AN ABSTRACT OF THE THESIS OF

Thai Duong for the degree of Master of Science in
Electrical and Computer Engineering presented on June 10, 2013.

Title: Adiabatic Markov Decision Process: Convergence of Value Iteration Algorithm

Abstract approved: ________________________________

Thinh P. Nguyen

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 thesis, 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.
Adiabatic Markov Decision Process: Convergence of Value Iteration Algorithm

by

Thai Duong

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 10, 2013
Commencement June 2013
Master of Science thesis of Thai Duong presented on June 10, 2013.

APPROVED:

__________________________
Major Professor, representing Electrical and Computer Engineering

__________________________
Director of the School of Electrical Engineering and Computer Science

__________________________
Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

__________________________
Thai Duong, Author
ACKNOWLEDGEMENTS

I would like to thank my advisor Prof. Thinh Nguyen for his extraordinary guidance and support. I also thank my committee members Prof. Bella Bose, Prof. Raviv Raich, Prof. Prasad Tadepalli and Prof. Yevgeniy Kovchegov for serving on my committee and reviewing this work.

I also thank my colleagues in my group for their useful and interesting discussions. I thank all my friends in Oregon State University and in Corvallis for helping me with my life in America.

Finally, I would like to express my special thanks to my parents, my sister and my fiance for their supports and encouragements.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Markov Decision Process</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Non-stationary Environment</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Related Work</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Overview</td>
<td>5</td>
</tr>
<tr>
<td>2 Mathematical Preliminaries</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Markov Decision Process</td>
<td>7</td>
</tr>
<tr>
<td>2.1.1 Definitions</td>
<td>7</td>
</tr>
<tr>
<td>2.1.2 Span Seminorm</td>
<td>9</td>
</tr>
<tr>
<td>2.1.3 Average Reward Criteria</td>
<td>11</td>
</tr>
<tr>
<td>2.1.4 Value Iteration</td>
<td>13</td>
</tr>
<tr>
<td>3 Adiabatic Markov Decision Process</td>
<td>15</td>
</tr>
<tr>
<td>3.1 Preliminaries on Adiabatic-Time Evolution</td>
<td>15</td>
</tr>
<tr>
<td>3.2 Problem Formulation</td>
<td>18</td>
</tr>
<tr>
<td>3.2.1 Adiabatic MDP Model</td>
<td>18</td>
</tr>
<tr>
<td>3.2.2 Convergence Rate of Adiabatic MDP</td>
<td>21</td>
</tr>
<tr>
<td>3.3 Main Results</td>
<td>24</td>
</tr>
<tr>
<td>4 Application of Adiabatic MDP: Queuing Systems</td>
<td>39</td>
</tr>
<tr>
<td>4.1 Queuing Theory Basics</td>
<td>39</td>
</tr>
<tr>
<td>4.2 Application to Continuous-time Queuing System with Arrival Rate Estimation</td>
<td>41</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2.1 MDP Formulation for Continuous-time Queuing System</td>
<td>41</td>
</tr>
<tr>
<td>4.2.2 Simulation Results</td>
<td>43</td>
</tr>
<tr>
<td>4.3 Application to Discrete-time Queuing System</td>
<td>47</td>
</tr>
<tr>
<td>4.3.1 MDP Formulation</td>
<td>47</td>
</tr>
<tr>
<td>4.3.2 Simulation Results</td>
<td>50</td>
</tr>
<tr>
<td>5 Conclusions and Future Work</td>
<td>52</td>
</tr>
<tr>
<td>Bibliography</td>
<td>53</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The classic Value Iteration</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>The Value Iteration in an adiabatic setting</td>
<td>23</td>
</tr>
<tr>
<td>4.1</td>
<td>An example of estimated $\hat{\lambda}_i$ and its bounds for actual $\lambda = 40$</td>
<td>44</td>
</tr>
<tr>
<td>4.2</td>
<td>The $\Phi(\cdot)$ function for the Simulation Scenario 1</td>
<td>45</td>
</tr>
<tr>
<td>4.3</td>
<td>The actual distance and its upper bound for $\hat{\lambda}_i = (1 + a_i)\lambda$ from Theorem 2</td>
<td>46</td>
</tr>
<tr>
<td>4.4</td>
<td>The actual distance and its upper bound for $\hat{\lambda}_i = (1 + a_i)\lambda$ from Theorem 3</td>
<td>47</td>
</tr>
<tr>
<td>4.5</td>
<td>The actual distance and its upper bound for $\hat{\lambda}_i = (1 + b_i)\lambda$ from Theorem 2</td>
<td>48</td>
</tr>
<tr>
<td>4.6</td>
<td>The actual distance and its upper bound for $\hat{\lambda}_i = (1 + b_i)\lambda$ from Theorem 3</td>
<td>49</td>
</tr>
<tr>
<td>4.7</td>
<td>The actual distance and its upper bound from Theorem 2</td>
<td>50</td>
</tr>
<tr>
<td>4.8</td>
<td>The actual distance and its upper bound from Theorem 3</td>
<td>51</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>The optimal decision rule for continuous-time queuing system example</td>
</tr>
<tr>
<td>4.2</td>
<td>The optimal decision rule for discrete-time queuing system example</td>
</tr>
<tr>
<td>4.3</td>
<td>The reward $r(s,a)$ for discrete-time queuing system</td>
</tr>
</tbody>
</table>
Chapter 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 based 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], 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 thesis, 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] [7] [8] [9]. Formally, the transition probability matrix $P^d_i$ at time step $i$ induced by decision rule $d$ is determined by:

$$P^d_i = \Phi(i) P^d_0 + (1 - \Phi(i)) P^d_f, \quad \forall d$$

(1.1)

where $P^d_0$ and $P^d_f$ are the transition probability matrices induced by the decision rule $d$ at time step 0 and $\infty$, and $\Phi(\cdot)$ 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, $P^d_i$ 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 $P^d_0$, then its dynamic is characterized by $\Phi(\cdot)$ and converges to $P^d_f$ 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 $P^d_0$. Thus, the actions/decisions are made based on inaccurate knowledge of the environment. Over time, the estimations of the environmental param-
eters 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 in varying environment is presented by Szita et al. in [10]. It shows that Q-learning can return a sub-optimal decision rule/policy in a varying environments.

