A Primal-Dual Approach to Constrained Markov Decision Processes

Yi Chen
Department of Industrial Engineering & Management Sciences, Northwestern University, Evanston, IL, 60208

Jing Dong
Columbia Business School, New York City, NY, 12007

Zhaoran Wang
Department of Industrial Engineering & Management Sciences, Northwestern University, Evanston, IL, 60208

In many operations management problems, we need to make decisions sequentially to minimize the cost while satisfying certain constraints. One modeling approach to study such problems is constrained Markov decision process (CMDP). When solving the CMDP to derive good operational policies, there are two key challenges: one is the prohibitively large state space and action space; the other is the hard-to-compute transition kernel. In this work, we develop a sampling-based 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 $O(\log(T)/\sqrt{T})$, where $T$ is the number of iterations. When the CMDP has a weakly coupled structure, our approach can substantially reduce the dimension of the problem through an embedded decomposition. We apply the algorithm to two important applications with weakly coupled structures: multi-product inventory management and multi-class queue scheduling, and show that it generates controls that outperform state-of-art heuristics.

Key words: Constrained Markov decision process, primal-dual algorithm, weakly coupled Markov decision process

1. Introduction

In many sequential decision-making problems, a single utility might not suffice to describe the real objectives faced by the decision-makers. A natural approach to study such problems is to optimize one objective while putting constraints on the others. In this context, the constrained Markov decision process (CMDP) has become an important modeling tool for sequential multi-objective decision-making problems under uncertainty. A CMDP aims to minimize one type of cost while keeping the other costs below certain thresholds. It has been successfully applied to analyze various important applications, including admission control and routing in telecommunication networks, scheduling for hospital admissions, and maintenance scheduling for infrastructures (Altman 1999). Due to the complicated system dynamics and the scale of the problem, exact optimal solutions to CMDPs can rarely be derived. Instead, numerical approximations become the main workhorse...
to study CMDPs. In this paper, we propose a sampling-based primal-dual algorithm that can efficiently solve a wide range of CMDPs.

One basic approach to solve the CMDP is to use a linear programming (LP) formulation based on the occupancy measure. This approach faces two key challenges in implementations: it requires knowledge of the transition kernel of the underlying dynamical system explicitly; it does not scale well as the state space and action space get large. An alternative approach is to apply the Lagrangian duality. In particular, by dualizing the constraints and utilizing strong duality, we can translate the CMDP into a max-min problem, where for a given Lagrangian multiplier, the inner minimization problem is just a standard Markov decision process (MDP). This approach allows us to solve the inner problem using standard dynamic programming based methods. It does not require direct knowledge of the transition kernel as long as we can estimate the value functions from simulated or empirical data. In implementations, one would iteratively update the MDP policy and the Langrangian multiplier. The current development of this approach requires solving the MDP to get the optimal policy for each updated Langrangian multiplier (see, for example, Le et al. (2019), Miryoosfet al. (2019)), which can be computationally costly. A more natural idea is to solve the MDP only approximately at each iteration. In this paper, we investigate this idea and show that at each iteration, we only need to do one iteration of policy update to achieve the optimal convergence rate (in terms of the number of primal-dual iterations). Compared to the existing algorithms utilizing Lagrangian duality, our primal-dual algorithm can be run at a much lower cost at each iteration. We also demonstrate that our algorithm can be easily combined with many other approximate dynamic programming techniques, such as Monte Carlo policy evaluation, TD-learning, and value function approximations (Sutton and Barto 2018).

A key ingredient of our algorithm is regularized policy iteration. The standard policy iteration includes two steps: policy evaluation and policy improvement. The policy evaluation step calculates the action-value function under a given policy. Then, the policy improvement step defines a new policy by taking the action that minimizes the action-value function. Through a Kullback-Leibler (KL) regularization term, the regularized policy iteration modifies the policy improvement step by reweighing the probability of taking each action via a softmax transformation of the action-value function. This modification allows us to view the policy update step as running mirror descent for the objective function in the policy space (Nemirovks 2012). In addition, we update the Lagrangian multiplier using subgradient ascent, which also belongs to the family of mirror descent methods. This unified viewpoint makes the improved primal-dual algorithm possible. Noticeably, many recent developments in reinforcement learning also benefit from regularization, which has been shown to improve exploration and robustness. For example, Trust Region Policy Optimization and Proximal Policy Optimization use KL divergence between two consecutive policies as a penalty in policy
improvement (Schulman et al. 2015, 2017). Soft-Q-learning uses Shannon entropy as a penalty in value iteration (Haarnoja et al. 2017). Geist et al. (2019) propose a unified framework to analyze the above algorithms via regularized Bellman operator (see also Liu et al. (2019a), Shani et al. (2019), Wang et al. (2019) for convergence analysis of regularized policy iteration).

In terms of applications of the algorithm, we study an important class of CMDPs which we refer to as weakly coupled CMDPs (Singh and Cohn 1998). A weakly coupled CMDP comprises multiple sub-problems that are independent except for a collection of coupling constraints. Due to the linking constraints, the scale of problem grows exponentially in the number of sub-problems. Hence, even in the case where each sub-problem is computationally tractable, it can be computationally prohibitive to solve the joint problem. Our primal-dual algorithm naturally helps break the curse of dimensionality in this case. In particular, the weakly coupled CMDP can be decomposed into independent sub-problems in the policy iteration step. In this case, the complexity only grows linearly with the number of sub-problems. We also comment that the weakly coupled CMDP can be viewed as a Lagrangian relaxation of the weakly coupled MDP (Adelman and Mersereau 2008). Even though there is a relaxation gap between the two, as we will demonstrate in our numerical experiments, the (modified) policy obtained via CMDP can perform very well for the original MDP problem in the applications we considered.

We apply the primal-dual algorithm to solve two classical operations management problems: inventory planning and queue scheduling. For the inventory planning problem, we consider a multi-product newsvendor problem with budget constraints (Turken et al. 2012). We formulate this problem as a weakly coupled CMDP and study a small-scale instance where we can numerically solve for the optimal policy. We show that our policy can indeed achieve $O(\log(T)/\sqrt{T})$ convergence in this case, where $T$ is the number of iterations. For the queue scheduling problem, we consider a multi-class multi-pool parallel-server system where the decision-maker needs to route different classes of customers to different pools of servers in order to minimize the performance cost (holding cost plus overflow cost). We allow the service rates to be both customer-class and server-pool dependent. Since each pool only has a finite number of servers, the routing policy needs to satisfy the capacity constraints. This optimal scheduling problem can be formulated as a weakly coupled MDP. We consider instances where it is prohibitive to solve for the optimal policy. Applying the Lagrangian relaxation, we solve the resulting weakly coupled CMDP by combining our primal-dual algorithm with value function approximation techniques. We show that our method generates comparable or even better policies than the state-of-art policies.

1.1. Literature review

In this section, we review some of the existing methods/results to solve CMDPs. The goal is to clearly state the contribution of our work. Most existing algorithms for CMDPs is adapted
from methods for MDPs, and can be roughly divided into three categories: LP based approaches, dynamic programming based approaches (including policy iteration and value iteration), and the policy gradient methods.

One LP based approach utilizes the occupation measure, which is the weighted proportion of time the system spends at each state-action pair. The objective and constraints can be written as the inner products of instantaneous cost functions and occupation measure. The other LP based approach utilizes the dynamic programming principle and treats the value function (defined on state space) as the decision variables. In particular, the optimal value function of an MDP is the largest super-harmonic function that satisfies certain linear constraints determined by transition dynamics. For the CMDP, we obtain an LP by combing the dynamic programming principle with the Lagrangian duality. These two LP formulations are dual of each other (Altman 1999). Many works on LP based approaches aim to find efficient ways to solve the LPs by exploiting the structures of some specific MDPs/CMDPs. For example, Bertsimas and Orlin (1994) use the ellipsoid method to derive efficient algorithms for problems with side constraints, including the traveling salesman problem and the vehicle routing problem. Neely (2011) studies a linear fractional programming method to solve CMDPs. More recently, Caramanis et al. (2014) propose two algorithms based on the column generation and the generalized experts framework, respectively. These algorithms are shown to be as efficient as value iteration. Another challenge of LP based approaches is that we need to know the transition kernel up front and in explicit forms. Some recent developments aim to overcome this challenge. For example, Chen and Wang (2016) reformulate the LP of an MDP as a saddle point problem and use stochastic approximation to solve it. Chen and Wang (2016) combine the saddle point formulation of LP with value function approximation and develop a proximal stochastic mirror descent method to solve it. However, most existing developments in this line focus on MDPs.

The policy/value iteration stems from the Bellman operator on the value function, and converges to the optimal value function linearly. In implementations, the value function can be estimated via simulation or data. Thus this class of methods does not require knowledge of the transition kernel up front. For example, in reinforcement learning, we learn the transition dynamics of the system while solving for the optimal policy (Sutton and Barto 2018). There are many works that apply policy/value iteration to solve CMDPs. For example, Gattami (2019) formulates the CMDP as a zero-sum game and uses primal-dual $Q$-learning to solve the game. It proves the almost sure convergence of the algorithm, but does not establish the rate of convergence. Le et al. (2019) study CMDPs in the offline learning setting, and combine various dynamic programming techniques with Lagrangian duality to solve it. Miryoosefi et al. (2019) extend the constraints of CMDPs to nonlinear forms and propose to solve it via Lagrangian duality and $Q$-learning as well. An $O(1/\sqrt{T})$
rate of convergence is obtained in both Le et al. (2019) and Miryoosefi et al. (2019). However, their algorithms require solving for the optimal policy at each updated Lagrangian multiplier. Our method can be viewed as an improved version of their algorithms. In particular, our algorithm only requires one policy iteration for each updated Lagrangian multiplier while achieving the same convergence rate. We comment that Le et al. (2019) and Miryoosefi et al. (2019) consider more complicated settings than the classical CMDPs studied in this paper. It would be interesting to extend our primal-dual algorithm to solve the more general problems.

Both LP based methods and policy/value iteration can suffer from the curse of dimensionality when dealing with a large action space. The policy gradient based approaches alleviate the dimensionality issue by approximating the policy using a parametric family of functions. In this case, searching over the policy space reduces to a finite dimensional optimization problem. Several works combine this idea with Lagrangian duality to solve large-scale CMDPs. For example, Borkar (2005) and Bhatnagar and Lakshmanan (2012) combine actor-critic algorithms with policy function approximations. Tessler et al. (2018) use two-timescale stochastic approximation. Beyond duality, several other policy gradient based approaches to solve CMDPs have been developed. For example, Achiam et al. (2017) propose a trust region method that focuses on safe exploration. Liu et al. (2019b) develop an interior point method with logarithmic barrier functions. Chow et al. (2018) propose to use Lyapunov functions to handle constraints. However, the key challenge of policy gradient based methods to solve CMDPs is that the corresponding optimization problems are non-convex. In most cases, only convergence to a local minimum can be guaranteed and the convergence rates are often hard to establish.

1.2. Organization of the paper and notations

The paper is organized as follows. We first introduce the CMDP and review some classical results that are relevant to our subsequent development in Section 2. We then introduce our algorithm in Section 3 and show that the algorithm achieves the optimal convergence rate in Section 4. In Section 5, we discuss how our algorithm can be applied to (approximately) solve weakly coupled CMDPs and weakly coupled MDPs. We then implement our algorithm to solve an inventory planning problem and a queue scheduling problem in Sections 6 and 7 respectively. Lastly, we conclude the paper and discuss some interesting future directions in Section 8.

