No Regret Reinforcement Learning Algorithms for Online Scheduling with Multi-Stage Tasks

Yongxin Xu , **Hengquan Guo** , **Ziyu Shao** and **Xin Liu***
ShanghaiTech University
{xuyx2022, guohq, shaozy, linxin7}@shanghaitech.edu.cn

Abstract

We study online task scheduling problems where tasks arrive sequentially and are processed by the platform or server. The service processes for tasks are multi-stage and are modeled as episodic Markov Decision Processes (MDPs). While processing a task, the system acquires rewards by consuming resources. The goal of the platform is to maximize the reward-to-cost ratio over a sequence of K tasks. Online scheduling with multi-stage tasks faces two major challenges: intra-dependence among the different stages within a task and inter-dependence among different tasks. These challenges are further exacerbated by the unknown rewards, costs, and task arrival distribution. To address these challenges, we propose the Robbins-Monro-based Value Iteration for Ratio Maximization ($\texttt{RM}^2\texttt{VI}$) algorithm. Specifically, $\texttt{RM}^2\texttt{VI}$ addresses "intra-dependence" through optimistic value iteration and handles "inter-dependence" using the Robbins-Monro method. The algorithm has a greedy structure and achieves a sub-linear regret of $O(K^{\frac{3}{4}})$, establishing the no-regret property (per-task). We test $\texttt{RM}^2\texttt{VI}$ in two synthetic experiments of sale promotion in E-commerce and machine learning job training in cloud computing. The results show $\texttt{RM}^2\texttt{VI}$ achieves the best reward-to-cost ratio compared with the baselines.

1 Introduction

Online task scheduling involves dynamically allocating incoming tasks to available resources for real-time processing, with applications spanning cloud computing [Agarwal and Jain, 2014; Arunarani *et al.*, 2019; Guo *et al.*, 2012], crowdsourcing [Alabbadi and Abulkhair, 2021; Khazankin *et al.*, 2011; Deng *et al.*, 2015], and advertising [Deane, 2012; Kanuri *et al.*, 2018]. Existing studies [Chen *et al.*, 2020; Cho *et al.*, 2015; Zhao *et al.*, 2022] have largely focused on homogeneous tasks and simplified the problem to single-stage processes. However, these approaches fail to address the heterogeneity and complexity inherent in practical systems, where tasks often involve multiple stages with distinct resource requirements. For instance, a typical machine learning workflow consists of several interdependent stages, including data processing, pre-training, fine-tuning, and validation. These stages not only exhibit strong coupling but also have heterogeneous resource demands. Pre-training may require significant memory and computational power to process large datasets, whereas fine-tuning and validation are less resource-intensive. Furthermore, different tasks frequently compete for shared hardware resources, introducing additional dependencies and contention. To capture this heterogeneity and complexity, we model the service process of each task as an episodic Markov Decision Process (MDP) and aim to optimize performance over a sequence of tasks. Specifically, our objective is to maximize the long-term reward-to-cost ratio (see the explicit formulation in (1)). This approach accounts for task heterogeneity, multi-stage dependencies, and shared resource constraints, addressing key challenges overlooked in prior work.

As discussed, online task scheduling faces two major challenges. The first challenge arises from the intra-dependence among different stages or steps within a task. This dependence is further complicated by the inherent uncertainty in practical systems, where the rewards, costs, and transitions between task stages are often unknown and must be learned during the process. The second challenge stems from the inter-dependence among tasks, where heterogeneous tasks arrive sequentially and compete for shared resources. This challenge is again exacerbated by the fact that the task arrival distribution is unknown.

To address the "intra-dependence" within a task, we adopt the principle of "optimism in the face of uncertainty" [Auer *et al.*, 2002] to learn both the reward and cost value functions. These functions capture the long-term impact of decisions made at each stage of a task. This approach allows the system to strategically allocate resources to critical stages within a task that promise high potential rewards while avoiding resource wastage on less beneficial stages. To address the "inter-dependence" among tasks, we directly track the average system performance, represented by the reward-to-cost ratio. This metric serves as a guiding signal to balance resource allocation across tasks, enabling the system to prioritize resources for tasks with high potential reward gains while

*Corresponding Author

avoiding resource expenditure on tasks with lower expected returns. We summarize our contribution in the following.

Algorithm Design: We propose a novel online scheduling algorithm, RM^{2}VI, for multi-stage task processing. It leverages the Robbins-Monro iteration to directly track the system's goal (i.e., the long-term reward-to-cost ratio) with only task samples, thereby avoiding explicitly estimating the task arrival distribution. It incorporates optimistic value iteration to effectively learn the unknown MDP and capture the long-term effects of decisions within tasks. By synthesizing a unified surrogate decision function that balances reward acquisition and cost consumption, RM^{2}VI enables an efficient greedy decision-making framework designed to maximize the reward-to-cost ratio.

Theoretical Analysis: We prove that RM^{2}VI achieves a no-regret learning performance against the optimal policy in hindsight (c.f., Theorem 1). Specifically, it achieve a sub-linear regret of $O(\sqrt{HMSA}T^{\frac{3}{4}})$ where M is the number of task types, and S and A are the sizes of state and action spaces, $T = HK$, H is the number of steps in a task and K is the number of tasks, respectively. To prove this result, we decompose the regret into two components corresponding to the greedy decision-making and value iteration learning processes. A key challenge lies in addressing the coupling between these two components (learning and decision). We address this challenge using a Lyapunov/potential drift technique, where the Lyapunov/potential functions capture the decision-making mismatches such that we can quantify the cumulative mismatches throughout the entire learning and decision-making process.

Experiments: We simulated two online task scheduling settings, sale promotion in E-commerce and machine learning job training in cloud computing, to justify RM^{2}VI. We compare the performance of RM^{2}VI against state-of-the-art reinforcement learning algorithms for the metric of reward-to-cost ratio. Our results demonstrate that RM^{2}VI achieves significant gains compared to these baselines.

2 Related Work

Online task scheduling has been extensively studied through the lens of online learning, particularly in the contexts of budget-constrained bandits/MDPs and reward-to-cost ratio bandits/MDPs.

Budget-constrained bandits have been widely explored in the literature [Tran-Thanh *et al.*, 2012; Ding *et al.*, 2013; Xia *et al.*, 2017; Badanidiyuru *et al.*, 2018; Agrawal and Devanur, 2014; Agrawal and Devanur, 2016; Badanidiyuru *et al.*, 2014; Han *et al.*, 2023; Slivkins *et al.*, 2023; Guo and Liu, 2024], where the objective is to maximize cumulative rewards under a finite resource budget. However, this line of research typically assumes prior knowledge of the resource budget, and the interaction process terminates when the budget is exhausted. This differs from our setting, as we do not impose a predefined budget limit, and the interaction process is designed to proceed continuously without termination. The budget-bandit settings have been further extended to constrained MDP (CMDP) to capture the stateful property in the learning process [Altman, 1999; Borkar, 2005; Bhatnagar, 2010; Wei *et al.*, 2022; Wachi *et al.*, 2024]. The key difference is again that our setting does not impose any constraints or require any prior knowledge of these constraints.

