Submitted to *Mathematics of Operations Research*
manuscript (Please, provide the mansucript number!)

Efficient Reinforcement Learning in Deterministic Systems with Value Function Generalization

Zheng Wen
Adobe Research, zwen@adobe.com,

Benjamin Van Roy
Stanford University, bvr@stanford.edu,

We consider the problem of reinforcement learning over episodes of a finite-horizon deterministic system and as a solution propose *optimistic constraint propagation (OCP)*, an algorithm designed to synthesize efficient exploration and value function generalization. We establish that when the true value function Q^* lies within the hypothesis class $\mathcal{Q}$, OCP selects optimal actions over all but at most $\dim_E[\mathcal{Q}]$ episodes, where $\dim_E$ denotes the *eluder dimension*. We establish further efficiency and asymptotic performance guarantees that apply even if Q^* does not lie in $\mathcal{Q}$, for the special case where $\mathcal{Q}$ is the span of pre-specified indicator functions over disjoint sets. We also discuss the computational complexity of OCP and present computational results involving two illustrative examples.

Key words: Reinforcement Learning, Efficient Exploration, Value Function Generalization, Approximate Dynamic Programming

1. Introduction

A growing body of work on efficient reinforcement learning provides algorithms with guarantees on sample and computational efficiency (see [13, 6, 2, 30, 4, 9] and references therein). This literature highlights the point that an effective exploration scheme is critical to the design of any efficient reinforcement learning algorithm. In particular, popular exploration schemes such as ϵ-greedy, Boltzmann, and knowledge gradient (see [27]) can require learning times that grow exponentially in the number of states and/or the planning horizon (see [38, 29]).

The aforementioned literature focusses on *tabula rasa* learning; that is, algorithms aim to learn with little or no prior knowledge about transition probabilities and rewards. Such algorithms require learning times that grow at least linearly with the number of states. Despite the valuable insights that have been generated through their design and analysis, these algorithms are of limited practical import because state spaces in most contexts of practical interest are enormous. There is a need for algorithms that generalize from past experience in order to learn how to make effective decisions in reasonable time.

There has been much work on reinforcement learning algorithms that generalize (see, e.g., [5, 31, 32, 24] and references therein). Most of these algorithms do not come with statistical or computational efficiency guarantees, though there are a few noteworthy exceptions, which we now discuss. A number of results treat policy-based algorithms (see [10, 3] and references therein), in which the goal is to select high-performers among a pre-specified collection of policies as learning progresses. Though interesting results have been produced in this line of work, each entails quite restrictive assumptions or does not make strong guarantees. Another body of work focuses on model-based algorithms. An algorithm proposed by Kearns and Koller [12] fits a factored model to observed data and makes decisions based on the fitted model. The authors establish a sample complexity bound that is polynomial in the number of model parameters rather than the number of states, but the algorithm is computationally intractable because of the difficulty of solving factored MDPs. Lattimore et al. [15] propose a novel algorithm for the case where the true environment is known to belong to a finite or compact class of models, and shows that its sample complexity is polynomial in the cardinality of the model class if the model class is finite, or the ϵ-covering-number if the model class is compact. Though this result is theoretically interesting, for most model classes of interest, the ϵ-covering-number is enormous since it typically grows exponentially in the number of free parameters. Ortner and Ryabko [20] establish a regret bound for an algorithm that applies to problems with continuous state spaces and Hölder-continuous rewards and transition kernels. Though the results represent an interesting contribution to the literature, a couple features of the regret bound weaken its practical implications. First, regret grows linearly with the Hölder constant of the transition kernel, which for most contexts of practical relevance grows exponentially in the number of state variables. Second, the dependence on time becomes arbitrarily close to linear as the dimension of the state space grows. Pazis and Parr [22] also consider problems with continuous state spaces. They assume that the Q-functions are Lipschitz-continuous or Hölder-continuous and establish a sample complexity bound. Though the results are interesting and significant, the sample complexity bound is log-linear in the covering number of the state-action space, which also typically grows exponentially in the number of free parameters for most practical problems. Reinforcement learning in linear systems with quadratic cost is treated in Abbasi-Yadkori and Szepesvári [1]. The method proposed is shown to realize regret that grows with the square root of time. The result is interesting and the property is desirable, but to the best of our knowledge, expressions derived for regret in the analysis exhibit an exponential dependence on the number of state variables, and further, we are not aware of a computationally efficient way of implementing the proposed method. This work was extended by Ibrahimi et al. [8] to address linear systems with sparse structure. Here, there are efficiency guarantees that scale gracefully with the number of state variables, but only under sparsity and other technical assumptions.

The most popular approach to generalization in the applied reinforcement learning literature involves fitting parameterized value functions. Such approaches relate closely to supervised learning in that they learn functions from state-action pairs to value, though a difference is that value is influenced by action and observed only through delayed feedback. One advantage over model learning approaches is that, given a fitted value function, decisions can be made without solving a potentially intractable control problem. We see this as a promising direction, though there currently is a lack of theoretical results that provide attractive bounds on learning time with value function generalization. A relevant paper along these lines is [16], which studies efficient reinforcement learning with value function generalization in the KWIK framework (see [17]) and reduces the problem to efficient KWIK online regression. However, the authors do not show how to solve the general KWIK online regression problem efficiently, and it is not even clear whether this is possible. Thus, though the result of Li and Littman [16] is interesting, it does not provide a provably efficient algorithm for general reinforcement learning problems. However, it is worth mentioning that Li et al. [17] has provided a solution to KWIK online regression with deterministic linear functions. As we will discuss later, this can be seen as a special case of the coherent learning problems we consider in Section 5.2.

