the work in both [BM15] (which only deals with two types of clients) and [SGMV17] (which uses an exponential number of virtual queues).

4.2 Main Results

We now present our main results for the performance guarantees of the proposed algorithm. For simplicity, we denote $\lambda_i = \lambda\rho_i$, and hence $\lambda = \sum_{i=1}^{I} \lambda_i$. Recall that $\mu = \sum_{j=1}^{J} \mu_j$ and $n(0) = \sum_{i=1}^{I} n_i(0) = 0$. The first results gives an upper bound to the mean number of backlogged clients in the system at any time t.

Theorem 1. *Suppose that the arrival rate λ is strictly less than the total service capacity, i.e., $\lambda < \mu$. Then for any time t,*

$$\mathbb{E}\left[n(t)\right] \leq \frac{2\mu}{\mu - \lambda}\left(1 + \frac{\mu^2\gamma}{\gamma - 1}\right) + \mu\gamma V. \tag{10}$$

Theorem 1 immediately implies the system is stable in the mean [KM95]. i.e.,

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[n(t)] < \infty.$$

The upper bound in (10) characterizes the effect of V. As V increases, our algorithm becomes more conservative in choosing low-payoff servers, leading to more clients backlogged in the system.

Departing from the standard Lyapunov technique in proving system stability [LSS08, DLK01, BM01], our proof of Theorem 1 relies on a careful coupling between $n(t)$ and a Geom/Geom/μ queue with Bernoulli arrivals and Binomial departures (details in Appendix A.6).

The second result below further characterizes the payoff gap of the proposed algorithm compared to the oracle.

Theorem 2. *Suppose $N \log N \geq 1$. The gap between the upper bound (4) and the expected payoff of Algorithm 1 at any time horizon T is bounded by:*

$$\begin{aligned}\sum_{i=1}^{I}\sum_{j=1}^{J} \lambda_i p_{ij}^* C_{ij}^* - \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{J} \mathbb{E}\left[\sum_{l=1}^{n(t)} p_j^l(t) C_{i(l),j}^*\right] \\ \leq \frac{\beta_1}{V} + \beta_2\sqrt{\frac{\log N}{N}} + \beta_3\frac{N(V+1)}{T},\end{aligned} \tag{11}$$

where $[p_{ij}^*]$ is the optimal solution to (4), and

$$\begin{aligned}\beta_1 &= \frac{I}{2} + \frac{\mu^2}{2}\left(1 + \frac{\gamma^2}{(\gamma-1)^2}\right)\sum_{i=1}^{I}\frac{1}{\lambda_i}, \\ \beta_2 &= 4\sqrt{2}\lambda(J+\mu) + 3\lambda\left(1 + \frac{J\gamma^2}{(1-\gamma)^2}\right)\mu, \\ \beta_3 &= \frac{2\mu\gamma}{\mu-\lambda} + \frac{2\mu^3\gamma^2}{(\mu-\lambda)(\gamma-1)} + \mu\gamma^2.\end{aligned} \tag{12}$$

To the best of our knowledge, this is the first result in the literature that characterizes the payoff gap for this type of utility-guided adaptive controllers under payoff uncertainty. The above result is quite appealing as it separately captures the impact of the uncertainty in client dynamics and the impact of the uncertainty

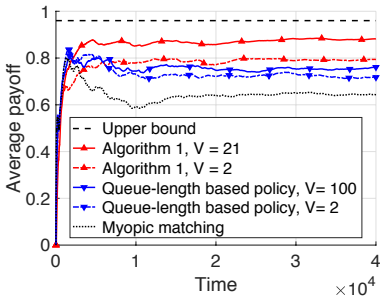

Figure 3: The evolution of the time-averaged payoff under the three policies as simulation time advances. The upper bound (4) is 0.96 (the horizontal line).

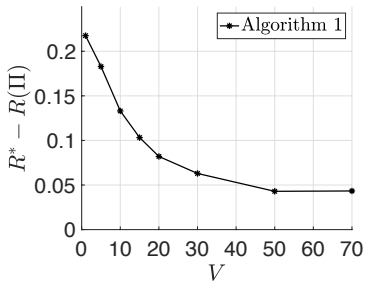

Figure 4: With a fixed $N = 100$, the payoff gap of Algorithm 1 decreases and eventually saturates as V increases.

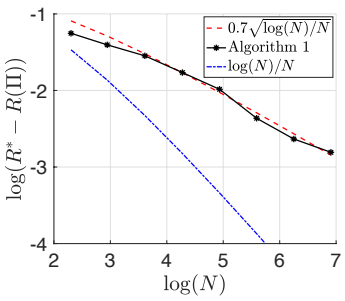

Figure 5: With $V = \sqrt{N/\log N}$, the payoff gap of Algorithm 1 scales as $\Theta(\sqrt{\log N/N})$.

in payoffs for any finite time horizon T. In (11), the first term on the right-hand-side is of the order $1/V$, capturing the impact of the uncertainty in client dynamics (e.g., we do not know what are the values of λ_i and N). The second term in (11) is of the order $\sqrt{\log N/N}$ and related to the notion of "regret" in typical MAB problems. It captures the payoff loss due to the uncertainty in payoffs as a function of the total number of tasks per client, i.e, the tradeoff between exploration and exploitation [LR85, ACBF02]. The third term in (11) is of the order $N(V+1)/T$, characterizing the payoff loss incurred by clients backlogged in the system. Given T, the first term and the third term reveal an interesting tradeoff as V increases. On the one hand, the first term will approach zero at the speed of $1/V$, indicating that the policy adapts to the unknown client dynamics; on the other hand, the third term will increase linearly with V due to the payoff loss incurred by clients backlogged in the system. If N and T are known in advance, we can tune V to be $\min\{T/N, \sqrt{T/N}\}$, so that the payoff-gap upper-bound is minimized and becomes of the order $\max\{N/T, \sqrt{N/T}\} + \sqrt{\log N/N}$. As a consequence, as T increases, the payoff gap first decreases at a rate of T^{-1} for $T \leq N$ and then at a rate of $T^{-1/2}$ for $N \leq T \leq N^2/\log N$ and finally gets saturated at $\sqrt{\log N/N}$ for $T \geq N^2/\log N$.

The $\sqrt{\log N/N}$ regret here may seem inferior compared to the $\log N/N$ regret in [JKK16, JKK17]. However, the regret in [JKK16, JKK17] only holds under the stationary setting $T \to \infty$ and assuming knowledge of various uncertainty, such as the class-dependent payoff vectors, the number of tasks per client, and/or "shadow prices" (which in turn require knowledge of the statistics of the client arrival dynamics). In contrast, our payoff gap characterizes the transient behavior of the system and holds for any finite time T and finite $N \geq 2$ without any such prior knowledge. It remains open whether our $\sqrt{\log N/N}$ regret can be further reduced to $\log N/N$.

Our proof of Theorem 2 builds upon the standard Lyapunov drift technique [NML05, LSS08, DLK01, BM01] and finite-time regret analysis [ACBF02]; however, our proof employs new techniques to carefully account for the closed-loop interactions between the queueing dynamics and the learning processes (see Section 6).

5 Numerical Results

We consider the same setup as described in the counter example in Section 3 and illustrated in Fig. 2. In particular, there are two servers and two classes of clients. Define the key parameters as follows: $\mu_j = 1$ for each server $j = 1, 2$; $\lambda_i = 0.6$ for each class $i = 1, 2$, and hence $\lambda = \lambda_1 + \lambda_2 = 1.2$. The expected number of tasks per client is initially set to be $N = 100$, but we will vary the value later. The true payoff vectors

for class 1 and class 2 are given by $[0.9\ 0.1]$ and $[0.9\ 0.3]$, respectively, although they are unknown to the operator. Note that server 1 has larger expected payoff for both class 1 and class 2. However, its service rate is insufficient to support all clients. Hence, this contention must be carefully controlled when the system aims to maximize the payoff. For a given policy Π and simulation time T, we report the average system payoff $R_T(\Pi)$ per unit time.

5.1 Two Other Policies for Comparison

We compare our proposed Algorithm 1 with the myopic matching policy and a queue-length based policy. These two policies also use UCB to estimate unknown payoff vectors. The myopic matching policy aims at maximizing the total payoff for the current time-slot. Specifically, at each time slot t, the policy solves a modified maximization problem of the form (7)–(8), but with the logarithmic term removed and with $\gamma = 0$. Then, based on the solution to this modified maximization problem, the tasks are assigned to each server in the same way as described in Step 3 in Algorithm 1. We expect that the myopic matching policy incurs relatively large payoff loss because it does not look at the future.

The queue-length based policy maintains a queue of tasks at each server j, and uses the length of this task-queue $q_j(t)$ at time-slot t to indicate the congestion level at server j. At each time-slot t and for each client l, the operator finds the server $j^*(l)$ with the highest queue-adjusted payoff, i.e., $j^*(l) = \arg\max_j\{C_j^l(t) - q_j(t)/V\}$. The operator then adds one task from each client l to the end of the task queue at the server $j^*(l)$. Every server j then processes $\mu_j = 1$ task from the head of its own queue and observes the random payoff generated. As discussed in Section 3, one weakness of such a queue-length based policy is that, when tasks are waiting in the queues, the system cannot observe their payoff feedback right away. Hence, there is significant payoff feedback delay, which in turn leads to suboptimal assignment decisions for subsequent tasks.

5.2 Performance Comparisons

We fix $\gamma = 1.1$ for our proposed Algorithm 1 in all experiements, but we may vary N and V. The first set of experiements give a general feel of the dynamics of different policies. In Fig. 3, we fix $N = 100$ and plot the evolution of the time-averaged system payoff up to time T under the three policies (with different V), as the simulation time T advances. We can see that, even when N is not very large ($N = 100$), the system payoff under our proposed Algorithm 1 with $V = 21$ (the solid curve with marker ▲) approaches the upper bound (4) (the horizontal dashed line). Further, comparing $V = 2$ (marker ▲, dashed curve) with $V = 21$ (marker ▲, solid curve), we observe significantly higher system payoff under Algorithm 1 when V increases. In comparison, the payoff achieved by the myopic matching policy (the lowest dotted curve) is significantly lower. The performance of the queue-length based policy (the two curves with marker ▼) also exhibits a noticable gap from that of the proposed Algorithm 1. Further, even when V increases from $V = 2$ (marker ▼, dashed curve) to $V = 100$ (marker ▼, solid curve), the improvement of the queue-length based policy is quite limited. This result suggests that, due to the increase in payoff-feedback delay, controlling V is not effective in improving the performance of the queue-length based policy.

In Fig. 4, we fix $N = 100$, increase V and plot the payoff gap (compared to the upper bound (4)) of our proposed Algorithm 1 over $T = 7 \times 10^5$ time slots. Each plotted data point represents the average of 5 independent runs. We can observe that initially the payoff gap decreases significantly with V, but eventually it saturates at $V \geq 50$. This is because the regret due to small N, i.e., the second term of (11), eventually dominates the payoff gap when V is large. A similar figure with increasing N but fixed V (not shown) shows a similar saturation behavior when N is large. Such saturation makes it difficult to assess whether the scaling reported in (11) is accurate. To answer this question, we next set $V = \sqrt{N/\log N}$. In this way, both the first and second terms in (11) are of the order $\Theta(\sqrt{\log N/N})$. We then increase N (and V simultaneously)

Table 1: List of Notations

Symbol	Meaning
$C_{ij}^k(t)$	UCB estimate for server j by client k of class i
$C_j^l(t)$	UCB estimate for server j by client l
C_{ij}^*	True expected payoff for server j and class i
$\overline{C}_{ij}^k(t), \overline{C}_j^l(t)$	Empirical payoff average
$\gamma > 1$	Parameter in (7) and (14)
p_{ij}^*	Solution to upper bounds (4) or (17)
$p_{ij}^k(t), p_j^l(t)$	Solution to (7) or (14) using UCB payoff estimates
$\widehat{p}_{ij}^k(t)$	Solution to (14) with $C_{ij}^k(t)$ replaced by C_{ij}^*

and show in Fig. 5 how the payoff gap of our Algorithm 1 decreases with N on a log-log scale. The result indeed matches well with a $\Theta(\sqrt{\log N/N})$ scaling. Note that there is a noticable difference in slope from the $\Theta(\log N/N)$ scaling. The difference remains even if we set V to be larger at $V = N/\log N$. Thus, the results suggest that the performance bound in (11) for our proposed Algorithm 1 may be tight.

