A Primal-Dual Approach to Constrained Markov Decision Processes with Applications to Queue Scheduling and Inventory Management

(Authors' names blinded for peer review)

In many operations management problems, we need to make decisions sequentially to minimize the cost while satisfying certain constraints. One modeling approach to such problems is the constrained Markov decision process (CMDP). In this work, we develop a data-driven primal-dual algorithm to solve CMDPs. Our approach alternatively applies regularized policy iteration to improve the policy and subgradient ascent to maintain the constraints. Under mild regularity conditions, we show that the algorithm converges at rate $O(1/\sqrt{T})$, where T is the number of iterations, for both the discounted and long-run average cost formulations. Our algorithm can be easily combined with advanced deep learning techniques to deal with large-scale problems, with the added benefit of straightforward convergence analysis. When the CMDP has a weakly coupled structure, our approach can further reduce the computational complexity through an embedded decomposition. We apply the algorithm to two operations management problems: multi-class queue scheduling and multi-product inventory management. Numerical experiments demonstrate that our algorithm, when combined with appropriate value function approximations, generates policies that achieve superior performance compared with state-of-art heuristics.

Key words: Constrained Markov decision process, primal-dual algorithm, queue scheduling, inventory management

1. Introduction

In sequential decision-making problems, a single objective might not suffice to describe the real considerations faced by decision-makers. There are in general two modeling approaches to incorporate multiple objectives. The first is to take a weighted average of different objectives. However, in many applications, it can be hard to determine the appropriate weight on vastly different considerations (e.g., staffing cost versus service quality). The second is to optimize one objective while putting the others as constraints. One modeling tool for the second approach is the constrained Markov decision process (CMDP).

CMDP has been successfully applied in various applications, including admission control and routing in telecommunication networks, scheduling for hospital admissions, and maintenance scheduling for infrastructures (Altman 1999). Girard et al. (2020) propose to model the emergency department (ED) scheduling as a CMDP where one tries to minimize the waiting time of the non-urgent patients while making sure that the waiting time of the urgent patients does not exceed a certain threshold. In practice, the waiting-time threshold for urgent patients is set by medical requirements. Meanwhile, it is also important to properly manage the waiting time of non-urgent patients, especially if they may leave without being seen when have waited for too long. Note that it can be challenging to combine the waiting time for urgent and non-urgent patients into a single objective, because it is difficult to quantify the relative costs of waiting between the two types of patients. CMDP can also be applied when we want to improve upon an existing policy to optimize a different performance metric. For example, we may have an existing policy that achieves a small staffing cost, but we want to improve the service quality, which is measured by the average waiting time. In this case, we can put the staffing cost under the current policy as a constraint, and try to minimize the average waiting time subject to that constraint.

Given the wide range of applications, it is highly desirable to have an efficient solution method for CMDPs. Due to the complicated system dynamics and the scale of the problems, exact optimal solutions to CMDPs can rarely be derived analytically. Instead, numerical approximations become the main workhorse to study CMDPs. In this paper, we propose a general data-driven primal-dual algorithm for CMDPs. Our algorithm can be applied to both the model-based setting, where we have direct access to the transition kernel of the underlying Markov chain, and the model-free setting, where the system transition dynamics can only be estimated from data.

When we know the transition kernel explicitly, one classic approach to solving CMDP is based on its linear programming (LP) formulations (Altman 1999). This approach works well when the state space is relatively small. When dealing with large-scale problems, approximate linear programming has been developed (Schweitzer and Seidmann 1985, De Farias and Van Roy 2003). However, it requires the value function parameterization to be linear. More advanced approximation techniques such as neural networks cannot be applied, which limits our ability to learn complicated value functions. An alternative approach is to apply the Lagrangian duality. In particular, by penalizing the constraints and utilizing strong duality, we can translate the CMDP into a max-min problem (Altman 1999). This enables us to apply subgradient-based algorithms where we iteratively update the policy and the Lagrangian multiplier. For a given Lagrangian multiplier, the inner minimization problem is just an unconstrained Markov decision process (MDP), which can be solved using standard dynamic programming (DP) techniques. The Lagrangian duality approach can be applied to both the model-based and model-free settings. When solving large-scale MDPs using policy/value iteration or policy gradient, we can apply a wider range of approximation techniques. However, the existing method requires solving the MDP to obtain the optimal policy for each updated Lagrangian multiplier (see, for example, Le et al. (2019), Miryoosefi et al. (2019)), i.e., we have to solve multiple MDPs, which can be computationally costly and unnecessary.

Our algorithmic development builds on a combination of the primal-dual update and mirror descent framework (Nemirovski 2004). This allows us to do only a single policy evaluation at each iteration instead of solving the MDP. In particular, for each primal update, we only need to execute one round of regularized policy iteration. The mirror descent framework also allows us to establish neat finite-time performance bounds. We show that the objective function and constraints converge at rate $O(1/\sqrt{T})$ where T is the number of primal-dual updates. Compared to existing algorithms, our primal-dual algorithm enjoys a much lower computational cost at each iteration (since we only need to do a policy evaluation rather than a policy optimization), while achieving the same convergence rate (Le et al. 2019, Miryoosefi et al. 2019). Our algorithm and convergence analysis applies to both the accumulated discounted cost and long-run average cost formulations. The mirror descent framework also allows us to establish performance bounds when the value function and policy function are estimated with errors.

We apply our primal-dual algorithm to solve two important classes of operations management problems: queue scheduling and inventory planning. For queue scheduling, we first consider a two-class queue motivated by ED operations, where we want to minimize the queue length (waiting time) of one class while making sure that the queue length of the other class does not exceed a certain threshold (Girard et al. 2020). We demonstrate through this relatively simple example that our algorithm learns the optimal policy. We also consider a multi-class multi-pool parallel-server system motivated by hospital inpatient flow management, where the decision-maker needs to route different classes of customers to different pools of servers to minimize the total waiting cost while maintaining a certain constraint on the number of mismatches between customers and servers (Dai and Shi 2019). Value function approximation with the quadratic basis is applied to handle the large state space. The policy learned by our algorithm achieves 5% - 18% cost reductions compared to well-known benchmark methods in the literature, such as the modified $c\mu$-rules and max-pressure policies (Chen et al. 2020).

For inventory planning, we focus on the multi-product multi-period setting with a weakly coupled structure (Singh and Cohn 1998), and show that our primal-dual algorithm can achieve dimension reduction through sub-problem decomposition. We demonstrate through a simple two-product newsvendor problem with a storage space constraint that our algorithm converges at the theoretical rate. We then consider more challenging problems with perishable goods and stochastic lead times. The goal is to minimize the holding and fixed order cost while making sure that the lost sale does not exceed a certain threshold. To handle the large state space, we apply neural networks to approximate the Q-function and policy. The policy learned our algorithm achieves 19% - 32% cost reductions compared to the modified (s, S) policies, which is known to be near optimal for special cases of the problem (Scarf 1960, Iglehart 1963).

1.1. Main contributions

Our main contributions are twofold. First, we develop a data-driven primal-dual algorithm to solve CMDPs. The algorithm and our development have several advantages. First and foremost, the algorithm achieves $O(1/\sqrt{T})$ convergence rate and only requires a single regularized policy iteration (which is essentially a policy evaluation) at each primal update. Second, it can be applied to both the accumulated discounted cost and long-run average cost formulations and achieves the same convergence rate. Third, our algorithm can also be easily combined with other approximate dynamic programming techniques, including nonlinear parameterization, to solve large-scale problems with unknown transition probabilities. We also quantify the effect of approximation errors on the convergence rate. Lastly, the primary-dual update leads to a natural decomposition of weakly coupled CMDPs, such that further reduction of computational complexity is possible.

Second, we demonstrate how to apply our algorithm to solve large-scale multi-class queue scheduling and multi-product inventory management problems. Several recent engineering developments, including choosing good basis functions for value function approximation and conducting efficient policy evaluation utilizing the special structure of the problem, can be easily adapted to our algorithm. For the multi-class multi-pool queue scheduling problem, we find that quadratic basis combined with the least-square temporal difference algorithm for policy evaluation works well. This generalizes the insights from Dai and Shi (2019) to the CMDP setting. For multi-product inventory management with perishable goods and stochastic lead times, we need to use neural networks to approximate the Q-function and policy. It is also important to initialize the training with a good exploration policy. In all problems tested, the policies learned by our algorithm achieve significant cost savings compared to the state-of-art benchmark heuristics.

1.2. Related literature

In terms of the problem formulation, multi-objective sequential decision-making problems has received considerable attention recently (see, e.g., Hayes et al. (2022) for a survey on multi-objective reinforcement learning). Existing approaches can be roughly divided into two categories: single-policy and multiple-policy. The single-policy approach focuses on finding the optimal policy for a given preference weight, which translate multiple objectives into a single one (Tesauro et al. 2007, Mannor and Shimkin 2001). The multi-policy approach aims to identify a set of policies that characterizes the Pareto frontier (Natarajan and Tadepalli 2005, Yang et al. 2019, Zhou et al. 2020). The CMDP formulation can be viewed as a special case of the second approach, where we optimize one objective while "fixing" the values of the other objectives. In this work, we focus on CMDPs with linear expectation constraints. Other forms of constraints are also studied in the literature, mostly for MDPs. For example, Miryoosefi et al. (2019) consider the feasibility problem where the expected costs need to be within a convex set. Chow et al. (2017) study MDPs with chance constraints.

Most existing algorithms for CMDPs can be divided into three categories: LP based, dynamic programming (DP) based, and policy gradient (PG) based. Many recent developments on LP based methods focus on large-scale problems and try to exploit the special problem structures. For example, Bertsimas and Orlin (1994) use the ellipsoid method to derive efficient algorithms for problems with side constraints, including the traveling salesman and vehicle routing problems. Neely (2011) studies a linear fractional programming method to solve CMDPs. More recently, Caramanis et al. (2014) propose two algorithms based on column generation and generalized expert's framework. One limitation of these LP based methods is that explicit knowledge of the transition kernel is required. For MDPs (without constraints), Chen and Wang (2016) reformulate the LP as a saddle point problem and use stochastic approximation to solve it. Lee and He (2019) propose a model-free off-policy algorithm based on an LP formulation of Q-learning. Lin et al. (2020) combine the Approximate LP (ALP) with proximal stochastic mirror descent. There are limited developments for CMDPs. In addition, ALP requires linear parameterization. More advanced nonlinear parameterizations such as neural networks cannot be directly applied.

The DP based methods, including policy iteration and value iteration, penalize the constraints via Lagrangian multipliers. Topaloglu (2009) applies the subgradient optimization method to find a good set of Lagrangian multipliers. Brown and Smith (2020) use a cutting plane method to solve for the optimal Lagrangian multipliers. Both papers consider finite-horizon CMDPs (as a relaxation of weakly-coupled MDPs) with explicitly known transition kernels, and assume that the subgradient, which corresponds to solving the penalized MDP at a given Lagrangian multiplier, can be evaluated exactly. One of the advantages of DP based methods is that the value function can be estimated via data, i.e., we do not need to know the transition kernel explicitly. For example, Gattami (2019) formulates the CMDP as a zero-sum game and applies a primal-dual Q-learning algorithm in the model-free setting. Singh and Kumar (2018) develop a two-timescale stochastic approximation algorithm where the policy is updated with a much large step-size than the Lagrangian multiplier. These papers only establish an almost sure convergence. Le et al. (2019) study CMDPs in the offline learning setting, and combine various approximate dynamic programming techniques with subgradient descent to solve it. Miryoosefi et al. (2019) consider the feasibility problem with a convex target set. An $O(1/\sqrt{T})$ convergence rate is obtained in both Le et al. (2019) and Miryoosefi et al. (2019). However, these algorithms require fully solving the penalized MDP at each updated Lagrangian multiplier. Our method can be viewed as an improved version of Le et al. (2019), since we only require a policy evaluation at each iteration while achieving the same convergence rate.

A common challenge in solving CMDPs is the curse of dimensionality. The PG based methods tackle this challenge by parameterizing the policies. In this line, Borkar (2005) and Bhatnagar and Lakshmanan (2012) combine actor-critic algorithms and Lagrangian duality to solve CMDPs. Tessler et al. (2018) develop a two-timescale stochastic approximation algorithm, and Chow et al. (2018) use Lyapunov functions to handle the constraints. Note that for PG based methods, the corresponding optimization problems are typically nonconvex. Thus, in most cases, only convergence to a local minimum can be guaranteed and the convergence rates are largely unknown.

From the application's perspective, our work is related to works that develop numerical algorithms to solve queue scheduling and inventory management problems. For queue scheduling, Veatch (2005), Moallemi et al. (2008), and Dai and Shi (2019) consider temporal difference learning algorithms and combine them with appropriate value function approximations. More recently, Dai and Gluzman (2020) use policy gradient combined with deep neural network approximation for queue scheduling problems under the long-run average cost formulation. For inventory management, many papers study ALP based algorithms (see, for example, Topaloglu and Kunnumkal (2006), Sun et al. (2014), Nadarajah et al. (2015), Lin et al. (2020)). The performance of ALP algorithms relies heavily on the choice of good basis functions. With the development of deep learning, there is also a growing body of literature applying deep reinforcement learning for inventory management problems (see Van Roy et al. (1997) for one of the earliest work). More recent developments include Gijsbrechts et al. (2021), which investigates various policy gradient methods for a single product newsvendor problem; Oroojlooyjadid et al. (2022), which applies deep Q-learning to the beer game. Most of these papers focus on the classic MDP formulation. Our work complements the existing literature by extending the algorithmic development to CMDPs.

1.3. Paper organization and notations

The rest of the paper is organized as follows. In Section 2, we first introduce the basic setup and review some classic results that are relevant to our subsequent developments. We then present our algorithm in Section 3, and show that the algorithm achieves $O(1/\sqrt{T})$ convergence rate in Section 4. In Section 5, we extend our algorithm to the reinforcement learning setting where the transition dynamic is not known explicitly but can be learned from data. We also quantify the convergence rate when there are approximation errors. In Sections 6 and 7, we demonstrate how to apply our algorithm to queue scheduling and inventory management problems. Lastly, we conclude the paper and discuss some future research directions in Section 8.

The following notations are used throughout the paper. For a positive integer K, we denote $[K]$ as the set $\{1,2,\dots,K\}$. For a vector $\lambda \in \mathbb{R}^K$, $[\lambda]_k$ denotes its k-th coordinate and $\|\lambda\| = (\sum_{k=1}^K [\lambda]_k^2)^{1/2}$ denotes its L_2 norm. Given two vectors $a, b \in \mathbb{R}^K$, we say $a \le b$ if the inequality holds coordinate-wise, i.e., $[a]_k \le [b]_k \ \forall k \in [K]$. Given a vector $x \in \mathbb{R}^K$, $[x]^+ = (\max\{[x]_1, 0\}, \dots, \max\{[x]_K, 0\})$.

Finally, given two sequences of real numbers $\{a_n\}_{n\geq 1}$ and $\{b_n\}_{n\geq 1}$, we say $b_n = O(a_n)$, $b_n = \Omega(a_n)$, and $b_n = \Theta(a_n)$ if there exist some constants $C, C' > 0$ such that $b_n \leq C a_n$, $b_n \geq C' a_n$, and $C' a_n \leq b_n \leq C a_n$, respectively. We also introduce the $\tilde{O}(\cdot)$ notation when we ignore the logarithmic factors on $O(\cdot)$. For example, if $b_n \leq C a_n \log(n)$, we can write $b_n = \tilde{O}(a_n)$.

2. Constrained Markov Decision Process

We start by introducing the discrete-time MDP with discounted costs. It is characterized by the tuple $(\mathcal{S}, \mathcal{A}, P, c, \gamma, \mu_0)$. Here, $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces; $P = \{P(\cdot|s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ is the collection of probability measures indexed by the state-action pair (s,a). For each (s,a), $P(\cdot|s,a)$ characterizes the one-step transition probability of the Markov chain conditional on being in state s and taking action a. The function $c = \{c(s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ is the expected instantaneous cost, where $c(s,a)$ is the cost incurred by taking action a at state s. Lastly, $\gamma \in (0,1)$ and $\mu_0 = \{\mu_0(s)\}_{s\in\mathcal{S}}$ are the discount rate and the distribution of the initial state, respectively. Given an MDP, a policy π determines what action to take at each state. We define the expected cumulative discounted cost with initial state s_0 under policy π as

$$V^\pi(s_0) = (1-\gamma)\cdot \mathbb{E}^\pi\big[\sum_{t=0}^{\infty} \gamma^t \cdot c(s_t, a_t)\big|s_0\big], \tag{1}$$

where s_t, a_t are the state and action at time t and $\mathbb{E}^\pi$ denotes the expectation with respect to the transition dynamics determined by policy π. We further weight the costs according to the initial state distribution and define

$$C(\pi) = \mathbb{E}_{s_0\sim\mu_0}\big[V^\pi(s_0)\big]. \tag{2}$$

Our goal is to minimize the cost $C(\pi)$ over a properly defined class of policies.

As an extension to MDP, CMDP optimizes one objective while keeping others satisfying certain constraints. Specifically, in addition to the cost c, we introduce K auxiliary instantaneous costs $d_k = \{d_k(s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}, \forall\ k \in [K]$. Then, the CMDP aims to find a policy that minimizes the cost defined in (2) while keeping the following constraints satisfied (Altman 1999)

$$D_k(\pi) = (1-\gamma)\cdot \mathbb{E}_{s_0\sim\mu_0}\Big[\mathbb{E}^\pi\big[\sum_{t=0}^{\infty} \gamma^t \cdot d_k(s_t, a_t)\big|s_0\big]\Big] \leq q_k,\ \forall\ k \in [K]. \tag{3}$$

To be concise, we define $D(\pi) := (D_1(\pi), \ldots, D_K(\pi))^\top$, $q := (q_1, \ldots, q_K)^\top$, and write the constraints in (3) as $D(\pi) \leq q$.

We also consider MDPs with the long-run average cost. Under policy π, the long-run average cost is defined as

$$C(\pi) = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}^\pi\Big[\sum_{t=0}^{T-1} c(s_t, d_t)|s_0\Big]. \tag{4}$$

With a slight abuse of notation, throughout the paper, we use the same notations for both the discounted and long-run average cost formulations. The actual formulation should be clear from the context. For technical tractability, we also assume the limit in (4) exists and does not depend on the initial state s_0. Moreover, there exists a function $V^\pi(\cdot)$ such that

$$\mathbb{E}^\pi\Big[\sum_{t=0}^{T-1} c(s_t, d_t)|s_0\Big] = C(\pi)\cdot T + V^\pi(s_0) + o(T).$$

$V^\pi(s)$ satisfies the following Poisson equation

$$C(\pi) + V^\pi(s) = \sum_{a\in\mathcal{A}}\Big(c(s,a) + \sum_{s'\in\mathcal{S}} V^\pi(s')P(s'|s,a)\Big)\cdot\pi(a|s). \tag{5}$$

It is often referred to as the relative value function, and plays a similar role as equation (1) for MDPs with discounted cost. Note that the solution to the Poisson equation (5) is unique up to a constant, i.e., shifting $V^\pi(s)$'s by a common constant remains a valid solution. In subsequent convergence analysis, we pick the solution with $\sum_{s\in\mathcal{S}} V^\pi(s)\cdot\nu^\pi(s) = 0$, where ν^π is the stationary measure of the Markov chain under policy π. For a CMDP with instantaneous auxiliary costs $d_k(s,a)$, the long-run average auxiliary costs are defined as

$$D_k(\pi) = \lim_{T\to\infty}\frac{1}{T}\mathbb{E}^\pi\Big[\sum_{t=0}^{T-1} d_k(s_t, d_t)|s_0\Big],\ \forall k\in[K]. \tag{6}$$

Then, we aim to solve

$$\min_\pi\ C(\pi),\ \text{s.t. } D_k(\pi)\le q_k,\ \forall k\in[K].$$

We remark that CMDP is only one modeling choice to study sequential decision problems with multiple objectives/constraints. This particular modeling choice turns out to enjoy a lot of analytical and computational tractability as we will discuss next.

2.1. Policy Spaces

Solving CMDPs requires finding the optimal policy over a properly defined policy space, which is a function space. Imposing suitable regularity conditions on the policy space facilitates the development of efficient computational algorithms. We next introduce some commonly used policy classes. It is natural to require that all policies are non-anticipative, which means that the decision-maker does not have access to future information. Define the history at time t to be the sequence of previous states and actions as well as the current state, i.e., $h_t := (s_0, a_0, \ldots, a_{t-1}, s_t)$. Then a non-anticipative policy can be viewed as a mapping from h_t and t to the action space. We refer to such a policy as a *"behavior policy"*. If a policy only depends on the current state s_t and time t instead of the whole history h_t, it is called a *"Markov policy"*. For a Markov policy, if it is independent of the time index t, it is referred to as a *"stationary policy"*. When a stationary policy is a deterministic mapping from the state space to the action space, it becomes a *"stationary deterministic policy"*. We use Π, Π_M, Π_S, Π_D to denote the space of behavior, Markov, stationary, and stationary deterministic policies, respectively.

Given a policy space U, we can further extend it by allowing initial randomization, which is referred to as the *"mixing policy"*. More precisely, let ρ be a probability measure defined on the σ-algebra generated by U. Here we implicitly assume the existence of a σ-algebra (see Section 6.3 of Altman (1999) for a detailed construction). To execute a mixing policy on U with initial randomization ρ, the decision-maker first uses ρ to sample a policy $\pi_g \in U$ and then proceeds with that policy for the entire horizon. We denote by $\mathcal{M}(U)$ the space of mixing policies constructed from U. An important special case is $\mathcal{M}(\Pi_S)$, i.e., the space of mixing stationary policies. Note that the cost under the mixing policy is simply a weighted average of costs under individual policies in the mixing, and the weight corresponds to the initial randomization. To see this, consider a simple example where $U = \{\pi_1, \pi_2\}$. For $\pi \in \mathcal{M}(U)$ with initial probability $\rho = (\rho_1, \rho_2)$, we have

$$C(\pi) = \rho_1 C(\pi_1) + \rho_2 C(\pi_2).$$

When solving CMDPs, considering the mixing policy space would greatly facilitate our algorithmic development and analysis. Many existing works on CMDP algorithms also consider the mixing policy space (see, for example, Le et al. (2019), Miryoosefi et al. (2019)). We will explain this in more detail in Section 3.

