Understanding Exploration Strategies in Model Based Reinforcement Learning

A THESIS

submitted by

PRASANNA P

for the award of the degree

of

MASTER OF SCIENCE

(by Research)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

MAY 2017

THESIS CERTIFICATE

This is to certify that the thesis titled **Understanding Exploration Strategies in Model Based Reinforcement Learning**, submitted by **Prasanna P**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Balaraman Ravindran
Research Guide
Associate Professor
Dept. of CSE
IIT-Madras, 600 036

Place: Chennai

Date: 15th May 2017

ACKNOWLEDGEMENTS

I thank my appa and amma, S. Parthasarathi and P. Amudhavalli, for their unconditional love and support, which served and will serve as a morale boost during my hard times.

I am thankful to my guide, friend and a well-wisher – Prof. B. Ravindran, who is one of my academic inspirations. His constant motivation and advice helped me a lot in compiling this piece of work. I would like to thank him for providing me a transforming experience under his tutelage and in helping me mature academically.

Further, I attribute my memorable life in IIT Madras to the time spent with my lab mates Santhosh, Praveen, Jana, Priyesh, Arvind Prakash, Sarath, Aravindram, Manimaran, Deepak, Kaushik, Neel, Vikas and Patanjali. The list is never ending. However, I will not be doing justice to my stay in IITM, if I fail to mention Naveen, Arjun, Vaishnavh, Aravind Srinivas, Arpita, Priyatosh and Deepak. I wish all of them good luck in their endeavors.

My sincere thanks to Prof. George Konidaris, Duke University, for his time during Summer 2015 when I did an internship under his guidance. The research, academic and career guidance provided by him during my internship gave a lot of clarity to my ambitions and approach.

With all their wishes, I present my research done during my M.S. at IIT Madras in this thesis.

ABSTRACT

KEYWORDS: Reinforcement learning, Exploration strategies, PAC-guarantees, Regret guarantees, Finite-episode agent, Exploration bonus.

Reinforcement Learning (RL) is a collection of learning approaches that solve for a behavior of an agent in a world maximizing some notion of payoff. Under certain circumstances the agent's world, described by a set of parameters, is essential to be learnt for it to discover an optimal behavior. This paradigm of problems is studied under the broad topic of model-based RL, and the class of algorithms that help an agent in learning the model parameters are known as model learning algorithms. In this thesis, we discuss exploration strategies in model-based RL from two different perspectives – asymptotic agent, where early convergence to an optimal solution is desirable but not a necessity, and finite-episode agent, which has a constraint on the number of episodes to converge to an optimal behavior.

The first part of this thesis proposes a novel *uncertainty based* exploration strategy, *Thompson Sampling with Exploration Bonus* (TSEB), for the asymptotic agent case. This work draws insights from Thompson Sampling, a Bayesian approach to model-based RL, and the homomorphism literature in RL. The proposed algorithm helps the agent learn close to true model parameters of the world with lesser sample complexity than the existing model learning algorithms. TSEB algorithm trades off the proposed exploration bonus with the sampled reward estimate in Thompson Sampling algorithm. This strategy, with the elegance of Bayesian approach, provides a better way to learn model parameters and is validated theoretically and empirically in this thesis.

Despite having sound strategies for the asymptotic case, the finite-episode case in RL has been seldom looked into. The other part of this thesis studies the exploration strategies of finite-episode agents and proposes a novel framework, *Structuring Finite-Episode Exploration (SFEE) in Markov Decision Processes*, that adapts an asymptotic exploration strategy for finite-episode setting. This work justifies the need for such a framework by showing that the asymptotic exploration strategies don't suit the finite-episode setting which can be because of them not being conscious of the agent's life-time. Aiming to make good use of the sound strategies of asymptotic learning algorithms, the proposed framework uses the policy of an asymptotic algorithm plugged into the framework for a certain number of episodes and then appropriately switches to a greedy policy. Further, this thesis shows, under some mild assumptions, that this is indeed the optimal way to explore in a finite-episode setting using that exploration algorithm. The framework also enables exporting theoretical guarantees from the *plugged* asymptotic algorithm to the proposed framework.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

RL	Reinforcement Learning
MDP	Markov Decision Process
TS	Thompson Sampling
TSEB	Thompson Sampling with Exploration Bonus
SFEE	Structuring Finite-Episode Exploration
ERM	Expected Return Margin
EER	Exploit Episode Return
L	Exploration Loss
G	Gain
PAC	Probably Approximately Correct

NOTATION

S	Set of states
A	Set of actions
R	Reward function
P	Transition function
γ	Discount factor
ρ	Exploration bonus
λ	Tuning parameter
Δ_i	Regret in i^{th} step
I	Set of states that are known
Ω	Set of states that will possibly be known in the next explore episode
S_t	Set of start states
π	Policy
π^x	Exploit policy
π^e	Explore policy
π^*	Optimal policy

CHAPTER 1

INTRODUCTION

1.1 Introduction

One of the primary objectives of Artificial Intelligence (AI) research is to build learning agents that surpass human intelligence. Reinforcement Learning (RL) (Sutton and Barto, 1998), a branch of AI motivated by research in behavioral psychology, addresses this intuitively and with mathematical sophistication. Reinforcement is a positive or negative payoff that is provided to a learning agent. The RL problem is to maximize the long-term or cumulative payoff.

1.1.1 Reinforcement Learning

A reinforcement learning problem is posed as a Markov Decision Process (MDP) $< S, A, R, P, \gamma >$ where

- *S* is a finite set of states
- *A* is a finite set of actions
- $R : S \times A \rightarrow \mathbb{R}$ is the reward function that provides the expected immediate payoff for every state-action pair
- $P : S \times A \times S \rightarrow [0, 1]$ is the transition function and $P(s, a, s')$ defines the probability of transitioning to s' from s by taking action *a*, where $s, s' \in S$ and *a* $\in A$
- $\gamma \in [0, 1]$ is the discount factor representing the importance of future and current rewards

Given such an MDP, the objective of the agent is to learn an optimal behaviour or policy $\pi^* : S \rightarrow A$ that maximizes the cumulative reward over a finite or infinite horizon, H.

The sequential nature of the problem makes it challenging for the agent since the decision taken at a particular time step not only affects the current reward that the agent

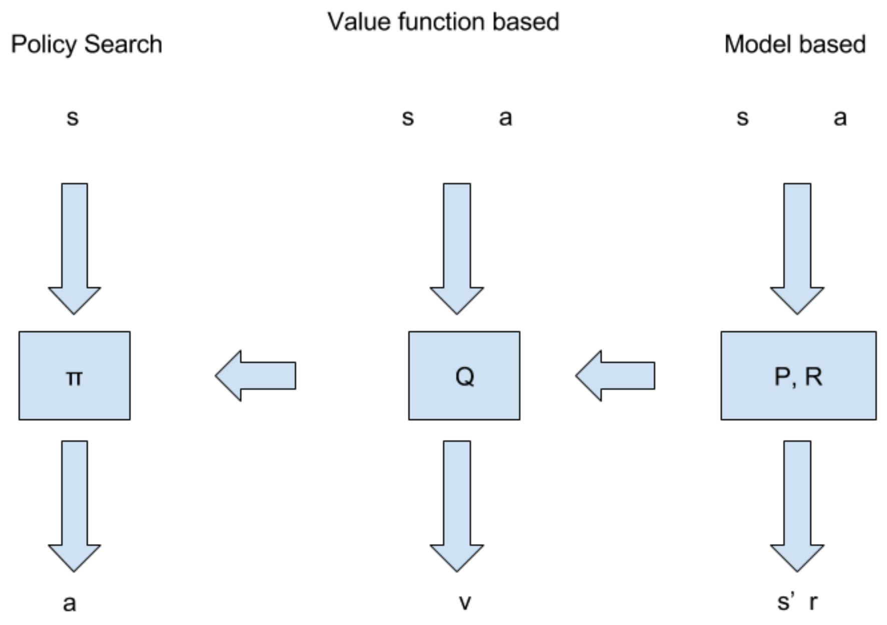


Figure 1.1: Spectrum of RL approaches.

gets, but also the future rewards by determining what the subsequent state will be. Hence, in such a sequential setup, it is useful for the agent to predict the cumulative reward obtained by following a policy π from any state. This is exactly captured by the *value* of a state under a policy.

1.1.2 Approaches in Solving an RL Problem

An RL problem posed as a reward maximization problem has traditionally been solved by different approaches (see Figure 1.1). Every approach has its significance in the literature. We shall briefly go over the approaches in this subsection.

Policy Search Based Approaches

An RL problem posed as an MDP, needs to be solved for an optimal policy π^*. Policy search approaches try to solve for the problem directly in the *policy space*, *i.e.,* the space of all possible policies. Techniques like policy gradient and actor-critic methods, as discussed in (Barto *et al.*, 1983), involve direct usage of policy and keep updating the parameters that define the policy until the policy parameters converge to a local optimum. By nature, these approaches are scalable as the policy can be represented by a differentiable approximator like a neural network. Apart from the local minima problem, policy search based approaches also suffer from over-fitting. This is not a problem *per se*, if the learning agent is able to explore the state space in full. In practice, in a very large discrete space or a continuous space the probability of exploring the entire state space is close to zero. Hence, the agent might learn only from a limited sample that might skew the behaviour. Despite the policy search approaches' inability to converge to global optima, they have remained a favorite when it comes to large state spaces.

Value Based Approaches

It is advantageous for a learning agent in an MDP, with its parameters known, to learn *value* function to base its decision. A value function can be defined for a state ($V^{\pi}(s)$) as the expected cumulative reward from a state s by following a policy π, or for a state-action pair ($Q^{\pi}(s, a)$). In a small discrete world, value functions are stored using look-up tables. When the parameters of the world are known, the optimal value function is computed using dynamic programming approaches like, *Value Iteration* or *Policy Iteration* (Puterman, 1994). However, these approaches require the transition and reward functions to be known a priori and hence cannot be scaled to larger domains. But, these approaches can guarantee convergence to the global optimum in the value function space, which is achieved using the Bellman operator (Puterman, 1994) defined in Equation. 1.1,

$$V^*(s) = \max_a \left[R(s, a) + \gamma \sum_{s'} P(s'|s, a) V^*(s') \right]. \tag{1.1}$$

The optimal value function obtained by using the following update equation,

$$V_{i+1} = \max_a \sum_{s'} \left[R(s, a) + \gamma P(s'|s, a) V_i(s') \right] \tag{1.2}$$

Equation 1.2 will converge to the value function of the optimal policy. Despite the disadvantages of being less scalable and requiring a model, this has the advantage of converging to the global optimum.

The scalability issue has been tackled by using function approximators like CMAC (Miller *et al.*, 1990), Radial Basis Functions (RBF) (Anderson, 1992), Proto Value

Functions (Mahadevan and Maggioni, 2007) and Artificial Neural Networks (Tesauro, 1995) to represent the value function. The approximators learn a parameterized representation for value function, avoiding the need for using large look up tables. Such approaches have been in use for more than two decades and, to an extent, have proven effective in mitigating the scalability issue. The drawback of such approximation techniques is that the approximation to the true value function learnt in certain complicated domains can be way off affecting the agent from learning the optimal behaviour.

There is another class of techniques, *model-free algorithms*, that do not require the agent to know the model but still learn estimates for the value functions like TD-learning (Sutton, 1988), SARSA (Rummery and Niranjan, 1994) and Q-learning (Watkins and Dayan, 1992). These approaches learn the estimates from the sample trajectories without needing an explicit model to be specified. Though they are useful in practice, they require a large number of sample trajectories for convergence.

Model Based Approaches

Most of the real world problems have unknown transitions and/or reward functions. The unavailability of the model prevents an agent from using value-based approaches in cases that require it to converge to an optimal behavior. Although model-free approaches help in learning an optimal policy without requiring an agent to learn the model parameters, they do not effectively help in long term planning. Model-based approaches like Dyna and Real-Time Dynamic Programming (RTDP) (Barto *et al.*, 1995) help in mitigating this issue by aiding an agent to learn an estimate of the true model and then use the estimated model to learn an optimal behavior. The optimality of the behavior depends on the quality of the estimated model.