The following notations are used throughout the paper. For a positive integer $K$, we denote $[K]$ as the set $\{1, 2, \ldots, K\}$. 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 $a \leq b$ if the inequality holds for each coordinate, i.e., $[a]_k \leq [b]_k \forall k \in [K]$. Given a vector $x \in \mathbb{R}^K$, $[x]^+ = (\max\{[x]_1, 0\}, \ldots, \max\{[x]_K, 0\})$. 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 \( b_n \leq Ca_n, b_n \geq C'a_n, \) and \( C'a_n \leq b_n \leq Ca_n, \) respectively. We also introduce the \( \tilde{O}(\cdot) \) notation when we ignore the logarithmic factors. For example, if \( b_n \leq Ca_n \cdot \log(n) \), we denote it by \( b_n = \tilde{O}(a_n) \).

2. Constrained Markov Decision Process

We start by considering a discrete-time MDP characterized by the tuple \((S, A, P, \gamma, \mu_0)\). Here, \( S \) and \( A \) denote the state and action spaces; \( P = \{P(\cdot|s,a)\}_{(s,a)\in S \times A} \) 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 conditioning on being in state \( s \) and taking action \( a \). Function \( c = \{c(s,a)\}_{(s,a)\in S \times A} \) 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 \( \mu_0 = \{\mu_0(s)\}_{s \in S} \) are the discount rate and the distribution of the initial state, respectively. Given an MDP \((S, A, P, \gamma, \mu_0)\), a policy \( \pi \) determines what action to take at each state. We define the expected cumulative discounted cost with initial state \( s_0 \) under policy \( \pi \) as

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

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 \( \pi \). We further weight the expected costs according to the initial state distribution and define

\[
C(\pi) = \mathbb{E}_{s_0 \sim \mu_0} \left[ V^\pi(s_0) \right].
\]

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

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

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

In order to make the expression more 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 remark that the CMDP is only one modeling choice to model 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. CMDP is also closely connected to an important class of MDPs – weakly coupled MDP. In particular, CMDPs can be viewed as a relaxation of weakly coupled MDPs [Adelman and Mersereau 2008]. We will provide more discussions about this in Section 5.
2.1. Policy Spaces

Solving a CMDP requires finding the optimal policy over a properly defined policy space, which is a function space. Imposing suitable regularity conditions on the policy space will facilitate the development of algorithms. We next introduce a few classes of commonly used policies. 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 referred to as a “Markov 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 $\Pi, \Pi_M, \Pi_S, \Pi_D$ to denote the space of behavior, Markov, stationary, and stationary deterministic policies, respectively.

Given an arbitrary policy space $U$, we can further generate a new type of policy called “mixing policies” via an initial randomization. Specifically, let $\rho$ be a probability measure on $U$. Under a mixing policy on $U$ with mixing probability $\rho$, we first draw a policy, say $\pi_g$, from $U$ following the distribution $\rho$. Then $\pi_g$ is executed for $t = 0, 1, 2, \ldots$. 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. When allowing mixing operation, we incorporate the randomness of the initial mixing into the calculation of the accumulated cost. In particular, for $\pi \in \mathcal{M}(U)$ with initial randomization $\rho$,

$$C(\pi) = (1 - \gamma) \cdot \mathbb{E}_{s_0 \sim \mu_0} \left[ \int_U \mathbb{E}^{\pi_g} \left[ \sum_{t=0}^{\infty} \gamma^t \cdot c(s_t, a_t) \mid s_0 \right] \rho(\text{d}g) \right].$$

By definition, we note that

$$\Pi \supseteq \Pi_M \supseteq \Pi_S \supseteq \Pi_D, \quad \mathcal{M}(\Pi_S) \supseteq \Pi_S.$$

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

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

For CMDPs, when the instantaneous costs $c(\cdot, \cdot)$ and $d_k(\cdot, \cdot)$ are uniformly bounded from below, $\Pi_S$ is dominating [Altman 1999]. The class of mixing stationary polices $\mathcal{M}(\Pi_S)$ is also dominating in this case (Theorem 8.4 in [Altman 1999]).
2.2. Classical Approaches to Solve CMDPs

There are two classical approaches to CMDPs. We use CMDPs with finite state and action spaces as examples. The first method utilizes the occupation measure. Given a policy \( \pi \), the occupation measure is defined as

\[
\nu^\pi(s, a) := (1 - \gamma) \cdot E_{s_0 \sim \mu_0} \left[ \sum_{t=0}^{\infty} \gamma^t P^\pi(s_t = s, a_t = a | s_0) \right],
\]

where \( P^\pi(\cdot, \cdot | s_0) \) denotes the probability measure induced by policy \( \pi \) with initial state \( s_0 \). Note that the occupation measure is the weighted long-run proportion of time that the system spends at each state-action pair. We can then express the accumulated costs in (2) and (3) as

\[
C(\pi) = \sum_{(s,a) \in S \times A} c(s, a) \cdot \nu^\pi(s, a)
\]

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

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

\[
\sum_{(s, a) \in S \times A} \nu(s, a) \left( 1(s = s') - \gamma P(s' | s, a) \right) = (1 - \gamma) \cdot \mu_0(s'), \quad \forall \ s' \in S, \quad \nu(s, a) \geq 0, \quad \forall \ (s, a) \in S \times A,
\]

where \( 1(\cdot) \) is the indicator function. Then we obtain the following LP formulation of CMDP

\[
\min_{(s,a) \in S \times A} \sum_{(s,a) \in S \times A} c(s, a) \cdot \nu(s, a)
\]

s.t. \( \{\nu(s, a)\}_{(s,a) \in S \times A} \in Q \),

\[
\sum_{(s,a) \in S \times A} d_k(s, a) \cdot \nu(s, a) \leq q_k, \quad \forall \ k \in [K].
\]

The second method utilizes the Lagrangian duality. Let \( \lambda \in \mathbb{R}^K \) denote the Lagrangian multiplier. Define

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

Then the CMDP can be equivalently formulated as \( \inf_{\pi \in \Pi_S} \sup_{\lambda \geq 0} L(\pi, \lambda) \). By Theorem 3.6 in Altman (1999), we can exchange the order of inf and sup and obtain,

\[
\inf_{\pi \in \Pi_S} \sup_{\lambda \geq 0} L(\pi, \lambda) = \sup_{\lambda \geq 0} \inf_{\pi \in \Pi_S} L(\pi, \lambda) = \sup_{\lambda \geq 0} \inf_{\pi \in \Pi_D} L(\pi, \lambda),
\]

where the last equation holds because for each fixed \( \lambda \), the inner problem is an unconstrained MDP and the optimal policy is a stationary deterministic policy. We emphasize that given the optimal
solution $\lambda^*$ to the dual problem, not every policy $\pi(\lambda^*)$ that minimizes $L(\pi, \lambda^*)$ is the optimal policy to the original CMDP. A necessary condition for $\pi(\lambda^*)$ to be optimal for the original CMDP is the complementary slackness: $[\lambda^*]_k \cdot D_k(\pi(\lambda^*)) = 0, \forall k \in [K]$.

The dual problem $\sup_{\lambda \geq 0} \inf_{\pi \in \Pi} L(\pi, \lambda)$ leads to the following LP formulation:

$$\max_{\phi, \lambda} \sum_{s \in S} \mu_0(s) \phi(s) - \sum_{k=1}^{K} [\lambda]_k q_k$$

s.t. $\phi(s) \leq (1 - \gamma) \left( c(s, a) + \sum_{k=1}^{K} [\lambda]_k d_k(s, a) \right) + \gamma \sum_{s' \in S} \phi(s') P(s'|s, a)$,

where $\phi(s)$ denotes the value function with initial state $s$. Note that (5) and (7) are dual of each other.

Various methods have been developed in the literature to solve the LPs (5) or (7). There are two main obstacles to solve the LPs in practice. First, it can be computationally prohibitive when dealing with a large state space or a large action space. Second, it requires explicit characterization of the transition kernel $P$. To overcome these difficulties, we next develop a sampling-based primal-dual algorithm to solve CMDPs.

3. The Primal-Dual Algorithm

Consider the Lagrangian dual problem

$$\sup_{\lambda \geq 0} \inf_{\pi \in \Pi} L(\pi, \lambda).$$

For each fixed $\lambda$, the inner problem is an unconstrained MDP. A natural idea is to solve the unconstrained MDP via a sampling-based method and then update the Lagrangian multipliers via subgradient ascent. Such an idea is exploited 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 single policy update at each iteration, i.e., we do not need to solve for the corresponding optimal policy at each iteration.

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

$$\inf_{\pi \in M(\Pi_S)} \sup_{\lambda \geq 0} L(\pi, \lambda) = \sup_{\lambda \geq 0} \inf_{\pi \in M(\Pi_S)} L(\pi, \lambda).$$

With the mixing operation, we can treat $C(\pi)$ and $D(\pi)$ as infinite-dimensional linear functions with respect to the distributions of initial randomization of policies in $\Pi_S$. Hence, the Lagrangian $L(\pi, \lambda)$ is a bilinear function and strong duality follows from the minimax theorem (Sion et al. 1955).
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. In particular, note that the objective and constraints are inner products of the cost functions and the occupation measures. Thus, what we need to average across are the occupation measures. However, since the mapping from the policy to the corresponding occupation measure is nonlinear, we cannot average the policy \( \pi(\cdot|s) \), i.e., the probability of taking each action at each state, directly. The mixing operation provides a simple way to average the occupation measures. In addition, given a mixing policy, under mild regularity conditions, there exists a non-mixing stationary policy that has the same occupation measure (Theorem 3.1 of Altman (1999)).

In particular, for \( \pi \in \mathcal{M}(\Pi_S) \), let \( \nu^\pi(s, a) \) be the corresponding occupation measure. Then, we can construct such a stationary policy \( \tilde{\pi} \) via

\[
\tilde{\pi}(a|s) = \frac{\nu^\pi(s, a)}{\sum_{a \in A} \nu^\pi(s, a)}.
\]  

Our algorithmic development is based on strong duality (9), which holds under certain regularity conditions (see Section 4 for details). By the minimax theorem, there exists a saddle point \((\pi^*, \lambda^*)\) such that

\[
L(\pi^*, \lambda) \leq L(\pi^*, \lambda^*) \leq L(\pi, \lambda^*), \quad \forall \lambda \in \mathbb{R}^K_+, \pi \in \mathcal{M}(\Pi_S).
\]  

Moreover, \( \pi^* \) is an optimal solution to the primal problem and \( \lambda^* \) is an optimal solution to the dual problem. In addition, \( L(\pi^*, \lambda^*) \) equals to the optimal cost of the CMDP. The saddle point property (11) suggests that we can use iterative primal-dual updates to find the saddle point.

We next introduce our actual algorithm. Note that for a fixed value of \( \lambda \), the inner inf-problem is an unconstrained MDP with modified instantaneous cost \( c^\lambda(s, a) \) such that

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

In what follows, we refer to the inner problem inf\( \pi \in \mathcal{M}(\Pi_S) \) \( L(\pi, \lambda) \) as the modified unconstrained MDP.