Finally, a class of policies U is called a *"dominating class"* for a CMDP, if for any policy $\pi \in \Pi$, there exists a policy $\bar{\pi} \in U$ such that

$$C(\bar{\pi}) \leq C(\pi) \text{ and } \; D_k(\bar{\pi}) \leq D_k(\pi), \; \forall \; k \in [K].$$

When the instantaneous costs $c(\cdot,\cdot)$ and $d_k(\cdot,\cdot)$ are uniformly bounded from below, Π_S is dominating. The class of mixing stationary policies $\mathcal{M}(\Pi_S)$ is also dominating (Theorem 3.1 and Theorem 8.4 in (Altman 1999)).

2.2. LP Based Approaches to Solving CMDPs

In this section, we introduced the LP formulations of CMDPs. LP based algorithms are model-based, which are quite different from our proposed approach. However, we utilize some of the concepts in the LP formulations in our development. To be concise, we only discuss CMDPs with discounted costs here. Similar results hold for the long-run average cost formulation as well (see Chapter 4 of Altman (1999)).

Traditionally, there are two LP formulations of CMDPs. The first utilizes the occupation measure. Given a policy π, the occupation measure is defined as

$$\nu^{\pi}(s,a) := (1-\gamma)\cdot \mathbb{E}_{s_0\sim\mu_0}\Big[\sum_{t=0}^{\infty}\gamma^t \bar{P}^{\pi}(s_t=s,a_t=a|s_0)\Big],\ \forall\ (s,a)\in\mathcal{S}\times\mathcal{A}, \tag{7}$$

where $\bar{P}^{\pi}(\cdot,\cdot|s_0)$ denotes the probability measure induced by policy π starting from state s_0. Note that the occupation measure is the weighted long-run proportion of time the system spends at each state-action pair. Due to discounting, it depends on the initial distribution μ_0. By the definition of $\nu^{\pi}(s,a)$, we can express the discounted costs in (2) and (3) as

$$C(\pi) = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} c(s,a)\cdot\nu^{\pi}(s,a), \quad D_k(\pi) = \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} d_k(s,a)\cdot\nu^{\pi}(s,a),\ \forall\ k\in[K]. \tag{8}$$

Let $\mathcal{Q}$ denote the set of feasible occupation measures, i.e., for any occupancy measure $\nu\in\mathcal{Q}$, there exists a policy π that leads to ν (Theorem 3.2 in Altman (1999)). The set $\mathcal{Q}$ can be represented as the collection of vectors $\{\nu(s,a)\}_{(s,a)\in\mathcal{S}\times\mathcal{A}}$ that satisfies the following system of linear equations:

$$\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \nu(s,a)\Big(1(s=s') - \gamma P(s'|s,a)\Big) = (1-\gamma)\cdot\mu_0(s'),\ \forall\ s'\in\mathcal{S},(s,a)\in\mathcal{S}\times\mathcal{A}, \tag{9}$$

and

$$\sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} \nu(s,a) = 1, \quad \nu(s,a)\geq 0,\ \forall\ (s,a)\in\mathcal{S}\times\mathcal{A}, \tag{10}$$

where $1(\cdot)$ is the indicator function. This characterization gives rise to an occupation measure based LP formulation of CMDP:

$$\begin{aligned} \min_{\nu}\ & \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} c(s,a)\cdot\nu(s,a) \\ \text{s.t.}\ & \sum_{(s,a)\in\mathcal{S}\times\mathcal{A}} d_k(s,a)\cdot\nu(s,a)\leq q_k,\ \forall\ k\in[K] \\ & \text{equations (9) and (10).} \end{aligned}$$

The second formulation utilizes Lagrangian duality, which is the basis of our primal-dual algorithm as well. We penalize the constraints with $\lambda\in\mathbb{R}^K$ and define the Lagrangian

$$L(\pi,\lambda) := C(\pi) + \sum_{k=1}^{K}[\lambda]_k\cdot(D_k(\pi)-q_k). \tag{11}$$

Then, the CMDP can be equivalently formulated as $\inf_{\pi\in\Pi_S}\sup_{\lambda\geq 0} L(\pi,\lambda)$. When the state and action spaces are finite, we can exchange the order of inf and sup (Theorem 3.6 in Altman (1999)), i.e., $\inf_{\pi\in\Pi_S}\sup_{\lambda\geq 0} L(\pi,\lambda) = \sup_{\lambda\geq 0}\inf_{\pi\in\Pi_S} L(\pi,\lambda)$. Note that for each fixed λ, $\inf_{\pi\in\Pi_S} L(\pi,\lambda)$ is an unconstrained MDP with instantaneous cost $c(s,a)+\sum_{k=1}^{K}\lambda_k d_k(s,a)$. The optimal value function $V_{\lambda}^{*}(\cdot)$ to problem $\inf_{\pi\in\Pi_S} L(\pi,\lambda)$ satisfies the Bellman equation:

$$V_{\lambda}^{*}(s) = \min_{a\in\mathcal{A}}(1-\gamma)\cdot\Big(c(s,a)+\sum_{k=1}^{K}\lambda_k\cdot d_k(s,a)\Big) + \gamma\cdot\sum_{s'\in\mathcal{S}} V_{\lambda}^{*}(s')P(s'|s,a),\ \forall s\in\mathcal{S}.$$

We can use a system of linear inequalities to represent the Bellman equation. Moreover, since $L(\pi,\lambda)$ is linear in λ, we obtain the following Lagrangian dual based LP formulation of CMDP:

$$\begin{aligned}
&\max_{V_\lambda^*,\lambda} \sum_{s\in\mathcal{S}} \mu_0(s)V_\lambda^*(s) - \sum_{k=1}^K \lambda_k q_k \\
&\text{s.t. } V_\lambda^*(s) \le (1-\gamma)\cdot\Big(c(s,a)+\sum_{k=1}^K \lambda_k \cdot d_k(s,a)\Big) + \gamma\cdot \sum_{s'\in\mathcal{S}} V_\lambda^*(s')P(s'|s,a), \\
&\qquad \forall s\in\mathcal{S}, \forall a\in\mathcal{A}, \lambda_k \ge 0,\ \forall k\in[K].
\end{aligned}$$

Various methods have been developed in the literature to solve the LPs introduced above more efficiently (see, for example, Bertsimas and Orlin (1994), Caramanis et al. (2014)). However, there are two main obstacles to LP-based approaches. First, it can be computationally prohibitive when dealing with a large state or action space. Second, it requires explicit characterization of the transition kernel P. To overcome these difficulties, we next develop a primal-dual algorithm that is data-driven and can be easily adapted to solve large-scale problems.

3. The Primal-Dual Algorithm

Consider the Lagrangian dual problem

$$\sup_{\lambda\ge 0} \inf_{\pi\in\Pi_S} L(\pi,\lambda). \tag{12}$$

For each fixed λ, the inner problem is an unconstrained MDP. A natural idea is to solve the unconstrained MDP via a data-driven method and then update the Lagrangian multipliers via subgradient ascent. Such an idea is explored in (Le et al. 2019). However, this method is computationally expensive, since we need to solve a new MDP every time the Lagrangian multipliers are updated. In contrast, our method only requires a policy evaluation at each iteration.

We develop the algorithm and analyze its convergence in $\mathcal{M}(\Pi_S)$, the space of mixing stationary policies. The benefits of allowing the mixing are twofold. First, it provides an intuitive way to understand strong duality:

$$\inf_{\pi\in\mathcal{M}(\Pi_S)} \sup_{\lambda\ge 0} L(\pi,\lambda) = \sup_{\lambda\ge 0} \inf_{\pi\in\mathcal{M}(\Pi_S)} L(\pi,\lambda). \tag{13}$$

With the mixing operation, we can treat $C(\pi)$ and $D(\pi)$ as linear functions with respect to the initial randomization. To see this, recall the simple example where $U=\{\pi_1,\pi_2\}$. For $\pi\in\mathcal{M}(U)$ with mixing probability $\rho=(\rho_1,\rho_2)$, $C(\pi)=\rho_1 C(\pi_1)+\rho_2 C(\pi_2)$, which is as a linear function of ρ_1 and ρ_2. Minimizing $C(\pi)$ is essentially minimizing over (ρ_1,ρ_2) with $\rho_1,\rho_2\in[0,1]$, $\rho_1+\rho_2=1$. In general, U may contain infinitely many policies. Then, the Lagrangian $L(\pi,\lambda)$ is infinite-dimensional bilinear (with respect to the Lagrangian multiplier λ and the mixing probability ρ) and strong duality follows from the minimax theorem (Sion et al. 1958).

Second, in primal-dual algorithms, we in general need to take the average of the trajectories to obtain convergence (Nedić and Ozdaglar 2009). In our case, caution needs to be taken when defining the average. Note that the objective and constraints are inner products of the costs and the occupation measure (as in (8)). The Lagrangian $L(\pi, \lambda)$ is also linear with respect to the occupation measure. Thus, what we need to take the average over are the occupation measures. Since the mapping from a policy to the corresponding occupation measure is not linear, we cannot average the policies directly, e.g., by averaging $\pi_i(a|s)$'s for each (s, a) pair. However, the mixing operation provides a straightforward way to average the occupation measures. For example, when $U = \{\pi_1, \pi_2\}$, for $\pi \in \mathcal{M}(U)$ with mixing probability $\rho = (\rho_1, \rho_2)$, $\nu^{\pi}(s,a) = \rho_1 \nu^{\pi_1}(s,a) + \rho_2 \nu^{\pi_2}(s,a)$. In addition, note that for CMDPs with finite state and action spaces, given a mixing policy, there exists a non-mixing stationary policy that has the same occupation measure (Theorem 3.1 of Altman (1999)). (See, also, Chapter 6 and 10 of Altman (1999) for more general results when the state space is infinite.) In particular, for $\pi \in \mathcal{M}(\Pi_S)$, let $\nu^{\pi}(\cdot, \cdot)$ be the corresponding occupation measure. Then, we can construct an "equivalent" stationary policy $\tilde{\pi}$ via

$$\tilde{\pi}(a|s) = \frac{\nu^{\pi}(s,a)}{\sum_{a \in \mathcal{A}} \nu^{\pi}(s,a)}. \tag{14}$$

Our algorithmic development is based on strong duality (13), which holds under certain regularity conditions (see Section 4 for details). By the minimax theorem, there exists a saddle point (π^*, λ^*) such that

$$L(\pi^*, \lambda) \leq L(\pi^*, \lambda^*) \leq L(\pi, \lambda^*), \ \forall \ \lambda \in \mathbb{R}_+^K, \pi \in \mathcal{M}(\Pi_S). \tag{15}$$

Moreover, π^* is an optimal solution to the primal problem, λ^* is an optimal solution to the dual problem, and $L(\pi^*, \lambda^*)$ equals to the optimal cost of the CMDP. The saddle point property (15) suggests that we can use mirror descent to find the saddle point. The key in the application of mirror descent is to choose the appropriate "distance function" (potential function) that is adapted to the geometry of the problem.

We first consider CMDPs with discounted costs. For the primal policy update, the policy $\pi(\cdot|s)$ at a given state s is a probability measure over the action space. In this case, we use KL divergence at each state as the state-wise potential function, which gives rise to a regularized policy iteration. For a given λ, the inner inf-problem in $\sup_{\lambda \geq 0} \inf_{\pi \in \mathcal{M}(\Pi_S)} L(\pi, \lambda)$ is an unconstrained MDP with modified instantaneous cost

$$c^{\lambda}(s,a) := c(s,a) + \sum_{k=1}^{K} [\lambda]_k (d_k(s,a) - q_k).$$

We refer to $\inf_{\pi\in\mathcal{M}(\Pi_S)} L(\pi,\lambda)$ as the modified unconstrained MDP. For a given policy π and Lagrangian multiplier λ, define

$$Q^{\pi,\lambda}(s,a) := (1-\gamma)\cdot\Big(c^{\lambda}(s,a) + \mathbb{E}^{\pi}\Big[\sum_{t=1}^{\infty}\gamma^t c^{\lambda}(s_t,a_t)\big|s_0=s,a_0=a\Big]\Big), \tag{16}$$

which is known as the action-value function or Q-function under policy π. Let π_m and λ_m denote the policy and Lagrangian multiplier obtained at iteration m. For each state $s\in\mathcal{S}$, the regularized policy iteration is defined as

$$\pi_m(a|s) = \mathop{\arg\min}_{\pi(\cdot|s)\in\Delta_{\mathcal{A}}}\Big\{\big\langle Q^{\pi_{m-1},\lambda_{m-1}}(s,\cdot),\pi(\cdot|s)\big\rangle + \eta_{m-1}^{-1}\cdot\mathrm{KL}\big(\pi(\cdot|s)\|\pi_{m-1}(\cdot|s)\big)\Big\}, \tag{17}$$

where $\eta_{m-1}>0$ is the stepsize that determines the magnitude of regularization. The minimization is taken over the probability simplex $\Delta_{\mathcal{A}} := \{\pi(\cdot|s) : 0\le\pi(a|s)\le 1, \sum_{a\in\mathcal{A}}\pi(a|s)=1\}$.

For the dual Lagrangian multiplier update, we work with the squared Euclidean distance, which gives rise to a projected subgradient ascent. Let Λ_M denote a suitably bounded domain that includes the dual optimal solution λ^* in its interior. We will provide an explicit construction of Λ_M in (24) in Section 4. The projected subgradient ascent takes the form

$$\lambda_m = \mathrm{Proj}_{\Lambda_M}\Big\{\lambda_{m-1} + \eta_{m-1}\cdot\partial_{\lambda}L(\pi_{m-1},\lambda_{m-1})\Big\}, \tag{18}$$

where $\mathrm{Proj}_{\Lambda_M}\{\cdot\}$ denotes the projection (in L^2-norm) onto Λ_M.

In actual implementations, the regularized policy iteration can be re-written as

$$\pi_m(\cdot|s) = Z_{m-1}^{-1}\cdot\pi_{m-1}(\cdot|s)\cdot\exp\big\{-\eta_{m-1}\cdot Q^{\pi_{m-1},\lambda_{m-1}}(s,\cdot)\big\}, \tag{19}$$

where Z_{m-1} is some normalizing constant. For the subgradient ascent update, we have

$$\big[\partial_{\lambda}L(\pi_{m-1},\lambda_{m-1})\big]_k = D_k(\pi_{m-1}) - q_k. \tag{20}$$

Both (19) and (20) can be evaluated/approximated using simulation or empirical data. In addition, we can apply advanced approximation techniques for policy evaluation, i.e., when evaluating $Q^{\pi_{m-1},\lambda_{m-1}}$, to improve the scalability of the algorithm.

The output of the algorithm is a weighted average of the policies and Lagrangian multipliers we obtain at each iteration. In particular, suppose that our algorithm runs $T-1$ iterations and generates a sequence $\{(\pi_m,\lambda_m)\}_{0\le m\le T-1}$. Then, we output a mixing policy and a Lagrangian multiplier of the forms:

$$\bar{\pi}_T = \sum_{m=0}^{T-1}\tilde{\eta}_m\pi_m,\ \ \bar{\lambda}_T = \sum_{m=0}^{T-1}\tilde{\eta}_m\lambda_m,\ \text{ where } \tilde{\eta}_m = \eta_m/\textstyle\sum_{m=0}^{T-1}\eta_m. \tag{21}$$

The averaging is required for convergence, since the objective $L(\pi, \lambda)$ is bilinear and does not possess sufficient convexity. Counter-examples that fail to converge without averaging exist. The summation in the definition of $\bar{\pi}_T$ is interpreted as the mixing operation, i.e., it mixes policies $(\pi_0, \dots, \pi_{T-1})$ with initial randomization $(\tilde{\eta}_0, \cdots, \tilde{\eta}_{T-1})$. From $\bar{\pi}_T$, we can apply (14) to define a non-mixing stationary policy that has the same occupancy measure.

We can easily extend the above development to CMDPs with long-run average costs. To apply the iterative updates, we replace the Q-function in (16) with the relative action-value function, which is defined as

$$Q^{\pi,\lambda}(s,a) = c^{\lambda}(s,a) - C^{\lambda}(\pi) + \sum_{s' \in \mathcal{S}} V^{\pi,\lambda}(s') P(s'|s,a) \tag{22}$$

where $V^{\pi,\lambda}$ is the solution to the Poisson equation induced by policy π:

$$C^{\lambda}(\pi) + V^{\pi,\lambda}(s) = \sum_{a \in \mathcal{A}} \Big(c^{\lambda}(s,a) + \sum_{s' \in \mathcal{S}} V^{\pi,\lambda}(s') P(s'|s,a) \Big) \cdot \pi(a|s), \tag{23}$$

and $C^{\lambda}(\pi)$ is the long-run average cost of the modified problem, i.e., with cost $c^{\lambda}(s,a)$. Similarly, we replace $D_k(\pi)$'s with the long-run average auxiliary cost as defined in (6). Then, the algorithm follows the same primal-dual updates as (18) and (19) at each iteration. The detailed primal-dual algorithm is summarized in Algorithm 1.

4. Performance Analysis

In this section, we conduct the performance analysis of Algorithm 1. To demonstrate the main idea, we start by assuming we can evaluate the Q-function, i.e., $Q^{\pi,\lambda}(s,a)$, exactly. In Section 5, we consider the case where the Q-function can only be estimated with error and establish performance bounds that account for approximation errors.

Our analysis builds on mirror descent for the saddle point. First recall that the primal-dual update builds on strong duality. For CMDPs with finite state and action spaces, strong duality always holds (Theorem 3.6 in (Altman 1999)). However, when the state space is countably infinite, we need more regularity conditions to ensure strong duality. One sufficient condition is that the instantaneous costs of the CMDP are uniformly bounded from below (see Definition 7.1, Theorem 9.9, and Chapter 10.3 in (Altman 1999)):

Assumption 1. *[Lower Bound of Instantaneous Costs] There exists a constant W such that for all $s \in \mathcal{S}$, $a \in \mathcal{A}$, and $k = 1, 2, \dots, K$,*

$$c(s,a) > W, \quad d_k(s,a) > W.$$

To establish the convergence result, we also require Slater's condition:

Algorithm 1 Primal-Dual Algorithm to CMDPs

```
Input: pre-specified projection domain $\Lambda_M$, stepsizes $\{\eta_m\}_{m\geq 0}$, initial policy $\pi_0$ and Lagrangian multiplier $\lambda_0$
for $m=1,\dots,T-1$ do
    Update Lagrangian multipliers and policy as
        $$\begin{cases}\lambda_m = \mathrm{Proj}_{\Lambda_M}\Big\{\lambda_{m-1}+\eta_{m-1}\cdot\partial_\lambda L(\pi_{m-1},\lambda_{m-1})\Big\},\\ \pi_m(\cdot|s)\propto\pi_{m-1}(\cdot|s)\cdot\exp\Big\{-\eta_{m-1}\cdot Q^{\pi_{m-1},\lambda_{m-1}}(s,\cdot)\Big\}.\end{cases}$$
    where
        $$\big[\partial_\lambda L(\pi_{m-1},\lambda_{m-1})\big]_k = D_k(\pi_{m-1})-q_k,\ \forall k\in[K],$$
    and
    • under discounted cost,
        $$Q^{\pi,\lambda}(s,a):=(1-\gamma)\cdot\Big(c^\lambda(s,a)+\mathbb{E}^\pi\Big[\sum_{t=1}^{\infty}\gamma^t c^\lambda(s_t,a_t)\big|s_0=s,a_0=a\Big]\Big);$$
    • under long-run average cost,
        $$Q^{\pi,\lambda}(s,a)=c^\lambda(s,a)-C^\lambda(\pi)+\sum_{s'\in\mathcal{S}}V^{\pi,\lambda}(s')P(s'|s,a),$$
    where $V^{\pi,\lambda}$ satisfies the Poisson equation
        $$C^\lambda(\pi)+V^{\pi,\lambda}(s)=\sum_{a\in\mathcal{A}}\Big(c^\lambda(s,a)+\sum_{s'\in\mathcal{S}}V^{\pi,\lambda}(s')P(s'|s,a)\Big)\cdot\pi(a|s);$$
end for
Output: mixing policy
        $$\bar{\pi}_T=\sum_{m=0}^{T-1}\tilde{\eta}_m\pi_m,\ \text{where}\ \tilde{\eta}_m=\eta_m/\sum_{m=0}^{T-1}\eta_m.$$
```

Assumption 2. *[Slater's Condition] There exists some policy $\tilde{\pi}$ such that $\forall 1\leq k\leq K$,*

$$D_k(\tilde{\pi})<q_k.$$

Slater's condition ensures the existence of a finite and bounded optimal Lagrangian multiplier (Chapter 5.2 of Boyd et al. (2004))

$$\lambda^*=\arg\max_{\lambda\geq 0}\Big\{\inf_{\pi\in\mathcal{M}(\Pi_S)}L(\pi,\lambda)\Big\}.$$

This condition is commonly assumed in the constrained optimization literature (Nocedal and Wright 1999). For many practical problems, Slater's condition holds naturally.

