2019 57th Annual Allerton Conference on Communication, Control, and Computing
(Allerton)
Allerton Park and Retreat Center
Monticello, IL, USA, September 24-27, 2019

Reinforcement Learning for Optimal Control of Queueing Systems

Bai Liu*, Qiaomin Xie†, and Eytan Modiano*
*Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA
†School of Operations Research and Information Engineering, Cornell University, Ithaca, NY

***Abstract*—With the rapid advance of information technology, network systems have become increasingly complex and hence the underlying system dynamics are typically unknown or difficult to characterize. Finding a good network control policy is of significant importance to achieving desirable network performance (e.g., high throughput or low average job delay). Online/sequential learning algorithms are well-suited to learning the optimal control policy from observed data for systems without the information of underlying dynamics. In this work, we consider using model-based reinforcement learning (RL) to learn the optimal control policy of queueing networks so that the average job delay (or equivalently the average queue backlog) is minimized. Existing RL techniques, however, cannot handle the unbounded state spaces of the network control problem. To overcome this difficulty, we propose a new algorithm, called Piecewise Decaying ϵ-Greedy Reinforcement Learning (PDGRL), which applies model-based RL methods over a finite subset of the state space. We establish that the average queue backlog under PDGRL with an appropriately constructed subset can be arbitrarily close to the optimal result. We evaluate PDGRL in dynamic server allocation and routing problems. Simulations show that PDGRL minimizes the average queue backlog effectively.**

I. Introduction

The rapid growth of information technology has resulted in increasingly complex network systems and poses challenges in obtaining explicit knowledge of system dynamics. For instance, due to security or economic concerns, a number of network systems are built as overlay networks, e.g. caching overlays, routing overlays and security overlays [1]. In these cases, only the overlay part is fully controllable by the network administrator, while the underlay part remains uncontrollable and/or unobservable. The "black box" components make network control policy design challenging.

In addition to the challenges brought by unknown system dynamics, many of the current network control algorithms (e.g. MaxWeight [2] and Drift-plus-Penalty [3]) aim at stabilizing the system, while general optimization methods for long-term performances metrics (e.g. queueing backlog and delay) are relatively rare.

To overcome the above challenges, it is desirable to apply inference and learning schemes. A natural approach is reinforcement learning, which learns and reinforces a good decision policy by repeatedly interacting with the environment. Reinforcement learning methods provide a framework that enables the design of learning policies for general networks. There have been two main lines of work on reinforcement learning methods: model-free reinforcement learning (e.g. Q-learning [4], policy gradient [5]) and model-based reinforcement learning (e.g., UCRL [6], PSRL [7]). In this work, we focus on the model-based framework.

Related Work. In network control, a widely used algorithm is the MaxWeight [2], which can be applied to general multi-server networks with an arbitrary topology. MaxWeight algorithm has been shown to be throughput-optimal (i.e. can stabilize the system whenever the system is stabilizable). Moreover, the MaxWeight algorithm does not require explicit arrival information but only the current queue backlog and the knowledge of service rates, which makes it suitable for complex systems. The work in [3] extends MaxWeight to utility optimization using the Drift-plus-Penalty algorithm.

To design control policies for networks with unknown dynamics, an intuitive approach is as follows: estimating the parameters of the underlay components first, and then applying classic network control techniques based on the estimated parameters. A variety of learning methods have been applied. A popular method is probing, i.e., sending probe packets at certain time intervals and collect tunnel information. For instance, the work in [8], [9] gathers direct and indirect path information by collecting ping data. In [10], simulation results illustrate that the probing approach could achieve optimal throughput. Recently reinforcement learning has emerged as a popular and powerful approach for complex systems. In [11], the authors apply the Q-learning algorithm in overlay non-cooperative multi-agent wireless sensor networks (WSNs) to achieve optimal mutual response between two agents. The work of [12] combines reinforcement learning with neural networks and improves scalability compared with probe-based inference methods.

Reinforcement learning is well-suited to learning the optimal control for a system with unknown parameters. We consider model-based reinforcement learning methods, which tend to be more tractable in analysis. Conventional model-based reinforcement learning methods like UCRL [6] and PSRL [7] only work for finite-state-space systems, yet queueing systems are usually modeled to have unbounded buffer sizes. The work in [13] assumes that the system has an admission control scheme to keep queue backlogs finite so that we can turn the system into a finite state MDP model and apply UCRL. However, the practical queueing systems may not have admission control schemes and this approach

978-1-7281-3151-1/19/$31.00 ©2019 IEEE

might not apply directly. In [14], the authors modify PSRL algorithm to deal with MDPs with large state space, yet the algorithm requires the MDP to have a finite bias span, which is unrealistic for the MDP problems with unbounded cost functions. In both [15] and [16], deep reinforcement learning methods are applied to spectrum access problems and outperform traditional baseline algorithm, but lack rigorous theoretical performance analysis.

To sum up, there have been some work on network control that aims at stabilizing the systems, yet the performance metric of interest here is queue backlog and delay. Among the existing work on queue backlog optimization, most propose ad-hoc solutions for some specific scenarios. Model-based reinforcement learning is a potential approach for the optimal control of the general queueing system, yet the classical methods (UCRL and PSRL) can only solve bounded-state-space MDPs.

Our contributions. We apply model-based reinforcement learning to queueing networks with unbounded state spaces and unknown dynamics. Our approach leverages the fact that for a vast class of stable queueing systems, the probability of the queue backlog being large is relatively small. This observation motivates us to focus on learning control policies over a finite subset of states where the system visits with high probability. Our main contributions are summarized as follows.

- We propose a model-based RL algorithm that can deal with unbounded state spaces. In particular, we introduce an auxiliary system with the state space bounded by a threshold of U. Our approach employs a piecewise policy: for states below the threshold, a model-based RL algorithm is used; for all other states, a simple baseline algorithm is applied.
- We establish that the average queue backlog under the proposed algorithm can be made arbitrarily close to the optimal result with a large threshold of U. In particular, by utilizing the Lyapunov analysis technique, we characterize how the gap to the optimal performance decays with the threshold U.
- Simulation results on dynamic server allocation and routing problems corroborate the validity of our theoretical guarantees. In particular, the proposed algorithm effectively achieves a small average queue backlog, with the gap to the optimal policy diminishing as the threshold U increases.