Ratio-maximization bandits have also been studied in the literature [Watanabe *et al.*, 2018; Cayci *et al.*, 2020; Cesa-Bianchi *et al.*, 2021; Heyden *et al.*, 2024], with the objective of maximizing the ratio of cumulative rewards to cumulative costs, which aligns with the goal of this paper. However, these settings represent a simplified subclass of our problem, as they consider homogeneous tasks with a single stage—meaning only one type of task exists, and each task follows a single-stage service process. Though this framework has been extended to reward-to-cost ratio MDPs (RMDPs) to incorporate stateful dynamics within the learning process [Abounadi *et al.*, 2001; REN and KROGH, 2005; Tanaka, 2017; Suttle *et al.*, 2021], they still consider the setting of homogeneous tasks. The most relevant work in this direction is [Suttle *et al.*, 2021], which introduced cost-aware reinforcement learning algorithms capable of achieving asymptotic convergence. However, we study heterogeneous tasks with unknown task distribution and provide a finite-time (no-regret) performance guarantee.

For the reward-to-cost ratio setting, [Neely, 2021] was the first to highlight the critical relationship between the optimal policy and the task arrival distribution, proposing a Robbins-Monro-based fast learning algorithm. [Neely, 2024] generalizes this work by introducing additional explicit constraints, ensuring that the reward-to-cost ratio is maximized while satisfying these constraints over the long term. However, both works assume full knowledge of the rewards and costs of tasks when making decisions. To model the uncertainty of rewards and costs for task processing, [Xu *et al.*, 2024] integrates bandit learning into the framework of [Neely, 2021], proposing an optimistic learning algorithm to estimate these unknown quantities. While these works account for heterogeneous tasks, they simplify the task structure to a single stage, which cannot handle the complexity of multi-stage tasks as addressed in our paper. The multi-stage structure introduces significant challenges due to the intra-dependence among stages (i.e., bandits vs. MDPs), and it becomes even more complicated when costs are involved. To address these challenges, we need to carefully design the value iteration learning with the estimated reward-to-cost ratio. This design is crucial for decoupling the inter-dependence between tasks and enables a greedy decision-making framework for optimizing the reward-to-cost ratio in the long term.

3 System Model

We study an online task scheduling system represented by a tuple $(\mathcal{M}, \mathcal{S}, \mathcal{A}, H, \mathbb{P}, r, c)$. At each time step $k \in [K]$, a task of type $m_k \in \mathcal{M}$ arrives in the system, drawn from an unknown arrival distribution $\mathbb{P}_{\mathcal{M}}$, where $\mathcal{M}$ is the set of task types. Each task undergoes a process consisting of H steps, modeled as an episodic Markov decision process (MDP). In the episodic MDP of a task, $\mathcal{S}$ is the state space with $|\mathcal{S}| = S$, $\mathcal{A}$ is the action space with $|\mathcal{A}| = A$, and $\mathbb{P} = \{\mathbb{P}_h\}_{h=1}^{H}$ is a collection of transition kernels. The reward function is denoted by r, and the cost function by c. For simplicity, we

assume all tasks share the same state space $\mathcal{S}$ and action space $\mathcal{A}$, and that $\mathbb{P}, r, c$ are explicitly presented as dependent on the task type m. At each step $h \in [H]$, $\mathbb{P}_h(\cdot|m, s, a)$ represents the transition probability to the next state, given state s, action a, and task type m at step h. The reward function at step h is defined as $r_h : \mathcal{M} \times \mathcal{S} \times \mathcal{A} \to [r_{\min}, r_{\max}]$, while the cost function is defined as $c_h : \mathcal{M} \times \mathcal{S} \times \mathcal{A} \to [c_{\min}, c_{\max}]$.

Given the definitions above, the interaction protocol operates as follows. At each time or episode k, the system observes a task type $m_k \in \mathcal{M}$, drawn from $\mathbb{P}_{\mathcal{M}}$. The initial state of the task, s_1^k, is sampled from $\mu(m_k)$. At each stage or step h, the system takes action a_h^k based on the current state s_h^k and then receives a reward $r_h(m_k, s_h^k, a_h^k)$ and incurs a cost $c_h(m_k, s_h^k, a_h^k)$. Subsequently, the task transitions to the next state, s_{h+1}^k, which is sampled from the transition probability $\mathbb{P}_h(\cdot \mid m_k, s_h^k, a_h^k)$. Note that task arrivals, along with the corresponding rewards and costs, are i.i.d. generated across task types, episodes, and internal steps.

For a given policy π, which is a collection of functions $\{\pi_h : \mathcal{M} \times \mathcal{S} \to \mathcal{A}\}_{h=1}^H$, we can define its reward value functions (r_h^k is the short notation for $r_h^k(m, s_h^k, a_h^k)$)

$$V_{k,h}(m, s) = \mathbb{E}\left[\sum_{i=h}^{H} r_i^k | s_h^k = s\right],$$

$$Q_{k,h}(m, s, a) = r_h^k + \mathbb{E}\left[\sum_{i=h+1}^{H} r_i^k | s_h^k = s, a_h^k = a\right].$$

Similarly, we define the cost value functions (c_h^k is the short notation for $c_h^k(m, s_h^k, a_h^k)$)

$$W_{k,h}(m, s) = \mathbb{E}\left[\sum_{i=h}^{H} c_i^k | s_h^k = s\right],$$

$$C_{k,h}(m, s, a) = c_h^k + \mathbb{E}\left[\sum_{i=h+1}^{H} c_i^k | s_h^k = s, a_h^k = a\right].$$

Reward-to-Cost Ratio Maximization: The platform needs to serve a sequence of K tasks, and the cumulative reward-to-cost ratio over the K tasks is defined as follows

$$\theta^\pi = \frac{\sum_{k=1}^K \mathbb{E}\left[V_{k,1}^\pi(m_k, s_1^k)\right]}{\sum_{k=1}^K \mathbb{E}\left[W_{k,1}^\pi(m_k, s_1^k)\right]}. \quad (1)$$

The goal of the platform is to design an optimal policy π^* such that the long-term reward-to-cost ratio is maximized, where θ^* is denoted as the optimal value. The model captures plenty of practical applications illustrated in the following.