Our last assumption is about the boundedness of "subgradient", which regularizes the movement of policies and Lagrangian multipliers at each iteration. Recall that in Algorithm 1, after applying subgradient ascent for the Lagrangian multiplier, we project λ onto a bounded domain Λ_M, which is defined as

$$\Lambda_M = \big\{\lambda \in \mathbb{R}_+^K : \|\lambda\| \le M + r\big\}, \tag{24}$$

where M is an upper bound of $\|\lambda^*\|$ and $r > 0$ is a slackness constant.

Assumption 3. *[Bounded Subgradient] There exists some constant $G > 0$ such that for any $\lambda \in \Lambda_M$ and policy $\pi \in \mathcal{M}(\Pi_S)$, the following inequalities hold*

$$\big\|\partial_\lambda L(\pi,\lambda)\big\| \le G, \qquad \sup_{s\in\mathcal{S}}\sup_{a\in\mathcal{A}}\big|Q^{\pi,\lambda}(s,a)\big| \le G.$$

Since $Q^{\pi,\lambda}(s,a)$ is linear in λ, it is necessary to restrict λ to a bounded domain Λ_M for Assumption 3 to hold. That is why we need the projection step in updating λ. Similar boundedness assumptions are commonly required for primal-dual algorithms (Nedić and Ozdaglar 2009, Le et al. 2019). Note that when the instantaneous cost functions $c(\cdot,\cdot)$ and $d_k(\cdot,\cdot)$ are uniformly bounded or when the state and action spaces are finite, Assumption 3 holds trivially.

Lastly, we comment that Slater's condition (Assumption 2) not only guarantees the existence and boundedness of λ^*, but also provides an explicit upper bound for $\|\lambda^*\|$. In particular, let $\tilde{\pi}$ be a Slater point (a policy that satisfies Slater's condition). Then, we have

$$\|\lambda^*\| \le -\frac{C(\tilde{\pi}) - \tilde{c}}{\max_{1\le k\le K}\big\{D_k(\tilde{\pi}) - q_k\big\}}, \tag{25}$$

where $\tilde{c} \le C(\pi^*)$ is an arbitrary lower bound for the dual problem.

To establish convergence, we need to construct an appropriate potential function for the policy update, which is also known as the distance function in mirror descent. Note that mirror descent can be viewed as minimizing a linear approximation of the objective function and a distance based penalty that prevents us from moving too far in each iteration.

Consider the *state* occupancy measure ν_s^π induced by a policy $\pi \in \Pi_S$, where the subscript s stands for *state*. For the discounted cost,

$$\nu_s^\pi(s) := (1-\gamma)\cdot \mathbb{E}_{s_0\sim\mu_0}\Big[\sum_{t=0}^{\infty}\gamma^t \bar{P}^\pi(s_t = s|s_0)\Big];$$

for the long-run average cost

$$\nu_s^\pi(s) = \lim_{T\to\infty}\frac{1}{T}\mathbb{E}^\pi\Big[\sum_{t=0}^{T-1} 1(s_t = s)|s_0\Big].$$

ν_s^π in the long-run average formulation is also known as the stationary distribution of the Markov chain under policy π. The KL-divergence between two stationary policies π_1 and π_2 weighted by ν_s^π is defined as

$$\Phi^\pi(\pi_1\|\pi_2) = \mathbb{E}_{s\sim\nu_s^\pi}\Big[\mathrm{KL}\big(\pi_1(\cdot|s)\|\pi_2(\cdot|s)\big)\Big]. \tag{26}$$

When π_1 and π_2 are mixing policies, we first transform them to the equivalent stationary policies via (14), and then define $\Phi^\pi(\pi_1\|\pi_2)$ as the weighted KL-divergence between the equivalent stationary policies. By definition, $\Phi^\pi(\pi_1\|\pi_2)$ measures the discrepancy between two policies weighted by a given state occupation measure. It connects the regularized policy iteration in (17), which is defined state-wise, with a single objective, and serves as the potential function in our mirror descent analysis. In our analysis, we use the weight corresponding to the state occupancy measure under the optimal policy. However, note that the weight does not affect the policy updates, since the regularized policy iteration is defined state-wise.

We are now ready to introduce the convergence result of our primal-dual algorithm. The violation of constraints is measured by

$$\big\|[D(\bar\pi_T)-q]^+\big\| := \left(\sum_{k=1}^K \big([D_k(\bar\pi_T)-q_k]^+\big)^2\right)^{1/2}. \tag{27}$$

Theorem 1. *(Convergence of Algorithm 1) Let $I_\gamma=(1-\gamma)$ for discounted cost criterion and $I_\gamma=1$ for long-run average cost criterion. Under Assumptions 1-3, if the step size $\eta_m=\Theta(1/\sqrt{m})$, then there exist positive constants κ_1 and κ_2, such that*

$$\big\|[D(\bar\pi_T)-q]^+\big\| \le \Big((M+r)^2+\frac{9}{8}G^2\kappa_2\log(T)+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{rI_\gamma\kappa_1\sqrt{T}},$$

and

$$C(\bar\pi_T)-L^* \le \Big(\frac{9}{8}G^2\kappa_2\log(T)+\Phi^{\pi^*}(\pi^*\|\pi_0)+\frac{\|\lambda_0\|^2}{2}\Big)\frac{1}{I_\gamma\kappa_1\sqrt{T}},$$
$$C(\bar\pi_T)-L^* \ge -\|\lambda^*\|\Big((M+r)^2+\frac{9}{8}G^2\kappa_2\log(T)+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{rI_\gamma\kappa_1\sqrt{T}},$$

where r is the slackness constant in (24). *If the step size is constant $\eta_m=\eta$, then*

$$\big\|[D(\bar\pi_T)-q]^+\big\| \le \Big((M+r)^2+\frac{1}{I_\gamma}\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{rT\eta}+\Big(1+\frac{1}{8I_\gamma}\Big)\frac{G^2\eta}{r}$$

and

$$C(\bar\pi_T)-L^* \le \Big(\frac{1}{I_\gamma}\Phi^{\pi^*}(\pi^*\|\pi_0)+\frac{\|\lambda_0\|^2}{2}\Big)\frac{1}{T\eta}+\frac{9G^2\eta}{8I_\gamma},$$
$$C(\bar\pi_T)-L^* \ge -\|\lambda^*\|\Big((M+r)^2+\frac{1}{I_\gamma}\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{rT\eta}-\|\lambda^*\|\Big(1+\frac{1}{8I_\gamma}\Big)\frac{G^2\eta}{r}.$$

In Theorem 1, the constants κ_1 and κ_2 are chosen such that

$$\sum_{m=0}^{T-1} \eta_m \geq \kappa_1 \sqrt{T} \text{ and } \sum_{m=0}^{T-1} \eta_m^2 \leq \kappa_2 \log(T).$$

Since $\sum_{m=0}^{T-1} 1/\sqrt{m} = O(\sqrt{T})$ and $\sum_{m=0}^{T-1} 1/m = O(\log T)$, such constants exist and only depend on η_m's, i.e., they do not depend on other model parameters. The proof of Theorem 1 is deferred to Appendix A.1.

Theorem 1 indicates that with decreasing stepsizes: $\eta_m = \Theta(1/\sqrt{m})$, our primal-dual algorithm achieves $O(\log(T)/\sqrt{T})$ convergence, i.e.,

$$\left\| [D(\bar{\pi}_T) - q]^+ \right\| = O(\log(T)/\sqrt{T}) \text{ and } |C(\bar{\pi}_T) - L(\pi^*, \lambda^*)| = O(\log(T)/\sqrt{T}).$$

For constant step sizes, $\eta_m = \eta$, our primal-dual algorithm converges to a neighborhood of the optimum at rate $O(1/T)$, i.e.,

$$\left\| [D(\bar{\pi}_T) - q]^+ \right\| = O(1/(\eta T) + \eta) \text{ and } |C(\bar{\pi}_T) - L(\pi^*, \lambda^*)| = O(1/(\eta T) + \eta)$$

In this case, if we set $\eta = \Theta(1/\sqrt{T})$, we achieve $O(1/\sqrt{T})$ convergence rate. These convergence rates match those in Le et al. (2019), which requires solving the modified unconstrained MDPs to optimal at each iteration.

We conclude this section by making a few comments about the bounds established in Theorem 1. First, it is unlikely to improve the convergence rate beyond $\Theta(1/\sqrt{T})$. This is because the dual is a finite-dimensional concave optimization problem without strong concavity. The convergence rate of the subgradient method in this case is lower bounded by $\Omega(1/\sqrt{T})$ (Bubeck 2014). Second, although the discount factor does not affect the convergence rate, it affects the constant that goes in front of the rate. In general, the larger the discount factor, the larger the constant, and the long-run average case tends to have a larger constant than the discounted case. (Note that our characterization of the constants is unlikely to be tight.) Third, although the slackness constant r appears in the denominators only, the constant G, which is an upper bound of the subgradients, grows linearly in r. In particular, by Assumption 3, G is determined by the shape of Λ_M. Hence, r cannot be set arbitrarily large. Second,

5. Extension to the Reinforcement Learning Setting

In Algorithm 1 and its performance analysis, we assume we can evaluate the Q-function exactly. However, in practice, this can be hard to achieve when we are dealing with a large state or action space, or in a model-free setting where the transition kernel is not known explicitly. In this section, we combine our algorithm with reinforcement learning (RL) techniques to deal with large-scale problems with unknown transition dynamics. We also establish convergence results that take the approximation errors into account. We focus on the online setting where we can interact with the environment to collect data (costs and transitions), but our algorithm can be combined with offline policy evaluation algorithms as well, in which only data generated by a behavior policy is available.

When dealing with a large state space, one commonly used approach is to approximate the Q-function and policy $\pi(a|s)$ by properly defined classes of parametric functions. For the policy, we consider parameterizing its energy function $f_\beta(s,a)$ where $\pi_\beta(s,a) \propto \exp\{f_\beta(s,a)\}$. We denote by $\mathcal{G} = \{Q_\alpha(s,a) : \alpha \in \mathbb{R}^m\}$ the parameterized Q-function class and $\mathcal{F} = \{f_\beta(s,a) : \beta \in \mathbb{R}^n\}$ the parameterized energy function class. For example, for linear parameterization, we assume $Q_\alpha(s,a) = \alpha^\top \phi(s,a)$ and $f_\beta(s,a) = \beta^\top \psi(s,a)$, where $\phi(s,a)$ and $\psi(s,a)$ are given feature functions. In this case, $\mathcal{F}$ and $\mathcal{G}$ are linear spaces generated by the feature functions. $\mathcal{G}$ and $\mathcal{F}$ can also be nonlinear function classes (nonlinear in α or β) such as those generated by deep neural networks.

In our primal-dual update, let π_{β_m} denote the policy obtained in the m-th iteration and λ_m denote the Lagrangian multiplier at iteration m. For the dual update, note that $D_k(\pi_{\beta_m})$'s can be accurately evaluated using standard Monte-Carlo simulation, and these evaluations in general do not suffer from the curse of dimensionality (we do not need to evaluate $D_k(\pi_{\beta_m})$ for each state-action pair). We update the Lagrangian multiplier as

$$\lambda_m = \mathrm{Proj}_{\Lambda_M}\{\lambda_{m-1} + \eta_{m-1} \cdot \partial_\lambda L(\pi_{\beta_{m-1}}, \lambda_{m-1})\}.$$

For the policy update, when evaluating the Q-function induced by policy $\pi_{\beta_{m-1}}$, we restrict to the function class $\mathcal{G}$ and consider minimizing the mean squared Bellman error (Sutton and Barto 2018):

$$\begin{aligned}\alpha_{m-1} = \arg\min_\alpha \mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}} \Big[\Big(Q_\alpha^{\lambda_{m-1}}(s,a) - \big((1-\gamma)c^{\lambda_{m-1}}(s,a) \\ + \gamma \mathbb{E}_{s'\sim P(\cdot|s,a)}[E_{a'\sim\pi_{\beta_{m-1}}(\cdot|s')}[Q_\alpha^{\lambda_{m-1}}(s',a')]]\big)\Big)^2\Big],\end{aligned}$$

where $\nu^{\pi_{\beta_{m-1}}}$ is the state-action occupancy measure induced by policy $\pi_{\beta_{m-1}}$. Note that when $Q_\alpha^{\lambda_{m-1}} = Q^{\pi_{\beta_m},\lambda_{m-1}}$, the Bellman error is equal to zero. The above minimization problem can be (approximately) solved using standard TD-learning algorithms. We provide one such algorithm in Appendix C. Given the approximated Q-function $Q_{\alpha_{m-1}}^{\lambda_{m-1}}$, we next update the policy. If we apply the regularized policy iteration directly, we have

$$\pi(s,a) \propto \pi_{\beta_{m-1}}(s,a) \cdot \exp\big\{-\eta_{m-1} \cdot Q_{\alpha_{m-1}}^{\lambda_{m-1}}(s,a)\big\}.$$

This suggests that we would like to update β_m such that $f_{\beta_m} = f_{\beta_{m-1}} - \eta_{m-1} \cdot Q_{\alpha_{m-1}}^{\lambda_{m-1}}$. When $\mathcal{G}$ is a linear space, we can set $\mathcal{F} = \mathcal{G}$ and we have $f_{\beta_{m-1}} - \eta_{m-1} \cdot Q_{\alpha_{m-1}}^{\lambda_{m-1}} \in \mathcal{F}$, which implies that $f_{\beta_{m+1}} =$

$f_{\beta_{m-1}} - \eta_{m-1} \cdot Q^{\lambda_{m-1}}_{\alpha_{m-1}}$. However, when $\mathcal{F}, \mathcal{G}$ are nonlinear in their parameters, $f_{\beta_{m-1}} - \eta_{m-1} \cdot Q^{\lambda_{m-1}}_{\alpha_{m-1}}$ may no longer fall into $\mathcal{F}$. In this case, we can project the ideal policy update onto $\mathcal{F}$ by minimizing the mean-squared error:

$$\beta_m = \arg\min_{\beta} \mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}} \left[\left(f_\beta(s,a) - (f_{\beta_{m-1}}(s,a) - \eta_{m-1} \cdot Q^{\lambda_{m-1}}_{\alpha_{m-1}}(s,a)) \right)^2 \right].$$

In this step, we can also define the projection based on other metrics, but to simplify the analysis, we focus on the weighted L_2 norm. We summarize our primal-dual algorithm with function approximations in Algorithm 2.

Algorithm 2 Primal-Dual Algorithm with Function Approximation

```
Input: pre-specified projection domain $\Lambda_M$, stepsizes $\{\eta_m\}_{m\geq 0}$, initial policy $\pi_{\beta_0}$ and Lagrangian multiplier $\lambda_0$
for $m = 1, \ldots, T-1$ do
    Evaluate the subgradient
        $\left[\partial_\lambda L(\pi_{\beta_{m-1}}, \lambda_{m-1})\right]_k = D_k(\pi_{\beta_{m-1}}) - q_k, \ \forall k \in [K].$
    Update Lagrangian multipliers as
        $\lambda_m = \mathrm{Proj}_{\Lambda_M}\left\{\lambda_{m-1} + \eta_{m-1} \cdot \partial_\lambda L(\pi_{\beta_{m-1}}, \lambda_{m-1})\right\}.$
    Update $\alpha_{m-1}$ as
        $\alpha_{m-1} = \arg\min_{\alpha} \mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}} \left[\left(Q^{\lambda_{m-1}}_{\alpha}(s,a) - \left((1-\gamma)c^{\lambda_{m-1}}(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[\mathbb{E}_{a'\sim\pi_{\beta_{m-1}}(\cdot|s')}[Q^{\lambda_{m-1}}_{\alpha}(s',a')]]\right)\right)^2\right].$
    Update $\beta_m$ as
        $\beta_m = \arg\min_{\beta} \mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}} \left[\left(f_\beta(s,a) - (f_{\beta_{m-1}}(s,a) - \eta_{m-1} \cdot Q^{\lambda_{m-1}}_{\alpha_{m-1}}(s,a))\right)^2\right].$
end for
Output: mixing policy
    $\bar{\pi}_T = \sum_{m=0}^{T-1} \tilde{\eta}_m \pi_{\beta_m}, \text{ where } \tilde{\eta}_m = \eta_m / \sum_{m=0}^{T-1} \eta_m.$
```

In Algorithm 2, due to parameterization, the approximated Q-function, $Q^{\lambda_{m-1}}_{\alpha_{m-1}}$, can be different from the true Q-function $Q^{\pi_{\beta_{m-1}}, \lambda_{m-1}}$. We next take the approximation errors into account and study the convergence of Algorithm 2. We start by introducing some assumptions. Our first assumption is an analog to Assumption 3.

ASSUMPTION 4. *[Bounded Approximated Subgradient] There exists a finite constant $G > 0$ such that for any $\lambda \in \Lambda_M$, policy $\pi \in \mathcal{M}(\Pi_S)$,*

$$\sup_{s\in\mathcal{S}}\sup_{a\in\mathcal{A}} \left|Q_{\alpha}^{\lambda}(s,a)\right| \le G,$$

where Q_{α}^{λ} is the best approximation of $Q^{\pi,\lambda}$.

Let the policy sequence generated by Algorithm 2 be $(\pi_{\beta_0}, \pi_{\beta_1}, \pi_{\beta_2}, \cdots)$ and the approximated Q-function sequence be $(Q_{\alpha_0}^{\lambda_0}, Q_{\alpha_1}^{\lambda_1}, Q_{\alpha_2}^{\lambda_2}, \cdots)$. The next assumption assumes that at every iteration, the Q-function can be approximated well enough.

ASSUMPTION 5. *[Upper Bound for Q-function Approximation Error] There exists a constant $\epsilon > 0$ such that for $m = 0, 1, 2, \cdots$, the approximated Q-function $Q_{\alpha_{m-1}}^{\lambda_{m-1}}$ obtained in Algorithm 2 satisfies*

$$\mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}}\left[\left|Q_{\alpha_{m-1}}^{\lambda_{m-1}}(s,a) - Q^{\pi_{\beta_{m-1}},\lambda_{m-1}}(s,a)\right|\right] \le \epsilon.$$

Note that the approximation error can be affected by the richness of the function class $\mathcal{G}$ and the sample size used for policy evaluation. When using fitted Q-evaluation, to achieve an ϵ-approximation accuracy, the required sample size is $O(\epsilon^{-2})$ (Theorem 4.2 in Le et al. (2019)).

Our next assumption requires that parametrized policy space $\mathcal{F}$ is rich enough such that we can always find an f_β that approximates the regularized policy iteration well.

ASSUMPTION 6. *[Richness of the Parametrized Policy Space] There exists a constant $\delta > 0$ such that for $m = 0, 1, 2, \cdots$, β_m satisfies*

$$\mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_{m-1}}}}\left[\left|f_{\beta_m}(s,a) - (f_{\beta_{m-1}}(s,a) - \eta_{m-1}\cdot Q_{\alpha_{m-1}}^{\lambda_{m-1}}(s,a))\right|\right] \le \delta.$$

Note that Assumption 6 holds trivially with $\delta = 0$ when $\mathcal{G}$ is a linear space and $\mathcal{F} = \mathcal{G}$.

Lastly, we impose a uniform upper bound for likelihood ratios of the occupancy measures for technical tractability.

ASSUMPTION 7. *[Bounded Likelihood Ratios] There exists a finite constant C_ℓ, such that for $m = 0, 1, 2, \cdots$,*

$$\left\|\frac{d\nu^{\pi^*}}{d\nu^{\pi_{\beta_m}}}\right\|_\infty, \left\|\frac{d\nu_s^{\pi^*}}{d\nu_s^{\pi_{\beta_m}}}\right\|_\infty, \left\|\frac{d\nu^{\pi_{\beta_{m+1}}}}{d\nu^{\pi_{\beta_m}}}\right\|_\infty \le C_\ell.$$

Assumption 7 requires that the occupancy measures at different iterations are not more concentrated than the occupancy measure under the optimal policy. In other words, the intermediate policies π_{β_m} should be exploratory enough. This is required because when running Algorithm 2 and analyzing the approximation errors, we work with $\nu^{\pi_{\beta_m}}$, while the target objective and constraints are with respect to ν^{π^*}. Assumption 7 allows us to apply proper change-of-measures to connect the two. This assumption is also imposed in Liu et al. (2019), which applies mirror descent to analyze the convergence of Trust Region Policy Optimization. Similar assumptions such as finite concentrability coefficients are commonly used in the literature (Munos and Szepesvári 2008, Antos et al. 2007, Farahmand et al. 2016).

With the above assumptions, we are ready to state the performance guarantee for Algorithm 2. To highlight the key insights, we focus on the dependency on the stepsize η, iteration number T, and approximation errors ϵ, δ.

Theorem 2. *(Convergence of Algorithm 2) Under Assumptions 1 – 7, when the stepsize is a constant, $\eta_m = \eta$, the output of Algorithm 2 $\bar{\pi}_T$ satisfies*

$$\left\|[D(\bar{\pi}_T) - q]^+\right\|, \left|C(\bar{\pi}_T) - L^*\right| \leq O\Big(\eta + \frac{1}{\eta T} + \epsilon + \frac{\delta}{\eta}\Big).$$

The proof of Theorem 2 is in Appendix A.2. Note that compared with Theorem 1, the performance bounds in Theorem 2 have two extra terms: ϵ and δ/η. The first one corresponds to the approximation error of the Q-function; the second one is due to policy projection. Theorem 2 shows that, with a constant stepsize, Algorithm 2 converges to a neighborhood of the optimal at rate $1/T$. The size of the neighborhood is affected by the stepsize η and two approximation errors: ϵ and δ. However, it does not depend on T, which implies that the errors do not accumulate over iterations. When setting $\eta = \Theta(1/\sqrt{1/T + \delta})$, we achieve the performance bound $O(\sqrt{1/T + \delta} + \epsilon)$. When $\delta = 0$ (e.g., when using linear parameterization with $\mathcal{F} = \mathcal{G}$), we achieve $O(1/\sqrt{T})$ convergence.