An important challenge that remains is to couple exploration and value function generalization in a provably effective way, and in particular, to establish sample and computational efficiency guarantees that scale gracefully with the planning horizon and model complexity. In this paper, we aim to make progress in this direction. To start with a simple context, we restrict our attention to deterministic systems that evolve over finite time horizons, and we consider episodic learning, in which an agent repeatedly interacts with the same system. As a solution to the problem, we propose *optimistic constraint propagation (OCP)*, a computationally efficient reinforcement learning algorithm designed to synthesize efficient exploration and value function generalization. We establish that when the true value function Q^* lies within the hypothesis class $\mathcal{Q}$, OCP selects optimal actions over all but at most $\dim_E[\mathcal{Q}]$ episodes. Here, $\dim_E$ denotes the *eluder dimension*, which quantifies complexity of the hypothesis class. A corollary of this result is that regret is bounded by a function that is constant over time and linear in the problem horizon and eluder dimension.

To put our aforementioned result in perspective, it is useful to relate it to other lines of work. Consider first the broad area of reinforcement learning algorithms that fit value functions, such as SARSA [25]. Even with the most commonly used sort of hypothesis class $\mathcal{Q}$, which is made up of linear combinations of fixed basis functions, and even when the hypothesis class contains the true value function Q^*, there are no guarantees that these algorithms will efficiently learn to make near-optimal decisions. On the other hand, our result implies that OCP attains near-optimal performance in time that scales linearly with the number of basis functions. Now consider the more specialized context of a deterministic linear system with quadratic cost and a finite time horizon. The analysis of Abbasi-Yadkori and Szepesvári [1] can be leveraged to produce regret bounds that scale exponentially in the number of state variables. On the other hand, using a hypothesis space $\mathcal{Q}$ consisting of quadratic functions of state-action pairs, the results of this paper show that OCP behaves near optimally within time that scales quadratically in the number of state and action variables.

We also establish efficiency and asymptotic performance guarantees that apply to agnostic reinforcement learning, where Q^* does not necessarily lie in $\mathcal{Q}$. In particular, we consider the case where $\mathcal{Q}$ is the span of pre-specified indicator functions over disjoint sets. Our results here add to the literature on agnostic reinforcement learning with such a hypothesis class [28, 33, 7, 34]. Prior work in this area has produced interesting algorithms and insights, as well as bounds on performance loss associated with potential limits of convergence, but no convergence or efficiency guarantees. These results build on and add to those reported in an earlier paper that we published in proceedings of a conference [37].

In addition to establishing theoretical results, we present computational results involving two illustrative examples: a synthetic deterministic Markov chain discussed in Van Roy and Wen [35] and the inverted pendulum problem considered in Lagoudakis et al. [14]. We compare OCP against least-squares value iteration (LSVI), a classical reinforcement learning algorithm. In both experiments, the performance of OCP is orders of magnitude better than that of LSVI. It is worth mentioning that in the inverted pendulum example, we consider a case in which there are small magnitude stochastic disturbances additive to the control. This result shows that, though OCP is motivated and developed in the framework of reinforcement learning in deterministic systems, it might also work well in some reinforcement learning problems with stochastic environments, especially when the magnitude of the stochastic disturbances is small.

Finally, it is worth pointing out that reinforcement learning algorithms are often used to approximate solutions to large-scale dynamic programs, where *the system models are known*. In such contexts, there is no need for statistical learning as challenges are purely computational. Nevertheless, reinforcement learning algorithms make up popular solution techniques for such problems, and our algorithm and results also serve as contributions to the field of large-scale dynamic programs. Specifically, prior approximate dynamic programming algorithms that know the system model and fit a linear combination of basis functions to the value function, even when the optimal value function is within the span, come with no guarantees that a near-optimal policy can be computed efficiently. In this paper, we establish such a guarantee for OCP.

2. Episodic Reinforcement Learning in Deterministic Systems We consider a class of reinforcement learning problems in which an agent repeatedly interacts with an unknown discrete-time deterministic finite-horizon Markov decision process (MDP). Each interaction is referred to as an *episode*, and the agent's objective is to maximize the expected cumulative reward over episodes. The system is identified by a sextuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, F, R, S)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, H is the horizon, F is a system function, R is a reward function and S is a sequence of states. If action $a \in \mathcal{A}$ is selected while the system is in state $x \in \mathcal{S}$ at period $t = 0, 1, \cdots, H-1$, a reward of $R_t(x,a)$ is realized; furthermore, if $t < H-1$, the state transitions to $F_t(x,a)$. Each episode terminates at period $H-1$, and then a new episode begins. The initial state of episode j is the jth element of S.

To represent the history of actions and observations over multiple episodes, we will often index variables by both episode and period. For example, $x_{j,t}$ and $a_{j,t}$ denote the state and action at period t of episode j, where $j = 0, 1, \cdots$ and $t = 0, 1, \cdots, H-1$. To count the total number of steps since the agent started learning, we say period t in episode j is time $jH + t$.

