Adiabatic Markov Decision Process: Convergence of Value Iteration Algorithm

Thai Duong, Duong Nguyen-Huu and Thinh Nguyen
School of Electrical Engineering and Computer Science,
Oregon State University,
Corvallis, Oregon 97331, USA
Email:{duong, nguyendu,thinhq}@eecs.oregonstate.edu

ABSTRACT

Markov Decision Process (MDP) is a well-known framework for devising the optimal decision making strategies under uncertainty. Typically, the decision maker assumes a stationary environment which is characterized by a time-invariant transition probability matrix. However, in many real-world scenarios, this assumption is not justified, thus the optimal strategy might not provide the expected performance. In this paper, we study the performance of the classic Value Iteration algorithm for solving an MDP problem under non-stationary environments. Specifically, the non-stationary environment is modeled as a sequence of time-variant transition probability matrices governed by an adiabatic evolution inspired from quantum mechanics. We characterize the performance of the Value Iteration algorithm subject to the rate of change of the underlying environment. The performance is measured in terms of the convergence rate to the optimal average reward. We show two examples of queuing systems that make use of our analysis framework.

1 Introduction

1.1 Markov Decision Process

The theory of Markov Decision Process (MDP) aims to study optimal decision making processes under uncertainty. It is widely used in economics, engineering, operation research, and artificial intelligence. In an MDP setting, there is a controller who interacts with its environment by taking actions based on its observations at every discrete time step. Each action by the controller induces a change in the environment. Typically, the environment is described by a finite set of states. An action will move the environment from the current state to some other states with certain probabilities. Associated with each action in each state is a reward given to the controller. The goal of the controller is to maximize the expected cumulative reward or average reward over some finite or infinite number of time steps by making sequential decisions based on its current observations.

It is not difficult to find many applications of the MDP framework. A classic application of MDP is the warehouse example in operation research. In this setup, a company's business is to buy and sell a number of merchandises. To operate smoothly, it uses a warehouse to store the merchandises that allows shipments to the buyers promptly. Everyday, it has to make the decision on how many and which items it should buy and store in its warehouse subject to the uncertainty of the market demands. Buying too many items would incur high storage costs while buying too little would run the risk of not having the items ready for shipping, and thus reducing profits. The MDP framework enables the company to decide on the optimal action, i.e., how many and which items it should buy on a given day in order to maximize the expected cumulative reward, i.e., its profits over a month, a year, or indefinitely. Naturally, the optimal action should base on the environmental states, i.e., the current status of different items in the stock and the current market demands.

A solution of an MDP problem is an optimal policy. A policy/decision rule is a mapping from the states to the action. The optimal policy would produce the maximum expected cumulative reward. For the infinite-horizon MDP models, to be discussed subsequently, there are a number of classic algorithms for finding such optimal policies. These algorithms include Value Iteration [1], Policy Iteration [2], Linear Programming [3, 4], which all are based on the Bellman equations [1, 5]. All also assume a stationary policy, i.e., a policy that does not change with time. This assumption is justified as it is well-known that for a stationary environment, there exists an optimal policy that is also stationary. Fundamentally, the MDP framework relies on the assumption that a given policy will induce a stationary dynamics on the states. Moreover, the state changes are characterized by a time-invariant transition probability matrix P.

1.2 Non-stationary Environment

For many real-world scenarios, this assumption is not justified, thus the optimal policy might not provide the expected performance. In this paper, we study the performance of the classic Value Iteration algorithm for solving an MDP problem under non-stationary environments. Specifically, the non-stationary environment is modeled as a sequence of time-variant transition probability matrices governed by an adiabatic evolution inspired from quantum mechanics [6–9]. Formally, the transition probability matrix P_i^d at time step i induced by decision rule d is determined by:

$$P_i^d = \Phi(i)P_0^d + (1-\Phi(i))P_f^d, \quad \forall d \tag{1}$$

where P_0^d and P_f^d are the transition probability matrices induced by the decision rule d at time step 0 and ∞, and $\Phi(.)$ characterizes the rate of change of the system with $\Phi(0) = 1$ and $\Phi(\infty) = 0$. The decision rule will be formalized in the next section.

The above transition model can be applied in two interesting scenarios. In the first scenario, P_i^d models the actual dynamics of the underlying non-stationary environment under the decision rule d at step i. In other words, the environment is initially characterized by P_0^d, then its dynamic is characterized by $\Phi(\cdot)$ and converges to P_f^d according to $\Phi(\cdot)$. In the second scenario, the environment is assumed to be stationary, and is characterized by P_f^d. However the estimation of the environmental parameters is initially inaccurate, and thought to be P_0^d. Thus, the actions/decisions are made based on inaccurate knowledge of the environment. Over time, the estimations of the environmental parameters become increasingly more accurate, i.e., P_i^d getting closer to P_f^d. Therefore, using an MDP algorithm such as Value Iteration will eventually produce the optimal solution due to increasing accuracy. Over time, the decisions are closer to the optimal ones, and eventually converge to the optimal policy. That said, we characterize the performance of the Value Iteration algorithm subject to the rate of change in the environment as characterized by $\Phi(\cdot)$. The performance is measured in terms of the convergence rate to the optimal average reward. We present two queuing system examples illustrating the two scenarios above that make use of our analysis framework.

1.3 Related Work

A learning scheme in varying environment is presented by Szita et al. in [10]. This work introduces the concept of $\varepsilon-$MDP where at each step, the transition matrix is perturbed by small noise bounded by ε around a base MDP. It shows that the Value Iteration algorithm for $\varepsilon-$MDP converges to a sub-optimal value which is $\delta-$ close to the optimal value where $\delta \propto \varepsilon$. In addtition, the paper shows that Q-learning can return a sub-optimal decision rule/policy in a varying environments. On the other hand, in this paper we consider the change in transition matrix (environment) in adiabatic settings which is characterized by a non-increasing function Φ. Given the function Φ, we show that the Value Iteration algorithm converges to the optimal value of the final MDP (base MDP), derive bounds on the convergence rate of the Value Iteration algorithm and therefore can predict its stopping time.

The adiabatic evolution was first studied in quantum mechanics by Born and Fock [6]. They provided a first important adiabatic theorem for unitary matrices. Basically, the theorem shows that in the evolution from the initial Hamiltonian to the final one, a system converges to the ground state of the final Hamiltonian after a large time. Recently, another type of adiabatic theorem in Markov chain was presented for linear evolution by Kovchegov in [8] and for general evolution by Bradford and Kovchegov in [9]. In [8], the adiabatic time of a time homogeneous Markov chain with linear evolution, which is the chain's mixing time, is shown as an order of the square of the mixing time of the final transition matrix. Generally, Bradford et al. shows that the stable adiabatic time, which is a general concept of adiabatic time, is an order of the maximum mixing time to the power of four [11]. The idea of applying adiabatic evolution to Partially Observable Markov Decision Process is mentioned in [12].

For queuing application, an adiabatic approach to analysis of adaptive queuing policies was proposed in [13]. It shows that for continuous-time queuing systems with arrival rate estimation, the queue converges to the final distribution after a large time with high probability. It also evaluates the adiabatic time for the queueing system. In our work [14], a Markov Decision Process in non-stationary environments is modeled as a sequence of time-variant transition matrices governed by adiabatic evolution. The convergence rate of the Value Iteration Algorithm is partly evaluated.

1.4 Our Contribution

In this paper, we study the Markov Decision Process in non-stationary environment modeled as an adiabatic evolution. Specifically, the convergence of Value Iteration Algorithm is evaluated by an upper bound on the distance between the actual average reward in Value Iteration and the optimal average reward. As a result, we can derive the necessary time for the Value Iteration Algorithm to get $\varepsilon-$close to the optimal one. Application to an M/M/1/K continuous-time queue with estimated arrival rate and a discrete-time queue with changing arrival rate are shown to verify the results.

The Section 2 provides some background on the theory of Markov Decision Process, Value Iteration which are necessary for the development of our results. In Section 3, we formulate the problem in term of the distance from the average reward to the optimal one using Value Iteration algorithm under changing environment. The theoretical results on the convergence rate

of Adiabatic Value Iteration Algorithm are also presented in this Section. Section 4 formulates Adiabatic Markov Decision Process for two examples of queuing systems based on the theoretical results. Finally, some conclusions and future work on noisy stochastic matrices are provided in the Section 5.

2 Mathematical Preliminaries

In this Section, we present some definitions, notations and some propositions for stationary environments.

2.1 Markov Decision Process

2.1.1 Definitions

A typical discrete-time MDP represents a dynamic system and is specified by a finite set of states S, representing the possible states of the system, a set of control actions A_s for each state $s \in S$, a transition probability matrix $P^{|S|\times|S|}$, and a reward function r. The transition probability specifies the dynamics of the system whose each entry $P_{ij} \triangleq P(s_{t+1} = j | s_t = i, a_t = a)$ represents the conditional probability of the system moving to state $s_{t+1} = j$ in the next time step after taking an action a in the current state $s_t = i$. The dynamics are Markovian in the sense that the probability of the next state j depends only on the current state i and the action a, and not on any previous history. The reward function $r(s,a)$ assigns a real number to the state s the action a, so that $r(s,a)$ represents the immediate reward of being in state s and taking action a. A policy $\pi = \{d_1, d_2, \ldots\}$ is a sequence of decision rules. Each decision rule d_t is a mapping from states to actions at each time step: $d_t : S \to A$, and induces a corresponding transition probability matrix where $A = \bigcup_{s\in S} A_s$. The policy π is called stationary if its actions depend only on the state s, independent of time, i.e., $\pi = \{d, d, d, \ldots\}$. A stationary policy induces a time-invariant transition probability matrix. Every policy π is associated with a value function $V^\pi(s)$ such that $V^\pi(s)$ gives the expected cumulative reward achieved by π when starting in state s.

Let Π be the policy space. It can be MD (Markovian and Deterministic), HD (History Dependent and Deterministic), MR (Markovian and Randomized) or HR (History Dependent and Randomized) [5]. Clearly, $\Pi^{MD} \subset \Pi^{MR} \subset \Pi^{HR}$ and, $\Pi^{MD} \subset \Pi^{HD} \subset \Pi^{HR}$. Moreover, for each $\pi \in \Pi^{HD}, s \in S$, there exists a $\pi' \in \Pi^{MD}$ such that their transition matrices are the same [15]. Therefore, we only consider Markovian Deterministic policy.