Lastly, we comment that convergence rate is the same whether the instantaneous costs are random or deterministic. When instantaneous costs are random, $c(s, a)$ and $d_k(s, a)$'s are the expected costs. The randomness in the instantaneous costs may affect the constant term in front of the convergence rate of the algorithm and the sample size required when using Monte Carlo simulation to estimate the Q-function. In addition, when using sampling-based method to estimate the Q-function, a higher discount factor would in general lead to a higher sampling variance due to a slower mixing rate of the underlying Markov chain (Dai and Gluzman 2020).

6. Application to Queue Scheduling

In this section, we apply our primal-dual algorithm to some queue scheduling problems motivated by healthcare applications.

6.1. ED Scheduling Problem

We first consider a relatively simple problem that is motivated by the hospital emergency department (ED) and has been studied in (Girard et al. 2020) analytically. ED patients are in general classified into different urgency/severity levels. It is important to ensure that urgent patients can receive treatment in a timely manner to avoid adverse consequences. At the same time, it is also important to properly manage the waiting time of non-urgent patients, especially when these patients may leave the hospital without being seen. It is in general difficult to define/quantify the relative weights between the waiting cost of urgent patients versus non-urgent patients. Thus, following (Girard et al. 2020), we introduce a CMDP formulation for the scheduling problem where we try to minimize the waiting time of one class while making sure that the waiting time of the other class is below a certain threshold.

We consider a single server queue with two classes of patients, 1 and 2. Patients arrive to the system according to Poisson processes with rates $\theta_1 = 1$ and $\theta_2 = 0.7$ respectively. The service times are exponentially distributed with rates $\mu_1 = 2$ and $\mu_2 = 1.5$ respectively. The queue length is truncated at 10 for each class. The goal is to minimize the long-run average queue length of class 1 patients (which is equivalent to minimizing the long-run average waiting time) subject to the constraint that the long-run average queue length of class 2 patients does not exceed the threshold $q = 1$. Preemption is allowed. We apply Algorithm 1, where we use LP to calculate the relative Q-function exactly for each updated policy. The exact optimal cost for this problem is 3.69. We use a constant stepsize of 0.25. After 20 iterations, the policy learned by our primal-dual algorithm achieves a cost of 3.91 (optimality gap = 5%); after 100 iterations, the cost reduces to 3.74 (optimality gap = 1%); after 5000 iterations, the cost further reduces to 3.70 (optimality gap = 0.2%). Figure 1 depicts the evolution of policies (the probability of serving a class-1 customer given the current state) learned by our primal-dual algorithm and the optimal policy. We observe that our primal-dual algorithm indeed learns the optimal policy.

6.2. Inpatient-flow Management Problem

In this section, we consider a more complicated queue scheduling problem where exact evaluation of the relative Q-function is practically infeasible. The problem is motivated by hospital inpatient-flow management (Dai and Shi 2019, Song et al. 2020, Dong et al. 2019).

Many hospitals partition inpatient ward beds into specialized units. Patients from different specialties in principle should receive treatment in their corresponding specialty wards, which are also known as the primary wards. However, due to a high level of occupancy and high uncertainty in patients' demand, the primary wards may be at or near full capacity from time to time. In these situations, to avoid excessive admission delays, the hospital may choose to place a patient in some

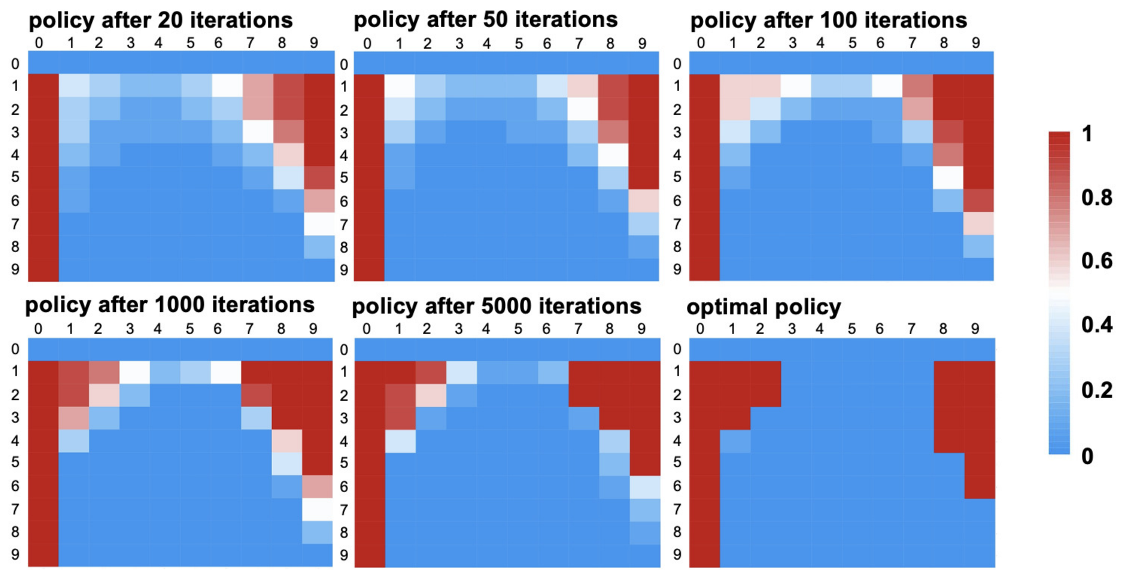


Figure 1 **Policy learned versus the optimal policy. The horizon axis represents the number of class 2 patients in the system and the vertical axis is the number of class 1 patients in the system. Different colors represent different probabilities of prioritizing the class 1 patients.**

non-primary ward (overflow). While placing patients in non-primary wards can help reduce admission delay and balance the loads between different wards, there are also costs associated with it. Most noticeably, overflow can lead to worse clinical outcomes (Song et al. 2020). It also imposes additional inconvenience costs for nurses and physicians.

We model the inpatient-flow dynamics as an $I \times I$ queueing network where there are I classes of customers (patients) and correspondingly I pools of primary servers. Pool i has N_i homogeneous servers. We consider a discrete time model. In each period, the number of class i arrivals follows a Poisson distribution with rate θ_i. The service time provided by a server from pool i follows a geometric distribution with success probability μ_i. For each class i customer waiting in the queue, we incur a holding cost of h_i per unit time. For each class i customer being overflowed to a non-primary pool j with $j \neq i$, we incur an overflow cost of r_{ij}. Our goal is to minimize the long-run average holding cost under the constraint that the long-run average overflow penalty does not exceed a certain threshold. We focus on non-preemptive scheduling policies, i.e., once a customer is assigned to a server, s/he stays there until service completion. Let $X_i(t)$ denote the number of class i customers in the queue at time period t and $U_{ij}(t)$ denote the number of class i customers newly assigned to pool j at t. Then, the scheduling problem takes the form

$$\min_{\pi} \lim_{T\to\infty} \frac{1}{T} \cdot \mathbb{E}^{\pi}\left[\sum_{t=0}^{T-1}\sum_{i=1}^{I} h_i X_i(t)\right] \tag{28}$$

$$\text{s.t.} \lim_{T\to\infty} \frac{1}{T} \cdot \mathbb{E}^{\pi} \left[\sum_{t=0}^{T-1} \sum_{i=1}^{I} \sum_{j=1}^{I} r_{ij} U_{ij}(t) \right] \le q.$$

Note that we put overflow penalty as a constraint rather than part of the objective, because it is hard to quantify how the cost of waiting compares to the cost of overflow.

When applying our primal-dual algorithm to solve (28), the size of the state-action space grows very fast as I or N_i grows. It can be computationally prohibitive to evaluate the relative action-value function $Q^{\pi,\lambda}(s,a)$ exactly. Indeed, even with $I = 3$, $N_i = 10$, if we truncate the queue at 30 for each class, we already have to evaluate $O(10^9)$ different state-action pairs. To overcome this, we use value function approximation with the quadratic basis and apply the least-square temporal difference algorithm to estimate the optimal coefficients (see Appendix B for more details).

For this class of problems, the optimal policy is not known. Thus, we compare the performance of the policy learned using our algorithm to some benchmark policies developed in the literature. When considering the holding cost alone, one well-known policy is called the $c\mu$-rule. The $c\mu$-rule or generalized versions of it have been shown to be asymptotically optimal in many parallel server systems (PSS) (see, for example, Mandelbaum and Stolyar (2004)). Another important policy is called the max-pressure policy, which is known to be throughput optimal for many queueing networks (Dai et al. 2008). Overflow costs have been much less studied in the literature (see, for example, Dai and Shi (2019)). To account for the overflow constraint, we propose the following modified versions of the $c\mu$-rule and max-pressure policy, respectively. Let Θ be a threshold for the overflow cost per period, which is a tuning parameter. Then, for each t, consider $U_{ij}(t)$ that solves

$$\max \sum_{i=1}^{I} \sum_{j=1}^{I} \omega_{ij}(t) U_{ij}(t)$$

$$\text{s.t.} \sum_{i=1}^{I} \sum_{j=1}^{I} r_{ij} U_{ij}(t) \le \Theta, \quad \sum_{j=1}^{I} U_{ij}(t) \le X_i(t), \sum_{i=1}^{I} Z_j(t) + U_{ij}(t) \le N_j, U_{ij}(t) \ge 0,$$

where $Z_j(t)$ is the number of patients in pool j before the new assignment in period t. When $\omega_{ij}(t) = h_i \mu_j$, we have the modified $c\mu$-rule. When $\omega_{ij}(t) = h_i X_i(t) \mu_j$, we have the modified max-pressure policy. We also consider a mixture of modified $c\mu$-rules and modified max-pressure policies with different values of Θ. Suppose there are K policies. The mixture weight p solves

$$\min_p \sum_{k=1}^{K} p_k C(\pi_K) \quad \text{s.t.} \sum_{k=1}^{K} p_k D(\pi_k) \le q, \quad \sum_{k=1}^{K} p_k = 1, \; p_k \ge 0, \tag{29}$$

where $C(\pi_k)$ is the long-run average holding cost of the k-th policy in the mixture and $D(\pi_k)$ is its long-run average overflow cost.

We first consider a 2×2 network with 10 servers in each server pool. We truncate the queue at 30 for each class. Two scenarios are tested. In the first scenario, we set arrival rates $\theta_1 = 8, \theta_2 = 3.5$ and service rates $\mu_1 = \mu_2 = 0.6$, such that the first class is overloaded and the second class is underloaded. The holding costs are $h_1 = 3, h_2 = 1$ and the overflow costs are $r_{12} = r_{21} = 1$. The budget for long-run average overflow penalty is $q = 1.5$. In the second scenario, we consider a symmetric system with arrival rates $\theta_1 = \theta_2 = 5.5$, service rates $\mu_1 = \mu_2 = 0.6$, holding costs $h_1 = h_2 = 2$, and overflow costs $r_{12} = r_{21} = 1$. The budget for long-run average overflow penalty is $q = 0.3$.

The performance of different policies is summarized in Tables 1 and 2 for the two scenarios respectively. The "optimal mixing" column denotes the mixing policy that solves (29). Comparing the policy learned by our primal-dual algorithm (within 100 iterations) to the best-performing benchmark policy – optimal mixing, our policy achieves a 18% cost reduction in scenario 1 (see Table 1) and an 12% cost reduction in scenario 2 (see Table 2).

Table 1 **Performance of different policies for a 2×2 queuing network scheduling problem, scenario 1**

	benchmark type	$\Theta = 1$	$\Theta = 2$	$\Theta = 3$	$\Theta = 4$	optimal mixing	primal-dual
holding cost	cμ	60.9	45.9	31.2	24.2	39.6	**33.0**
	max-pressure	60.8	46.2	32.1	24.5		
overflow cost	cμ	0.87	1.51	1.83	1.96	1.50	1.50
	max-pressure	0.87	1.52	1.84	1.96		

Table 2 **Performance of different policies for a 2×2 queueing network scheduling problem, scenario 2**

	benchmark type	$\Theta = 0$	$\Theta = 1$	$\Theta = 2$	$\Theta = 3$	optimal mixing	primal-dual
holding cost	cμ	19.2	12.3	10.4	9.4	12.7	**11.4**
	max-pressure	18.7	12.1	10.2	9.5		
overflow cost	cμ	0	0.33	0.50	0.56	0.30	0.30
	max-pressure	0	0.33	0.49	0.56		

We also test a 3×3 network with 10 servers in each server pool. The queue for each class is truncated at 30. The system parameters are set as $(\theta_1, \theta_2, \theta_3) = (2, 4, 10)$, $(\mu_1, \mu_2, \mu_3) = (0.4, 0.5, 0.6)$, $(h_1, h_2, h_3) = (1, 2, 3)$ $r_{12} = r_{21} = r_{31} = 1$, and $r_{13} = r_{23} = r_{32} = 2$. The overflow penalty budget is $q = 2$. Note that if we are to evaluate the transition matrix $P(\cdot|s,a)$ in this case, we have $O(10^9)$ different (s,a) pairs. However, in our primal-dual algorithm, we only generate $O(10^6)$ samples in total. The performance of different policies is summarized in Table 3. The policy learned by our primal-dual algorithm (within 15 iterations) achieves a cost reduction of 5% compared with the best benchmark policy – optimal mixing.

7. Application to Inventory Planning

In this section, we apply the primal-dual algorithm to solve the multi-product multi-period inventory management problems (we refer to Turken et al. (2012) for a comprehensive review of these problems).

Table 3 **Performance of different policies for a 3×3 queueing network scheduling problem**

	benchmark type	$\Theta=1$	$\Theta=2$	$\Theta=3$	$\Theta=4$	optimal mixing	primal-dual
holding cost	cμ	98.2	94.8	91.6	88.7	92.9	**88.5**
	max-pressure	98.3	94.9	91.8	88.4		
overflow cost	cμ	0.93	1.78	2.40	2.90	2.0	2.0
	max-pressure	0.93	1.78	2.39	2.91		

7.1. Weakly coupled CMDP

One important feature of many multi-product inventory management problems is a weakly coupled structure, i.e., the problem can almost be decomposed into I sub-problems (I is often the number of products) except for a linkage constraint (e.g., a finite storage space) that links these sub-problems together (Turken et al. 2012). We consider weakly couple CMDPs (Singh and Cohn 1998), which consists of I sub-problems $\{(\mathcal{S}^i, \mathcal{A}^i, P^i, c^i(\cdot,\cdot), \gamma, \mu_0^i)\}_{i\in[I]}$ with the following properties:

P1) The state and action spaces can be expressed in the form of Cartesian products, i.e.,

$$\boldsymbol{s} = (s^1, \ldots, s^I),\ \boldsymbol{\mathcal{S}} = \mathcal{S}^1 \times \mathcal{S}^2 \times \ldots \times \mathcal{S}^I,\ \boldsymbol{a} = (a^1, \ldots, a^I),\ \boldsymbol{\mathcal{A}} = \mathcal{A}^1 \times \mathcal{A}^2 \times \ldots \times \mathcal{A}^I.$$

P2) For each state $\boldsymbol{s}_t$ and action $\boldsymbol{a}_t$, the instantaneous cost and auxiliary costs admit an additively separable form

$$c(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{i=1}^{I} c^i(s_t^i, a_t^i),\ d(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{i=1}^{I} d^i(s_t^i, a_t^i).$$

P3) The joint initial distribution satisfies $\boldsymbol{\mu}_0(\boldsymbol{s}) = \mu_0^1(s^1) \cdot \mu_0^2(s^2) \cdot \ldots \cdot \mu_0^I(s^I)$ and the one-step transition dynamic is of the form

$$P(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, \boldsymbol{a}_t) = \prod_{i=1}^{I} P^i(s_{t+1}^i | s_t^i, a_t^i).$$

For the multi-product inventory management problem, when demands for different products are independent, it naturally has a decomposable structure. However, constraints on a common storage space or aggregated service quality measures create linkages across the products.

Our primal-dual algorithm can be easily adapted to allow decomposability for weakly coupled CMDPs. We refer to a policy $\boldsymbol{\pi}$ as decomposable if it takes the product form $\boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{s}) = \prod_{i=1}^{I} \pi^i(a^i|s^i)$, i.e., the action for each product only depends on the inventory level of that product. Since our algorithm converges with any initial policy, we shall start with a decomposable policy. Then, when applying the primal-dual algorithm, the policies obtained in all subsequent iterations are decomposable. To see this, let $\{\boldsymbol{s}_t\}_{t\geq 0} = \{(s_t^1, \ldots, s_t^I)\}_{t\geq 0}$ and $\{\boldsymbol{a}_t\}_{t\geq 0} = \{(a_t^1, \ldots, a_t^I)\}_{t\geq 0}$ be the trajectory of the CMDP under a decomposable policy $\boldsymbol{\pi} = (\pi^1, \ldots, \pi^I)$. For each $i \in [I]$, we define

$$C^i(\pi^i) = (1-\gamma) \cdot \mathbb{E}_{s_0^i \sim \mu_0^i}^{\pi_i} \Big[\sum_{t=0}^{\infty} \gamma^t \cdot c^i(s_t^i, a_t^i) \big| s_0^i \Big],\ D^i(\pi^i) = (1-\gamma) \cdot \mathbb{E}_{s_0^i \sim \mu_0^i}^{\pi_i} \Big[\sum_{t=0}^{\infty} \gamma^t \cdot d^i(s_t^i, a_t^i) \big| s_0^i \Big].$$

Then, the objective and constraints in the CMDP become $\sum_{i=1}^{I} C^i(\pi^i)$ and $\sum_{i=1}^{I} D^i(\pi^i) \leq q$, and the Lagrangian is

$$L(\boldsymbol{\pi}, \lambda) = \sum_{i=1}^{I} \left(C^i(\pi^i) + \lambda^\top D^i(\pi^i) \right) - \lambda^\top q.$$

Hence, the Q-function is also decomposable,

$$Q^{\boldsymbol{\pi}_m, \lambda}(\boldsymbol{s}, \cdot) = \sum_{i=1}^{I} Q^{\pi_m^i, \lambda}(s^i, \cdot), \tag{30}$$

where $Q^{\pi_m^i, \lambda}(\cdot, \cdot)$ is the Q-function of the i-th modified sub-MDP with instantaneous cost $c^i(\cdot, \cdot) + \lambda^\top d^i(\cdot, \cdot)$. (We ignore $\lambda^\top q$ here because subtracting a common constant from the Q-functions does not change the regularized policy iteration.) This further indicates that the regularized policy iteration, including policy evaluation, can be implemented for each sub-problem in parallel via

$$\pi_{m+1}^i(\cdot|s^i) \propto \pi_m^i(\cdot|s^i) \cdot \exp\left\{ -\eta_m \cdot Q^{\pi_m^i, \lambda}(s^i, \cdot) \right\}, \ \forall i \in [I].$$

Moreover, for the subgradient of the Lagrangian multiplier, $\partial_\lambda L(\boldsymbol{\pi}_m, \lambda) = \sum_{i=1}^{I} D^i(\pi_m^i) - q$, $D^i(\pi_m^i)$'s can be evaluated in parallel as well. As a result, the primal-dual algorithm improves the computational complexity from exponential dependence on I to linear dependence on I.

We remark that a weakly coupled CMDP can also be viewed as a relaxation of a weakly coupled MDP, where we replace the hard (path-by-path almost sure) constraints in the MDP with expectation constraints. It has been shown that the optimal cost of the relaxed problems provides a lower bound for that of the original MDP (Adelman and Mersereau 2008). Our primal-dual algorithm provides an efficient way to solve the weakly coupled CMDP. However, how to translate the optimal policy for the relaxed problem to a good MDP policy that satisfies all the hard constraints is not studied here and would be an interesting future research direction (see Topaloglu and Kunnumkal (2006), Bertsimas and Mišić (2016), Brown and Smith (2020) for some recent developments).

7.2. Multi-Product Newsvendor Problem with a Storage Space Constraint

In this section, we study a relatively simple multi-period newsvendor problem with I products. At the beginning of each period, we need to decide the quantities to order based on the current inventory levels. New orders are filled immediately, and after the inventory is replenished, a random demand is realized. Excess supply (inventory) or demand (backlog) is carried to the next period with certain costs. We assume that demands for different products are independent and they are identically distributed across different periods. For each product $i \in [I]$, we denote its inventory level at the beginning of period t by s_t^i, the quantity we order as a_t^i, and the demand in period t as w_t^i. If the demand does not exceed the current inventory level, i.e., $w_t^i \leq s_t^i + a_t^i$, all the demand is fulfilled and the remaining inventory, $(s_t^i + a_t^i - w_t^i)$, is carried to the next period. Otherwise, only $s_t^i + a_t^i$ units are met in the current period. The remaining $(w_t^i - s_t^i - a_t^i)$ units are carried to the next period as backlog, i.e., we allow s_t^i to be negative to represent backlog. For product i, inventory incurs a holding cost of h_i per unit per period and backlog incurs a backlog cost of b_i per unit per period. In addition, each unit in inventory consumes v_i units of storage space.