For a given policy \( \pi \) and Lagrangian multiplier \( \lambda \), define

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

which is known as the action-value function or Q-function. Let \( \pi_m \) and \( \lambda_m \) denote the policy and the Lagrangian multiplier obtained at iteration \( m \). For the policy update, we use KL divergence as the regularization (Geist et al. 2019). In particular, the regularized policy iteration is defined as

\[
\pi_m(a|s) = \arg\min_{\pi(\cdot|s) \in \Delta_A} \left\{ \langle Q^{\pi_m-1,\lambda_m-1}(s, \cdot), \pi(\cdot|s) \rangle + \eta_{m-1}^{-1} \cdot \text{KL}\left( \pi(\cdot|s) \| \pi_{m-1}(\cdot|s) \right) \right\},
\]  

where \( \text{KL}(\pi(\cdot|s) \| \pi_{m-1}(\cdot|s) \) is the Kullback-Leibler divergence. The Lagrangian multiplier update is given by

\[
\lambda_m = \lambda_{m-1} + \eta_m (\pi^* - \pi_m).
\]
where $\eta_{m-1} > 0$ is the stepsize that determines the power of regularization. Note that the regularized policy iteration is defined state-wise, i.e., for each $s \in S$. The minimization is taken over the probability simplex $\Delta_A := \{\pi(\cdot|s) : 0 \leq \pi(a|s) \leq 1, \sum_{a \in A} \pi(a|s) = 1\}$.

Let $\Lambda_M$ denote a suitably bounded domain that includes the dual optimal solution $\lambda^*$. We will provide an explicit construction of $\Lambda_M$ in Section 4. To update the Lagrangian multiplier, we use the projected subgradient ascent:

\[
\lambda_m = \text{Proj}_{\Lambda_M} \left\{ \lambda_{m-1} + \eta_{m-1} \cdot \partial_{\lambda} L(\pi_{m-1}, \lambda_{m-1}) \right\},
\]

where $\text{Proj}_{\Lambda_M} \{\cdot\}$ denotes the projection (in $L^2$-norm) on $\Lambda_M$. We need such a projection to ensure the boundedness of “subgradient” in order to establish convergence.

By the definition of KL-divergence, the regularized policy iteration can be re-written as

\[
\pi_m(\cdot|s) = Z_{m-1}^{-1} \cdot \pi_{m-1}(\cdot|s) \cdot \exp \left\{-\eta_{m-1} \cdot Q^{\pi_{m-1}, \lambda_{m-1}}(s, \cdot)\right\},
\]

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

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

Both (15) and (16) can be evaluated/approximated using simulation. More advanced approximation techniques for policy evaluation like TD-learning can also be applied here.

Suppose that our algorithm runs $T-1$ iterations and generates a sequence $\{(\pi_m, \lambda_m)\}_{0 \leq m \leq T-1}$. Then, the algorithm outputs a mixing policy and Lagrangian multiplier by taking a weighted average of the outputs:

\[
\bar{\pi}_T = \sum_{m=0}^{T-1} \tilde{\eta}_m \pi_m, \quad \bar{\lambda}_T = \sum_{m=0}^{T-1} \tilde{\eta}_m \lambda_m, \text{ where } \tilde{\eta}_m = \eta_m / \sum_{m=0}^{T-1} \eta_m.
\]

The averaging operation is required for convergence, since the objective $L(\pi, \lambda)$ is bilinear and does not possess sufficient convexity. In particular, 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 $(\pi_0, \ldots, \pi_{T-1})$ with initial randomization distribution $(\eta_0, \cdot \cdot \cdot, \eta_{T-1})$. Note that this essentially takes the average of the occupation measures of $\pi_m$’s. From $\bar{\pi}_T$, we can apply (10) to define a non-mixing stationary policy that has the same occupation measure.

Above all, our primal-dual algorithm is summarized in Algorithm 1.
Algorithm 1 Primal-Dual Algorithm to CMDP

Input: pre-specified projection domain $\Lambda_M$, stepsizes $\{\eta_m\}_{m \geq 0}$, initial policy $\pi_0$ and Lagrangian multiplier $\lambda_0$

for $m = 1, \ldots, T - 1$ do

update Lagrangian multipliers and policy as

$$\begin{align*}
\lambda_m &= \text{Proj}_{\Lambda_M} \left\{ \lambda_{m-1} + \eta_{m-1} \cdot \partial_{\lambda} L(\pi_{m-1}, \lambda_{m-1}) \right\}, \\
\pi_m(\cdot|s) &\propto \pi_{m-1}(\cdot|s) \cdot \exp \left\{ - \eta_{m-1} \cdot Q^{\pi_{m-1}, \lambda_{m-1}}(s, \cdot) \right\}.
\end{align*}$$

end for

Output: mixing policy $\bar{\pi}_T = \sum_{m=0}^{T-1} \bar{\eta}_m \pi_m$, where $\bar{\eta}_m = \eta_m / \sum_{m=0}^{T-1} \eta_m$.

4. Convergence Analysis

In this section, we conduct detailed performance analysis of Algorithm 1. In particular, we study the performance of policy $\bar{\pi}_T$ by analyzing the values of the objective $C(\bar{\pi}_T)$ and the constraints $D(\bar{\pi}_T)$. We show that the objective value $C(\bar{\pi}_T)$ converges to the optimal $C^* := C(\pi^*) = L(\pi^*, \lambda^*)$ at a rate of $O(\log(T)/\sqrt{T})$. In addition, even though $\bar{\pi}_T$ may be infeasible, we show that the violation of constraints, which is measured by

$$\|D(\bar{\pi}_T) - q\|^+ := \left( \sum_{k=1}^{K} \left( [D_k(\bar{\pi}_T) - q_k]^+ \right)^2 \right)^{1/2},$$

converges to zero at a rate of $O(\log(T)/\sqrt{T})$. The analysis builds on a combination of subgradient method for saddle point problem and mirror descent for regularized policy iteration.

Recall that our algorithmic development builds on the strong duality of CMDP. For CMDPs with finite state and action spaces, the 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 the 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)). Specifically, we impose the following assumption.

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

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

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

Assumption 2. [Slater’s Condition] There exists some policy $\tilde{\pi}$ such that

$$D_k(\tilde{\pi}) < q_k, \quad \forall 1 \leq k \leq K.$$
Slater’s condition ensures the existences of finite and bounded optimal Lagrangian multipliers
\[ \lambda^* = \arg \max_{\lambda \geq 0} \left\{ \inf_{\pi \in \mathcal{M}(\Pi_S)} L(\pi, \lambda) \right\}. \]

This condition is commonly assumed in the constrained optimization literature. For many practical problems, the Slater’s condition holds trivially.

Our last assumption is about the boundedness of the “subgradient”, which regularizes the movement of policies and Lagrangian multipliers at each iteration. Recall that in Algorithm 1, after applying the subgradient ascent for Lagrangian multipliers, we project \( \lambda \) onto a bounded domain \( \Lambda_M \), which takes the form
\[ \Lambda_M = \{ \lambda \in \mathbb{R}_+^K : \| \lambda \| \leq M + r \} , \tag{19} \]
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 \( \pi \in \mathcal{M}(\Pi_S) \),
\[ \| \partial_\lambda L(\pi, \lambda) \| \leq G, \quad \sup_{s \in S} \sup_{a \in A} |Q^{\pi, \lambda}(s, a)| \leq G. \]

Since \( Q^{\pi, \lambda}(s, a) \) is linear in \( \lambda \), it is necessary to restrict \( \lambda \) to a bounded domain \( \Lambda_M \) for Assumption 3 to hold. That is why we need the projection step in updating \( \lambda \). 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 the Slater’s condition (Assumption 2) not only guarantees the existence and boundedness of \( \lambda^* \), but also provides an explicit upper bound of \( \| \lambda^* \| \). In particular, let \( \tilde{\pi} \) be a Slater point (a policy that satisfies the Slater’s condition), then we have
\[ \| \lambda^* \| \leq -\frac{C(\tilde{\pi}) - \tilde{c}}{\max_{1 \leq k \leq K} \{ D_k(\tilde{\pi}) - q_k \}} , \tag{20} \]
where \( \tilde{c} \leq C(\pi^*) \) is an arbitrary lower bound for the dual problem. In many applications, it is possible to obtain a better upper bound of \( \| \lambda^* \| \) than (20) by exploiting the structure of the specific problem.

Next, to establish the convergence, we need to construct an appropriate potential function, which is also known as Bregman divergence in the optimization literature. The potential function ensures that the regularized policy iteration is equivalent to minimize the sum of a linear approximation of the objective function and the potential function. We next introduce this potential function, which is essentially a weighted KL-divergence.
Consider the state occupation measure $\nu^\pi_s$ induced by a policy $\pi \in \Pi_S$, i.e.,

$$
\nu^\pi_s(s) := (1 - \gamma) \cdot \mathbb{E}_{s_0 \sim \mu_0} \left[ \sum_{t=0}^{\infty} \gamma^t P^\pi(s_t = s|s_0) \right].
$$

The KL-divergence between two stationary policies $\pi_1$ and $\pi_2$ weighted by $\nu^\pi_s$ is defined as

$$
\Phi^\pi(\pi_1||\pi_2) = \mathbb{E}_{s \sim \nu^\pi_s} \left[ \text{KL}(\pi_1(\cdot|s)||\pi_2(\cdot|s)) \right].
$$

When $\pi_1$ and $\pi_2$ are mixing policies, we first transform them to the equivalent stationary policies via (10), 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 (13), which is defined state-wise, with a single objective and serves as the Bregman divergence in mirror descent analysis. Unlike the traditional analysis of mirror descent where the potential function is fixed (Nemirovski 2012), in the analysis of regularized policy iteration, we need to construct a policy-dependent potential function and cannot fix the weight of KL-divergence. However, since policy updates are defined state-wise, for an arbitrary weight, the regularized policy iteration always takes the form of minimizing a linear approximation of the objective function regularized by a certain potential function. Thus, the analysis of mirror descent can be applied here with some modifications.

We are now ready to introduce the convergence results of our primal-dual algorithm.

**Theorem 1. (Convergence of Main Algorithm) Under Assumptions 1-3, if the step size $\eta_m = \Theta(1/\sqrt{m})$, then there exist positive constants $\kappa_1$ and $\kappa_2$, such that**

$$
||[D(\pi_T) - q]^+]|| \leq \left( G^2 (1 + \frac{5}{8} \kappa_2 \log(T)) + \Phi^\pi(\pi^*||\pi_0) \right) \frac{1}{2r(1 - \gamma)\kappa_1 \sqrt{T}},
$$

and

$$
C(\pi_T) - L(\pi^*, \lambda^*) \leq \left( \frac{5G^2}{8} \cdot \kappa_2 \cdot \log(T) + \Phi^\pi(\pi^*||\pi_0) + \frac{\|\lambda_0\|^2}{2} \right) \frac{1}{(1 - \gamma)\kappa_1 \sqrt{T}},
$$

$$
C(\pi_T) - L(\pi^*, \lambda^*) \geq -\|\lambda*\| \left( G^2 (1 + \frac{5}{8} \kappa_2 \log(T)) + \Phi^\pi(\pi^*||\pi_0) \right) \frac{1}{2r(1 - \gamma)\kappa_1 \sqrt{T}}.
$$