A (deterministic) policy $\mu = (\mu_0, \ldots, \mu_{H-1})$ is a sequence of functions, each mapping $\mathcal{S}$ to $\mathcal{A}$. For each policy μ, define a value function $V_t^\mu(x) = \sum_{\tau=t}^{H-1} R_\tau(x_\tau, a_\tau)$, where $x_t = x$, $x_{\tau+1} = F_\tau(x_\tau, a_\tau)$, and $a_\tau = \mu_\tau(x_\tau)$. The optimal value function is defined by $V_t^*(x) = \sup_\mu V_t^\mu(x)$. A policy μ^* is said to be optimal if $V^{\mu^*} = V^*$. Throughout this paper, we will restrict attention to systems $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, F, R, S)$ that admit optimal policies. Note that this restriction incurs no loss of generality when the action space is finite.

It is also useful to define an action-contingent optimal value function: $Q_t^*(x,a) = R_t(x,a) + V_{t+1}^*(F_t(x,a))$ for $t < H-1$, and $Q_{H-1}^*(x,a) = R_{H-1}(x,a)$. Then, a policy μ^* is optimal if $\mu_t^*(x) \in \arg\max_{a\in\mathcal{A}} Q_t^*(x,a)$ for all (x,t).

This paper considers a reinforcement learning framework in which the agent initially knows the state space $\mathcal{S}$, the action space $\mathcal{A}$, the horizon H, and possibly some prior information about the value function, but does not know anything else about the system function F, the reward function R, or the sequence of the initial states S. A reinforcement learning algorithm generates each action $a_{j,t}$ based on observations made up to the tth period of the jth episode, including all states, actions, and rewards observed in previous episodes and earlier in the current episode, as well as $\mathcal{S}$, $\mathcal{A}$, H, and possible prior information. In each episode, the algorithm realizes reward $R^{(j)} = \sum_{t=0}^{H-1} R_t\left(x_{j,t}, a_{j,t}\right)$. Note that $R^{(j)} \leq V_0^*(x_{j,0})$ for each jth episode. To quantify the performance of a reinforcement learning algorithm, for any $\epsilon \geq 0$, we define the *ϵ-suboptimal sample complexity* of that algorithm as the number of episodes J_L for which $R^{(j)} < V_0^*(x_{j,0}) - \epsilon$. Moreover, we say a reinforcement learning algorithm is sample efficient in a given setting if for some reasonable choice of ϵ, the worst-case ϵ-suboptimal sample complexity of that algorithm is small for that setting. Note that if the reward function R is bounded, with $|R_t(x,a)| \leq \overline{R}$ for all (x,a,t), then a bound on ϵ-suboptimal sample complexity J_L also implies a bound on regret over episodes experienced prior to time T, defined by $\mathrm{Regret}(T) = \sum_{j=0}^{\lfloor T/H \rfloor - 1} (V_0^*(x_{j,0}) - R^{(j)})$. In particular, $\mathrm{Regret}(T) \leq 2\overline{R} H J_L + \epsilon \lfloor T/H \rfloor$.

3. Inefficient Exploration Schemes Before proceeding, it is worth pointing out that for the reinforcement learning problem proposed above, even in the *tabula rasa* case, a number of popular exploration schemes give rise to unsatisfactory sample complexities. Boltzmann[1] and ϵ-greedy exploration schemes (see, e.g., [23]), for example, lead to worst-case regret exponential in H and/or $|\mathcal{S}|$. Also, the knowledge gradient exploration scheme (see, e.g., [24] and [27]) can converge to suboptimal policies, and even when the ultimate policy is optimal, the time required can grow

[1] Notice that in this paper, we assume that the state transition model of the deterministic system is unknown. Some literature (see [18] and references therein) considers settings in which the state transition model is known (but the reward function is unknown), and shows that exploration schemes similar to Boltzmann exploration achieve regrets polynomial in H (or equivalently, mixing time) and $|\mathcal{S}|$.

exponentially in H and/or $|\mathcal{S}|$. Thus, even for the tabula rasa case, efficient exploration schemes are necessary for an algorithm to achieve a regret polynomial in H and $|\mathcal{S}|$.

In the remainder of this section, we provide an example that takes tabular Q-learning with Boltzmann exploration exponentially many episodes to learn an optimal policy. One can construct such examples for ϵ-greedy exploration similarly.

EXAMPLE 1. Consider the deterministic system described in Figure 1. Specifically, in Figure

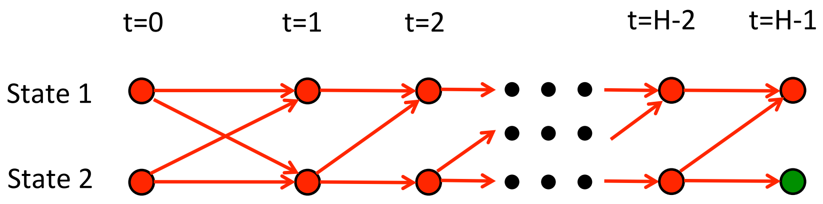


FIGURE 1. Deterministic system for which Boltzmann exploration is inefficient

1, each node represents a state-time pair, and each arrow corresponds to a possible deterministic state transition. We further assume that the rewards only depend on the state-time pair, with $R_t(x) = 0$ if the node is red and $R_t(x) = 1$ if the node is green. Obviously, for this example, the optimal policy is to follow the unique path to the green node.