6 Proofs

The rest of the paper is devoted to proving our main results. In this section, we will focus on the proof of Theorem 2 to highlight how to deal with the main difficulties due to the interactions between the queueing dynamics and the learning process. The proof for Theorem 1 is provided in Appendix A.6.

6.1 Equivalent Reformulation

We will use a Lyapunov-drift analysis with special modifications to account for the learning of uncertain payoffs. Note that for the purpose of this drift analysis, we can use the underlying class label of each client to keep track of the system dynamics, even though our algorithm does not know this underlying class label. Thus, we re-label the $n(t)$ clients at time-slot t as follows. Let $n_i(t)$ be the number of clients at time-slot t whose underlying class is i and $\mathcal{I}(t) = \{i : n_i(t) \geq 1\}$. We have $\sum_{i\in\mathcal{I}(t)} n_i(t) = n(t)$. For each class $i \in \mathcal{I}(t)$, we use $k = 1, ..., n_i(t)$ to index the clients of this class at time t. Similar to the notations $C_{i(l),j}^*, \overline{C}_j^l(t)$ and $C_j^l(t)$, we denote $C_{ij}^*, \overline{C}_{ij}^k(t)$ and $C_{ij}^k(t)$ as the true value, empirical average of past payoffs at time t, and the UCB estimate at time t, respectively, for the payoff of server j serving a task from the k-th client of the underlying class i. We also define $h_{ij}^k(t)$, $h_i^k(t) = \sum_{j=1}^J h_{ij}^k(t)$, and $p_{ij}^k(t)$ analogously to $h_j^l(t)$, $h^l(t)$ and $p_j^l(t)$. Thus, the UCB estimate (9) in Step 1 of our proposed Algorithm 1 is equivalent to

$$C_{ij}^k(t) \leftarrow \min\left\{\overline{C}_{ij}^k(t-1) + \sqrt{\frac{2\log h_i^k(t-1)}{h_{ij}^k(t-1)}},\ 1\right\}, \tag{13}$$

and the maximization problem (7) in Step 2 of our proposed Algorithm 1 is equivalent to:

$$\max_{[p_{ij}^k]\geq 0} \sum_{i\in\mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \left\{\frac{1}{V}\log\left(\sum_{j=1}^J p_{ij}^k\right) + \sum_{j=1}^J p_{ij}^k\left(C_{ij}^k(t) - \gamma\right)\right\} \tag{14}$$

$$\text{s.t.} \sum_{i\in\mathcal{I}(t)} \sum_{k=1}^{n_i(t)} p_{ij}^k \leq \mu_j, \quad \text{for all } j \in \mathcal{S}. \tag{15}$$

Our proof below will use these equivalent forms of the proposed Algorithm 1. In the rest of the analysis, we will use boldface variables (e.g., $\mathbf{n}$, $\mathbf{p}$, and $\mathbf{C}$) to denote vectors, and use regular-font variables (e.g., n_i, p_{ij}^k, and C_{ij}^k) to denote scalars. A list of notations is provided in Table 1.

6.2 Handling Uncertain Client Dynamics

Recall that $\lambda_i = \lambda\rho_i$, $\lambda = \sum_{i=1}^I \lambda_i$ and $\mu = \sum_{j=1}^J \mu_j$. Define the vector $\mathbf{n}(t) = [n_i(t), i = 1, \ldots, I]$ and $\mathbf{p}(t) = [p_{ij}^k(t), i \in \mathcal{I}(t), j = 1, \ldots, J, k = 1, \ldots, n_i(t)]$. With the above re-indexing of the clients, we choose the Lyapunov function $L(\mathbf{n}(t))$ as

$$L(\mathbf{n}(t)) = \frac{1}{2}\sum_{i=1}^I \frac{n_i^2(t)}{\lambda_i}. \tag{16}$$

Let $[p_{ij}^*]$ be the optimal solution of the upper bound in (4). Note that if we replace each C_{ij}^* in (4) by $C_{ij}^* - \gamma$, this will result in the same optimal solution. Thus, $[p_{ij}^*]$ is also the optimal solution to the following optimization problem:

$$\max_{[p_{ij}]\geq 0} \sum_{i=1}^I \lambda_i \sum_{j=1}^J p_{ij}(C_{ij}^* - \gamma), \text{ subject to (5) and (6).} \tag{17}$$

Next, we will add a properly-scaled version of the following term to the drift of the Lyapunov function $L(\mathbf{n}(t+1)) - L(\mathbf{n}(t))$:

$$\Delta(t) = \sum_{i=1}^I \sum_{j=1}^J \lambda_i p_{ij}^*(C_{ij}^* - \gamma) - \sum_{i\in\mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \sum_{j=1}^J p_{ij}^k(t)(C_{ij}^* - \gamma). \tag{18}$$

The value of $\Delta(t)$ captures the gap between the achieved payoff and the upper bound in (17), both adjusted by γ. The following lemma is the first step towards bounding the Lyapunov drift plus this payoff gap.

Lemma 3. *The expected drift plus payoff gap is bounded by*

$$\begin{aligned}&\mathbb{E}\left[L(\mathbf{n}(t+1)) - L(\mathbf{n}(t)) + \frac{V}{N}\Delta(t) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \\ &\leq \frac{1}{N}A_1(t) + \frac{V}{N}\sum_{i\notin\mathcal{I}(t)}\sum_{j=1}^J \lambda_i p_{ij}^*(C_{ij}^* - \gamma) + \frac{c_1 + c_2}{N},\end{aligned} \tag{19}$$

where $c_1 = \frac{I}{2}$, $c_2 = \frac{\mu^2}{2}\left(1 + \frac{\gamma^2}{(\gamma-1)^2}\right)\sum_{i=1}^I \frac{1}{\lambda_i}$, *and*

$$A_1(t) \triangleq \sum_{i\in\mathcal{I}(t)}\sum_{k=1}^{n_i(t)}\sum_{j=1}^J \left(\frac{n_i(t)}{\lambda_i} + V(C_{ij}^* - \gamma)\right)\left(\frac{\lambda_i p_{ij}^*}{n_i(t)} - p_{ij}^k(t)\right).$$

Proof. We first note that

$$n_i(t+1) = n_i(t) + U_i(t+1) - D_i(t),$$

where $U_i(t+1)$ and $D_i(t)$ are the number of client arrivals at the beginning of time $t+1$ and the number of departures at the end of time t, respectively, of class i. Note that $U_i(t)$ is a Bernoulli random variable with mean λ_i/N. For $i \notin \mathcal{I}(t)$, i.e., $n_i(t) = 0$, we have $D_i(t) = 0$ and $n_i(t+1) = U_i(t+1)$. It follows that

$$\mathbb{E}[n_i^2(t+1) - n_i^2(t) \mid n_i(t) = 0] = \mathbb{E}\left[U_i^2(t+1)\right] = \frac{\lambda_i}{N}. \tag{20}$$

For $i \in \mathcal{I}(t)$, the expected time-difference of $n_i^2(t)$ is bounded by

$$\begin{aligned}
&\mathbb{E}[n_i^2(t+1) - n_i^2(t) \mid \mathbf{n}(t), \mathbf{p}(t)] \\
&= \mathbb{E}\left[2n_i(t)\left(U_i(t+1) - D_i(t)\right) + \left(U_i(t+1) - D_i(t)\right)^2 \mid \mathbf{n}(t), \mathbf{p}(t)\right] \\
&\le 2n_i(t)\left(\frac{\lambda_i}{N} - \mathbb{E}\left[D_i(t)|\mathbf{n}(t), \mathbf{p}(t)\right]\right) + \frac{\lambda_i}{N} + \mathbb{E}\left[D_i^2(t)|\mathbf{n}(t), \mathbf{p}(t)\right] \\
&\le \frac{2n_i(t)}{N}\left(\lambda_i - \sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t)\right) + \frac{\lambda_i}{N} + \frac{\mu^2}{N}\left(1 + \frac{\gamma^2}{(\gamma-1)^2 N}\right),
\end{aligned} \tag{21}$$

where in the last step we have used the following bounds shown in Lemma 8 in Appendix A.1:

$$\mathbb{E}\left[D_i(t)|\mathbf{n}(t), \mathbf{p}(t)\right] \ge \frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) - \frac{\gamma^2\mu^2}{2(\gamma-1)^2 n_i(t) N^2}, \tag{22}$$

and $\mathbb{E}\left[D_i^2(t) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \le \frac{\mu^2}{N}$. Combining (20) and (21), the expected Lyapunov drift is then bounded by

$$\begin{aligned}
&\mathbb{E}\left[L((\mathbf{n}(t+1))) - L(\mathbf{n}(t)) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \\
&\le \frac{1}{N}\sum_{i\in\mathcal{I}(t)} \frac{n_i(t)}{\lambda_i}\left(\lambda_i - \sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t)\right) + \frac{c_1 + c_2}{N} \\
&= \frac{1}{N}\sum_{i\in\mathcal{I}(t)} \frac{n_i(t)}{\lambda_i}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J}\left(\frac{\lambda_i p_{ij}^*}{n_i(t)} - p_{ij}^k(t)\right) + \frac{c_1 + c_2}{N},
\end{aligned} \tag{23}$$

where c_1 and c_2 are defined in the lemma, and we use $N \ge 1$ in the inequality. Adding $\frac{V}{N}\Delta(t)$ in (18), the result then follows. Note that we need to single out the first term of (18) corresponding to $n_i(t) = 0$, which produces the second term of (19). □

Proof of Theorem 2. Recall that in Step 2 of our proposed algorithm, we have chosen $\gamma > 1 \ge C_{ij}^*$. Hence, the second term on the right-hand-side of (19) is always less than 0. Then, taking expectations of (19) over $\mathbf{n}(t)$ and $\mathbf{p}(t)$, summing over $0 \le t \le T-1$, and divided by T, we have

$$\begin{aligned}
&\frac{1}{T}\mathbb{E}\left[L(\mathbf{n}(T)) - L(\mathbf{n}(0))\right] + \frac{V}{TN}\sum_{t=0}^{T-1}\mathbb{E}\left[\Delta(t)\right] \\
&\le \frac{1}{NT}\sum_{t=0}^{T-1}\mathbb{E}\left[A_1(t)\right] + \frac{c_1 + c_2}{N}.
\end{aligned}$$

Since the system at time $t = 0$ is empty, i.e., $n(t) = 0$, it follows that $L(\mathbf{n}(0)) = 0$. In view of $L(\mathbf{n}(T)) \ge 0$, the last displayed equation gives

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\Delta(t)\right] \le \frac{1}{TV}\sum_{t=0}^{T-1}\mathbb{E}\left[A_1(t)\right] + \frac{c_1 + c_2}{V}. \tag{24}$$

Let R_T denote the expected payoff per unit time achieved by Algorithm 1 for a given time T as defined in (3). By definitions of $\Delta(t)$, R_T, and R^*, we get that

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\Delta(t)\right] \\
&= R^* - R_T - \gamma\lambda + \frac{\gamma}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{J}\mathbb{E}\left[\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t)\right].
\end{aligned} \tag{25}$$

We show in Appendix A.1 that the total number $D(t)$ of departures at time t satisfies

$$\mathbb{E}\left[D(t) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \le \frac{1}{N}\sum_{i\in I(t)}\sum_{j=1}^{J}\sum_{k=1}^{n_i(t)} p_{ij}^k(t).$$

It follows that

$$\begin{aligned}
&\lambda - \frac{1}{T}\sum_{t=0}^{T-1}\sum_{j=1}^{J}\mathbb{E}\left[\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t)\right] \\
&\le \lambda - \frac{N}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[D(t)\right] \\
&= \frac{N}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[U(t+1) - D(t)\right] \\
&= \frac{N}{T}\mathbb{E}\left[n(T)\right] \le \frac{\beta_3 N(V+1)}{\gamma T},
\end{aligned}$$

where $U(t)$ denotes the total number of arrivals at time t, and the last inequality holds in view of Theorem 1 and the definition of β_3 in (12). By combining the last displayed equation with (25) and (24), we get that

$$R^* - R_T \le \frac{\beta_3 N(V+1)}{T} + \frac{1}{TV}\sum_{t=1}^{T}\mathbb{E}\left[A_1(t)\right] + \frac{c_1+c_2}{V}. \tag{26}$$

In the rest of this section, we show that for all T,

$$\begin{aligned}
&\frac{1}{TV}\sum_{t=1}^{T}\mathbb{E}\left[A_1(t)\right] \\
&\le \frac{\lambda}{N}\left[4J\sqrt{2N\log N} + \mu\big(4\sqrt{2\log N} + 3 + \frac{3J\gamma^2}{(1-\gamma)^2}\big)\right].
\end{aligned} \tag{27}$$

Substituting (27) into (26) and invoking the assumption $N\log N \ge 1$, the result of Theorem 2 readily follows.