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].

For queuing application, an adiabatic approach to analysis of adaptive queuing policies was proposed in [12]. 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 queuing system.

1.4 Overview

In this thesis, we study the followings:

- Markov Decision Process in non-stationary environment modeled as an adiabatic evolution.
- Value Iteration Algorithm for Markov Decision Process in non-stationary environments.
- Upper bound on the distance between the actual average reward in Value Iteration and the optimal average reward (main result).
- The necessary time needed for the Value Iteration to converge (main result).
- Application to an M/M/1/K continuous-time queue with estimated arrival rate and a discrete-time queue with changing arrival rate.
The Chapter 2 provides some background on the theory of Markov Decision Process, Value Iteration which are necessary for the development of our results. In Chapter 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 Chapter. Chapter 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 Chapter 5.
Chapter 2: Mathematical Preliminaries

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

2.1 Markov Decision Process

2.1.1 Definitions

Markov Decision Process is a stochastic which is modeled as follows [5]:

**Definition 1** (Markov Decision Process (MDP)). *We define the following components of an Markov Decision Process:*

- **State** \( s \in S \) where \( S \) is the state space.
- **Action** \( a \in A_s \) where \( A_s \) is the set of actions for state \( s \). Let \( A = \bigcup_{s \in S} A_s \)
- **Reward** \( r_t(s_t, a_t) \) is the reward received by taking action \( a_t \) in state \( s_t \) at the time step \( t \).
- **Transition probability** \( p_t(s_{t+1}|s_t, a) \) is the probability of transiting from state \( s_t \) to state \( s_{t+1} \) by taking action \( a \).
• Decision rule $d$ is a mapping from the set of state $S$ to $A$ where $s$ is the current state: $d : S \to A$. Let $D$ be the set of all possible decision rules.

• $\pi$ is a policy which is a sequence of decision rule that we apply it to MDP at each time step: $\pi = \{d_1, d_2, \ldots, d_n, \ldots\}$. Besides, the stationary policy $\pi = d^\infty$ means that $\pi = \{d, d, \ldots, d, \ldots\}$

• $\Pi$ is 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 [13]. Therefore, we only consider Markovian Deterministic policy.

\textbf{Definition 2} (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 $\pi^*$ 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:

\[
V_{\text{dis}}^\pi(s) = E_s \left\{ \sum_{t=1}^{\infty} \alpha^t r_t(s_t, a_t) \right\},
\]

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:

\[
V_{\text{ave}}^\pi(s) = \lim_{N \to \infty} \frac{V_N^\pi(s)}{N},
\]

where

\[
v_N^\pi(s) = E_s \left\{ \sum_{t=1}^{N} r(s_t, a_t) \right\}.
\]

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 \(\alpha^n\) which we can easily find the convergence rate. Therefore, in this thesis, we only consider the average reward function criteria presented in section 2.1.3.

### 2.1.2 Span Seminorm

The span seminorm defined below is used to evaluate the convergence rate of value iteration for MDP using Average Reward Criterion ([5]).

**Definition 3** (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:

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

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

\[ sp(P^d v) \leq \delta_d sp(v) \]

where:

\[ \delta_d = 1 - \min_{s,u \in S \times S} \sum_{j \in S} \min \{ P^d(j|s), P^d(j|u) \} \]

\( P^d \) is the transition matrix corresponding to the decision rule \( d \)

Furthermore, \( 0 \leq \delta_d \leq 1 \), and there exists a \( v \in V \) such that \( sp(P^d v) = \delta_d sp(v) \)
Note that $\delta_d$ is called Hajnal measure or delta coefficient of $P^d$ [14]. It is an upper bound on the second largest eigenvalue modulus (SLEM) $|\lambda_2|$ of $P^d$, and:

$$\delta_d \leq 1 - \sum_{j \in S} \min_{s \in S} P^d(j|s)$$

2.1.3 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: $|r(s,a)| \leq M < \infty \quad \forall a \in A, s \in S$

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

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

**Definition 4** (Average Reward). Let

$$v^\pi_{N+1}(s) = E^\pi \sum_{t=1}^{N} r(s_t,a_t)$$

be the total reward we get at the time step $N + 1$ starting from the state $s$ under the policy $\pi$, 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:

\[ g^\pi(s) = \lim_{N \to \infty} \frac{1}{N} v^\pi_{N+1}(s) = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} P^{\pi-1}_n r_{d_n}(s) \]

where \( d_n \) is the decision rule at the time step \( n \) under the policy \( \pi \).

**Proposition 2 ([5]).** Let \( S \) be countable, \( \pi = d^\infty, P^\infty_d = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} (P^d)^n \) is stochastic, then \( g^{d^\infty}(s) \) exists and:

\[ g^{d^\infty}(s) = \lim_{N \to \infty} \frac{1}{N} v_{N+1}^{d^\infty}(s) = P^\infty_d r^d(s) \]

Obviously, for finite \( S \), this proposition holds.

**Proposition 3 ([5]).** Suppose \( P^\infty_d \) 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.
2.1.4 Value Iteration

The Value Iteration algorithm is an iterative algorithm for finding an $\varepsilon$-optimal policy for the infinite-horizon MDP. More precisely, given an $\varepsilon$, the Value Iteration algorithm guarantees to produce a reward value within an $\varepsilon$ 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 5 (The Value Iteration, [5]).** The algorithm for the Value Iteration with Average Reward Criteria is shown below:

- Choose any initial reward vector $v_0$, for a given $\varepsilon > 0$. Let $n = 0$.