II. Model

In this paper, we target at optimizing the average queue backlog of a general discrete-time queueing network system that can be formulated by Markov decision process (MDP) framework. The system consists of a set of nodes and links. Each node maintains one or more queues for the undelivered packets, and each queue has an unbounded buffer. The system may have an arbitrary topology and the underlying dynamics can be partially or fully unknown.

A. Real System

The system can be modeled as a countable-state MDP M as follows.

- **State space $\mathcal{S}$:**
  For ease of exposition, we focus on queueing networks where the system state can be represented by the lengths of all queues. We denote the number of queues as D. The system state is given by a D-dimensional queue backlog vectors $\boldsymbol{Q}$. The system space is denoted as $\mathcal{S} = \underbrace{\mathbb{N} \times \cdots \times \mathbb{N}}_{D \text{ times}}$.
- **Action space $\mathcal{A}$:**
  The exact form of action space depends on the problem setting. For instance, in the server allocation problem where D parallel queues compete for the service of a single server [17], the action is the queue served by the server at each time slot and the action space is naturally the set of queue indexes. The action space is represented by $\mathcal{A}$ and we assume that $|\mathcal{A}| < \infty$.
- **State-transition probability p:**
  The system dynamics satisfy the Markov property. Formally, the probability of transitioning into a particular state $\boldsymbol{Q}'$ only depends on the current state $\boldsymbol{Q}$ and the action a, denoted as $p\left(\boldsymbol{Q}' \mid \boldsymbol{Q}, a\right)$. We assume that the number of newly arrived and served packets during each time slot are both bounded. That is, there exists a constant W such that for every $\boldsymbol{Q}(t) \in \mathcal{S}$,
  $$\|\boldsymbol{Q}(t+1) - \boldsymbol{Q}(t)\|_\infty \leqslant W.$$
  We define the set of states within the one-step reachable region of $\boldsymbol{Q}$ as
  $$\mathcal{R}\left(\boldsymbol{Q}, a\right) \triangleq \left\{\boldsymbol{Q}' \in \mathcal{S} : p\left(\boldsymbol{Q}' \mid \boldsymbol{Q}, a\right) > 0\right\}.$$
- **Cost function $c(\boldsymbol{Q})$:**
  Since we aim at minimizing the average queue backlog, we define the cost function as $c(\boldsymbol{Q}) = \sum_i Q_i$. We denote the optimal average queue backlog as ρ^*, and the corresponding optimal policy as π^*.

We define that $Q_{max} \triangleq \max_i Q_i$. To deal with the unbounded state space, we consider a natural assumption on the existence of a policy that stabilizes the system.

Assumption 1. *There exists a* known *policy π_0 satisfying the following condition: there exist a Lyapunov function $\Phi_0 : \mathcal{S} \to \mathbb{R}_+$ and constants $a, \alpha, \epsilon_0, B_0 > 0$ such that $\Phi(\boldsymbol{Q}) \leqslant aQ_{max}^\alpha$ for every $\boldsymbol{Q}(t) \in \mathcal{S}$. If $Q_{max}(t) \geqslant B_0$, we have*

$$\mathbb{E}_{\pi_0}\left[\Phi_0\left(\boldsymbol{Q}(t+1)\right) - \Phi_0\left(\boldsymbol{Q}(t)\right) \mid \boldsymbol{Q}(t)\right] \leqslant -\epsilon_0,$$

where $\mathbb{E}_{\pi_0}$ denotes expectations under policy π_0.

A broad class of queueing systems have been proven to have a π_0 that satisfies Assumption 1. For instance, stabilizing policies are proposed for dynamic server allocation problem [17]–[19], multiclass routing network [20]–[24] and inventory control [25], [26], with linear or quadratic form of Lyapunov functions. This assumption allows us to upper bound the tail probability of queue backlog.

B. Auxiliary System

The key challenge of applying model-based reinforcement learning to countable-state MDP is that classical model-based reinforcement learning techniques usually operate in episodic manners: for each episode, the system dynamics are estimated and an approximated optimal policy is obtained based on the learned dynamics. However, there is no effective solution for general countable-state MDPs with optimal average cost (in contrast to discounted cost) criteria.

Here we introduce an auxiliary system $\tilde{M}$ with bounded state space. The auxiliary system has threshold U: the system has exact dynamics as the real one, with the only difference that each queue has buffer size U. In the bounded system, for each queue, when the queue backlog reaches U, new packets to the queue will get dropped. Mathematically, the state space of $\tilde{M}$ can be defined as $\tilde{\mathcal{S}} \triangleq \{\boldsymbol{Q} \in \mathcal{S} : Q_{max} \leqslant U\}$. $\tilde{M}$ shares the same action space $\mathcal{A}$ and cost function $c(\boldsymbol{Q})$ as M. We use $\tilde{p}$ to represent the state-transition function in $\tilde{M}$. We denote the optimal average queue backlog in $\tilde{M}$ by $\tilde{\rho}^*$, and the corresponding optimal policy by $\tilde{\pi}^*$.

We make an assumption on $\tilde{\pi}^*$ for technical reasons. For clarity, we use $\tilde{\mathbb{E}}[\cdot]$ for the expectations in $\tilde{M}$ (to distinguish from $\mathbb{E}[\cdot]$ for M).