Sale Promotion: E-commerce platforms (e.g., Amazon) frequently employ discounts to boost sales during promotional campaigns. These discounts constitute the cost incurred by the platform, while the reward is the revenue generated from customer purchases. The process of a customer's interaction with a discount can be modeled as a multi-stage task: for instance, stage 1 involves the customer noticing the promotion, stage 2 involves the customer adding discounted items to their cart, and stage 3 involves the customer completing the purchase. The transition probabilities between stages depend on various factors, including the customer type, discount size, and their current stage in the process. Different discount strategies may elicit varying customer responses, directly impacting the reward (sales revenue) and the cost (discount size). Maximizing the reward-to-cost ratio in this scenario ensures the platform's promotional spending is efficient, balancing revenue generation with discount expenditures.

Cloud Computing: Cloud computing platforms (e.g., Microsoft Azure or Google Cloud) routinely train various machine learning models, which consume computational resources. In this setting, the reward corresponds to the accuracy or quality of the trained model, while the cost represents the computational resources consumed, such as GPU hours or energy usage. The process of training a model can be modeled as a multi-stage task. For example, stage 1 involves initializing the model and conducting initial training epochs, stage 2 involves intermediate evaluation and adjustments (e.g., hyperparameter tuning), and stage 3 involves final training and validation. The transition probabilities between these stages depend on the model's complexity, the amount of data, and the number of epochs allocated. Maximizing the reward-to-cost ratio ensures that computational resources are used efficiently, achieving a good balance between a high model accuracy and resource consumption.

Convergence Gap & Regret: The metric we use for evaluating a policy π is convergence gap against θ^*:

$$Gap(K) = |\theta^* - \theta^\pi|.$$

Based on the convergence gap, we define its regret performance for a given policy π:

$$\mathcal{R}(K) = K \cdot Gap(K).$$

4 Algorithm Design

To find the best policy and maximize the reward-to-cost ratio, we face two major challenges: inter-dependence among different tasks and intra-dependence among different steps within a task. To make things even more challenging, the task arrival distribution and MDP of a task itself are both unknown. To decouple the intra-dependence with an unknown MDP model, we leverage the idea of optimism in performing value iteration to learn the individual reward/cost value function, which captures the long-term effect of decisions. To decouple the inter-dependence challenge with unknown arrival distribution of task types $\mathbb{P}_{\mathcal{M}}$, we leverage a stochastic gradient type algorithm (i.e., Robbins-Monro algorithm) to learn the best value and policy alternatively. Interestingly, with these two decompositions, we can just make a "greedy" decision for any task given any state. We then break two major components and their intuition in our algorithm design. To simplify the notation, we define $(V - W)(s) = V(s) - W(s)$ and $(Q - C)(s, a) = Q(s, a) - C(s, a)$.

4.1 Stochastic Approximation

According to the definition of the optimal θ^* in (1), we can rewrite it below (its proof can be found in Appendix A)

$$\max_\pi \left[\sum_{k=1}^K \mathbb{E}\left[V_{k,1}^\pi(m_k, s_1^k) - \theta^* W_{k,1}^\pi(m_k, s_1^k)\right]\right] = 0. \quad (2)$$

This becomes a fixed point problem of (θ, π) and (θ^*, π^*) is the fixed point. Now let's build some intuition by first assuming θ^* is known, we can find a policy such that for any time $k \in [K]$:

$$\arg\max_{\pi} \; V^{\pi}_{k,h}(m_k, s^k_1) - \theta^* W^{\pi}_{k,h}(m_k, s^k_1).$$

This is reasonable because the task arrival is independent and finding a policy that maximizes $(V^{\pi}_{k,h} - \theta^* W^{\pi}_{k,h})(m_k, s^k_1)$ for every task will certainly maximize its overall expectation $\mathbb{E}[(V^{\pi}_{k,h} - \theta^* W^{\pi}_{k,h})(m_k, s^k_1)]$. Though the problem in 2 can be solved by dynamic programming (DP) [Held and Karp, 1962], it requires the full knowledge of MDP (i.e., reward, cost, and transition kernel) and might suffer from large computational overhead. Therefore, we turn to the value iteration type algorithm to improve the value functions and their associated policy. Given the current value or action-value functions, we are able to make the greedy decision a^k_h, for any task m_k given any state s^k_h as follows

$$a^k_h = \arg\max_{a \in \mathcal{A}} \; Q_{k,h}(m_k, s^k_1, a) - \theta^* C_{k,h}(m_k, s^k_h, a).$$

This would keep improving our policy and ideally converge to the optimal value functions and policy. Now, the left challenge is to estimate the optimal θ^*. This can be done with the Robbins-Monro algorithm using the "task sample" as follows

$$\theta_{k+1} = [\theta_k + \eta(V_{k,1}(s^k_1) - \theta_k W_{k,1}(s^k_1)]^{\theta_{\max}}_{\theta_{\min}}$$

where V and W are the current value functions, $\eta = 1/c_{\min}\sqrt{K}$ is the learning rate carefully chosen to guarantee a sub-linear regret with the lower and upper bounds $\theta_{\max} = r_{\max}/c_{\min}$ and $\theta_{\min} = r_{\min}/c_{\max}$. These are two major components to improve the policy given the value functions at time k. However, as MDP models are unknown, we need to learn the reward and cost value functions as introduced in the next section.

4.2 Double-Optimistic Value Iteration

In this section, we leverage the idea of optimism to learn the value functions motivated by the optimistic value iteration method, UCBVI [Azar *et al.*, 2017]. We first estimate the unknown transition kernel with history information and then use the empirical transition kernel and realized reward or cost to update value functions for the current policy. Note that we need to implement carefully the idea of optimism for MDP with constraint constraints. In particular, we impose "UCB" and "LCB" to estimate reward and cost value functions, respectively, such that the reward-to-cost ratio is guaranteed to be optimistic against (or larger than) its true value.