Now assume that we apply the Q-learning algorithm with Boltzmann exploration to this example, with initial Q-values $Q_t(x,a) = 0$, $\forall (x,a,t)$. Thus, from the Q-learning algorithm, $Q_t(x,a) = 0$, $\forall (x,a,t)$, will hold until the first visit to the green node. We also note that if $Q_t(x,a) = 0$, $\forall (x,a,t)$, then with Boltzmann exploration, the Q-learning algorithm will choose actions uniformly randomly at every state-time pair. Thus, in this case, the probability that the algorithm will visit the green node in one episode is $1/2^{H-1}$. Consequently, in expectation, it takes the algorithm 2^{H-1} episodes to first visit the green node.

4. Optimistic Constraint Propagation

Our reinforcement learning algorithm – optimistic constraint propagation (OCP) – takes as input the state space $\mathcal{S}$, the action space $\mathcal{A}$, the horizon H, and a hypothesis class $\mathcal{Q}$ of candidates for Q^*. The algorithm maintains a sequence of subsets of $\mathcal{Q}$ and a sequence of scalar "upper bounds", which summarize constraints that past experience suggests for ruling out hypotheses. Each constraint in this sequence is specified by a state $x \in \mathcal{S}$, an action $a \in \mathcal{A}$, a period $t = 0, \ldots, H-1$, and an interval $[L, U] \subseteq \Re$, and takes the form $\{Q \in \mathcal{Q} : L \le Q_t(x,a) \le U\}$. The upper bound of the constraint is U. Given a sequence $\mathcal{C} = (\mathcal{C}_1, \ldots, \mathcal{C}_{|\mathcal{C}|})$ of such constraints and upper bounds $\mathcal{U} = (U_1, \ldots, U_{|\mathcal{C}|})$, for any $i, j = 1, \cdots, |\mathcal{C}|$ s.t. $i \neq j$, we say $\mathcal{C}_i < \mathcal{C}_j$, or constraint $\mathcal{C}_i$ has higher priority than $\mathcal{C}_j$, if (1) $U_i < U_j$ or (2) $U_i = U_j$ and $j > i$. That is, priority is assigned first based on upper bound, with smaller upper bound preferred, and then, in the event of ties in upper bound, based on position in the sequence, with more recent experience (larger index) preferred. A set $\mathcal{Q}_{\mathcal{C}}$ is defined constructively by Algorithm 1. Note that if the constraints do not conflict then $\mathcal{Q}_{\mathcal{C}} = \mathcal{C}_1 \cap \cdots \cap \mathcal{C}_{|\mathcal{C}|}$.

OCP, presented below as Algorithm 2, at each time t computes for the current state $x_{j,t}$ and each action a the greatest state-action value $Q_t(x_{j,t}, a)$ among functions in $\mathcal{Q}_{\mathcal{C}}$ and selects an action that attains the maximum. In other words, an action is chosen based on the most optimistic feasible outcome subject to constraints. The subsequent reward and state transition give rise to a new constraint that is used to update $\mathcal{C}$. Note that the update of $\mathcal{C}$ is postponed until one episode is completed.

As we will prove in Lemma 1, if $Q^* \in \mathcal{Q}$ then each constraint appended to $\mathcal{C}$ does not rule out Q^*, and therefore, the sequence of sets $\mathcal{Q}_{\mathcal{C}}$ generated as the algorithm progresses is decreasing and

Algorithm 1 Constraint Selection

```
Require: $\mathcal{Q}$, $\mathcal{C}$
  $\mathcal{Q}_{\mathcal{C}} \leftarrow \mathcal{Q}$
  Sort constraints in $\mathcal{C}$ s.t. $\mathcal{C}_{k_1} < \mathcal{C}_{k_2} < \cdots < \mathcal{C}_{k_{|\mathcal{C}|}}$
  for $\tau = 1$ to $|\mathcal{C}|$ do
      if $\mathcal{Q}_{\mathcal{C}} \cap \mathcal{C}_{k_\tau} \neq \varnothing$ then
          $\mathcal{Q}_{\mathcal{C}} \leftarrow \mathcal{Q}_{\mathcal{C}} \cap \mathcal{C}_{k_\tau}$
      end if
  end for
  return $\mathcal{Q}_{\mathcal{C}}$
```

Algorithm 2 Optimistic Contraint Propagation

```
Require: $\mathcal{S}$, $\mathcal{A}$, $H$, $\mathcal{Q}$
  Initialize $\mathcal{C} \leftarrow \varnothing$
  for episode $j = 0, 1, \cdots$ do
      Set $\mathcal{C}' \leftarrow \mathcal{C}$
      for period $t = 0, 1, \cdots, H-1$ do
          Apply $a_{j,t} \in \arg\max_{a \in \mathcal{A}} \sup_{Q \in \mathcal{Q}_{\mathcal{C}}} Q_t(x_{j,t}, a)$
          if $t < H-1$ then
              $U_{j,t} \leftarrow \sup_{Q \in \mathcal{Q}_{\mathcal{C}}} \left(R_t(x_{j,t}, a_{j,t}) + \sup_{a \in \mathcal{A}} Q_{t+1}(x_{j,t+1}, a)\right)$
              $L_{j,t} \leftarrow \inf_{Q \in \mathcal{Q}_{\mathcal{C}}} \left(R_t(x_{j,t}, a_{j,t}) + \sup_{a \in \mathcal{A}} Q_{t+1}(x_{j,t+1}, a)\right)$
          else
              $U_{j,t} \leftarrow R_t(x_{j,t}, a_{j,t})$, $L_{j,t} \leftarrow R_t(x_{j,t}, a_{j,t})$
          end if
          $\mathcal{C}' \leftarrow \mathcal{C}' \frown \{Q \in \mathcal{Q} : L_{j,t} \leq Q_t(x_{j,t}, a_{j,t}) \leq U_{j,t}\}$
      end for
      Update $\mathcal{C} \leftarrow \mathcal{C}'$
  end for
```

contains Q^* in its intersection. In the agnostic case, where Q^* may not lie in $\mathcal{Q}$, new constraints can be inconsistent with previous constraints, in which case selected previous constraints are relaxed as determined by Algorithm 1.