Definition 1 (Unichain Markov Decision Process). *A MDP is unichain if the transition matrix corresponding to every deterministic stationary policy consists of a single recurrent class plus a possibly empty set of transient states.*

The solution to an MDP problem is an optimal policy π^* that maximizes the expected cumulative reward over some finite or infinite number of time steps. The former and latter are termed finite-horizon MDP and infinite-horizon MDP, respectively. An infinite-horizon model has two typical forms of reward functions: the discounted and the average reward functions. The discounted reward function is defined as:

$$V^\pi_{dis}(s) = E^\pi_s \left\{ \sum_{t=1}^{\infty} \alpha^t r_t(s_t, a_t) \right\}, \tag{2}$$

where $0 < \alpha < 1$ denotes a given discount factor that provides convergence of $V^\pi(s)$, but also carries the notion of discounting the future reward, i.e., putting less emphasis on the rewards in the far future than those in the near future. The average reward function is defined as:

$$V^\pi_{ave}(s) = \lim_{N\to\infty} \frac{v_N}{N}, \tag{3}$$

where $v^\pi_N(s) = E^\pi_s \left[\sum_{t=1}^{N} r(s_t, a_t)\right]$.

For discounted reward function with discount factor $0 < \alpha < 1$, [5] shows that the convergence of value iteration for discounted reward function is controlled by function α^n which we can easily find the convergence rate. Therefore, in this paper, we only consider the average reward function criteria presented in Appendix A.

2.1.2 Span Seminorm

The span seminorm $sp(v)$ of a vector v is used to evaluate the convergence rate of value iteration for MDP using Average Reward Criterion ([5]). See Appendix B for definition and properties of span seminorm.

Proposition 1 ([5]). *Let $v \in V$ and $d \in D$ where D is the set of decision rule, then:*

$$sp(P^d v) \le \delta_d sp(v)$$

where delta coefficient $\delta_d = 1 - \min_{s,u\in S\times S} \sum_{j\in S} \min\{P^d(j|s), P^d(j|u)\}$, P^d is the transition matrix corresponding to the decision rule d
Furthermore, $0 \le \delta_d \le 1$, and there exists a $v \in V$ such that $sp(P^d v) = \delta_d sp(v)$

2.1.3 Value Iteration

The Value Iteration algorithm is an iterative algorithm for finding an ε-optimal policy for the infinite-horizon MDP. More precisely, given an ε, the Value Iteration algorithm guarantees to produce a reward value within an ε of the optimal value. The key to the Value Iteration algorithm is that each step of the algorithm can be viewed as applying a contracting operator L on v. Running the algorithm iteratively, or equivalently, applying the operator L repeatedly, will guarantee that v will converge to the optimal value based on Bellman equation. Specifically, for a unichain MDP, at each iteration n, we have: $v_{n+1} = Lv_n$, where L is defined as $Lv = \max_{d\in D}\{r^d + P^d v\}$, r^d and P^d denote the reward and the transition probability matrix induced by the decision rule d. The pseudo-code for the Value Iteration algorithm with average reward objective is shown below.

Definition 2 (The Value Iteration, [5]). *. The algorithm for the Value Iteration with Average Reward Criteria is shown below:*

1) Choose any initial reward vector v_0, for a given $\varepsilon > 0$. Let $n = 0$.
2) For each $s \in S$, we have: $v_{n+1}(s) = \max_{a\in A}\{r(s,a) + \sum_{j\in S} p(j|s,a)v_n(j)\}$.
3) Increasing n until $sp(v_{n+1} - v_n) < \varepsilon$, then choose: $d_\varepsilon \in \arg\max\{r(s,a) + \sum_{j\in S} p(j|s,a)v_n(j)\}$.

where $sp(v)$ is the span seminorm of vector v.

We note that the ε-optimal policy approaches to an optimal policy as ε reduces to zero when the number of iterations goes to infinity.

Definition 3 (Gamma coefficient). *The gamma coefficient is defined as follows [5]:*

$$\gamma = \max_{s\in S, a\in A_s, s'\in S, a'\in A_{s'}} \left[1 - \sum_{j\in S} \min\{p(j|s,a), p(j|s',a')\}\right]$$

Easily, we can see that the gamma coefficient is an upper bound of the delta coefficient δ_d for all decision rule d. Therefore, from Proposition 1, we have the following Proposition:

Proposition 2. *Let $v \in V$ and $d \in D$ where D is the set of decision rule, then:*

$$sp(P^d v) \le \gamma sp(v)$$

Proposition 3 (Convergence of Value Iteration, [5]). *For unichain MDPs, we have:*

$$sp(v^{n+2} - v^{n+1}) \le \gamma sp(v^{n+1} - v^n),$$

where $\gamma = \max_{s\in S, a\in A_s, s'\in S, a'\in A_{s'}} \left[1 - \sum_{j\in S} \min\{p(j|s,a), p(j|s',a')\}\right]$. Then if $\gamma < 1$, the Value Iteration algorithm will stop after a finite step n.

3 Adiabatic Markov Decision Process

This Section formalizes the adiabatic evolution and provides the necessary background for performance study of Value Iteration algorithm in non-stationary setting.

3.1 Preliminaries on Adiabatic-Time Evolution

We introduce Adiabatic-time framework to model non-stationary environments. The initial adiabatic setting was first described in quantum mechanics by Born and Fock in 1928 [6]. Recently, the adiabatic setting for both discrete-time and continuous-time Markov chain was proposed by Kovchegov [8] which is shown below:

$$\begin{aligned} P_i &= \Phi(i)P_0 + (1 - \Phi(i))P_f \\ &= P_f + \Phi(i)(P_0 - P_f) \end{aligned} \tag{4}$$

where P_0, P_f are the initial transition matrix and the final transition matrix, respectively. P_i is the transition matrix at time i. $\Phi(\cdot)$ is a function to characterize the adiabatic evolution such that $\Phi(i) : [0, +\infty) \to [0, 1]$, $\Phi(0) = 1$ and $\lim_{i\to\infty} \Phi(i) = 0$.

Because of the properties of $\Phi(\cdot)$, we have: $\lim_{i\to\infty} P_i = P_f$. For each applications, we have different $\Phi(\cdot)$ function. In this paper, we suppose that the Φ is given.

The following propositions and corollaries are needed for the development of convergence of Value Iteration with Adiabatic Setting.

Proposition 4 (The bound on gamma coefficients). *Given $P_i = \Phi(i)P_0 + (1 - \Phi(i))P_f, \Phi(i) \in [0, 1]$:*

$$\gamma_i \le \Phi(i)\gamma_0 + (1 - \Phi(i))\gamma_f$$

Proof. See Appendix C for details.

Corollary 1. *Given $P_i = \Phi(i)P_0 + (1-\Phi(i))P_f$ with non-increasing function $\Phi(i) > 0, \Phi(i) \in [0,1]$ for any $n_0 \leq i$*

$$\gamma_i \leq \max(\gamma_{n_0}, \gamma_f) \tag{5}$$

Proof. See Appendix C for details.

Note that one way to ensure that $0 < \gamma < 1$ is that for each decision rule d, there exists a column of the corresponding P having all positive entries.

3.2 Problem Formulation

3.2.1 Adiabatic MDP Model

Typically, the Value Iteration algorithm is used to find an ε-optimal stationary policy in an offline manner using a number of iterations, assuming a stationary environment such that every stationary policy π induces a time-invariant transition probability matrix. The resulting policy is then used in an online manner with the assumption that the environment is stationary and governed by the time-invariant transition probability matrices in the Value Iteration algorithm. In this paper, we study an adiabatic MDP setting in which, we assume that the "environment" is no longer stationary. Instead, it might change at every iteration of the Value Iteration algorithm, resulting in a sequence of time-variant transition probability matrices under a stationary policy. The precise meaning of the "environment" will be clear shortly.

Instead of running the Value Iteration algorithm offline to find an ε-optimal policy, we apply the decision rule found after each iteration immediately and repeatedly in an online manner. Our goal is to determine how good the reward is for such a scheme. The analysis of such a setting is useful in the rapidly-changing environments where decisions must be made quickly. Unlike the classic MDP setting where for each decision rule d, there is a time-invariant transition probability matrix P^d, in our setting, for a fixed decision rule d, there is sequence of time-variant transition probability matrices:

$$\begin{aligned} P_i^d &= \Phi(i)P_0^d + (1-\Phi(i))P_f^d \\ &= P_f^d + \Phi(i)(P_0^d - P_f^d), \quad \forall d \end{aligned} \tag{6}$$

where Φ is a function such that:

$$\Phi(i) : [0,+\infty) \to [0,1], \Phi(0) = 1, \lim_{i\to\infty} \Phi(i) = 0. \tag{7}$$

$\Phi(\cdot)$ characterizes the change in the environment at the iteration i of the Value Iteration algorithm, and P_i^d is the induced transition probability matrix due to the decision rule found at the iteration i of the Value Iteration algorithm. A slowly changing $\Phi(i)$ implies a slow change in the environment. We note that the notion of optimal reward is not well defined if the environment fluctuates arbitrarily. Thus in the model above, we assume that the environment will approach to a final stationary environment characterized by P_f^d to ensure a well-defined reward. This can be seen as $\lim_{i\to\infty} P_i^d = P_f^d$. In this paper, we also assume that the function $\Phi(\cdot)$ is the same for all decision rules d as seen in the example below.

A Simple Example. Consider a discrete-time queue with size $K = 2$. Let p, q be the probabilities that a packet arrives and departs the queue, respectively. The state space is $S = \{0,1,2\}$. The action is the value of q. For this queue, the transition matrix is:

$$P = \begin{bmatrix} 1-p(1-q) & p(1-q) & 0 \\ q(1-p) & pq+(1-p)(1-q) & p(1-q) \\ 0 & q(1-p) & 1-q(1-p) \end{bmatrix}$$

If we let p change over time following the rule: $p_i = \phi(i)p_0 + (1-\phi(i))p_f$ where p_0, p_f are the initial and final arrival probabilities, respectively. Since each entry in the transition matrix is a linear function of p, for any decision rule d which is a mapping from state space S to a set of values of q, the transition matrix is changing follow the rule $P_i^d = \Phi(i)P_0^d + (1-\Phi(i))P_f^d$ where $\Phi(\cdot) = \phi(\cdot)$. Note that the function $\Phi(\cdot)$ is the same for all decision rule d.

We now articulate a bit more on the meaning of the "environment." Note that the induced P_i^d depends on both actions and environments. Therefore, a change in the environment implies possible changes in the underlying environments, or the set of actions, or the combination of both over time. For example, let us consider a queuing system in which the controller attempts to send packets (actions) at some varying rates based on the number of packets in the queue in order to maximize a given reward. In one scenario, we assume the traffic arrival rate at the queue increases steadily from a initial rate of λ_0 to a final rate of λ_f. As a result, P_i^d varies as the underlying environment changes over time. In another scenario, the arrival rate of the packets remains the same, however, the controller has inaccurate estimation of the arrival rate initially due to few observations. Consequently, it makes the decision on what rate it should send based on an inaccurate arrival rates, and P_i^d characterizes the change based on its decision rule d at iteration i. However, over time with more observations, its estimation of the arrival rate becomes more accurate. Therefore, its decision rule approaches the optimal one for which, the state dynamic is characterized by P_f^d. We will discuss these two examples in more detail in the later section.