Assumption 2. *There exist a Lyapunov function $\tilde{\Phi}^*(\cdot) : \tilde{\mathcal{S}} \to \mathbb{R}_+$, and constants $0 \leqslant \beta < 1$, $b_1, b_2, b_3, \tilde{B}^*, \tilde{\epsilon}^* > 0$, such that for any $U > 0$, the following properties hold:*

- *For each $\boldsymbol{Q} \in \tilde{\mathcal{S}}$, $b_1 Q_{max}^{1+\beta} \leqslant \tilde{\Phi}^*(\boldsymbol{Q}) \leqslant b_2 Q_{max}^{1+\beta}$;*
- *For each $\boldsymbol{Q}(t) \in \tilde{\mathcal{S}}$, $\left|\tilde{\Phi}^*\left(\boldsymbol{Q}(t+1)\right) - \tilde{\Phi}^*\left(\boldsymbol{Q}(t)\right)\right| \leqslant b_3 U^{\beta}$;*
- *If Q_i increases (while other entries remain the same), $\tilde{\Phi}^*(\boldsymbol{Q})$ will not decrease;*
- *When $\tilde{\Phi}^*(\boldsymbol{Q}) \geqslant \tilde{B}^*$, we have*

$$\tilde{\mathbb{E}}_{\tilde{\pi}^*}\left[\tilde{\Phi}^*\left(\boldsymbol{Q}(t+1)\right) - \tilde{\Phi}^*\left(\boldsymbol{Q}(t)\right) \mid \boldsymbol{Q}(t) = \boldsymbol{Q}\right] \leqslant -\tilde{\epsilon}^*.$$

For network systems that are stabilized under Max-Weight-α policy [27] or Back-Pressure-α policy [28], the Lyapunov functions have been proven to have the form of weighted sum of $Q_i^{1+\alpha}$. When $0 < \alpha < 1$ and $\tilde{\pi}^*$ is Max-Weight-α or Back-Pressure-α method, Assumption 2 holds. Other examples include the multi-class Markovian queueing network in [24] and the server allocation problem described in Section V-A. Both problems have linear Lyapunov functions under backlog optimal policies.

C. Our Approach

For simplicity, we partition $\mathcal{S}$ as follows:

$$\begin{cases} \mathcal{S}^{in} \triangleq \left\{\boldsymbol{Q} \in \tilde{\mathcal{S}} : \tilde{\Phi}^*(\boldsymbol{Q}) \leqslant b_1 (U - W)^{1+\beta}\right\} \\ \mathcal{S}^{out} \triangleq \mathcal{S} \setminus \mathcal{S}^{in} \end{cases}.$$

Our reinforcement learning method can be decomposed into two stages: exploration stage and exploitation stage. For each episode, with some relatively small probability, we apply randomized policy to learn the dynamics of $\tilde{M}$ and estimate $\tilde{\pi}^*$. Meanwhile, in our exploitation scheme, we apply a piecewise policy: the policy obtained from exploration stages is used for states in $\mathcal{S}^{in}$; while for $\mathcal{S}^{out}$, a stabilizing π_0 (defined in Assumption 1) is applied. See Figure 1 for illustration. A detailed description of the algorithm is provided in Section III.

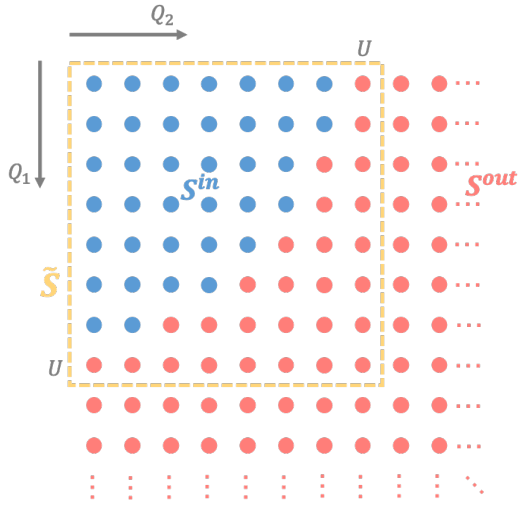


Fig. 1: Schemetic illustration of our approach (when $D = 2$).

For performance analysis, we decouple the process into two stages: before $\tilde{\pi}^*$ is learned and after $\tilde{\pi}^*$ is learned. For the first stage, our model-based reinforcement learning approach applies ϵ-greedy exploration. We establish that the proposed algorithm gradually obtains $\tilde{\pi}^*$ with arbitrarily high probability (cf. Theorem 1). For the second stage, by applying drift analysis on Markov chain, we show that when $\tilde{\pi}^*$ is applied for states in $\mathcal{S}^{in}$ and π_0 for states in $\mathcal{S}^{out}$, the probability of queue backlog exceeding into $\mathcal{S}^{out}$ decays exponentially with U. In addition, every time when queue backlog leaves $\mathcal{S}^{in}$, the expected queue backlog can be bounded as polynomial terms of U. Together, we show that the gap between our result and ρ^* is upper bounded by $\mathcal{O}\left(poly(U)/\exp\left(poly(U)\right)\right)$, which diminishes as U increases (cf. Theorem 2).

D. Other Assumptions

Due to mathematical requirements, we impose some restrictions on the communication properties on $\tilde{M}$. We denote $T_{\boldsymbol{Q}\to\boldsymbol{Q}'}$ as the first hitting time from $\boldsymbol{Q}$ to $\boldsymbol{Q}'$ and propose the following assumption.

Assumption 3. *There exist constants $c, \gamma > 0$, such that for any $U > 0$ and every $\boldsymbol{Q}, \boldsymbol{Q}' \in \tilde{\mathcal{S}}$,*

$$\min_{\tilde{\pi}} \tilde{\mathbb{E}}_{\tilde{\pi}}\left[T_{\boldsymbol{Q}\to\boldsymbol{Q}'}\right] \leqslant c\|\boldsymbol{Q}' - \boldsymbol{Q}\|_1^{\gamma},$$

where $\tilde{\pi}$ is a policy that can be applied to $\tilde{S}$.

Directly verifying Assumption 3 might be computationally difficult. Theorem 11.3.11 from [29] offers a drift analysis approach for justifying Assumption 3.

Lemma 1 (Theorem 11.3.11 in [29]). *For a ψ-irreducible Markov chain, if there exists a Lyapunov function $\Phi(\cdot)$ and a petite set C such that for every $\boldsymbol{Q}(t) \in \mathcal{S}$,*

$$\mathbb{E}_\pi \left[\Phi\left(\boldsymbol{Q}(t+1)\right) - \Phi\left(\boldsymbol{Q}(t)\right) \mid \boldsymbol{Q}(t) \right] \leqslant -1 + b\mathbb{1}_{\boldsymbol{Q}(t)\in C},$$

then for every $\boldsymbol{Q}, \boldsymbol{Q}' \in \mathcal{S}$, there exists $c\left(\boldsymbol{Q}'\right) < \infty$ such that

$$\mathbb{E}\left[T_{\boldsymbol{Q}\to\boldsymbol{Q}'}\right] \leqslant \Phi\left(\boldsymbol{Q}\right) + c\left(\boldsymbol{Q}'\right).$$

A possible method is to analyze the Markov chain under π_0 and (re-scaled) $\Phi_0(\cdot)$ in Assumption 1. In this case, if we select a suitable measure for the ψ-irreducible Markov chain, $c\left(\boldsymbol{Q}\right)$ may have a polynomial upper bound regarding Q_{max} and Assumption 3 holds.