Let us briefly discuss several contexts of practical relevance and/or theoretical interest in which OCP can be applied.

- **Finite state/action tabula rasa case.** With finite state and action spaces, Q^* can be represented as a vector, and without special prior knowledge, it is natural to let $\mathcal{Q} = \Re^{|\mathcal{S}| \cdot |\mathcal{A}| \cdot H}$.
- **Polytopic prior constraints.** Consider the aforementioned example, but suppose that we have prior knowledge that Q^* lies in a particular polytope. Then we can let $\mathcal{Q}$ be that polytope and again apply OCP.
- **Linear systems with quadratic cost (LQ).** In this classical control model, if $\mathcal{S} = \Re^n$, $\mathcal{A} = \Re^m$, and R is a positive semidefinite quadratic, then for each t, Q^*_t is known to be a positive semidefinite quadratic, and it is natural to let $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0$ denoting the set of positive semidefinite quadratics.
- **Finite hypothesis class.** Consider a context when we have prior knowledge that Q^* can be well approximated by some element in a finite hypothesis class. Then we can let $\mathcal{Q}$ be that finite hypothesis class and apply OCP. This scenario is of particular interest from the perspective of learning theory. Note that this context entails agnostic learning, which is accommodated by OCP.

- **Linear combination of features.** It is often effective to hand-select a set of features $\phi_1, \ldots, \phi_K$, each mapping $\mathcal{S} \times \mathcal{A}$ to $\Re$, and, then for each t, aiming to compute weights $\theta^{(t)} \in \Re^K$ so that $\sum_k \theta_k^{(t)} \phi_k$ approximates Q_t^* without knowing for sure that Q_t^* lies in the span of the features. To apply OCP here, we would let $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \text{span}(\phi_1, \ldots, \phi_K)$. Note that this context also entails agnostic learning.
- **State aggregation.** This is a special case of the "linear combination of features" case discussed above. Specifically, for any $t = 0, 1, \cdots, H-1$, the state-action space at period t, $\mathcal{Z}_t = \{(x, a, t) : x \in \mathcal{S}, a \in \mathcal{A}\}$, is partitioned into K_t disjoint subsets $\mathcal{Z}_{t,1}, \mathcal{Z}_{t,2}, \cdots, \mathcal{Z}_{t,K_t}$, and we choose as features indicator functions for partition $\mathcal{Z}_{t,k}$'s.
- **Sigmoid.** If it is known that rewards are only received upon transitioning to the terminal state and take values between 0 and 1, it might be appropriate to use a variation of the aforementioned feature based model that applies a sigmoidal function to the linear combination. In particular, we could have $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \{\psi\left(\sum_k \theta_k \phi_k(\cdot)\right) : \theta \in \Re^K\}$, where $\psi(z) = e^z/(1+e^z)$.
- **Sparse linear combination of features.** Another case of potential interest is where Q^* can be encoded by a sparse linear combination of a large number of features $\phi_0, \cdots, \phi_K$. In particular, suppose that $\Phi = [\phi_0, \cdots, \phi_K] \in \Re^{|\mathcal{S}||\mathcal{A}| \times K}$, and $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \{\Phi\theta : \theta \in \Re^K, \|\theta\|_0 \leq K_0\}$, where $\|\theta\|_0$ is the L_0-"norm" of θ and $K_0 \ll K$.

It is worth mentioning that OCP, as we have defined it, assumes that an action a maximizing $\sup_{Q \in \mathcal{Q}_C} Q_t(x_{j,t}, a)$ exists in each iteration. Note that this assumption always holds if the action space $\mathcal{A}$ is finite, and it is not difficult to modify the algorithm so that it addresses cases where this is not true. But we have not presented the more general form of OCP in order to avoid complicating this paper.

Finally, we compare OCP with some classical reinforcement learning algorithms. It is worth mentioning that in the finite state/action tabula rasa case, OCP is equivalent to the Q-learning algorithm with *learning rate* 1 and initial Q-value $Q_t(x, a) = \infty$, $\forall (x, a, t)$. Please refer to the appendix for the justification of this argument. On the other hand, in the linear generalization/approximation case with $\mathcal{Q} = \text{span}(\phi_1, \ldots, \phi_K)^H$, OCP is very different from the classical approaches where the weights are estimated using either temporal-difference learning (e.g. Q-learning with linear approximation) or least squares (e.g. least-squares value iteration).