- For each $s \in S$, we have: $v_{n+1}(s) = \max_{a \in A} \{r(s, a) + \sum_{j \in S} p(j|s, a)v_n(j)\}$.

- 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 $\varepsilon$-optimal policy approaches to an optimal policy as $\varepsilon$ reduces to zero when the number of iterations goes to infinity.

**Definition 6 (Gamma coefficient).** The gamma coefficient is defined as follows [5]:

$$
\gamma = \max_{s \in S, a \in A, s' \in S, a' \in A} 1 - \sum_{j \in S} \min\{p(j|s, a), p(j|s', a')\}
$$

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

**Proposition 4.** Let $v \in V$ and $d \in D$ where $D$ is the set of decision rule, then:

$$
sp(p^d v) \leq \gamma sp(v)
$$

**Proposition 5 (The convergence of Value Iteration Algorithm, [5]).** For unichain MDPs, we have:

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

where

$$
\gamma = \max_{s \in S, a \in A, s' \in S, a' \in A} 1 - \sum_{j \in S} \min\{p(j|s, a), p(j|s', a')\}
$$

Then if $\gamma < 1$, the Value Iteration algorithm will stop after a finite step $n$. \qed
Chapter 3: Adiabatic Markov Decision Process

This chapter 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:

\[
P_i = \Phi(i)P_0 + (1 - \Phi(i))P_f
\]

\[
= P_f + \Phi(i)(P_0 - P_f)
\]

(3.1)

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:

- \(\Phi(i) : [0, +\infty) \rightarrow [0, 1]\)
- \(\Phi(0) = 1\)
• \( \lim_{i \to \infty} \Phi(i) = 0 \)

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 thesis, we suppose that the \( \Phi \) is given.

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

**Proposition 6** (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 \( P_i = \Phi(i)P_0 + (1 - \Phi(i))P_f \), \( \Phi(i) \in [0, 1] \), we have:

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

Therefore:

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

In another word,
\[ \gamma \leq \Phi(i)\gamma_0 + (1 - \Phi(i))\gamma_f \tag{3.2} \]

**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 \( n_0 \):
\[ \gamma \leq \max(\gamma_{n_0}, \gamma_f) \tag{3.3} \]

**Proof:** We have:

\[ P_i = \Phi(i)P_0 + (1 - \Phi(i))P_f \]

\[ = P_f + \Phi(i)(P_0 - P_f) \]
\[ = P_f + \Phi(n_0)\frac{\Phi(i)}{\Phi(n_0)}(P_0 - P_f) \]
\[ = P_f + \Phi(n_0)\frac{\Phi(i)}{\Phi(n_0)}(\Phi(n_0)P_0 - \Phi(n_0)P_f) \]
\[ = P_f + \Phi(n_0)\frac{\Phi(i)}{\Phi(n_0)}(\Phi(n_0)P_0 + (1 - \Phi(n_0))P_f - P_f) \]
\[ = P_f + \Phi(n_0)\frac{\Phi(i)}{\Phi(n_0)}(P_{n_0} - P_f) \tag{3.4} \]
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 6, we have:

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

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 $\varepsilon$-optimal stationary policy in an offline manner using a number of iterations, assuming a stationary environment such that every stationary policy $\pi$ 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 thesis, 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 $\epsilon$-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:

$$P^d_i = \Phi(i)P^d_0 + (1 - \Phi(i))P^d_f = P^d + \Phi(i)(P^d_0 - P^d_f), \quad \forall d$$

(3.5)

where $\Phi$ is a function such that:

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

(3.6)

$\Phi(\cdot)$ characterizes the change in the environment at the iteration $i$ of the Value Iteration algorithm, and $P^d_i$ 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 $P^d_f$ to ensure a well-defined reward. This can be seen as $\lim_{i \rightarrow \infty} P^d_i = P^d_f$. In this thesis, 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 an initial rate of $\lambda_0$ to a final rate of $\lambda_f$. 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 $P^d_i$ 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 $P^d_f$. We will discuss these two examples in more detail in the later section.

### 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 $P^d_f$ for any decision rule $d$. In addition, there exists an optimal decision rule $d^*$ corresponding to a transition probability matrix $P^{d^*}_f$ 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:

$$g^* = P^{d^*}_f r^{d^*}_f = \pi^{d^*}_p r^{d^*}_f e,$$
where by Cesaro mean,

\[ P_{d^*}^\infty = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} (P_{f}^{d^*})^{n-1} = \lim_{n \to \infty} (P_{f}^{d^*})^{n} = \begin{bmatrix} d^*_f \\ \pi_{d+f}^* \\ \pi_{d+f}^* \\ \pi_{d+f}^* \\ \cdots \end{bmatrix}, \]

\( \pi_{d+f}^* \) is the stationary distribution for \( P_{f}^{d^*} \), \( e = [1 \ldots 1]^T \).

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 thesis 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 \( \forall n > N \),

\[ E = \left\| \frac{v_n}{n} - g^* \right\|_{\infty} \leq \varepsilon \quad (3.7) \]

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 3.1 and 3.2. 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 $i$.