As the learning process proceeds, the estimation for $\tilde{M}$ becomes increasingly accurate. However, it is unrealistic for us to obtain the exact $\tilde{M}$. Therefore, we make the assumption that if we estimate the state-transition function accurate enough (i.e. within a certain error bound), the solution to the estimated $\tilde{M}$ is the same as $\tilde{\pi}^*$. The assumption is stated as follows.

Assumption 4. *There exists a $\Delta p > 0$, such that for any MDP $\tilde{M}'$ with the same state space, action space and cost function as $\tilde{M}$, if for each $\boldsymbol{Q} \in \tilde{\mathcal{S}}$ and $a \in \mathcal{A}$, we have*

$$\left\| \tilde{p}\left(\cdot \mid \boldsymbol{Q}, a\right) - p'\left(\cdot \mid \boldsymbol{Q}, a\right) \right\|_1 \leqslant \Delta p,$$

then the optimal policy of M' is the same as the otimal policy $\tilde{\pi}^*$ for $\tilde{M}$.

Notice that in most queueing networks, when system dynamics (e.g. exogenous arrival rates, service rates, channel capacities) vary slightly, the optimal policy remains the same. Assumption 4 is reasonable for queueing systems.

III. Algorithm

We propose an algorithm called Piecewise Decaying ϵ-Greedy Reinforcement Learning (PDGRL). PDGRL operates in an episodic manner: at the beginning of episode k, we uniformly draw a real number $\xi \in [0, 1]$ to decide whether to explore or exploit during this episode. The length of the episodes is not fixed but depends on the observations.

- If $\xi \leqslant \epsilon_k \triangleq l/\sqrt{k}$ (where $0 < l \leqslant 1$), we perform exploration during this episode. For states in $\tilde{\mathcal{S}}$, we apply a random policy π_{rand}, which selects an action in $\mathcal{A}$ uniformly. For states in $\mathcal{S} \setminus \tilde{\mathcal{S}}$, we apply π_0.
- If $\xi > \epsilon_k$, we enter the exploitation stage. We first calculate sample means to estimate the state-transition function $\tilde{p}$ of $\tilde{M}$. We then apply value iteration on the estimated system $\tilde{M}_k$ and obtain an estimated optimal policy $\tilde{\pi}_k^*$. For the rest of the episode, we apply $\tilde{\pi}_k^*$ for states in $\mathcal{S}^{in}$ and π_0 otherwise.
- When the number of visits to states in $\mathcal{S}^{in}$ exceeds $L_k = L \cdot \sqrt{k}$ (where $L > 0$), PDGRL enters episode $k+1$ and repeat the process above.

To further illustrate the algorithm, we define a mapping $TR(\cdot) : \mathcal{S} \to \tilde{\mathcal{S}}$ that describes the packet dropping scheme in the bounded system:

$$\tilde{\boldsymbol{Q}} = TR\left(\boldsymbol{Q}\right) \triangleq \left\{\min\left\{U, Q_i\right\}\right\}_{i=1}^{D}.$$

The details are presented in Algorithm 1.

Algorithm 1 The PDGRL algorithm

Input: $\mathcal{A}$, U, $l > 0$, $L > 0$
Initialization: $t \leftarrow 1$, $N(\cdot,\cdot) \leftarrow 0$, $\tilde{P}(\cdot,\cdot) \leftarrow 0$
for episodes $k \leftarrow 1, 2, \cdots, K$ **do**
  Set $L_k \leftarrow L \cdot \sqrt{k}$, $\epsilon_k \leftarrow l/\sqrt{k}$ and uniformly draw $\xi \in [0, 1]$.
  if $\xi \leqslant \epsilon_k$ **then**
    Set

$$\pi_k\left(\boldsymbol{Q}\right) = \begin{cases} \pi_{rand}\left(\boldsymbol{Q}\right), & \text{for } \boldsymbol{Q} \in \tilde{\mathcal{S}} \\ \pi_0\left(\boldsymbol{Q}\right), & \text{for } \boldsymbol{Q} \in \mathcal{S} \setminus \tilde{\mathcal{S}} \end{cases}.$$

  else
    For each $\boldsymbol{Q}, \boldsymbol{Q}' \in \tilde{\mathcal{S}}$ and $a \in \mathcal{A}$, estimate that $\tilde{p}\left(\boldsymbol{Q}' \mid \boldsymbol{Q}, a\right) = \tilde{P}\left(\boldsymbol{Q}, a, \boldsymbol{Q}'\right)/N\left(\boldsymbol{Q}, a\right)$ for $N\left(\boldsymbol{Q}, a\right) > 0$ and $\tilde{p}\left(\boldsymbol{Q}' \mid \boldsymbol{Q}, a\right) = 1/\left|\mathcal{R}\left(\boldsymbol{Q}, a\right)\right|$ otherwise.
    Solve the estimated MDP $\tilde{M}_k$ and obtain the estimated optimal policy $\tilde{\pi}_k^*$.
    Set

$$\pi_k\left(\boldsymbol{Q}\right) = \begin{cases} \tilde{\pi}_k^*\left(\boldsymbol{Q}\right), & \text{for } \boldsymbol{Q} \in \mathcal{S}^{in} \\ \pi_0\left(\boldsymbol{Q}\right), & \text{for } \boldsymbol{Q} \in \mathcal{S}^{out} \end{cases}.$$

  end if
  while visits to states in $\mathcal{S}^{in}$ is smaller that L_k **do**
    Take $a_t = \pi_k\left(\boldsymbol{Q}(t)\right)$.
    Implement a_t to the real system and observe the next state $\boldsymbol{Q}(t+1)$.
    if $\boldsymbol{Q}(t) \in \mathcal{S}^{in}$ **then**
      Increase $N\left(\boldsymbol{Q}(t), a_t\right)$ by 1.
      Increase $\tilde{P}\left(\boldsymbol{Q}(t), a_t, TR\left(\boldsymbol{Q}(t+1)\right)\right)$ by 1.
    end if
    $t \leftarrow t + 1$.
  end while
end for
Output: estimated optimal policy $\tilde{\pi}_K^*$

IV. Performance Analysis

We analyze the performance of our algorithm from both exploration and exploitation perspectives. We first prove that PDGRL can learn $\tilde{\pi}^*$ within finite episodes with high probability, which implies that PDGRL explores different states sufficiently to obtain an accurate estimation of $\tilde{M}$ (cf. Theorem 1). We then show that PDGRL exploits the estimated optimal policy and achieves a performance close to the true optimal result of ρ^* (cf. Theorem 2). Due to the space limit, for all the results presented in this paper, we only provide proof outlines and omit proof details.