The lack of better trajectory generation in model based approaches mentioned above warranted techniques that have effective exploration strategies. A good exploration strategy aids an agent in generating sample trajectories that help the agent speed up the model learning phase. One effective exploration strategy is to give an exploration bonus – an incentive for the agent to explore – for effective sample generation. Earlier work on exploration bonus based approaches for model-based RL include (Brafman and Tennenholtz, 2003), (Kearns and Singh, 2002) and some of the recent ones are BEB (Kolter and Ng, 2009), Approximate Bayesian Reinforcement Learning (Sorg *et al.*, 2010) and

UCRL (Sorg and Singh, 2009). The results of these works provide improved learning guarantees and have garnered reasonable attention in the RL community. One of the major advantages of the exploration bonus based approaches is that they provide greater control on an agent's learning. The control on learning is established by improved upper bounds on the expected number of samples for convergence, number of mistakes, etc. Such guarantees are difficult to obtain in model-free approaches.

Bayesian Reinforcement Learning

A very early approach to model based setting, Thompson Sampling (TS) (Thompson, 1933), was proposed for non-sequential problems. Though the TS algorithm was proposed in 1933, the learning guarantees for it have been proved very recently (Agrawal and Goyal, 2012). This was also adopted for the MDP setting (Strens, 2000) and has been used in solving different problems in model based RL setting. TS comes under the broad category of Bayesian Reinforcement Learning (BRL) (Ghavamzadeh *et al.*, 2015). This represents a class of algorithms to solve model-based RL assuming only the knowledge of the state space, *S*, and the action space, *A*. The agent starts with prior beliefs over the transition function, *P*, and the reward function, *R*. For convenience, the beliefs over transition and reward functions are often assumed to be Dirichlet and Gaussian distributions respectively.

The description of Thompson Sampling is as follows. The algorithm assumes a prior belief over the model parameters, b_{e0}. At any episode T, the agent samples an MDP from $b_{e_{T-1}}$ and solves for an optimal policy in the sampled MDP using any of the dynamic programming (DP) methods. This policy is then used to generate trajectories that look like $s_t, a_t, s_{t+1}, r_{t+1}, s_{t+1}, a_{t+1}, s_{t+2}, r_{t+2}, ...$ A trajectory can be viewed as a set of tuples where the i^{th} tuple is represented as $(s_{i-1}, a_{i-1}, s_i, r_i)$. A trajectory with N time-steps has N such tuples that the agent uses to update the posterior belief, b_{eT}. With increasing number of episodes, the belief over the model parameters converges to the true parameters. Asymptotically the sampled MDP will converge to the true MDP, solving which provides an optimal policy. This approach guarantees convergence to a policy that has a near optimal regret (Auer and Ortner, 2006; Osband *et al.*, 2013).

1.2 Similarity Between Reinforcement Learning and Supervised Learning

The problem of learning a mapping from states to discrete actions is similar to the problems in supervised learning paradigm. In a supervised learning problem, a learning agent is expected to learn a mapping function from the input space to the output symbols minimizing on an error function. In RL, the state space, analogous to the input space, is mapped to action symbols maximizing the cumulative reward. Despite the similarity in the learning function, the learning process is different between the two paradigms.

A supervised learning approach has a labelled dataset, *i.e.,* pairs of input vector and output label, known as the *training set* $(x_i, y_i)_{i=1}^{N}$, where $x_i \in X$, and $y_i \in Y$ and X and Y are input and output spaces respectively. A learning function is provided with this training set of N examples to learn a mapping function $\hat{f} : X \rightarrow Y$.

On the other hand, in an RL approach, an agent has to generate experience from the world in the form of trajectories. The agent uses the information in the trajectories to learn an optimal policy, π^*. Unlike the supervised learning approach, an RL agent has control over the training data it generates, similar to the *active-learning* (Bonwell and Eison, 1991) set-up. Hence, the convergence to an optimal behavior using RL approach is vastly affected by the samples generated by the learning agent.

1.3 Contributions of this Thesis

Exploration in RL aids an agent in generating sample trajectories that help in learning more about the model parameters, thereby helping the agent improve its policy learning from improved model parameters. The recent researches in model-based approaches have shown that structured incentive for agents to explore is useful for better sample generation. Along the lines of exploration based approaches for model based agents, we propose and analyze a novel *value-based* exploration bonus for asymptotic model based agents that provides a sample efficient way to learn the model parameters.

Further, the thesis also motivates and discusses model-based approaches from a finite-episode agent's perspective, which, to the best of our knowledge, is a novel contribution at least in a sequential learning set up. In regard to the finite-episode set up problem proposed, the thesis proposes a simple framework that can adapt an asymptotic model-based strategy to learn a better policy in a finite episode sequential learning problem. The two contributions of the thesis can be summarized as follows,

1. Proposing and analyzing a TS like sample efficient exploration strategy for asymptotic model-based exploration agents.
2. Proposing and analyzing a exploration framework for finite-episode model-based agents, which can be used to extend any of the existing asymptotic strategies.

The organization of the thesis is as follows. Chapter 2 discusses the required mathematical background and a brief literature survey. In chapter 3, we address the problem of exploration strategy of an asymptotic agent in model-based RL. Chapter 4 discusses the finite-episode exploration strategy. In chapter 5, we conclude the thesis with a brief discussion on the possible extensions and future work.

CHAPTER 2

BACKGROUND AND LITERATURE SURVEY

2.1 Background

2.1.1 Markov Decision Process

Reinforcement Learning problem is posed as a Markov Decision Process (MDP) $< S, A, R, P, \gamma >$ where

- S is a finite set of states
- A is a finite set of actions
- $R : S \times A \rightarrow \mathbb{R}$ is the reward function that provides the immediate payoff for every state-action pair
- $P : S \times A \times S \rightarrow [0, 1]$ is the transition function that defines the probability of transitioning to s' from s by taking action a, where $s, s' \in S$ and $a \in A$
- $\gamma \in [0, 1]$ is the discount factor representing the importance of future and current rewards

Given such an MDP, the objective of the agent is to learn an optimal behaviour or policy $\pi^* : \ S \rightarrow A$ that maximizes the cumulative reward over a finite or infinite horizon, H. With this notation, we define below the *value* of a state under a policy π,

$$V^\pi(s) = \mathbb{E}_\pi \left[R(s, \pi(s)) + \gamma P(s'|s, \pi(s)) \sum_{s'} V^\pi(s') \right] \quad \forall s \in S \qquad (2.1)$$

While reward captures the immediate payoff for a particular state-action pair, value function with respect to a policy (V^π) captures the expected cumulative payoff that the agent would receive if it starts from that state and follows a policy π.

Algorithm 1: Thompson Sampling Algorithm (Strens, 2000)

```
Input: Prior distribution over all possible MDPs $f$, t=1.
  Define: $M^k$  ▷ MDP sampled from $f$.
      $\pi_k^*$  ▷ Optimal Policy for $M^k$, obtained from an MDP solver.
      $a_t$  ▷ action sampled from $\pi_k^*$.
      $s_t$  ▷ state at $t$.
      $r_t$  ▷ scalar reward obtained for taking $a_t$ in $s_t$.
for episodes k =1,2,3,4,... do
    $M^k \sim f(.|H_{tk})$
    Compute $\pi_k^*$ = solve($M^k$)
    for timesteps j=1,2,3,4,...$\tau$ do
        $a_t \sim \pi_k^*(s_t, j)$
        observe $r_t$ and transition to $s_{t+1}$
        $t \leftarrow t + 1$
    end
end
```

2.1.2 Thompson Sampling

Thompson Sampling, a Bayesian RL algorithm, is described in Algorithm 1.

For convenience, the prior distribution over the parameters is assumed to be Gaussian and Dirichlet for reward and transition functions respectively. To be specific, when the expected rewards are standardized, $r \in [-1, 1]$, the distribution is usually a standard Gaussian. An unbiased Dirichlet prior representing the transition function of a state action pair will have a $|S|$ length vector of ones ($Dir(1, 1, 1...,))$).

In Algorithm 1, the history of trajectories till episode k and time step t. π_k^* represents the optimal policy for the sampled MDP, M^k. The optimal policy, π_k^*, is returned by `solve()`, which can be any of the methods discussed in the previous chapter. The obtained policy is used to generate trajectories, which is used to update the posterior distribution over the model parameters. This is repeated until the agent's performance – cumulative reward of the max policy– in the true world converges.

2.1.3 Distance Between Two MDPs

The expression for the value function in Equation 2.1 is useful in understanding the metric to bound the distance between two MDPs M_1 and M_2. Consider two MDPs M_1 (S,A,R,P) and M_2 (S,A,R',P') that have the same state and action spaces but different reward and transition dynamics. Let the *max* norm over the difference in their rewards be K_r, similarly the differences in the transition be K_p and the difference between the maximum reward and the minimum reward be δ_r. The following expression provides a notion of upper bound on the distance between two MDPs in the value space (Ravindran and Barto, 2004):

$$f(K_r, K_p, \gamma) = \frac{2}{1-\gamma}\left[K_r + \frac{\gamma}{1-\gamma}\delta_r K_p\right] \tag{2.2}$$

where,

$$K_r = \max_{s\in S, a\in A} |R(s,a) - R'(s,a)|$$
$$K_p = \max_{s\in S, a\in A} |\sum_{s'} T(s,a,s') - T'(s,a,s')|$$
$$\delta_r = \max_{s\in S, a\in A} R(s,a) - \min_{s\in S, a\in A} R(s,a)$$

In the case of Bayesian Reinforcement Learning (see Algorithm 1), M_1 and M_2 can be associated with the true and sampled MDPs respectively. The true MDP is not known in a Bayesian setting and M_1 is approximated with the samples generated from the MDP so far. Further, as the transition probabilities are generated by a Dirichlet distribution, K_p can be upper-bounded by $\frac{1}{n(s,a)}$ (Sorg *et al.*, 2010), where $n(s,a)$ is the number of times the action *a* was taken in state *s* since the first episode. Equation 2.2 can be rewritten as follows,

$$f(K_r, \gamma) = \frac{2}{1-\gamma}\left[K_r + \frac{\gamma\,\delta_r}{(1-\gamma)\min\limits_{s\in\mathcal{S}, a\in\mathcal{A}} n(s,a)}\right] \tag{2.3}$$

In Equation 2.3, K_r is estimated as the maximum difference between the expected reward sampled from the posterior reward distribution for state-action pairs in an episode to the empirical mean of the rewards for state-action pairs that are computed based on the samples obtained till that episode.

2.1.4 Learning Guarantees

A learning algorithm's performance is indicated by the learning guarantees it provides. Each learning guarantee uses a different setting to analyze an algorithm's performance. This thesis considers the following two notions of learning guarantees,

Definition 2.1.1. ***PAC-MDP:*** *An algorithm A is said to be PAC-MDP (Probably Ap-*

proximately Correct - Markov Decision Process), if for any $\epsilon > 0$ and $0 < \delta < 1$ the number of sub-optimal steps, selection of an action other than the one chosen by an optimal policy, is less than some polynomial in $\left(S, A, 1/\epsilon, 1/\delta, \frac{1}{1-\gamma}\right)$ with probability at least (1-δ).

Definition 2.1.2. ***Regret:*** *Let G^* be the optimal discounted T-step cumulative reward from any state $s \in S$. Total regret of algorithm, A, after T steps from a state s in an MDP M is defined as,*

$$\Delta(M, A, s, T) = G^* - G(M, A, s, T)$$

Where G(.) is the T step cumulative reward from state s.

2.2 Literature Survey

In standard Reinforcement Learning (RL) framework, the environment with which an agent interacts is modeled as a Markov Decision Process (MDP). The goal of a learning agent is to learn a policy such that the long term measure of a performance is maximized over a finite or an infinite horizon. If the transition and reward parameters of the MDP are known, then the learning process is straight forward, and the optimal policy can be learnt with traditional DP-methods (Bertsekas and Tsitsiklis, 1996). However, in most real life applications, the parameters of the MDP are not known *a priori*. In such scenarios, the agent can try to directly learn a policy that maximizes the return (model-free learning) or the agent can try to estimate the parameters of the MDP and learn a policy based on the learnt MDP (model-based learning).