Figure 3.1: The classic Value Iteration

**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 $P_f^d$. 

Figure 3.2: The Value Iteration in an adiabatic setting
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 $0 < \gamma = \max(\gamma_f, \gamma_n) < 1$, then

$$E \leq \frac{||v_{n_0} - n_0g^*||_{\infty}}{n} + \frac{1}{n} \sum_{i=n_0}^{n-1} \left( \prod_{k=n_0}^{i-1} \gamma_k \right) sp(v_{n_0+1} - v_{n_0})$$

$$+ YM \left( e_i + 2 \sum_{j=n_0+1}^{i} sp(v_j)|\Delta \Phi_{j-1}|(\prod_{k=j}^{i-1} \gamma_k) \right),$$

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

Proof: From (3.5), for any $d \in D$ we have:

$$\Rightarrow |P_i^d - P_{i-1}^d| \leq |\Phi(i) - \Phi(i-1)||P_0^d - P_f^d|,$$

$$\leq |\Phi(i) - \Phi(i-1)||P_0^d - P_f^d|,$$

$$\leq 2Y|\Delta \Phi_{i-1}|,$$

where fixed $2Y = \max_{d \in D} ||P_0 - P_f||_{\infty}$. \hfill (3.9)
Consider $E = \frac{v_n}{n} - g^*\infty$:

\[
E = \frac{v_n}{n} - g^*\infty, \quad (3.10)
\]

\[
= \frac{v_n - ng^*}{n}\infty,
\]

\[
= \sum_{i=n_0}^{n-1} \left( \frac{v_{i+1} - v_i - g^*}{n} \right) + \frac{v_{n_0} - n_0 g^*}{n}\infty,
\]

\[
\leq \sum_{i=n_0}^{n-1} \frac{\|v_{i+1} - v_i - g^*\|\infty}{n} + \frac{\|v_{n_0} - n_0 g^*\|\infty}{n}.
\]

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 $L_{i-1} v_{i-1}(x_i) \geq L^d_{i-1} v_{i-1}(x_i)$

\[
L_i v_i(x_i) - L_{i-1} v_{i-1}(x_i) \leq L^d_i v_i(x_i) - L^d_{i-1} v_{i-1}(x_i),
\]

\[
= (r_{di}(x_i) + P^d_i v_i(x_i)) - (r_{di}(x_i) + P^d_{i-1} v_{i-1}(x_i)),
\]

\[
= P^d_i v_i(x_i) - P^d_{i-1} v_{i-1}(x_i),
\]

\[
= (P^d_i - P^d_{i-1}) v_i(x_i) + P^d_{i-1} (v_i - v_{i-1})(x_i).
\]

Let $\alpha_i = \arg\max_{s \in S} (v_i(s))$, $\beta_i = \arg\min_{s \in S} (v_i(s))$, $\Delta P^d_i = P^d_i - P^d_{i-1}$. We have:

\[
a = \Delta P^d_i v_i(x_i) = (\Delta P^d_i(x_i, \cdot)) v_i = (\Delta P^d_i(x_i, \cdot))(v_i - v_i(\beta_i)) e
\]

where $e = [11 \ldots 1]^T$ 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:

\[
\begin{align*}
a &= \sum_{s \in S} \Delta P_{i}^{d_{i}}(x_{i}, s)(v_{i}(s) - v_{i}(\beta_{i})), \\
&\leq \sum_{\Delta P_{i}^{d_{i}}(x_{i}, s) \geq 0} \Delta P_{i}^{d_{i}}(x_{i}, s)(v_{i}(s) - v_{i}(\beta_{i})), \\
&\leq (v_{i}(\alpha_{i}) - v_{i}(\beta_{i})) \sum_{\Delta P_{i}^{d_{i}}(x_{i}, s) \geq 0} \Delta P_{i}^{d_{i}}(x_{i}, s), \\
&\leq (v_{i}(\alpha_{i}) - v_{i}(\beta_{i})) \frac{||P_{i}^{d_{i}} - P_{i-1}^{d_{i}}||_{\infty}}{2}, \\
&\leq Y|\Delta \Phi_{i-1}|sp(v_{i}) \quad \text{since (3.9)}.
\end{align*}
\]

Then, we have:

\[
L_{i}v_{i}(x_{i}) - L_{i-1}v_{i-1}(x_{i}) \leq YMsp(v_{i})|\Delta \Phi_{i-1}| + P_{i}^{d_{i}}(v_{i} - v_{i-1})(x_{i})). \quad (3.11)
\]

Similarly,

\[
L_{i}v_{i}(y_{i}) - L_{i-1}v_{i-1}(y_{i}) \geq -YMsp(v_{i})|\Delta \Phi_{i-1}| + P_{i}^{d_{i}-1}(v_{i} - v_{i-1})(y_{i}). \quad (3.12)
\]

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:

\[
v_{t+1}(x_{t}) - v_{t}(x_{t}) = L_{t}v_{t}(x_{t}) - L_{t-1}v_{t-1}(x_{t}) \leq YM\sum_{j=t}^{t+1} (sp(v_{j})|\Delta \Phi_{j-1}|) + (v_{t+1}(x_{t}) -
\]
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,

$$g^* \leq YM \sum_{j=i+1}^{\infty} sp(v_j) |\Delta \Phi_{j-1}| + (v_{i+1}(x_i) - v_i(x_i))$$

for any $i$.

Similarly, we have:

$$g^* \geq -YM \sum_{j=i+1}^{\infty} sp(v_j) |\Delta \Phi_{j-1}| + (v_{i+1}(y_i) - v_i(y_i))$$

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

$$-YM e_i - (v_{i+1}(x_i) - v_i(x_i)) + (v_{i+1}(y_i) - v_i(y_i))$$

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

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

$$\|v_{i+1} - v_i - g^*\|_{\infty} \leq YM e_i + (v_{i+1}(x_i) - v_i(x_i)) - (v_{i+1}(y_i) - v_i(y_i)),$$

$$\leq sp(v_{i+1} - v_i) + YM e_i.$$
Now, we bound $sp(v_{i+1} - v_i)$. Since (3.11), (3.12), we have:

\[
sp(v_{i+1} - v_i) = (L_i v_i(x_i) - L_{i-1} v_{i-1}(x_i)) - (L_i v_i(y_i) - L_{i-1} v_{i-1}(y_i)),
\]