In this paper, we focus on MDPs such that all states are reachable from each other under the following policies: (a) $\pi_{rand}+\pi_0$: applying π_{rand} to $\tilde{\mathcal{S}}$ and π_0 to $\mathcal{S}\setminus\tilde{\mathcal{S}}$; (b) $\tilde{\pi}^*+\pi_0$: applying $\tilde{\pi}^*$ to $\mathcal{S}^{in}$ and π_0 to $\mathcal{S}^{out}$; and (c) π^* : applying (a truncated version of) π^* to $\tilde{\mathcal{S}}$ in $\tilde{M}$. That is, the corresponding Markov chains under the above policies are irreducible.

A. Convergence to the Optimal Policy

The following theorem states that, with arbitrarily high probability, PDGRL learns $\tilde{\pi}^*$ within a finite number of episodes.

Theorem 1. *Suppose Assumption 4 holds. For each $\delta \in (0,1)$, there exists $k^* < \infty$ such that* PDGRL *learns $\tilde{\pi}^*$ (i.e. $\tilde{\pi}^*_{k^*} = \tilde{\pi}^*$) within k^* episodes with probability at least $1-\delta$.*

Proof. We first calculate the required sample size of $(\boldsymbol{Q}, a)$ pairs to obtain $\tilde{\pi}^*$ with high probability under Assumption 4. We then show that the number of samples obtained by following the policy π_{rand} for states in $\tilde{\mathcal{S}}$ is unbounded as $K \to \infty$. □

Theorem 1 indicates that PDGRL explores (i.e. samples) state-transition functions of each state-action pair $(\boldsymbol{Q}, a)$ in $\tilde{M}$ sufficiently.

B. Average Queue Backlog

Section IV-A illustrates the sufficient exploration aspect of PDGRL. In reinforcement learning, the trade-off between exploration and exploitation is of significant importance to the algorithm performance. In this section, we show that PDGRL also exploits the learned policy such that the expected average queue backlog is bounded and can get arbitrarily close to the optimal performance of ρ^* as we increase U.

We denote the time step at the end of episode k by t_k and the length of episode k by L'_k (i.e. $L'_k = t_k - t_{k-1}$ with $t_0 \triangleq 0$). We use π^{in}_k to represent the policy applied to $\mathcal{S}^{in}$ during episode k and $p^{\tilde{\pi}+\pi_0}(\cdot)$ to denote the stationary distribution of states when applying $\tilde{\pi}$ to states in $\mathcal{S}^{in}$ and π_0 to states in $\mathcal{S}^{out}$.

By Theorem 1, PDGRL learns $\tilde{\pi}^*$ with high probability. Note that the probability of utilizing the learned policy converges to 1 as episode increases. Hence the episodic average queue backlog when $\pi^{in}_k = \tilde{\pi}^*$ constitutes a large proportion of the overall expected average queue backlog. Therefore, the key step to upper bound the expected average queue backlog is to upper bound the episodic average queue backlog when $\pi^{in}_k = \tilde{\pi}^*$.

For the purpose of analysis, we further partition $\mathcal{S}^{in}$ as follows:

$$\begin{cases} \mathcal{S}^{in}_{in} \triangleq \left\{ \boldsymbol{Q} \in \mathcal{S}^{in} : \tilde{\Phi}^*(\boldsymbol{Q}) \leqslant b_1 (U-W)^{1+\beta} - b_3 U^{\beta} \right\} \\ \mathcal{S}^{in}_{bd} \triangleq \mathcal{S}^{in} \setminus \mathcal{S}^{in}_{in} \end{cases}.$$

We first upper bound the episodic average queue backlog when $\pi^{in}_k = \tilde{\pi}^*$, as stated in the following lemma.

Lemma 2. *Under Assumptions 1-3, we have*

$$\lim_{k\to\infty} \mathbb{E}\left[\frac{\sum_{t=t_{k-1}+1}^{t_k} \sum_i Q_i(t)}{L'_k} \mid \pi^{in}_k = \tilde{\pi}^* \right] = \tilde{\rho}^* + p^{\tilde{\pi}^*+\pi_0}\left(\mathcal{S}^{in}_{bd}\right) \cdot \mathcal{O}\left(U^{1+\max\{2\alpha,\gamma\}}\right).$$

Proof. For a given k such that $\pi^{in}_k = \tilde{\pi}^*$, we decompose the accumulated queue backlog into the three parts: the accumulated queue backlog when $\boldsymbol{Q} \in \mathcal{S}^{in}_{in}$, $\boldsymbol{Q} \in \mathcal{S}^{in}_{bd}$ and $\boldsymbol{Q} \in \mathcal{S}^{out}$. For the first part, we utilize Bellman equation analysis and Assumption 3. The second part is trivially upper bounded by NU. For the third part, we use Assumption 1 and prove that the time it takes to return back $\mathcal{S}^{in}$ is polynomial (using techniques as the proof of Theorem 1.1 in Chapter 5 of [34]). The details are omitted in this paper. □

We further establish the following lemma, which upper bounds $p^{\tilde{\pi}^*+\pi_0}\left(\mathcal{S}^{in}_{bd}\right)$.