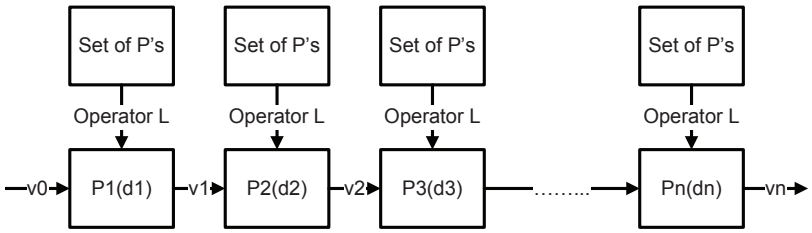

Fig. 1. The classic Value Iteration

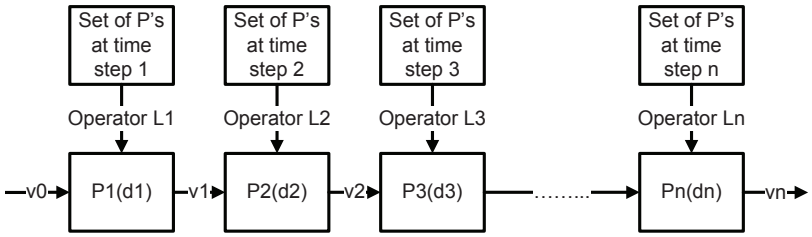

Fig. 2. The Value Iteration in an adiabatic setting

3.2.2 Convergence Rate of Adiabatic MDP

It is important to emphasize again that the environment will be asymptotically stationary corresponding to an induced transition probability matrix P_f^d for any decision rule d. In addition, there exists an optimal decision rule d^* corresponding to a transition probability matrix $P_f^{d^*}$ which can be obtained when running the Value Iteration algorithm under a stationary environment [5]. Importantly, running the Value Iteration algorithm in an adiabatic setting for a sufficiently large number of steps would produce the same optimal decision rule d_f^* as that of the classic Value Iteration algorithm, and also the same average reward g^* (see Appendix A):

$$g^* = P_{d^*}^{\infty} r^{d_f^*} = \pi_{P_f}^{d^*} r^{d_f^*} e,$$

where by Cesaro mean, $P_{d^*}^{\infty} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} (P_f^{d^*})^{n-1} = \lim_{n\to\infty} (P_f^{d^*})^n = \begin{bmatrix} \pi_{P_f}^{d_f^*} \\ \pi_{P_f}^{d_f^*} \\ \dots \\ \pi_{P_f}^{d_f^*} \end{bmatrix}$, $\pi_{P_f}^{d_f^*}$ is the stationary distribution for $P_f^{d_f^*}$, and $e = [11\dots1]^T$.

However, the convergence rates to this final reward g^* are quite different for the classic MDP and adiabatic MDP settings. One would expect that the rate in the former setting would be faster since the environment does not change, thus the Value Iteration algorithm can learn faster than that of the latter. Therefore, our primary focus of this paper is to characterize the convergence rate of the Value Iteration algorithm in an adiabatic setting given the dynamics specified by $\Phi(\cdot)$. Specifically, we want to find an integer N such that $\forall n > N$,

$$E = \left\| \frac{v_n}{n} - g^* \right\|_{\infty} \leq \varepsilon \tag{8}$$

Note that the index n is both the step index in Value Iteration algorithm and the time step. In other words, we run the Value Iteration algorithm while the environment is changing. The reason for this on-line manner is that it might take a long time to wait until the environment is stable then run the classic Value Iteration algorithm or the environment keeps changing.

The difference between the classic Value Iteration and the Value Iteration in adiabatic setting is illustrated in Figures 1 and 2. We can see that, compared to the classic Value Iteration algorithm, the set of transition matrix P, where we are looking for the optimal one, is changing at every time step, i.e. also the step in Value Iteration algorithm. In the Value Iteration in adiabatic setting, we apply the same operator on different set of matrices at each step. Denote L_i as the operator L applied on the set of matrices at step i.

Finding N is not trivial. Therefore, we will provide a lower a bound on N which depends on $\Phi(\cdot)$ as well as the set of all possible matrices P_f^d.

3.3 Main Results

Theorem 1 (Main Result 1). *Consider a unichain adiabatic-time MDP with S and A_s both finite, $|r(s,a)|$ bounded by a number M. Suppose $0 < \gamma = \max(\gamma_f, \gamma_{n_0}) < 1$, then*

$$E \leq \frac{||v_{n_0} - n_0 g^*||_\infty}{n} + \frac{1}{n}\sum_{i=n_0}^{n-1}\left[\left(\prod_{k=n_0}^{i-1}\gamma_k\right) sp(v_{n_0+1} - v_{n_0}) + YM\left(e_i + 2\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right)\right],$$

for any n_0, where $\prod_{k=u}^{v}(\cdot) = 1$ if $v < u$, $e_i = \sum_{j=i+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|), 2Y = \max_{d\in D}\|P_0 - P_f\|_\infty$.

Proof. From (6), for any $d \in D$ we have:

$$\begin{aligned}|P_i^d - P_{i-1}^d| &\leq |\Phi(i) - \Phi(i-1)||P_0^d - P_f^d| \leq |\Phi(i) - \Phi(i-1)||P_0^d - P_f^d|,\\ &\leq 2Y|\Delta\Phi_{i-1}|, \text{where fixed } 2Y = \max_{d\in D}|P_0 - P_f|. \end{aligned} \tag{9}$$

Consider $E = \left\|\frac{v_n}{n} - g^*\right\|_\infty$:

$$\begin{aligned} E &= \left\|\frac{v_n}{n} - g^*\right\|_\infty, \\ &= \left\|\frac{v_n - ng^*}{n}\right\|_\infty = \left\|\sum_{i=n_0}^{n-1}\left(\frac{v_{i+1} - v_i - g^*}{n}\right) + \frac{v_{n_0} - n_0 g^*}{n}\right\|_\infty, \\ &\leq \sum_{i=n_0}^{n-1}\frac{\|v_{i+1} - v_i - g^*\|_\infty}{n} + \frac{||v_{n_0} - n_0 g^*||_\infty}{n}. \end{aligned} \tag{10}$$

First, we bound $\|v_{i+1} - v_i - g^*\|_\infty$. Let $x_i = \arg\max_{s\in S}(v_{i+1} - v_i) = \arg\max_{s\in S}(L_i v_i - L_{i-1}v_{i-1})$, and $y_i = \arg\min_{s\in S}(v_{i+1} - v_i) = \arg\min_{s\in S}(L_i v_i - L_{i-1}v_{i-1})$. Let d_i be the optimal decision rule corresponding to the operator L_i. Let d_{i-1} be the optimal decision rule corresponding to the operator L_{i-1}. Let L^d be the operator that we apply the decision rule d: $L^d v = r^d + P^d v$. Since $L_{i-1}v_{i-1}(x_i) \geq L_{i-1}^{d_i}v_{i-1}(x_i)$,

$$\begin{aligned} L_i v_i(x_i) - L_{i-1}v_{i-1}(x_i) \leq L_i^{d_i}v_i(x_i) - L_{i-1}^{d_i}v_{i-1}(x_i) &= (r_{d_i}(x_i) + P_i^{d_i}v_i(x_i)) - (r_{d_i}(x_i) + P_{i-1}^{d_i}v_{i-1}(x_i)),\\ &= P_i^{d_i}v_i(x_i) - P_{i-1}^{d_i}v_{i-1}(x_i),\\ &= (P_i^{d_i} - P_{i-1}^{d_i})v_i(x_i) + P_{i-1}^{d_i}(v_i - v_{i-1})(x_i). \end{aligned}$$

Let $\alpha_i = \arg\max_{s\in S}(v_i(s)), \beta_i = \arg\min_{s\in S}(v_i(s)), \Delta P_i^{d_i} = P_i^{d_i} - P_{i-1}^{d_i}$. We have: $a = \Delta P_i^{d_i}v_i(x_i) = (\Delta P_i^{d_i}(x_i,\cdot))v_i = (\Delta P_i^{d_i}(x_i,\cdot))(v_i - v_i(\beta_i)e)$ where $e = [11\ldots1]^T$ since $P_i^{d_i}, P_{i-1}^{d_i}$ are transition matrices, then $\sum_{s\in S}(\Delta P_i^{d_i}(x_i,s)) = 0$. Then,

$$\begin{aligned} a &= \sum_{s\in S}\Delta P_i^{d_i}(x_i,s))(v_i(s) - v_i(\beta_i)) \leq \sum_{\Delta P_i^{d_i}(x_i,s))\geq 0}\Delta P_i^{d_i}(x_i,s))(v_i(s) - v_i(\beta_i)),\\ &\leq (v_i(\alpha_i) - v_i(\beta_i))\sum_{\Delta P_i^{d_i}(x_i,s))\geq 0}\Delta P_i^{d_i}(x_i,s)),\\ &\leq (v_i(\alpha_i) - v_i(\beta_i))\frac{||P_i^{d_i} - P_{i-1}^{d_i}||_\infty}{2},\\ &\leq Y|\Delta\Phi_{i-1}|sp(v_i) \qquad \text{since (9).} \end{aligned}$$

Then, we have:

$$L_i v_i(x_i) - L_{i-1}v_{i-1}(x_i) \leq YMsp(v_i)|\Delta\Phi_{i-1}| + P_{i-1}^{d_i}(v_i - v_{i-1})(x_i). \tag{11}$$

Similarly,

$$L_i v_i(y_i) - L_{i-1}v_{i-1}(y_i) \geq -YMsp(v_i)|\Delta\Phi_{i-1}| + P_{i-1}^{d_{i-1}}(v_i - v_{i-1})(y_i). \tag{12}$$

Consider $b = P_{i-1}^{d_i}(v_i(x_i) - v_{i-1}(x_i)) = P_{i-1}^{d_i}(v_i - v_{i-1})(x_i) \leq v_i(x_{i-1}) - v_{i-1}(x_{i-1})$ since $x_{i-1} = \arg\max_{s\in S}(v_i - v_{i-1})$. Therefore, $L_i v_i(x_i) - L_{i-1}v_{i-1}(x_i) \leq YMsp(v_i)|\Delta\Phi_{i-1}| + v_i(x_{i-1}) - v_{i-1}(x_{i-1})$. Similarly, $L_i v_i(y_i) - L_{i-1}v_{i-1}(y_i) \geq -YMsp(v_i)|\Delta\Phi_{i-1}| + v_i(y_{i-1}) - v_{i-1}(y_{i-1})$.