If the step size is constant $\eta_m = \eta$, then

$$
||[D(\pi_T) - q]^+]|| \leq \left( G^2 + (1 - \gamma)^{-1} \cdot \Phi^\pi(\pi^*||\pi_0) \right) \frac{1}{2rT\eta} + \left( \frac{1}{2} + \frac{1}{8(1 - \gamma)} \right) \frac{G^2\eta}{2r},
$$

and

$$
C(\pi_T) - L(\pi^*, \lambda^*) \leq \left( (1 - \gamma)^{-1} \cdot \Phi^\pi(\pi^*||\pi_0) + \frac{\|\lambda_0\|^2}{2} \right) \frac{1}{T\eta} + \frac{5G^2\eta}{8(1 - \gamma)},
$$

$$
C(\pi_T) - L(\pi^*, \lambda^*) \geq -\|\lambda^*\| \left( G^2 + (1 - \gamma)^{-1} \cdot \Phi^\pi(\pi^*||\pi_0) \right) \frac{1}{2rT\eta} - \|\lambda^*\| \left( \frac{1}{2} + \frac{1}{8(1 - \gamma)} \right) \frac{G^2\eta}{2r},
$$

where $r$ is the slackness constant in (19).
Theorem 1 indicates that with decreasing step size, $\eta_m = \Theta(1/\sqrt{m})$, our primal-dual algorithm achieves $O(\log(T)/\sqrt{T})$ convergence. In particular,

$$\| [D(\tilde{\pi}_T) - q]^+ \| = O(\log(T)/\sqrt{T})$$

and

$$|C(\tilde{\pi}_T) - L(\pi^*, \lambda^*)| = O(\log(T)/\sqrt{T}).$$

For constant step size, $\eta_m = \eta$, our primal-dual algorithm converges to a neighborhood of the optimal at rate $O(1/T)$. In particular,

$$\| [D(\tilde{\pi}_T) - q]^+ \| = O(1/(\eta T) + \eta)$$

and

$$|C(\tilde{\pi}_T) - L(\pi^*, \lambda^*)| = O(1/(\eta T) + \eta).$$

These convergence rates match those in Le et al. (2019), which requires solving the modified unconstrained MDP to the optimal at each iteration. We also note that it is unlikely to improve the convergence rate beyond $\Theta(1/\sqrt{T})$. This is because the dual problem 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). The proof of Theorem 1 is deferred to the appendix.

We comment that in the bounds in Theorem 1 although the slackness constant $r$ appears in denominators only, the constant $G$, which is an upper bound of the subgradients, grows linearly in $r$. In particular, by Assumption $\bar{G}$, $G$ is determined by the shape of $\Lambda_M$. Hence, $r$ cannot be set too large.

5. Weakly Coupled MDP and Weakly Coupled CMDP

One fundamental challenge in solving MDPs and CMDPs is the curse of dimensionality. However, there is an important class of problems that has certain decomposable structures. These problems, which are often referred to as weakly coupled MDPs/CMDPs, contain multiple subproblems which are almost independent of each other except for some linking constraints on the action space (Singh and Cohn 1998). More precisely, for a weakly coupled MDP consisting of $I$ sub-problems $\{(S^i, A^i, P^i, c^i(\cdot, \cdot), \gamma, \mu_0^i)\}_{i \in [I]}$, we have the following structural properties: P1. Its state and action spaces can be expressed in the form of Cartesian products, i.e.,

$$s = (s^1, \ldots, s^I), \quad S = S^1 \times S^2 \times \ldots \times S^I,$$

$$a = (a^1, \ldots, a^I), \quad A = A^1 \times A^2 \times \ldots \times A^I.$$

P2. For each state $s_i$ and action $a_i$, the instantaneous cost admits an additively separable form

$$c(s_i, a_i) = \sum_{i=1}^I c^i(s^i_i, a^i_i).$$
P3. The joint initial distribution satisfies

\[ \mu_0(s) = \mu_0^1(s^1) \cdot \mu_0^2(s^2) \cdot \ldots \cdot \mu_0^I(s^I) \] and the one-step transition dynamics of the sub-MDPs are independent of each other, i.e.,

\[ P(s_{t+1}|s_t, a_t) = \prod_{i=1}^I P_i(s^i_{t+1}|s^i_t, a^i_t). \]

For the linking constraints, let \( b^i(\cdot, \cdot): \mathcal{S}^i \times \mathcal{A}^i \to \mathbb{R}^K \) be a \( K \)-dimensional real function, which can be interpreted as the resource consumed by the \( i \)-th sub-problem, \( i \in [I] \). Then, at each state \( s \), the feasible actions need to satisfy

\[ b(s, a) = \sum_{i=1}^I b^i(s^i, a^i) \leq q. \] (22)

where \( q \in \mathbb{R}^K \). Note that the linking constraint (22) is a hard constraint and needs to be satisfied path-by-path almost surely. For a weakly coupled CMDP, it satisfies the same structural properties, P1-P3, as the weakly coupled MDP. The only difference is that the linking constraint now takes the form

\[ (1 - \gamma) \cdot \mathbb{E}_{s_0 \sim \mu_0} \left[ \sum_{t=0}^{\infty} \gamma^t \cdot \sum_{i=1}^I b^i(s^i_t, a^i_t) | s_0 \right] \leq q. \] (23)

The weakly coupled MDP and the weakly coupled CMDP are closely related to each other. Let

\[ \bar{A}(s) = \left\{ a = (a^1, \ldots, a^I) \in \mathcal{A} : \sum_{i=1}^I b^i(s^i, a^i) \leq q \right\} \]

be the (joint) action space of a weakly coupled MDP. Then, the Bellman equation is

\[ V_{\mu_0}^*(s) = \min_{a \in \bar{A}(s)} \left\{ \sum_{i=1}^I c^i(s^i, a^i) + \gamma \cdot \sum_{s' \in \mathcal{S}} V_{\mu_0}^*(s') \cdot P(s'|s, a) \right\}. \]

When the number of sub-MDPs \( I \) is large, even if the scale of each subproblem is small, the size of joint state space \( \mathcal{S} \) can be prohibitively large. Hence, solving the MDP directly can be intractable. Two decomposition schemes have been proposed to alleviate the curse of dimensionality: LP-based ADP relaxation and Lagrangian relaxation (Adelman and Mersereau 2008). Both of them lead to \( I \) independent sub-LPs, which reduces the complexity significantly. The LP-based ADP relaxation approximates the value function with additively separable functions, i.e., \( V_{\mu_0}^*(s) \approx \sum_{i=1}^I V_{\mu_0}^i(s^i) \). The Lagrangian relaxation dualizes the constraints (22) based on the LP representation of the Bellman equation. The latter relaxation translates the weakly coupled MDP to a weakly coupled CMDP. It has been established that the optimal cost of the relaxed CMDP provides a lower bound for the optimal cost of the original MDP (Adelman and Mersereau 2008).

Many Operations Management problems can be formulated as weakly coupled MDPs/CMDPs. Examples include inventory planning problems with multiple types of inventories and budget constraints, and scheduling of parallel-server queues with multiple classes of customers. We provide
more details about these problems in Sections 6 and 7, where we apply our primal-dual algorithms to solve them.

When applying the primal-dual algorithm to solve weakly coupled CMDPs, it can be easily adapted to enjoy the decomposability. We call a policy $\pi$ decomposable if it takes the product form:

$$\pi(a|s) = \prod_{i=1}^{I} \pi_i(a^i|s^i).$$

Since our algorithm converges with any initial policy, we shall start with a decomposable policy. Let $\{s_t\}_{t \geq 0} = \{(s^1_t, \ldots, s^I_t)\}_{t \geq 0}$ and $\{a_t\}_{t \geq 0} = \{(a^1_t, \ldots, a^I_t)\}_{t \geq 0}$ be the trajectory of the CMDP under policy $\pi = (\pi^1, \ldots, \pi^I)$. To simplify the notations, for each $i \in [I]$, we define

$$C^i(\pi^i) = (1 - \gamma) \cdot \mathbb{E}_{s^i_0 \sim \mu^i_0} \left[ \sum_{t=0}^{\infty} \gamma^t \cdot c^i(s^i_t, a^i_t) \big| s^i_0 \right],$$

$$B^i(\pi^i) = (1 - \gamma) \cdot \mathbb{E}_{s^i_0 \sim \mu^i_0} \left[ \sum_{t=0}^{\infty} \gamma^t \cdot b^i(s^i_t, a^i_t) \big| s^i_0 \right].$$

Then, the CMDP can be written as

$$\min_{(\pi^1, \ldots, \pi^I)} \sum_{i=1}^{I} C^i(\pi^i), \quad \text{s.t.} \quad \sum_{i=1}^{I} B^i(\pi^i) \leq q. \quad (24)$$

When applying the primal-dual algorithm, if we start with a decomposable policy, then the policies obtained in all subsequent iterations are decomposable. To see this, we note that the Lagrangian function,

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

can be decomposed into $I$ independent subproblems. If $\pi^*_m$ is decomposable,

$$Q^{\pi^*_m, \lambda}(s, \cdot) = \sum_{i=1}^{I} Q^{\pi^*_m, \lambda}(s^i, \cdot),$$

where $Q^{\pi^*_m, \lambda}(\cdot, \cdot)$ is the $Q$-function of the $i$-th modified sub-MDP with instantaneous cost $c^i(\cdot, \cdot) + \lambda^\top b^i(\cdot, \cdot)$. Here, we ignore the constant $\lambda^\top q$, since subtracting a common constant in the $Q$-function does not change the updates of regularized policy iteration. This indicates that the regularized policy iteration, including policy evaluation and improvement, can be implemented separately in parallel via

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

Moreover, as the subgradient of Lagrangian multiplier takes form $\partial_\lambda L(\pi_m, \lambda) = \sum_{i=1}^{I} B^i(\pi^i_m) - q$, it can be evaluated for the sub-MDPs in parallel as well. Above all, in this case, the primal-dual algorithm improves the computational complexity from depending exponentially on $I$ to linearly on $I$. 
6. Application to an Inventory Planning Problem

In this section, we apply the primal-dual algorithm to solve a multi-product multi-period newsvendor problem with budget constraints.

Consider the inventory planning problem with $I$ distinct products. At the beginning of each period, we need to decide the quantities to order based on the current inventory levels. The orders are assumed to be fulfilled without delay. After the inventory is replenished, a random demand is realized. We assume the demands for each product are independent. Let $F_i$ denote the cumulative distribution function of the demand for product $i$ in each period. In particular, for each period, the demand for product $i$ is an independent draw from the distribution $F_i$. For each product $i \in [I]$, we denote its inventory level at the beginning of period $t$ by $s_{i,t}$, the quantity we ordered by $a_{i,t}$, and the demand in period $t$ by $w_{i,t}$. For product $i$ in period $t$, if the demand does not exceed the current inventory level, i.e., $w_{i,t} \leq s_{i,t} + a_{i,t}$, all the demand is fulfilled and the remaining inventory can be carried to the next period. Otherwise, only $s_{i,t} + a_{i,t}$ units are fulfilled in the current period. The remaining $(w_{i,t} - s_{i,t} - a_{i,t})$ units are carried to the next period as backlog. We allow $s_{i,t}$’s to be negative to represent backlogs. 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, product $i$ in inventory consumes $v_i$ resource per unit per period. For a fixed $q > 0$, we impose the following budget constraint

$$(1 - \gamma) \cdot \mathbb{E} \left[ \sum_{t=0}^{\infty} \sum_{i=1}^{I} \gamma^t \cdot (s_{i,t} + a_{i,t})^+ \cdot v_i \right] \leq q. \quad (25)$$

The resource can be interpreted as, for example, the volume of each product. In this case, the above constraint put restrictions on the warehouse space.