□

The remainder of the section will prove (27).

6.3 Bounding $A_1(t)$: Handling Payoff Uncertainty

As we will see soon, if there were no errors in the payoff estimates $[C_{ij}^k(t)]$, we would have obtained $A_1(t) \leq 0$ (see discussions after (31)). Thus, the key in bounding $A_1(t)$ is to account for the impact of the errors of the payoff estimates. In the rest of this subsection, we fix $\mathbf{n} = \mathbf{n}(t)$. Recall that $\mathcal{I}(t) = \{i | n_i(t) \geq 1\}$, and $\mathbf{p}(t) = [p_{ij}^k(t)]$ is the solution to the optimization problem (14). Denote vectors $\mathbf{W} = [W_{ij}^k, i \in \mathcal{I}(t), j = 1, ..., J, k = 1, ..., n_i(t)]$ and $\boldsymbol{\pi} = [\pi_{ij}^k, i \in \mathcal{I}(t), j = 1, ..., J, k = 1, ..., n_i(t)]$. We define the function

$$f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W}) \triangleq \sum_{i \in \mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \left[\log \left(\sum_{j=1}^{J} \pi_{ij}^k \right) + V \sum_{j=1}^{J} (W_{ij}^k - \gamma) \pi_{ij}^k \right]. \tag{28}$$

Note that when $\mathbf{W} = [C_{ij}^k(t)]$, the value of this function at $\boldsymbol{\pi} = \mathbf{p}(t)$ is precisely the objective function of (14) multiplied by V. Now, let $\widetilde{\mathbf{p}}(t)$ denote the vector $[\widetilde{p}_{ij}^k(t)]$ such that $\widetilde{p}_{ij}^k(t) = \dfrac{\lambda_i p_{ij}^*}{n_i(t)}$ for all $i \in \mathcal{I}(t)$, $j = 1, ..., J$ and $k = 1, ..., n_i(t)$. Using the concavity of the function f and the fact that $\frac{n_i(t)}{\lambda_i} + V(W_{ij}^k - \gamma)$ is the partial derivative of f at $\widetilde{\mathbf{p}}(t)$, the following lemma can be easily shown (see Appendix A.2 for details).

Lemma 4. *For any* $\mathbf{W} = [W_{ij}^k]$ *and* $\boldsymbol{\pi} = [\pi_{ij}^k]$, *it holds that*

$$\begin{aligned} &\sum_{i \in \mathcal{I}(t)} \sum_{j=1}^{J} \sum_{k=1}^{n_i(t)} \left(\frac{n_i(t)}{\lambda_i} + V(W_{ij}^k - \gamma) \right) \left(\frac{\lambda_i p_{ij}^*}{n_i(t)} - \pi_{ij}^k \right) \\ &\leq f(\widetilde{\mathbf{p}}(t)|\mathbf{n}, \mathbf{W}) - f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W}). \end{aligned} \tag{29}$$

The significance of this lemma is as follows. Let $\mathbf{C}^*$ denote the vector $[W_{ij}^k]$ such that $W_{ij}^k = C_{ij}^*$ for all $i \in \mathcal{I}(t)$, $j = 1, ..., J$ and $k = 1, ..., n_i(t)$. If we choose $\mathbf{W} = \mathbf{C}^*$ and $\boldsymbol{\pi} = \mathbf{p}(t)$ in Lemma 4, then $A_1(t)$ is simply the left-hand-side of (29). Applying Lemma 4, we then have

$$A_1(t) \leq f(\widetilde{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}^*) - f(\mathbf{p}(t)|\mathbf{n}, \mathbf{C}^*). \tag{30}$$

Next, denote $\widehat{\mathbf{p}}(t) = \left[\widehat{p}_{ij}^k(t)\right]$ as the maximizer of $f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{C}^*)$ over the constraint (15). Since $\widetilde{\mathbf{p}}(t)$ also satisfies the constraint (15), we have $f(\widetilde{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}^*) \leq f(\widehat{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}^*)$. Combining with (30), we get

$$A_1(t) \leq f(\widehat{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}^*) - f(\mathbf{p}(t)|\mathbf{n}, \mathbf{C}^*). \tag{31}$$

To appreciate the difference between the two terms, recall that $\widehat{\mathbf{p}}(t)$ maximizes $f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{C}^*)$, while $\mathbf{p}(t)$ maximizes $f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{C}(t))$, where we have denoted $\mathbf{C}(t) = [C_{ij}^k(t)]$. Clearly, if there were no errors in the payoff estimates $\mathbf{C}(t)$, i.e., if $\mathbf{C}(t) = \mathbf{C}^*$, we would have obtained $\mathbf{p}(t) = \widehat{\mathbf{p}}(t)$ and $A_1(t) \leq 0$ trivially. In order to bound the difference when $\mathbf{C}(t) \neq \mathbf{C}^*$, we use

$$\begin{aligned} &f(\widehat{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}^*) = f(\widehat{\mathbf{p}}(t)|\mathbf{n}, \mathbf{C}(t)) + A_3(t) \\ &\leq f(\mathbf{p}(t)|\mathbf{n}, \mathbf{C}(t)) + A_3(t) \qquad \text{(by the optimality of } \mathbf{p}(t)) \\ &= [f(\mathbf{p}(t)|\mathbf{n}, \mathbf{C}^*) + A_2(t)] + A_3(t), \end{aligned} \tag{32}$$

where the two equalities follow directly from the definition of f and that of $A_2(t)$ and $A_3(t)$ below:

$$A_2(t) = V \sum_{i \in \mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \sum_{j=1}^{J} \left(C_{ij}^k(t) - C_{ij}^* \right) p_{ij}^k(t),$$

$$A_3(t) = V \sum_{i \in \mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \sum_{j=1}^{J} \left(C_{ij}^* - C_{ij}^k(t) \right) \widehat{p}_{ij}^k(t).$$

Combining (31) and (32), we thus have

$$A_1(t) \leq A_2(t) + A_3(t). \tag{33}$$

6.4 Bounding $A_2(t)$

In this subsection, we wish to bound $\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[A_2(t)]$ for a given T. We define Λ as a particular realization of the sequence of client arrival-times up to time slot T. We use $\mathbb{E}_\Lambda$ to denote the conditional expectation given Λ. Given Λ, we slightly abuse notation and use the index k now to denote the k-th client of class i that arrives to the system. Let $t_1(k)$ denote the arrival time of this client, and $t_2(k)$ denote the *minimum* of her departure time and T. The following lemma is useful to bound the contribution of this client to $\sum_{t=1}^{T}\mathbb{E}[A_2(t)]$. Since the proof of this lemma is along similar lines as that of Lemma 7 in Section 6.5, we move it to Appendix A.4.

Lemma 5. *Suppose that the k-th arriving client of the underlying class i brings a total number a_k of tasks into the system. For all $1 \leq j \leq J$, it holds that*

$$\begin{aligned}&\mathbb{E}_\Lambda\left[\sum_{s=t_1(k)}^{t_2(k)}(C_{ij}^k(s) - C_{ij}^*)p_{ij}^k(s)\right]\\ &\leq 4\sqrt{2a_k\log a_k} + 4\mu_j\sqrt{2\log a_k} + 3\mu_j.\end{aligned} \tag{34}$$

Remark: The result of the lemma can be interpreted as follows. Suppose that $p_{ij}^k(s) = 1$ for all time s. Then, after s time-slots, the gap $C_{ij}^k(s) - C_{ij}^*$ is within the order of $\Theta(\sqrt{\log s/s})$ with high probability by the Chernoff-Hoeffding bound (see Lemma 9 in Appendix A.3). Summing over all $1 \leq s \leq a_k$, we get an expression on the order of the first term in (34). Of course, under our algorithm $p_{ij}^k(t)$ is random. Fortunately, here the loss given by the left-hand-side of (34) accumulates at the same rate as the probability $p_{ij}^k(t)$ of choosing a server (which will not be the case in Section 6.5). Thus, we may view the time as being slowed down at the rate of $\sum_{j=1}^{J}p_{ij}^k(t)$, and expect the bound in (34) to hold. However, the tricky part is that the value of $p_{ij}^k(t)$ also depends on previous values of $C_{ij}^k(s)$, $s < t$. Our proof uses a delicate martingale argument to take care of such dependency. We also note that this lemma is the main reason that we get $\Theta(\sqrt{\log N/N})$ regret in the second-term of (11). In the classical MAB problem in [ACBF02], there is a non-zero gap δ between the best arm and the second-best arm. Thus, once the estimation error is within δ, which takes $\Theta(\log N)$ time, learning can stop. As a result, the regret is on the order of $\Theta(\log N/N)$. In contrast, in our problem the notion of "best/second-best arms" is more fluid because they depend on the number of clients in the system. As a result, we do not have such a fixed gap δ, which is why we have a larger $\Theta(\sqrt{\log N/N})$ loss due to learning.

Now, we are ready to bound $\mathbb{E}\left[A_2(t)\right]$. First, given a sequence of arrival times Λ, by re-indexing the clients in the order of their arrival times, we have

$$\sum_{t=1}^{T}A_2(t) = V\sum_{i=1}^{I}\sum_{k=1}^{m_i(T)}\sum_{t=t_1(k)}^{t_2(k)}\sum_{j=1}^{J}\left(C_{ij}^k(t) - C_{ij}^*\right)p_{ij}^k(t),$$

where $m_i(T)$ is the total number of arrivals of class i up to time slot T. Since Lemma 5 holds for any client k and any server j, we have

$$\sum_{t=1}^{T}\mathbb{E}_\Lambda\left[A_2(t)\right] \leq V\sum_{i=1}^{I}\sum_{k=1}^{m_i(T)}\left(4J\sqrt{2a_k\log a_k} + \left(4\sqrt{2\log a_k}+3\right)\mu\right).$$

Note that a_k has mean N, and further both $\sqrt{x \log x}$ and $\sqrt{\log x}$ are concave functions. Moreover, $\mathbb{E}[m_i(T)] = \lambda_i T/N$. Taking the expectation of a_k and then Λ over both sides of the last displayed equation and applying Jensen's inequality, we get

$$\frac{1}{TV}\sum_{t=1}^{T}\mathbb{E}\left[A_2(t)\right] \leq \frac{\lambda}{N}\left(4J\sqrt{2N\log N} + \left(4\sqrt{2\log N}+3\right)\mu\right). \tag{35}$$

6.5 Bounding $A_3(t)$

We now bound the contribution to $\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[A_3(t)]$ by the k-th arriving client of class i, using similar ideas as in Section 6.4. Here, however, we face a major new difficulty related to the issue of "learning slow-down" discussed in Section 3. Note that the rate with which tasks of this client are assigned to the servers at time t is given by $\sum_{j=1}^{J} p_{ij}^k(t)$, which may decrease as there are more clients in the system. We thus refer to this value as the "learning rate," since it determines how quickly (or slowly) the system can receive payoff feedback for this client and improve her payoff estimate. However, the loss in $A_3(t)$ accumulates at a different rate $\widehat{p}_{ij}^k(t)$. We therefore refer to $\sum_{j=1}^{J}\widehat{p}_{ij}^k(t)$ as the "loss-accummulation rate." If the learning rate is small and the loss-accummulation rate is large, it is possible that the total loss may grow unboundedly because the system does not learn as quickly as it incurs losses. The following lemma thus becomes crucial. It shows that, thanks to our choice in Algorithm 1 that all the UCB estimates are no greater than 1 and $\gamma > 1$, the learning rate for any client will be at most a constant factor away from the loss-accummulation rate. We will then use this lemma to show that $\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[A_3(t)]$ must then be upper-bounded by a constant.