Now, by keep expanding, we have: $v_{t+1}(x_t) - v_t(x_t) = L_t v_t(x_t) - L_{t-1}v_{t-1}(x_t) \leq YM\sum_{j=i+1}^{t}\left(sp(v_j)|\Delta\Phi_{j-1}|\right) + (v_{i+1}(x_i) - v_i(x_i))$. Let $t \to \infty$, when $t = \infty$, we know exactly P_f and run Value Iteration algorithm for it, we will have a reward received at one time step $\lim_{t\to\infty}(v_{t+1}(x_t) - v_t(x_t)) = g^*$ which is the optimal average reward [5].

Therefore, $g^* \leq YM\sum_{j=i+1}^{\infty}\left(sp(v_j)|\Delta\Phi_{j-1}|\right) + (v_{i+1}(x_i) - v_i(x_i))$ for any i.

Similarly, we have: $g^* \geq -YM\sum_{j=i+1}^{\infty}\left(sp(v_j)|\Delta\Phi_{j-1}|\right) + (v_{i+1}(y_i) - v_i(y_i))$ for any i.

Denote $e_i = \sum_{j=i+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|)$ represents the total error from the time step i to ∞. This error comes from the fact

that we use the inaccurate matrix at each time step instead of the true one. Now, for any $s \in S$:

$$-YMe_i - (v_{i+1}(x_i) - v_i(x_i)) + (v_{i+1}(y_i) - v_i(y_i)) \leq (v_{i+1}(s) - v_i(s)) - g^* \leq YMe_i + (v_{i+1}(x_i) - v_i(x_i)) - (v_{i+1}(y_i) - v_i(y_i)).$$

Hence, if we use $\infty-$norm, we have

$$\begin{aligned}\|v_{i+1} - v_i - g^*\|_\infty &\leq YMe_i + (v_{i+1}(x_i) - v_i(x_i)) - (v_{i+1}(y_i) - v_i(y_i)),\\ &\leq sp(v_{i+1} - v_i) + YMe_i.\end{aligned}$$

Now, we bound $sp(v_{i+1} - v_i)$. Since (11), (12), we have:

$$\begin{aligned}sp(v_{i+1} - v_i) &= (L_i v_i(x_i) - L_{i-1}v_{i-1}(x_i)) - (L_i v_i(y_i) - L_{i-1}v_{i-1}(y_i)),\\ &\leq 2YMsp(v_i)|\Delta\Phi_{i-1}| + P_{i-1}^{d_i}(v_i - v_{i-1})(x_i) - P_{i-1}^{d_{i-1}}(v_i - v_{i-1})(y_i),\\ &\leq 2YMsp(v_i)|\Delta\Phi_{i-1}| + \max_{s\in S} P_{i-1}^{d_i}(v_i - v_{i-1})(s) - \min_{s\in S} P_{i-1}^{d_{i-1}}(v_i - v_{i-1})(s),\\ &\leq 2YMsp(v_i)|\Delta\Phi_{i-1}| + sp([P_{i-1}^{d_i}/P_{i-1}^{d^{i-1}}](v_i - v_{i-1})),\\ &\leq 2YMsp(v_i)|\Delta\Phi_{i-1}| + \gamma_{i-1}sp(v_i - v_{i-1}) \quad \text{(Proposition 2).}\end{aligned} \tag{13}$$

where $[P1/P2]$ denotes the stacked matrix in which the rows of $P1$ follow the rows of $P2$. Based on the Definition 3, the gamma coefficient of the set of stacked matrices at time step $i-1$ is at most γ_{i-1}. (13) is similar to Proposition 3 except there is an error $2YMsp(v_i)|\Delta\Phi_{i-1}|$.
Since $0 < \gamma_i \leq \gamma = \max(\gamma_{n_0}, \gamma_f) < 1, \forall i \geq n_0$ (Corollary 1), we have:

$$sp(v_{i+1} - v_i) \leq (\prod_{k=n_0}^{i-1}\gamma_k)sp(v_{n_0+1} - v_{n_0}) + 2YM\left(\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right) \text{ for all } i \geq n_0. \tag{14}$$

Then,

$$\begin{aligned}\sum_{i=n_0}^{n-1}\frac{\|v_{i+1} - v_i - g^*\|_\infty}{n} &\leq \frac{1}{n}\sum_{i=n_0}^{n-1}[sp(v_{i+1} - v_i) + YMe_i],\\ \sum_{i=n_0}^{n-1}\frac{\|v_{i+1} - v_i - g^*\|_\infty}{n} &\leq \frac{1}{n}\sum_{i=n_0}^{n-1}\left[(\prod_{k=n_0}^{i-1}\gamma_k)sp\left(v_{n_0+1} - v_{n_0}\right) + YM\left(e_i + 2\left(\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right)\right)\right].\end{aligned}$$

Therefore, we have a upper bound of A which is as follows:

$$E \leq \frac{||v_{n_0} - n_0 g^*||_\infty}{n} + \frac{1}{n}\sum_{i=n_0}^{n-1}\left[\left(\prod_{k=n_0}^{i-1}\gamma_k\right)sp(v_{n_0+1} - v_{n_0}) + YM\left(e_i + 2\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right)\right]. \tag{15}$$

Comments on the Theorem 1. From Theorem 1, if we want to get an integer N so that for $n \geq N$, $E < \varepsilon$, we have to find out an n_0 so that we can bound the last term in the right hand side of (15). If we can run Value Iteration algorithm until time steps n_0, we have some more information about the value vector v. Therefore, we can get a good bound as shown in Theorem 2. Whereas, if we are not able to run Value Iteration algorithm for some steps, we have to find a general bound on all the terms in the right hand side of (15) as described in Theorem 3. As a consequence, we obtain a looser bound on the left hand side. We will see this in Section 4.

Note that, in Theorem 1, we consider general function Φ while in Theorem 2 and Theorem 3, we consider positive non-increasing function Φ.

Theorem 2 (Main Result 2). *Consider a unichain adiabatic-time MDP with S and A_s both finite, $|r(s,a)|$ bounded by a number M. Suppose $0 < \gamma = \max(\gamma_f, \gamma_{n_0}) < 1$ and $\Phi(i)$ is a positive non-increasing function on $[n_0, +\infty)$, then for*

$$n \geq \max\left(n_0, \frac{2}{\varepsilon}\left(||v_{n_0}||_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM\frac{e_{n_0}}{1-\gamma}\right)\right), \tag{16}$$

we guarantee: $E = \left\|\frac{v_n}{n} - g^*\right\|_\infty < \varepsilon$ where $e_i = \sum_{j=i+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|), 2Y = \max_{d\in D}\|P_0 - P_f\|_\infty$, n_0 is the smallest integer satisfying $\frac{2YM\Phi(n_0)}{1-\gamma} < 1$ and $\frac{\left(sp(v_{n_0}) + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma}\right)\Phi(n_0)}{\left(1 - \frac{2YM\Phi(n_0)}{1-\gamma}\right)} \leq \frac{\varepsilon}{2YM}$

Proof. By applying the Theorem 1 with n_0, we have:

$$E \leq \frac{||v_{n_0} - n_0 g^*||_\infty}{n} + \frac{1}{n}\sum_{i=n_0}^{n-1}[(\prod_{k=n_0}^{i-1}\gamma_k)sp(v_{n_0+1}-v_{n_0}) + YM\left(e_i + 2\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right)]$$
$$\leq \frac{||v_{n_0} - n_0 g^*||_\infty}{n} + \frac{sp(v_{n_0+1}-v_{n_0})}{n(1-\gamma)} + \frac{1}{n}\sum_{i=n_0}^{n-1} YM(e_i + 2\sum_{j=n_0+1}^{i}(\gamma^{i-j}sp(v_j)|\Delta\Phi_{j-1}|)))$$

since $0 < \gamma_i \leq \gamma = \max(\gamma_{n_0}, \gamma_f) < 1, \forall i \geq n_0$ (Corollary 1).

Given n_0, $||v_{n_0}||_\infty$ and $sp(v_{n_0+1} - v_{n_0})$ are fixed. Let $y_i = \sum_{j=n_0+1}^{i}(\gamma^{i-j}sp(v_j)|\Delta\Phi_{j-1}|))$. We have:

$$\begin{aligned}
y_{n_0} &= 0 \\
y_{n_0+1} &= sp(v_{n_0+1})\Delta\Phi_{n_0} \\
y_{n_0+2} &= \gamma y_{n_0+1} + sp(v_{n_0+2})\Delta\Phi_{n_0+1} \\
y_{n_0+3} &= \gamma y_{n_0+2} + sp(v_{n_0+3})\Delta\Phi_{n_0+2} \\
&\cdots \\
y_n &= \gamma y_{n-1} + sp(v_n)\Delta\Phi_{i-1} \\
&\cdots
\end{aligned}$$

Then, $(1-\gamma)\sum_{i=n_0}^{\infty} y_i = \sum_{j=n_0+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|)$ or $\sum_{i=n_0}^{\infty} y_i = \frac{e_{n_0}}{1-\gamma}$.

Now, we find conditions on n_0 so that $e_{n_0} = \sum_{j=n_0+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|) \leq \frac{\varepsilon}{2YM}$. In order to do that, we need to bound $sp(v_j)$ because we do not know $sp(v_j)$ ahead of time when $j > n_0$. By the triangle property of seminorm, we get:

$$\begin{aligned}
sp(v_{j+1}) &\leq sp(v_j) + sp(v_{j+1} - v_j) \\
&\leq sp(v_{j-1}) + sp(v_j - v_{j-1}) + sp(v_{j+1} - v_j) \\
&\leq sp(v_{n_0}) + \sum_{i=n_0}^{j} sp(v_{i+1} - v_i)
\end{aligned} \tag{17}$$

From (14), we have:

$$\begin{aligned}
sp(v_{i+1} - v_i) &\leq 2YM\left(\sum_{j=n_0+1}^{i} sp(v_j)|\Delta\Phi_{j-1}|\gamma^{i-j}\right) + \gamma^{j-n_0}sp(v_{n_0+1} - v_{n_0}) \\
&\leq 2YMy_i + \gamma^{j-n_0}sp(v_{n_0+1} - v_{n_0})
\end{aligned}$$

Then, for all $j \geq n_0$,

$$\begin{aligned}
sp(v_{j+1}) &\leq sp(v_{n_0}) + 2YM\sum_{i=n_0}^{j} y_i + sp(v_{n_0+1} - v_{n_0})\sum_{i=n_0}^{j}\gamma^{i-n_0} \\
&\leq sp(v_{n_0}) + 2YM\sum_{i=n_0}^{\infty} y_i + sp(v_{n_0+1} - v_{n_0})\sum_{i=n_0}^{\infty}\gamma^{i-n_0} \\
&\leq sp(v_{n_0}) + 2YM\frac{e_{n_0}}{1-\gamma} + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma}
\end{aligned}$$