The inventory planning problem can be formulated as a weakly coupled CMDP with state $s = (s^1, \ldots, s^I)$, action $a = (a^1, \ldots, a^I)$, and transition dynamics

$$s_{i,t+1} = s_{i,t} + a_{i,t} - w_{i,t}, \quad w_{i,t} \sim F_i(w), \quad \forall i \in [I].$$

As the demands are independent, $P(s_{i+1}|s_i, a_{i+1}) = \prod_{i=1}^{I} P(s_{i+1}|s_i, a_{i+1})$. The instantaneous cost function and auxiliary cost function are

$$c(s_t, a_t) = \sum_{i=1}^{I} h_i \cdot (s_{i,t} + a_{i,t} - w_{i,t})^+ + b_i \cdot (w_{i,t} - s_{i,t} - a_{i,t})^+,$n

$$b(s_t, a_t) = \sum_{i=1}^{I} (s_{i,t} + a_{i,t})^+ \cdot v_i.$$

To verify the correctness of our convergence analysis, we consider a small-scale instance of the problem with appropriate truncations. Such a truncation makes the state and action spaces finite.
In this case, the optimal cost can be solved numerically (using the LP formulation). In particular, consider \( I = 2 \), and demands for the two products are both uniformly distributed on set \( \{1, 2, \ldots, 10\} \). We impose an upper bound 10 and a lower bound \(-10\) for the state space. In particular, when backlogs drop 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 \( b_1 = 2, b_2 = 3 \), resource consumptions \( v_1 = 1.5, v_2 = 1 \), threshold \( q = 10 \), and discount rate \( \gamma = 0.75 \).

When implementing the primal-dual algorithm, we use the standard Monte Carlo method to estimate the \( Q \)-function for a given policy. Since the system scale is small, we can enumerate all the state-action pairs in policy evaluation, i.e., no approximation of the value function is needed. The estimation of \( Q \)-function is based on an average of 400 independent replications of the inventory process over 40 periods of time. We implement two versions of the algorithm, one with constant step sizes \( \eta_m = 0.2 \), the other with decreasing step size \( \eta_m = 0.2/\sqrt{m + 1} \). In each experiment, we run 500 iterations in total and calculate the objective values for each iteration. The results of numerical experiments are summarized in Figures 1 and 2. Figure 1 shows the trajectories of objective values and the constraint violations for different iterations with constant step size. We observe that after 500 iterations, the averaged CMDP cost (without multiplying the \( (1 - \gamma) \) factor) converge to \( 49.26 \), which is close to the optimal value \( 46.47 \). In terms of feasibility, we calculate the violation of constraints, which is the expected value of the auxiliary cost minus the budget threshold. We observe that the averaged violation value converges to \( 0.1 \) and many policies in the last iterations do not violate the constraint at all. Figure 2 shows the relationship between \( \sum_{t=0}^{T-1} \tilde{\eta}_t C(\pi_t) \) and the reciprocal of the number of iterations (for constant step size) or the reciprocal of square root of the number of iterations (for decreasing step size). In both cases, we observe a straight line, which confirms the rates of convergence developed in Theorem 1.

7. Application to Queueing Scheduling

In this section, we apply our primal-dual algorithm to a queue scheduling problem, which is motivated by applications in service operations management. Service systems often feature multiple classes of customers with different service needs and multiple pools of servers with different skillsets. Efficiently matching customers with compatible servers is critical to the management of these systems. In this context, we consider a parallel-server system (PSS) with multiple classes of customers and multiple pools (types) of servers. Customers waiting in queue incur some holding costs and routing customers to different pools leads to different routing costs. The goal is to find a scheduling policy that minimizes the performance cost (holding cost plus routing cost). This class of problems is known as the skill-based routing problem and has been widely studied in the literature. We refer to [Chen et al., 2020] for a comprehensive survey of related works.
Figure 1  Trajectories of costs and constraints with constant step sizes

Figure 2  Convergence rate with constant and decreasing step sizes
In what follows, we first introduce the queueing model and some heuristic policies adapted from policies developed in the literature. We then present the implementation details of our primal-dual algorithm in this setting. Due to the large state and action spaces, we combine our primal-dual algorithm with several approximation techniques. Lastly, we compare the performance of our policy with the benchmark policies numerically.

7.1. Model and Benchmarks

The multi-class multi-pool queuing network has $I$ classes of customers and $J$ pools of servers. We consider a discrete time model. In each period, the number of arrivals of class $i$ customers follows a Poisson distribution with rate $\theta_i$. There are $N_j$ homogeneous servers in pool $j$, $j \in [J]$. We assume that each customer can only be served by one server and each server can only serve one customer at a time. If a class $i$ customer is served by a server from pool $j$, its service time follows a geometric distribution with success probability $\mu_{ij}$. When there is no compatibility between customer class $i$ and server type $j$, $\mu_{ij} = 0$. Figure 3 provides a pictorial illustration of such a system.

![Figure 3 Multi-class multi-pool queueing system](image)

We consider non-preemptive scheduling policies. Let $A_i(t)$ denote the number of new class $i$ arrivals in time period $t$, i.e., $A_i(t)$ follows a Poisson distribution with rate $\theta_i$. Let $Z_{ij}(t)$ denote the number of class $i$ customers in service in pool $j$ at the beginning time period $t$. We also denote $U_{ij}(t)$ as the number of class $i$ customers assigned to pool $j$ for time period $t$. Note that $U_{ij}(t)$'s are determined by our scheduling policy. Then the number of class $i$ departures from pool $j$ at the end of time period $t$, $R_{ij}(t)$, follows a Binomial distribution with parameter $Z_{ij}(t) + U_{ij}(t)$ and $\mu_{ij}$. Let $X_i(t)$ denote the number of class $i$ customers waiting in queue at the beginning of period $t$. Then we have the following system dynamics,

$$
X_i(t+1) = X_i(t) + A_i(t) - \sum_{j=1}^{J} U_{ij}(t), \quad \forall i \in [I]
$$

$$
Z_{ij}(t+1) = Z_{ij}(t) + U_{ij}(t) - R_{ij}(t), \quad \forall i \in [I], j \in [J].
$$
The state of the system is \( s(t) = (X_i(t), Z_{ij}(t) : i \in [I], j \in [J]) \in \mathbb{N}^{I \times (J+1)}. \) The action is \( a(t) = (U_{ij}(t) : i \in [I], j \in [J]) \in \mathbb{N}^{I \times J}. \)

The routing policy needs to satisfy the following constraints

\[
U_{ij}(t) \in \mathbb{N}, \sum_{j=1}^{J} U_{ij}(t) \leq X_i(t), \quad \forall i \in [I], j \in [J], \quad \forall t \geq 0. \tag{27}
\]
i.e., we can not schedule more customers than there are waiting, and

\[
\sum_{i=1}^{I} Z_{ij}(t) + U_{ij}(t) \leq N_j, \quad \forall j \in [J], \quad \forall t \geq 0, \tag{28}
\]
i.e., the number of customers in service can not exceed the capacity. Note that constraints (27)-(28) are hard constraints, i.e., they need to be satisfied path-by-path.

Each class \( i \) customer waiting in queue incurs a holding cost of \( h_i \) per period. There is also a one-shot routing cost of \( r_{ij} \) for scheduling a class \( i \) customer to a pool \( j \) server. The overall cost for period \( t \) is given by

\[
c(s(t), a(t)) = \sum_{i=1}^{I} h_i X_i(t) + \sum_{i=1}^{I} \sum_{j=1}^{J} r_{ij} U_{ij}(t).
\]

Our goal is to minimize the cumulative discounted costs:

\[
(1 - \gamma) \cdot \mathbb{E}^s \left[ \sum_{t=0}^{\infty} \gamma^t \cdot c(s(t), a(t)) \right].
\]

The problem we consider here is a weakly coupled MDP with \( I \) sub-problems, where each sub-problem is an inverted-V model (i.e., a single customer class and multiple server pools). In particular, for the \( i \)-th sub-problem, define state and action as \( s^i(t) = (X_i(t), Z_{ii}(t), \ldots, Z_{ij}(t)) \) and \( a^i(t) = (U_{ii}(t), \ldots, U_{ij}(t)) \). The transition dynamics of the \( i \)-th sub-system follows

\[
X_i(t+1) = X_i(t) + A_i(t) - \sum_{j=1}^{J} U_{ij}(t), \quad Z_{ij}(t+1) = Z_{ij}(t) + U_{ij}(t) - R_{ij}(t), \quad \forall j \in [J].
\]

Given \( s^i(t) \), the corresponding action space is defined as

\[
\mathcal{A}^i(s^i(t)) = \left\{ \{U_{ij}(t)\}_{j \in [J]} : U_{ij}(t) \in \mathbb{N}, \sum_{j=1}^{J} U_{ij}(t) \leq X_i(t), \quad Z_{ij}(t) + U_{ij}(t) \leq N_j, \quad \forall j \in [J] \right\}.
\]

We also define the auxiliary cost function

\[
b^i(s^i(t), a^i(t)) = (Z_{i1}(t) + U_{i1}(t), \ldots, Z_{ij}(t) + U_{ij}(t))^\top \in \mathbb{N}^J.
\]

Then the capacity constraints (28) can be expressed as

\[
\sum_{i=1}^{I} b^i(s^i(t), a^i(t)) \leq (N_1, \ldots, N_J)^\top,
\]
which takes the same form as the linking constraint in (22).

There are three important features of the problem that we attempt to address in this section: 1) non-preemptive routing; 2) both class-and-pool dependent service rate; 3) routing cost (overflow cost). The first two features require us to keep track of a very high dimensional state space, i.e., $I(J + 1)$. The third feature has not been extensively studied in the literature.

We next introduce two heuristic policies adapted from policies developed in the literature. For PSS with multiple classes of customers and multiple pools of servers, a myopic policy called the $cµ$-rule (or generalization of it), has been shown to be asymptotically optimal in some systems, where the goal is to minimize the holding cost (Mandelbaum and Stolyar 2004). The idea is to minimize the instantaneous cost-reduction rate at each decision epoch. Another policy is called the max-pressure policy, which is known to be throughput optimal and asymptotically cost optimal for some forms of convex holding cost (Stolyar et al. 2004, Dai et al. 2008). We next consider modified versions of the above routing policies, which take the routing costs into account (Chen et al. 2020).

At each decision epoch $t$, we choose $U_{ij}(t)$’s that solve the following optimization problem:

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

s.t. $\sum_{j=1}^{J} U_{ij}(t) \leq X_i(t)$, $\forall i \in [I], j \in [J]$, $\forall t \geq 0$,

$$\sum_{i=1}^{I} \sum_{j=1}^{J} Z_{ij}(t) + U_{ij}(t) \leq N_j$, $\forall j \in [J]$, $\forall t \geq 0$,

$$U_{ij}(t) \in \mathbb{N}$ $\forall i \in [I], j \in [J]$, $\forall t \geq 0$,

where $\omega_{ij}(t)$’s are some modified instantaneous costs we introduce next. We consider two different forms of $w_{ij}(t)$’s. The first one sets $\omega_{ij}(t) = h_i - r_{ij}$, which is adapted from the $cµ$-rule. We refer to this policy as the modified $cµ$-rule. The second one sets $w_{ij}(t) = h_i X_i(t) - r_{ij}$, which is adapted from the max-pressure policy. We refer to this policy as the modified max-pressure policy.