\[
\leq 2YMsp(v_i)|\Delta \Phi_{i-1}| + p^{d_i}_{i-1}(v_i - v_{i-1})(x_i) - p^{d_{i-1}}_{i-1}(v_i - v_{i-1})(y_i),
\]

\[
\leq 2YMsp(v_i)|\Delta \Phi_{i-1}| + \max_{s \in S} p^{d_i}_{i-1}(v_i - v_{i-1})(s) - \min_{s \in S} p^{d_{i-1}}_{i-1}(v_i - v_{i-1})(s),
\]

\[
\leq 2YMsp(v_i)|\Delta \Phi_{i-1}| + sp([p^{d_i}_{i-1}/p^{d_{i-1}}_{i-1}](v_i - v_{i-1})),
\]

\[
\leq 2YMsp(v_i)|\Delta \Phi_{i-1}| + \gamma_{i-1} sp(v_i - v_{i-1}) \quad \text{(Proposition 4).} \tag{3.13}
\]

where $[P_1/P_2]$ denotes the stacked matrix in which the rows of $P_1$ follow the rows of $P_2$. Based on the Definition 6, the gamma coefficient of the set of stacked matrices at time step $i - 1$ is at most $\gamma_{i-1}$. (3.13) is similar to Proposition 5 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:

\[
sp(v_{i+1} - v_i) \leq (\prod_{k=n_0}^{i-1} \gamma_k) sp(v_{n_0+1} - v_{n_0}) + 2YM \sum_{j=n_0+1}^{i} sp(v_j)|\Delta \Phi_{j-1}|(\prod_{k=j}^{i-1} \gamma_k). \tag{3.14}
\]
for all $i \geq n_0$. Then,

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

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

$$E \leq \frac{\|v_{n_0} - n_0 g^*\|_\infty}{n} + \frac{1}{n} \sum_{i=n_0}^{i-1} \prod_{k=n_0}^{i-1} \gamma_k sp(v_{n_0+1} - v_{n_0}) + \\
+ YM e_i + 2 \sum_{j=n_0+1}^{i} sp(v_j) |\Delta \Phi_{j-1}| (\prod_{k=j}^{i-1} \gamma_k). \quad (3.15)$$

**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 (3.15). 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 (3.15) as described in Theorem 3. As a consequence, we obtain a looser bound on the left hand side. We will see this in chapter 4.

Note that, in Theorem 1, we consider general function $\Phi$ while in Theorem 2 and The-
Theorem 3, we consider positive non-increasing function \( \Phi \).

**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 when \( i \geq n_0 \), then for

\[
n \geq \max \left( n_0, \frac{2}{\epsilon} \left\| v_{n_0} \right\|_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1 - \gamma} + 2YM \frac{e_{n_0}}{1 - \gamma} \right) , \tag{3.16}
\]

we guarantee: \( E = \frac{v_n - g^*}{n} < \epsilon \)

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

\[
\frac{2YM\Phi(n_0)}{1 - \gamma} < 1 \tag{3.17}
\]

and

\[
e_{n_0} \leq \frac{sp(v_{n_0}) + \frac{sp(v_{n_0+1} - v_{n_0})}{1 - \gamma}}{1 - \frac{2YM\Phi(n_0)}{1 - \gamma}} \Phi(n_0) \leq \frac{\epsilon}{2YM} \tag{3.18}
\]
Proof: By applying the Theorem 1 with \( n_0 \), we have:

\[
E \leq \frac{||v_{n_0} - n_0g^*||_\infty}{n} + \frac{1}{n} \sum_{i=n_0}^{n-1} \left[ \left( \prod_{k=n_0}^{i-1} \gamma_k \right) sp(v_{n_0+1} - v_{n_0}) + \right.
\]

\[
+ YM \ e_i + 2 \sum_{j=n_0+1}^{i} sp(v_j)|\Delta \Phi_{j-1}| \left( \prod_{k=1}^{i-1} \gamma_k \right) \right]
\]

\[
\leq \frac{||v_{n_0} - n_0g^*||_\infty}{n} + \frac{sp(v_{n_0+1} - v_{n_0})}{n(1 - \gamma)} + 
\]

\[
+ \frac{1}{n} \sum_{i=n_0}^{n-1} YM(e_i + 2 \sum_{j=n_0+1}^{i} (\gamma^{i-j} sp(v_j)|\Delta \Phi_{j-1}|))
\]

(3.19)

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:

\[
y_{n_0} = 0
\]
\[
y_{n_0+1} = sp(v_{n_0+1})\Delta \Phi_{n_0}
\]
\[
y_{n_0+2} = \gamma y_{n_0+1} + sp(v_{n_0+2})\Delta \Phi_{n_0+1}
\]
\[
y_{n_0+3} = \gamma y_{n_0+2} + sp(v_{n_0+3})\Delta \Phi_{n_0+2}
\]
\[
\ldots
\]
\[
y_n = \gamma y_{n-1} + sp(v_n)\Delta \Phi_{n-1}
\]
\[
\ldots
\]
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 \(e_{n_0} = \sum_{j=n_0+1}^{\infty} (sp(v_j) |\Delta \Phi_{j-1}|) \leq \frac{\epsilon}{2YM}\). 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:

\[
sp(v_{j+1}) \leq sp(v_j) + sp(v_{j+1} - v_j)
\]
\[
\leq sp(v_{j-1}) + sp(v_j - v_{j-1}) + sp(v_{j+1} - v_j)
\]
\[
\leq \ldots
\]
\[
\leq sp(v_{n_0}) + \sum_{i=n_0}^{j} sp(v_{i+1} - v_i)
\]

(3.20)

From (3.14), we have:

\[
sp(v_{i+1} - v_i) \leq 2YM \sum_{j=n_0+1}^{i} sp(v_j) |\Delta \Phi_{j-1}| \gamma^{j-1} + \gamma^{j-n_0} sp(v_{n_0+1} - v_{n_0})
\]
\[
\leq 2YM y_i + \gamma^{j-n_0} sp(v_{n_0+1} - v_{n_0})
\]