At the beginning of the episode k, the agent observes the task type $m_k = m$, we estimate the transition kernel with the historical information as follows:

$$\hat{\mathbb{P}}^k_h(s'|m, s, a) = \frac{N^k_h(m, s, a, s')}{N^k_h(m, s, a)}, \; \forall s' \in \mathcal{S}, \tag{3}$$

where $N^k_h(m, s, a, s')$ and $N^k_h(m, s, a)$ are the visit counts at (m, s, a, s') and (m, s, a) observed in the historical dataset

Algorithm 1 RM^{2}VI

Input: K, H,ι, $c_{\min}$, $r_{\max}$, $\theta_{\max}$ and $\theta_{\min}$.
for $k = 1, 2, \ldots, K$ **do**
  Given a task m_k with initial state $s^k_1(m_k)$.
  for $h = 1, 2, \ldots, H$ **do**
    Cost-aware greedy decision:

$$a^k_h = \arg\max_{a \in \mathcal{A}} (\hat{Q}_{k,h} - \theta_k \check{C}_{k,h})(m_k, s^k_h, a). \tag{6}$$

    Observe: reward r^k_h, c^k_h and s^k_{h+1} and include the data $\{h, m_k, s^k_h, a^k_h, s^k_{h+1}, r^k_h, c^k_h\}$ to $\mathcal{H}$.
  end for
  Reward-to-Cost Ratio Learning:

$$\theta_{k+1} = [\theta_k + \eta(\hat{V}_{k,1} - \theta_k \check{W}_{k,1})(m_k, s^k_1)]^{\theta_{\max}}_{\theta_{\min}}. \tag{7}$$

  Double–Optimistic Value Estimation:

$$\hat{V}_{k+1,h}, \hat{Q}_{k+1,h}, \check{W}_{k+1,h}, \check{C}_{k+1,h} = \text{DOVI}(\theta_{k+1}, \mathcal{H}).$$

end for

$\mathcal{H}$. Then the estimated reward and cost action-value functions are updated as follows:

$$\begin{aligned} \hat{Q}_{k,h}(m, s, a) = &\min\{Hr_{\max}, r_h(m, s, a) \\ &+ (\hat{\mathbb{P}}^k_h \hat{V}^k_{h+1})(m, s, a) + \beta(N^k_h(m, s, a))\Big\} \end{aligned} \tag{4}$$

$$\begin{aligned} \check{C}_{k,h}(m, s, a) = &\max\{Hc_{\min}, c_h(m, s, a) \\ &+ (\hat{\mathbb{P}}^k_h \check{W}^k_{h+1})(m, s, a) - \beta(N^k_h(m, s, a))\Big\} \end{aligned} \tag{5}$$

where $(\hat{\mathbb{P}}^k_h \hat{V}^k_{h+1})(m, s, a) = \sum_{s'} \hat{\mathbb{P}}_h(s'|m, s, a) \hat{V}^k_{h+1}(m, s')$. The bonus term $\beta(N) \triangleq H\sqrt{\frac{\log(SAT/\iota)}{N}}$ with ι being parameters. As discussed, we impose the positive bonus into $\hat{Q}_{k,h}$ and the negative bonus into $\check{C}_{k,h}$ to achieve an overall optimism over the reward-to-cost ratio.

Then, for all $s \in \mathcal{S}$, we estimate reward and cost value function by assigning the value of reward and cost action-value function whose action can maximize $\hat{Q}_{k,h}(m, s, a) - \theta_k \check{C}_{k,h}(m, s, a)$, which coincides with stochastic approximation based decision leading to the convergence of policy.

We combine the above to get RM^{2}VI.

5 Theoretical Results

In this section, we state our main result of the RM^{2}VI algorithm in the following theorem.

Theorem 1. *Under RM^{2}VI algorithm, for a large $K \geq HM^2S^2A^2$, we have the convergence gap and regret to be bounded as follows*

$$Gap(K) = O\left(\sqrt{HMAS}T^{-\frac{1}{4}}\right),$$
$$\mathcal{R}(K) = O\left(\sqrt{HMAS}T^{\frac{3}{4}}\right).$$

Algorithm 2 DOVI

Input: θ_k, $\mathcal{H}$.
for $h = H, H-1, \ldots, 1$ **do**
  for $(s,a) \in \mathcal{S} \times \mathcal{A}$ **do**
    if $(s,a) \in \mathcal{H}$ **then**
      Estimate $\hat{\mathbb{P}}_h^k(s^{'}|m_k, s, a)$ according to (3).
      Estimate $\hat{Q}_{k,h}(m_k, s, a)$ and $\check{C}_{k,h}(m_k, s, a)$ according to (4)-(5).
    end if
  end for
  for $s \in \mathcal{S}$ **do**
    Estimate $\hat{V}_{k,h}(m_k, s)$ and $\check{W}_{k,h}(m_k, s)$:

$$a^{'} = \arg\max_{a \in \mathcal{A}} (\hat{Q}_{k,h}(m, s, a) - \theta_k \check{C}_{k,h}(m, s, a))$$

$$\hat{V}_{k,h}(m, s) = \hat{Q}_{k,h}(m, s, a^{'}), \tag{8}$$

$$\check{W}_{k,h}(m, s) = \check{C}_{k,h}(m, s, a^{'}). \tag{9}$$

  end for
end for
return $\hat{V}_{k,h}, \hat{Q}_{k,h}, \check{W}_{k,h}, \check{C}_{k,h}$

Theorem 1 implies RM^{2}VI achieve a sub-linear regret performance $O(T^{3/4})$ when $K \geq HM^2S^2A^2$, which implies the convergence gaps of $O(T^{-1/4})$. These are the first results for online task scheduling, where the tasks have a complicated Markovian structure without any prior knowledge of the MDP model and the task arrival distributions. These results also indicate that RM^{2}VI can quickly identify an effective and efficient policy that converges to the optimal ratio θ^* with the integral design of cost-aware decision and double-optimistic value iteration.

Next, we present the detailed proof of Theorem 1 and focus on the analysis of the convergence gap $Gap(K)$. We first decompose $Gap(K)$ into the two items related to the cost-aware decision and double-optimistic value iteration, respectively, and then establish them, individually.

5.1 Proof of Theorem 1

Recall the definition of θ^π with π being our RM^{2}VI algorithm and define its optimistic ratio

$$\hat{\theta}^\pi = \frac{\sum_{k=1}^K \mathbb{E}\left[\hat{V}_{k,1}(m_k, s_1^k)\right]}{\sum_{k=1}^K \mathbb{E}\left[\check{W}_{k,1}(m_k, s_1^k)\right]},$$

which is the optimistic estimation of θ^π to bridge the optimal value θ^* in the Gap(K). By triangle inequality, we have

$$Gap(K) \leq \underbrace{|\theta^* - \hat{\theta}^\pi|}_{\text{decision error}} + \underbrace{|\hat{\theta}^\pi - \theta^\pi|}_{\text{estimation error}} \tag{10}$$

The first term in (10) is related to the optimality of cost-aware decision, which is intuitively small because RM^{2}VI takes the greedy decision to maximize $\hat{Q}_{k,h} - \theta_k \check{C}_{k,h}$ for every step that is consistent with maximizing the reward-cost ratio $\hat{Q}_{k,h}/\check{C}_{k,h}$ (or $\hat{V}_h/\check{W}_h$). The bound of decision error is established in Lemma 1. The second term is related to the estimation errors of value iteration because it is associated with the same policy π. This captures the bias of optimistic learning on reward and cost value functions as in Lemma 2.