7.2. Solution method

We consider the CMDP relaxation of the weakly coupled MDP:

$$\min_{\pi} (1 - \gamma) \cdot \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \cdot c(s(t), a(t))]$$

s.t. $(1 - \gamma) \cdot \mathbb{E}\left[\sum_{i=1}^{I} \sum_{t=0}^{\infty} \gamma^t \cdot d^i(s^i(t), a^i(t))\right] \leq (N_1, \ldots, N_J)^T$, 
and apply the primal-dual algorithm to solve it. The decoupling allows us to translate the original problem to \( I \) sub-problems. In particular, in each iteration, we use regularized policy iteration to update the scheduling policy for a single-class multi-pool system with modified instantaneous cost:

\[
c'_i(s^i(t), a^i(t)) = h_i X_i(t) + \sum_{j=1}^{J} r_{ij} U_{ij}(t) + \sum_{j=1}^{J} \lambda_j (Z_{ij}(t) + U_{ij}(t))
\]

for the \( i \)-th sub-problem.

Even with the decomposition, the state and policy spaces are still too large in this case. We next introduce some further approximations to reduce the dimension of the problem. We shall omit the index \( i \) in subsequent discussions as the development focuses on each sub-problem.

**Policy space reduction:** For each sub-problem, the policy space is still prohibitive. To see this, consider a system with 3 pools and 30 servers in each pool. When the queue length is 90 and all pools are empty, there are roughly \( 30^3 \) feasible actions. To overcome the challenge, we reduce the action space to only include priority rules. State-dependent extreme policies have been shown to be asymptotically optimal in the scheduling of PSS due to the linear system dynamics and linear holding costs ([Harrison and Zeevi 2004](#)). Denote \(-1\) as the waiting option. The priority rule is denoted by a priority list that ends with \(-1\). For example, priority \((1, 2, -1)\) means pool 1 is preferred to pool 2, which is preferred to waiting. When following priority \((1, 2, -1)\), we first assign as many customers to pool 1 as possible. If there are still customers waiting after pool 1 assignment, we start assigning them to pool 2. After that, if there are still customers waiting, we keep them in the queue. We denote this reduced policy space as \( \tilde{A} \).

**Value function approximation:** In our policy iteration step, given a policy \( \pi \), we need to estimate the function \( Q^{\pi, \lambda}(s, a) \) for all \( s \in S, a \in \tilde{A} \), where the state \( s = (x, z_1, \ldots, z_J) \). Due to the large state space, we can not enumerate all the states to evaluate the value function. Instead, we use value function approximation with quadratic basis. The idea is to find \( \theta^{\pi, a} \in \mathbb{R}^{(J+1)^2+1} \) such that

\[
Q^{\pi, \lambda}(s, a) \approx \langle \phi(s), \theta^{\pi, a} \rangle.
\]

where \( \phi(s) \) is the quadratic basis. To obtain \( \theta^{\pi, a} \) at each iteration, we first randomly sample \( M \) states \( \{s_i\}_{i \in [M]} \) and use Monte Carlo simulation to estimate \( Q^{\pi, \lambda}(s_i, a) \). Then, set

\[
\theta^{\pi, a} = \arg\min_{\theta} \left\{ \frac{1}{M} \sum_{i=1}^{M} \left( Q^{\pi, \lambda}(s_i, a) - \langle \phi(s_i), \theta \rangle \right)^2 \right\}.
\]

### 7.3. Experiment Results

For the numerical experiments, we consider a similar setting as that in [Dai and Shi 2019](#), which is motivated by hospital inpatient-flow management. In particular, we consider a network with
We run two sets of experiments, corresponding to large routing/overflow costs: has a primary server pool with the fastest service rate and zero routing cost, we always give the problems with different routing costs and discount rates. For customer class $i$, we define its nominal traffic intensity as $\rho_i$, the nominal traffic intensity of the three classes are $\rho_1 = 1, \rho_2 = 16/15, \text{and} \rho_3 = 5/6$. This indicates that the first two classes are unstable if we do not do any “overflow”.

We initialize the system with $X_i(0) = 50$ and $Z_{i1}(0) = 20, Z_{i2}(0) = 30, Z_{i3}(0) = 40, \text{and} Z_{ij}(0) = 0 \text{for} i \neq j, i, j \in [3]$. We compare the performance of our policy with the two benchmark policies for problems with different routing costs and discount rates.

When constructing the policy space for our primal-dual algorithms, because each customer class has a primary server pool with the fastest service rate and zero routing cost, we always give the primary pool the highest priority. In particular, the action spaces for three classes are defined as

\[
\mathcal{A}_1 = \{(1, -1), (1, 2, -1), (1, 3, -1), (1, 2, 3, -1), (1, 3, 2, -1)\}, \\
\mathcal{A}_2 = \{(2, -1), (2, 1, -1), (2, 3, -1), (2, 1, 3, -1), (2, 3, 1, -1)\}, \text{and} \\
\mathcal{A}_3 = \{(3, -1), (3, 2, -1), (3, 1, -1), (3, 2, 1, -1), (3, 1, 2, -1)\}, \text{respectively.}
\]

In our primal-dual update, we use the constant stepsize 0.1. When using simulation to estimate the value function, we truncate at $T = 100, 150, 800$ for $\gamma = 0.9, 0.95, 0.99$ respectively. This ensures that $\gamma^T \approx 10^{-4}$, i.e., the truncation errors are almost negligible. When fitting the parameters for the quadratic value function approximation, we sample 1000 states and use simulation to estimate the $Q$-function at these states. For each value of $\gamma$, we start with the Lagrangian multipliers $\lambda_0 = (10, 10, 10)$ and run the primal-dual algorithm for 30 iterations, and take the policy obtained.

3 classes of customers and 3 pools of servers. Pool $i$ is considered the primary pool for class $i$ customers with $r_{ii} = 0, \forall i \in [I]$. The major difference between our model and the model considered in Dai and Shi (2019) is that we allow the service rates to vary for different server types, i.e., $\mu_{ij}$ depends on both $i$ and $j$. This captures the potential slowdown effect due to off-service placement (Song et al. 2020).

For the system parameters, we set the arrival rates $(\theta_1, \theta_2, \theta_3) = (12, 16, 20)$, the holding costs $(h_1, h_2, h_3) = (3, 2, 1)$, the pool sizes $(N_1, N_2, N_3) = (40, 50, 60)$, and the service rates $(\mu_{11}, \mu_{12}, \mu_{13}) = (0.3, 0.25, 0.2), (\mu_{21}, \mu_{22}, \mu_{23}) = (0.15, 0.3, 0.2), (\mu_{31}, \mu_{32}, \mu_{33}) = (0.25, 0.1, 0.4)$.

We run two sets of experiments, corresponding to large routing/overflow costs:

\[
(r_{11}, r_{12}, r_{13}) = (0, 2, 2), (r_{21}, r_{22}, r_{23}) = (3, 0, 3), (r_{31}, r_{32}, r_{33}) = (1, 1, 0),
\]
(29)

and small routing costs:

\[
(r_{11}, r_{12}, r_{13}) = (0, 0.2, 0.2), (r_{21}, r_{22}, r_{23}) = (0.3, 0, 0.3), (r_{31}, r_{32}, r_{33}) = (0.1, 0.1, 0).
\]
(30)

Note that for class $i$ customers, the primary server pool $i$ has the largest service rate and zero routing cost. For customer class $i$, we define its nominal traffic intensity as $\rho_i = \theta_i/(N_i \mu_{ii})$. Then the nominal traffic intensity of the three classes are $\rho_1 = 1, \rho_2 = 16/15, \text{and} \rho_3 = 5/6$. This indicates that the first two classes are unstable if we do not do any “overflow”.

We initialize the system with $X_i(0) = 50$ and $Z_{i1}(0) = 20, Z_{i2}(0) = 30, Z_{i3}(0) = 40, \text{and} Z_{ij}(0) = 0 \text{for} i \neq j, i, j \in [3]$. We compare the performance of our policy with the two benchmark policies for problems with different routing costs and discount rates.
in the last iteration. Note that this policy may not be feasible to the original weakly coupled MDP.

In order to obtain a feasible policy, we adopt the following modification. In each period, for each pool, when the number of scheduled customers exceeds the capacity, the primary customers are prioritized for admission. We then admit the “overflowed” customers uniformly at random until the capacity is reached. The customers who are not admitted to service will be sent back to their corresponding queues and wait for the next decision epoch. For example, suppose that there are 20 servers available in pool 1 but the policy schedules (15, 5, 5) customers from the three classes to this pool. The modified policy first admits the 15 customers from class 1 and then randomly picks 5 among the 10 customers of classes 2 and 3 to admit.

Given a policy, to evaluate its performance, we estimate the cumulative discounted costs from 500 independent replications of the system over \( T \) periods of time. The results are summarized in Tables 1 and 2.

**Table 1** Cumulative discounted costs under different policies with large routing costs under different discount factors. (Numbers in bracket are the standard errors from simulation estimation.)

| Policy                          | \( \gamma = 0.90 \) | \( \gamma = 0.95 \) | \( \gamma = 0.99 \) |
|---------------------------------|----------------------|----------------------|----------------------|
| modified \( c\mu \)-rule        | 270.49 (1.50)         | 286.11 (2.07)         | 467.13 (4.26)         |
| modified max-pressure rule      | 271.62 (1.25)         | 269.19 (1.74)         | 278.31 (2.31)         |
| primal-dual algorithm           | 252.77 (1.47)         | 243.87 (2.26)         | 218.95 (2.11)         |

**Table 2** Cumulative discounted costs under different policies with small routing costs under different discount factors. (Numbers in bracket are the standard errors from simulation estimation.)

| Policy                          | \( \gamma = 0.90 \) | \( \gamma = 0.95 \) | \( \gamma = 0.99 \) |
|---------------------------------|----------------------|----------------------|----------------------|
| modified \( c\mu \)-rule        | 232.22 (1.20)         | 230.54 (1.71)         | 266.81 (3.65)         |
| modified max-pressure rule      | 260.89 (1.21)         | 266.83 (1.77)         | 308.89 (2.06)         |
| primal-dual algorithm           | 253.53 (1.50)         | 251.20 (2.40)         | 210.34 (2.24)         |

We observe that the policies obtained via the primal-dual algorithm performs well. It outperforms the two benchmark policies in most cases. When the routing cost is large (Table 1), the cost under the modified \( c\mu \)-rule increases substantially as the discount rate \( \gamma \) increases. When taking a closer look at \( w_{ij}(t) \)'s, we note that in this case, \( w_{21}(t) = -2.7 \) and \( w_{23}(t) = -2.6 \). This implies that the modified \( c\mu \)-rule would never overflow class 2 customers. As a result, the system is unstable, i.e., the class 2 queue blow up as \( t \) increases. (The cumulative discounted cost is well-defined as the
discount rate decays exponentially in $t$ while the queue length grows linearly in $t$.) The modified max-pressure is able to achieve reasonably good performance in this case. When $\gamma$ is small, our algorithm achieves comparable (slightly better) performance as the max-pressure policy. When $\gamma$ is large, i.e., $\gamma = 0.99$, our policy is able to achieve a substantially lower cost than the max-pressure policy, i.e., a 21% cost reduction. This is because the max-pressure policy only starts overflowing when the queues are large enough. In this example where overflow is necessary to achieve system stability, we need more aggressive overflow. Our policy is able to “learn” this through the primal-dual training.

When the overflow cost is small (Table (2), the modified $c_\mu$-rule is able to achieve better performance than the modified max-pressure policy. Note that in this case, all $w_{ij}(t)$'s are nonnegative for both the modified $c_\mu$-rule and the modified max-pressure policy (when $X_i(t) > 0$). When $\gamma$ is small, our policy achieves comparable performance as the modified $c_\mu$-rule, when $\gamma$ is large, i.e., $\gamma = 0.99$, our policy can achieve a 21% cost reduction over the modified $c_\mu$-rule. This suggests that overflow needs to be exercised carefully.