Recently, model-based learning approaches have been receiving increased attention (e.g. (Brafman and Tennenholtz, 2003; Sutton and Barto, 1998; Strens, 2000; Kolter and Ng, 2009; Sorg *et al.*, 2010; Russo and Roy, 2014)). In model-based RL, the goal of an agent is two-fold. First, it should estimate the true parameters of the model. Second, it should not sacrifice too much on performance while trying to learn the model parameters. The agent has to explore to learn the model parameters, but trying to *over-explore* in improving the belief over the model parameters may increase the regret. Typically, a Bayesian learning agent (Thompson, 1933) maintains a distribution (or a belief) over the model parameters, and it gets updated as and when the agent receives a sample, $(s_t, a_t, s_{t+1}, r_{t+1})$, where s_t is the state the agent is at time t, a_t is the action the agent took at time t and s_{t+1} and r_{t+1} are next state and its corresponding reward respectively. As the number of samples increases, the belief converges to the true parameters of the

MDP.

Among the model based methods, Bayesian approaches are particularly attractive due to their amenability for theoretical analysis, and for their convenient posterior update rule. Much of the recent work has been focused on *Thompson sampling* (TS) (Thompson, 1933) based approaches both in simpler bandit settings (Chapelle and Li, 2011; Agrawal and Goyal, 2012, 2013; Gopalan *et al.*, 2013), as well as the full MDP problem (Strens, 2000; Gopalan and Mannor, 2015; Russo and Roy, 2014). Ever since (Chapelle and Li, 2011) discussed the efficacy of TS approaches for reinforcement learning, there have been concerted attempts to achieve better understanding of such approaches. Apart from the results in the bandit setting, TS approach for full RL has been shown to work well in practice and has been shown to be regret optimal (Gopalan and Mannor, 2015). However, there are no PAC guarantees in the literature for the TS approach. To achieve better PAC guarantees an algorithm has to encourage *more aggressive* exploration than enjoined by the basic TS approach and one way to do that is to use an exploration bonus.

A widely applied strategy for exploration bonus comes under the broad technique of *Optimism in the face of uncertainty (Abbasi-Yadkori* et al.*, 2011).* This technique illustrates that heuristically over-estimated *state-value* or *action-value* aids in the exploration of an agent. In (Kaelbling, 1990), an algorithm proposed as *Interval estimation Q-learning* (IEQ) for the non-sequential multi-arm bandit setting chooses the action with the highest upper bound on the underlying Q-value. This approach also asserts that the gradual decay of the over-estimation lets the agent converge to an optimal policy. This has been followed in approaches as early as UCB(Auer *et al.*, 2002), where the empirical mean, $\hat{\mu}_i$, of an arm i is over-estimated by the confidence interval of the estimated mean. UCRL (Auer and Ortner, 2006) takes an approach inspired by the UCB technique for over estimation to aid exploration in MDPs.

Recent approaches to model learning in full RL setting have adopted exploration bonus parameter from the multi-arm bandit setting. This has resulted in early convergence to a better model and significant improvements in PAC bound of an algorithm. There are quite a few algorithms proposed with different exploration strategies, for example, R-max (Brafman and Tennenholtz, 2003) assumes that the belief over the model parameters is improved by visiting a state large number of times, which is estimated as a function of (ϵ, δ) Definition 2.1.1. (Kearns and Singh, 2002) proposed a theoretical framework that chooses exploration or exploitation based on the number of visits to the state and an estimate over the expected cumulative reward from the state. (Kolter and Ng, 2009) proposed Bayesian Exploration Bonus (BEB) algorithm which adds a constant exploration bonus for every state/state-action pair whose parameter is unknown. This algorithm improves on another exploration bonus approach, MBIE-EB (Strehl and Littman, 2008), in terms of the PAC bounds. Though these algorithms have good strategies, they have a weak assumption of uncertainty being uniform over the state space. Recently, (Russo and Roy, 2014) highlighted an information-theoretic analysis of TS, giving a better regret bound, considering the entropy in the action-distribution. (Sorg *et al.*, 2010) use the variance in the model parameters to derive an approach that is adaptive and more sophisticated than the earlier works. These methods provide an adaptive exploration bonus thereby minimizing the number of samples needed to have a better estimate. To reduce the variance in the sampled model Best of Sampled Sets (Asmuth *et al.*, 2009), *BOSS*, samples multiple MDPs and solves an MDP that is the average of the sampled MDPs. Similar to R-max, the framework visits every state large number of times to estimate the parameters with greater belief.

It is important for the model learning strategies to maintain a distribution over the model parameters such that the probability mass over the true parameters is non-zero. Approaches like (Sorg *et al.*, 2010) and (Kolter and Ng, 2009) compute a point estimate which makes it theoretically unlikely for the probability mass over the true model parameters to be non-zero. Such approaches may converge to a very bad estimate in certain cases. On the other hand, TS approaches can be appreciated for their Bayesian update rule that doesn't make the probability mass over the true parameters insignificant provided the prior is not too biased. But, TS approach can only guarantee a regret optimal solution with a suboptimal PAC guarantee.

So far we have been looking at the unconstrained-time setting for learning model parameters. A similar theme exist for finite-time setting of non-sequential learning (stochastic multi-arm bandit problems). To name a few approaches in this setting, (Auer and Ortner, 2010) provides an arm deletion strategy that deletes an arm after every fixed number of time steps repeatedly, if the estimated reward of that arm is less than the threshold until only single arm is left. Recent work like OFUL (Abbasi-Yadkori *et al.*, 2011), a parameterized bandit setting, uses a similar optimism in the face of uncertainty strategy to improve on the regret in a non-sequential setting. Another related work is that of budgeted bandits, (Bubeck *et al.*, 2009), where the goal is to use a finite budget to discover what the best action is and then to repeatedly pick the action thereafter. The budget can be spent purely on exploration. Most of the finite-time results pertain to a non-sequential setting with no corresponding results for the sequential setting.

CHAPTER 3

MORE EFFICIENT THOMPSON SAMPLING WITH EXPLORATION BONUS

3.1 Introduction

In this chapter, we propose an asymptotic exploration algorithm, TSEB (Thompson Sampling with Exploration Bonus), which is a modification over the traditional Thompson Sampling algorithm with an exploration bonus for every state-action pair. TSEB uses an uncertainty based exploration bonus that takes insights from the homomorphism literature in RL.

The underlying principle of TSEB is similar to other model learning algorithms, *i.e.,* to provide incentive for an agent to explore new state-action pairs. But, the specific form of the exploration bonus is what makes TSEB effective. The proposed exploration bonus, for a state-action pair, is related to the uncertainty in its estimated transition and reward functions. This dependence of exploration bonus on the estimated parameters of a state-action pair makes TSEB adaptive.

The major contributions in this part of the thesis are,

- Introducing an uncertainty based exploration bonus.
- Providing a PAC guarantee for TSEB.

The organization of this chapter is as follows, Section 2 describes TSEB algorithm followed by theoretical analysis in Section 3 and Section 4 provides the empirical validation of TSEB.

3.2 TSEB Algorithm

The core of the TSEB algorithm is TS, and it proceeds by first sampling a model from a posterior distribution and then solving the sampled model. The crucial difference is that the sampled MDP is augmented by an exploration bonus that is based on the statistics of the posterior. The solution of this augmented MDP is then used to generate sample trajectories with which the posterior is updated. Since the parameters of the MDP are known, we use value iteration to solve for an optimal policy in the augmented MDP.

The key observation in deriving our exploration bonus is that the posterior distribution captures the uncertainty in the belief over the model parameters of the true MDP. Hence, if we tailor our exploration bonus to drive us towards states where the uncertainty is the highest, we will converge to the true estimate of the model.

As a surrogate to this measure, we use Equation 2.3 in Section 2.1.3, which provides an upper bound to the difference of the value functions of two states. If for a particular state-action pair in the sampled MDP, if this upper bound as measured from the expected value function is large (as explained in Section 2.1.3), then it stands to reason that the posterior is very uncertain about the parameters for that state-action pair. Hence, we make the exploration bonus of a state-action pair proportional to the upper bound,

$$f_{s,a}(K_r, \gamma) = \frac{2}{1-\gamma}\left[K_r + \frac{\delta_r\,\gamma}{(1-\gamma)n(s,a)}\right] \tag{3.1}$$

where *n(s,a)* is the number of times action *a* was taken in state *s* across all the episodes in the true MDP. In a normalized reward setting, where $R(s,a) \in [-1,1]$, δ_r can be upper bounded by 2.

The exploration bonus being dependent on K_r and number of visits (Equation 3.1) illustrates its dependence on the uncertainty in the model parameters. This makes the exploration bonus adaptive (exploration bonus can be computed for every state-action pair individually) and dynamic (the exploration bonus for a state-action pair depends on the samples used to update its estimated parameters). The modified Bellman update to accommodate the exploration bonus is as follows,

$$V_t(s) = \max_{a\in A}\left[\lambda R(s,a) + (1-\lambda)\rho_t(s,a) + \gamma\sum_{s'} P(s'|s,a)V_{t-1}(s')\right] \tag{3.2}$$

$$\lambda \in (0,1]$$

$$\rho_t(s,a) = \frac{(n(s,a)-1)\rho_{t-1}(s,a) + f_{s,a}(K_r,\gamma)}{n(s,a)} \tag{3.3}$$

As can be seen from Equation 3.1, the more uncertain the posterior about a state-action pair, the higher will be the exploration bonus for that pair. Further, from Equation 3.2 it can be inferred that the probability of visiting a state-action pair is directly dependent on its exploration bonus, which decays linearly with the number of visits (Equation 3.3). Trajectories generated by TSEB agent, using the proposed exploration bonus, help in deriving better estimates for the posterior model distribution.

As the exploration bonus is computed using the estimated model parameters, the quality of the exploration bonus is affected by the number of samples used to estimate it. This lets the initial values for exploration bonus to have high variance but it gets better with the increase in the number of samples. The initial high variance doesn't affect the value function estimates, as TSEB uses a convex combination of the reward estimated in the sampled world and the exploration bonus computed for that state-action pair $\rho(s, a)$ (Equation 3.2).

The pseudocode for TSEB is depicted in Algorithm 2. The algorithm uses *value iteration* to solve for the optimal policy in an episode. The algorithm converges to an optimal policy, as the exploration bonus for the state-action pairs become insignificant. Hence, with the increasing number of samples generated with each episode, the estimated model distribution grows closer to the true model distribution.

TSEB, unlike most other previous algorithms (with exception (Sorg *et al.*, 2010)), uses the uncertainty in the estimates to structure its exploration bonus. This gives a better structure to exploration thereby providing better PAC guarantees for the algorithm. Further theoretical analysis shows that the bound is indeed tighter than the PAC bound obtained in earlier approaches. This can be attributed to the nature of the exploration bonus that enables TSEB to identify and concentrate much of its exploration around uncertain regions.

As with any Bayesian algorithm, in TSEB too, the initial beliefs over the model parameters play a vital role in the convergence. If the priors are *bad*, and the true model is assigned a low probability then the convergence maybe delayed. The effect of the prior is evident in the sample complexity of TSEB, discussed later, that is directly proportional to $f_0(K_r, \gamma)$ – the expected initial distance of the estimated model parameters from the true parameters. TSEB can also be extended to provide a prior over exploration bonus, which may help in learning the model faster. But, we don't analyze *a priori* exploration bonus in this work.