5. Sample Efficiency of Optimistic Constraint Propagation

We now establish results concerning the sample efficiency (performance) of OCP. Our results bound the ϵ-suboptimal sample complexities of OCP for appropriate choices of ϵ. Obviously, these sample complexity bounds must depend on the complexity of the hypothesis class. As such, we begin by defining the eluder dimension, as introduced in Russo and Van Roy [26], which is the notion of hypothesis class complexity we will use.

5.1. Eluder Dimension

Let $\mathcal{Z} = \{(x, a, t) : x \in \mathcal{S}, a \in \mathcal{A}, t = 0, \ldots, H-1\}$ be the set of all state-action-period triples, and let $\mathcal{Q}$ to denote a nonempty set of functions mapping $\mathcal{Z}$ to $\Re$. For all $(x, a, t) \in \mathcal{Z}$ and $\tilde{\mathcal{Z}} \subseteq \mathcal{Z}$, (x, a, t) is said to be *dependent* on $\tilde{\mathcal{Z}}$ with respect to $\mathcal{Q}$ if any pair of functions $Q, \tilde{Q} \in \mathcal{Q}$ that are equal on $\tilde{\mathcal{Z}}$ are equal at (x, a, t). Further, (x, a, t) is said to be *independent* of $\tilde{\mathcal{Z}}$ with respect to $\mathcal{Q}$ if (x, a, t) is not dependent on $\tilde{\mathcal{Z}}$ with respect to $\mathcal{Q}$.

The *eluder dimension* $\dim_{\text{E}}[\mathcal{Q}]$ of $\mathcal{Q}$ is the length of the longest sequence of elements in $\mathcal{Z}$ such that every element is independent of its predecessors. Note that $\dim_{\text{E}}[\mathcal{Q}]$ can be zero or infinity, and it is straightforward to show that if $\mathcal{Q}_1 \subseteq \mathcal{Q}_2$ then $\dim_{\text{E}}[\mathcal{Q}_1] \leq \dim_{\text{E}}[\mathcal{Q}_2]$. Based on results of Russo and Van Roy [26], we can characterize the eluder dimensions of various hypothesis classes presented in the previous section.

- **Finite state/action tabula rasa case.** If $\mathcal{Q} = \Re^{|\mathcal{S}| \cdot |\mathcal{A}| \cdot H}$, then $\dim_{\text{E}}[\mathcal{Q}] = |\mathcal{S}| \cdot |\mathcal{A}| \cdot H$.
- **Polytopic prior constraints.** If $\mathcal{Q}$ is a polytope of dimension d in $\Re^{|\mathcal{S}| \cdot |\mathcal{A}| \cdot H}$, then $\dim_{\text{E}}[\mathcal{Q}] = d$.

- **Linear systems with quadratic cost (LQ).** If $\mathcal{Q}_0$ is the set of positive semidefinite quadratics with domain $\Re^{m+n}$ and $\mathcal{Q} = \mathcal{Q}_0^H$, then $\dim_E[\mathcal{Q}] = (m+n+1)(m+n)H/2$.
- **Finite hypothesis space.** If $|\mathcal{Q}| < \infty$, then $\dim_E[\mathcal{Q}] \le |\mathcal{Q}| - 1$.
- **Linear combination of features.** If $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \mathrm{span}(\phi_1, \ldots, \phi_K)$, then $\dim_E[\mathcal{Q}] \le KH$.
- **State aggregation.** This is a special case of a linear combination of features. If $\mathcal{Q} = \mathcal{Q}_0^H$, and $\mathcal{Q}_0$ is the span of indicator functions for K partitions of the state-action space, then $\dim_E[\mathcal{Q}] \le KH$.
- **Sigmoid.** If $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \{\psi\left(\sum_k \theta_k \phi_k(\cdot)\right) : \theta \in \Re^K\}$, then $\dim_E[\mathcal{Q}] \le KH$.
- **Sparse linear combination of features.** If $\mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \{\Phi\theta : \theta \in \Re^K, \|\theta\|_0 \le K_0\}$ and $2K_0 \le \min\{|\mathcal{S}||\mathcal{A}|, K\}$, and any $2K_0 \times 2K_0$ submatrix of Φ has full rank, then $\dim_E[\mathcal{Q}] \le 2K_0 H$. We will establish this eluder dimension bound in the appendix.

5.2. Learning with a Coherent Hypothesis Class We now present results that apply when OCP is presented with a coherent hypothesis class; that is, where $Q^* \in \mathcal{Q}$. We refer to such cases as coherent learning cases. Our first result establishes that OCP can deliver less than optimal performance in no more than $\dim_E[\mathcal{Q}]$ episodes.

Theorem 1. *For any system $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, F, R, S)$, if OCP is applied with $Q^* \in \mathcal{Q}$, then* $|\{j : R^{(j)} < V_0^*(x_{j,0})\}| \le \dim_E[\mathcal{Q}]$.

That is, Theorem 1 bounds the 0-suboptimal sample complexity of OCP in coherent learning cases. This theorem follows from an "exploration-exploitation lemma" (Lemma 3), which asserts that in each episode, OCP either delivers optimal reward (exploits) or introduces a constraint that reduces the eluder dimension of the hypothesis class by one (explores). Consequently, OCP will experience sub-optimal performance in at most $\dim_E[\mathcal{Q}]$ episodes. We outline the proof of Theorem 1 at the end of this subsection and the detailed analysis is provided in the appendix. An immediate corollary bounds regret.