We next discuss the structure of the policies obtained via primal-dual algorithm. We observe that our policies in general follow a threshold structure: overflow customers only when the queue length exceeds some threshold. However, the thresholds are highly dependent on the states of the system. Take the scheduling policy for class 1 and 2 customers with discount rate $\gamma = 0.9$ as an example. In Figure 4, we plot the values of the threshold of starting overflowing for different values of $Z_{11}$'s and $Z_{22}$'s. We observe that holding $Z_{12}$ and $Z_{13}$ fixed, as $Z_{11}$ increases, the threshold for overflow decreases. Similarly, holding $Z_{21}$ and $Z_{23}$ fixed, as $Z_{22}$ increases, the threshold for overflow also decreases.

![The threshold for class 1 and 2 queues when starting overflowing.](attachment:image.png)
8. Conclusion and Future Directions

In this work, we propose a sampling-based primal-dual algorithm to solve CMDPs. Our approach alternatively applies regularized policy iteration to improve the policy and subgradient ascent to maintain the constraints. The algorithm achieves $O(\log(T)/\sqrt{T})$ convergence rate and only requires one policy update at each primal-dual iteration. Our algorithm also enjoys the decomposability property for weakly coupled CMDPs. We demonstrate the applications of our algorithm to solve two important operations management problems with weakly coupled structures: multi-product inventory management and multi-class queue scheduling.

In Section 7, we also show the good empirical performance of our algorithm to solve an important class of weakly coupled MDPs. This opens two directions for future research. First, it is important to quantify the optimality gap between the weakly coupled MDP and its CMDP relaxation theoretically. The gap can be large in some problems as demonstrate in Adelman and Mersereau (2008). It would be interesting to establish easy-to-verify conditions about when the gap is small. Second, the policy obtained via the Lagrangian relaxation may not satisfy the hard constraints in the original MDP. One approach to overcome the issue is to use more stringent thresholds when defining constraints in the CMDP relaxation (Balseiro et al., 2019). The other approach is to modify the CMDP based policies to construct good MDP policies. For example, Brown and Smith (2020) study a dynamic assortment problem and propose an index heuristic from the relaxed problem, and show that the policy achieves asymptotic optimality. In Section 7, we apply a rather straightforward modification to the CMDP based policy in order to satisfy the hard constraints in the original MDP. In general, how to “translate” the policy derived based on the relaxed problem to the original MDP would be an interesting research direction.

References

Achiam J, Held D, Tamar A, Abbeel P (2017) Constrained policy optimization. arXiv preprint arXiv:1705.10528.

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

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

Balseiro SR, Brown DB, Chen C (2019) Dynamic pricing of relocating resources in large networks. ACM SIGMETRICS Performance Evaluation Review 47(1):29–30.

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

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.

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 manage-ment. 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, Nachum O, Duenez-Guzman E, Ghavamzadeh M (2018) A Lyapunov-based approach to safe rein-
forcement learning. Advances in neural information processing systems, 8092–8101.

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.

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

Geist M, Scherrer B, Pietquin O (2019) A theory of regularized Markov decision processes. arXiv preprint arXiv:1901.11275.

Haarnoja T, Tang H, Abbeel P, Levine S (2017) Reinforcement learning with deep energy-based policies. arXiv preprint arXiv:1702.08165.

Harrison JM, Zeevi A (2004) Dynamic scheduling of a multiclass queue in the halfin-whitt heavy traffic
regime. Operations Research 52(2):243–257.

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

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

Liu B, Cai Q, Yang Z, Wang Z (2019a) Neural proximal/trust region policy optimization attains globally
optimal policy. arXiv preprint arXiv:1906.10306.
Liu Y, Ding J, Liu X (2019b) Ipo: Interior-point policy optimization under constraints. arXiv preprint arXiv:1910.09615 .

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

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.

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 (2012) Tutorial: Mirror descent algorithms for large-scale deterministic and stochastic convex optimization. Conference on Learning Theory (COLT).

Schulman J, Levine S, Abbeel P, Jordan M, Moritz P (2015) Trust region policy optimization. International conference on machine learning, 1889–1897.

Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 .

Shani L, Efroni Y, Mannor S (2019) Adaptive trust region policy optimization: Global convergence and faster rates for regularized mdps. arXiv preprint arXiv:1909.02769 .

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.

Stolyar AL, et al. (2004) Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. The Annals of Applied Probability 14(1):1–53.

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

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

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).

Wang L, Cai Q, Yang Z, Wang Z (2019) Neural policy gradient methods: Global optimality and rates of convergence. arXiv preprint arXiv:1909.01150 .
Appendix. Proof of Main Results

The proof of Theorem 1 relies the following lemma, which upper and lower bounds the movement of the Lagrangian after a single iteration/update of the policy and the Lagrangian multipliers.

**Lemma 1.** Let \( \{(\pi_m, \lambda_m)\}_{m \geq 0} \) be the sequences of stationary policies and Lagrangian multipliers generated by Algorithm 2. Then for arbitrary \( \lambda \in \mathbb{R}_+^{K} \) and \( \pi \in \Pi_S \), we have the upper bound

\[
L(\pi_m, \lambda) - L(\pi_m, \lambda_m) \leq (2\eta_m)^{-1} \cdot (\|\lambda - \lambda_m\|^2 - \|\lambda - \lambda_{m+1}\|^2) + \eta_m/2 \cdot \|\partial_\lambda L(\pi_m, \lambda_m)\|^2,
\]

and the lower bound

\[
L(\pi, \lambda_m) - L(\pi_m, \lambda_m) \geq (1-\gamma)\eta_m^{-1} \left( \Phi^*(\pi_{m+1}) - \Phi^*(\pi_m) \right) - \frac{\eta_m}{\delta(1-\gamma)} \cdot (\sup_{s \in S} \sup_{a \in A} [Q^\lambda_m, \pi_m(s, a)])^2.
\]

Before we prove Lemma 1, we first present two auxiliary lemmas. The first lemma (Lemma [2]) is rather standard. A similar version of the result can be found in Proposition 3.2.2 in [Bertsekas and Scientific 2013]. For self-completeness, we still provide the proof here.

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

\[
x^* = \arg\min_{x \in C} \left\{ f(x) + \frac{1}{\eta} \Psi_\xi(x\|x_0) \right\}.
\]

Then we have

\[
f(x) - f(x^*) \geq \frac{1}{\eta} \left( \Psi_\xi(x^*\|x_0) + \Psi_\xi(x\|x^*) - \Psi_\xi(x\|x_0) \right), \quad \forall \ x \in \Omega.
\]

By symmetry, for a concave function \( g \) on \( \Omega \) and

\[
\hat{x}^* = \arg\max_{x \in C} \left\{ g(x) - \frac{1}{\eta} \Psi_\xi(x\|x_0) \right\}.
\]

Then

\[
g(x) - g(\hat{x}^*) \leq -\frac{1}{\eta} \left( \Psi_\xi(\hat{x}^*\|x_0) + \Psi_\xi(x\|\hat{x}^*) - \Psi_\xi(x\|x_0) \right), \quad \forall \ x \in \Omega.
\]

**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 \( C \), there exists a subgradient of the form

\[
p^* = q^* + \eta^{-1} \cdot \partial_x \Psi_\xi(x^*\|x_0) = q^* + \eta^{-1} \cdot (\nabla \xi(x^*) - \nabla \xi(x_0))
\]

such that

\[
\langle p^*, x - x^* \rangle \geq 0, \quad \forall x \in C.
\]

Here \( q^* \in \partial_x f(x^*) \) is some subgradient of \( f(x) \) at \( x^* \). As a result, by the property of subgradient, for all \( x \in C \), we have

\[
f(x) \geq f(x^*) + \langle q^*, x - x^* \rangle
\]

\[
\geq f(x^*) + \eta^{-1} \cdot \langle \nabla \xi(x_0) - \nabla \xi(x^*), x - x^* \rangle
\]

\[
= f(x^*) + \eta^{-1} \cdot \left( \Psi_\xi(x^*\|x_0) + \Psi_\xi(x\|x^*) - \Psi_\xi(x\|x_0) \right),
\]

\[
\forall x \in C.
\]
where the last equality follows from the definition of Bregman divergence, i.e.,
\[ \Psi_{\xi}(x \| y) = \xi(x) - \xi(y) - \langle \nabla \xi(y), x - y \rangle. \]

For the maximization problem, we only need to consider \(-g\) and apply above result. \(\Box\)

The next lemma is Lemma 6.1 in (Kakade and Langford 2002). Given two policies, it characterizes the difference of expected accumulated costs as the inner product of the advantage function of one policy and the occupation measure of another policy. Note that the value function \(V^\pi\) of an MDP under policy \(\pi\) are defined in (11) and (12).

**Lemma 3.** For arbitrary policies \(\pi, \pi' \in \Pi_S\),
\[ \mathbb{E}_{s \sim \mu_0} \left[ V^\pi(s) \right] - \mathbb{E}_{s \sim \mu_0} \left[ V^{\pi'}(s) \right] = \frac{1}{1 - \gamma} \mathbb{E}_{(s,a) \sim \nu^\pi} \left[ Q^\pi(s,a) - V^\pi(s) \right]. \]
where \(\nu^\pi(\cdot, \cdot)\) is the occupation measure associated with \(\pi'\).

**Proof of Lemma 3.** For the upper bound, note that because \(L(\pi_m, \lambda)\) is linear in \(\lambda\),
\[ \lambda_{m+1} = \text{Proj}_{\Lambda_M} \left\{ \lambda_m + \eta_m \cdot \partial L(\pi_m, \lambda_m) \right\} \]
is equivalent to
\[ \lambda_{m+1} = \arg \max_{\lambda \in \Lambda_M} \left\{ L(\pi_m, \lambda) - \frac{1}{2\eta_m} \| \lambda - \lambda_m \|^2 \right\}. \]
Then, by Lemma 2 we have
\[ L(\pi_m, \lambda) - L(\pi_m, \lambda_{m+1}) \leq (2\eta_m)^{-1} \left( \| \lambda - \lambda_m \|^2 - \| \lambda - \lambda_{m+1} \|^2 - \| \lambda_{m+1} - \lambda_m \|^2 \right) \]
\[ \leq (2\eta_m)^{-1} \left( \| \lambda - \lambda_m \|^2 - \| \lambda - \lambda_{m+1} \|^2 \right). \]

Next,
\[ L(\pi_m, \lambda) - L(\pi_m, \lambda_m) \leq (2\eta_m)^{-1} \left( \| \lambda - \lambda_m \|^2 - \| \lambda - \lambda_{m+1} \|^2 \right) + L(\pi_m, \lambda_{m+1}) - L(\pi_m, \lambda_m) \]
\[ = (2\eta_m)^{-1} \left( \| \lambda - \lambda_m \|^2 - \| \lambda - \lambda_{m+1} \|^2 \right) + \left\langle \partial L(\pi_m, \lambda_m), \lambda_{m+1} - \lambda_m \right\rangle \]
\[ \leq (2\eta_m)^{-1} \left( \| \lambda - \lambda_m \|^2 - \| \lambda - \lambda_{m+1} \|^2 \right) + \frac{\eta_m}{2} \| \partial L(\pi_m, \lambda_m) \|^2, \]
where the last inequality follows from the definition of \(\lambda_{m+1}\) and the non-expansive property of the projection. Then we obtain the upper bound.