Our goal is to minimize the cumulative discounted holding and backlog costs, while maintaining a certain constraint on the total storage space. We model the storage space constraint as a soft constraint that only needs to be satisfied on average, because in practice, temporary extra storage space may be easy to obtain. Following the CMDP formulation, we have

$$\begin{aligned} \min\ & (1-\gamma)\cdot\mathbb{E}_{\nu_0}\Big[\sum_{t=0}^{\infty}\sum_{i=1}^{I}\gamma^t[h_i(s_t^i+a_t^i-w_t^i)^+ + b_i(w_t^i-s_t^i-a_t^i)^+]\Big] \\ \text{s.t.}\ & (1-\gamma)\cdot\mathbb{E}_{\nu_0}\Big[\sum_{t=0}^{\infty}\sum_{i=1}^{I}\gamma^t v_i(s_t^i+a_t^i)^+\Big] \le q. \end{aligned} \tag{31}$$

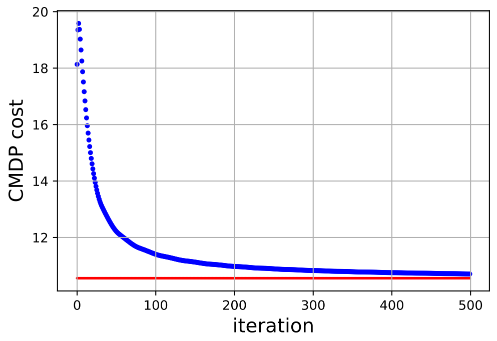
This CMDP is weakly coupled with state $\boldsymbol{s}=(s^1,\ldots,s^I)$, action $\boldsymbol{a}=(a^1,\ldots,a^I)$, and transition dynamics $s_{t+1}^i = s_t^i + a_t^i - w_t^i, \forall i \in [I]$. To see this, note that as the demands are independent across products, i.e., $P(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t,\boldsymbol{a}_{t+1}) = \prod_{i=1}^I P(s_{t+1}^i|s_t^i,a_{t+1}^i)$. The instantaneous cost function and auxiliary cost function are

$$c(\boldsymbol{s}_t,\boldsymbol{a}_t) = \sum_{i=1}^{I} h_i\cdot(s_t^i+a_t^i-w_t^i)^+ + b_i\cdot(w_t^i-s_t^i-a_t^i)^+,\ d(\boldsymbol{s}_t,\boldsymbol{a}_t) = \sum_{i=1}^{I} v_i\cdot(s_t^i+a_t^i)^+,$$

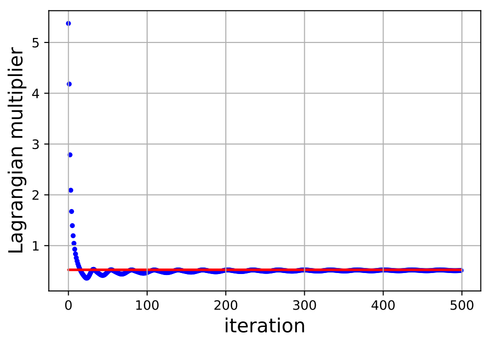
which are additively separable.

We consider a small-scale setting with $I=2$, and the demands for the two products are both uniformly distributed on $\{1,2,\ldots,10\}$. We also truncate the state space at $[-10,10]\times[-10,10]$, i.e., the inventory level cannot go above 10 and the backlog level can not go below -10. For the backlog, when it drops below -10, the excess demands are lost without incurring any cost. For other systems parameters, we set the holding costs $h_1=1, h_2=2$, backlog costs $b_1=2, b_2=3$, storage space requirement $v_1=1.5, v_2=1$, storage space threshold $q=10$, and discount rate $\gamma=0.75$.

When implementing the primal-dual algorithm, we can use LP to evaluate the Q-function exactly under a given policy. We implement two types of stepsize: a constant stepsize of 0.5 and decreasing stepsize with $\eta_m = 0.5/\sqrt{m+1}$. We run 2000 iterations in total and calculate the objective values at different iterations. Figure 2 plots averaged objective values and Lagrangian multipliers at different iterations when we use a constant step size. We observe that after 2000 iterations, the averaged CMDP cost is 10.55, which is close to the optimal value of 10.50. The averaged Lagrangian multiplier is 0.523, which is again close to the optimal value is 0.517. Figure 3 shows the relationships between $\sum_{t=0}^{T-1}\tilde{\eta}_t C(\pi_t)$ and the reciprocal of the number of iterations for constant stepsizes, and the reciprocal of the square root of the number of iterations for decreasing stepsizes. In both cases, we see straight lines, which confirm the convergence rates established in Theorem 1.

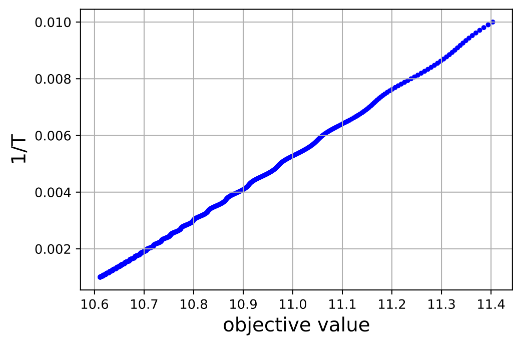


(a) $\sum_{t=0}^{T-1} \tilde{\eta}_t C(\pi_t)$

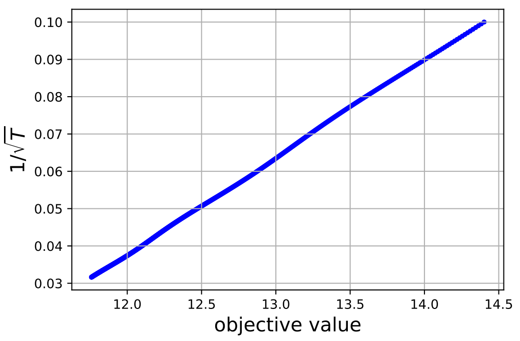


(b) $\sum_{t=0}^{T-1} \tilde{\eta}_t \lambda_t$

Figure 2 Trajectories of the objective and Lagrangian multiplier when $\eta_m = 0.5$



(a) $\eta_m = 0.5$



(b) $\eta_m = 0.5/\sqrt{m+1}$

Figure 3 Convergence rate with constant and decreasing step sizes

7.3. Multi-Product Newsvendor Problem with Perishable Goods and Stochastic Lead Time

In this section, we consider a large-scale newsvendor problem with perishable goods and stochastic lead times. In particular, unsold goods need to be discarded when their shelf life expires. When placing new orders, the replenished goods may arrive after a non-zero and random amount of time. These two features substantially enlarge the state space of the problem and the optimal replenishment and stocking policies are largely unknown.

Assume that there are N unique products. For each product, the maximal shelf life is $\mathcal{P}$, and the maximal lead time is $\mathcal{L}$. We denote the state of the i-th product in period t as

$$s_t^i = (I_t^i(1), \cdots, I_t^i(\mathcal{P}), O_t^i(1), \cdots, O_t^i(\mathcal{L})),$$

where $I_t^i(j)$ denotes the inventory level of product i that will expire in j periods, $j = 1, \cdots, \mathcal{P}$; $O_t^i(k)$ denotes the order quantity of product i that was placed k periods ago and has not arrived yet, $k = 1, \cdots, \mathcal{L}$. If the order placed k periods ago arrived before t, then $O_t^i(k) = 0$. The lead times of different products are independent and have the same distribution $(p_0, p_1, \cdots, p_{\mathcal{L}})$, where p_k denotes the probability that the order placed in period t will arrive in period $t+k$, $k = 0, 1, \cdots, \mathcal{L}$.

For the transition dynamics, at the beginning of period t, we first decide the quantity of new orders $a_t = (a_t^1, \ldots, a_t^N)$ based on the state $s_t = (s_t^1, \ldots, s_t^N)$. Second, for each product i, we update the inventory levels due to newly delivered orders. Let $\bar{p}_k = p_k / \sum_{j=k}^{\mathcal{L}} p_j$, $k = 0, \cdots, \mathcal{L}$, i.e., it is the probability that an order placed in period $t-k$ which has not arrived yet will arrive in period t. Let E_k^i denote a Bernoulli random variable with success probability $\bar{p}_k$. Then,

$$\tilde{I}_t^i(\mathcal{P}) = I_t^i(\mathcal{P}) + \sum_{k=1}^{\mathcal{L}} E_k^i \cdot O_t^i(k) + E_0^i \cdot a_t^i, \text{ and } \tilde{I}_t^i(k) = I_t^i(k), k = 1, \cdots, \mathcal{P}-1.$$

Third, for each product i, we further update the inventory level based on the newly arrived demand D_t^i that is drawn from the distribution $\mathcal{D}_i$. When filling the demand, we prioritize goods with shorter remaining shelf life. Algorithmically, we initialize the remaining demand $R^i = D^i$ and recursively update inventory level from $j = 1$ to $\mathcal{P}$ as $\breve{I}_t^i(j) \leftarrow \tilde{I}_t^i(j) - \min\{\tilde{I}_t^i(j), R^i\}$, $R^i \leftarrow R^i - \min\{\tilde{I}_t^i(j), R^i\}$. Demands that are not fulfilled, $(D^i - \sum_{j=1}^{\mathcal{P}} \tilde{I}_t^i(j))^+$, are lost, and goods that expire, $\breve{I}_t^i(1)$, are discarded. Lastly, we set

$$\begin{aligned} &I_{t+1}^i(j) = \breve{I}_t^i(j+1), j = 1, \cdots, \mathcal{P}-1, \ I_{t+1}^i(\mathcal{P}) = 0, \\ &O_{t+1}^i(k+1) = O_t^i(k) \cdot (1 - E_k^i), k = 1, \cdots, \mathcal{L}-1, \ O_{t+1}^i(1) = a_t^i \cdot (1 - E_0^i), \end{aligned}$$

as the state of product i in the next period s_{t+1}^i.

In terms of cost, first, we assume each unit of product i in inventory incurs a holding cost of h_i per period. Second, when placing an order, a fixed order cost of f_i is charged, which is independent with the order quantity. Third, the perished product will incur a discarding cost of r_i per unit. Lastly, unfulfilled demands are lost forever at a penalty of ℓ_i per unit. We consider a CMDP formulation. Our primary cost is the total operating cost that includes holding, fixed order, and discarding costs, i.e.,

$$c^i(s_t^i, a_t^i) = h_i \cdot \sum_{j=1}^{\mathcal{P}} I_t^i(j) + f_i \cdot 1(a_t^i > 0) + r_i \cdot \breve{I}_t^i(1).$$

The auxiliary cost is the lost sale penalty, i.e.,

$$d^i(s_t^i, a_t^i) = \ell_i \cdot \Big(D^i - \sum_{j=1}^{\mathcal{P}} \tilde{I}_t^i(j)\Big)^+,$$

which captures the service quality. Then, the CMDP takes the following form

$$\min_{\pi} \ \mathbb{E}_\pi \Big[\sum_{i=1}^{N} \sum_{t=0}^{\infty} \gamma^t \cdot c^i(s_t^i, a_t^i)\Big], \text{ s.t. } \mathbb{E}_\pi \Big[\sum_{i=1}^{N} \sum_{t=0}^{\infty} \gamma^t \cdot \ell_i \cdot d^i(s_t^i, a_t^i)\Big] \le q.$$

Since the optimal policy is unknown in this case and solving the CMDP exactly is computationally prohibitive, we adapt a classic heuristic developed in literature as the benchmark. In particular, we consider the $(\tilde{s}, \tilde{S})$ policy: For each product, we place a new order and replenish its total inventory level (including the upcoming orders) to $\tilde{S}$ if and only if its existing total inventory level (including the upcoming orders) drops below $\tilde{s}$. This policy with properly chosen s and S has been shown to be optimal for systems with non-perishable inventory and constant lead time (Scarf 1960, Iglehart 1963, Veinott Jr and Wagner 1965). Since different (s, S) parameters lead to different lost sale penalties and operating costs, we enumerate all possible (s, S) policies and seek the optimal mixing policy such that the primary cost is minimized while the constraint is maintained. Specifically, let π_ω be an $(\tilde{s}, \tilde{S})$ policy with parameter $\omega = (\tilde{s}_\omega, \tilde{S}_\omega)$. We denote C_ω and D_ω as the associated operating cost and lost sale penalty. Then, the optimal mixing policy is obtained by solving

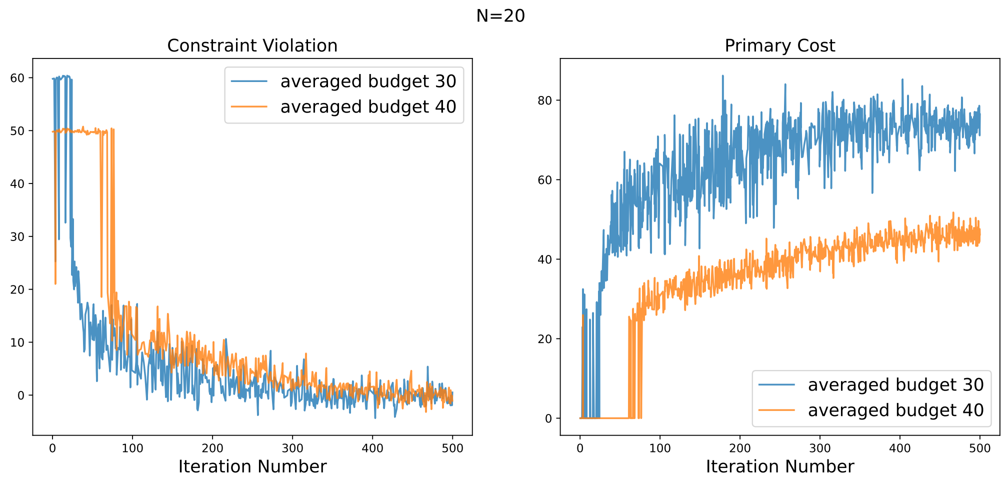
$$\min_p \sum_{\omega\in\Omega} p_\omega C_\omega \quad \text{s.t.} \sum_{\omega\in\Omega} p_\omega D_\omega \le q, \;\; \sum_{\omega\in\Omega} p_\omega = 1, \;\; p_\omega \ge 0,$$

where Ω is the collection of all feasible (s, S) parameters.

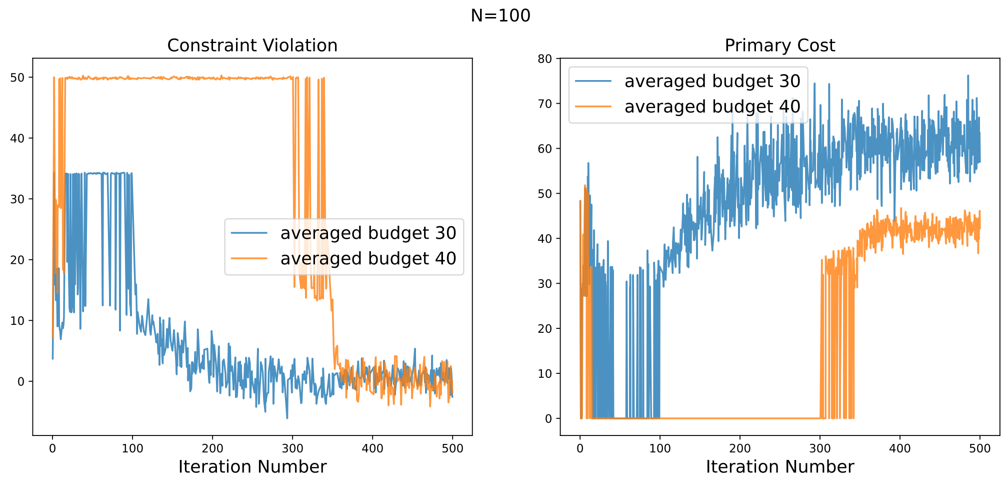
In our numerical experiments, we consider four problem instances with $N = 20, 100$ distinct products and averaged lost sale constraints (accumulated in time) $q/N = 30, 40$. For each product, the unit holding costs and fixed order costs are uniformly sampled from intervals $[0.5, 1.5]$ and $[2, 4]$. The unit lost sale penalty and discarding cost are fixed at 1 and 10 respectively for all products. The discount factor γ is 0.95. For each product, the inventory level takes non-negative integer values, and the action space is $\{0, 1, 2, \cdots, 9\}$. The demand of each product follows a uniform distribution over the set $\{1, 2, \cdots, 10\}$. The maximal shelf life is $\mathcal{P} = 5$ and the maximal lead time is $\mathcal{L} = 4$. For the stochastic lead time, we set $(\bar{p}_0, \bar{p}_1, \bar{p}_2, \bar{p}_3, \bar{p}_4) = (0.2, 0.4, 0.6, 0.8, 1)$. In this example, the number of unique (s, a) pairs for each product is approximately $O(10^{10})$, and the scale of the joint problem is $O(10^{12})$ when $N = 100$.

To overcome the curse of dimensionality, we apply an adapted version of Algorithm 2 where we use neural networks to approximate the Q-functions and policies (we fit different neural networks for different products). We initialize the neural networks via the Xavier initialization (Glorot and Bengio 2010), the Lagrangian multiplier is initialized at zero, and the stepsize is fixed at 0.02. In each primal-dual iteration, we use the stochastic gradient descent-based temporal difference learning algorithm with 10^4 steps to find the approximated Q-function, and then use stochastic gradient descent with 10^4 steps to find the approximated policy function. The implementation details are provided in Appendix C.

Figure 4 plots the averaged primary objective values, $C(\pi_m)/N$, and the averaged constrain violation, $(D(\pi_m) - q)/N$, at different iterations m. We observe that the lost sale penalty violates the constraint by quite a bit at the beginning. As the number of iterations increases, the constraint violation reduces to zero. The primary cost first increases and then stabilizes. Table 4 compares the policy learned by our primal-dual algorithm with 500 iterations and the optimal mixing $(\tilde{s}, \tilde{S})$ policy (benchmark policy) in different scenarios. We observe that our policy achieves a 19%-32% cost reduction compared to the benchmark policy while maintaining the constraint. This demonstrates the superior performance of our primal-dual algorithm for large-scale CMDPs.



(a) $N = 20$



(b) $N = 100$

Figure 4 Trajectories of objective and constraint violation

8. Conclusion and Future Directions

In this work, we propose a data-driven primal-dual algorithm to solve CMDPs. Our approach alternatively applies regularized policy iteration to improve the policy and subgradient ascent to

Table 4 Comparison of policies for the muli-product newsvendor problem

# of Products N	Average Threshold q/N	Primal-Dual average penalty	Primal-Dual operating cost	Benchmark operating cost
20	30	29.6	**74.2**	91.6
20	40	39.8	**46.1**	63.6
100	30	30.3	**61.4**	89.4
100	40	40.1	**42.3**	62.6

maintain the constraints. The algorithm achieves $O(1/\sqrt{T})$ convergence and only requires a policy evaluation at each iteration. It can be easily combined with advanced reinforcement learning techniques to deal with large-scale problems, with the added benefit of neat convergence analysis. It also enjoys the decomposability property for CMDPs with weakly coupled structures, which can help further reduce the computational complexity. Lastly, we apply our algorithm to solve two important classes of operations management problems: multi-class queue scheduling and multi-product inventory management. These applications demonstrate the scalability of our algorithm.

There are a few directions for future research. First, this work focuses on CMDPs with linear expectation constraints. It would be of interest to extend the primal-dual algorithm to other forms of constraints such as the auxiliary costs being in a convex set or chance constraints. The key challenge in handling more general forms of constraints is how to solve the relaxed unconstrained problem for a given Lagrangian multiplier. In our case, the relaxed problem is still an MDP, which may not be the case for more general forms of constraints (Miryoosefi et al. 2019, Chow et al. 2017). Second, to achieve better scalability, in addition to the weakly coupled structure explored in this paper, it would also be of great value to look at other special problem structures such as sparse networks (Gu et al. 2021) or latent low-rank structure (Sam et al. 2022). Third, as mentioned in Section 7.1, weakly coupled CMDPs can be viewed as a relaxation of weakly coupled MDPs. How to translate the optimal policy for the relaxed problem to a good MDP policy that satisfies all the hard constraints continues to be an interesting research direction. Lastly, from the perspective of applications, the implementation of our primal-dual algorithm for large-scale problems relies on efficient policy evaluation through value function approximation. This is relatively easy when the state space is large but the policy space is small. When the policy space is very large, we may need to develop further approximation to reduce the policy space (e.g., focus on base-stock policy as in inventory management (Agrawal and Jia 2019) or focus on certain index-based scheduling policies in queue scheduling (Zhong et al. 2022)).

References

Adelman D, Mersereau AJ (2008) Relaxations of weakly coupled stochastic dynamic programs. *Operations Research* 56(3):712–727.

Agrawal S, Jia R (2019) Learning in structured mdps with convex cost functions: Improved regret bounds for inventory management. *Proceedings of the 2019 ACM Conference on Economics and Computation*, 743–744.

Altman E (1999) *Constrained Markov decision processes*, volume 7 (CRC Press).

Antos A, Szepesvári C, Munos R (2007) Fitted q-iteration in continuous action-space mdps. *Advances in neural information processing systems* 20.

Bertsekas DP (2011) Dynamic programming and optimal control 3rd edition, volume ii. *Belmont, MA: Athena Scientific* .

Bertsekas DP (2015) *Convex optimization algorithms* (Athena Scientific Belmont).

Bertsimas D, Mišić VV (2016) Decomposable markov decision processes: A fluid optimization approach. *Operations Research* 64(6):1537–1555.

Bertsimas D, Orlin JB (1994) A technique for speeding up the solution of the Lagrangian dual. *Mathematical Programming* 63(1-3):23–45.

Bhatnagar S, Lakshmanan K (2012) An online actor–critic algorithm with function approximation for constrained markov decision processes. *Journal of Optimization Theory and Applications* 153(3):688–708.

Borkar VS (2005) An actor-critic algorithm for constrained markov decision processes. *Systems & control letters* 54(3):207–213.

Boyd S, Boyd SP, Vandenberghe L (2004) *Convex optimization* (Cambridge university press).

Brown DB, Smith JE (2020) Index policies and performance bounds for dynamic selection problems. *Management Science* .

Bubeck S (2014) Convex optimization: Algorithms and complexity. *arXiv preprint arXiv:1405.4980* .

Caramanis C, Dimitrov NB, Morton DP (2014) Efficient algorithms for budget-constrained markov decision processes. *IEEE Transactions on Automatic Control* 59(10):2813–2817.

Chen J, Dong J, Shi P (2020) A survey on skill-based routing with applications to service operations management. *Queueing Systems* 1–30.