Lemma 1. *Under RM^{2}VI, when $K \geq HM^2S^2A^2$, we have*

$$|\theta^* - \hat{\theta}^\pi| \leq O\left(\sqrt{HMAS}T^{-\frac{1}{4}}\right).$$

Lemma 2. *Under RM^{2}VI, we have*

$$|\hat{\theta}^\pi - \theta^\pi| \leq O\left(HMAST^{-\frac{1}{2}}\right).$$

To prove Lemma 1 and 2, we introduce the following key lemma.

Lemma 3. *Let $V_{k,h}^*(m_k, s)$ and $W_{k,h}^*(m_k, s)$ be the optimal real reward and cost value functions; $V_{k,h}(m_k, s)$ and $W_{k,h}(m_k, s)$ are the true reward and cost value functions under RM^{2}VI. We have for any time $k \in [K]$ and state $s \in \mathcal{S}$ such that*

$$\mathbb{E}\left[(\hat{V}_{k,h} - \theta_k \check{W}_{k,h})(m_k, s)\right] \geq \mathbb{E}\left[(V_{k,h}^* - \theta_k W_{k,h}^*)(m_k, s)\right]$$

$$\mathbb{E}\left[(\hat{V}_{k,h} - \theta_k \check{W}_{k,h})(m_k, s)\right] \geq \mathbb{E}\left[(V_{k,h} - \theta_k W_{k,h})(m_k, s)\right]$$

The first inequality shows the optimistic property of RM^{2}VI algorithm over the optimal surrogate value functions, which is the key to prove Lemma 1 and also the most challenging part in our analysis; the second inequality shows it is also an optimistic estimation of reward-to-cost ratio. This is intuitively true if the individual value function has the optimistic property and is necessary for proving Lemma 1. The detailed proof can be found in Appendix C.1.

5.2 Analysis of Estimation Errors

Based on Lemma 3, we then study the errors of value iteration in Lemma 2. We sketch the key steps in the proof, and the completed version can be found in Appendix C. We further decompose the estimation error by the triangle inequality

$$|\hat{\theta}^\pi - \theta^\pi| \leq \epsilon(K) + \kappa(K)$$

where $\epsilon(K)$ and $\kappa(K)$ are the estimation errors w.r.t. reward and cost value functions, respectively,

$$\epsilon(K) := \left|\hat{\theta}^\pi - \frac{\sum_{k=1}^K \mathbb{E}\left[V_{k,1}(m_k, s_1^k)\right]}{\sum_{k=1}^K \mathbb{E}\left[\check{W}_{k,h}(m_k, s_1^k)\right]}\right|,$$

$$\kappa(K) := \left|\frac{\sum_{k=1}^K \mathbb{E}\left[V_{k,1}(m_k, s_1^k)\right]}{\sum_{k=1}^K \mathbb{E}\left[\check{W}_{k,h}(m_k, s_1^k)\right]} - \theta^\pi\right|.$$

Then, we can bound these two errors as follows

$$\begin{aligned} &\epsilon(K) + \kappa(K) \\ \leq &\frac{1}{KHc_{\min}} \left|\sum_{k=1}^K \mathbb{E}\left[\hat{V}_{k,h}(m_k, s_1^k) - V_{k,1}(m_k, s_1^k)\right]\right| \\ + &\frac{r_{\max}}{KHc_{\min}^2} \left|\sum_{k=1}^K \mathbb{E}\left[W_{k,1}(m_k, s_1^k) - \check{W}_{k,h}(m_k, s_1^k)\right]\right|, \end{aligned} \tag{11}$$

where the inequality holds due to the boundedness of reward and cost functions.

Lemma 4. *Under RM^{2}VI, we have*

$$\left|\sum_{k=1}^{K}\mathbb{E}\left[(\hat{V}_{k,h}-V_{k,1})(m_k,s_1^k)\right]\right| = O(H^2MAS\sqrt{T}),$$
$$\left|\sum_{k=1}^{K}\mathbb{E}\left[(W_{k,1}-\check{W}_{k,h})(m_k,s_1^k)\right]\right| = O(H^2MAS\sqrt{T}).$$

The proof of the lemma is motivated by [Azar *et al.*, 2017], and the key step is to compare the Bellman equation update with the value iteration under RM^{2}VI. The detailed proof can be found in Appendix C.1. Finally, we plug Lemma 4 to (11) such that

$$\epsilon(K)+\kappa(K)\le O(HMAST^{-\frac{1}{2}}),$$

which proves Lemma 2.

5.3 Analysis of Decision Errors

To prove a no-regret performance in Lemma 1, we first establish an upper bound of $\mathbb{E}\left[(\theta_k-\theta^*)^2\right]$ in Lemma 5, and we then carefully control the cumulative errors in stochastic approximation.

Lemma 5. *Under RM^{2}VI, we bound the stochastic approximation convergence gap as follows:*

$$\left|\theta^*-\hat{\theta}^\pi\right| \le \max\Bigg\{|\hat{\theta}^\pi-\theta^\pi|, \tag{12}$$
$$\frac{\sqrt{2C_2}}{Kc_{\min}}\sum_{k=1}^{K}\sqrt{\mathbb{E}\left[(\theta_k-\theta^*)^2\right]}\Bigg\}. \tag{13}$$

We sketch the proof by considering two cases.

1) When $\theta^*\le\hat{\theta}^\pi$, the term in (12) is upper bound of $\hat{\theta}^\pi-\theta^\pi$ according to the definition of θ^*.

2) When $\theta^*\ge\hat{\theta}^\pi$, we have the decision in our algorithm such that

$$\begin{aligned}&\mathbb{E}\left[(\hat{V}_{k,1}-\theta_k\check{W}_{k,1})(m_k,s_1^k)|\theta_k\right]\\ \ge&\mathbb{E}\left[V^*_{k,1}(m_k,s_1^k)|\theta_k\right]-\theta_k\mathbb{E}\left[W^*_{k,1}(m_k,s_1^k)|\theta_k\right]\\ \ge&v^*-\theta_t w^*,\end{aligned}$$

where π^* is any possible policy, the first inequality holds due to the greedy decision and Lemma 3; the second inequality holds because (v^*,w^*) lies within the closure of the possible policy set. By adding $(\theta_k-\theta^*)\check{W}_{k,1}(m_k,s_1^k)$ on both sides of the above inequality, we further have

$$\begin{aligned}&\mathbb{E}\left[(\hat{V}_{k,1}-\theta^*\check{W}_{k,1})(m_k,s_1^k)\right]\\ \ge&\mathbb{E}\left[(\theta_k-\theta^*)(W^*_{k,1}(m_k,s_1^k)-w^*)\right].\end{aligned}$$

Taking the summation and rearranging the above inequality, we have

$$\left|\theta^*-\hat{\theta}^\pi\right|\le\frac{\sum_{k=1}^{K}\mathbb{E}\left[|\theta_k-\theta^*|\left|\check{W}_{k,1}(m_k,s_1^k)-w^*\right|\right]}{KHc_{\min}}$$

Finally, we use Cauchy-Schwarz inequality to establish (13) in Lemma 5. More details can be found in the Appendix.