Lemma 6. *At any time t, suppose that $[\widehat{p}_{ij}^k(t)]$ and $[p_{ij}^k(t)]$ are the maximizers of $f(\boldsymbol{\pi}|\mathbf{n}(t), \mathbf{C}^*)$ and $f(\boldsymbol{\pi}|\mathbf{n}(t), \mathbf{C}(t))$ defined in (28), respectively, subject to the constraint (15). Then, for all clients k of all classes i, the following holds*

$$\frac{\sum_{j=1}^{J}\widehat{p}_{ij}^k(t)}{\sum_{j=1}^{J}p_{ij}^k(t)} \leq \left(\frac{\gamma}{\gamma-1}\right)^2.$$

The proof is available in Appendix A.5. The main idea is as follows. Note that for any $\mathbf{W}$, the maximizer of $f(\boldsymbol{\pi}|\mathbf{n}(t), \mathbf{W})$ subject to server capacity constraint (15) must satisfy the KKT condition:

$$\sum_{j=1}^{J}\pi_{ij}^k(t) = \frac{1}{V\min_j\{q_j(t)+\gamma-W_{ij}^k\}},$$

where $q_j(t) \geq 0$ is the optimal dual variable corresponding to server j's capacity constraint. For either $\mathbf{W} = \mathbf{C}^*$ or $\mathbf{W} = \mathbf{C}(t)$, the value of W_{ij}^k cannot differ by more than 1. Hence, the value of $q_j(t)+\gamma-W_{ij}^k$ cannot differ by more than a factor $\frac{\gamma}{\gamma-1}$. This implies that the value of $\sum_{j=1}^{J}p_{ij}^k(t)$ or $\sum_{j=1}^{J}\widehat{p}_{ij}^k(t)$ across different clients cannot differ much either. The result of the lemma can then be readily shown.

Lemma 7. *Suppose that the k-th arriving client of class i arrives at time $t_1(k)$, and let $t_2(k)$ be the minimum of T and her departure time. Then, the following holds,*

$$\mathbb{E}_\Lambda\left[\sum_{j=1}^{J}\sum_{s=t_1(k)}^{t_2(k)}\left(C_{ij}^*-C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s)\right] \leq 3J\left(\frac{\gamma}{\gamma-1}\right)^2\mu. \tag{36}$$

We give the intuition behind this lemma before the proof. Recall that $h_{ij}^k(t-1)$ is the number of tasks from this particular client that have been assigned to server j by the end of time $t-1$, and $h_i^k(t-1) =$

$\sum_{j=1}^{J} h_{ij}^k(t-1)$. Further, let $h_{ij}^k(0) = h_i^k(0) = 0$. For each time-slot s and $1 \le j \le J$, define the "good" event

$$G_j(s) = \left\{ h_{ij}^k(s-1) \ge 1, C_{ij}^* - \overline{C}_{ij}^k(s-1) \le \sqrt{\frac{2\log h_i^k(s-1)}{h_{ij}^k(s-1)}} \right\} \cup \left\{ h_{ij}^k(s-1) = 0 \right\},$$

where $\overline{C}_{ij}^k(s-1)$ is the empirical average of the received payoffs at the end of time slot $s-1$. Let $G(s) = \cap_{j=1}^{J} G_j(s)$. Note that on the event $G(s)$, the UCB estimates satisfy $C_{ij}^k(s) \ge C_{ij}^*$ and thus the terms $\left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s)$ on the left hand side of (36) are negative. Hence, to prove the lemma, it suffices to bound the sum due to the probability of the "bad" event $G^c(s)$ (i.e., the complement of $G(s)$), which accumulates at the rate of $\sum_{j=1}^{J} \widehat{p}_{ij}^k(s)$. Thanks to Lemma 6, this loss-accumulation rate is upper bounded by a constant multiplied by the learning rate $\sum_{j=1}^{J} p_{ij}^k(s)$. Finally, using concentration bounds and a delicate martingale argument, we can show that the probability of the bad event $G^c(s)$ decreases polynomially fast as the learning rate accumulates, resulting to the bound in (36).

Proof. Without loss of generality, we take $t_1(k) = 1$, i.e., we label the first time-slot as the time-slot when this particular client arrives to the system. Let $c_\gamma = \gamma^2/(\gamma-1)^2$. Let $1_{G(s)}$ denote the indicator variable which is 1 if $G(s)$ holds and 0 otherwise. Then

$$\begin{aligned}
&\sum_{s=1}^{t_2(k)} \left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s) \\
&= \sum_{s=1}^{t_2(k)} \left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s)1_{G(s)} + \sum_{s=1}^{t_2(k)} \left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s)1_{G^c(s)} \\
&\le \sum_{s=1}^{t_2(k)} \widehat{p}_{ij}^k(s)1_{G^c(s)},
\end{aligned}$$

where the last inequality holds because $C_{ij}^* \le 1$ and $C_{ij}^* \le C_{ij}^k(s)$ on the event $G(s)$. Summing over all server j, we obtain

$$\begin{aligned}
\sum_{j=1}^{J}\sum_{s=1}^{t_2(k)} \left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s) &\le \sum_{j=1}^{J}\sum_{s=1}^{t_2(k)} \widehat{p}_{ij}^k(s)1_{G^c(s)} \\
&\le c_\gamma \sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} p_{ij}^k(s),
\end{aligned}$$

where the last inequality holds because $1_{G^c(s)}$ does not depend on the index j, and in view of Lemma 6. Taking the conditional expectation given Λ (i.e., the realization of arriving times) over both sides of the last displayed equation, we obtain

$$\mathbb{E}_\Lambda\left[\sum_{j=1}^{J}\sum_{s=1}^{t_2(k)} \left(C_{ij}^* - C_{ij}^k(s)\right)\widehat{p}_{ij}^k(s)\right] \le c_\gamma \mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} p_{ij}^k(s)\right]. \tag{37}$$

Similar to $Y_l^j(t)$ in Section 4, denote $Y_{ij}^k(t)$ as the actual number of tasks from client k of class i served by server j at time slot t. We now show that

$$\mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} p_{ij}^k(s)\right] = \mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} Y_{ij}^k(s)\right]. \tag{38}$$

To see this, let $\mathcal{F}_t$ denote the filtration (which contains all the system information) up to the end of time-slot t. In particular, $Y_{ij}^k(s)$ and $1_{G^c(s+1)}$ are measurable with respect to $\mathcal{F}_t$ for all $s \le t$. Let $M_t \triangleq \sum_{s=1}^{t} 1_{G^c(s)} \sum_{j=1}^{J} \left(Y_{ij}^k(s) - p_{ij}^k(s)\right)$. Since $\mathbb{E}_\Lambda[Y_{ij}^k(s+1) - p_{ij}^k(s+1)|\mathcal{F}_s] = 0$, it follows that M_t is martingale. Further, note that $t_2(k)$ is the minimum between T and the first time that $h_i^k(t)$ exceeds a_k. Hence, $t_2(k)$ is a stopping time with respect to the filtration $\mathcal{F}_t$ and is upper bounded by T. Invoking the Optional Stopping Theorem [Haj15, Section 10.4], we then have $\mathbb{E}[M_{t_2(k)}] = 0$, which is precisely (38). In view of (37) and (38), to prove the lemma it suffices to show that

$$\mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} Y_{ij}^k(s)\right] \le 3J\mu. \tag{39}$$

Towards this end, for each integer $a = 0, 1, ..., a_k$, define

$$\tau_a = \min\left\{T+1, \inf\left\{s \ge 1 \mid h_i^k(s-1) \ge a\right\}\right\}.$$

In other words, τ_a is the first time-slot s such that the number of client k's tasks already assigned to servers at the end of the previous time-slot $s-1$ is at least a. Note that $1 = \tau_0 \le \tau_1 \le \cdots \le \tau_{a_k} = t_2(k) + 1$. Then, we have

$$\begin{aligned}
\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} Y_{ij}^k(s) &= \sum_{a\in[0,a_k-1]:\tau_{a+1}>\tau_a} \sum_{s=\tau_a}^{\tau_{a+1}-1} 1_{G^c(s)} \sum_{j=1}^{J} Y_{ij}^k(s) \\
&= \sum_{a\in[0,a_k-1]:\tau_{a+1}>\tau_a} 1_{G^c(\tau_{a+1}-1)} \sum_{j=1}^{J} Y_{ij}^k(\tau_{a+1}-1) \\
&\le \mu + \mu \sum_{a\in[1,a_k-1]:\tau_{a+1}>\tau_a} 1_{G^c(\tau_{a+1}-1)},
\end{aligned} \tag{40}$$

where the second equality holds because across all $s \in [\tau_a, \tau_{a+1}-1]$, the value of $\sum_{j=1}^{J} Y_{ij}^k(s)$ can be non-zero only at the last time-slot $\tau_{a+1}-1$ (otherwise, $h_i^k(s)$ will be at least $a+1$ before $\tau_{a+1}-1$, contradicting the definition of τ_{a+1}); and the last inequality holds because $\sum_{j=1}^{J} Y_{ij}^k(s) \le \mu$ and it can also be non-zero at most once across all $s \in [\tau_0, \tau_1 - 1]$.

Now, fix an integer a such that $\tau_{a+1} > \tau_a$. It follows that the number $h_i^k(\tau_a - 1)$ of tasks assigned by the end of time-slot $\tau_a - 1$ must be exactly equal to a. Note that $\tau_a - 1 \le \tau_{a+1} - 2 < \tau_{a+1} - 1$. Hence, we must also have $h_i^k(\tau_{a+1} - 2) = a$. By the union bound,

$$\mathbb{P}_\Lambda\left\{G^c(\tau_{a+1}-1)\right\} \le \sum_{j=1}^{J} \mathbb{P}_\Lambda\left\{G_j^c(\tau_{a+1}-1)\right\}. \tag{41}$$

Fix a $1 \le j \le J$. When $h_{ij}^k(\tau_{a+1}-2) \ge 1$, the empirical average $\overline{C}_{ij}^k(\tau_{a+1}-2)$ is the average of $h_{ij}^k(\tau_{a+1}-2)$ i.i.d. Bernoulli random variables $X(1), X(2), \ldots$ with mean C_{ij}^*. Although $h_{ij}^k(\tau_{a+1}-2)$ is

random, it holds that $h_{ij}^k(\tau_{a+1}-2) \le h_i^k(\tau_{a+1}-2) = a$. Therefore,

$$\begin{aligned}
&\mathbb{P}_\Lambda\left\{G_j^c(\tau_{a+1}-1)\right\} \\
&\le \mathbb{P}_\Lambda\left\{\forall 1 \le r \le a : \frac{1}{r}\sum_{u=1}^{r} X(u) - C_{ij}^* < -\sqrt{\frac{2\log a}{r}}\right\} \\
&\le \sum_{r=1}^{a} \mathbb{P}\left\{\frac{1}{r}\sum_{u=1}^{r} X(u) - C_{ij}^* < -\sqrt{\frac{2\log a}{r}}\right\} \le \sum_{r=1}^{a}\frac{1}{a^4} \le \frac{1}{a^3},
\end{aligned} \tag{42}$$

where the second inequality is due to the Chernoff-Hoeffding Bound (see Lemma 9 in Appendix A.3). Combining (40)-(42) yields

$$\mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} 1_{G^c(s)} \sum_{j=1}^{J} Y_{ij}^k(s)\right] \le \mu + \mu J \sum_{a=1}^{\infty}\frac{1}{a^3} \le \mu + 3\mu J/2 \le 3\mu J,$$

which is exactly (39). The result of the lemma then follows.

□

Using Lemma 7 and following the same steps at the end of Section 6.4, we then get

$$\frac{1}{TV}\sum_{t=1}^{T}\mathbb{E}[A_3(t)] \le \frac{3\lambda\mu J}{N}\left(\frac{\gamma}{1-\gamma}\right)^2. \tag{43}$$

Finally, the desired (27) follows by combining (33), (35), and (43).

7 Conclusion

In this paper, we propose a new utility-guided task assignment algorithm for queueing systems with uncertain payoffs. Our algorithm seamlessly integrates online learning and adaptive control, without relying on any prior knowledge of the statistics of the client dynamics or the payoff vectors. We show that, compared to an oracle that knows all client dynamics and payoff vectors beforehand, the gap of the expected payoff per unit time of our proposed algorithm at any finite time T is on the order of $1/V + \sqrt{\log N/N} + N(V+1)/T$, where V is a parameter controlling the weight of a logarithmic utility function, and N is the average number of tasks per client. Our analysis carefully accounts for the closed-loop interactions between the queueing and learning processes. Numerical results indicate that the proposed algorithm outperforms a myopic matching policy and a standard queue-length based policy that does not explicitly address such closed-loop interactions.

There are a number of future research directions. First, a straightforward lower bound on the payoff loss is $O(\log N/N)$ [ACBF02]. Thus, it would be interesting to study whether the proposed algorithm can be further improved to reduce the second regret term of (11) to $O(\log N/N)$. Second, in order to limit the payoff-feedback delay, our algorithm assumes that the service of each server is deterministic. We plan to generalize to the case with random service or even unknown service rates. Finally, although our model does not assume any prior knowledge of the class-dependent payoff vectors, it would be interesting to study whether the idea from this paper can be used to improve the policies in [JKK16, MX16, JKK17] when such knowledge is available.