Then, for all \(j \geq n_0\),

\[
sp(v_{j+1}) \leq sp(v_{n_0}) + 2YM \sum_{i=n_0}^{j} y_i + sp(v_{n_0+1} - v_{n_0}) \sum_{i=n_0}^{j} \gamma^{j-n_0}
\]
\[
\leq sp(v_{n_0}) + 2YM \sum_{i=n_0}^{\infty} y_i + sp(v_{n_0+1} - v_{n_0}) \sum_{i=n_0}^{\infty} \gamma^{j-n_0}
\]
\[
\leq sp(v_{n_0}) + 2YM \frac{e_{n_0}}{1 - \gamma} + \frac{sp(v_{n_0+1} - v_{n_0})}{1 - \gamma}
\]
By plugging back into $e_{n_0}$, we get:

\[ e_{n_0} = \sum_{j=n_0+1}^{\infty} (sp(v_j)|\Delta \Phi_{j-1}|) \]

\[ \leq sp(v_0) + 2YM \frac{e_{n_0}}{1-\gamma} + sp(v_{n_0+1} - v_{n_0}) \sum_{j=n_0+1}^{\infty} |\Delta \Phi_{j-1}| \]

\[ \leq sp(v_0) + 2YM \frac{e_{n_0}}{1-\gamma} + sp(v_{n_0+1} - v_{n_0}) \Phi(n_0) \]

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

or $e_{n_0} \left( 1 - \frac{2YM\Phi(n_0)}{1-\gamma} \right) \leq \left( sp(v_0) + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} \right) \Phi(n_0)$.

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

\[ \frac{2YM\Phi(n_0)}{1-\gamma} < 1 \] (3.21)

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

\[ e_{n_0} \leq \frac{sp(v_0) + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} \Phi(n_0)}{1 - \frac{2YM\Phi(n_0)}{1-\gamma}} \leq \frac{\varepsilon}{2YM} \] (3.22)

- Now, for $n \geq n_0$,

\[ E \leq \frac{||v_0 - n_0g^*||_\infty}{n} + \frac{1}{n} sp(v_{n_0+1} - v_{n_0}) + \frac{YM\sum_{i=n_0}^{n-1} e_i}{n} + \frac{1}{n} 2YM \frac{e_{n_0}}{1-\gamma} \]

\[ \leq \frac{1}{n} ||v_0 - n_0g^*||_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM \frac{e_{n_0}}{1-\gamma} + \frac{YM(n-n_0)\frac{\varepsilon}{2YM}}{n} \]

\[ \leq \frac{1}{n} ||v_0 - n_0g^*||_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM \frac{e_{n_0}}{1-\gamma} + \frac{\varepsilon}{2} \]
Let \( \frac{1}{n} \left( \|v_{n_0} - n_0 g^*\|_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM \frac{e_{n_0}}{1-\gamma} \right) \leq \frac{\varepsilon}{2} \), or
\[
\begin{align*}
n \geq \frac{2}{\varepsilon} \left\| v_{n_0} - n_0 g^* \right\|_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM \frac{e_{n_0}}{1-\gamma}.
\end{align*}
\]

Then, for all \( n \geq \max \left( n_0, \frac{2}{\varepsilon} \left( \|v_{n_0} - n_0 g^*\|_\infty + \frac{sp(v_{n_0+1} - v_{n_0})}{1-\gamma} + 2YM \frac{e_{n_0}}{1-\gamma} \right) \right) \),
\[
E \leq \varepsilon
\]

**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 \( A \) both finite, \( |r(s,a)| \) bounded by a number \( M \). Suppose \( 0 < \gamma = \max(\gamma_f, \gamma_0); \gamma' = \max(\gamma_f, \gamma_0) < 1 \) and \( \Phi(i) \) is a positive non-increasing function on \([n_0, +\infty)\), then for
\[
\begin{align*}
n \geq \frac{2}{\varepsilon} \left( 2n_0M + \|v_0\|_\infty + \frac{\varepsilon}{1-\gamma} + \frac{M + (1+\gamma) \left[ \frac{M(1-\gamma')^{n_0}}{1-\gamma} + \gamma' n_0 (sp(v_0)) \right]}{1-\gamma} \right),
\end{align*}
\]
we guarantee:
\[
E = \frac{v_n}{n} - g^*_\infty < \varepsilon
\]
where $2Y = \max_{d \in D} \|P_0 - P_f\|_\infty$, $n_0$ is the smallest integer satisfying

$$\left[\frac{M}{1 - \gamma} + \gamma^{(n_0)}sp(v_0)\right] \Phi(n_0) \leq \frac{\epsilon}{2YM}$$  \hspace{1cm} (3.24)

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

$$E \leq \frac{\|v_{n_0} - n_0g^*\|_\infty}{n} + \frac{1}{n} \sum_{i=n_0}^{n-1} \frac{1}{i} \prod_{k=n_0}^{i-1} \gamma_k \cdot sp(v_{n_0+1} - v_{n_0}) +$$

$$+ YM \cdot e_i + 2 \sum_{j=n_0+1}^{i} sp(v_j) \Delta \Phi_{j-1} \left(\prod_{k=j}^{i-1} \gamma_k\right) \cdot$$

$$\leq \frac{\|v_{n_0} - n_0g^*\|_\infty}{n} + \frac{1}{n} \cdot sp(v_{n_0+1} - v_{n_0}) +$$

$$+ \frac{1}{n} \sum_{i=n_0}^{n-1} YM \cdot e_i + 2 \sum_{j=n_0+1}^{i} (\gamma^{j-1}sp(v_j)\Delta \Phi_{j-1}) \cdot$$