Note (12) in Lemma 5 has been proved in the previous subsection. We focus on bounding the cumulative quadratic errors $\sum_{t=1}^{T}\sqrt{\mathbb{E}\left[(\theta_k-\theta^*)^2\right]}$. We use the Lyapunov drift analysis [Neely, 2010; Srikant and Ying, 2014] to study this key term. Define the Lyapunov function $\delta_k\triangleq\frac{1}{2}(\theta_k-\theta^*)^2$ and its drift $\Delta(k)\triangleq\delta_{k+1}-\delta_k$. We have the following lemma.

Lemma 6. *Under RM^{2}VI, we have the expected Lyapunov drift to be bounded as follows:*

$$\begin{aligned}&\mathbb{E}\left[\Delta(k)\right]\le-2Hc_{\min}\eta\mathbb{E}[\delta_k]+\eta^2b+\\ &\theta_{gap}\eta\mathbb{E}\left[\left(\hat{V}_{k,1}-V_{k,1}-\theta_k(\check{W}_{k,1}-W_{k,1})\right)(m_k,s_1^k)\right],\end{aligned}$$

in which $b\ge\frac{1}{2}\mathbb{E}\left[(\hat{V}_{k,1}-\theta_k\check{W}_{k,1})^2(m_k,s_1^k)\right]$, $\forall k$ *and* $\theta_{gap}\triangleq\theta_{\max}-\theta_{\min}$.

Therefore, according to the Cauchy-Schwarz inequality, we have the following lemma.

Lemma 7. *Under RM^{2}VI, we have*

$$\sum_{k=1}^{K}\sqrt{\mathbb{E}\left[(\theta_k-\theta^*)^2\right]}=O(\sqrt{HMAS}T^{\frac{3}{4}}). \tag{14}$$

Finally, by combining Lemma 2 and 7 into Lemma 5, we prove Lemma 1.

6 Experiment

We evaluate RM^{2}VI via two sets of simulated experiments: 1) sale promotion; and 2) cloud computing. The metric is the empirical cumulative reward-to-cost ratio $\frac{\sum_{k=1}^{K}\sum_{h=1}^{H}r_h(m_k,s_h^k,a_h^k)}{\sum_{k=1}^{K}\sum_{h=1}^{H}c_h(m_k,s_h^k,a_h^k)}$. In RM^{2}VI algorithm, we let the learning rate be $\eta_k=1/(c_{\min}\sqrt{K})$. We use a simplified version of samples to substitute for value iteration in RM^{2}VI for fairness since all the baselines are model-free algorithms.

Baselines: We consider the following algorithms to compare the performance of RM^{2}VI:

- Q-learning with UCB (Q-UCB) [Jin *et al.*, 2018].
- Cost-Aware Relative Value Iteration (CARVI) [Suttle *et al.*, 2021].
- Cost-Aware Actor-Critic (CAAC) [Suttle *et al.*, 2021].

These baselines all make greedy decisions (i.e., maximize the reward-cost ratio for every task) and ignore the distribution of task types. We use $\frac{r_h^k}{c_h^k}$ as the reward function in Q-UCB since it cannot handle the cost function. We run the experiments for 10 trails and plot the average results with the light-shaded areas indicating the standard deviation.

6.1 Sale Promotion Scenario

We simulate a sales promotion scenario where the users arrive at the shopping platform sequentially, and each user will have at most $H=5$ rounds of promotion decided by the platform. The total number of users is $K=10000$. We assume that there are two groups of users, $\{male, female\}$, and two types of goods, $\{electronicproducts, cosmetics\}$. We have four types of tasks in $\mathcal{M}$ by concatenating user

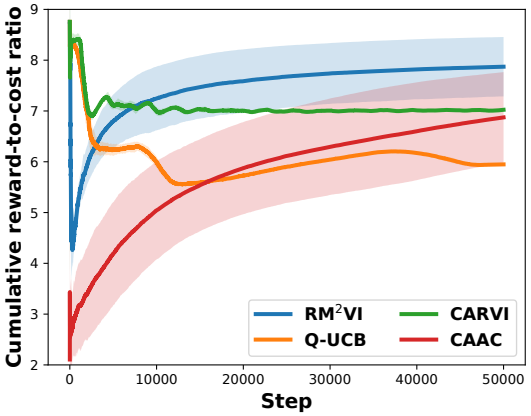

Figure 1: Performance Comparison in a Sales Promotion Scenario.

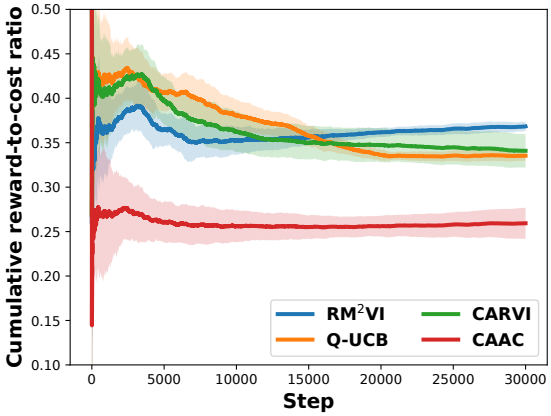

Figure 2: Performance Comparison in a Cloud Computing Scenario.

and goods types. The distribution of these four task types $\{male \times electronicproducts, male \times cosmetics, female \times electronicproducts, female \times cosmetics\}$ is draw from $(0.4, 0.1, 0.2, 0.3)$.

The states of users are in $\{not\ interested, cart, buy\}$, and three actions/discounts that the shopping platform can take to persuade the users to buy their products: $(90\%, 75\%, 50\%)$. We set the transition to reflect the behavior of users in real-life. For example, the tasks $(male \times electronicproducts)$ and $(female \times cosmetics)$ have a higher probability of transitioning to the $cart$ and buy states compared to the tasks $(male \times cosmetics)$ and $(female \times electronicproducts)$. Additionally, the higher the discount the shopping platform offers, the more likely the guests are to transition to the $cart$ and buy states. The detailed probability transition can be found in Appendix E.