Algorithm 2: Thompson Sampling with Exploration Bonus (TSEB)

```
Input: Parameter Space $\Theta$, prior over $\Theta$, action space $\mathcal{A}$, state space $\mathcal{S}$, and $\gamma$.
 Define: E  ▷ Number of episodes.
      $T_e$  ▷ Number of time-steps in episode $e$.
      $\rho_t$  ▷ Exploration Bonus in time-step $t$ of an episode.
      $r_t^e$  ▷ reward at time step $t$ in episode $e$.
      $r$  ▷ reward obtained by taking an action.
      $V_e$  ▷ Value function in episode $e$.
      $R_e$  ▷ Reward function sampled for episode $e$.
      $P_e$  ▷ Transition function sampled for episode $e$.
Output: policy $\pi$
for e in range (E) do
    $M_\theta$ ←Sample $R_e$ and $P_e$ from posterior
    $V_e$ ← Value_Iteration(M$_\theta$)
    for t in range ($T_e$) do
        $\pi(s_t^e)$ ← argmax$_a$ $(\lambda R_e(s_t^e, a) + (1-\lambda)\rho_t(s_t^e, a)$
                        $+ \gamma \sum_{s'} P_e(s'|, a) V_e(s'))$
        r←get_onestep_reward($\pi(s_t^e)$)
        n($s_t^e$)← n($s_t^e$)+1
        n($s_t^e, a$)← n($s_t^e, a$)+1
        r($s_t^e, a$)←r($s_t^e, a$)+$\frac{1}{n(s_t^e,a)}$ $[r - r(s_t^e, a)]$
        $K_r \leftarrow \frac{1}{n(s_t^e,a)} \sum_{i=1}^{n(s_t^e,a)-1} r_i(s_t^e, a) - R_e(s_t^e, a)$
        $\rho(s_t^e, a) \leftarrow \frac{(n(s_t^e,a)-1)\rho(s_t^e,a)+f_{s_t^e,a}(K_r,\gamma)}{n(s_t^e,a)}$
    end
    Update the posterior: $\pi_{t+1}(d\theta) \propto p(S_t, A_t, R_t, S_t')\pi(d\theta)$
end
```

3.3 Theoretical Analysis of TSEB

In this section, we derive PAC guarantee for the algorithm to upper bound the number of steps that the agent takes before converging to an ϵ-optimal solution with probability $1-\delta$. Existence of a PAC bound for TSEB, despite TS approach not having any, can be attributed to the ability of TSEB to engage in more aggressive exploration than TS.

As the exploration bonus obtained using $f_{s,a}(K_r, \gamma)$ (Equation 3.1) provides an upper bound on the distance between the true and the sampled parameters, with increasing samples obtained through the episodes this distance keeps decreasing. An upper bound on the number of sub-optimal steps TSEB takes before it learns the true model parameters is discussed in Theorem 3.3.1.

Definition 3.3.1. *Variance Bounds: Let $X = X_1 + X_2 + ... + X_n$, where X_i's be independent and $X_i \in [0,1]$ for each $i \in n$. Let $\mathbb{E}[X] = \mu$ and $Var[X] = \sigma^2$. Then for ϵ >0, which is defined as $\frac{t}{n}$,*

$$P\left(X \geq \mu + \frac{t}{n}\right) \leq e^{\frac{-\left(\frac{t}{n}\right)^2}{4\sigma^2}} \tag{3.4}$$

This is an extension of the Chernoff bounds (Herman, 1952) in a known variance setting (Dubhashi and Panconesi, 2009). Variance-based concentration measure has a slower decay rate than Bernstein inequality (Bernstein, 1924). Still we use this bound to account for the moderate increase in sample size that may be caused by the variance in the computation of exploration bonus.

Theorem 3.3.1. *For a given value of $\epsilon > 0$ and $0 \leq \delta \leq 1$, the number of sub-optimal steps that TSEB takes is upper bounded by M = $\mathcal{O}\left(\frac{SAf_0(K_r,\gamma)}{\epsilon^2}\right)$ with probability 1-δ.*

Proof. Consider any state-action pair, $(s, a) \in S \times A$. Let the rewards obtained over the visits to state s by an agent and selecting action a in that state be represented by a sequence, $(R)_i$. Let $\mathbb{E}[(R)_i] = R^*$, and $Var[R_i] = \sigma^2$, using Definition 3.3.1 and replacing $\frac{t}{n}$ with ϵ,

$$P\left(\frac{1}{n}\sum_{i=1}^{n} R_i \geq R^* + \epsilon\right) \leq e^{\frac{-\epsilon^2}{4\sigma^2}} \tag{3.5}$$

As we are interested in an upper-bound, δ, for the probability term in Equation 3.5, the inequality in Equation 3.5 can be re-written with the inclusion of the upper bound as,

$$\sigma^2 \geq \frac{\epsilon^2}{4\log\frac{1}{\delta}} \tag{3.6}$$

The above equation expresses the relation between (ϵ, δ) and (σ^2). The exploration bonus $\rho(s, a)$ (the subscript t is dropped for convenience of expression) computed as in Equation 3.3 for a state-action pair in an episode can be understood as a cumulative sum of differences between the sampled state parameters and an unbiased estimate of the expectation of parameters till that episode. This is because of the K_r factor in $f_{s,a}(K_r, \gamma)$ which is the max norm difference between the reward functions of the sampled and the expected MDP. Hence, the exploration bonus $\rho(s, a)$ can also be expressed as follows,

$$\rho(s, a) = \frac{1}{n-1}\sum_{i=1:n} ||\hat{\mathbb{E}}_i[\theta_{s,a}] - \theta^i_{s,a}||_\infty \tag{3.7}$$

where, $\theta^i_{s,a}$ is the sampled parameter in an episode i and $\hat{\mathbb{E}}_i[\theta_{s,a}]$ is an unbiased estimate of the mean computed from the samples collected till the episode i. By the equivalence of norms in finite dimensional Euclidean spaces, $||.||_\infty \leq \sqrt{C}||.||_2$ (for some constant C), we justify the use of variance based measure (Definition 3.3.1) to derive an upper bound on the number of samples. σ^2 in Equation 3.6 is an unbiased estimate of the summation of differences in *value* of states between the true and sampled MDP, which is upper bounded by $f_{s,a}(K_r, \gamma)$. Hence,

$$\frac{f(K_r, \gamma)}{n_{s,a}} \geq \frac{\epsilon^2}{4\log\frac{1}{\delta}} \tag{3.8}$$

Where $n_{s,a}$ be the number of visits to a state-action pair (s, a).

Let $f_0(K_r, \gamma)$ be the expected initial distance of the sampled MDP from the true MDP with respect to the prior. In an uninformative prior setting, f_0 assumes a uniform prior over the models and in expectation is the maximum over the f-values computed in the algorithm. Hence,

$$\frac{f_0(K_r, \gamma)}{n_{s,a}} \geq \frac{\epsilon^2}{4 \log \frac{1}{\delta}} \tag{3.9}$$

Further, it is important to factor in the effect of λ, the trade-off parameter, in the PAC bound as it affects the actions taken. To understand the effect of λ, consider the ratio between the action probabilities using the modified Bellman update with a non-zero λ and λ=0. The ratio expresses the role of λ in action probabilities directly affecting the action selection. Let η be the ratio quantifying the expected increase in samples with a non-zero λ,

$$\eta = \frac{e^{(\lambda R + (1-\lambda)\rho + \gamma P(s'|s)V(s'))}}{e^{(\rho + \gamma P(s'|s)V(s'))}} \tag{3.10}$$

or

$$\eta = e^{\lambda(R-\rho)} \tag{3.11}$$

Lemma 3.3.1. *Popoviciu's inequality on variance (Popoviciu, 1935): Let X be a bounded random variable such that $X \in [a, b]$. Then Popoviciu's inequality on variance states,*

$$\sigma^2 \leq \frac{1}{4}(b-a)^2 \tag{3.12}$$

where σ^2 is the variance of the random variable X.

In a normalized reward setting, ρ, which is analogous to the variance of reward (Equation 3.7 and the explanation followed) can be upper-bounded by 1 using Lemma 3.3.1 and R can be loosely upper bounded by 2. Using the earlier statements, η can be upper bounded by e^{λ}. Thus, a λ value closer to 1 indicates that the agent is following a conservative policy that leads to a higher sample complexity.

Combining Equation 3.9 and Equation 3.11 we can derive the upper bound on the visits as an effect of λ. The updated expression is as follows,

$$n_{s,a} \leq e^{\lambda} \frac{4 f_0(K_r, \gamma)}{\epsilon^2} \log \frac{1}{\delta} \tag{3.13}$$

The total sample complexity, M, for a given (ϵ, δ) pair on an MDP with $|S|$ and $|A|$ being the cardinality of its state and action spaces is given by,

$$M \leq e^{\lambda} \frac{4|S||A| f_0(K_r, \gamma)}{\epsilon^2} \log \frac{1}{\delta} \tag{3.14}$$

$$M = \mathcal{O}\left(\frac{|S||A| f_0(K_r, \gamma)}{\epsilon^2}\right) \tag{3.15}$$

Equation 3.15 provides an upper bound on the number of sub-optimal steps of TSEB algorithm. It is also interesting that this bound is dependent on the initial estimates, prior, of the model parameters– $f_0(K_r, \gamma)$. ■

Algorithm	**PAC- Bounds**
MBIE	$O\left(\frac{S^2 A R_{\max}^5 ln^3 \frac{SAR_{\max}}{(1-\gamma)\epsilon\delta}}{(1-\gamma)^6 \epsilon^3}\right)$
BEB	$O\left(\frac{SAH^6}{\epsilon^2} \log \frac{SA}{\delta}\right)$
Variance Based	$O\left(\frac{\gamma^2 S^4 A^2}{\delta \epsilon^2 (1-\gamma)^2}\right)$
TSEB	$O\left(\frac{e^{\lambda} SA f_0(K_r,\gamma)}{\epsilon^2} \log \frac{1}{\delta}\right)$

Table 3.1: The table shows the existing PAC bounds for a model based learning setting. The cardinality of the sets of states and actions is defined by S and A in the table.

Table 3.1 shows a comparison of the PAC guarantee of TSEB against other existing PAC guarantees. The $f_0(K_r, \gamma)$ can be upper bounded by $\frac{R_{max}}{1-\gamma}$. A direct comparison of TSEB's guarantee with the rest shows that it is at least SR^4 times better than MBIE's, H^5 times better than BEB's and S^3A^2 times better than Variance based approach. Hence, TSEB's PAC bound is tighter than the earlier known bounds for exploration bonus approaches.

3.4 Empirical Analysis

In this section, we empirically analyze the performance of TSEB on two simulated domains, *Chain world* (Kolter and Ng, 2009) and *Queuing world* (Gopalan and Mannor, 2015), with different values of the trade-off parameter, $\lambda \in [0, 1]$ averaged for 50 experiments.

3.4.1 Chain World

The chain domain has five states and two actions a and b. The agent has access to both the actions from every state. With probability 0.2 the agent ends up taking the other action than the one selected. The transitions and rewards are as shown in Figure 3.1 except for the first state, which has a stochastic reward sampled from $\mathcal{N}(0.2, 0.5)$. We fix γ as 0.8.

Figure 3.2a shows the evolution of the $f_{s,a}(K_r, \gamma)$ with number of episodes and Figure 3.2b shows the evolution of the corresponding upper bound for different choices

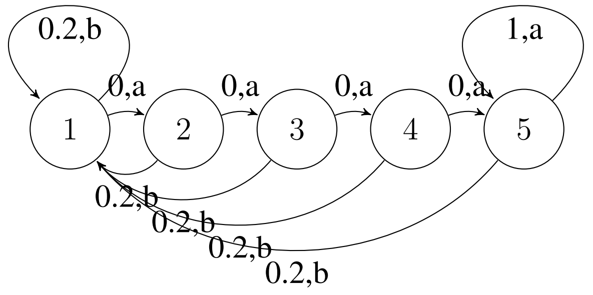


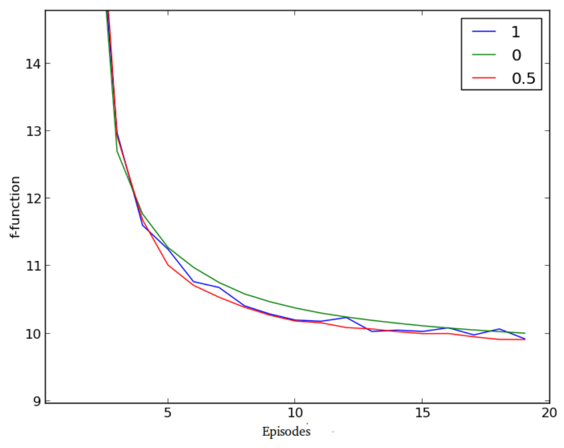
Figure 3.1: Chain World Domain. The arc labels denote expected reward and action.

of λ. As can be seen from Figure 3.2a, $f_{s,a}(K_r, \gamma)$ quickly converges to its minimal value with fastest convergence being achieved for $\lambda = 0.5$. This indicates that the trade-off parameter indeed makes a difference to the performance of the algorithm, and settling on one extreme is not always the optimal choice. Figure 3.2b shows that the estimated values of the parameters is closer to the true parameter values when $\lambda = 0.5$ as opposed to the two extreme values. The poor performance when $\lambda = 0$ can be attributed to the high variance in the estimated exploration bonus and the agent's total reliance on it to guide its trajectories. In the case of $\lambda = 1$ (TSEB acts as TS), the performance is poor due to the lack of useful samples owing to the conservative policy chosen by TS. On the other hand, combining the two extremes with $\lambda = 0.5$ provides a better performance than relying only on one of them.