By plugging back into e_{n_0}, we get:

$$\begin{aligned}
e_{n_0} &= \sum_{j=n_0+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|) \\
&\leq \left(sp(v_{n_0}) + 2YM\frac{e_{n_0}}{1-\gamma} + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma}\right) \times \sum_{j=n_0+1}^{\infty}|\Delta\Phi_{j-1}|) \\
&\leq \left(sp(v_{n_0}) + 2YM\frac{e_{n_0}}{1-\gamma} + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma}\right)\Phi(n_0)
\end{aligned}$$

since $\Phi(i)$ is non-increasing, $i \geq n_0$, then $|\Delta\Phi_{j-1}| = \Phi(j-1) - \Phi(j)$. In other words,

$$e_{n_0}\left(1 - \frac{2YM\Phi(n_0)}{1-\gamma}\right) \leq \left(sp(v_{n_0}) + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma}\right)\Phi(n_0)$$

Since $\lim_{i\to\infty}\Phi(i)=0$, then there exists n_0 so that:

$$\frac{2YM\Phi(n_0)}{1-\gamma}<1 \tag{18}$$

Then, we only need to run the Value Iteration until the following condition is satisfied.

$$e_{n_0}\leq\frac{\left(sp(v_{n_0})+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}\right)\Phi(n_0)}{\left(1-\frac{2YM\Phi(n_0)}{1-\gamma}\right)}\leq\frac{\varepsilon}{2YM} \tag{19}$$

Now, for $n\geq n_0$,

$$\begin{aligned}E&\leq\frac{||v_{n_0}-n_0g^*||_\infty}{n}+\frac{1}{n}\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+\frac{YM\sum_{i=n_0}^{n-1}e_i}{n}+\frac{1}{n}2YM\frac{e_{n_0}}{1-\gamma}\\&\leq\frac{1}{n}\left(||v_{n_0}-n_0g^*||_\infty+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+2YM\frac{e_{n_0}}{1-\gamma}\right)+\frac{YM(n-n_0)\frac{\varepsilon}{2YM}}{n}\\&\leq\frac{1}{n}\left(||v_{n_0}-n_0g^*||_\infty+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+2YM\frac{e_{n_0}}{1-\gamma}\right)+\frac{\varepsilon}{2}\end{aligned}$$

Let $\frac{1}{n}\left(||v_{n_0}-n_0g^*||_\infty+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+2YM\frac{e_{n_0}}{1-\gamma}\right)\leq\frac{\varepsilon}{2}$, or $n\geq\frac{2}{\varepsilon}\left(||v_{n_0}-n_0g^*||_\infty+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+2YM\frac{e_{n_0}}{1-\gamma}\right)$. Then, for all $n\geq\max(n_0,\frac{2}{\varepsilon}(||v_{n_0}-n_0g^*||_\infty+\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+2YM\frac{e_{n_0}}{1-\gamma}))$, we guarantee $E\leq\varepsilon$.

Comments on the Theorem 2. From Theorem 2, if we run Value Iteration algorithm until step n_0 satisfying the conditions in the theorem, then we can predict that number of steps necessary for Value Iteration algorithm to ensure that $E<\varepsilon$. If running Value Iteration algorithm to find n_0 is not satisfactory then in that case, we provide an alternative on the upper bound without n_0 as shown in the Theorem 3. However, this bound is looser then the bound in Theorem 2.

Theorem 3 (Main Result 3). *Consider a unichain adiabatic-time MDP with S and A_s both finite, $|r(s,a)|$ bounded by a number M. Suppose $0<\gamma=\max(\gamma_f,\gamma_{n_0}),\gamma'=\max(\gamma_f,\gamma_0)<1$ and $\Phi(i)$ is a positive non-increasing function on $[n_0,+\infty)$, then for*

$$n\geq\frac{2}{\varepsilon}\left(2n_0M+||v_0||_\infty+\frac{\varepsilon}{1-\gamma}+\frac{M+(1+\gamma)\left[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'}+(\gamma')^{n_0}(sp(v_0))\right]}{1-\gamma}\right), \tag{20}$$

we guarantee: $E=\left\|\frac{v_n}{n}-g^*\right\|_\infty<\varepsilon$ where $2Y=\max_{d\in D}\|P_0-P_f\|_\infty$, n_0 is the smallest integer satisfying

$$\left[\frac{M}{1-\gamma'}+\gamma'^{(n_0)}sp(v_0)\right]\Phi(n_0)\leq\frac{\varepsilon}{2YM} \tag{21}$$

Proof. By applying the Theorem 1 with n_0, we have:

$$\begin{aligned}E&\leq\frac{||v_{n_0}-n_0g^*||_\infty}{n}+\frac{1}{n}\sum_{i=n_0}^{n-1}\left[\left(\prod_{k=n_0}^{i-1}\gamma_k\right)sp(v_{n_0+1}-v_{n_0})+YM\left(e_i+2\sum_{j=n_0+1}^{i}sp(v_j)|\Delta\Phi_{j-1}|(\prod_{k=j}^{i-1}\gamma_k)\right)\right].\\&\leq\frac{||v_{n_0}-n_0g^*||_\infty}{n}+\frac{1}{n}\frac{sp(v_{n_0+1}-v_{n_0})}{1-\gamma}+\frac{1}{n}\sum_{i=n_0}^{n-1}YM\left(e_i+2\sum_{j=n_0+1}^{i}(\gamma^{i-j}sp(v_j)|\Delta\Phi_{j-1}|)\right).\end{aligned}$$

We have:
$||v_{n_0}-n_0g^*||_\infty\leq||r_{d_{n_0-1}}+P_i^{d_{n_0-1}}v_{n_0-1}-n_0g^*||_\infty\leq||(r_{d_{n_0-1}}-g^*)||_\infty+||P_i^{d_{n_0-1}}(v_{n_0-1}-(n_0-1)g^*)||_\infty\leq||(r_{d_{n_0-1}}-g^*)||_\infty+||(v_{n_0-1}-(n_0-1)g^*)||_\infty\leq\sum_{k=1}^{n_0}||r_{d_{k-1}}-g^*||_\infty+||v_0||_\infty\leq2n_0M+||v_0||_\infty$. since $0<|r_{d_{k-1}}|,g^*<M$
and
$sp(v_{n_0+1}-v_{n_0})=sp(r_{d_{n_0}}+P_{d_{n_0}}v_{n_0}-v_{n_0})\leq sp(r_{d_{n_0}})+sp(P_{d_{n_0}}v_{n_0})+sp(v_{n_0})\leq M+(1+\gamma)sp(v_{n_0})$.
Consider

$$\begin{aligned}sp(v_i)&=sp(r^{d_i}+P_{i-1}^{d_{i-1}}v_{i-1}),\\&\leq\left[sp(r^{d_i})+sp(P_{i-1}^{d_{i-1}}v_{i-1})\right]\leq[M+\gamma_{i-1}sp(v_{i-1})]\quad\text{(Proposition 2)},\\&\leq M\left[1+\sum_{j=1}^{i-1}\prod_{k=1}^{j}\gamma_{i-k}\right]+\prod_{k=1}^{i}\gamma_{i-k}(sp(v_0)),\\&\leq\frac{M(1-(\gamma')^i)}{1-\gamma'}+(\gamma')^i(sp(v_0)),\text{ where }\gamma'=\max(\gamma_0,\gamma_f).\end{aligned} \tag{22}$$

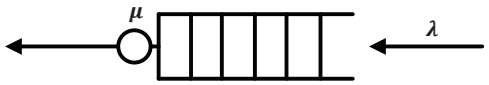

Fig. 3. Queuing Systems

Then,

$$\begin{aligned} sp(v_{n_0+1} - v_{n_0}) &\leq M + (1+\gamma) sp(v_{n_0}), \\ &\leq M + (1+\gamma)\left[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'} + (\gamma')^{n_0}(sp(v_0))\right]. \end{aligned} \tag{23}$$

Let $y_i = \sum_{j=n_0+1}^{i}(\gamma^{i-j} sp(v_j)|\Delta\Phi_{j-1}|))$. Then $\sum_{i=n_0}^{\infty} y_i = \frac{e_{n_0}}{1-\gamma}$ as shown in Theorem 2's proof. Now, we find conditions on n_0 so that $e_{n_0} = \sum_{j=n_0+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|) \leq \frac{\varepsilon}{2YM}$.

Since $sp(v_j) \leq \frac{M(1-(\gamma')^j)}{1-\gamma'} + (\gamma')^j sp(v_0) \leq \frac{M}{1-\gamma'} + \gamma'^{(n_0)} sp(v_0)$, for all $j > n_0$,

$$\begin{aligned} e_{n_0} &\leq \sum_{j=n_0+1}^{\infty}(sp(v_j)|\Delta\Phi_{j-1}|)\left[\frac{M}{1-\gamma'} + (\gamma')^{n_0} sp(v_0)\right] \sum_{j=n_0+1}^{\infty}(|\Delta\Phi_{j-1}|), \\ &\leq \left[\frac{M}{1-\gamma'} + \gamma'^{(n_0)} sp(v_0)\right]\Phi(n_0). \end{aligned} \tag{24}$$

since $\Phi(i)$ is non-increasing, $i \geq n_0$, then $|\Delta\Phi_{j-1}| = \Phi(j-1) - \Phi(j)$. Easily, we can see $\frac{M}{1-\gamma'} + \gamma'^{(n_0)} sp(v_0)$ is bounded. Therefore, there exists n_0 so that $e_{n_0} \leq \left[\frac{M}{1-\gamma'} + \gamma'^{(n_0)} sp(v_0)\right]\Phi(n_0) \leq \frac{\varepsilon}{2YM}$. Then for all $i \geq n_0$, $e_i \leq e_{n_0} \leq \frac{\varepsilon}{2YM}$. Now, for $n \geq n_0$,

$$\begin{aligned} E &\leq \frac{||v_{n_0} - n_0 g^*||_\infty}{n} + \frac{1}{n}\frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + \frac{YM\sum_{i=n_0}^{n-1} e_i}{n} + \frac{1}{n} 2YM \frac{e_{n_0}}{1-\gamma} \\ &\leq \frac{1}{n}\left(2n_0 M + ||v_0||_\infty + \frac{M + (1+\gamma)\left[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'} + (\gamma')^{n_0}(sp(v_0))\right]}{1-\gamma} + \frac{\varepsilon}{1-\gamma}\right) + \frac{YM(n-n_0)\frac{\varepsilon}{2YM}}{n}, \\ &\leq \frac{1}{n}\left(2n_0 M + ||v_0||_\infty + \frac{M + (1+\gamma)\left[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'} + (\gamma')^{n_0}(sp(v_0))\right]}{1-\gamma} + \frac{\varepsilon}{1-\gamma}\right) + \frac{\varepsilon}{2}. \end{aligned}$$

Let $\frac{1}{n}\left(\frac{M+(1+\gamma)\left[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'} + (\gamma')^{n_0}(sp(v_0))\right]}{1-\gamma} + 2n_0 M + ||v_0||_\infty + \frac{\varepsilon}{1-\gamma}\right) \leq \frac{\varepsilon}{2}$, or

$$n \geq \frac{2}{\varepsilon}\left(2n_0 M + ||v_0||_\infty + \frac{\varepsilon}{1-\gamma} + \frac{M + (1+\gamma)[\frac{M(1-(\gamma')^{n_0})}{1-\gamma'} + (\gamma')^{n_0}(sp(v_0))]}{1-\gamma}\right).$$

Then, $E \leq \varepsilon$.