Lemma 3. *Under Assumption 2, we have*

$$p^{\tilde{\pi}^*+\pi_0}\left(\mathcal{S}^{in}_{bd}\right) = \mathcal{O}\left(\exp\left(-U^{1-\beta}\right)\right).$$

Proof. We show that under Assumption 2 when $\pi^{in}_k = \tilde{\pi}^*$, there exists a sub-quadratic Lyapunov function with negative drift for states that have relatively large Lyapunov value regarding $\tilde{\Phi}^*(\cdot)$. We then apply similar techniques as Lemma 1 in [30] to complete the proof. The details are omitted in this paper. □

From Lemma 2 and Lemma 3, we can upper bound the episodic average queue backlog when $\pi^{in}_k = \tilde{\pi}^*$. By combining with Theorem 1, we establish the following main result of this paper.

Theorem 2. *Suppose Assumptions 1-4 hold. Applying* PDGRL *to M, the expected average queue backlog is upper bounded as*

$$\lim_{K\to\infty} \frac{\mathbb{E}\left[\sum_{t=1}^{t_K} \sum_i Q_i(t)\right]}{t_K} \leq \rho^* + \mathcal{O}\left(\frac{U^{1+\max\{2\alpha,\gamma\}}}{\exp\left(U^{1-\beta}\right)}\right).$$

Proof. We first show that the expected average queue backlog brought by episodes where $\pi^{in}_k \neq \tilde{\pi}^*$ becomes negligible as $K \to \infty$ and obtain the gap between the expected average queue backlog and $\tilde{\rho}^*$. We then apply similar techniques as the proof of Lemma 2 to characterize the relationship between ρ^* and $\tilde{\rho}^*$. □

Theorem 2 provides an upper bound on the performance guarantee of PDGRL regarding the threshold parameter U: by increasing U, the long-term average queue backlog approaches ρ^* exponentially fast.

V. Numerical Experiments

In this section, we evaluate the performance of proposed PDGRL for three queueing networks.

A. Dynamic Server Allocation

We first consider a simple server allocation problem: exogenous packets arrive at two nodes according to Bernoulli process with rate λ_1 and λ_2 respectively. Both nodes have unbounded buffers. At each time slot, a central server needs to select one of the two queues to serve. The head of line job in the selected queue i then completes the required service and leaves the system with probability p_i. The system model and parameters are illustrated in Figure 2.

According to [17], whenever $\lambda_1/p_1 + \lambda_2/p_2 < 1$, a stabilizing policy is to always serve the node with the longest

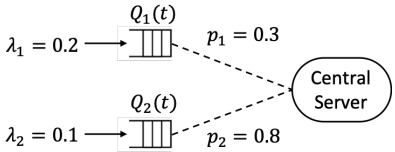


Fig. 2: System model of the dynamic server allocation problem with permanent connectivity.

connected queue (LCQ). Therefore, we can use the LCQ policy as π_0. Note that in our setting, the channels are always connected, π_0 is actually serving the node with the longest queue (LQ).

On the other hand, according to $c\mu$-rule in [19], the optimal policy π^* that minimizes the average queue backlog is to select the node with the largest successful transmission rate among all the non-empty queues. However, $c\mu$-rule requires knowledge of the dynamics, which is infeasible under our setting. We include it in simulation only to study the performance gap between PDGRL and optimum.

For the model depicted in Figure 2, under policy π^* the node 2 is served whenever it is nonempty. However, since node 1 has a larger arrival rate and a smaller successful transmission rate, the queue of node 1 is easier to get queued up. Therefore, we would expect that node 1 is served more frequently under policy π_0, leading a performance gap to the optimal result under π^*.

We compare the performances of four policies: π_0 (LCQ), PDGRL, $\tilde{\pi}^*+\pi_0$ (applying π^* for $\boldsymbol{Q} \in \mathcal{S}^{in}$ and π_0 otherwise) and π^* (true optimal policy). Note that the policy $\tilde{\pi}^* + \pi_0$ is exactly the best policy PDGRL can learn. We simulate it to illustrate the convergence rate of PDGRL.

We conduct the simulation with $U = 5$ and $U = 10$ and the results are plotted in Figure 3. It can be observed from Figure 3 that PDGRL outperforms π_0, and quickly converges to $\tilde{\pi}^*+\pi_0$ in both cases. The performance gap between PDGRL and π^* is not negligible when $U = 5$. Theorem 2 shows that when U increases, the average queue backlog of PDGRL approaches the optimal result. Figure 3 (b) shows that when $U = 10$ the gap between PDGRL and π^* almost diminishes.

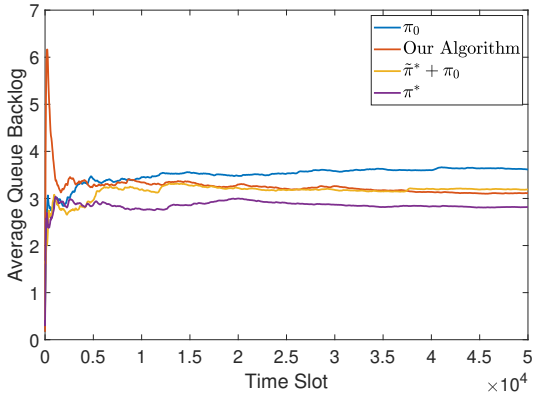
B. Dynamic Server Allocation (with Stochastic Connectivity)

We now turn to a more general case with stochastic connectivity: during each time slot, the nodes are connected with the central server with probability c_1 and c_2 respectively. The connectivity status is known before taking action. Only when the selected queue i is connected, it is successfully served with probability p_i. The system model and parameters are shown in Figure 4.

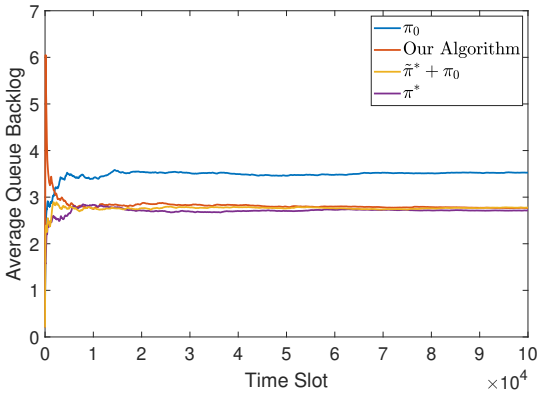
According to [17], whenever the condition that

$$\begin{cases} \frac{\lambda_1}{p_1} + \frac{\lambda_2}{p_2} < 1 - (1-c_1)(1-c_2) \\ \frac{\lambda_1}{p_1} < c_1, \ \frac{\lambda_2}{p_2} < c_2 \end{cases}$$

is satisfied, a stabilizing policy is to always serve the node with the longest connected queue (LCQ). Therefore, we can use LCQ policy as π_0.