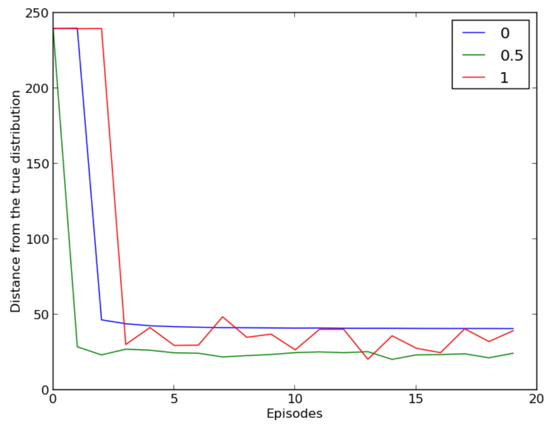
Despite being aggressive in its exploration, TSEB agent managed to come very close to the regret optimal performance of TS. Figure 3.2c shows a comparison plot of the average reward in a time step for the three cases considered. This is partly because of the environment and, hence, cannot be generalized across domains. The periodic drop to a small value in Figure 3.2c is due to the reset to the initial state.

3.4.2 Queuing World

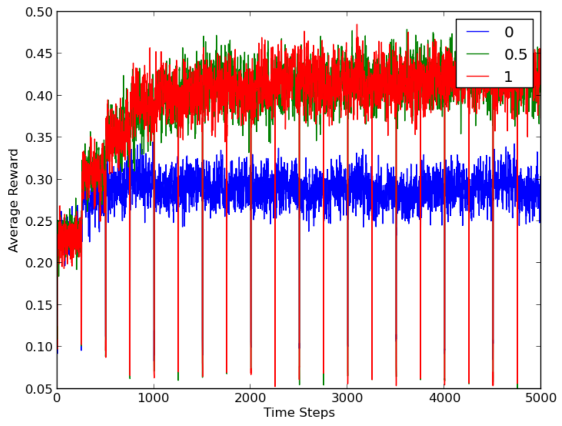
The description of the world is as follows. The states of the MDP is the number of packets in the queue at any given time, i.e., S=$\{0, 1, 2, ..., 50\}$ and at each step the agent receives a new packet with probability 0.7. At any given step, the agent has to take one of the two actions: Action 1 (SLOW service) and Action 2 (FAST service), i.e., A=$\{1, 2\}$. Applying SLOW (resp. FAST) service results in serving one packet from the queue with probability 0.3 (resp. 0.8) if it is not empty, i.e., the service model is Bernoulli(μ_i) where μ_i is the packet processing probability under service type $i = 1, 2$.



(a) Convergence of $f_{s,a}(K_r, \gamma)$, for different λ values in Chain world.



(b) Convergence of upper bound on $f_{s,a}(K_r, \gamma)$ for different λ values in Chain world.



(c) Average reward in Chain world.

Figure 3.2: Experiment results on Chain World domain

Actions 1 and 2 incur a per-instant cost of 0 and -0.25 units respectively. In addition to this cost, there is a holding cost of -0.1 per packet in the queue at all times. The system gains a reward of $+1$ units whenever a packet is served from the queue.

λ	**Queuing World**	**Chain World**
0.1	-5061	**1963.74**
0.2	-5038	1951.72
0.3	-5048	1944.65
0.4	-5051	1954.06
0.5	-5042	1956.22
0.6	-5040	1955.14
0.7	-5062	1953.99
0.8	-5038	1940.77
0.9	-5026	1934.99
1.0	**-5023**	1942.63

Table 3.2: Average cumulative reward for different λ in the test domains

We also experimented on this world with the same values of λ as used in the first experiment. Table 3.2 shows that the maximum performance was achieved under different λ values in the two worlds. We don't evaluate the cumulative reward when $\lambda = 0$. As seen from Figure 3.2c, the performance when $\lambda = 0$ is similar to random action selection. This is because, in this case, the agent relies only on exploration bonus which becomes insignificant as the number of visits becomes large. This leads to the agent converging to a value function that is 0 for all state-action pairs. This uninformative constant value function doesn't show the effect of λ on the agent's performance.

The results can be further interpreted considering the characteristics of the two worlds. The Chain world doesn't offer negative rewards to the agent and exploration seem to pay off well for the agent. But relying only on exploration bonus, λ=0, doesn't let it converge to the optimal policy. On the other hand, in the Queuing world, the agent receives negative rewards, which inhibits the agent from being explorative. Also the lesser variance in the reward function in this case resulted in $\lambda = 1$ accumulating a better cumulative reward than the other values. This explains the agent accumulating better rewards in Queuing world despite the agent not being exploration centric. To summarize, the results portray the need for tuning λ value for different domains (best performance recorded for different λ values in the two worlds) justifying one of the claims of having a tunable parameter. This could not be better justified had the maximum performance was achieved for any particular value of λ in both the worlds.

CHAPTER 4

STRUCTURING FINITE-EPISODE EXPLORATIONS IN MARKOV DECISION PROCESSES

4.1 Introduction

In the previous chapter we looked at explore-exploit dilemma in RL from an asymptotic agent's perspective. In this chapter, we look at this dilemma from an RL agent that can exist only for a finite number of episodes. We refer to such RL agents as *finite-episode agents*. A finite-episode agent has the need to trade-off the amount of exploration it engages in versus exploiting the partial knowledge acquired thus far. Such finite-episode agents have been studied reasonably well under the non-sequential setting in the stochastic multi-arm bandits (MAB) literature (Budgeted bandits (Bubeck *et al.*, 2009)). But to the best of our knowledge, little has been done under the sequential MDP setting.

The objective of such a finite-episode agent is to balance the need to learn new things about the environment with the primary objective of maximizing the total cumulative reward over its lifetime. To address this dilemma of model-based RL agents (Section 1.1.2) with finite episodes, in this chapter we propose an exploration framework, *Structuring Finite-Episode Exploration* (SFEE). The model based RL problem has received a great deal of attention from an asymptotic agent's perspective that led to a number of theoretical results for the agent's performance.

Unfortunately, these results are of little practical relevance to applications where the agents must learn in finite time. In such cases the explore-exploit dilemma is significant as the choice between the two can make a drastic difference in the total return. Ultimately, we propose that the optimal behavior, under some moderate assumptions, for a finite lifetime agent is achieved by stopping the exploration after a certain number of episodes and adopt a possibly sub-optimal exploitation policy.

The major contributions of this chapter are as follows:

- Proposing a framework that can adapt asymptotic model learning algorithms for a finite-episode setting.

- Proposing Gain value, which aids in deciding between explore and exploit.
- Providing PAC and regret guarantees for the proposed framework.
- Providing empirical analysis of performance of the proposed framework.

The rest of this chapter is structured as follows. Section 2 introduces SFEE framework and defines *gain* parameter and a few other parameters of interest upon which the decision to explore or exploit depends on. Section 3 discusses that the right policy for a finite-episode agent is to explore consecutively for some number of episodes and permanently switching to exploitation. Section 4 discusses on importing the exploration algorithm's performance guarantees into SFEE and the regret guarantees of the framework. Empirical demonstration of the framework in two grid worlds is provided in Section 5.

4.2 Exploration for Finite-Episode Lifetime Agents

The SFEE agent is provided with two policies: an exploit policy, π^x, and an explore policy, π^e. In an episode, SFEE agent chooses either the *exploit policy*, which is a greedy behavior with respect to the agent's estimated belief over the model till that episode, or the *explore policy*, which is the behavior directed by any of the existing model-based exploration algorithm (like R-max (Brafman and Tennenholtz, 2003), BEB (Kolter and Ng, 2009) etc.). The SFEE agent decides between the two policies at the beginning of every episode. Once chosen, the per step action choices in that episode are governed by the chosen policy. The goal of a SFEE agent, then, is to choose between the two policies appropriately such that the total reward obtained over its lifetime of T episodes is maximized.

We make the following assumptions about the problem:

- The MDP is *ergodic* i.e., each state in the MDP is visited infinitely often (*recurrent*) and without any systematic period (*aperiodic*).
- The size of the MDP is very large compared to the length of an episode.
- Every episode is of finite length.
- Each exploration episode results in the agent obtaining data for at least one *unknown* state, if one exists.

- An exploitation episode ends if the agent encounters a new state, as this will be a rare event and the agent by then has a better knowledge (within the limits of its lifetime) of the world to forgo learning a new state.

It is useful to familiarize with the following descriptions for the states to understand the working of the framework. The states are initially assumed to be *unknown* – the agent has no knowledge of the states. When the state is visited at least once, the state becomes *visited* and once its parameters are estimated with high confidence the state becomes *known*.

4.2.1 Gain Parameter

The agent's decision to explore or exploit is defined by the *gain* parameter. The decision of the agent in an episode, at a high-level, should be to *explore* when it may benefit by improved return in the future episodes by exploring in that episode and to *exploit* otherwise. The decision parameter, *gain*, depends on a few other parameters that are defined below.

Let S_t be the set of start states in the MDP, I be the set of states that are known to the agent, and Ω be the set of states that the agent will get to know by exploring in the current episode.

Definition 4.2.1. *The* Expected Return Margin *(ERM) is the increase in future per-episode return obtained by the exploit policy, given one additional exploration episode:*

$$ERM = \mathbb{E}_{s_0 \sim S_t}\left[V^{\pi^x}_{I \cup \Omega}(s_0)\right].$$

ERM estimates the benefit of augmenting newly known states (Ω) after an exploration episode to the states that the agent already knows (I) ; this benefit is expressed as the expected change in the return of the subsequent exploit policy. The notation, $V^{\pi^x}_{I \cup \Omega}(s_0)$, denotes the value of a state s_0 by following policy π^x over the set of states already known(I) *and* the states the agent will know after a single explore episode (Ω) – return of the subsequent exploit policy.

Definition 4.2.2. *The* Exploit Episode Return *(EER) is the return obtained by executing the exploit policy for a single episode, without further exploration:*

$$EER = \mathbb{E}_{s_0 \sim S_t}\left[V^{\pi^x}_{I}(s_0)\right].$$

EER is simply the value function of the exploit policy, averaged over the start states.

Definition 4.2.3. *The Exploration Loss (L) is the loss incurred by an agent by following explore policy rather than the exploit policy:*

$$L = \mathbb{E}_{s_0 \sim S_t}\left[V_I^{\pi^x}(s_0)\right] - \mathbb{E}_{s_0 \sim S_t}\left[V_{I \cup \Omega}^{\pi^e}(s_0)\right].$$

L expresses the single-episode loss of the agent for opting to explore than to exploit.

We combine these quantities to define *Gain*:

Definition 4.2.4. Gain *is the expected improvement in return over the agent's remaining T_r episodes, after exploring in the current episode and choosing to exploit from then on.*

$$G = T_r \times (ERM - EER) - L. \tag{4.1}$$

Computing *gain* in an episode takes into account the improvement on future returns by following π^e in that episode, the return of π^x in that episode and the loss due to exploration in that episode. The *gain* parameter, taking these quantities into account, helps in deciding between *explore* and *exploit* policies in an episode. SFEE chooses to explore if *gain* is positive, and exploits otherwise.

4.2.2 Switching to Exploitation

The gain estimated in an episode only supports the *explore/exploit* decision made in *that* episode. In this section, we show that under some mild assumptions the right policy for a finite-episode agent is to explore consecutively for some number of episodes, before permanently switching to exploitation and that can be achieved by acting according to the gain parameter estimated in an episode.