Comments on the Theorem 3. From Theorem 3, we can predict that number of steps necessary for the Value Iteration algorithm to ensure that $E < \varepsilon$ without running Value Iteration algorithm for a while. This is useful for some applications that we would like to have a prediction in advance although the prediction might not be good. Note that (22) and (23) together show the existence of n_0 in Theorem 2.

According to [16], the delta coefficient of a matrix is an upper bound of the second largest eigenvalue modulus λ^*. Thus, the gamma coefficient is also an upper bound of λ^*. Then, the term $\frac{1}{1-\gamma}$ is a upper bound of the relaxation time $t_{rel} = \frac{1}{1-\lambda^*}$ of P_f^d for all $d \in D$. Moreover, the relaxation time is proportional to the the mixing time of P_f^d or the convergence rate of the corresponding Markov chain [17]. Therefore, in any of the theorems above, the convergence rate of the Adiabatic-Time MDP is proportional to the convergence rate of a Markov chain with $P_f^{d_f^*}$. This is reasonable to the intuition that when we let n goes to infinity, the environment stops changing and the Value Iteration process becomes a Markov Chain with the decision rule d_f^*.

4 Application of Adiabatic MDP: Queuing Systems

In this Section we apply the above result to a continuous queueing system with estimated λ.

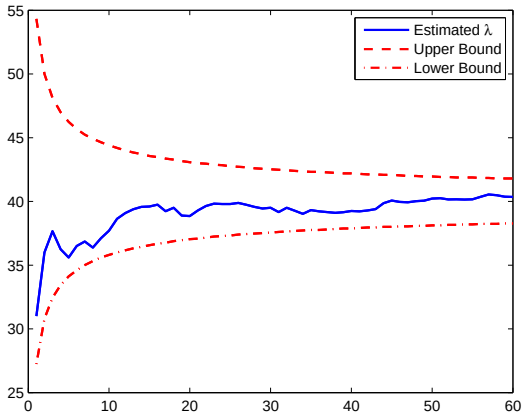

Fig. 4. An example of estimated $\hat{\lambda}_i$ and it bounds for actual $\lambda = 40$

4.1 Queuing Theory Basics

Queuing systems are used in many applications in engineering and management such as transferring packets over a network, waiting line of customers, etc. [18] [19] [20]. A queuing system typically includes a queue and one or many servers as seen in Figure 3. It is characterized by the arrival rate λ (of packets or customers), the service rate μ (of packets or customers), the number of servers and the system's size.

For continuous-time queuing system, the arrival process to the queue is modeled by a Poisson process with the arrival rate λ. The arrival rate λ could be known or estimated by taking the average number of arrival packets per unit time. Similarly, the service or departure process is also modeled by a Poisson process with the service rate μ. The arrival sequence and departure sequence are independent or Markovian. In this paper, we consider queue with only one server. According to Kendall's notation, this queuing system is called $M|M|1|K$ where the first M stands for Markovian arrival process, the second M stands for Markovian departure process, 1 is the number of server, and K is the size of the system including 1 in the server and $K-1$ in the queue. This queuing system can be modeled as a continuous-time Markov Chain with transition matrix in time interval t: $P^t = e^{Qt}$ where Q is generator matrix [21] [22]:

$$Q = \begin{pmatrix} -\lambda & \lambda & 0 & 0 & \dots \\ \mu & -(\mu+\lambda) & \lambda & 0 & \dots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \dots & 0 & \mu & -(\mu+\lambda) & \lambda \\ \dots & 0 & 0 & \mu & -\mu \end{pmatrix}.$$

For the discrete-time queuing system, time is divided into small discrete time slots. Suppose the time slot is small enough that there is only at most one packets coming in. Then, the arrival process can be modeled as a Bernoulli process with parameter p defined as the probability of one packet arrived in a time slot. This probability could be changed over time due to the network's environment or the source. Similarly, the departure process is also modeled as a Bernoulli process with parameter q which is the probability that one packet is served and departs the queue. In this paper, we also consider the discrete-time queuing system with only one server and system's size K. This queuing can be modeled as a Markov Chain with tridiagonal transition matrix as follows assuming that if there is no packet in the queue, a packet coming in will be served immediately.

$$P = \begin{bmatrix} 1-p(1-q) & p(1-q) & 0 & \dots & 0 \\ q(1-p) & pq+(1-p)(1-q) & p(1-q) & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & q(1-p) & pq+(1-p)(1-q) & p(1-q) \\ 0 & \dots & \dots & q(1-p) & 1-q(1-p) \end{bmatrix}$$

A more general analysis of discrete-time queuing system could be found in [23].

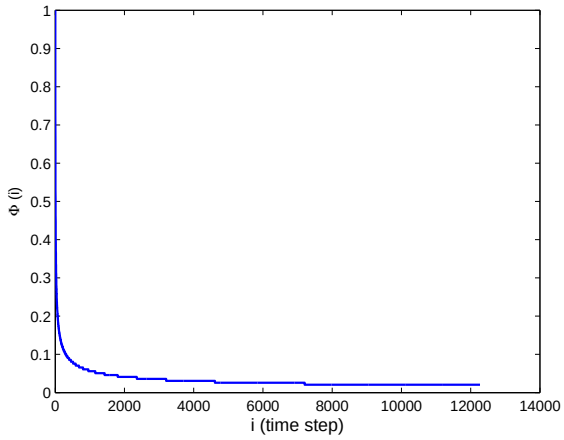

Fig. 5. The $\Phi(\cdot)$ functionfor Simulation Scenario 1

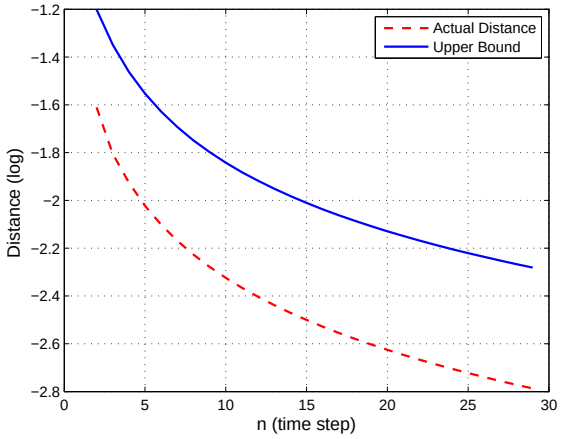

Fig. 6. The actual distance and its upper bound for $\hat{\lambda}_i = (1+a_i)\lambda$ from Theorem 2 (Simulation Scenario 1)

4.2 Application to Continuous-time Queuing System with Arrival Rate Estimation

4.2.1 MDP Formulation for Continuous-time Queuing System

Consider an $M|M|1|K$ queue with fixed arrival rate λ which is unknown. We estimate λ at time $i\Delta t$ denoted as $\hat{\lambda}_i$ and decide packet departure rate, $\mu_i = f(\hat{\lambda}_i)$ based on this estimate as follows:

$$\hat{\lambda}_i = \frac{1}{i\Delta t}\sum_{k=1}^{i} X_k,$$

$$\mu_i = f(\hat{\lambda}_i) = (1+\beta_i)\hat{\lambda}_i,$$

where $X_k \sim \text{Poisson}(\lambda\Delta t)$ is the number of packets in the kth slot of duration Δt and $\beta_i > 0$ is an action we choose from a set of constant numbers. The goal is to find the departure rate that maximizes the reward. Intuitively, high departure rate incurs high costs (high power consumed) and low departure rate may lead to overflow [24]. We define the state s as the number of packets in the queue, therefore, $s \in S = \{0,1,2,\ldots,K-1,K\}$. Let d be the decision rule which is a mapping which maps each state to a value of β_i or $\beta_i(s) = d_i(s)$. The generator matrix in time interval $(i\Delta t,(i+1)\Delta t]$ is shown as following:

$$Q_i = \begin{pmatrix} -\lambda_i & \lambda_i & 0 & 0 & \ldots \\ \mu_i & -(\mu_i+\lambda_i) & \lambda_i & 0 & \ldots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \ldots & 0 & \mu_i & -(\mu_i+\lambda_i) & \lambda \\ \ldots & 0 & 0 & \mu_i & -\mu_i \end{pmatrix}. \tag{25}$$

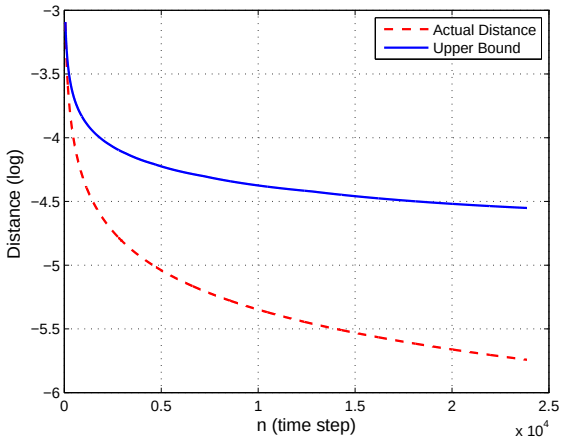

Fig. 7. The actual distance and its upper bound for $\hat{\lambda}_i = (1+a_i)\lambda$ from Theorem 3 (Simulation Scenario 1)

Note that at each time step, we have different Q_i since the arrival rate is estimated over time. The corresponding transition probability matrix $P(i\Delta t,(i+1)\Delta t)$:

$$P_i = P(i\Delta t,(i+1)\Delta t) = e^{Q_i\Delta t}. \tag{26}$$

where $Q_i = \hat{\lambda}_i \begin{pmatrix} -1 & 1 & 0 & 0 & \dots \\ 1+\beta_i(1) & -(2+\beta_i(1)) & 1 & 0 & \dots \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \dots & 0 & 1+\beta_i(K-1) & -(2+\beta_i(K-1)) & 1 \\ \dots & 0 & 0 & 1+\beta_i(K) & -(1+\beta_i(K)) \end{pmatrix}$. Suppose $\beta \in [0,1]$. If we only care about the delay of packets in the queue and the cost of high serving rate, we can set and normalize the reward function: $r(s,\beta) = -\frac{M(s-(K)\beta)^2}{\max_{s,\beta}(s-K\beta)^2}$ or we can define the cost function: $c(s,\beta) = \frac{M(s-K\beta)^2}{\max_{s,\beta}(s-K\beta)^2}$. In this reward function, for a value of s, the cost is high if we use either too high departure rate (high cost) or too low departure rate (overflow cost). Since $s \in [0,K]$, we have to scale $\beta \in [0,1]$ by K.