References

[ACBF02] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[ACBFS02] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

[AdW] Adwords. https://adwords.google.com/. Accessed: 2017-08-30.

[AHL13] Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. The online stochastic generalized assignment problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 11–25. Springer, 2013.

[Air] Airbnb. https://www.airbnb.com/. Accessed: 2017-08-30.

[Alf16] Attahiru S Alfa. *Applied discrete-time queues*. Springer, 2016.

[AMTS11] Animashree Anandkumar, Nithin Michael, Ao Kevin Tang, and Ananthram Swami. Distributed algorithms for learning and cognitive medium access with logarithmic regret. *IEEE Journal on Selected Areas in Communications*, 29(4):731–745, 2011.

[BCB+12] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[BLS14] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful contextual bandits. In *Conference on Learning Theory*, pages 1109–1134, 2014.

[BM01] T. Bonald and L. Massoulie. Impact of Fairness on Internet Performance. In *Proceedings of ACM SIGMETRICS*, pages 82–91, Cambridge, MA, June 2001.

[BM15] Kostas Bimpikis and Mihalis G Markakis. Learning and hierarchies in service systems. *under review*, 2015.

[BTZ15] Swapna Buccapatnam, Jian Tan, and Li Zhang. Information sharing in distributed stochastic bandits. In *IEEE INFOCOM*, pages 2605–2613, 2015.

[Chi04] M. Chiang. To Layer or Not to Layer: Balancing Transport and Physical Layers in Wireless Multihop Networks. In *IEEE INFOCOM*, Hong Kong, March 2004.

[CJS15] Richard Combes, Chong Jiang, and Rayadurgam Srikant. Bandits with budgets: Regret lower bounds and optimal algorithms. *ACM SIGMETRICS Performance Evaluation Review*, 43(1):245–257, 2015.

[Cou] Coursera. https://www.coursera.org/. Accessed: 2017-08-30.

[CP15] Richard Combes and Alexandre Proutiere. Dynamic rate and channel selection in cognitive radio systems. *IEEE Journal on Selected Areas in Communications*, 33(5):910–921, 2015.

[CZL08] Deeparnab Chakrabarty, Yunhong Zhou, and Rajan Lukose. Online knapsack problems. In *Workshop on Internet and Network Economics (WINE)*, 2008.

[DLK01] G. De Veciana, T. J. Lee, and T. Konstantopoulos. Stability and Performance Analysis of Networks Supporting Elastic Services. *IEEE/ACM Trans. on Networking*, 9(1):2–14, February 2001.

[ES05] A. Eryilmaz and R. Srikant. Fair Resource Allocation in Wireless Networks Using Queue-length-based Scheduling and Congestion Control. In *IEEE INFOCOM*, Miami, FL, March 2005.

[FKM+09] Jon Feldman, Nitish Korula, Vahab S Mirrokni, S Muthukrishnan, and Martin Pál. Online ad assignment with free disposal. In *WINE*, volume 9, pages 374–385. Springer, 2009.

[GGW11] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-armed bandit allocation indices*. John Wiley & Sons, 2011.

[Git79] John C Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979.

[GKJ12] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking (TON)*, 20(5):1466–1478, 2012.

[GOW04] Ayalvadi J Ganesh, Neil O'Connell, and Damon J Wischik. *Big queues*. Springer, 2004.

[Haj15] Bruce Hajek. *Random Processes for Engineers*. Cambridge University Press, March 2015.

[HLH14] Longbo Huang, Xin Liu, and Xiaohong Hao. The power of online learning in stochastic network optimization. In *ACM SIGMETRICS Performance Evaluation Review*, volume 42, pages 153–165. ACM, 2014.

[HXLBml] Wei-Kang Hsu, Jiaming Xu, Xiaojun Lin, and Mark R. Bell. Integrate Learning and Control in Queueing Systems with Uncertain Payoff. Technical report, Purdue University, 2017, available at https://engineering.purdue.edu/%7elinx/papers.html.

[HZCL16] Longbo Huang, Shaoquan Zhang, Minghua Chen, and Xin Liu. When backpressure meets predictive scheduling. *IEEE/ACM Transactions on Networking*, 24(4):2237–2250, 2016.

[Ipe10] Panagiotis G. Ipeirotis. Analyzing the amazon mechanical turk marketplace. *XRDS*, 17(2):16–21, December 2010.

[JKK16] Ramesh Johari, Vijay Kamble, and Yash Kanoria. Know your customer: Multi-armed bandits with capacity constraints. *arXiv preprint arXiv:1603.04549v1*, 2016.

[JKK17] Ramesh Johari, Vijay Kamble, and Yash Kanoria. Matching while learning. In *Proceedings of the 2017 ACM Conference on Economics and Computation (EC '17)*, Cambridge, MA, June 2017.

[JNN14] Shweta Jain, Balakrishnan Narayanaswamy, and Y Narahari. A multiarmed bandit incentive mechanism for crowdsourcing demand response in smart grids. In *AAAI*, pages 721–727, 2014.

[JS13] Chong Jiang and R Srikant. Bandits with budgets. In *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*, pages 5345–5350. IEEE, 2013.

[KM95] P. R. Kumar and S. P. Meyn. Stability of Queueing Networks and Scheduling Policies. *IEEE Transactions on Automatic Control*, 40:251–260, February 1995.

[KNJ14] Dileep Kalathil, Naumaan Nayyar, and Rahul Jain. Decentralized learning for multiplayer multiarmed bandits. *IEEE Transactions on Information Theory*, 60(4):2331–2345, 2014.

[KO16] Ashish Khetan and Sewoong Oh. Achieving budget-optimality with adaptive schemes in crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 4844–4852, 2016.

[KOS11a] David R Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal crowdsourcing using low-rank matrix approximations. In *49th Annual Allerton Conference on Communication, Control, and Computing*, pages 284–291, 2011.

[KOS11b] David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in Neural Information Processing Systems*, pages 1953–1961, 2011.

[KOS13] David R Karger, Sewoong Oh, and Devavrat Shah. Efficient crowdsourcing for multi-class labeling. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):81–92, 2013.

[LCS15] Tor Lattimore, Koby Crammer, and Csaba Szepesvári. Linear multi-resource allocation with semi-bandit feedback. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, NIPS'15, pages 964–972, Cambridge, MA, USA, 2015.

[LEGJP11] Lifeng Lai, Hesham El Gamal, Hai Jiang, and H Vincent Poor. Cognitive medium access: Exploration, exploitation, and competition. *IEEE Transactions on Mobile Computing*, 10(2):239–253, 2011.

[LPT13] Marc Lelarge, Alexandre Proutiere, and M Sadegh Talebi. Spectrum bandit optimization. In *IEEE Information Theory Workshop (ITW)*, pages 1–5, 2013.

[LR85] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[LS05] X. Lin and N. B. Shroff. The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Multihop Wireless Networks. In *IEEE INFOCOM*, Miami, FL, March 2005.

[LSS06] X. Lin, N. B. Shroff, and R. Srikant. A Tutorial on Cross-Layer Optimization in Wireless Networks. *IEEE Journal on Selected Areas in Communications (JSAC)*, 24(8):1452–1463, 2006.

[LSS08] X. Lin, N. B. Shroff, and R. Srikant. On the Connection-Level Stability of Congestion-Controlled Communication Networks. *IEEE Trans. on Inf. Theory*, 54(5):2317–2338, May 2008.

[LY88] Tze Leung Lai and Zhiliang Ying. Open bandit processes and optimal scheduling of queueing networks. *Advances in Applied Probability*, 20(2):447–472, 1988.

[LZ08] John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems*, pages 817–824, 2008.

[MX16] Laurent Massoulie and Kuang Xu. On the capacity of information processing systems. In *Conference on Learning Theory*, pages 1292–1297, 2016.

[Neu81] Marcel F Neuts. *Matrix-geometric solutions in stochastic models: an algorithmic approach*. Courier Corporation, 1981.

[NKJ16] Naumaan Nayyar, Dileep Kalathil, and Rahul Jain. On regret-optimal learning in decentralized multi-player multi-armed bandits. *IEEE Transactions on Control of Network Systems*, 2016.

[NML05] M. J. Neely, E. Modiano, and C. Li. Fairness and Optimal Stochastic Control for Heterogeneous Networks. In *IEEE INFOCOM*, Miami, FL, March 2005.

[PO07] Sandeep Pandey and Christopher Olston. Handling advertisements of unknown quality in search advertising. In *Advances in Neural Information Processing Systems*, pages 1065–1072, 2007.

[SGMV17] Virag Shah, Lennart Gulikers, Laurent Massoulie, and Milan Vojnovic. Adaptive matching for expert systems with uncertain task types. *arXiv preprint arXiv:1703.00674*, 2017.

[SRJ17] Shahin Shahrampour, Alexander Rakhlin, and Ali Jadbabaie. Multi-armed bandits in multi-agent networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2786–2790. IEEE, 2017.

[TE92] L. Tassiulas and A. Ephremides. Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks. *IEEE Trans. on Automatic Control*, 37(12):1936–1948, December 1992.

[TS12] Bo Tan and R Srikant. Online advertisement, optimization and stochastic networks. *IEEE Transactions on Automatic Control*, 57(11):2854–2868, 2012.

[TTSRJ14] Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artificial Intelligence*, 214:89–111, 2014.

[Upw] Upwork. https://www.upwork.com/. Accessed: 2017-08-30.

[Whi81] Peter Whittle. Arm-acquiring bandits. *The Annals of Probability*, 9(2):284–292, 1981.

[Whi88] Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, 25(A):287–298, 1988.

[WZ13] Xianyi Wu and Xian Zhou. Open bandit processes with uncountable states and time-backward effects. *Journal of Applied Probability*, 50(2):388–402, 2013.

[ZCXL13] Feifeng Zheng, Yongxi Cheng, Yinfeng Xu, and Ming Liu. Competitive strategies for an online generalized assignment problem with a service consecution constraint. *European Journal of Operational Research*, 229(1):59–66, 2013.

A Appendix

A.1 Bounds on the Expected Number of Client Departures

Lemma 8. *Given* $\mathbf{n}(t) = [n_i(t)]$, *let* $\mathbf{p}(t) = [p_{ij}^k(t)]$ *denote the vector of assignment probabilities computed in Step 2 of the proposed Algorithm 1 at time-slot* t. *Fix a class* i *with* $n_i(t) \geq 1$. *Let* $D_i(t)$ *denote the number of client departures of class* i *at time-slot* t. *Conditioned on* $\mathbf{n}(t)$ *and* $\mathbf{p}(t)$, *the expectation of* $D_i(t)$ *satisfies the following bounds:*

$$\mathbb{E}\left[D_i(t)|\mathbf{n}(t),\mathbf{p}(t)\right] \geq \frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) - \frac{\gamma^2\mu^2}{2(\gamma-1)^2 n_i(t) N^2}, \tag{44}$$

$$\mathbb{E}\left[D_i(t)|\mathbf{n}(t),\mathbf{p}(t)\right] \leq \frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) \leq \frac{\mu}{N}, \tag{45}$$

and

$$\mathbb{E}\left[D_i^2(t) \mid \mathbf{n}(t),\mathbf{p}(t)\right] \leq \frac{\mu^2}{N}, \tag{46}$$

where $\mu = \sum_{j=1}^{J}\mu_j$.

The intuition behind this lemma is as follows. Note that in Step 3 of the proposed Algorithm 1, each unit service of server j has $\sum_{k=1}^{n_i(t)} p_{ij}^k(t)/\mu_j$ probability to pick a client from class i. Hence, if all units of the μ_j service rate of all servers j were fully used, the expected number of departures of class i would have been $\frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t)$, which appears in both (44) and (45). However, note that if a client of class i already runs out of tasks after she is picked by one server, there will be no additional departure when a subsequent server picks this client again. The additional term $\frac{\gamma^2\mu^2}{2(\gamma-1)^2 n_i(t)N^2}$ in the lower bound (44) accounts for such descrepancy. We also note that this additional term decreases with N and $n_i(t)$, which is true because the larger the values of N and $n_i(t)$, the smaller the departure probability $\sum_{j=1}^{J} p_{ij}^k(t)/(N\mu_j)$ of a particular client, and hence the smaller the discrepancy.