Chen Y, Wang M (2016) Stochastic primal-dual methods and sample complexity of reinforcement learning. *arXiv preprint arXiv:1612.02516* .

Chow Y, Ghavamzadeh M, Janson L, Pavone M (2017) Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research* 18(1):6070–6120.

Chow Y, Nachum O, Duenez-Guzman E, Ghavamzadeh M (2018) A Lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems*, 8092–8101.

Dai J, Gluzman M (2020) Queueing network controls via deep reinforcement learning. *arXiv preprint arXiv:2008.01644* .

Dai J, Shi P (2019) Inpatient overflow: An approximate dynamic programming approach. *Manufacturing & Service Operations Management* 21(4):894–911.

Dai JG, Lin W, et al. (2008) Asymptotic optimality of maximum pressure policies in stochastic processing networks. *The Annals of Applied Probability* 18(6):2239–2299.

De Farias DP, Van Roy B (2003) The linear programming approach to approximate dynamic programming. *Operations research* 51(6):850–865.

Dong J, Shi P, Zheng F, Jin X (2019) Off-service placement in inpatient ward network: Resource pooling versus service slowdown. *Columbia Business School Research Paper Forthcoming* .

Farahmand Am, Ghavamzadeh M, Szepesvári C, Mannor S (2016) Regularized policy iteration with non-parametric function spaces. *The Journal of Machine Learning Research* 17(1):4809–4874.

Gattami A (2019) Reinforcement learning for multi-objective and constrained markov decision processes. *arXiv preprint arXiv:1901.08978* .

Gijsbrechts J, Boute RN, Van Mieghem JA, Zhang D (2021) Can deep reinforcement learning improve inventory management? performance on dual sourcing, lost sales and multi-echelon problems. *Manufacturing & Service Operations Management* .

Girard C, Green LV, Lewis ME, Xie J (2020) Two-class constrained optimization with applications to queueing control. *Naval Research Logistics (NRL)* .

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256 (JMLR Workshop and Conference Proceedings).

Gu H, Guo X, Wei X, Xu R (2021) Mean-field multi-agent reinforcement learning: A decentralized network approach. *arXiv preprint arXiv:2108.02731* .

Hayes CF, Rădulescu R, Bargiacchi E, Källström J, Macfarlane M, Reymond M, Verstraeten T, Zintgraf LM, Dazeley R, Heintz F, et al. (2022) A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems* 36(1):1–59.

Iglehart DL (1963) Optimality of (s, s) policies in the infinite horizon dynamic inventory problem. *Management science* 9(2):259–267.

Kakade S, Langford J (2002) Approximately optimal approximate reinforcement learning. *ICML*, volume 2, 267–274.

Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Le HM, Voloshin C, Yue Y (2019) Batch policy learning under constraints. *arXiv preprint arXiv:1903.08738* .

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. *nature* 521(7553):436–444.

Lee D, He N (2019) Stochastic primal-dual q-learning algorithm for discounted mdps. *2019 american control conference (acc)*, 4897–4902 (IEEE).

Lin Q, Nadarajah S, Soheili N (2020) Revisiting approximate linear programming: Constraint-violation learning with applications to inventory control and energy storage. *Management Science* 66(4):1544–1562.

Liu B, Cai Q, Yang Z, Wang Z (2019) Neural proximal/trust region policy optimization attains globally optimal policy. *arXiv preprint arXiv:1906.10306* .

Mandelbaum A, Stolyar AL (2004) Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized cμ-rule. *Operations Research* 52(6):836–855.

Mannor S, Shimkin N (2001) The steering approach for multi-criteria reinforcement learning. *Advances in Neural Information Processing Systems* 14.

Miryoosefi S, Brantley K, Daume III H, Dudik M, Schapire RE (2019) Reinforcement learning with convex constraints. *Advances in Neural Information Processing Systems*, 14093–14102.

Moallemi CC, Kumar S, Van Roy B (2008) Approximate and data-driven dynamic programming for queueing networks. *Submitted for publication* .

Munos R, Szepesvári C (2008) Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research* 9(5).

Nadarajah S, Margot F, Secomandi N (2015) Relaxations of approximate linear programs for the real option management of commodity storage. *Management Science* 61(12):3054–3076.

Natarajan S, Tadepalli P (2005) Dynamic preferences in multi-criteria reinforcement learning. *Proceedings of the 22nd international conference on Machine learning*, 601–608.

Nedić A, Ozdaglar A (2009) Subgradient methods for saddle-point problems. *Journal of optimization theory and applications* 142(1):205–228.

Neely MJ (2011) Online fractional programming for Markov decision systems. *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 353–360 (IEEE).

Nemirovski A (2004) Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization* 15(1):229–251.

Nocedal J, Wright SJ (1999) *Numerical optimization* (Springer).

Oroojlooyjadid A, Nazari M, Snyder LV, Takáč M (2022) A deep q-network for the beer game: Deep reinforcement learning for inventory optimization. *Manufacturing & Service Operations Management* 24(1):285–304.

Puterman ML (2014) *Markov decision processes: discrete stochastic dynamic programming* (John Wiley & Sons).

Sam T, Chen Y, Yu CL (2022) Overcoming the long horizon barrier for sample-efficient reinforcement learning with latent low-rank structure. *arXiv preprint arXiv:2206.03569* .

Scarf H (1960) The optimality of (s, s) policies in the dynamic inventory problem .

Schweitzer PJ, Seidmann A (1985) Generalized polynomial approximations in markovian decision processes. *Journal of mathematical analysis and applications* 110(2):568–582.

Singh R, Kumar P (2018) Throughput optimal decentralized scheduling of multihop networks with end-to-end deadline constraints: Unreliable links. *IEEE Transactions on Automatic Control* 64(1):127–142.

Singh SP, Cohn D (1998) How to dynamically merge markov decision processes. *Advances in neural information processing systems*, 1057–1063.

Sion M, et al. (1958) On general minimax theorems. *Pacific Journal of mathematics* 8(1):171–176.

Song H, Tucker AL, Graue R, Moravick S, Yang JJ (2020) Capacity pooling in hospitals: The hidden consequences of off-service placement. *Management Science* 66(9):3825–3842.

Sun P, Wang K, Zipkin P (2014) Quadratic approximation of cost functions in lost sales and perishable inventory control problems. *Fuqua School of Business, Duke University, Durham, NC* .

Sutton RS, Barto AG (2018) *Reinforcement learning: An introduction* (MIT press).

Tesauro G, Das R, Chan H, Kephart J, Levine D, Rawson F, Lefurgy C (2007) Managing power consumption and performance of computing systems using reinforcement learning. *Advances in neural information processing systems* 20.

Tessler C, Mankowitz DJ, Mannor S (2018) Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074* .

Topaloglu H (2009) Using lagrangian relaxation to compute capacity-dependent bid prices in network revenue management. *Operations Research* 57(3):637–649.

Topaloglu H, Kunnumkal S (2006) Approximate dynamic programming methods for an inventory allocation problem under uncertainty. *Naval Research Logistics (NRL)* 53(8):822–841.

Turken N, Tan Y, Vakharia AJ, Wang L, Wang R, Yenipazarli A (2012) The multi-product newsvendor problem: Review, extensions, and directions for future research. *Handbook of newsvendor problems*, 3–39 (Springer).

Van Roy B, Bertsekas DP, Lee Y, Tsitsiklis JN (1997) A neuro-dynamic programming approach to retailer inventory management. *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 4, 4052–4057 (IEEE).

Veatch MH (2005) Approximate dynamic programming for networks: Fluid models and constraint reduction. *preprint* .

Veinott Jr AF, Wagner HM (1965) Computing optimal (s, s) inventory policies. *Management Science* 11(5):525–552.

Yang R, Sun X, Narasimhan K (2019) A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *Advances in Neural Information Processing Systems* 32.

Zhong Y, Birge JR, Ward A (2022) Learning the scheduling policy in time-varying multiclass many server queues with abandonment. *Available at SSRN* .

Zhou D, Chen J, Gu Q (2020) Provable multi-objective reinforcement learning with generative models. *arXiv preprint arXiv:2011.10134* .

Appendix

A. Proofs of the Main Results

A.1. Proof of Theorem 1

A.1.1. Proof under the discounted cost criterion We first introduce a lemma, which provides upper and lower bounds for the movement of the Lagrangian after a single update.

LEMMA 1. *Let $\{(\pi_m, \lambda_m)\}_{m\geq 0}$ be the sequences of stationary policies and Lagrangian multipliers generated by Algorithm 1. Then for arbitrary $\lambda \in \mathbb{R}_+^K$ and $\pi \in \Pi_S$, we have the upper bound*

$$L(\pi_m, \lambda) - L(\pi_m, \lambda_m) \leq \frac{1}{2\eta_m}\left(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2\right) + \eta_m \cdot \left\|\partial_\lambda L(\pi_m, \lambda_m)\right\|^2,$$

and the lower bound

$$L(\pi, \lambda_m) - L(\pi_m, \lambda_m) \geq \frac{1}{(1-\gamma)\eta_m}\Big(\Phi^\pi(\pi\|\pi_{m+1}) - \Phi^\pi(\pi\|\pi_m)\Big) - \frac{\eta_m}{8(1-\gamma)}\big(\sup_{s\in\mathcal{S}}\sup_{a\in\mathcal{A}}|Q^{\lambda_m,\pi_m}(s,a)|\big)^2,$$

where Φ^π is defined in (26)*.*

Before we prove Lemma 1, we present two auxiliary lemmas. The first lemma (Lemma 2) is quite standard. A similar version of it can be found in Proposition 3.2.2 in Bertsekas (2015). We provide the proof here for self-completeness.

LEMMA 2. *Let f be a proper convex function on a space Ω (not necessarily a Euclidean space). Let $\mathcal{C}$ be an open set in Ω, and $\Psi_\xi(\cdot\|\cdot)$ be the Bregman divergence induced by a strictly convex function ξ on Ω. For an arbitrary constant $\eta > 0$ and a point $x_0 \in \Omega$, define*

$$x^* = \arg\min_{x\in\mathcal{C}}\Big\{f(x) + \frac{1}{\eta}\Psi_\xi\big(x\|x_0\big)\Big\}.$$

Then we have

$$f(x) - f(x^*) \geq \frac{1}{\eta}\Big(\Psi_\xi\big(x^*\|x_0\big) + \Psi_\xi\big(x\|x^*\big) - \Psi_\xi\big(x\|x_0\big)\Big),\ \forall\ x\in\Omega.$$

By symmetry, for a concave function g on Ω and

$$\hat{x}^* = \arg\max_{x\in\mathcal{C}}\Big\{g(x) - \frac{1}{\eta}\Psi_\xi\big(x\|x_0\big)\Big\}.$$

Then

$$g(x) - g(\hat{x}^*) \leq -\frac{1}{\eta}\Big(\Psi_\xi\big(\hat{x}^*\|x_0\big) + \Psi_\xi\big(x\|\hat{x}^*\big) - \Psi_\xi\big(x\|x_0\big)\Big),\ \forall\ x\in\Omega.$$

Proof of Lemma 2 We first consider the minimization problem. Since x^* minimizes the objective $f(x) + \eta^{-1}\cdot\Psi_\xi(x\|x_0)$ on set $\mathcal{C}$, there exists a subgradient $q^* \in \partial_x f(x^*)$, such that for

$$p^* = q^* + \frac{1}{\eta}\partial_x\Psi_\xi(x^*\|x_0) = q^* + \frac{1}{\eta}\big(\nabla\xi(x^*) - \nabla\xi(x_0)\big),$$

we have

$$\langle p^*, x - x^*\rangle \geq 0,\ \forall x\in\mathcal{C}.$$

Then, for any $x \in \mathcal{C}$,

$$\begin{aligned} f(x) &\geq f(x^*) + \langle q^*, x - x^* \rangle \\ &\geq f(x^*) + \eta^{-1} \cdot \langle \nabla \xi(x_0) - \nabla \xi(x^*), x - x^* \rangle \\ &= f(x^*) + \eta^{-1} \cdot \Big(\Psi_\xi\big(x^* \| x_0\big) + \Psi_\xi\big(x \| x^*\big) - \Psi_\xi\big(x \| x_0\big) \Big), \end{aligned}$$

where the last equality follows from the definition of Bregman divergence, i.e.,

$$\Psi_\xi(x \| y) = \xi(x) - \xi(y) - \big\langle \nabla \xi(y), x - y \big\rangle.$$

For the maximization problem, we only need to consider $-g$ and apply the above result. $\square$

The next lemma is Lemma 6.1 in (Kakade and Langford 2002). Given two policies, it characterizes the difference of expected cumulative costs as the inner product of the advantage function of one policy and the occupation measure of another policy. Note that the value function V^π and the action-value function Q^π of an MDP under policy π are defined in (1) and (16).

Lemma 3. *For arbitrary policies $\pi, \pi' \in \Pi_S$,*

$$\mathbb{E}_{s \sim \mu_0}\big[V^{\pi'}(s)\big] - \mathbb{E}_{s \sim \mu_0}\big[V^{\pi}(s)\big] = \frac{1}{1-\gamma} \mathbb{E}_{(s,a) \sim \nu^{\pi'}}\big[Q^\pi(s,a) - V^\pi(s)\big],$$

where $\nu^{\pi'}(\cdot,\cdot)$ is the occupation measure associated with π'.

Proof of Lemma 1 For the upper bound, note that because $L(\pi_m, \lambda)$ is linear in λ,

$$\lambda_{m+1} = \mathrm{Proj}_{\Lambda_M}\big\{\lambda_m + \eta_m \cdot \partial_\lambda L(\pi_m, \lambda_m)\big\}$$

is equivalent to

$$\lambda_{m+1} = \operatorname*{arg\,max}_{\lambda \in \Lambda_M} \Big\{ L(\pi_m, \lambda) - \frac{1}{2\eta_m} \|\lambda - \lambda_m\|^2 \Big\}.$$

Then, by Lemma 2, we have

$$\begin{aligned} L(\pi_m, \lambda) - L(\pi_m, \lambda_{m+1}) &\leq (2\eta_m)^{-1} \big(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2 - \|\lambda_{m+1} - \lambda_m\|^2 \big) \\ &\leq (2\eta_m)^{-1} \big(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2 \big). \end{aligned}$$

Next,

$$\begin{aligned} L(\pi_m, \lambda) - L(\pi_m, \lambda_m) \leq& (2\eta_m)^{-1} \big(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2 \big) + L(\pi_m, \lambda_{m+1}) - L(\pi_m, \lambda_m) \\ =& (2\eta_m)^{-1} \big(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2 \big) + \big\langle \partial_\lambda L(\pi_m, \lambda_m), \lambda_{m+1} - \lambda_m \big\rangle \\ \leq& (2\eta_m)^{-1} \big(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2 \big) + \eta_m \cdot \big\|\partial_\lambda L(\pi_m, \lambda_m)\big\|^2, \end{aligned}$$

where the last inequality follows from the definition of λ_{m+1} and the non-expansive property of the projection. We thus obtain the upper bound.

For the lower bound, recall that we update π_m via

$$\pi_{m+1}(\cdot|s) = \operatorname*{arg\,min}_{\pi(\cdot|s) \in \Delta_{\mathcal{A}}} \Big\{ \big\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi(\cdot|s) \big\rangle + \frac{1}{\eta_m} \mathrm{KL}\big(\pi(\cdot|s) \| \pi_m(\cdot|s)\big) \Big\},$$

for each state $s \in \mathcal{S}$. Then, for an arbitrary stationary policy $\pi' \in \Pi_S$, we have

$$\pi_{m+1} = \underset{\pi \in \Pi_S}{\arg\min} \left\{ \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi(\cdot|s) \right\rangle + \frac{1}{\eta_m} \mathrm{KL}\big(\pi(\cdot|s) \| \pi_m(\cdot|s)\big) \right] \right\}$$

where $\nu_s^{\pi'}$ is the state occupation measure associated with π'.

Note that the space of the stationary policy, Π_S, can be represented as the product space of simplex $\Delta_{\mathcal{A}}$. Consider $\Omega := \Pi_S = \big(\Delta_{\mathcal{A}}\big)^{\otimes |\mathcal{S}|}$ and let

$$\begin{aligned} f(\pi) &:= \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi(\cdot|s) \right\rangle \right], \\ \Psi_\xi(\pi) &:= \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\mathrm{KL}\big(\pi(\cdot|s) \| \pi_m(\cdot|s)\big) \right] = \Phi^{\pi'}(\pi \| \pi_m). \end{aligned}$$

where $\Phi^{\pi'}$ is defined in (26). Since $f(\pi)$ is linear in π, setting $\pi = \pi'$, by Lemma 2, we obtain

$$\mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi'(\cdot|s) - \pi_{m+1}(\cdot|s) \right\rangle \right] \geq \eta_m^{-1} \Big(\Phi^{\pi'}(\pi_{m+1} \| \pi_m) + \Phi^{\pi'}(\pi' \| \pi_{m+1}) - \Phi^{\pi'}(\pi' \| \pi_m) \Big),$$

which can be equivalently written as

$$\begin{aligned} &\eta_m^{-1} \cdot \Big(\Phi^{\pi'}(\pi' \| \pi_{m+1}) - \Phi^{\pi'}(\pi' \| \pi_m) + \Phi^{\pi'}(\pi_{m+1} \| \pi_m) \Big) \\ \leq& \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi'(\cdot|s) - \pi_m(\cdot|s) \right\rangle \right] + \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \right\rangle \right]. \end{aligned} \tag{32}$$

We next derive an upper bound for the right-hand side of inequalities (32). Let $\| \cdot \|_{\mathrm{TV}}$ denote the total variation norm of probability distributions. First, for each state $s \in \mathcal{S}$,

$$\begin{aligned} &\eta_m \cdot \left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \right\rangle \\ \leq& \eta_m \cdot \sup_{a \in \mathcal{A}} \left| Q^{\pi_m, \lambda_m}(s, a) \right| \cdot \left\| \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \right\|_{\mathrm{TV}} \\ \leq& \frac{\eta_m^2}{8} \cdot \Big(\sup_{s \in \mathcal{S}} \sup_{a \in \mathcal{A}} \left| Q^{\pi_m, \lambda_m}(s, a) \right| \Big)^2 + 2 \cdot \left\| \pi_{m+1}(\cdot|s) - \pi_m(\cdot|s) \right\|_{\mathrm{TV}}^2 \\ \leq& \frac{\eta_m^2}{8} \cdot \Big(\sup_{s \in \mathcal{S}} \sup_{a \in \mathcal{A}} \left| Q^{\pi_m, \lambda_m}(s, a) \right| \Big)^2 + \mathrm{KL}\big(\pi_{m+1}(\cdot|s) \| \pi_m(\cdot|s)\big) \text{ by Pinsker's inequality.} \end{aligned}$$

Hence, by taking the average, we obtain

$$\mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \right\rangle \right] \leq \frac{\eta_m}{8} \cdot \Big(\sup_{s \in \mathcal{S}} \sup_{a \in \mathcal{A}} \left| Q^{\pi_m, \lambda_m}(s, a) \right| \Big)^2 + \eta_m^{-1} \cdot \Phi^{\pi'}(\pi_{m+1} \| \pi_m). \tag{33}$$

Second, recall that $\nu^\pi(s, a) = \nu_s^\pi(s) \cdot \pi(a|s)$ and $V^\pi(s) = \langle Q^\pi(s, \cdot), \pi(\cdot|s) \rangle$. Then, by Lemma 3, for the modified unconstrained MDP, we have

$$\begin{aligned} \mathbb{E}_{s \sim \nu_s^{\pi'}} \left[\left\langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi'(\cdot|s) - \pi_m(\cdot|s) \right\rangle \right] =& \mathbb{E}_{(s,a) \sim \nu^{\pi'}} \left[Q^{\pi_m, \lambda_m}(s, a) - V^{\pi_m, \lambda_m}(s) \right] \\ =& (1 - \gamma) \cdot \big(L(\pi', \lambda_m) - L(\pi_m, \lambda_m) \big). \end{aligned} \tag{34}$$

Finally, combining (32)-(34), we obtain

$$L(\pi', \lambda_m) - L(\pi_m, \lambda_m) \geq \frac{1}{(1 - \gamma)\eta_m} \Big(\Phi^{\pi'}(\pi' \| \pi_{m+1}) - \Phi^{\pi'}(\pi' \| \pi_m) \Big) - \frac{\eta_m}{8(1 - \gamma)} \big(\sup_{s \in \mathcal{S}} \sup_{a \in \mathcal{A}} |Q^{\pi_m, \lambda_m}(s, a)| \big)^2.$$

□

We are now ready to prove Theorem 1.

Proof of Theorem 1 **We prove the bound for $D(\bar{\pi}_T) - q$ first.** For this, we only need to establish an upper bound for $\big\|[\partial_\lambda L(\bar{\pi}_T, \lambda)]^+\big\|$.

Since $L(\pi_m, \lambda)$ is linear in λ, we have

$$L(\pi_m, \lambda_m) - L(\pi_m, \lambda^*) = (\lambda_m - \lambda^*)^\top \partial_\lambda L(\pi_m, \lambda_m). \tag{35}$$

by the saddle point property of (π^*, λ^*), we also have

$$L(\pi_m, \lambda^*) \geq L(\pi^*, \lambda^*). \tag{36}$$

In the following, we denote by $L^* := L(\pi^*, \lambda^*)$.