4.2.2 Simulation Results

Let $Z_i = \sum_{k=1}^{i} X_k \sim \text{Poisson}(i\lambda\Delta t), E[Z_i] = VAR[Z_i] = i\lambda\Delta t$. By applying Chernoff's bound, we get:

$$P(Z_i > (1+a_i)i\lambda\Delta t) < e^{(-i\lambda\Delta t)}\frac{(ei\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}}{((1+a_i)i\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}}$$

Since $e^{(-i\lambda\Delta t)}\frac{(ei\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}}{((1+a_i)i\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}}$ is a decreasing function and goes to 0 when $a_i > 0$. Therefore, given a small α, we can find $a_i = f(i\lambda\Delta t)$ which is a decreasing function of i so that $e^{(-i\lambda\Delta t)}\frac{(ei\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}}{((1+a_i)i\lambda\Delta t)^{(1+a_i)i\lambda\Delta t}} < \alpha$ and $\lim_{i\to\infty} a_i = 0$. Then, $P(\frac{Z_i}{i\Delta_t} > (1+a_i)\lambda) < \alpha$ or

$$P(\hat{\lambda}_i > (1+a_i)\lambda) < \alpha$$

Therefore, with probability α, we have $\hat{\lambda}_i > (1+a_i)\lambda$. Similarly, with probability α, we have $\hat{\lambda}_i < (1-b_i)\lambda$ where $b_i > 0$ can be calculated using Chernoff bound. Both a_i and b_i go to 0 when i goes to infinity. Overall, with probability $1-2\alpha$, $(1-b_i)\lambda \le \hat{\lambda}_i \le (1+a_i)\lambda$. This is illustrated in Figure 4.

To use a decreasing function $\Phi(\cdot)$ for queue, we consider the cases $\hat{\lambda}_i$ follows the upper and lower bounds on $\hat{\lambda}_i$ above to model the dynamic of P_i in (26), i.e. $\hat{\lambda}_i = (1+a_i)\lambda$ and $\hat{\lambda}_i = (1-b_i)\lambda$, respectively. These are two worst cases for $\hat{\lambda}_i$ with probability $1-2\alpha$. We know that $P_i = P_f + \Phi(i)(P_0 - P_f)$, hence

$$|P_i - P_f|_\infty = \Phi(i)|P_0 - P_f|_\infty$$

or $\Phi(i) = \frac{|P_i - P_f|_\infty}{|P_0 - P_f|_\infty}$ can be computed in term of a_i or b_i, respectively.

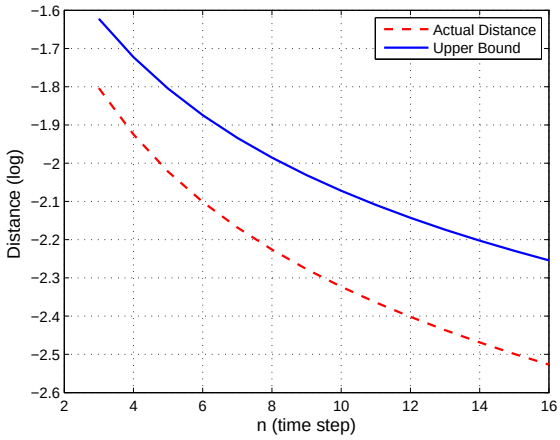

Fig. 8. The actual distance and its upper bound for $\hat{\lambda}_i = (1-b_i)\lambda$ from Theorem 2 (Simulation Scenario 1)

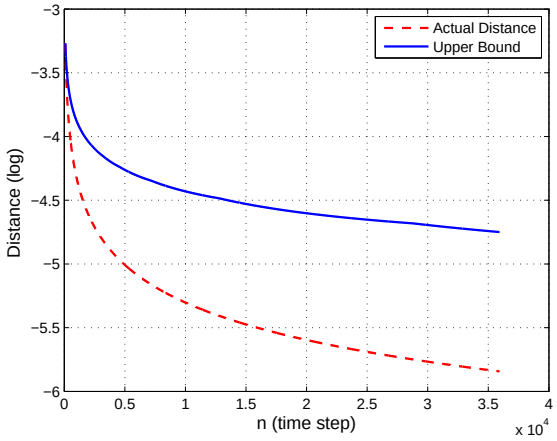

Fig. 9. The actual distance and its upper bound for $\hat{\lambda}_i = (1-b_i)\lambda$ from Theorem 3 (Simulation Scenario 1)

Table 1. The optimal decision rule for continuous-time queuing system example

State	0	1	2	3	4	5	6	7	8	9	10
Action	0.2	0.2	0.2	0.2	0.4	0.4	0.6	0.6	0.8	0.8	0.8

Since a_i, b_i decreases to 0 when i goes to ∞, $\Phi(i)$ decreases to 0 from $\Phi(0) = 1$. Therefore, $\Phi(i)$ is a decreasing function satisfying our conditions on $\Phi(i)$. We now show the bound on the average reward provided by Theorem 1 and Theorem 3 applied to the following scenario.

Simulation Scenario 1. *Let* $\varepsilon = 0.01, \lambda = 40, \Delta_t = 1, \alpha = 0.1, M = 1, K = 10$, $A = \{0.2, 0.4, 0.6, 0.8\}, v_0 = 0e$.

Now, we consider the upper bound the estimated arrival rate $\hat{\lambda}_i = (1 + a_i\lambda)$. The function $\Phi(\cdot)$ is shown on the Figure 5. From Theorem 2, we obtained $n_0 = 1, N = 29$.. From Theorem 3, we obtained $n_0 = 58, N = 23870$. We can see that the value of N from Theorem 2 is much better than the value from Theorem 3 as expected.

Figure 6 and Figure 7 show the actual distance to the optimal reward obtained from the Value Iteration algorithm and its upper bound by Theorem 2 and Theorem 3, respectively. As seen, they are well correlated with each other.

Similarly, for the lower bound on estimated arrival rate $\hat{\lambda}_i = (1-b_i)\lambda$, we have $n_0 = 2, N = 16$ from Theorem 2 and $n_0 = 88, N = 35925$, respectively. As anticipated, the results from Theorem 2 are better than ones from Theorem 3.

The actual distance and its upper bound for each theorem are shown in Figure 8 and Figure 9. As seen, they are also well correlated with each other.

From both Value Iteration for non-stationary environment with adiabatic-time setting and Value Iteration for stationary environment, we have the optimal decision rule as shown in Table 1. The optimal average reward $g^* = -0.0235$. We can see that the optimal policy is reasonable since we need high departure rate for high queue occupancy and vice versa.

Table 2. The reward $r(s,a)$ for discrete-time queuing system

$r(s,a)$		a								
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
s	0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
	1	0.1	0.2	0.3	0.4	0.5	0.4	0.3	0.2	0.1
	2	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9

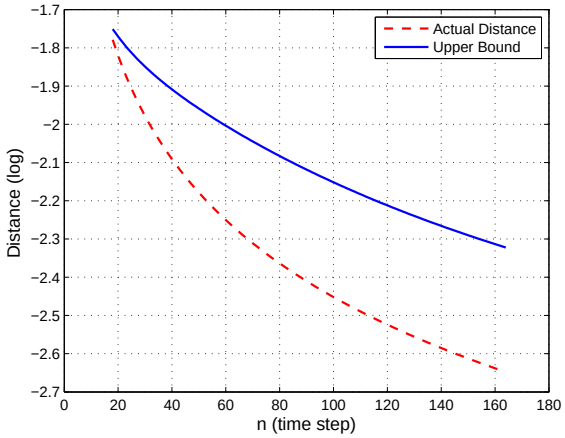

Fig. 10. The actual distance and its upper bound from Theorem 2 (Simulation Scenario 2)

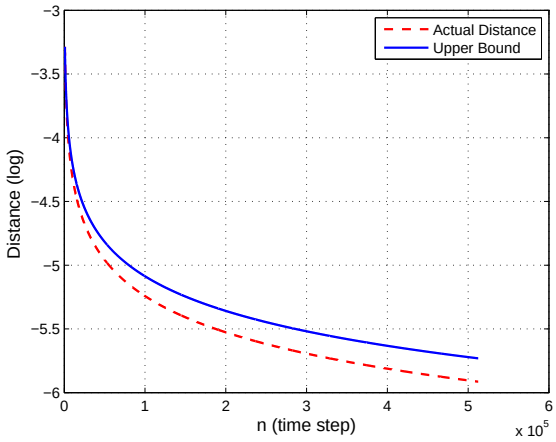

Fig. 11. The actual distance and its upper bound from Theorem 3 (Simulation Scenario 2)

4.3 Application to Discrete-time Queuing System

4.3.1 MDP Formulation

In this section, we show a simple example illustrating the application of our framework to the time-varying underlying environments. Specifically, we consider a discrete-time queuing system of size $K = 2$.

Assume at each time step, there are probabilities p and q that a packet will be arriving and departing the queue, respectively. In this case, the state space $S = \{0,1,2\}$, the time-varying environment is described by changing the values of p over time, while the action is the value of q. The transition matrix has the following form:

$$P = \begin{bmatrix} 1-p(1-q) & p(1-q) & 0 \\ q(1-p) & pq+(1-p)(1-q) & p(1-q) \\ 0 & q(1-p) & 1-q(1-p) \end{bmatrix}$$

Table 3. The optimal decision rule for discrete-time queuing system example

State	0	1	2
Action	0.1	0.5	0.9

Because all entries in the matrix P are linear functions of p, if we set $p_i = \phi(i)p_0 + (1-\phi(i))p_f$ to model the change in the arrival rates, then: $P_i = \Phi(i)P_0 + (1-\Phi(i))P_f$ where $\Phi(i) = \phi(i)$ for all i. Note that $\Phi(\cdot)$ satisfies the conditions for the adiabatic setting. For the reward function, when the number of packets s is small, we should not use high serving rate. Whereas, when the number of packets s is large, we have to use high serving rate to avoid overflow. Therefore, we choose the reward function as shown in Table 2.

4.3.2 Simulation Results

To examine the theoretical results, we run simulation using the following parameters:

Simulation Scenario 2. *Let* $\varepsilon = 0.01, p_0 = 0.4, p_f = 0.6, \phi(i) = \frac{1}{i+1}, v_0 = 0e, A = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$.

From the Theorem 2, we obtained $n_0 = 17, N = 164$. From the Theorem 2, we obtained $n_0 = 811, N = 512509$. Once again, the results from Theorem 2 are better than ones from Theorem 3. Figure 10 and Figure 11 shows the simulated distance to the optimal reward obtained from the Value Iteration algorithm and its upper bound from Theorem 2 and Theorem 3. As seen, they are very correlated.