Proof. Denote $E_i^k(t)$ as the event that client k of class i remains in the system at the end of time t, $k = 1, ..., n_i(t)$. Recall that in Step 3 of our proposed Algorithm 1, each unit service of server j has $p_{ij}^k(t)/\mu_j$ probability of picking client k of class i, independently of other units of service of the same or other servers. Thus, if this client has not run out of tasks yet, it has $p_{ij}^k(t)/(N\mu_j)$ probability to leave the system. We then have

$$\mathbb{P}[E_i^k(t)|\mathbf{n}(t),\mathbf{p}(t)] = \prod_{j=1}^{J}\left(1 - \frac{p_{ij}^k(t)}{N\mu_j}\right)^{\mu_j},$$

and

$$\mathbb{E}[D_i(t)|\mathbf{n}(t),\mathbf{p}(t)] = \sum_{k=1}^{n_i(t)}\left(1 - \mathbb{P}[E_i^k(t)]\right). \tag{47}$$

Next, we derive the desired lower and upper bounds. In order to derive a lower bound, we use the inequality $1 - x \leq e^{-x} \leq 1 - x + x^2/2$ for $x \geq 0$. We have

$$\begin{aligned} 1 - \mathbb{P}[E_i^k(t)|\mathbf{n}(t),\mathbf{p}(t)] &\geq 1 - \exp\left(-\sum_{j=1}^{J}\frac{p_{ij}^k(t)}{N}\right) \\ &\geq \frac{1}{N}\sum_{j=1}^{J} p_{ij}^k(t) - \frac{1}{2N^2}\left(\sum_{j=1}^{J} p_{ij}^k(t)\right)^2. \end{aligned}$$

Hence,

$$\mathbb{E}[D_i(t)|\mathbf{n}(t),\mathbf{p}(t)] \geq \frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) - \frac{1}{2N^2}\sum_{k=1}^{n_i(t)}\left(\sum_{j=1}^{J} p_{ij}^k(t)\right)^2. \tag{48}$$

To bound the second term of (48), we now show that, for all $k = 1, ..., n_i(t)$,

$$\sum_{j=1}^{J} p_{ij}^k(t) \leq \frac{\gamma\mu}{(\gamma-1)n(t)} \leq \frac{\gamma\mu}{(\gamma-1)n_i(t)}. \tag{49}$$

To see this, recall that $\mathbf{p}(t)$ solves the maximization problem (14) subject to the capacity constraint (15). Let $q_j(t) \geq 0$ be the optimal dual variable corresponding to server j's capacity constraint in (15). Then, consider any two different clients: client k of class i and client k' of class i', $(i', k') \neq (i, k)$. We have

$$q_j(t) + \gamma - C_{ij}^k(t) \geq q_j(t) + \gamma - 1 \geq \frac{\gamma-1}{\gamma}\left(q_j(t) + \gamma - C_{i'j}^{k'}\right),$$

where we have used $C_{ij}^k(t) \leq 1$ and $C_{i'j}^{k'} \geq 0$. Hence, by the KKT condition, we have,

$$\begin{aligned}\sum_{j=1}^{J} p_{ij}^k(t) &= \frac{1}{V \min_j\{q_j(t) + \gamma - C_{ij}^k(t)\}} \\ &\leq \frac{\gamma/(\gamma-1)}{V \min_j\{q_j(t) + \gamma - C_{i'j}^{k'}\}} = \frac{\gamma}{\gamma-1}\sum_{j=1}^{J} p_{i'j}^{k'}(t).\end{aligned}$$

By the capacity constraint (15), we have

$$\sum_{i\in\mathcal{I}(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t) \leq \mu_j.$$

Thus, summing over all j, we have

$$\begin{aligned}\mu = \sum_{j=1}^{J}\mu_j &\geq \sum_{i\in\mathcal{I}(t)}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) \\ &\geq \sum_{j=1}^{J} p_{ij}^k(t) + \sum_{\substack{i'\in\mathcal{I}(t),k'=1,...,n_{i'}(t)\\(i',k')\neq(i,k)}}\sum_{j=1}^{J} p_{i'j}^{k'}(t) \\ &\geq \frac{(\gamma-1)n(t)}{\gamma}\sum_{j=1}^{J} p_{ij}^k(t).\end{aligned}$$

The inequality (49) then follows. Substituting (49) into (48), we then have,

$$\mathbb{E}[D_i(t)|\mathbf{n}(t), \mathbf{p}(t)] \geq \frac{1}{N}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) - \frac{\gamma^2\mu^2}{2(\gamma-1)^2 n_i(t) N^2}.$$

This proves (44).

On the other hand, in order to obtain an upper bound on (47), we iteratively apply the relationship $(1-x)(1-y) \geq 1-(x+y)$ for all $x, y \geq 0$. We then obtain

$$\mathbb{P}[E_i^k(t)|\mathbf{n}(t), \mathbf{p}(t)] \geq 1 - \sum_{j=1}^{J}\mu_j\frac{p_{ij}^k}{N\mu_j} = 1 - \frac{1}{N}\sum_{j=1}^{J} p_{ij}^k.$$

Substituting into (47), the inequality (45) then follows.

Finally, to show (46), we use $D_i(t) \leq \sum_{j=1}^{J} \mu_j = \mu$. Combining with (45), we then have

$$\mathbb{E}\left[D_i^2(t) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \leq \mathbb{E}\left[D_i(t) \mid \mathbf{n}(t), \mathbf{p}(t)\right] \mu \leq \frac{\mu^2}{N}.$$

□

A.2 Proof of Lemma 4

Proof. For any given $\mathbf{n}$ and $\mathbf{W} = [W_{ij}^k]$, the gradient of $f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W})$ (defined in (28)) with respect to $\boldsymbol{\pi}$ is given by $\nabla f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W}) = \left[\frac{\partial f}{\partial \pi_{ij}^k}\right]$ with

$$\frac{\partial f}{\partial \pi_{ij}^k} = \frac{1}{\sum_{j=1}^{J} \pi_{ij}^k} + V(W_{ij}^k - \gamma). \tag{50}$$

Since the function f is concave in π_{ij}^k, we thus have, for any $\widetilde{\mathbf{p}}$,

$$f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W}) - f(\widetilde{\mathbf{p}}|\mathbf{n}, \mathbf{W}) \leq [\nabla f(\widetilde{\mathbf{p}}|\mathbf{n}, \mathbf{W})]'(\boldsymbol{\pi} - \widetilde{\mathbf{p}}). \tag{51}$$

Using the value of $\widetilde{\mathbf{p}}(t)$ given in the lemma, the partial derivative of f at $\widetilde{\mathbf{p}}(t)$ is equal to

$$\left.\frac{\partial f}{\partial \pi_{ij}^k}\right|_{\pi_{ij}^k = \frac{\lambda_i p_{ij}^*}{n_i(t)}} = \frac{n_i(t)}{\lambda_i} + V(W_{ij}^k - \gamma). \tag{52}$$

Therefore, for any $\boldsymbol{\pi}$, we have,

$$\begin{aligned} &f(\boldsymbol{\pi}|\mathbf{n}, \mathbf{W}) - f(\widetilde{\mathbf{p}}(t)|\mathbf{n}, \mathbf{W}) \\ &\leq \sum_{i \in \mathcal{I}(t)} \sum_{k=1}^{n_i(t)} \sum_{j=1}^{J} \left(\frac{n_i(t)}{\lambda_i} + V(W_{ij}^k - \gamma)\right) \left(\pi_{ij}^k - \frac{\lambda_i p_{ij}^*}{n_i(t)}\right). \end{aligned}$$

The result of the lemma then follows. □

A.3 Chernoff-Hoeffding Bound

We cite the following lemma from Fact 1 in [ACBF02], which is simply the Chernoff-Hoeffding bound.

Lemma 9. *Let $X(s), s = 1, 2, ...$ be a sequence of* i.i.d. *Bernoulli random variables with mean C_{ij}^*. For any given positive integer a and any real number $t \geq 1$, the followings hold:*

$$\mathbb{P}\left\{\frac{\sum_{s=1}^{a} X(s)}{a} - C_{ij}^* \geq \sqrt{\frac{2\log t}{a}}\right\} \leq \frac{1}{t^4},$$

and

$$\mathbb{P}\left\{\frac{\sum_{s=1}^{a} X(s)}{a} - C_{ij}^* \leq -\sqrt{\frac{2\log t}{a}}\right\} \leq \frac{1}{t^4}.$$

A.4 Proof of Lemma 5

Fix k, i and j. Without loss of generality, we can take $t_1(k) = 1$, i.e., we label the first time-slot as the time-slot when this particular client arrives to the system. Recall that $h_{ij}^k(t)$ is the number of tasks from this particular client that have been assigned to server j by the end of time t, and $h_i^k(t) = \sum_{j=1}^J h_{ij}^k(t)$. For $t = 0$, we take $h_{ij}^k(0) = h_i^k(0) = 0$. In the definition of the event $F_j(t)$ below, in order to avoid the difficulty of division by zero, we further define $\widehat{h}_{ij}^k(t) = \max\{h_{ij}^k(t), 1\}$ and $\widehat{h}_i^k(t) = \max\{h_i^k(t), 1\}$. Then, define the event

$$F_j(t) = \left\{ \overline{C}_{ij}^k(t-1) - C_{ij}^* \le \sqrt{\frac{2\log \widehat{h}_i^k(t-1)}{\widehat{h}_{ij}^k(t-1)}} \right\},$$

where $\overline{C}_{ij}^k(t-1)$ is the empirical average of the received payoffs at the end of time slot $t-1$. We let $Q_{ij}^k(t) = C_{ij}^k(t) - C_{ij}^* \le 1$. Then

$$\sum_{s=1}^{t_2(k)} Q_{ij}^k(s) p_{ij}^k(s) \le \sum_{s=1}^{t_2(k)} Q_{ij}^k(s) p_{ij}^k(s) 1_{F_j(s)} + \sum_{s=1}^{t_2(k)} p_{ij}^k(s) 1_{F_j^c(s)}. \tag{53}$$

We first bound the expectation of the first term in (53). By the definition of the event $F_j(t)$ and the definition of the UCB estimate $C_{ij}^k(t)$,

$$\begin{aligned}
&\sum_{s=1}^{t_2(k)} Q_{ij}^k(s) p_{ij}^k(s) 1_{F_j(s)} \le \sum_{s=1}^{t_2(k)} 2\sqrt{\frac{2\log \widehat{h}_i^k(s-1)}{\widehat{h}_{ij}^k(s-1)}} p_{ij}^k(s) \\
&\le 2\sqrt{2\log a_k} \sum_{s=1}^{t_2(k)} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} p_{ij}^k(s).
\end{aligned} \tag{54}$$

Recall that $Y_{ij}^k(t)$ is the actual number of tasks served by server j at time slot t. We now show that

$$\mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} Y_{ij}^k(s) \right] = \mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} p_{ij}^k(s) \right]. \tag{55}$$

To see this, let $\mathcal{F}_t$ denote the filtration (which contains all the system information) up to the end of time-slot t. In particular, $Y_{ij}^k(s)$ and $p_{ij}^k(s)$ are measurable with respect to $\mathcal{F}_t$ for all $s \le t$. Let

$$M_t \triangleq \sum_{s=t_1(k)}^{t} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} (Y_{ij}^k(s) - p_{ij}^k(s)).$$

Since $\mathbb{E}_\Lambda[Y_{ij}^k(s+1)|p_{ij}^k(s+1)] = p_{ij}^k(s+1)$, we have $\mathbb{E}_\Lambda[Y_{ij}^k(s+1) - p_{ij}^k(s+1)|\mathcal{F}_s] = 0$. It then follows that M_t is martingale. Further, note that $t_2(k)$ is the minimum between T and the first time that $h_i^k(t)$ exceeds a_k. Hence, $t_2(k)$ is a stopping time with respect to the filtration $\mathcal{F}_t$ and is upper bounded by T. Invoking the Optional Stopping Theorem [Haj15, Section 10.4], we then have $\mathbb{E}[M_{t_2(k)}] = 0$, which is precisely (55).