This problem is closely related to the optimal stopping problem described in (Weber, 1975). The problem, hence, boils down to proving the existence of a stopping point for the explore episodes. We prove the existence of the stopping point and also show that *gain* parameter is a sufficient quantity to decide the stopping point for the explore episodes.

As the number of states initially known to the agent is small and increases with the number of exploration episodes, we make a mild assumption that the loss due to exploration will be low initially (as the agent visits new states and improves its belief over the model parameters) and then increases (with more number of exploration episodes the agent has a reasonable belief over the parameters and the exploration episodes bring in less information with following a sub-optimal policy) as a function, α, that is *weakly* monotonic in episode number. Hence the loss at step $i + 1$ can be represented as:

$$l_{i+1} = l_i + \alpha(i + 1), \tag{4.2}$$

where l_i is the loss in the ith episode.

Similarly, we assume that the expected improvement over future returns following exploration policy decreases with increasing episode number—explore policy shows diminishing improvements in future returns with increase in episodes. Let $(ERM - EER)$ in an episode i be represented by g_i. The additive function ω characterizes the increase in future returns due to exploration. Hence the gain at step $i + 1$ can be represented as :

$$g_{i+1} = g_i + \omega(i + 1). \tag{4.3}$$

This function should decrease with episodes, as the information gained in every episode decreases. Gain at k^{th} episode, g_k, can also be represented as:

$$g_k = g_0 + \sum_{i=1}^{k} \omega(i), \tag{4.4}$$

and loss l_k represented likewise as:

$$l_k = l_0 + \sum_{i=1}^{k} \alpha(i), \tag{4.5}$$

Considering that an agent behaves greedy with respect to the *gain* term defined in Equation 4.1, we show in the following theorem that greedy selection between explore and exploit policies will lead to a finite sequence of explore episodes and switching to exploit from then on. We thus show that the Gain value computed is sufficient to decide between explore and exploit policies.

Theorem 4.2.1. *For an agent with a T-episode lifetime, if exploring at episode $(k + 1)$ is useful, then so is exploring at episode k.*

Proof. We are given that exploration is useful at the $(k+1)$th episode, so we know that:

$$(T-k-1)g_{k+1} - l_{k+1} \geq 0 \tag{4.6}$$
$$(T-k)g_{k+1} - g_{k+1} - l_k - \alpha_k \geq 0 \tag{4.7}$$
$$(T-k)(g_k + \omega_k) - g_{k+1} - l_k - \alpha_k \geq 0 \tag{4.8}$$
$$(T-k)g_k - l_k \geq g_{k+1} + \alpha_k - (T-k)\omega_k. \tag{4.9}$$

To prove that exploration is also useful in step k, it is sufficient to prove that:

$$g_{k+1} + \alpha_k - (T-k)\omega_k \geq 0, \tag{4.10}$$

which can be re-written as:

$$g_0 + \sum_{i=1}^{k} \omega_i + \alpha_k - (T-k)\omega_k \geq 0 \tag{4.11}$$
$$g_0 + \alpha_k + \sum_{i=1}^{k-1} \omega_i \geq (T-k-1)\omega_k. \tag{4.12}$$

By using induction on episodes, we show that Equation 4.12 holds at every episode. Let k be any episode between 1 and T. For the base case, consider the $(T-1)^{th}$ episode was *explore*. Then, this should mean that $(T-2)^{th}$ episode was *explore* indeed. Substitute $k = T-2$ in Equation 4.12

$$g_0 + \alpha_{T-2} + \sum_{i=1}^{T-3} \omega_i \geq (T-T+2-1)\omega_{T-2} \tag{4.13}$$
$$g_0 + \alpha_{T-2} + \sum_{i=1}^{T-3} \omega_i \geq \omega_{T-2} \tag{4.14}$$
$$g_0 + \alpha_{T-2} + \sum_{i=1}^{T-3} \omega_i \geq \omega_{T-2}. \tag{4.15}$$

By assumption α and ω are monotonically increasing and decreasing functions respectively and the statement holds for the base case.

Let us assume that this holds for some k = $t \leq T-2$,

$$g_0 + \alpha_t + \sum_{i=1}^{t-1} \omega_i - (T-t-1)\omega_t \geq 0. \tag{4.16}$$

Then, for k = $t-1$, the following equation has to be proven.

$$g_0 + \alpha_{t-1} + \sum_{i=1}^{t-2} \omega_i \geq (T-t-2)\omega_{t-1} \geq 0 \tag{4.17}$$

By the initial assumption $\alpha_{t-1} = \alpha_t + c_1$ and $\omega_{t-1} = \omega_t - c_2$ where $c_1, c_2 \in \mathbb{R}^+$, we rewrite Equation 4.17 as,

$$g_0 + \alpha_t + c_1 + \sum_{i=1}^{t-2} + \omega_{t-1} - (T-t-1)(\omega_t - c_2) \geq 0 \tag{4.18}$$

Re-arranging Equation 4.18

$$g_0 + \alpha_t + \sum_{i=1}^{t-1} \omega_i - (T-t-1)\omega_t \geq -c_1 - (T-t-1)c_2$$

From Equation 4.16, the LHS of the above equation is positive and RHS of the equation is clearly negative.

Hence, this holds for any for all t between 1 and T.

■

Corollary: The proof of this theorem can be extended to a generic case of $k-1$ explore episodes being useful if k explore episodes are useful. Had the $k-1^{th}$ episode been exploit and the k^{th} episode is explore, then it would mean that the exploration episode was not useful in $k-1^{th}$ episode which is a contradiction with Theorem 4.2.1. This shows that there exists an optimal stopping time for the explore episodes.

Since SFEE explores until a certain episode and exploits then on, it can be inferred that SFEE avoids premature exploit episodes and chooses to exploit a mature knowledge of the world to reap better pay offs. It should be noted that a finite stopping time in exploration is possible with the agent exploring only a fraction of the large state space and halting with the agent not *knowing* the entire state space. This helps in achieving a behavior that is lifetime optimal and may not be the true optimal. We discuss more on this in the theoretical guarantees section. Algorithm 3 describes the pseudocode of SFEE framework.

4.3 Bounding the Loss in Estimated Gain

As the agent explores, it maintains estimates of the true model parameters (P^* and R^*). The gain, G, is computed from estimates of the model parameters. The true gain, G^*, can only be computed if the agent had access to the true model parameters. This results in a loss in estimating gain, which can be attributed to the sum of individual losses in estimating ERM and EER.

Algorithm 3: Structuring Finite Time Exploration (SFEE)

```
Input: $S_t$, $I$, $\Omega$, $\pi^e$, $\pi^x$, $V_I^{\pi^x}$, $V_I^{\pi^e}$, $V_{I\cup\Omega}^{\pi^x}$ and $V_{I\cup\Omega}^{\pi^e}$.
 Define: N  ▷ Total number of episodes.
     T$_r$  ▷ Number of remaining episodes.
     $V_I^{\pi^e}$  ▷ Value function over I by following $\pi^e$
     $V_I^{\pi^x}$  ▷ Value function over I by following $\pi^x$
     $V_{I\cup\Omega}^{\pi^e}$  ▷ Value function over $I \cup \Omega$ by following $\pi^e$
     $V_{I\cup\Omega}^{\pi^x}$  ▷ Value function over $I \cup \Omega$ by following $\pi^x$
for e in range (N) do
    if $e > k$ then
        ERM $\leftarrow \mathbb{E}_{s_0\sim S_t}\left[V_{I\cup\Omega}^{\pi^x}(s_0)\right]$
        EER $\leftarrow \mathbb{E}_{s_0\sim S_t}\left[V_I^{\pi^x}(s_0)\right]$.
        L $\leftarrow \mathbb{E}_{s_0\sim S_t}\left[V_I^{\pi^x}(s_0)\right] - \mathbb{E}_{s_0\sim S_t}\left[V_{I\cup\Omega}^{\pi^e}(s_0)\right]$
        G $\leftarrow$ T$_r$ $* (ERM - EER) - L$
    end
    if $G > 0$ and $e > k$ then
        Follow $\pi^x$ until an unknown state is reached
    end
    else
        Follow $\pi^e$
        Collect samples (s, a, s', r)
    end
    Construct $\Omega$
    Update the estimated model parameters
    Compute $\pi^e$, $\pi^x$, $V_I^{\pi^x}$, $V_I^{\pi^e}$, $V_{I\cup\Omega}^{\pi^x}$ and $V_{I\cup\Omega}^{\pi^e}$
end
```

Let the value function computed using the true model parameters (this is not known; we are using this to get an upper bound on the loss) be $\hat{V}$.

$$G^* - G = T_r * [(EER^* - EER) + (ERM^* - ERM)] \tag{4.19}$$

$$loss = T_r * \mathbb{E}_{s_0\sim S_t}\left[| \left(\hat{V}_I^{\pi^x}(s_0) - V_I^{\pi^x}(s_0)\right) + \left(\hat{V}_{I\cup\Omega}^{\pi^x}(s_0) - V_{I\cup\Omega}^{\pi^x}(s_0)\right) | \right]. \tag{4.20}$$

The bounding *loss* is different from the Exploration loss (L) discussed earlier.

Lemma 4.3.1. *The gain estimation loss is directly proportional to the variance in the model parameters, K_p and K_r.*

Proof. Let the true MDP, M^*, be defined by $< S', A, P^*, R^*, \gamma >$ and the sampled MDP, M, be defined by $< S', A, P, R, \gamma >$. M^* is defined with the true parameters of the model and the structure constructed from the states visited by the agent, S'. Considering the bijection between the state space of M^* and M, the policy learnt in the image M, when lifted to M^* suffers a loss owing to the dissimilarities in the model parameters.

Let us define a few more parameters. In an episode i,

$$k_r^i = \max_{s \in S',\ a \in A} |R^*(f(s), a) - R(s, a)|, \tag{4.21}$$

$$k_p^i = \max_{s \in S',\ a \in A} |P^*(f(s), a, f(t)) - P(f(s), a, f(t))| \tag{4.22}$$

The maximum over N episodes for k_p and k_r are given by,

$$K_r = \max_{i=1}^{N} k_r^i \tag{4.23}$$

$$K_p = \max_{i=1}^{N} k_p^i \tag{4.24}$$

which provides an unbiased estimation of the expected variance of transition and reward parameters respectively (ref. Theorem 3.3.1). The range of the rewards in the image MDP M is defined as:

$$\delta_{r'} = \max_{s \in S\ a \in A} R(s, a) - \min_{s \in S\ a \in A} R(s, a). \tag{4.25}$$

From the theorem in §4 in (Ravindran and Barto, 2004), the max norm difference between the value functions learnt in M, V, and M^*, V^*, is given by,

$$||V^* - V|| \leq \frac{2}{1-\gamma}\left[K_r + \frac{\gamma}{1-\gamma}\delta_{r'} K_p\right]. \tag{4.26}$$

This can be extended to upper bound the *loss* expression in Equation 4.20. Let the K_p, and K_r to estimate the difference between EER^* and EER be K_p^{EER}, and K_r^{EER}. Similarly, for ERM, let these parameters be K_p^{ERM}, and K_r^{ERM}. Upper bounding the difference between the δ_r' by C, Equation 4.20 can be rewritten as

$$loss \leq \frac{2}{1-\gamma}\left[\Theta_{K_r} + \frac{\gamma}{1-\gamma} C \Theta_{K_p}\right] \tag{4.27}$$

where, $\Theta_{K_r} = K_r^{EER} + K_r^{ERM}$ and $\Theta_{K_p} = K_p^{EER} + K_p^{ERM}$.

The gain parameter computed in an episode biases an agent's decision to explore or exploit. Hence the *loss* is interesting to calculate, as it gives an upper bound on the loss due to *lifting* of policy. From the expression it is clear that the *loss* is directly dependent on the variance in the model parameters. One way to mitigate this loss in estimation is to have informative prior on the model parameters. A good prior that has a small variance and is closer to the true parameters will help in minimizing this error, which is difficult to get in most of the real world problems. ■

4.4 Importing PAC guarantees

PAC analysis provides an upper bound on the number of *non* ϵ-optimal steps an agent takes before converging to an ϵ-optimal policy. PAC guarantees for SFEE is obtained by importing the guarantees of the asymptotic exploration algorithm accessed by the framework. The decision made by the framework is dependent on the belief over the estimated model parameters, which in-turn is dependent on the exploration algorithm adopted in the framework. We provide a tool to estimate the framework's PAC guarantee using the PAC guarantee of the exploration algorithm.