We have:

$$\|v_{n_0} - n_0g^*\|_\infty \leq \|r_{d_{n_0-1}} + P_{d_{n_0-1}}^0 v_{n_0-1} - n_0g^*\|_\infty \leq \| (r_{d_{n_0-1}} - g^*)\|_\infty + \|P_{d_{n_0-1}}^0 (v_{n_0-1} - (n_0 - 1)g^*)\|_\infty \leq \| (r_{d_{n_0-1}} - g^*)\|_\infty + \| (v_{n_0-1} - (n_0 - 1)g^*)\|_\infty \leq \sum_{k=1}^{n_0} \| r_{d_{k-1}} - g^*\|_\infty + \|v_0\|_\infty \leq 2n_0M + \|v_0\|_\infty \cdot$$

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

\[ sp(v_i) = sp(r^{d_i} + P^{d_{i-1}} v_{i-1}), \]
\[ \leq sp(r^{d_i}) + sp(P^{d_{i-1}} v_{i-1}), \]
\[ \leq [M + \gamma_{i-1} sp(v_{i-1})] \quad \text{(Proposition 4)}, \]
\[ \leq M \left[ 1 + \sum_{j=1}^{i-1} \prod_{k=1}^{j} \gamma_{i-k} + \prod_{k=1}^{i} \gamma_{i-k}(sp(v_0)) \right], \]
\[ \leq \frac{M(1 - (\gamma')^i)}{1 - \gamma'} + (\gamma')^i(sp(v_0)), \text{ where } \gamma' = \max(\gamma_0, \gamma_f). \quad (3.25) \]

Then,

\[ sp(v_{n_0+1} - v_{n_0}) \leq M + (1 + \gamma) sp(v_{n_0}), \]
\[ \leq M + (1 + \gamma) \left( \frac{M(1 - (\gamma')^{n_0})}{1 - \gamma'} + (\gamma')^{n_0}(sp(v_0)) \right). \quad (3.26) \]

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

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 \( j > n_0 \).

\[ e_{n_0} \leq \sum_{j=n_0+1}^{\infty} (sp(v_j) | \Delta \Phi_{j-1}|), \]
\[ \leq \frac{M}{1 - \gamma'} + (\gamma')^{n_0} sp(v_0) \sum_{j=n_0+1}^{\infty} (| \Delta \Phi_{j-1}|), \]
\[ \leq \frac{M}{1 - \gamma'} + \gamma^{(n_0)} sp(v_0) \Phi(n_0). \quad (3.27) \]
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 $e_{n_0} \leq \frac{M}{1-\gamma} + \gamma^{(n_0)} sp(v_0)$, $\Phi(n_0) \leq \frac{\epsilon}{2YM}$. Then for all $i \geq n_0$, $e_i \leq e_{n_0} \leq \frac{\epsilon}{2YM}$. Now, for $n \geq n_0$.

$$E \leq \frac{||v_{n0} - n_0g^*||_{\infty}}{n} + \frac{1}{n} sp(v_{n0+1} - v_{n_0}) + \frac{YM \sum_{i=n_0}^{n-1} e_i}{n} + \frac{1}{n} 2YM \frac{e_{n_0}}{1-\gamma},$$

$$\leq \frac{1}{n} \left( 2n_0M + ||v_0||_{\infty} + \frac{M + (1+\gamma) \frac{M(1-(\gamma)^{n_0})}{1-\gamma} + (\gamma)^{n_0}(sp(v_0))}{1-\gamma} + \frac{\epsilon}{1-\gamma} \right) \frac{\epsilon}{2},$$

Let $\frac{1}{n} \left( 2n_0M + ||v_0||_{\infty} + \frac{M + (1+\gamma) \frac{M(1-(\gamma)^{n_0})}{1-\gamma} + (\gamma)^{n_0}(sp(v_0))}{1-\gamma} + \frac{\epsilon}{1-\gamma} \right) \frac{\epsilon}{2} \leq \frac{\epsilon}{2},$ or

$$n \geq \frac{2}{\epsilon} \left( 2n_0M + ||v_0||_{\infty} + \frac{\epsilon}{1-\gamma} + \frac{M + (1+\gamma) \frac{M(1-(\gamma)^{n_0})}{1-\gamma} + (\gamma)^{n_0}(sp(v_0))}{1-\gamma} \right).$$

Then, $E \leq \epsilon$.

**Comments on the Theorem 3.** From Theorem 3, we can predict that number of steps necessary for the Value Iteration algorithm to ensure that $E < \epsilon$ 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 (3.25) and (3.26) together show the existence of $n_0$ in Theorem 2.

According to [14], the delta coefficient of a matrix is an upper bound of the second largest eigenvalue modulus $\lambda_*$. Thus, the gamma coefficient is also an upper bound of $\lambda_*$. Then, the term $\frac{1}{1 - \gamma}$ is a upper bound of the relaxation time $t_{rel} = \frac{1}{1 - \lambda_*}$ of $P_d$ for all $d \in D$. Moreover, the relaxation time is proportional to the the mixing time of $P_d$ or the convergence rate of the corresponding Markov chain [15]. 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 $P_d$. 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 $d^\star$. 
Chapter 4: Application of Adiabatic MDP: Queuing Systems

In this chapter we apply the above result to a continuous queueing system with $\lambda$ estimated.

4.1 Queuing Theory Basics

A queue is characterized by:

- Arrival Process
- Service Process
- Number of servers
- Queue size

We consider both continuous-time queueing system and discrete-time queueing system. They can be considered as Markov chains with transition matrices $P$.