(a) $U = 5$



(b) $U = 10$

Fig. 3: Simulation results for the dynamic server allocation problem.

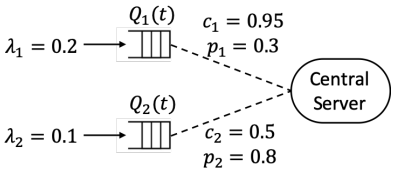
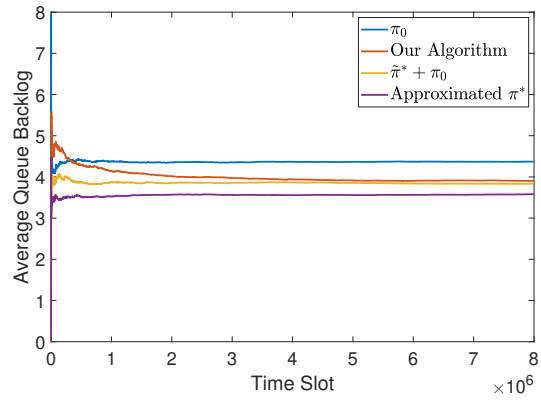


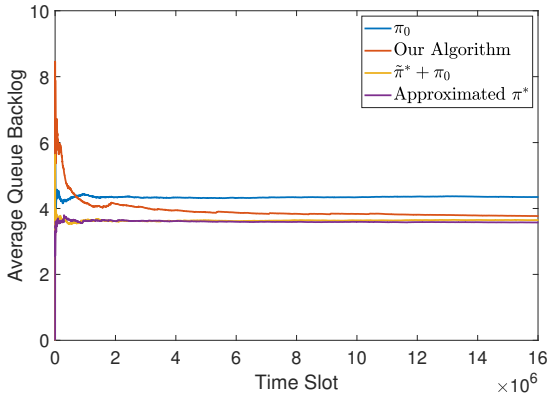
Fig. 4: System model of the dynamic server allocation problem with stochastic connectivity.

However, unlike the setting in Section V-A, for stochastic connectivity cases, unless the parameters are highly symmetric (i.e. $\lambda_1 \equiv \lambda$, $p_i \equiv p$ and $c_i \equiv c$), the optimal policy that minimizes average queue backlog remains an open problem [31].

From simulations we find that for the system depicted in Figure 4, the optimal policy $\tilde{\pi}^*$ for the truncated system always chooses serving queue 2 whenever it is connected and nonempty. When only one queue is connected, $\tilde{\pi}^*$ selects the connected queue. We thus consider a policy that inherits the behavior of $\tilde{\pi}^*$ to approximate the true optimal policy π^* for the real system: when only one queue is connected, select that queue; when both queues are connected, select the nonempty queue with the largest successful service probability p_i.



(a) $U = 5$



(b) $U = 10$

Fig. 5: Simulation results for the dynamic server allocation problem with stochastic connectivity.

We implement the simulation for $U = 5$ and $U = 10$. In this setting, we incorporate the connectivity status into the state (i.e. $(Q_1, Q_2, C_1, C_2) \in \mathcal{S} = \mathbb{N} \times \mathbb{N} \times \{0, 1\} \times \{0, 1\}$). The results are shown in Figure 5. Similar to the results in Section V-A, PDGRL outperforms π_0, and learns $\tilde{\pi}^*$ gradually. However, since the state space expands (connectivity is also incorporated into the state space), the convergence rate is smaller. Also, by comparing the gap between $\tilde{\pi}^* + \pi_0$ and the approximated π^* with different thresholds U, we observe that the average queue backlog of PDGRL approaches the optimal result as U increases.

C. Routing

We consider a routing problem: exogenous packets arrive at the source node s according to Bernoulli process with rate λ. Node 1 and node 2 are two intermediate nodes with unbounded buffers and can serve at most one packet during each time slot, with probability p_1 and p_2 respectively. Node d is the destination node. At each time slot, node s has to choose between routes $s \to 1 \to d$ and $s \to 2 \to d$ to transit new exogenous packets. Specifically, the system model and parameters are shown in Figure 6.

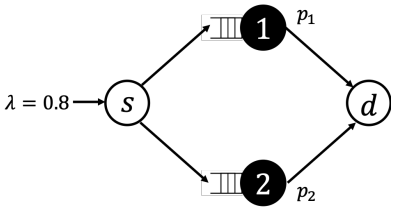


Fig. 6: System model of routing

The parameter p_is are queue-dependent here:

$$(p_1, p_2) = \begin{cases} (0.9, 0.1), & Q_2(t) \leqslant 5 \\ (0.1, 0.9), & Q_2(t) > 5 \end{cases}.$$

An obvious policy that stabilizes the queue backlog is to always choose $s \to 2 \to d$, which maintains $Q_2(t) > 5$. Therefore, we can use the policy that always routing through $s \to 2 \to d$ as π_0.

We simulate $U = 10$. The results are plotted in Figure 7, which shows that PDGRL outperforms π_0 and learns $\tilde{\pi}^*$ quickly.

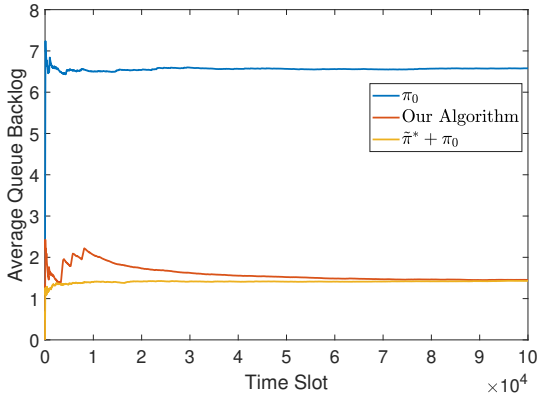


Fig. 7: Simulation results of routing.