Corollary 1. *For any $\overline{R}$, any system $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, F, R, S)$ with $\sup_{(x,a,t)} |R_t(x,a)| \le \overline{R}$, and any T, if OCP is applied with $Q^* \in \mathcal{Q}$, then* $\mathrm{Regret}(T) \le 2\overline{R}H\dim_E[\mathcal{Q}]$.

Note the regret bound in Corollary 1 does not depend on time T, thus, it is an $O(1)$ bound. Furthermore, this regret bound is linear in $\overline{R}$, H and $\dim_E[\mathcal{Q}]$. Thus, if $\dim_E[\mathcal{Q}]$ does not depend on $|\mathcal{S}|$ or $|\mathcal{A}|$, then this regret bound also does not depend on $|\mathcal{S}|$ or $|\mathcal{A}|$. The following result demonstrates that the bounds of the above theorem and corollary are sharp.

Theorem 2. *For any $\overline{R} \ge 0$, any $K, H' = 1, 2, \cdots$ and any reinforcement learning algorithm $\tilde{\mu}$ that takes as input a state space, an action space, a horizon and a coherent hypothesis class, there exist a system $\mathcal{M} = (\mathcal{S}, \mathcal{A}, H, F, R, S)$ and a hypothesis class $\mathcal{Q}$ satisfying (1) $\sup_{(x,a,t)} |R_t(x,a)| \le \overline{R}$, (2) $H = H'$, (3) $\dim_E[\mathcal{Q}] = K$ and (4) $Q^* \in \mathcal{Q}$ such that if we apply $\tilde{\mu}$ to $\mathcal{M}$ with input $(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$, then $|\{j : R^{(j)} < V_0^*(x_{j,0})\}| \ge \dim_E[\mathcal{Q}]$ and $\sup_T \mathrm{Regret}(T) \ge 2\overline{R}H\dim_E[\mathcal{Q}]$.*

A constructive proof of these lower bounds is provided at the end of this subsection. Following our discussion in previous sections, we discuss several interesting contexts in which the agent knows a coherent hypothesis class $\mathcal{Q}$ with finite eluder dimension.

- **Finite state/action tabula rasa case.** If we apply OCP in this case, then it will deliver sub-optimal performance in at most $|\mathcal{S}| \cdot |\mathcal{A}| \cdot H$ episodes. Furthermore, if $\sup_{(x,a,t)} |R_t(x,a)| \le \overline{R}$, then for any T, $\mathrm{Regret}(T) \le 2\overline{R}|\mathcal{S}||\mathcal{A}|H^2$.
- **Polytopic prior constraints.** If we apply OCP in this case, then it will deliver sub-optimal performance in at most d episodes. Furthermore, if $\sup_{(x,a,t)} |R_t(x,a)| \le \overline{R}$, then for any T, $\mathrm{Regret}(T) \le 2\overline{R}Hd$.

- **Linear systems with quadratic cost (LQ).** If we apply OCP in this case, then it will deliver sub-optimal performance in at most $(m+n+1)(m+n)H/2$ episodes.
- **Finite hypothesis class case.** Assume that the agent has prior knowledge that $Q^* \in \mathcal{Q}$, where $\mathcal{Q}$ is a finite hypothesis class. If we apply OCP in this case, then it will deliver sub-optimal performance in at most $|\mathcal{Q}|-1$ episodes. Furthermore, if $\sup_{(x,a,t)} |R_t(x,a)| \leq \overline{R}$, then for any T, $\mathrm{Regret}(T) \leq 2\overline{R}H\left[|\mathcal{Q}|-1\right]$.
- **Linear combination of features.** Assume that $Q^* \in \mathcal{Q} = \mathcal{Q}_0^H$ with $\mathcal{Q}_0 = \mathrm{span}(\phi_1, \ldots, \phi_K)$. If we apply OCP in this case, then it will deliver sub-optimal performance in at most KH episodes. Furthermore, if $\sup_{(x,a,t)} |R_t(x,a)| \leq \overline{R}$, then for any T, $\mathrm{Regret}(T) \leq 2\overline{R}KH^2$. Notice that this result can also be derived based on the KWIK online regression with deterministic linear functions (see [17]).
- **Sparse linear combination case.** Assume that the agent has prior knowledge that $Q^* \in \mathcal{Q}$, where $\mathcal{Q} = \left\{\Phi\theta : \theta \in \Re^K, \|\theta\|_0 \leq K_0\right\}^H$ and $2K_0 \leq \min\{|\mathcal{S}||\mathcal{A}|, K\}$, and any $2K_0 \times 2K_0$ submatrix of Φ has full rank. If we apply OCP in this case, then it will deliver sub-optimal performance in at most $2K_0H$ episodes. Furthermore, if $\sup_{(x,a,t)} |R_t(x,a)| \leq \overline{R}$, then for any T, $\mathrm{Regret}(T) \leq 4\overline{R}K_0H^2$.

Before proceeding, it is worth pointing out that one key feature of OCP, which distinguishes it from other reinforcement learning algorithms and makes it sample efficient when presented with a coherent hypothesis class, is that it updates the feasible set of candidates for Q^* in a conservative manner that never rues out Q^* and always uses optimistic estimates from this feasible set to guide action.