The price of $electronicproducts$ is 1000 and the price of $cosmetics$ is 500. The reward is the price that the user pays and the cost is the discount the platform offers when purchase occurs, which means reward is $(1-discount)*price$ and cost is $discount * price$. The other two states only cause a little cost and do not bring a reward.

Figure 1 shows that the Q-UCB method performs the worst, as it only optimizes the ratio of reward to cost at every step, without actually learning the optimal policy for each task. In contrast, the CARVI and CAAC methods perform better than Q-UCB, because they do learn the optimal policy for each task individually. However, they still perform worse than RM^{2}VI, as they do not take the distribution of task types into account. The results demonstrate RM^{2}VI can converge to a global policy compared to the greedy decision-making approaches.

6.2 Cloud Computing Scenario

We simulate a cloud computing scenario where a server receives ML training tasks sequentially, and it has two chances $H = 2$ to configure the task to train (i.e., choose the number of training episodes). When the first episode choice ends, the platform extracts its accuracy and configures for the next step. The total number of training tasks is $K = 15000$. We assume that there are 3 types of ML training tasks, which are the insect recognition task, the atomic force microscopy images analysis task and the speech-emotion recognition task. The distribution of the three task types $\{insect, AFM, speech\}$ is drawn from $(0.8, 0.05, 0.15)$.

For each task, according to complexity, we set a minimal selectable number of epochs, which is $(1, 50, 25)$. The three actions/epoch numbers that the server can choose: $(1, 2, 3)$ times of the minimal selectable number of epochs. The states of tasks are $(0, 1, 2, 3, 4, 5, 6)$ times of the minimal selectable number of epochs, in which 0 is the initial state, since the server can only choose two times. The transition kernel $\mathbb{P}_h(s'|m, s, a) = 1$ when $s + a = s'$ and stays 0 otherwise.

The accuracy of the model is regarded as a reward and the total number of episodes is the cost. We use the data in [Martineau *et al.*, 2018; Zhang *et al.*, 2024; Abdul Qayyum *et al.*, 2019] to figure out reward, which is accuracy, instead of really running them on a server.

Figure 2 shows that though RM^{2}VI still outperforms the other baselines, we find that Q-UCB performs as well as CARVI, which is different from the result in the former experiment. This is because the setting of reward and cost functions causes the optimal policy of the individual task and the policy learned by Q-UCB to be the same by accident. The poor performance of CAAC is because CAAC learns more slowly than the other baselines, which is consistent with the result in the former experiment.

7 Conclusion

In this paper, we studied online scheduling for multi-stage tasks and proposed a novel RM^{2}VI algorithm. RM^{2}VI integrates optimistic value learning methods and stochastic approximation techniques to balance the trade-off between rewards acquisition and costs consumption in the face of uncertainty. Through theoretical analysis, we demonstrated that RM^{2}VI achieves a sub-linear regret bound over the total number of tasks K Furthermore, we conducted simulation experiments that showed RM^{2}VI outperforms state-of-the-art baseline methods.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62302305, Shanghai Sailing Program under Grant 22YF1428500 and the Core Facility Platform of Computer Science and Communication, SIST, ShanghaiTech University.

References

[Abdul Qayyum *et al.*, 2019] Alif Bin Abdul Qayyum, Asiful Arefeen, and Celia Shahnaz. Convolutional neural network (cnn) based speech-emotion recognition. In *2019 IEEE International Conference on Signal Processing, Information, Communication & Systems (SPICSCON)*, pages 122–125, 2019.

[Abounadi *et al.*, 2001] Jinane Abounadi, Dimitrib Bertsekas, and VS Borkar. Learning algorithms for Markov decision processes with average cost. *SIAM Journal on Control and Optimization*, 40(3):681–698, 2001.

[Agarwal and Jain, 2014] Dr Amit Agarwal and Saloni Jain. Efficient optimal algorithm of task scheduling in cloud computing environment. *International Journal of Computer Trends and Technology*, 9(7):344–349, 2014.

[Agrawal and Devanur, 2014] Shipra Agrawal and Nikhil R. Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, page 989–1006, New York, NY, USA, 2014. Association for Computing Machinery.

[Agrawal and Devanur, 2016] Shipra Agrawal and Nikhil Devanur. Linear contextual bandits with knapsacks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016.

[Alabbadi and Abulkhair, 2021] Afra A Alabbadi and Maysoon F Abulkhair. Multi-objective task scheduling optimization in spatial crowdsourcing. *Algorithms*, 14(3):77–97, 2021.

[Altman, 1999] Eitan Altman. *Constrained Markov Decision Processes*. CRC Press, 1999.

[Arunarani *et al.*, 2019] AR Arunarani, Dhanabalachandran Manjula, and Vijayan Sugumaran. Task scheduling techniques in cloud computing: A literature survey. *Future Generation Computer Systems*, 91:407–415, 2019.

[Auer *et al.*, 2002] P. Auer, Paul Fischer, and N. Cesa-Bianchi. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(3):235–256, 2002.

[Azar *et al.*, 2017] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 263–272, Sydney, NSW, Australia, 2017. JMLR.org.

[Badanidiyuru *et al.*, 2014] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful contextual bandits. In *Proceedings of The 27th Conference on Learning Theory*, pages 1109–1134, Barcelona, Spain, June 2014. PMLR.

[Badanidiyuru *et al.*, 2018] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. *J. ACM*, 65(3), March 2018.

[Bhatnagar, 2010] Shalabh Bhatnagar. An actor–critic algorithm with function approximation for discounted cost constrained markov decision processes. *Systems & Control Letters*, 59(12):760–766, 2010.

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

[Cayci *et al.*, 2020] Semih Cayci, Atilla Eryilmaz, and R Srikant. Budget-constrained bandits over general cost and reward distributions. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 4388–4398. PMLR, August 2020.

[Cesa-Bianchi *et al.*, 2021] Nicolò Cesa-Bianchi, Tom Cesari, Yishay Mansour, and Vianney Perchet. Roi maximization in stochastic online decision-making. In *Advances in Neural Information Processing Systems*, pages 9152–9166. Curran Associates, Inc., 2021.

[Chen *et al.*, 2020] Xuan Chen, Long Cheng, Cong Liu, Qingzhi Liu, Jinwei Liu, Ying Mao, and John Murphy. A woa-based optimization approach for task scheduling in cloud computing systems. *IEEE Systems Journal*, 14(3):3117–3128, 2020.