VI. Conclusion

In this work, we apply a model-based reinforcement learning framework to general queueing networks with unbounded state space. We propose the PDGRL algorithm, which applies a ϵ-greedy exploration scheme. By leveraging Lyapunov analysis, we show that the average queue backlog of the proposed approach can get arbitrarily close to the optimal average queue backlog under oracle policy. The proposed PDGRL algorithm requires the knowledge of a stable policy. An interesting future direction is to investigate this problem when such information is not available.

References

[1] Ramesh K Sitaraman, Mangesh Kasbekar, Woody Lichtenstein, and Manish Jain. Overlay networks: An akamai perspective. *Advanced Content Delivery, Streaming, and Cloud Services*, 51(4):305–328, 2014.

[2] Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In *29th IEEE Conference on Decision and Control*, pages 2130–2132. IEEE, 1990.

[3] Michael J Neely, Eytan Modiano, and Chih-Ping Li. Fairness and optimal stochastic control for heterogeneous networks. *IEEE/ACM Transactions On Networking*, 16(2):396–409, 2008.

[4] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[5] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[6] Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

[7] Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, pages 3003–3011, 2013.

[8] Junghee Han, David Watson, and Farnam Jahanian. Topology aware overlay networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 4, pages 2554–2565. IEEE, 2005.

[9] Zhi Li and Prasant Mohapatra. Qron: Qos-aware routing in overlay networks. *IEEE Journal on Selected Areas in Communications*, 22(1):29–40, 2004.

[10] Anurag Rai, Rahul Singh, and Eytan Modiano. A distributed algorithm for throughput optimal routing in overlay networks. *arXiv preprint arXiv:1612.05537*, 2016.

[11] Sajee Singsanga, Wipawee Hattagam, and Ewe Hong Tat. Packet forwarding in overlay wireless sensor networks using nashq reinforcement learning. In *2010 Sixth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 85–90. IEEE, 2010.

[12] Olivier Brun, Lan Wang, and Erol Gelenbe. Big data for autonomic intercontinental overlays. *IEEE Journal on Selected Areas in Communications*, 34(3):575–583, 2016.

[13] Qingkai Liang and Eytan Modiano. Optimal network control in partially-controllable networks. *arXiv preprint arXiv:1901.01517*, 2019.

[14] Georgios Theocharous, Zheng Wen, Yasin Abbasi-Yadkori, and Nikos Vlassis. Posterior sampling for large scale reinforcement learning. *arXiv preprint arXiv:1711.07979*, 2017.

[15] Oshri Naparstek and Kobi Cohen. Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–7. IEEE, 2017.

[16] Shangxing Wang, Hanpeng Liu, Pedro Henrique Gomes, and Bhaskar Krishnamachari. Deep reinforcement learning for dynamic multichannel access in wireless networks. *IEEE Transactions on Cognitive Communications and Networking*, 4(2):257–265, 2018.

[17] Leandros Tassiulas and Anthony Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, 1993.

[18] JS Baras, D-J Ma, and AM Makowski. K competing queues with geometric service requirements and linear costs: The μc-rule is always optimal. *Systems & control letters*, 6(3):173–180, 1985.

[19] C Buyukkoc, P Varaiya, and J Walrand. The cμ rule revisited. *Advances in applied probability*, 17(1):237–238, 1985.

[20] Dimitris Bertsimas, Ioannis Ch Paschalidis, and John N Tsitsiklis. Optimization of multiclass queueing networks: Polyhedral and nonlinear characterizations of achievable performance. *The Annals of Applied Probability*, pages 43–75, 1994.

[21] Carlos Humes Jr, J Ou, and PR Kumar. The delay of open markovian queueing networks: Uniform functional bounds, heavy traffic pole multiplicities, and stability. *Mathematics of Operations Research*, 22(4):921–954, 1997.

[22] Sunil Kumar and PR Kumar. Performance bounds for queueing networks and scheduling policies. *IEEE Transactions on Automatic Control*, 39(8):1600–1611, 1994.

[23] PR Kumar and Sean P Meyn. Stability of queueing networks and scheduling policies. *IEEE Transactions on Automatic Control*, 40(2):251–260, 1995.

[24] Dimitris Bertsimas, David Gamarnik, John N Tsitsiklis, et al. Performance of multiclass markovian queueing networks via piecewise linear lyapunov functions. *The Annals of Applied Probability*, 11(4):1384–1428, 2001.

[25] Michael J Neely and Longbo Huang. Dynamic product assembly and inventory control for maximum profit. In *49th IEEE Conference on Decision and Control (CDC)*, pages 2805–2812. IEEE, 2010.

[26] Jim G Dai and Wuqin Lin. Maximum pressure policies in stochastic processing networks. *Operations Research*, 53(2):197–218, 2005.

[27] Mihalis G Markakis, Eytan Modiano, and John N Tsitsiklis. Max-weight scheduling in queueing networks with heavy-tailed traffic. *IEEE/ACM Transactions on Networking (TON)*, 22(1):257–270, 2014.

[28] Mihalis G Markakis, Eytan Modiano, John N Tsitsiklis, Mihalis G Markakis, Eytan Modiano, and John N Tsitsiklis. Delay stability of back-pressure policies in the presence of heavy-tailed traffic. *IEEE/ACM Transactions on Networking (TON)*, 24(4):2046–2059, 2016.

[29] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[30] Dimitris Bertsimas, David Gamarnik, John N Tsitsiklis, et al. Geometric bounds for stationary distributions of infinite markov chains via lyapunov functions. 1998.

[31] Anand Ganti, Eytan Modiano, and John N Tsitsiklis. Optimal transmission scheduling in symmetric communication models with intermittent connectivity. *IEEE Transactions on Information Theory*, 53(3):998–1008, 2007.

[32] Tsachy Weissman, Erik Ordentlich, Gadiel Seroussi, Sergio Verdu, and Marcelo J Weinberger. Inequalities for the l1 deviation of the empirical distribution. *Hewlett-Packard Labs, Tech. Rep*, 2003.

[33] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 2017.

[34] Pierre Brémaud. Lyapunov functions and martingales. In *Markov Chains*, pages 167–193. Springer, 1999.

[35] Zhenting Hou and Qingfeng Guo. *Homogeneous denumerable Markov processes*. Springer Science & Business Media, 2012.