Consider the scenario where an agent follows a sequence of k explore episodes in its *T-lifetime*, and the episodes contain N_t-time-steps on average. Let an exploration algorithm, A_E adopted in the framework provide an (ϵ, δ) guarantee for N number of samples. The guarantee states that for the converged solution to be ϵ-optimal with $1 - \delta$ probability, the bound requires *N* samples. Now, the analysis forks into two. One, where the number of samples collected during the explore episodes is more than *N*. i.e. $kN_t \geq N$ and the other, where the number of samples is less than *N*.

In the former case, the PAC guarantee of the exploration algorithm holds. By the definition of PAC guarantee, the (ϵ, δ) guarantee is provided for the expected number of samples. In the latter case, plug in the number of samples in the PAC expression to obtain a (ϵ, δ) curve. As it is relatively costlier to improve on the ϵ, we pick a smaller value of δ, δ_1, from the curve, such that $\delta_1 > \delta$. Now, by plugging the number of samples, and δ in the guarantee of the algorithm, we get an ϵ_1. This provides an (ϵ_1, δ_1) guarantee in expectation for A_E in the finite-episode setting with kN_t number of samples.

4.5 Discussion on Regret Guarantees

The regret analysis in this section provides an upper bound on the total loss that an agent accrues by following a policy that the framework has converged to. An asymptotic agent will converge to the true MDP and will follow an optimal policy asymptotically. As a finite-episode agent runs on a budget of episodes, it is not fair to estimate regret with respect to the optimal policy. The finite-episode agent is inhibited by the setting (very large state space) to converge to an optimal policy. But, it can try to approach an optimum that can be achieved within its lifetime. The true optimal and the optimal with respect to the agent's lifetime differ depending on the cardinality of the states in an MDP, and the lifetime of the agent. The optimal policy is always better or equal in performance to a policy obtained in a bounded lifetime.

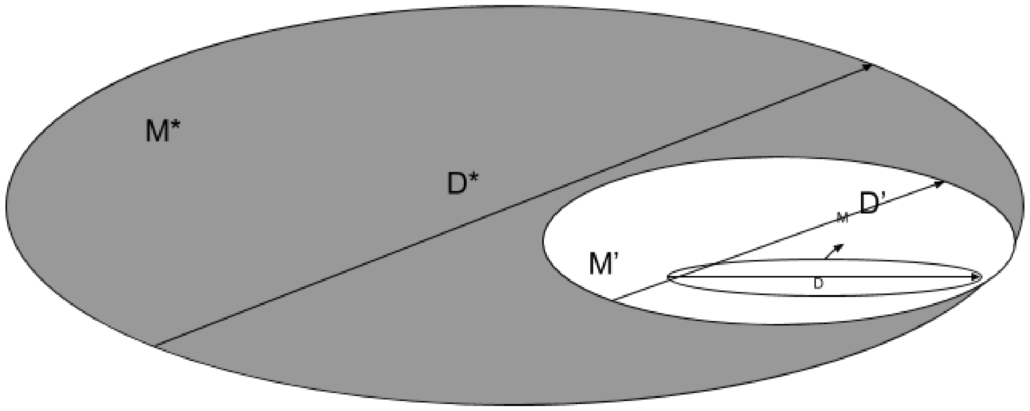


Figure 4.1: Pictorial description of regret comparison. The true MDP that an agent can learn asymptotically is M^*. The smaller lifetime prevents the agent from converging to M^* and at possible lets it learn M'. Often the stochasticity in transition hinders the agent from learning M' too and eventually the agent ends up only learning M. We refer M' as the lifetime optimal and M^* as the true/optimal MDP.

Consider an MDP, M, $< S, A, R, P, \gamma >$, $\gamma \in (0,1)$ that the agent gets to know in its lifetime. This is a small portion of the large MDP, which it may not get to explore entirely given its finite lifetime. Let the largest possible MDP constructed in its lifetime be M',$< S, A, R, P', \gamma >$. Let D and D' be the diameters of MDPs M and M' respectively, where the diameter of an MDP is defined as the maximum expected distance (expectation considered over all possible policies) between any two states in that MDP. The efficiency of an exploration strategy is quantified by the diameter of the estimated MDP. In general, smaller the diameter poorer is the exploration strategy.

The agent's regret with respect to the lifetime optimal policy, max policy in M', is estimated by the difference between the expected return of an episode of length corresponding to the diameter of the MDP, M, and the expected return of an episode of length corresponding to the diameter of the MDP, M'.

Assume the reward is distributed in [0,R_{max}]. Let $\mathcal{G}'$ and $\hat{\mathcal{G}}$ be the maximum possible return from M' and M respectively by following a policy as described earlier in both the MDPs. Let the expected lifetime of an agent be T episodes.

$$\Delta = \mathcal{G}' - \hat{\mathcal{G}}. \tag{4.28}$$

Let the discount factor be $\gamma \in (0,1)$. Then,

$$\mathcal{G}' \leq R_{max}(1 + \gamma + \gamma^2 + ... + \gamma^{D'-1}).$$

$$\hat{\mathcal{G}} \leq R_{max}(1 + \gamma + \gamma^2 + ... + \gamma^{D-1}).$$

Updating Equation 4.28,

$$\Delta \leq R_{max}(1 + \gamma + \gamma^2 + ... + \gamma^{D'-1}) - R_{max}(1 + \gamma + \gamma^2 + ... + \gamma^{D-1}) \tag{4.29}$$

$$\leq R_{max}\gamma^{D-1}(1-\gamma)^{-1}. \tag{4.30}$$

The diameter D of the known MDP can be loosely upper bounded as D $\leq$ 2L, where L is the expected length of a trajectory.

$$\Delta \leq \frac{R_{max}\gamma^{2L-1}}{(1-\gamma)}. \tag{4.31}$$

With T as the expected lifetime of the agent,

$$\mathbb{E}(Regret) \leq T \times \Delta \tag{4.32}$$

$$\mathbb{E}(Regret) \leq \frac{T \times R_{max}\gamma^{2L}}{(1-\gamma)}, \tag{4.33}$$

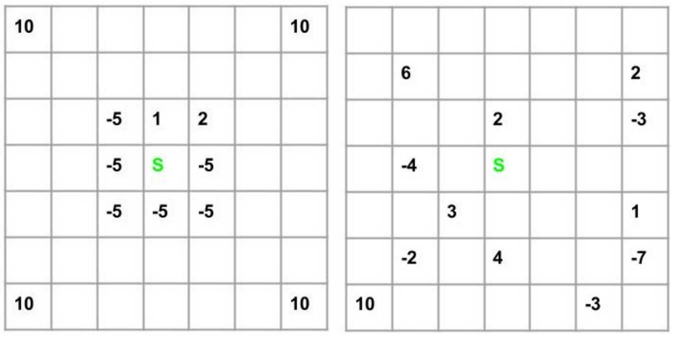
Equation 4.33 provides an expression for the notion of upper bound on the regret of a finite-episode agent with respect to its lifetime.

Equation 4.33 shows that the expected regret of a finite lifetime agent is upper bounded by a function of the length of an episode, which is a loose upper bound on the diameter of the MDP that the agent has constructed in its lifetime. As $\gamma \in (0, 1)$, the regret decays with increase in episode length. It is interesting to see the effect of increase in the number of episodes and the time-steps on the regret expression. As the episode length increases and reaches a very large number, the ergodicity of the MDP will lead to the convergence to the true MDP. But, the increase in number of episodes is going to be of little use when the number of time steps per episode is limited and fixed. This will still inhibit exploring large number of states in an episode. By Equation 4.33, the expected performance of an agent that learns a larger MDP is better than an agent that converges to a smaller MDP.

4.6 Empirical Results

To analyze the framework we conducted experiments in two 7×7 grid worlds (shown in Figure 4.2) with the center of each grid as the start state (S). The agent can move

North, **S**outh, **E**ast, or **W**est, and after each action receives a reward sampled from a Gaussian with the mean indicated in each grid, and a variance of 1.0. The results are averaged over 50 trials.



(a) World 1 *(b)* World 2

Figure 4.2: The two gridworlds used in our experiments.

We conducted three experiments as part of analyzing the framework. The first experiment compares the performance of SFEE (Algorithm 3) framework with various exploration algorithms adopted as the exploration component, π^e, against the same exploration algorithm applied directly on the world. In the second experiment we analyze the exploration pattern of SFEE agent. In the third experiment we analyze the total return after limited exploration to understand the need for switching to exploit policy in a finite-episode setting.

4.6.1 Experiment 1

The objective of this experiment is to compare the performance of various asymptotic algorithms when used directly, and the same algorithms when used as the exploration component in SFEE. An experiment has 200 episodes with 10 time-steps each. The exploration algorithms used are BEB (Kolter and Ng, 2009), R-max (Brafman and Tennenholtz, 2003), and TSEB. The results of the experiment are shown in Table 4.1.

World	Rmax		BEB		TSEB	
	Pure	SFEE	Pure	SFEE	Pure	SFEE
World 1	2.00	**6.69**	2.01	*2.57*	1.99	*2.61*
World 2	3.58	**6.82**	3.94	*6.69*	3.96	*6.68*

Table 4.1: Comparison of average episodic return in the two worlds, for various exploration algorithms, when used as-is or as the exploration component of SFEE. The results are averaged over 50 experiments.

Table 4.1 shows a comparison of pure asymptotic algorithms against itself adopted in SFEE framework. It can be seen from the table that irrespective of the worlds and algorithms, an algorithm performed better when it is adopted in the framework. This highlights two things, one, the asymptotic algorithms cannot manage finite-episode exploration and two, the proposed framework indeed adapts the asymptotic algorithms to obtain better returns in finite-episode setting. The following two experiments help in strengthening this result.

4.6.2 Experiment 2

This experiment shows the exploration pattern of an algorithm in a MDP to understand the working of the framework. Though the exploration pattern may differ with the algorithm adopted in the framework, the working of the framework can be explained considering the exploration pattern of any of these algorithm. We ran this experiment with TSEB.

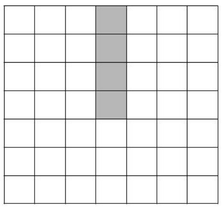
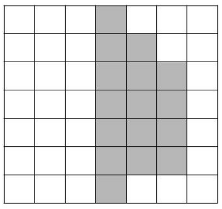
Figure 4.3 shows that the agent, after 10 episodes, got to know only a few states (shaded states) above the start state. The notion of *known* differs with each algorithm. In R-max, a state is considered to be *known*, if it is visited a polynomial number of times, B. In the case of TSEB, a state is *known* if the exploration bonus is close to 0.

After getting to know a few states and having visited a few others, the agent explores and learns a few more states in the episodes. This can be seen in Figure 4.3. The exploration saturates after 40 episodes and the agent follows a greedy policy. Though there is a reward 10 available in the bottom left corner, the lifetime of the agent inhibits it from learning a better policy to reach there. Given that the gain value is computed by factoring in the lifetime of the agent, the agent sticks to the greedy policy and exploits the known region of the state space.

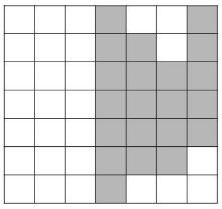
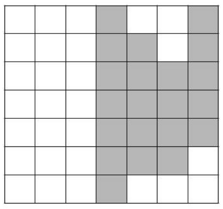
It is acceptable for a finite-time agent to converge to a policy before exploring a large portion of the state space.