[Cho *et al.*, 2015] Keng-Mao Cho, Pang-Wei Tsai, Chun-Wei Tsai, and Chu-Sing Yang. A hybrid meta-heuristic algorithm for vm scheduling with load balancing in cloud computing. *Neural Computing and Applications*, 26:1297–1309, 2015.

[Deane, 2012] Jason Deane. Hybrid genetic algorithm and augmented neural network application for solving the online advertisement scheduling problem with contextual targeting. *Expert Systems with Applications*, 39(5):5168–5177, 2012.

[Deng *et al.*, 2015] Dingxiong Deng, Cyrus Shahabi, and Linhong Zhu. Task matching and scheduling for multiple workers in spatial crowdsourcing. In *Proceedings of ACM SIGSPATIAL*, 2015.

[Ding *et al.*, 2013] Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-armed bandit with budget constraint and variable costs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 27, pages 232–238, 2013.

[Guo and Liu, 2024] Hengquan Guo and Xin Liu. Stochastic constrained contextual bandits via lyapunov optimization based estimation to decision framework. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 2204–2231, 2024.

[Guo *et al.*, 2012] Lizheng Guo, Shuguang Zhao, Shigen Shen, and Changyuan Jiang. Task scheduling optimization in cloud computing based on heuristic algorithm. *Journal of Networks*, 7(3):547–553, 2012.

[Han *et al.*, 2023] Yuxuan Han, Jialin Zeng, Yang Wang, Yang Xiang, and Jiheng Zhang. Optimal contextual bandits with knapsacks under realizability via regression oracles. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, pages 5011–5035. PMLR, April 2023.

[Held and Karp, 1962] Michael Held and Richard M Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied mathematics*, 10(1):196–210, 1962.

[Heyden *et al.*, 2024] Marco Heyden, Vadim Arzamasov, Edouard Fouché, and Klemens Böhm. Budgeted multi-armed bandits with asymmetric confidence intervals. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, page 1073–1084, New York, NY, USA, 2024. Association for Computing Machinery.

[Jin *et al.*, 2018] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I. Jordan. Is Q-learning provably efficient? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 4868–4878, Red Hook, NY, USA, 2018. Curran Associates Inc.

[Kanuri *et al.*, 2018] Vamsi K Kanuri, Yixing Chen, and Shrihari Sridhar. Scheduling content on social media: Theory, evidence, and application. *Journal of Marketing*, 82(6):89–108, 2018.

[Khazankin *et al.*, 2011] Roman Khazankin, Harald Psaier, Daniel Schall, and Schahram Dustdar. Qos-based task scheduling in crowdsourcing environments. In *Proceedings of ICSOC*, 2011.

[Martineau *et al.*, 2018] Maxime Martineau, Romain Raveaux, Clément Chatelain, Donatello Conte, and Gilles Venturini. Effective training of convolutional neural networks for insect image recognition. In *Advanced Concepts for Intelligent Vision Systems: 19th International Conference, ACIVS 2018, Poitiers, France, September 24–27, 2018, Proceedings 19*, pages 426–437. Springer, 2018.

[Neely, 2010] Michael J Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[Neely, 2021] Michael J Neely. Fast learning for renewal optimization in online task scheduling. *Journal of Machine Learning Research*, 22(1):12785–12828, 2021.

[Neely, 2024] Michael J Neely. Adaptive optimization for stochastic renewal systems. *arXiv preprint arXiv:2401.07170*, 2024.

[REN and KROGH, 2005] ZHIYUAN REN and Bruce H KROGH. Markov decision processes with fractional costs. *IEEE transactions on automatic control*, 50(5):646–650, 2005.

[Slivkins *et al.*, 2023] Aleksandrs Slivkins, Karthik Abinav Sankararaman, and Dylan J Foster. Contextual bandits with packing and covering constraints: A modular lagrangian approach via regression. In *Proceedings of Thirty Sixth Conference on Learning Theory*, pages 4633–4656. PMLR, 12–15 Jul 2023.

[Srikant and Ying, 2014] Rayadurgam Srikant and Lei Ying. *Communication Networks: An Optimization, Control, and Stochastic Networks Perspective*. Cambridge University Press, 2014.

[Suttle *et al.*, 2021] Wesley Suttle, Kaiqing Zhang, Zhuoran Yang, Ji Liu, and David Kraemer. Reinforcement Learning for Cost-Aware Markov Decision Processes. In *Proceedings of the 38th International Conference on Machine Learning*, pages 9989–9999. PMLR, 2021.

[Tanaka, 2017] Teruo Tanaka. A partially observable discrete time markov decision process with a fractional discounted reward. *Journal of Information and Optimization Sciences*, 38(1):21–37, 2017.

[Tran-Thanh *et al.*, 2012] Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas Jennings. Knapsack based optimal policies for budget–limited multi–armed bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 1134–1140, 2012.

[Wachi *et al.*, 2024] Akifumi Wachi, Xun Shen, and Yanan Sui. A survey of constraint formulations in safe reinforcement learning. *arXiv preprint arXiv:2402.02025*, 2024.

[Watanabe *et al.*, 2018] Ryo Watanabe, Junpei Komiyama, Atsuyoshi Nakamura, and Mineichi Kudo. Ucb-sc: A fast variant of kl-ucb-sc for budgeted multi-armed bandit problem. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 101(3):662–667, 2018.

[Wei *et al.*, 2022] Honghao Wei, Xin Liu, and Lei Ying. Triple-q: A model-free algorithm for constrained reinforcement learning with sublinear regret and zero constraint violation. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 3274–3307. PMLR, 2022.

[Xia *et al.*, 2017] Yingce Xia, Tao Qin, Wenkui Ding, Haifang Li, Xudong Zhang, Nenghai Yu, and Tie-Yan Liu. Finite budget analysis of multi-armed bandit problems. *Neurocomputing*, 258:13–29, 2017.

[Xu *et al.*, 2024] Yongxin Xu, Shangshang Wang, Hengquan Guo, Xin Liu, and Ziyu Shao. Learning to Schedule Online Tasks with Bandit Feedback. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, page 1975–1983, Richland, SC, 2024. International Foundation for Autonomous Agents and Multiagent Systems.

[Zhang *et al.*, 2024] Juntao Zhang, Juan Ren, and Shuiqing Hu. Afm imaging defect detection and classification using deep learning. *IEEE Access*, 12:132027–132037, 2024.

[Zhao *et al.*, 2022] Zihui Zhao, Xiaoyu Shi, and Mingsheng Shang. Performance and cost-aware task scheduling via deep reinforcement learning in cloud environment. In *Service-Oriented Computing*, pages 600–615, Cham, 2022. Springer Nature Switzerland.