By definition, $Y_{ij}^k(t) = h_{ij}^k(t) - h_{ij}^k(t-1)$. It follows that

$$\begin{aligned}\sum_{s=1}^{t_2(k)} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} Y_{ij}^k(s) &= \sum_{s=1}^{t_2(k)} \sqrt{\frac{1}{\widehat{h}_{ij}^k(s-1)}} \left(h_{ij}^k(s) - h_{ij}^k(s-1)\right) \\ &\leq 2\mu_j + \int_1^{a_k - 1} \frac{1}{\sqrt{x}} \mathrm{d}x \\ &\leq 2\mu_j + 2\sqrt{a_k},\end{aligned}$$

where the second-to-last step follows because whenever $h_{ij}^k(t) = 0$, we must have $h_{ij}^k(t) - h_{ij}^k(t-1) = 0$ and we always have $h_{ij}^k(t) - h_{ij}^k(t-1) \leq \mu_j$. Combining the last displayed equation with (54) and (55), we have

$$\mathbb{E}_\Lambda \left[\sum_{s=1}^{t_2(k)} Q_{ij}^k(s) p_{ij}^k(s) 1_{F_j(s)} \right] \leq 4\sqrt{2a_k \log a_k} + 4\mu_j \sqrt{2 \log a_k}. \tag{56}$$

To bound the expected value of the second term in (53), we first note that

$$\mathbb{E}_\Lambda \left[\sum_{s=1}^{t_2(k)} Y_{ij}^k(s) 1_{F_j^c(s)} \right] = \mathbb{E}_\Lambda \left[\sum_{s=1}^{t_2(k)} p_{ij}^k(s) 1_{F_j^c(s)} \right], \tag{57}$$

which can be derived similarly to (55). For each integer $0 \leq a \leq a_k$, define

$$\tau_a = \min \left\{ T+1, \inf \left\{ s \geq 1 \mid h_{ij}^k(s-1) \geq a \right\} \right\}.$$

Note that $1 = \tau_0 \leq \tau_1 \leq \cdots \leq \tau_{a_k} = t_2(k) + 1$. Then,

$$\begin{aligned}\sum_{s=1}^{t_2(k)} Y_{ij}^k(s) 1_{F_j^c(s)} &= \sum_{a \in [0, a_k - 1] : \tau_{a+1} > \tau_a} \sum_{s=\tau_a}^{\tau_{a+1}-1} Y_{ij}^k(s) 1_{F_j^c(s)} \\ &= \sum_{a \in [0, a_k - 1] : \tau_{a+1} > \tau_a} Y_{ij}^k(\tau_{a+1} - 1) 1_{F_j^c(\tau_{a+1}-1)} \\ &\leq \mu_j + \mu_j \sum_{a \in [1, a_k - 1] : \tau_{a+1} > \tau_a} 1_{F_j^c(\tau_{a+1}-1)},\end{aligned} \tag{58}$$

where the second equality holds because across all $s \in [\tau_a, \tau_{a+1} - 1]$, the value of $Y_{ij}^k(s)$ can be non-zero only at $\tau_{a+1} - 1$ (otherwise, $h_{ij}^k(s)$ will be at least $a+1$ before $\tau_{a+1} - 1$, contradicting the definition of τ_{a+1}). Combining (57) and (58) yields that

$$\begin{aligned}&\mathbb{E}_\Lambda \left[\sum_{s=1}^{t_2(k)} p_{ij}^k(s) 1_{F_j^c(s)} \right] \\ &\leq \mu_j + \mu_j \sum_{a \in [1, a_k - 1] : \tau_{a+1} > \tau_a} \mathbb{P}_\Lambda \left\{ F_j^c(\tau_{a+1} - 1) \right\}.\end{aligned} \tag{59}$$

Fix an integer a such that $\tau_{a+1} > \tau_a$. It follows that the number $h_{ij}^k(\tau_a - 1)$ of tasks assigned to server j by the end of time-slot $\tau_a - 1$ must be exactly equal to a. Note that $\tau_a - 1 \leq \tau_{a+1} - 2 < \tau_{a+1} - 1$. Hence, we

must also have $h_{ij}^k(\tau_{a+1}-2)=a$ and thus the empirical average $\overline{C}_{ij}^k(\tau_{a+1}-2)$ is the average of exactly a i.i.d. Bernoulli random variables $X(1),\ldots,X(a)$ with mean C_{ij}^*. Moreover, $a \le h_i^k(\tau_{a+1}-2) \le a_k$. Therefore,

$$F_j^c(\tau_{a+1}-1) \subset \cup_{\eta=a}^{a_k}\left\{\frac{1}{a}\sum_{u=1}^{a}X(u)-C_{ij}^* > \sqrt{\frac{2\log\eta}{a}}\right\}.$$

By the union bound,

$$\begin{aligned}\mathbb{P}_\Lambda\left\{F_j^c(\tau_{a+1}-1)\right\} &\le \sum_{\eta=a}^{a_k}\mathbb{P}\left\{\frac{\sum_{u=1}^{a}X(u)}{a}-C_{ij}^* > \sqrt{\frac{2\log\eta}{a}}\right\}\\ &\le \sum_{\eta=a}^{a_k}\frac{1}{\eta^4} \le \frac{1}{a^4}+\int_a^\infty \frac{1}{\eta^4}d\eta \le \frac{4}{3a^3},\end{aligned}$$

where the second inequality is due to Lemma 9. Substituting it into (59), we then have

$$\mathbb{E}_\Lambda\left[\sum_{s=1}^{t_2(k)}p_{ij}^k(s)1_{F_j^c(s)}\right] \le \mu_j+\mu_j\sum_{a=1}^{\infty}\frac{4}{3a^3}=3\mu_j. \tag{60}$$

The final result follows by combining (56) and (60) with (53).

A.5 Proof of Lemma 6

Fix t and client k of class i. Let $\mathbf{n}=\mathbf{n}(t)$ and recall that $C_{ij}^k(t)\le 1$. First, consider $[\widehat{p}_{ij}^k(t)]$, which is the maximizer of $f(\mathbf{p}|\mathbf{n},\mathbf{C}^*))$ over the constraint (15). Let $q_j(t)\ge 0$ be the optimal dual variable corresponding to server j's capacity constraint in (15). By the KKT condition, for any client k of any class i, we have

$$\sum_{j=1}^{J}\widehat{p}_{ij}^k(t)=\frac{1}{V\min_j\{q_j(t)+\gamma-C_{ij}^*\}} \le \frac{1}{V(\gamma-1)}, \tag{61}$$

where the inequality holds because $q_j(t)\ge 0$ and $C_{ij}^*\le 1$. We now compare the pair (i,k) with another pair $(i',k')\ne(i,k)$. Note that for any $c,c'\in[0,1]$, we have

$$q_j(t)+\gamma-c \ge q_j(t)+\gamma-1 \ge \frac{\gamma-1}{\gamma}\left(q_j(t)+\gamma-c'\right). \tag{62}$$

Applying this relationship to the KKT condition (61), we have

$$\begin{aligned}\sum_{j=1}^{J}\widehat{p}_{ij}^k(t) &= \frac{1}{V\min_j\{q_j(t)+\gamma-C_{ij}^*\}}\\ &\le \frac{\gamma/(\gamma-1)}{V\min_j\{q_j(t)+\gamma-C_{i'j}^*\}} = \frac{\gamma}{\gamma-1}\sum_{j=1}^{J}\widehat{p}_{i'j}^{k'}(t).\end{aligned}$$

By the capacity constraint (15),

$$\sum_{j=1}^{J}\widehat{p}_{ij}^k(t)+\sum_{(i',k')\ne(i,k)}\sum_{j=1}^{J}\widehat{p}_{i'j}^{k'}(t)\le\mu,$$

where $\mu \triangleq \sum_{j=1}^{J} \mu_j$. Combining the above two inequalities, we have

$$\sum_{j=1}^{J} \widehat{p}_{ij}^{k}(t) \leq \frac{\mu}{1+(n(t)-1)(\gamma-1)/\gamma} \leq \frac{\gamma\mu}{(\gamma-1)n(t)} \triangleq P_A,$$

where $n(t) = \sum_{i=1}^{I} n_i(t)$. Combining the last displayed equation with (61), we have

$$\sum_{j=1}^{J} \widehat{p}_{ij}^{k}(t) \leq \min\left\{P_A, \frac{1}{V(\gamma-1)}\right\}. \tag{63}$$

Next, we consider $[p_{ij}^{k}(t)]$, which is the maximizer of $f(\mathbf{p}|\mathbf{n}, \mathbf{C}(t)))$ over the constraint (15). Let $q_j(t)$ now denote the optimal dual variable for this optimization problem. Using the KKT condition again, we have

$$\sum_{j=1}^{J} p_{ij}^{k}(t) = \frac{1}{V \min_j\{q_j(t)+\gamma-C_{ij}^{k}(t)\}}.$$

Combining the last displayed equation with (62) again, we have that for any pairs (i,k) and (i',k'),

$$\sum_{j=1}^{J} p_{ij}^{k}(t) \geq \frac{(\gamma-1)/\gamma}{V\gamma \min_j\{q_j(t)+\gamma-C_{i'j}^{k'}(t)\}} = \frac{\gamma-1}{\gamma}\sum_{j=1}^{J} p_{i'j}^{k'}(t).$$

Now, consider the following two sub-cases. In sub-case 1, suppose that $q_j(t) > 0$ for all j, which means that all servers are fully utilized. Then, we get

$$\sum_{j=1}^{J} \mu_j = \sum_{j=1}^{J} p_{ij}^{k}(t) + \sum_{(i',k')\neq(i,k)} \sum_{j=1}^{J} p_{i'j}^{k'}(t).$$

Hence, it follows from the last two displayed equations that

$$\sum_{j=1}^{J} p_{ij}^{k}(t) \geq \frac{\mu}{1+\gamma(n(t)-1)/(\gamma-1)} \geq \frac{(\gamma-1)\mu}{\gamma n(t)} \triangleq P_B.$$

In sub-case 2, suppose that there is some server j_0 such that $q_{j_0}(t) = 0$. Then

$$\begin{aligned}\sum_{j=1}^{J} p_{ij}^{k}(t) &= \frac{1}{V \min_j\{q_j(t)+\gamma-C_{ij}^{k}(t)\}} \\ &\geq \frac{1}{V(q_{j_0}(t)+\gamma-C_{i,j_0}^{k})} \geq \frac{1}{V\gamma}.\end{aligned}$$

Combining these two sub-cases together, we have

$$\sum_{j=1}^{J} p_{ij}^{k}(t) \geq \min\left\{P_B, \frac{1}{V\gamma}\right\}. \tag{64}$$

Combining (63) and (64), we have

$$\frac{\sum_{j=1}^{J} \widehat{p}_{ij}^{k}(t)}{\sum_{j=1}^{J} p_{ij}^{k}(t)} \leq \frac{\min\{P_A, \frac{1}{V(\gamma-1)}\}}{\min\{P_B, 1/(V\gamma)\}} \leq \left(\frac{\gamma}{\gamma-1}\right)^2.$$

A.6 Proof of Theorem 1: Upper-Bound on the Expected Number of Clients in the System

To prove Theorem 1, we first show that when the number of users in the system is large enough, all the servers must be busy (Lemma 10). Then, we show that when the number of users in the system is sufficiently large, the total number of departures will first-order stochastically dominate a Binomial random variable with mean $(\mu+\lambda)/(2N)$ (Lemma 11). Finally, we construct a coupling between the system and a Geom/Geom/μ queue to bound $n(t)$ and obtain the final result (Lemma 12 and Lemma 13).

Lemma 10. *If $n(t) \geq \mu V\gamma$, then for every server j,*

$$\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t) = \mu_j.$$

Proof. We prove it by contradiction. Suppose that the conclusion does not hold. Then there must exist a server j_0 such that $q_{j_0}(t) = 0$. Since $[p_{ij}^k(t)]$ solves (14), it follows from the KKT condition that for every user k of class i,

$$\sum_{j=1}^{J} p_{ij}^k(t) = \frac{1}{V\min_j\{q_j(t)+\gamma-C_{ij}^k(t)\}} \geq \frac{1}{V\gamma}.$$

Hence, we have

$$\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)}\sum_{j=1}^{J} p_{ij}^k(t) \geq \frac{n(t)}{V\gamma} \geq \mu = \sum_{j=1}^{J}\mu_j,$$

where the second inequality holds due to the assumption that $n(t) \geq \mu V\gamma$. On the other hand, due to the server capacity constraint,

$$\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t) \leq \mu_j.$$

Combining the last two displayed equations yields that

$$\sum_{i\in I(t)}\sum_{k=1}^{n_i(t)} p_{ij}^k(t) = \mu_j$$

for all j, which leads to a contradiction. □

Next, let

$$\tau^* = \mu\gamma\max\left\{V, \frac{2\mu^2}{(\mu-\lambda)(\gamma-1)}\right\}. \tag{65}$$

We present the following lemma which is the key to bound Theorem 1.