Meanwhile, by the first part of Lemma 1, for any λ, we have

$$\begin{aligned} \eta_m \cdot (\lambda - \lambda_m)^\top \partial_\lambda L(\pi_m, \lambda_m) &= \eta_m \cdot (L(\pi_m, \lambda) - L(\pi_m, \lambda_m)) \\ &\leq \big(\|\lambda_m - \lambda\|^2 - \|\lambda_{m+1} - \lambda\|^2\big)/2 + \eta_m^2 G^2. \end{aligned} \tag{37}$$

Combining inequalities (35)-(37), we obtain

$$\begin{aligned} &\eta_m \cdot (\lambda - \lambda^*)^\top \partial_\lambda L(\pi_m, \lambda_m) \\ =&\eta_m \cdot (\lambda - \lambda_m)^\top \partial_\lambda L(\pi_m, \lambda_m) + \eta_m \cdot (\lambda_m - \lambda^*)^\top \partial_\lambda L(\pi_m, \lambda_m) \\ \leq&\frac{1}{2}\big(\|\lambda_m - \lambda\|^2 - \|\lambda_{m+1} - \lambda\|^2\big) + \eta_m^2 G^2 + \eta_m \cdot \big(L(\pi_m, \lambda_m) - L^*\big). \end{aligned}$$

By taking the telescoping sum of the above inequality, we have for any $\lambda \in \Lambda_M$,

$$\begin{aligned} &\sum_{m=0}^{T-1} \eta_m \cdot (\lambda - \lambda^*)^\top \partial_\lambda L(\pi_m, \lambda_m) \\ \leq&\frac{1}{2}\big(\|\lambda_0 - \lambda\|^2 - \|\lambda_T - \lambda\|^2\big) + \Big(\sum_{m=0}^{T-1} \eta_m^2\Big) \cdot G^2 + \sum_{m=0}^{T-1} \eta_m \cdot \big(L(\pi_m, \lambda_m) - L^*\big). \end{aligned} \tag{38}$$

For the left hand side of (38), let

$$\zeta_T := \sum_{m=0}^{T-1} \eta_m \cdot \partial_\lambda L(\pi_m, \lambda_m) = \Big(\sum_{m=0}^{T-1} \eta_m\Big) \cdot \partial_\lambda L(\bar{\pi}_T, \lambda),$$

where the last equality follows from the definition of $\bar{\pi}_T$ and the linearity of value function under the mixing operation. If $[\zeta_T]^+ = 0$, then the upper bound holds trivially. Otherwise, let

$$\tilde{\lambda} = \lambda^* + r \cdot \frac{[\zeta_T]^+}{\big\|[\zeta_T]^+\big\|},$$

where r is the slackness constant in the definition of Λ_M in (24). Then it is easy to see that $\tilde{\lambda} \in \Lambda_M$. By (38), we have

$$(\tilde{\lambda} - \lambda^*)^\top \zeta_T \leq \frac{1}{2} \max_{\lambda \in \Lambda_M} \|\lambda - \lambda_0\|^2 + \Big(\sum_{m=0}^{T-1} \eta_m^2\Big) \cdot G^2 + \sum_{m=0}^{T-1} \eta_m \cdot \big(L(\pi_m, \lambda_m) - L^*\big).$$

By the definition of $\tilde{\lambda}$, we also have

$$(\tilde{\lambda} - \lambda^*)^\top \zeta_T = r \cdot \frac{([\zeta_T]^+)^\top \zeta_T}{\big\|[\zeta_T]^+\big\|} = r \cdot \big\|[\zeta_T]^+\big\| = r \cdot \Big(\sum_{m=0}^{T-1} \eta_m\Big) \cdot \big\|[\partial_\lambda L(\bar{\pi}_T, \lambda)]^+\big\|.$$

Hence,

$$\big\|[\partial_\lambda L(\bar{\pi}_T,\lambda)]^+\big\| \le \frac{\max_{\lambda\in\Lambda_M}\|\lambda-\lambda_0\|^2}{2r\cdot\sum_{m=0}^{T-1}\eta_m} + G^2\frac{\sum_{m=0}^{T-1}\eta_m^2}{r\cdot\sum_{m=0}^{T-1}\eta_m} + \frac{\sum_{m=0}^{T-1}\eta_m\cdot\big(L(\pi_m,\lambda_m)-L^*\big)}{r\cdot\sum_{m=0}^{T-1}\eta_m}. \tag{39}$$

Next, recall that $\bar{\pi}_T=\sum_{m=0}^{T-1}\tilde{\eta}_m\pi_m$, where $\tilde{\eta}_m=\eta_m/(\sum_{m=0}^{T-1}\eta_m),\ m=0,\dots,T-1$. Since λ^* is the optimal solution of the dual problem and $L(\pi^*,\lambda^*)\ge L(\pi^*,\bar{\lambda}_T)$, by the saddle point property, we have

$$\begin{aligned}\sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L^*\big) &= \sum_{m=0}^{T-1}\tilde{\eta}_m\cdot L(\pi_m,\lambda_m)-L^* \\ &\le \sum_{m=0}^{T-1}\tilde{\eta}_m\cdot L(\pi_m,\lambda_m)-L(\pi^*,\bar{\lambda}_T) \\ &= \sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L(\pi^*,\lambda_m)\big). \end{aligned} \tag{40}$$

Similarly, under Assumption 3, by the second part in Lemma 1, we have for $\pi=\pi^*$,

$$\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L(\pi^*,\lambda_m)\big) \le \Big((1-\gamma)\cdot\sum_{m=0}^{T-1}\eta_m\Big)^{-1}\left(\Phi^{\pi^*}(\pi^*\|\pi_m)-\Phi^{\pi^*}(\pi^*\|\pi_{m+1})+\frac{\eta_m^2G^2}{8}\right)$$

By taking the telescoping sum of the above inequality, we have

$$\sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L(\pi^*,\lambda_m)\big) \le \Big((1-\gamma)\cdot\sum_{m=0}^{T-1}\eta_m\Big)^{-1}\cdot\Big(\frac{G^2}{8}\cdot\sum_{m=0}^{T-1}\eta_m^2+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big), \tag{41}$$

as the weighted KL divergence $\Phi^{\pi^*}(\cdot||\cdot)$ is nonnegative.

Lastly, combining inequalities (39)-(41), we have

$$\big\|[\partial_\lambda L(\bar{\pi}_T,\lambda)]^+\big\| \le \frac{(M+r)^2}{r\cdot\sum_{m=0}^{T-1}\eta_m} + \Big(1+\frac{1}{8(1-\gamma)}\Big)G^2\frac{\sum_{m=0}^{T-1}\eta_m^2}{r\cdot\sum_{m=0}^{T-1}\eta_m} + \frac{(1-\gamma)^{-1}\Phi^{\pi^*}(\pi^*\|\pi_0)}{r\cdot\sum_{m=0}^{T-1}\eta_m}.$$

If we set $\eta_m=\Theta(1/\sqrt{m})$, there exists finite constants κ_1 and κ_2 such that

$$\sum_{m=0}^{T-1}\eta_m\ge\kappa_1\sqrt{T} \text{ and } \sum_{m=0}^{T-1}\eta_m^2\le\kappa_2\log(T).$$

Subsequently, we obtain

$$\big\|[\partial_\lambda L(\bar{\pi}_T,\lambda)]^+\big\| \le \Big((M+r)^2+\frac{9}{8}G^2\kappa_2\log(T)+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{r(1-\gamma)\kappa_1\sqrt{T}}.$$

Similarly, if we set $\eta_m=\eta$ (constant step size), then

$$\big\|[\partial_\lambda L(\bar{\pi}_T,\lambda)]^+\big\| \le \Big((M+r)^2+\frac{1}{1-\gamma}\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\frac{1}{rT\eta} + \Big(1+\frac{1}{8(1-\gamma)}\Big)\frac{G^2\eta}{r},$$

We next prove the bound for $C(\bar{\pi}_T)-L^*$. We start with the upper bound. By the definition of $\bar{\pi}_T$, we have

$$C(\bar{\pi}_T)-L^* = \sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L^*\big) - \sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\lambda_m^\top(D(\pi_m)-q). \tag{42}$$

From inequalities (40) and (41), we have

$$\sum_{m=0}^{T-1}\tilde{\eta}_m\cdot\big(L(\pi_m,\lambda_m)-L^*\big) \le \Big((1-\gamma)\cdot\sum_{m=0}^{T-1}\eta_m\Big)^{-1}\cdot\Big(\frac{G^2}{8}\sum_{m=0}^{T-1}\eta_m^2+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big).$$

Next, since $D(\pi_m) - q = \partial_\lambda L(\pi_m, \lambda_m)$, setting $\lambda = 0$ in (37) and taking the telescoping sum, we obtain

$$-\sum_{m=0}^{T-1} \tilde{\eta}_m \cdot \lambda_m^\top (D(\pi_m) - q) \leq \frac{\|\lambda_0\|^2/2 + G^2 \cdot \sum_{m=0}^{T-1} \eta_m^2}{\sum_{m=0}^{T-1} \eta_m}.$$

Hence,

$$C(\bar{\pi}_T) - L^* \leq \left(\Big(1 - \gamma + \frac{1}{8}\Big) G^2 \sum_{m=0}^{T-1} \eta_m^2 + \Phi^{\pi^*}(\pi^* \| \pi_0) + \frac{(1-\gamma)\|\lambda_0\|^2}{2} \right) \frac{1}{(1-\gamma)\sum_{m=0}^{T-1} \eta_m}.$$

For the lower bound, by the saddle point property, we have

$$C(\bar{\pi}_T) = L(\bar{\pi}_T, \lambda^*) - (\lambda^*)^\top (D(\bar{\pi}_T) - q) \geq L^* - (\lambda^*)^\top (D(\bar{\pi}_T) - q).$$

Since $\lambda^* \geq 0$ and $D(\bar{\pi}_T) - q \leq [D(\bar{\pi}_T) - q]^+ = [\partial_\lambda L(\bar{\pi}_T, \lambda)]^+$,

$$C(\bar{\pi}_T) - L^* \geq -\|\lambda^*\| \big\| [\partial_\lambda L(\bar{\pi}_T, \lambda)]^+ \big\|.$$

When $\eta_m = \Theta(1/\sqrt{m})$, we have

$$C(\bar{\pi}_T) - L^* \leq \Big(\frac{9G^2}{8} \kappa_2 \log(T) + \Phi^{\pi^*}(\pi^* \| \pi_0) + \frac{\|\lambda_0\|^2}{2} \Big) \frac{1}{(1-\gamma)\kappa_1 \sqrt{T}}.$$

and

$$C(\bar{\pi}_T) - L^* \geq -\|\lambda^*\| \Big((M + r)^2 + \frac{9}{8} G^2 \kappa_2 \log(T) + \Phi^{\pi^*}(\pi^* \| \pi_0) \Big) \frac{1}{r(1-\gamma)\kappa_1 \sqrt{T}}.$$

Similarly, when $\eta_m = \eta$, we have

$$C(\bar{\pi}_T) - L^* \leq \Big(\frac{1}{1-\gamma} \Phi^{\pi^*}(\pi^* \| \pi_0) + \frac{\|\lambda_0\|^2}{2} \Big) \frac{1}{T\eta} + \frac{9G^2 \eta}{8(1-\gamma)},$$

$$C(\bar{\pi}_T) - L^* \geq -\|\lambda^*\| \Big((M + r)^2 + \frac{1}{1-\gamma} \Phi^{\pi^*}(\pi^* \| \pi_0) \Big) \frac{1}{rT\eta} - \|\lambda^*\| \Big(1 + \frac{1}{8(1-\gamma)} \Big) \frac{G^2 \eta}{r}.$$

A.1.2. Proof under the long-run average cost criterion The proof follows exactly the same lines of argument as in the discounted case. The only difference is that we need to replace Lemma 3 with the following lemma, which characterizes the difference of long-run average costs under different policies. Note that in what follows, $V^\pi(s)$ and $Q^\pi(s,a)$ denote the relative value function and relative Q-function as defined in equations (22) and (23).

LEMMA 4. *Under long-run average cost criterion, for arbitrary policies $\pi, \pi' \in \Pi_S$,*

$$C(\pi') - C(\pi) = \mathbb{E}_{(s,a) \sim \nu^{\pi'}} \big[Q^\pi(s,a) - V^\pi(s) \big].$$

where $\nu^{\pi'}(\cdot,\cdot)$ is the stationary distribution of the Markov chain under policy π'.

Proof of Lemma 4 Note that

$$\begin{aligned}
\mathbb{E}_{(s,a) \sim \nu^{\pi'}} \big[Q^\pi(s,a) - V^\pi(s) \big] &= \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \nu^{\pi'}(s,a) \cdot \Big(c(s,a) - C(\pi) + \sum_{s \in \mathcal{S}} V^\pi(s') P(s'|s,a) - V^\pi(s) \Big) \\
&= C(\pi') - C(\pi) + \sum_{s \in \mathcal{S}, a \in \mathcal{A}} \nu^{\pi'}(s,a) \cdot \Big(\sum_{s \in \mathcal{S}} V^\pi(s') P(s'|s,a) - V^\pi(s) \Big), \\
&= C(\pi') - C(\pi),
\end{aligned}$$

where the last step follows since

$$\sum_{s \in \mathcal{S}, a \in \mathcal{A}} \nu^{\pi'}(s,a) P(s'|s,a) = \nu_s^{\pi'}(s'),$$

and $\sum_{a \in \mathcal{A}} \nu^{\pi'}(s,a) = \nu_s^{\pi'}(s)$. $\square$

A.2. Proof of Theorem 2

The proof follows similar lines of argument as the proof of Theorem 1, except that we now need to incorporate the approximation errors.

We first modify Lemma 1 to incorporate the approximation errors.

LEMMA 5. *Let $\{(\pi_{\beta_m}, \lambda_m)\}_{m\geq 0}$ be the sequences of stationary policies and Lagrangian multipliers generated by Algorithm 2. Then for arbitrary $\lambda \in \mathbb{R}_+^K$ and the optimal policy π^*, we have the upper bound*

$$L(\pi_{\beta_m}, \lambda) - L(\pi_{\beta_m}, \lambda_m) \leq \frac{1}{2\eta_m}\left(\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2\right) + \eta_m \cdot \left\|\partial_\lambda L(\pi_m, \lambda_m)\right\|^2,$$

and the lower bound

$$\begin{aligned} L(\pi^*, \lambda_m) - L(\pi_{\beta_m}, \lambda_m) \geq & \frac{1}{(1-\gamma)\eta_m}\Big(\Phi^{\pi^*}(\pi^*\|\pi_{\beta_{m+1}}) - \Phi^{\pi^*}(\pi^*\|\pi_{\beta_m})\Big) \\ & - \frac{\eta_m}{8(1-\gamma)}\Big(\sup_{s\in\mathcal{S}}\sup_{a\in\mathcal{A}}|Q^{\pi_{\beta_m},\lambda_m}(s,a)|\Big)^2 + \mathbb{E}_{s\sim\nu_s^{\pi^*}}\left[\epsilon_m(s) + \eta_m^{-1}\cdot\delta_m(s)\right]. \end{aligned}$$

where Φ^π is defined in (26), $\epsilon_m(s)$ and $\delta_m(s)$ are the approximation error terms specified in later proof.

Proof of Lemma 5 The upper bounds follow directly from Lemma 1.

For the lower bound, we first introduce some new notations. Let

$$\begin{aligned} \pi_{m+1}(\cdot|s) &\propto \pi_{\beta_m}(\cdot|s)\cdot\exp\big\{-\eta_m\cdot Q^{\pi_{\beta_m},\lambda_m}(s,\cdot)\big\}, \\ \tilde{\pi}_{m+1}(\cdot|s) &\propto \pi_{\beta_m}(\cdot|s)\cdot\exp\big\{-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,\cdot)\big\}. \end{aligned}$$

Note that $\pi_{m+1}(\cdot|s)$ is the ideal updates of policy at the $(m+1)$-th iteration if we can evaluate $Q^{\pi_{\beta_m},\lambda_m}(s,a)$ exactly. $\tilde{\pi}_{m+1}(\cdot|s)$ is the ideal update when we replace the true Q-function with its approximation $Q^{\lambda_m}_{\alpha_m}(s,a)$, but do not project the policy onto the parameterized policy space. π_{m+1} and $\tilde{\pi}_{m+1}$ are not required in the actual algorithm. We only use them in the theoretical analysis.

By Lemma 2, we have that for each fixed state s,

$$\begin{aligned} &\eta_m^{-1}\cdot\Big(\mathrm{KL}\big(\pi_{m+1}(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big) + \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big)\Big) \\ &\leq \Big\langle Q^{\pi_{\beta_m},\lambda_m}(s,\cdot), \pi^*(\cdot|s) - \pi_{m+1}(\cdot|s)\Big\rangle. \end{aligned} \tag{43}$$

The left-hand side (LHS) of inequality (43) can be decomposed as

$$\begin{aligned} \mathrm{LHS} = & \eta_m^{-1}\cdot\Big(\mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_{m+1}}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big)\Big) \\ & + \eta_m^{-1}\cdot\Big(\underbrace{\mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\tilde{\pi}_{m+1}(\cdot|s)\big)}_{Q\text{-function evaluation error}}\Big) \\ & + \eta_m^{-1}\cdot\Big(\underbrace{\mathrm{KL}\big(\pi^*(\cdot|s)\|\tilde{\pi}_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_{m+1}}(\cdot|s)\big)}_{\text{policy projection error}}\Big) + \eta_m^{-1}\cdot\mathrm{KL}\big(\pi_{m+1}(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big). \end{aligned}$$

Note that the second part of the above equation represents the gap of KL-divergence due to inaccurate evaluation of Q-function; while the third part is due to the projection to the parameterized policy space. For the policy evaluation error, we have

$$\mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\tilde{\pi}_{m+1}(\cdot|s)\big)$$

$$
\begin{aligned}
=&\Big\langle \pi^*(\cdot|s), \log\Big(\frac{\tilde\pi_{m+1}(\cdot|s)}{\pi_{m+1}(\cdot|s)}\Big)\Big\rangle\\
=&\Big\langle \pi^*(\cdot|s)-\tilde\pi_{m+1}(\cdot|s), \log\Big(\frac{\tilde\pi_{m+1}(\cdot|s)}{\pi_{m+1}(\cdot|s)}\Big)\Big\rangle + \Big\langle \tilde\pi_{m+1}(\cdot|s), \log\Big(\frac{\tilde\pi_{m+1}(\cdot|s)}{\pi_{m+1}(\cdot|s)}\Big)\Big\rangle\\
=&\Big\langle \pi^*(\cdot|s)-\tilde\pi_{m+1}(\cdot|s), \log\Big(\frac{\tilde\pi_{m+1}(\cdot|s)}{\pi_{m+1}(\cdot|s)}\Big)\Big\rangle + \mathrm{KL}(\tilde\pi_{m+1}(\cdot|s)\|\pi_{m+1}(\cdot|s)).
\end{aligned}
$$

Note that by definition, we have

$$
\log\Big(\frac{\tilde\pi_{m+1}(\cdot|s)}{\pi_{m+1}(\cdot|s)}\Big) = \log(Z_{m+1}/\tilde Z_{m+1}) - \eta_m\cdot\big(Q^{\pi_{\beta_m},\lambda_m}(s,\cdot) - Q^{\lambda_m}_{\alpha_m}(s,\cdot)\big),
$$

where $Z_{m+1}, \tilde Z_{m+1}$ are the normalization constants of policies π_{m+1} and $\tilde\pi_{m+1}$. Since the KL-divergence is non-negative, we have

$$
\mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\tilde\pi_{m+1}(\cdot|s)\big) \geq -\eta_m\cdot\big\langle \pi^*(\cdot|s)-\tilde\pi_{m+1}(\cdot|s), \big(Q^{\pi_{\beta_m},\lambda_m}(s,\cdot) - Q^{\lambda_m}_{\alpha_m}(s,\cdot)\big)\big\rangle.
$$

To simplify notation, we denote

$$
\epsilon_m(s) := -\langle \pi^*(\cdot|s)-\tilde\pi_{m+1}(\cdot|s), (Q^{\pi_{\beta_m},\lambda_m}(s,\cdot) - Q^{\lambda_m}_{\alpha_m}(s,\cdot))\rangle.
$$

For the policy projection error, we have

$$
\begin{aligned}
&\mathrm{KL}\big(\pi^*(\cdot|s)\|\tilde\pi_{m+1}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_{m+1}}(\cdot|s)\big)\\
=&\Big\langle \pi^*(\cdot|s), \log\Big(\frac{\pi_{\beta_{m+1}}(\cdot|s)}{\tilde\pi_{m+1}(\cdot|s)}\Big)\Big\rangle\\
=&\Big\langle \pi^*(\cdot|s)-\pi_{\beta_{m+1}}(\cdot|s), \log\Big(\frac{\pi_{\beta_{m+1}}(\cdot|s)}{\tilde\pi_{m+1}(\cdot|s)}\Big)\Big\rangle + \mathrm{KL}(\pi_{\beta_{m+1}}\|\tilde\pi_{m+1})\\
\geq&\big\langle \pi^*(\cdot|s)-\pi_{\beta_{m+1}}(\cdot|s), f_{\beta_{m+1}}(s,\cdot) - (f_{\beta_m}(s,\cdot) - \eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,\cdot))\big\rangle
\end{aligned}
$$

Here, the last inequality holds since $\pi_{\beta_{m+1}}(\cdot|s)\propto\exp\{f_{\beta_{m+1}}(s,\cdot)\}$ and

$$
\tilde\pi_{m+1}(\cdot|s)\propto\exp\{f_{\beta_m}(s,\cdot)\}\cdot\exp\{-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,\cdot))\}.
$$

To simplify notation, we denote

$$
\delta_m(s) := \big\langle \pi^*(\cdot|s)-\pi_{\beta_{m+1}}(\cdot|s), f_{\beta_{m+1}}(s,\cdot) - (f_{\beta_m}(s,\cdot) - \eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,\cdot))\big\rangle.
$$