4.6.3 Experiment 3

We argue that the prolonged exploration might not be useful for a finite-time agent. This is because, the agent's lifetime and episode length are finite and the diameter of

(a) After 10 episodes *(b)* After 35 episodes

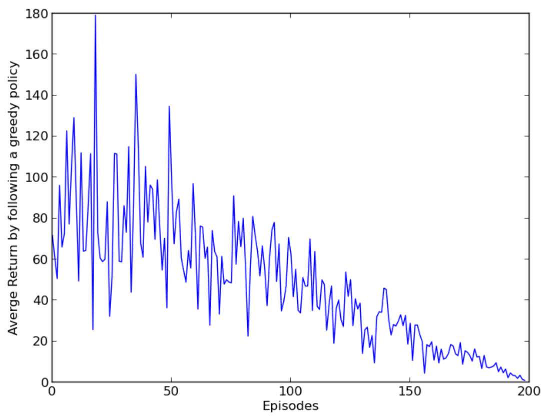
(c) After 40 episodes *(d)* Saturated known states

Figure 4.3: Exploration pattern of Finite-Episode agent in World 2.

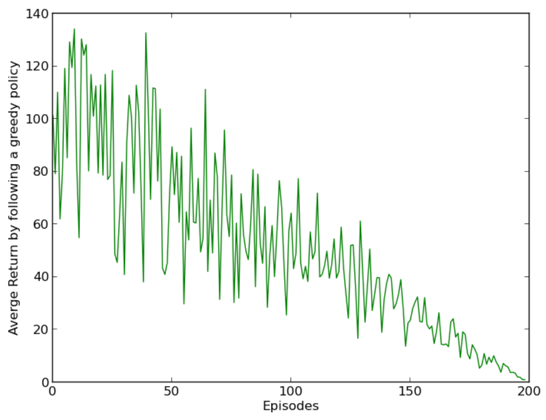
the MDP learnt by the agent is smaller compared to the size of the actual MDP. Such a scenario demands an agent to explore and converge to a max policy sooner in its finite-lifetime. The longer it takes, lesser will be the total return from the episodes following greedy policy.

An agent explores for i episodes and exploits thereafter. The graphs are shown in Figure 4.4 in which the Y-axis represent the average total return of exploit policy till the end of an agent's lifetime and X-axis shows the episode till which an agent was exploring with a strategy and started following the exploit policy then on. The curve gradually decays in the values of Y-axis showing that a finite-episode agent after a certain number of episodes is not going to be helpful for the total return. We observed a similar pattern with all the three algorithms. As for the experiments in World 1 (Figure 4.2a) with the adopted exploration algorithms, an average of $10 - 12$ explore episodes seemed to be the optimal point to terminate the exploration with a 200-episode lifetime. However, the optimal number of explore episodes may vary with a different exploration strategy depending on the world.

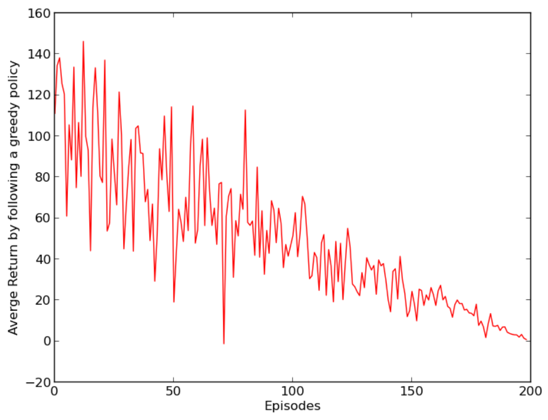
This experiment provides the reason for the SFEE algorithms outperforming in Experiment 4.6.1. The SFEE algorithms followed a greedy policy early on, whereas their pure counterparts kept exploring without factoring in the agent's lifetime.



(a) TSEB



(b) BEB



(c) Rmax

Figure 4.4: The graphs shows greedy policy's cumulative return after exploring for a certain number of episodes in World 1.

CHAPTER 5

CONCLUSION

This thesis elaborately discussed on two important aspects in model-learning approaches for RL. First, an asymptotic approach to model-learning which proposes a novel exploration bonus to accelerate model learning. We showed that the proposed exploration bonus provides a better learning guarantee than the earlier exploration bonus based approaches. We also use a trade-off parameter – a hyper-parameter – in TSEB to balance between explorative and exploitative policies. The approach, though, is at a certain level of maturity but can be improved in several ways. One possible improvement can be to establish a theory of "useful" exploration bonus to address *safe exploration* in RL. Further, the model estimation can be extended to a non-parameterized setting that will be useful for a wider range of problems.

Second, to address finite-episode exploration strategy for model based RL, we propose a framework (SFEE) to adapt asymptotic algorithms for the finite-episode setting. We theoretically show that the optimal behavior of a finite-episode agent is to stop exploring after a finite number of episodes and exploit thereafter. Though the proposed framework provides a better way to adapt asymptotic exploration algorithms for a finite-episode setting, there are a few challenges in the model learning approaches themselves. One of the primary issues is the dependence of sample trajectories on the prior of the model. This is primarily because of the lack of amenable ways to provide informative priors on model parameters and on exploration bonus over the states in a MDP. Providing informative priors will enhance the performance of exploration strategies and may help in early convergence. The informative priors will also improve the loss in gain value estimation as is clear from the established relation between the loss in *gain* estimation and parameter variance.

REFERENCES

1. **Abbasi-Yadkori, Y.**, **D. Pál**, and **C. Szepesvári**, Improved algorithms for linear stochastic bandits. *In Advances in Neural Information Processing Systems*. 2011.

2. **Agrawal, S.** and **N. Goyal**, Analysis of thompson sampling for the multi-armed bandit problem. *In Proceedings of the 25th Annual Conference on Learning Theory*. 2012.

3. **Agrawal, S.** and **N. Goyal** (2013). Thompson sampling for contextual bandits with linear payoffs. *30th International Conference on Machine Learning*.

4. **Anderson, C. W.**, Q-learning with hidden-unit restarting. *In Advances in Neural Information Processing Systems*. Morgan Kaufmann Publishers Inc., 1992.

5. **Asmuth, J.**, **L. Li**, **M. L. Littman**, **A. Nouri**, and **D. Wingate** (2009). A bayesian sampling approach to exploration in reinforcement learning. *Uncertainty in Artificial Intelligence*.

6. **Auer, P.**, **N. Cesa-Bianchi**, and **P. Fischer** (2002). Finite-time analysis of the multi-armed bandit problem. *Machine learning*.

7. **Auer, P.** and **R. Ortner**, Logarithmic Online Regret Bounds for Undiscounted Reinforcement Learning. *In Advances in Neural Information Processing Systems*. 2006.

8. **Auer, P.** and **R. Ortner** (2010). UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*.

9. **Barto, A. G.**, **S. J. Bradtke**, and **S. P. Singh** (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*.

10. **Barto, A. G.**, **R. S. Sutton**, and **C. W. Anderson** (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*.

11. **Bernstein, S.** (1924). On a modification of chebyshev's inequality and of the error formula of laplace. *Ann. Sci. Inst. Sav. Ukraine, Sect. Math.*

12. **Bertsekas, D. P.** and **J. N. Tsitsiklis**, *Neuro-Dynamic Programming*. Athena Scientific, 1996, 1st edition.

13. **Bonwell, C. C.** and **J. A. Eison**, *Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports.*. ERIC, 1991.

14. **Brafman, R. I.** and **M. Tennenholtz** (2003). R-max—A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. *The Journal of Machine Learning Research*.

15. **Bubeck, S.**, **R. Munos**, and **G. Stoltz**, Pure exploration in multi-armed bandits problems. *In Proceedings of the 20th International Conference on Algorithmic Learning Theory (ALT 2009)*, Lecture Notes in Computer Science. Springer, 2009.

16. **Chapelle, O.** and **L. Li**, An empirical evaluation of thompson sampling. *In Advances in neural information processing systems*. 2011.

17. **Dubhashi, D. P.** and **A. Panconesi**, *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

18. **Ghavamzadeh, M.**, **S. Mannor**, **J. Pineau**, and **A. Tamar** (2015). Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*.

19. **Gopalan, A.** and **S. Mannor**, Thompson sampling for learning parameterized markov decision processes. *In Proceedings of The 28th Conference on Learning Theory*. 2015.

20. **Gopalan, A.**, **S. Mannor**, and **Y. Mansour** (2013). Thompson Sampling for Complex Bandit Problems. *arXiv:1311.0466*.

21. **Herman, C.** (1952). A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*.

22. **Kaelbling, L. P.** (1990). Learning in Embedded Systems. Technical report, DTIC Document.

23. **Kearns, M. J.** and **S. P. Singh** (2002). Near-optimal reinforcement learning in polynomial time. *Machine Learning*.

24. **Kolter, J. Z.** and **A. Y. Ng**, Near-Bayesian exploration in polynomial time. *In Proceedings of the 26th Annual International Conference on Machine Learning*. 2009.

25. **Mahadevan, S.** and **M. Maggioni** (2007). Proto-value functions: A Laplacian framework for learning representation and control in Markov decision processes. *Journal of Machine Learning Research*.

26. **Miller, W. T.**, **F. H. Glanz**, and **L. G. Kraft** (1990). Cmac: An associative neural network alternative to backpropagation. *Proceedings of the IEEE*.

27. **Osband, I.**, **D. Russo**, and **B. Van Roy**, (more) efficient reinforcement learning via posterior sampling. *In Advances in Neural Information Processing Systems*. 2013.

28. **Popoviciu, T.** (1935). Sur les équations algébriques ayant toutes leurs racines réelles. *Mathematica*.

29. **Puterman, M. L.** (1994). Markov decision processes: Discrete stochastic dynamic programming.

30. **Ravindran, B.** and **A. Barto**, Approximate Homomorphisms: A Framework for Nonexact Minimization in Markov Decision Processes. *In Proceedings of the 5th International Conference on Knowledge Based Computer Systems*. 2004.

31. **Rummery, G. A.** and **M. Niranjan**, *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering, 1994.

32. **Russo, D.** and **B. V. Roy** (2014). An information-theoretic analysis of thompson sampling. *CoRR*, **abs/1403.5341**.

33. **Sorg, J.** and **S. Singh**, Transfer via soft homomorphisms. *In Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

34. **Sorg, J.**, **S. Singh**, and **R. L. Lewis**, Variance-based Rewards for Approximate Bayesian Reinforcement Learning. *In Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*. 2010.

35. **Strehl, A. L.** and **M. L. Littman** (2008). An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*.

36. **Strens, M.**, A Bayesian Framework for Reinforcement learning. *In Proceedings of the International Conference on Machine Learning*. 2000.

37. **Sutton, R. S.** (1988). Learning to predict by the methods of temporal differences. *Machine learning*.

38. **Sutton, R. S.** and **A. G. Barto**, *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1998.

39. **Tesauro, G.** (1995). TD-Gammon: A Self-Teaching Backgammon Program.

40. **Thompson, W. R.** (1933). On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika*.

41. **Watkins, C. J.** and **P. Dayan**, Q-learning. *In Machine learning*. Springer, 1992.

42. **Weber, R.** (1975). The theory of optimal stopping. Downing College, Cambridge.

LIST OF PAPERS BASED ON THESIS

1. Prasanna P., Sarath Chandar, A. P., and Ravindran, B. "iBayes: A Thompson Sampling Approach to Reinforcement Learning with Instructions" Presented at the NIPS workshop on Novel Trends and Applications in Reinforcement Learning (2014).

2. Prasanna P., Sarath Chandar, A. P., and Ravindran, B. "Thompson Sampling with Adaptive Exploration Bonus for Near-Optimal Policy Learning" Presented at The Multi-disciplinary Conference on Reinforcement Learning and Decision Making (RLDM) (2015). `http://goo.gl/sns6J9`

CURRICULUM VITAE

Name	Prasanna P
Date of Birth	14 March 1991
E-Mail	pp1403@gmail.com
Permanent Address	G1, Rohini Mridula Apartments, Amma mandapam Road, Srirangam, Trichy Dt., Tamilnadu - 620006.
Education	M.S. by Research (CSE), IIT Madras B.Tech (CSE), SASTRA University, Thanjavur (2012)

GENERAL TEST COMITTEE

Chairman	Dr. C. Chandra Sekhar Department of Computer Science and Engineering, Indian Institute of Technology, Madras.
Guide	Dr. Balaraman Ravindran Department of Computer Science and Engineering, Indian Institute of Technology, Madras.
Members	Dr. Krishna M. Sivalingam Department of Computer Science and Engineering, Indian Institute of Technology, Madras.
	Dr. Asokan Thondiyath Department of Engineering Design, Indian Institute of Technology, Madras.