**For continuous-time queueing system $M|M|1|K$:**

- Arrival Process: Poison process with rate $\lambda$
- Service Process: Poison process with rate $\mu$
• Number of servers: 1

• Queue size: \( K \)

The transition matrix for this queuing system in time \( t \): \( P^t = e^{Qt} \) where \( Q \) is generator matrix [16] [17]:

\[
Q = \begin{pmatrix}
-\lambda & \lambda & 0 & 0 & \cdots \\
\mu & -(\mu + \lambda) & \lambda & 0 & \cdots \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & 0 & \mu & -(\mu + \lambda) & \lambda \\
\cdots & 0 & 0 & \mu & -\mu
\end{pmatrix}
\]

For discrete-time queuing system:

• Arrival Process: Bernoulli process with parameter \( p \)

• Service Process: Bernoulli process with parameter \( q \)

• Number of servers: 1

• Queue size: \( K \)

The transition matrix for this queuing system is a triagonal matrix:

\[
P = \begin{pmatrix}
1 - p(1-q) & p(1-q) & 0 & \cdots & \cdots & \cdots & 0 \\
q(1-p) & pq + (1-p)(1-q) & p(1-q) & 0 & \cdots & \cdots & 0 \\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 0 & q(1-p) & pq + (1-p)(1-q) & p(1-q) & 0 \\
0 & 0 & \cdots & \cdots & \cdots & \cdots & q(1-p) & 1-q(1-p)
\end{pmatrix}
\]
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 $\lambda$ which is unknown. We estimate $\lambda$ 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 $k$th slot of duration $\Delta t$ 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 and low departure rate may lead to overflow. 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 $\beta_i$ or
\( \beta_i(s) = d_i(s) \). The generator matrix in time interval \((i \Delta t, (i+1) \Delta t]\) is shown as following:

\[
Q_i = \begin{pmatrix}
-\lambda_i & \lambda_i & 0 & 0 & \ldots \\
\mu_i & -(\mu_i + \lambda_i) & \lambda_i & 0 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \ddots \\
\ldots & 0 & \mu_i & -(\mu_i + \lambda_i) & \lambda \\
\ldots & 0 & 0 & \mu_i & -\mu_i
\end{pmatrix}.
\] (4.1)

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

\[
P_i = P(i \Delta t, (i+1) \Delta t) = e^{Q_i \Delta t}.
\] (4.2)

where

\[
Q_i = \hat{\lambda}_i \begin{pmatrix}
-1 & 1 & 0 & 0 & \ldots \\
1 + \beta_i(1) & -(2 + \beta_i(1)) & 1 & 0 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \ddots \\
\ldots & 0 & 1 + \beta_i(K-1) & -(2 + \beta_i(K-1)) & 1 \\
\ldots & 0 & 0 & 1 + \beta_i(K) & -(1 + \beta_i(K))
\end{pmatrix}
\]

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: 

\[ r(s, \beta) = -\frac{M(s-(K-1)\beta)^2}{\max_s \beta(s-K\beta)^2} \]

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 \( \beta \in [0,1] \).
4.2.2 Simulation Results

Let $Z_i = \sum_{k=1}^{i} X_k \sim \text{Poisson}(i\lambda \Delta t)$, $E[Z_i] = \text{VAR}[Z_i] = i\lambda \Delta t$.

By applying Chernoff 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 $\alpha$, 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,

$$P\left(\frac{Z_i}{i\Delta t} > (1 + a_i)\lambda\right) < \alpha$$

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

Therefore, with probability $\alpha$, we have $\hat{\lambda}_i > (1 + a_i)\lambda$. Similarly, with probability $\alpha$, 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 $1 - 2\alpha$, $(1 - b_i)\lambda \leq \hat{\lambda}_i \leq (1 + a_i)\lambda$. This is illustrated in Figure 4.1.
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 (4.2), 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.

Since $a_i, b_i$ decreases to 0 when $i$ goes to $\infty$, $\Phi(i)$ 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
Figure 4.2: The $\Phi(\cdot)$ function for the Simulation Scenario 1

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 4.2. 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 4.3 and Figure 4.4 show the actual distance to the optimal reward obtained from the Value Iteration algorithm and its upper bound by Theorem 2 and Theorem 2, respectively. As seen, they are well correlated with each other.
Figure 4.3: The actual distance and its upper bound for $\lambda = (1 + a_i)\lambda$ from Theorem 2

Similarly, for the lower bound on estimated arrival rate $\hat{\lambda}_i = (1 - b_i)\lambda$, we have $n_0 = 2, N = 15$ 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 is shown in Figure 4.5 and Figure 4.6. 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 the table 4.1. 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.
Figure 4.4: The actual distance and its upper bound for $\hat{\lambda}_i = (1 + a_i)\lambda$ from Theorem 3

Table 4.1: The optimal decision rule for continuous-time queuing system example

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

Table 4.2: The optimal decision rule for discrete-time queuing system example
Figure 4.5: The actual distance and its upper bound for $\hat{\lambda}_i = (1 + b_i)\lambda$ from Theorem 2 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:

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

Because all entries in the matrix $P$ are linear functions of $p$, if we set $p_i = \phi(i)p_0 +
Figure 4.6: The actual distance and its upper bound for $\hat{\lambda}_i = (1 + b_i)\lambda$ from Theorem 3

$(1 - \phi(i))P_f$ 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 the Table 4.3.

<table>
<thead>
<tr>
<th>$r(s, a)$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.9</td>
<td>0.8</td>
<td>0.7</td>
<td>0.6</td>
<td>0.5</td>
<td>0.4</td>
<td>0.3</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.4</td>
<td>0.3</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>0.2</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td>0.7</td>
<td>0.8</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 4.3: The reward $r(s, a)$ for discrete-time queuing system
4.3.2 Simulation Results

To examine the theoretical results, we run simulation using the following parameters: $\varepsilon = 0.01, p_0 = 0.4, p_f = 0.6, \phi(i) = \frac{1}{i+1}, A = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}, v_0 = 0e$.

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 4.7 and Figure 4.8 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.
Figure 4.8: The actual distance and its upper bound from Theorem 3

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 the table 4.2. 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.
Chapter 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 $\varepsilon$-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.
Bibliography