For the lower bound, recall that we update \(\pi_m\) via
\[ \pi_{m+1}(\cdot | s) = \arg \min_{\pi(\cdot | s) \in \Delta_A} \left\{ \langle Q^{\pi_m, \lambda_m}(s, \cdot), \pi(\cdot | s) \rangle + \frac{1}{\eta_m} \text{KL}(\pi(\cdot | s) \| \pi_m(\cdot | s)) \right\}, \]
for each state \(s \in S\). Then, for an arbitrary stationary policy \(\pi' \in \Pi_S\), we have
\[ \pi_{m+1} = \arg \min_{\pi \in \Pi_S} \left\{ \mathbb{E}_{s \sim \nu^\pi} \left[ \langle Q^{\pi, \lambda_m}(s, \cdot), \pi(\cdot | s) \rangle + \frac{1}{\eta_m} \text{KL}(\pi(\cdot | s) \| \pi_m(\cdot | s)) \right] \right\} \]
where \(\nu^\pi_s\) is the state occupation measure associated with \(\pi'\).
Note that the space of the stationary policy, Πs, can be represented as the product space of simplex Δₐ. Consider Ω := Πs = (Δₐ)⊗|S|. and let

\[ g(\pi) := \mathbb{E}_{s \sim \nu'} \left[ \langle Q^{\pi, \lambda m}(s, \cdot), \pi(\cdot|s) \rangle \right], \]

\[ \Psi_\xi(\pi) := \mathbb{E}_{s \sim \nu'} \left[ \text{KL}(\pi(\cdot|s)\|\pi_m(\cdot|s)) \right] = \Phi^{\pi'}(\pi\|\pi_m), \]

where \( \Phi^{\pi'} \) is defined in [21]. Since \( g(\pi) \) is linear in \( \pi \), setting \( \pi = \pi' \), by Lemma 2, we obtain

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

which can be equivalently written as

\[ \eta_m^{-1} \cdot \left( \Phi^{\pi'}(\pi'_m\|\pi_{m+1}) - \Phi^{\pi'}(\pi'_m\|\pi_m) \right) \leq \mathbb{E}_{s \sim \nu'} \left[ \langle Q^{\pi, \lambda m}(s, \cdot), \pi'(\cdot|s) - \pi_{m+1}(\cdot|s) \rangle \right] + \mathbb{E}_{s \sim \nu'} \left[ \langle Q^{\pi, \lambda m}(s, \cdot), \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \rangle \right]. \]

We next derive an upper bound for the right-hand side of inequalities (31). Let \( \| \cdot \|_{TV} \) denotes the total variation norm of probability distributions. First, for each state \( s \in S \),

\[ \eta_m \cdot \mathbb{E}_{s \sim \nu} \left[ \langle Q^{\pi, \lambda m}(s, \cdot), \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \rangle \right] \leq \eta_m \cdot \sup_{a \in \mathcal{A}} \| Q^{\pi, \lambda m}(s, a) \cdot \| \pi_m(\cdot|s) - \pi_{m+1}(\cdot|s) \|_{TV} \]

\[ \leq \eta_m^2 \cdot \left( \sup_{s \in S} \sup_{a \in \mathcal{A}} \| Q^{\pi, \lambda m}(s, a) \| \right)^2 + 2 \cdot \| \pi_{m+1}(\cdot|s) - \pi_m(\cdot|s) \|_{TV} \]

\[ \leq \eta_m^2 \cdot \left( \sup_{s \in S} \sup_{a \in \mathcal{A}} \| Q^{\pi, \lambda m}(s, a) \| \right)^2 + \text{KL}(\pi_{m+1}(\cdot|s)\|\pi_m(\cdot|s)) \] (by Pinsker’s inequality).

Hence, by taking the average, we obtain

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

(32)

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

\[ \mathbb{E}_{s \sim \nu} \left[ \langle Q^{\pi, \lambda m}(s, \cdot), \pi'(\cdot|s) - \pi_m(\cdot|s) \rangle \right] = \mathbb{E}_{(s,a) \sim \nu} \left[ Q^{\pi, \lambda m}(s, a) - V^{\pi, \lambda m}(s) \right] \]

\[ = (1 - \gamma) \cdot (L(\pi', \lambda_m) - L(\pi_m, \lambda_m)). \]

(33)

Finally, combining (31)-(33), we obtain

\[ L(\pi', \lambda_m) - L(\pi_m, \lambda_m) \]

\[ \geq (1 - \gamma) \eta_m^{-1} \left( \Phi^{\pi'}(\pi_m + 1\|\pi_m) - \Phi^{\pi'}(\pi'_m\|\pi_m) \right) - \frac{\eta_m}{8(1 - \gamma)} \left( \sup_{s \in S} \sup_{a \in \mathcal{A}} \| Q^{\pi, \lambda m}(s, a) \| \right)^2. \]

□

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 \( \| \partial_\lambda L(\bar{\pi}_T, \lambda) \|^+ \).

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

\[
L(\pi_m, \lambda_m) - L(\pi_m, \lambda^*) = (\lambda_m - \lambda^*)^T \partial_\lambda L(\pi_m, \lambda_m).
\]

By the first part of Lemma 1, for any \( \lambda \), we have

\[
\eta_m \cdot (\lambda - \lambda_m)^T \partial_\lambda L(\pi_m, \lambda_m) = \eta_m \cdot (L(\pi_m, \lambda) - L(\pi_m, \lambda_m)) \\
\leq (\|\lambda_m - \lambda\|^2 - \|\lambda_{m+1} - \lambda\|^2) / 2 + \eta^2_m G^2 / 2.
\]

On the other hand, by the saddle point property of \((\pi^*, \lambda^*)\), we also have

\[
L(\pi_m, \lambda^*) \geq L(\pi^*, \lambda^*).
\]

In the following, we denote by \( L^* := L(\pi^*, \lambda^*) \). By combining inequalities \(34-35\), we obtain

\[
\eta_m \cdot (\lambda - \lambda^*)^T \partial_\lambda L(\pi_m, \lambda_m) = \eta_m \cdot (\lambda_m - \lambda^*)^T \partial_\lambda L(\pi_m, \lambda_m) \\
\leq (\|\lambda_m - \lambda\|^2 - \|\lambda_{m+1} - \lambda\|^2) / 2 + \eta^2_m G^2 / 2 + \eta_m \cdot (L(\pi_m, \lambda_m) - L^*)
\]

By taking the telescope sum of above inequality, for any \( \lambda \in \Lambda_M \), we have

\[
\sum_{m=0}^{T-1} \eta_m \cdot (\lambda - \lambda^*)^T \partial_\lambda L(\pi_m, \lambda_m) \\
\leq (\|\lambda_0 - \lambda\|^2 - \|\lambda_{T} - \lambda\|^2) / 2 + \left( \sum_{m=0}^{T-1} \eta^2_m / 2 \right) \cdot G^2 \\
+ \sum_{m=0}^{T-1} \eta_m \cdot (L(\pi_m, \lambda_m) - L^*)
\]

For the left hand side of \(37\), let

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

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

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

where \( r \) is the slackness constant in the definition of \( \Lambda_M \) in \(13\). Then it is easy to see that \( \bar{\lambda} \in \Lambda_M \). By \(37\), we have

\[
(\bar{\lambda} - \lambda^*)^T \zeta_T \leq \max_{\lambda \in \Lambda_M} \|\lambda - \lambda_0\|^2 / 2 + \left( \sum_{m=0}^{T-1} \eta^2_m / 2 \right) \cdot G^2 + \sum_{m=0}^{T-1} \eta_m \cdot (L(\pi_m, \lambda_m) - L^*)
\]

By the definition of \( \bar{\lambda} \), we also have

\[
(\bar{\lambda} - \lambda^*)^T \zeta_T = r \cdot \frac{([\zeta_T]^+)^T \zeta_T}{\|[\zeta_T]^+\|} = r \cdot \|[\zeta_T]^+\| = r \cdot \left( \sum_{m=0}^{T-1} \eta_m \right) \cdot \|[\partial_\lambda L(\bar{\pi}_T, \lambda)]^+\|.
\]
Similarly, under Assumption 3, by the second part in Lemma 1, we have

\[ \eta \]

Next, since \( D \) from inequalities (39) and (40), we have

\[ \pi \]

If we set \( \pi \) and \( \eta \), there exists finite constants \( \kappa_1 \) and \( \kappa_2 \) such that

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

Subsequently, we obtain

\[ \left\| \partial_\lambda L(\pi_T, \lambda) \right\| \leq \left( G^2 \cdot \left( 1 + \frac{5}{8} \kappa_2 \log(T) \right) + \Phi^{*\star}(\pi^* \| \pi_0) \right) \frac{1}{2r(1-\gamma)\kappa_1 \sqrt{T}}. \]

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

\[ \left\| \partial_\lambda L(\pi_T, \lambda) \right\| \leq \left( G^2 + (1-\gamma) \cdot \Phi^{*\star}(\pi^* \| \pi_0) \right) \frac{1}{2rT\eta} + \frac{1}{2} + \frac{1}{8(1-\gamma)} \frac{G^2 \eta}{2r}. \]

**We next prove the bound for** \( C(\pi_T) - L^* \). **We start with the upper bound. By the definition of** \( \pi_T \), we have

\[ C(\pi_T) - L^* = \sum_{m=0}^{T-1} \eta_m \cdot (L(\pi_m, \lambda_m) - L^*) - \sum_{m=0}^{T-1} \eta_m \cdot \lambda_m^T (D(\pi_m) - q). \]

From inequalities [39] and [40], we have

\[ \sum_{m=0}^{T-1} \eta_m \cdot (L(\pi_m, \lambda_m) - L^*) \leq \left( 1 - \gamma \right) \cdot \sum_{m=0}^{T-1} \eta_m \cdot \lambda_m^T (D(\pi_m) - q). \]

Next, since \( D(\pi_m) - q = \partial_\lambda L(\pi_m, \lambda_m) \), setting \( \lambda = 0 \) in [55], similarly, we obtain

\[ - \sum_{m=0}^{T-1} \eta_m \cdot \lambda_m^T (D(\pi_m) - q) \leq \frac{\|\lambda_0\|^2 + G^2 \cdot \sum_{m=0}^{T-1} \eta_m^2}{2 \sum_{m=0}^{T-1} \eta_m}. \]
Hence, if $\eta_m = \Theta(1/\sqrt{m})$, we have

$$C(\bar{\pi}_T) - L^* \leq \left( \frac{5G^2}{8} \cdot \kappa_2 \cdot \log(T) + \Phi^{\pi^*}(\pi^*\|\pi_0) + \frac{\|\lambda_0\|^2}{2} \right) \frac{1}{(1 - \gamma)\kappa_1\sqrt{T}}.$$  

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

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

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

$$C(\bar{\pi}_T) - L^* \geq -\|\lambda^*\| \|D(\bar{\pi}_T)\| + (\lambda^*)^T \Phi^{\pi^*}(\pi^*\|\pi_0) \frac{1}{2r(1 - \gamma)\kappa_1\sqrt{T}}.$$

Similarly, when if $\eta_m = \eta$,

$$C(\bar{\pi}_T) - L^* \leq \left( \frac{G^2}{8} \cdot \kappa_2 \cdot \log(T) \right) \frac{1}{T\eta} + \frac{5G^2\eta}{8(1 - \gamma)}.$$  

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

\[\square\]