From both Value Iteration for non-stationary environment with adiabatic-time setting and Value Iteration for stationary environment, we have the optimal decision rule as shown in Table 3. The optimal average reward $g^* = 0.7185$. We can see that the optimal policy is reasonable since we need high departure rate for high queue occupancy and vice versa.

5 Conclusions and Future Work

We provide an analysis framework for studying the Value Iteration algorithm under the adiabatic setting. We provide theoretical bounds on the convergence rate of the Value Iteration algorithm with the average reward objective. Specifically, our work provide a lower bound on the number of time iterations in the Value Iteration algorithm needed in order to ensure that the resulting policy produces an average reward that is ε-close to the optimal average reward value. In the future, we can apply these results to noisy stochastic matrices with parameter estimation. This is more general than the continuous-time queuing systems we considered above.

References

[1] Bellman, R., 1957. *Dynamic Programming*. Priceton University Press.

[2] Howard, R., 1960. *Dynamic Programming and Markov Processes*. MIT Press.

[3] d'Epenoux, F., 1960. "Sur un probleme de production et de stockage dans laléatoire". *Revue Francaise Recherche Oprationelle,* **14**, pp. 3–16.

[4] Derman, C., 1970. *Finite State Markovian Decision Processes*. Academic Press, Inc., Orlando, FL, USA.

[5] Puterman, M. L., 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed. John Wiley & Sons, Inc., New York, NY, USA.

[6] Born, M., and Fock, V., 1928. "Beweis des adiabatensatzes". *Zeitschrift fr Physik,* **51**, pp. 165–180.

[7] Messiah, A., 1962. *Quantum mechanics*, first ed., Vol. 2. John Wiley & Sons.

[8] Kovchegov, Y., 2010. "A note on adiabatic theorem for markov chains". *Statistics and Probability Letters,* **80**, February, pp. 186–190.

[9] Bradford, K., and Kovchegov, Y., 2011. "Adiabatic times for markov chains and applications". *Journal of Statistical Physics,* **143**, pp. 955–969.

[10] Szita, I., Takcs, B., Lrincz, A., and Mahadevan, S., 2002. epsilon-mdps: Learning in varying environments.

[11] Bradford, K., Kovchegov, Y., and Nguyen, T., 2012. "Stable adiabatic times for markov chains". *arXiv preprint arXiv:1207.4733*.

[12] Rosenwald, R., Meyer, D., and Schmitt, H., 2004. "Applications of quantum algorithms to partially observable markov decision processes". In Control Conference, 2004. 5th Asian, Vol. 1, pp. 420–427 Vol.1.

[13] Zacharias, L., Nguyen, T., Kovchegov, Y., and Bradford, K., 2012. "Analysis of adaptive queueing policies via adiabatic approach". In Proceedings of the 2013 International Conference on Computing, Networking and Communications, Network Algorithm and Performance Evaluation Symposium.

[14] Duong, T., Nguyen-Huu, D., and Nguyen, T., 2013. "Adiabatic markov decision process with application to queuing systems". In Information Sciences and Systems (CISS), 2013 47th Annual Conference on, IEEE, pp. 1–6.

[15] Derman, C., and Strauch, R. E., 1966. "A note on memoryless rules for controlling sequential control processes". *The Annals of Mathematical Statistics,* **37**, February, pp. 276–278.

[16] Seneta, E., 1981. *Non-negative Matrices and Markov Chains*. Springer-Verlag.

[17] A.Levin, D., Peres, Y., and L.Wilmer, E., 2008. *Markov Chains and Mixing Times*. American Mathematical Society.

[18] Kleinrock, L., 1976. *Queueing Systems: Theory*. No. v. 1 in A Wiley-Interscience publication. Wiley.
[19] Kleinrock, L., 1976. *Queueing Systems: Computer applications*. Wiley-Interscience Publication. Wiley.
[20] Gautam, N., 2012. *Analysis of Queues: Methods and Applications*. Operations Research Series. Taylor & Francis.
[21] Lawler, G., 1995. *Introduction to stochastic processes*. Chapman and Hall/CRC Probability Series. Chapman & Hall/CRC.
[22] Bremaud, P., 1999. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*. Texts in Applied Mathematics. Springer.
[23] Briem, U., Theimer, T., and Kröner, H., 1991. "A general discrete-time queueing model: analysis and applications". In Proc. ITC, Vol. 13, pp. 13–19.
[24] Morrison, J. A., 1980. "Analysis of some overflow problems with queuing". *Bell System Technical Journal,* **59**(8), pp. 1427–1462.

Appendix A: Average Reward Criteria

In this section, we show some previous results on Average Reward Criteria for stationary environment. Therefore, we have three following assumptions:

Assumption 1. *The reward $r(s,a)$ and the probability $p(j|s,a)$ are stationary.*

Assumption 2. *The reward $r(s,a)$ is bounded: $|r(s,a)| \leq M < \infty \quad \forall a \in A_s, s \in S$*

Assumption 3. *The set of states S is finite*

Note that the stationary probability $p(j|s,a)$ condition is only necessary for this Section where the results for stationary environment are shown.

Definition 4 (Average Reward). *Let $v_{N+1}^{\pi}(s) = E_s^{\pi}\left[\sum_{t=1}^{N} r(s_t,a_t)\right]$ be the total reward we get at the time step $N+1$ starting from the state s under the policy π, where s_t, a_t are the state and action at time step t. Then, the average reward $g^{\pi}(s)$ is defined as follows:*

$$g^{\pi}(s) = \lim_{N\to\infty} \frac{1}{N} v_{N+1}^{\pi}(s) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} P_{\pi}^{n-1} r_{d_n}(s)$$

where d_n is the decision rule at the time step n under the policy π.

Proposition 5 ([5]). *Let S be countable, $\pi = d^{\infty}, P_d^{\infty} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} (P^d)^{n-1}$ is stochastic, then $g^{d^{\infty}}(s)$ exists and:*

$$g^{d^{\infty}}(s) = \lim_{N\to\infty} \frac{1}{N} v_{N+1}^{d^{\infty}}(s) = P_d^{\infty} r^d(s)$$

Obviously, for finite S, this proposition holds.

Proposition 6 ([5]). *Suppose P_d^{∞} is stochastic. Then if j and k are in the same closed recurrent class, $g(j) = g(k)$. Furthermore, if the chain is irreducible and aperiodic or has a single recurrent class and may has some transient states, $g(s)$ is constant function.*

Unichain Markov Decision Process satisfies these conditions since it only has one single recurrent class plus possibly empty set of transient states.

Appendix B: Span-seminorm and its properties

Definition 5 (Span Seminorm). *The Span Seminorm of a vector $v \in V$ is defined as follows:*

$$sp(v) = \max_{s\in S} v(s) - \min_{s\in S} v(s)$$

where V is the vector space and s is state.

This seminorm has following properties:

1) $sp(v) \geq 0$
2) $sp(u+v) \leq sp(u) + sp(v), \quad \forall u, v \in V$
3) $sp(kv) = |k| sp(v) \quad \forall k \in \mathbb{R}, v \in V$
4) $sp(v+ke) = sp(v) \quad \forall k \in \mathbb{R}$, where e is the constant vector $e = [11\ldots1]^T$.
5) $sp(v) = sp(-v)$
6) $sp(v) \leq 2\|v\|$

Note that δ_d is called Hajnal measure or delta coefficient of P^d [16]. It is an upper bound on the second largest eigenvalue modulus (SLEM) $|\lambda_2|$ of P^d , and $\delta_d \leq 1 - \sum_{j\in S} \min_{s\in S} P^d(j|s)$

Appendix C: Additional Details on Proof

Proposition 4 (The bound on gamma coefficients). *Given* $P_i = \Phi(i)P_0 + (1-\Phi(i))P_f, \Phi(i) \in [0,1]$,

$$\gamma_i \leq \Phi(i)\gamma_0 + (1-\Phi(i))\gamma_f$$

Proof. Since $P_i = \Phi(i)P_0 + (1-\Phi(i))P_f, \Phi(i) \in [0,1]$, for all j,s,a,s',a' we have:

$$\min\{p_i(j|s,a), p_i(j|s',a')\} \geq \Phi(i)\min\{p_0(j|s,a), p_0(j|s',a') + (1-\Phi(i))\min\{p_f(j|s,a), p_f(j|s',a')$$

Therefore:

$$\sum_{j\in S}\min\{p_i(j|s,a), p_i(j|s',a') \geq \Phi(i)\sum_{j\in S}\min\{p_0(j|s,a), p_0(j|s',a') + (1-\Phi(i))\sum_{j\in S}\min\{p_f(j|s,a), p_f(j|s',a')$$

or:

$$1-\sum_{j\in S}\min\{p_i(j|s,a), p_i(j|s',a') \leq \Phi(i)(1-\sum_{j\in S}\min\{p_0(j|s,a), p_0(j|s',a')) + (1-\Phi(i))(1-\sum_{j\in S}\min\{p_f(j|s,a), p_f(j|s',a'))$$

In other words, $\gamma_i \leq \Phi(i)\gamma_0 + (1-\Phi(i))\gamma_f$

Corollary 1. *Given* $P_i = \Phi(i)P_0 + (1-\Phi(i))P_f$ *with non-increasing function* $\Phi(i) > 0, \Phi(i) \in [0,1]$ *for any* $n_0 \leq i$

$$\gamma_i \leq \max(\gamma_{n_0}, \gamma_f) \tag{27}$$

Proof. We have:

$$\begin{aligned}
P_i &= \Phi(i)P_0 + (1-\Phi(i))P_f \\
&= P_f + \Phi(i)(P_0 - P_f) \\
&= P_f + \frac{\Phi(i)}{\Phi(n_0)}(\Phi(n_0)P_0 - \Phi(n_0)P_f) \\
&= P_f + \frac{\Phi(i)}{\Phi(n_0)}(\Phi(n_0)P_0 + (1-\Phi(n_0))P_f - P_f) \\
&= P_f + \frac{\Phi(i)}{\Phi(n_0)}(P_{n_0} - P_f)
\end{aligned} \tag{28}$$

Let $0 < \Phi'(i) = \frac{\Phi(i)}{\Phi(n_0)} \leq 1$ since $\Phi(\cdot)$ is a positive non-increasing function. Then, $P_i = P_f + \Phi'(i)(P_{n_0} - P_f) = \Phi'(i)P_{n_0} + (1-\Phi'(i))P_f$. By applying the Proposition 4, we have:

$$\gamma_i \leq \Phi(i)\gamma_{n_0} + (1-\Phi(i))\gamma_f \leq \max(\gamma_{n_0}, \gamma_f)$$

List of Figure Captions

List of Table Captions