Lemma 11. *When the number of clients in the system is sufficiently large, i.e., if*

$$n(t) \geq \tau^*, \tag{66}$$

then the number of total departures at time t, $D(t)$, first-order stochastically dominates W, where

$$W \sim \text{Binom}\left(\mu, \frac{\mu+\lambda}{2N\mu}\right),$$

i.e., $\mathbb{P}\{D(t) \geq x\} \geq \mathbb{P}\{W \geq x\}$ for all $x \geq 0$.

Proof. According to Algorithm 1 (lines 11 to 14), each server has μ_j capacity and thus makes μ_j independent selections of users to serve. In total, there are μ such selections at every time-slot. Let us index these selections by $1, 2, \ldots, \mu$. Note that different selections may select the same user. Let S_ℓ denote the set of users (i,k) who has been selected before the ℓ-th selection. Let $X_\ell = 1$ if the ℓ-th selected user is not in S_ℓ, and $X_\ell = 0$, otherwise. Suppose that server j makes the ℓ-th selection. Then, conditional on S_ℓ, $X_\ell(t)$ is a Bernoulli random variable with mean

$$\begin{aligned}\mathbb{P}\left\{X_\ell(t) = 1 | S_\ell\right\} &= \frac{1}{\mu_j} \sum_{i \in I(t)} \sum_{k=1}^{n_i(t)} p_{ij}^k(t) \mathbf{1}_{\{(i,k) \notin S_\ell\}} \\ &= 1 - \frac{1}{\mu_j} \sum_{i \in I(t)} \sum_{k=1}^{n_i(t)} p_{ij}^k(t) \mathbf{1}_{\{(i,k) \in S_\ell\}} \\ &\geq 1 - \mu \max_{i \in I(t)} \max_{1 \leq k \leq n_i(t)} p_{ij}^k(t),\end{aligned}$$

where the second equality holds due to Lemma 10, and the last inequality holds because $\mu_j \geq 1$ and $|S_\ell| \leq \mu$. Using (49) in the proof of Lemma 8, we have

$$\sum_{j=1}^{J} p_{ij}^k(t) \leq \frac{\gamma \mu}{(\gamma - 1) n(t)}, \quad \forall (i,k).$$

Combining the last two displayed equations yields that

$$\begin{aligned}\mathbb{P}\left\{X_\ell(t) = 1 | S_\ell\right\} &\geq 1 - \frac{\gamma \mu^2}{(\gamma - 1) n(t)} \\ &\geq \frac{\mu + \lambda}{2\mu},\end{aligned} \tag{67}$$

where the last inequality holds due to the assumption (66). Let

$$Y_\ell \overset{\text{i.i.d.}}{\sim} \text{Bern}\left(\frac{\mu + \lambda}{2\mu}\right), \quad \ell = 1, \ldots, \mu. \tag{68}$$

Then we can couple X_ℓ and Y_ℓ such that $X_\ell \geq Y_\ell$ for all $1 \leq \ell \leq \mu$ in the following way by starting from $\ell = 1$.

If $X_\ell = 1$, i.e., a user $(i,k) \notin S_\ell$ is chosen at the ℓ-th selection, then independently of everything else generate $Y_\ell = 1$ with probability

$$\frac{\mu + \lambda}{2\mu \mathbb{P}\left\{X_\ell = 1 | S_\ell\right\}} \overset{(67)}{\leq} 1$$

and $Y_\ell = 0$ otherwise. If $X_\ell = 0$, i.e., a user $(i,k) \in S_\ell$ is chosen at the ℓ-th selection, then let $Y_\ell = 0$. Note that by construction,

$$\mathbb{P}\left\{Y_\ell = 1 | S_\ell\right\} = \mathbb{P}\left\{X_\ell = 1 | S_\ell\right\} \frac{\mu + \lambda}{2\mu \mathbb{P}\left\{X_\ell = 1 | S_\ell\right\}} = \frac{\mu + \lambda}{2\mu}.$$

Hence, Y_ℓ is independent of S_ℓ. As a consequence,

$$\begin{aligned}&\mathbb{P}\left\{Y_\ell = 1 | Y_1, \ldots, Y_{\ell-1}, Y_{\ell+1}, Y_\mu\right\} \\ &= \mathbb{E}_{S_\ell}\left[\mathbb{P}\left\{Y_\ell = 1 | S_\ell, Y_1, \ldots, Y_{\ell-1}, Y_{\ell+1}, Y_\mu\right\}\right] \\ &= \mathbb{E}_{S_\ell}\left[\mathbb{P}\left\{Y_\ell = 1 | S_\ell\right\}\right] = \frac{\mu + \lambda}{2\mu}.\end{aligned}$$

Thus, Y_ℓ's are independently and identically distributed as specified in (68). Moreover, $Y_\ell \le X_\ell$. Let $Z_\ell \overset{\text{i.i.d.}}{\sim} \text{Bern}(1/N)$, which are independent of Y_ℓ's. For each ℓ such that $Y_\ell = 1$ or $X_\ell = 1$, the user selected at the ℓ-th selection has not been selected before. Thus, it runs out of tasks and leaves the system immediately with probability equal to $1/N$, independently of everything else. Thus, we can further construct a coupling such that the ℓ-th selected user leaves the system immediately after the ℓ-th selection if and only if $Z_\ell = 1$. Hence, the total number of departures $D(t)$ satisfies that

$$D(t) \ge \sum_{\ell=1}^{\mu} Y_\ell Z_\ell.$$

The proof is complete by letting

$$W \triangleq \sum_{\ell=1}^{\mu} Y_\ell Z_\ell \sim \text{Binom}\left(\mu, \frac{\mu+\lambda}{2N\mu}\right).$$

□

To complete the proof of Theorem 1, we first establish a coupling between $n(t)$ and $m(t)$, which is a Geom/Geom/μ queue with dynamics given by

$$m(t+1) = \left[m(t) + U(t+1) - D'(t)\right]_+, \tag{69}$$

where $[x]_+ = \max\{x, 0\}$ and

$$D'(t) \sim \text{Binom}\left(\mu, \frac{\mu+\lambda}{2\mu N}\right).$$

Then, we prove that $\mathbb{E}[m(t)]$ is bounded and $\mathbb{E}[n(t)] \le \mathbb{E}[m(t)] + \tau^* + 1$ for all t in Lemma 12 and Lemma 13, respectively.

Lemma 12. *The queue $m(t)$ following (69) is stable and satisfies that*

$$\mathbb{E}[m(t)] \le \mathbb{E}[m(\infty)] \le \frac{\lambda+\mu}{\mu-\lambda}.$$

where $\mathbb{E}[m(\infty)]$ denotes the mean queue length in steady state.

Proof. We first show that $m(t)$ is stable and bound $\mathbb{E}[m(\infty)]$ from the above. Let

$$b_j = \mathbb{P}\left\{D'(t) = j\right\}, \quad \forall j \ge 0.$$

and $\overline{b} = \mathbb{E}[D'(t)]$. Note that $m(t)$ is a discrete-time Markov chain with transition matrix given by

$$P = \begin{bmatrix} 1-a & a & & & & \\ c_1 & d_1 & d_0 & & & \\ c_2 & d_2 & d_1 & d_0 & & \\ c_3 & d_3 & d_2 & d_1 & d_0 & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where

$$a = \lambda/N,\ d_0 = ab_0,\ d_j = ab_j + (1-a)b_{j-1},\ j \ge 1$$
$$c_j = 1 - \sum_{k=0}^{j} d_k,\ j \ge 1.$$

Since $a < \bar{b}$, it is shown in [Neu81] (also cf. [Alf16, Section 4.8.1]) that P is positive recurrent and the steady-state distribution is given by

$$\pi_0 = \frac{b_0(1-r)}{b_0(1-r)+r}, \quad \pi_k = \frac{r^k(1-r)}{b_0(1-r)+r}, \quad \forall k \geq 1,$$

where r is the unique solution in $(0,1)$ of

$$\sum_{k=0}^{\infty} d_k r^{k-1} = 1.$$

Plugging in the expression of d_k, we find that

$$\begin{aligned}\sum_{k=0}^{\infty} d_k r^{k-1} &= \left(1 + a\frac{1-r}{r}\right) \mathbb{E}\left[r^{D'(t)}\right] \\ &= \left(1 + a\frac{1-r}{r}\right)\left(1 - \frac{\bar{b}(1-r)}{\mu}\right)^{\mu},\end{aligned}$$

where the last equality holds in view of the moment generating function of Binomial distribution. Combining the last two displayed equation gives that

$$\left(1 + a\frac{1-r}{r}\right)\left(1 - \frac{\bar{b}(1-r)}{\mu}\right)^{\mu} = 1.$$

Using the expression $1 + x \leq e^x$ for all $x \in \mathbb{R}$, we get that

$$e^{a(1-r)/r - \bar{b}(1-r)} \geq 1,$$

which further implies that $r \leq a/\bar{b}$. Hence, the mean queue length in steady state satisfies

$$\begin{aligned}\mathbb{E}\left[m(\infty)\right] &= \sum_{k\geq 1} k\pi_k \\ &= \frac{r}{(b_0(1-r)+r)(1-r)} \\ &\leq \frac{1}{1-r} \\ &\leq \frac{\bar{b}}{\bar{b}-a} = \frac{\lambda+\mu}{\mu-\lambda}.\end{aligned}$$

Finally, we complete the proof by showing that $\mathbb{E}\left[m(t)\right]$ is non-decreasing in t. Let $Z(t-1) = U(t) - D'(t-1)$. By recursively applying the Lindely's recursion [GOW04, Lemma1.1]:

$$m(t) = \left[m(t-1) + Z(t-1)\right]_+,$$

we get that

$$m(t) = \left[\sum_{s=0}^{t-1} Z(s), \sum_{s=1}^{t-1} Z(s), \ldots, Z(t-1)\right]_+.$$

Let $X(0) = 0$ and $X(s) = Z(0) + \ldots + Z(s-1)$. Since $Z(t)$ are *i.i.d.* across t,

$$\begin{aligned} m(t) &\stackrel{d}{=} [Z(0), Z(0) + z(1), \ldots, Z(0) + \ldots + Z(t-1)]_+ \\ &\stackrel{(d)}{=} \max_{0 \le s \le t} \{X(s)\}, \end{aligned}$$

where $\stackrel{d}{=}$ means "equal in distribution." Thus,

$$m(t+1) \stackrel{(d)}{=} \max_{0 \le s \le t+1} \{X(s)\}$$

first-order stochastically dominates $m(t)$. As a consequence, $\mathbb{E}[m(t+1)] \ge \mathbb{E}[m(t)]$. □

Next, we construct the coupling between $n(t)$ and $m(t)$ and show that $n(t)$ is upper bounded by $m(t)$ plus a constant. Recall that

$$n(t+1) = n(t) + U(t+1) - D(t),$$

where $U(t) \sim \text{Bern}(\lambda/N)$. In view of Lemma 11, when $n(t) \ge \tau^*$, we can couple $D(t)$ and $D'(t)$ such that $D(t) \ge D'(t)$. We than have the following lemma.

Lemma 13. *For any time t, the total number of users $n(t)$ in the system is bounded by*

$$n(t) \le m(t) + \tau^* + 1, \tag{70}$$

where $m(t)$ is the Geom/Geom/μ queue defined in (69).

Proof. We prove this by induction. When $t = 0$, (70) holds trivially. Suppose (70) holds for t, we prove that it also holds for $t+1$. In particular, if $n(t) \le \tau^*$, since $U(t+1) \le 1$, it holds that $n(t+1) \le \tau^* + 1$ and thus (70) holds. If $n(t) \ge \tau^*$, since $D'(t) \le D(t)$, it follows that

$$\begin{aligned} n(t+1) &= n(t) + U(t+1) - D(t) \\ &\le n(t) + U(t+1) - D'(t) \\ &\le m(t) + U(t+1) - D'(t) + \tau^* + 1 \\ &\le m(t+1) + \tau^* + 1. \end{aligned}$$

Therefore, (70) holds for $t+1$ in all cases.

□

It immediately follows from Lemma 13 and Lemma 12 that

$$\begin{aligned} \mathbb{E}[n(t)] &\le \mathbb{E}[m(t)] + \tau^* + 1 \\ &\le \frac{\lambda + \mu}{\mu - \lambda} + \tau^* + 1 \\ &= \frac{2\mu}{\mu - \lambda} \left(1 + \frac{\mu^2 \gamma}{\gamma - 1}\right) + \mu\gamma V, \end{aligned}$$

where the last equality follows from (65).