Then, by inequality (43), we have

$$
\begin{aligned}
\Big\langle Q^{\pi_{\beta_m},\lambda_m}(s,\cdot), \pi^*(\cdot|s)-\pi_{m+1}(\cdot|s)\Big\rangle \geq&\eta_m^{-1}\cdot\Big(\mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_{m+1}}(\cdot|s)\big) - \mathrm{KL}\big(\pi^*(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big)\Big)\\
&+\eta_m^{-1}\cdot\mathrm{KL}\big(\pi_{m+1}(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big) + \epsilon_m(s) + \eta_m^{-1}\cdot\delta_m(s).
\end{aligned}
\tag{44}
$$

Next, following the proof of Lemma 1, we can show that

$$
\begin{aligned}
&\Big\langle Q^{\pi_{\beta_m},\lambda_m}(s,\cdot), \pi_{\beta_m}(\cdot|s)-\pi_{m+1}(\cdot|s)\Big\rangle\\
\leq&\frac{\eta_m}{8}\cdot\Big(\sup_{s\in\mathcal S}\sup_{a\in\mathcal A}|Q^{\pi_{\beta_m},\lambda_m}(s,a)|\Big)^2 + \eta_m^{-1}\cdot\mathrm{KL}\big(\pi_{m+1}(\cdot|s)\|\pi_{\beta_m}(\cdot|s)\big),
\end{aligned}
\tag{45}
$$

and

$$
\mathbb{E}_{s\sim\nu_s^{\pi^*}}\Big[\Big\langle Q^{\pi_{\beta_m},\lambda_m}(s,\cdot), \pi^*(\cdot|s)-\pi_{\beta_m}(\cdot|s)\Big\rangle\Big] = (1-\gamma)\cdot\big(L(\pi^*,\lambda_m) - L(\pi_{\beta_m},\lambda_m)\big).
\tag{46}
$$

Combining (44) - (46), we have

$$L(\pi^*,\lambda_m)-L(\pi_{\beta_m},\lambda_m)\geq\frac{1}{(1-\gamma)\eta_m}\Big(\Phi^{\pi^*}(\pi^*\|\pi_{\beta_{m+1}})-\Phi^{\pi^*}(\pi^*\|\pi_{\beta_m})\Big)$$
$$-\frac{\eta_m}{8(1-\gamma)}\Big(\sup_{s\in\mathcal{S}}\sup_{a\in\mathcal{A}}|Q^{\pi_{\beta_m},\lambda_m}(s,a)|\Big)^2+\mathbb{E}_{s\sim\nu_s^{\pi^*}}\big[\epsilon_m(s)+\eta_m^{-1}\cdot\delta_m(s)\big].$$

□

Proof of Theorem 2 With the bounds in Lemma 5, the proof of Theorem 2 follows exactly the same lines of argument as the proof of Theorem 1. Specifically, for the constraints violation, similar to (39), we have

$$\big\|[\partial_\lambda L(\bar\pi_T,\lambda)]^+\big\|\leq\frac{\max_{\lambda\in\Lambda_M}\|\lambda-\lambda_0\|^2}{2r\cdot\sum_{m=0}^{T-1}\eta_m}+G^2\frac{\sum_{m=0}^{T-1}\eta_m^2}{r\cdot\sum_{m=0}^{T-1}\eta_m}+\frac{1}{r}\cdot\sum_{m=0}^{T-1}\tilde\eta_m\cdot\big(L(\pi_{\beta_m},\lambda_m)-L^*\big).$$

By the saddle point property and Lemma 5, we also have

$$\begin{aligned}\sum_{m=0}^{T-1}\tilde\eta_m\cdot\big(L(\pi_{\beta_m},\lambda_m)-L^*\big)\leq&\sum_{m=0}^{T-1}\tilde\eta_m\cdot\big(L(\pi_{\beta_m},\lambda_m)-L(\pi^*,\lambda_m)\big)\\ \leq&\Big((1-\gamma)\cdot\sum_{m=0}^{T-1}\eta_m\Big)^{-1}\cdot\Big(\frac{G^2}{8}\cdot\sum_{m=0}^{T-1}\eta_m^2+\Phi^{\pi^*}(\pi^*\|\pi_0)\Big)\\ &+\sum_{m=0}^{T-1}\tilde\eta_m\cdot\mathbb{E}_{s\sim\nu_s^{\pi^*}}\big[\big|\epsilon_m(s)+\eta_m^{-1}\cdot\delta_m(s)\big|\big].\end{aligned}$$

Hence, when we choose constant stepsize $\eta_m=\eta$, we have

$$\begin{aligned}\big\|[\partial_\lambda L(\bar\pi_T,\lambda)]^+\big\|\leq&\Big((M+r)^2+\frac{1}{1-\gamma}\Phi^{\pi^*}(\pi^*\|\pi_{\beta_0})\Big)\frac{1}{rT\eta}+\Big(1+\frac{1}{8(1-\gamma)}\Big)\frac{G^2\eta}{r}\\ &+\frac{1}{T}\sum_{m=0}^{T-1}\mathbb{E}_{s\sim\nu_s^{\pi^*}}\big[\epsilon_m(s)+\eta^{-1}\cdot\delta_m(s)\big]\\ =&O\Big(\frac{1}{T\eta}+\eta\Big)+\frac{1}{T}\sum_{m=0}^{T-1}\mathbb{E}_{s\sim\nu_s^{\pi^*}}\mathbb{E}_{s\sim\nu_s^{\pi^*}}\big[\big|\epsilon_m(s)+\eta^{-1}\cdot\delta_m(s)\big|\big].\end{aligned}$$

Similarly, for the optimality gap of the primary cost $C(\bar\pi_T)-L^*$, we have

$$\big|C(\bar\pi_T)-L^*\big|\leq O\Big(\frac{1}{T\eta}+\eta\Big)+\frac{1}{T}\sum_{m=0}^{T-1}\mathbb{E}_{s\sim\nu_s^{\pi^*}}\big[\big|\epsilon_m(s)+\eta^{-1}\cdot\delta_m(s)\big|\big].$$

We next analyze the error term $\mathbb{E}_{s\sim\nu_s^{\pi^*}}[|\epsilon_m(s)|$ and $\mathbb{E}_{s\sim\nu_s^{\pi^*}}[|\delta_m(s)|]$. For the policy evaluation error, by definition, we have

$$\begin{aligned}\mathbb{E}_{s\sim\nu_s^{\pi^*}}[|\epsilon_m(s)|]=&\mathbb{E}_{s\sim\nu_s^{\pi^*}}[\langle\pi^*(\cdot|s)-\tilde\pi_{m+1}(\cdot|s),(Q^{\pi_{\beta_m},\lambda_m}(s,\cdot)-Q^{\lambda_m}_{\alpha_m}(s,\cdot))\rangle]\\ \leq&\mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_m}}}\Big[\frac{d\nu^{\pi^*}(s,a)}{d\nu^{\pi_{\beta_m}}(s,a)}\cdot\Big|Q^{\pi_{\beta_m},\lambda_m}(s,a)-Q^{\lambda_m}_{\alpha_m}(s,a)\Big|\Big]\\ &+\mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_m}}}\Big[\frac{d\nu_s^{\pi^*}(s)}{d\nu_s^{\pi_{\beta_m}}(s)}\cdot\frac{\tilde\pi_{m+1}(a|s)}{\pi_{\beta_m}(a|s)}\cdot\Big|Q^{\pi_{\beta_m},\lambda_m}(s,a)-Q^{\lambda_m}_{\alpha_m}(s,a)\Big|\Big].\end{aligned}$$

By Assumptions 5 and 7, the first term in the last inequality can be upper bounded by $C_\ell\epsilon$. For the second term, note that

$$\frac{\tilde\pi_{m+1}(a|s)}{\pi_{\beta_m}(a|s)}=\tilde Z_{m+1}^{-1}(s)\cdot\exp\{-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,a)\}\leq\tilde Z_{m+1}^{-1}(s)\cdot\exp\{\eta_m\cdot G\},$$

where the normalization constant

$$\tilde{Z}_{m+1}(s) = \sum_{a\in\mathcal{A}} \pi_{\beta_m}(a|s)\cdot \exp\big\{-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,a)\big\} \geq \exp\{-\eta_m\cdot G\}.$$

Thus, the second term is upper bounded by $C_\ell \exp(2\eta_m G)\epsilon$. Then, we have

$$\mathbb{E}_{s\sim\nu_s^{\pi^*}}[|\epsilon_m(s)|] \leq C_\ell\big(1+\exp\{2\eta\cdot G\}\big)\cdot\epsilon = O(\epsilon).$$

Similarly, for the policy projection error, by Assumptions 6 and 7, we have

$$\begin{aligned}\mathbb{E}_{s\sim\nu_s^{\pi^*}}[|\delta_m(s)|] =& \mathbb{E}_{s\sim\nu_s^{\pi^*}}\Big[\big|\big\langle \pi^*(\cdot|s)-\pi_{\beta_{m+1}}(\cdot|s), f_{\beta_{m+1}}(s,\cdot)-(f_{\beta_m}(s,\cdot)-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,\cdot))\big\rangle\big|\Big]\\ =& \mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_m}}}\Big[\frac{d\nu^{\pi^*}(s,a)}{d\nu^{\pi_{\beta_m}}(s,a)}\cdot\Big|f_{\beta_{m+1}}(s,a)-(f_{\beta_m}(s,a)-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,a))\Big|\Big]\\ &+\mathbb{E}_{(s,a)\sim\nu^{\pi_{\beta_m}}}\Big[\frac{d\nu_s^{\pi^*}(s)}{d\nu_s^{\pi_{\beta_m}}(s)}\cdot\Big|f_{\beta_{m+1}}(s,a)-(f_{\beta_m}(s,a)-\eta_m\cdot Q^{\lambda_m}_{\alpha_m}(s,a))\Big|\Big]\\ \leq& 2C_\ell\delta.\end{aligned}$$

Above all, we have

$$\begin{aligned}\big\|[\partial_\lambda L(\bar{\pi}_T,\lambda)]^+\big\|, \big|C(\bar{\pi}_T)-L^*\big| &\leq O\Big(\frac{1}{T\eta}+\eta\Big)+\frac{1}{T}\sum_{m=0}^{T-1} C_\ell\Big((1+\exp\{2\eta\cdot G\})\cdot\epsilon+\frac{2\delta}{\eta}\Big)\\ &\leq O\Big(\frac{1}{T\eta}+\eta+\epsilon+\frac{\delta}{\eta}\Big).\end{aligned}$$

□

B. Value Function Approximation for Queue Scheduling

In this section, we provide the details of value function approximation used in solving the queue scheduling problems in Section 6.2. The approximation scheme we employ is based on the development in Dai and Shi (2019). There are two key ingredients in the approximation.

1. Relative value function approximation: In the regularized policy iteration step, given a policy π, we first need to estimate the associated relative value function $\bar{V}^{\pi,\lambda}(s)$ for all $s\in\mathcal{S}$, where the state

$$s=(x_1,\dots,x_I,z_1,\dots,z_J),$$

x_i denotes the length of queue i and z_j denotes the number of customers in pool j. Due to the large state space, we approximate the relative value function with a linear parameterization:

$$\bar{V}^{\pi,\lambda}(s)\approx\langle\phi(s),\theta^{\pi,\lambda}\rangle.$$

where $\phi(\cdot)\in\mathbb{R}^K$ denotes the basis functions. Given a proper class of basis functions, we use the least-square temporal difference (LSTD) algorithm to find the optimal coefficients, $\theta^{\pi,\lambda}$. In particular, we solve for the optimal $\theta^{\pi,\lambda}$ that minimizes a weighted L_2 distance between $\bar{V}^{\pi,\lambda}(s)$ and $\langle\phi(s),\theta^{\pi,\lambda}\rangle$ (Puterman 2014, Bertsekas 2011). The details can be found in Algorithm 3.

As for the basis functions, following Veatch (2005), Moallemi et al. (2008), Dai and Shi (2019), we adopt the quadratic bases:

$$\{x_i,x_i^2\}_{1\leq i\leq I},\{z_j,z_j^2\}_{1\leq j\leq J},\{x_jz_k\}_{1\leq i\leq I,1\leq j\leq J},\{x_jx_k\}_{1\leq j,k\leq I}\text{ and }\{z_jz_k\}_{1\leq j,k\leq J},$$

Algorithm 3 LSTD Algorithm

```
Input: Simulator of the multi-class and multi-pool queueing network, policy $\pi$, instantaneous cost function $c^{\lambda}(\cdot,\cdot)$, basis functions $\phi(\cdot) \in \mathbb{R}^K$, initial state $s_0$, and simulation horizon $H$.
for $t = 0, \ldots, H$ do
    Generate the action $a_t$ under policy $\pi$.
    Generate the next state $s_{t+1}$ conditional on $(s_t, a_t)$.
end for
Estimate long-run average cost: $\hat{C}^{\pi,\lambda} = \frac{1}{H}\sum_{t=0}^{H-1} c^{\lambda}(s_t, a_t)$.
Calculate: $\hat{A}_H = \frac{1}{H}\sum_{t=0}^{H-1}\phi(s_t)\big(\phi(s_t) - \phi(s_{t+1})\big)^{\top}$, $\hat{b}_H = \frac{1}{H}\cdot\sum_{t=0}^{H-1}\phi(s_t)\big(c^{\lambda}(s_t,a_t) - \hat{C}^{\pi,\lambda}\big)$.
Output: $\hat{\theta} = \hat{A}_H^{-1}\hat{b}_H$.
```

which is motivated by the fluid approximation of the MDP (Dai and Shi 2019).

2. Relative action-value function evaluation: To execute the regularized policy iteration, we need to calculate the relative action-value function $\bar{Q}^{\pi,\lambda}(s,a)$ defined in (22). After we obtain $\theta^{\pi,\lambda}$, we can approximate $\bar{Q}^{\pi,\lambda}(s,a)$ via

$$\bar{Q}^{\pi,\lambda}(s,a) \approx \hat{Q}^{\pi,\lambda}(s,a) := c^{\lambda}(s,a) - \bar{C}^{\pi,\lambda} + \sum_{s' \in \mathcal{S}} \langle \phi(s'), \theta^{\pi,\lambda}\rangle \cdot P(s'|s,a).$$

Since a constant shift in the relative action-value function does not change the regularized policy iteration, we drop the constant term $\bar{C}^{\pi,\lambda}$. We next develop a closed-from expression for $\sum_{s'\in\mathcal{S}} \phi(s')\cdot P(s'|s,a)$ under the quadratic basis. Recall that the transition dynamics of the queueing system takes the form:

$$x_i' = x_i - (z_1 + \cdots + z_J) + \delta x_i \;\; i =, 1, \ldots, I$$
$$z_j' = z_j + u_j - \delta d_j, \;\; j =, 1, \ldots, J.$$

where the new arrivals $\delta x_i \sim$Poisson(Λ_i) and the new departures $\delta d_j \sim$Binomial$(z_j + u_j, \mu_j)$. Then,

$$\mathbb{E}[x_i'|s,a] = x_i - (a_1 + \cdots + a_J) + \Lambda_i, \;\; \mathbb{E}[z_j'|s,a] = (z_j + a_j)\cdot(1-\mu_j)$$
$$\mathbb{E}[x_i' z_j'|s,a] = (x_i - (a_1 + \cdots + a_J) + \Lambda_i)\cdot(z_j + a_j)\cdot(1-\mu_j),$$
$$\mathbb{E}[{x_i'}^2|s,a] = (x_i - (a_1 + \cdots + a_J) + \Lambda_i)^2 + 2\Lambda_i\cdot(x_i - (a_1 + \cdots + a_J) + \Lambda_i + \Lambda_i^2$$
$$\mathbb{E}[{z_j'}^2|s,a] = (z_j + a_j)\cdot(1-\mu_j)\cdot\mu_j + (z_j + a_j)^2\cdot(1-\mu_j)^2, \;\; \mathbb{E}[z_j'{z_k'}^2|s,a] = (z_j + a_j)(z_k + a_k)(1-\mu_j)(1-\mu_k).$$

C. Value Function Approximation for Inventory Management

In this section, we provide the details of approximations we use to solve the inventory management problems in Section 6.2. We use a multi-layer perceptron with ReLU activation function to approximate the Q-function $Q(s,a)$. This is a classic network architecture in deep learning (LeCun et al. 2015). It contains 4 hidden layers and 1 output layer. It takes the state vector s as input and outputs a 10-dimensional vector, whose i-th component corresponds to the value of $Q(s,i)$, $i = 0, 1, \cdots, 9$. Each hidden layer contains 32 nodes (see Figure 5 for a pictorial illustration). In addition to the Q-function approximation, we use a separate neural

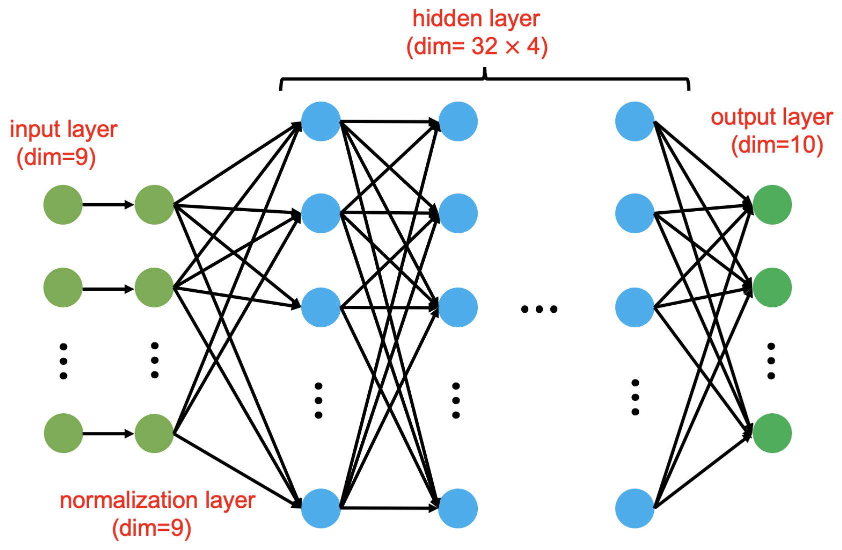


Figure 5 **Structure of neural network used in inventory experiment**

network with the same architecture to represent the policy. There, we apply an additional softmax transform at the output layer so that it generates a probability distribution.

To ensure training stability, we use “state normalization”. In particular, we first simulate the system using a randomized policy that takes feasible actions uniformly at random. We then estimate the mean and standard deviation of the state component-wise using the simulated trajectory. Before we feed the state vector s into the neural network, we first normalize it via $(s(k)-\hat{\mu}(k))/\hat{\sigma}(k)$, where $s(k)$ denotes the k-th component of s, and $\hat{\mu}(k)$ and $\hat{\sigma}(k)$ are the corresponding estimated mean and standard deviation (calculated based on the simulated sample path). We use the same method to normalize the costs as well.

For policy evaluation, we apply the temporal difference (TD) learning algorithm to find the optimal parameters of the corresponding neural network (Chapter 11 of Bertsekas (2011)), i.e., α_m (in the m-th primal-dual iteration). Algorithm 4 provides the details of an SGD-based TD algorithm, where our objective is to minimize the mean squared Bellman error. We run the SGD-based TD learning algorithm for $T=10^4$ steps with stepsize $h=0.0003$ initialized from α_{m-1}. We also use momentum to accelerate the convergence (Kingma and Ba 2014). To update the policy, we aim to find β_m (in the m-th primal-dual iteration) that minimizes the KL divergence between the parameterized policy and the updated policy:

$$J(\beta)=\mathbb{E}_{s\sim\nu_s^{\pi_{\beta_{m-1}}}}\Big[\mathrm{KL}\big(\pi_\beta(\cdot|s)\big\|Z_m^{-1}\cdot\exp(-\eta_m\cdot Q_{\alpha_m}^{\lambda_m}(s,\cdot))\big)\Big],$$

where $Q_{\alpha_m}^{\lambda_m}$ denotes the approximated Q-function in the m-th iteration. In implementation we run stochastic gradient descent for $T=10^4$ steps with stepsize $h=0.0003$, initialized from β_{m-1}. See Algorithm 5 for more details.

Algorithm 4 SGD-based TD learning

```
Input: target policy $\pi_{\beta_{m-1}}$, stepsize $h$, initial state $s_0$, initial parameter $\alpha_0$, and computational budget $T$.
for $t=0,\ldots,T$ do
    Sample $a_t \sim \pi_{\beta_{m-1}}$ and $s_{t+1} \sim P(\cdot|s_t,a_t)$.
    Update parameter
    $$\alpha^{t+1} = \alpha^t - h \cdot \Big(Q^{\lambda_m}_{\alpha^t}(s_t,a_t) - \big((1-\gamma)c^{\lambda_m}(s,a) + \gamma \mathbb{E}_{a\sim\pi_{\beta_{m-1}}(\cdot|s_{t+1})}[Q^{\lambda_m}_{\alpha^t}(s_{t+1},a)]\big)\Big) \cdot \nabla_\alpha Q^{\lambda_m}_{\alpha^t}(s_t,a_t).$$
end for
Output: $\alpha_m = \alpha^T$.
```

Algorithm 5 SGD-based policy update

```
Input: target $Q$-function $Q^{\lambda_m}_{\alpha_m}(s,\cdot)$, stepsize $h$, initial weight $\beta_0$, temperature parameter $\eta_{m-1}$, and computational budget $T$.
for $t=0,\ldots,T$ do
    Sample $a_t \sim \pi_{\beta_{m-1}}$ and $s_{t+1} \sim P(\cdot|s_t,a_t)$.
    Update parameter
    $$\beta^{t+1} = \beta^t + h \cdot \Big(2\log(\pi_{\beta_t}(s_t,a_t)) + \eta_{m-1} \cdot Q^{\lambda_m}_{\alpha_m}(s_t,a_t)\Big) \cdot \nabla_\beta \pi_{\beta_t}(s_t,a_t).$$
end for
Output: $\beta_m = \beta^T$.
```