5.2.1. Sketch of Proof for Theorem 1 We start by defining some useful notations. Specifically, we use $\mathcal{C}_j$ to denote the $\mathcal{C}$ in episode j to distinguish $\mathcal{C}$'s in different episodes, and use z as a shorthand notation for a state-action-time triple (x,a,t). We first prove that if $Q^* \in \mathcal{Q}$, then each constraint appended to $\mathcal{C}$ does not rule out Q^*, and thus we have $Q^* \in \mathcal{Q}_{\mathcal{C}_j}$ for any $j = 0, 1, \cdots$.

Lemma 1. *If $Q^* \in \mathcal{Q}$, then (a) $Q^* \in \mathcal{Q}_{\mathcal{C}_j}$ for all $j = 0, 1, \cdots$, and (b) $L_{j,t} \leq Q_t^*(x_{j,t}, a_{j,t}) \leq U_{j,t}$ for all t and all $j = 0, 1, \cdots$.*

Please refer to the appendix for the proof of Lemma 1. Notice that Lemma 1(b) implies that no constraints are conflicting if $Q^* \in \mathcal{Q}$ since Q^* satisfies all the constraints. For any episode $j = 0, 1, \cdots$, we define $\mathcal{Z}_j$ and t_j^* by Algorithm 3.

Algorithm 3 Definition of $\mathcal{Z}_j$ and t_j^*

```
Initialize $\mathcal{Z}_0 \leftarrow \varnothing$
for $j = 0, 1, \cdots$ do
    Set $t_j^* \leftarrow$ NULL
    if $\exists t = 0, 1, \cdots, H-1$ s.t. $(x_{j,t}, a_{j,t}, t)$ is independent of $\mathcal{Z}_j$ with respect to $\mathcal{Q}$ then
        Set
            $t_j^* \leftarrow$ last period $t$ in episode $j$ s.t. $(x_{j,t}, a_{j,t}, t)$ is independent of $\mathcal{Z}_j$ with respect to $\mathcal{Q}$
        and $\mathcal{Z}_{j+1} \leftarrow \left[\mathcal{Z}_j, (x_{j,t_j^*}, a_{j,t_j^*}, t_j^*)\right]$
    else
        Set $\mathcal{Z}_{j+1} \leftarrow \mathcal{Z}_j$
    end if
end for
```

Note that by definition, in each episode j, $\mathcal{Z}_j$ is a sequence (ordered set) of elements in $\mathcal{Z}$. Furthermore, each element in $\mathcal{Z}_j$ is independent of its predecessors. Moreover, if $t_j^* \neq \text{NULL}$, then it is the last period in episode j s.t. $(x_{j,t}, a_{j,t}, t)$ is independent of $\mathcal{Z}_j$ with respect to $\mathcal{Q}$. As we will show in the analysis, if $t_j^* \neq \text{NULL}$, another interpretation of t_j^* is that it is the first period (in backward order) in episode j when the value of a new state-action-period triple is learned perfectly. Based on the notions of $\mathcal{Z}_j$ and t_j^*, we have the following technical lemma:

LEMMA 2. *$\forall j = 0, 1, \cdots$ and $\forall t = 0, 1, \cdots, H-1$, we have*
(a) *$\forall z \in \mathcal{Z}_j$ and $\forall Q \in \mathcal{Q}_{\mathcal{C}_j}$, we have $Q(z) = Q^*(z)$.*
(b) *If $(x_{j,t}, a_{j,t}, t)$ is dependent on $\mathcal{Z}_j$ with respect to $\mathcal{Q}$, then (1) $a_{j,t}$ is optimal and (2) $Q_t(x_{j,t}, a_{j,t}) = Q_t^*(x_{j,t}, a_{j,t}) = V_t^*(x_{j,t})$, $\forall Q \in \mathcal{Q}_{\mathcal{C}_j}$.*

Please refer to the appendix for the proof of Lemma 2. Based on Lemma 2, we have the following exploration/exploitation lemma, which states that in each episode j, OCP algorithm either achieves the optimal reward (exploits), or updates $\mathcal{Q}_{\mathcal{C}_{j+1}}$ based on the Q-value at an independent state-action-time triple (explores).

LEMMA 3. *For any $j = 0, 1, \cdots$, if $t_j^* \neq \text{NULL}$, then $(x_{j,t_j^*}, a_{j,t_j^*}, t_j^*)$ is independent of $\mathcal{Z}_j$, $|\mathcal{Z}_{j+1}| = |\mathcal{Z}_j| + 1$ and $Q_{t_j^*}(x_{j,t_j^*}, a_{j,t_j^*}) = Q_{t_j^*}^*(x_{j,t_j^*}, a_{j,t_j^*})$ $\forall Q \in \mathcal{Q}_{\mathcal{C}_{j+1}}$* (Exploration). Otherwise, if $t_j^* = \text{NULL}$, then $R^{(j)} = V_0^*(x_{j,0})$ (Exploitation).

Theorem 1 follows from Lemma 3. Please refer to the appendix for the detailed proofs for Lemma 3 and Theorem 1.

5.2.2. Constructive Proof for Theorem 2 We start by defining some useful terminologies and notations. First, for any state space $\mathcal{S}$, any time horizon $H = 1, 2, \cdots$, any action space $\mathcal{A}$, and any hypothesis class $\mathcal{Q}$, we use $\mathbb{M}(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$ to denote the set of all finite-horizon deterministic system $\mathcal{M}$'s with state space $\mathcal{S}$, action space $\mathcal{A}$, horizon H and $Q^* \in \mathcal{Q}$. Notice that for any reinforcement learning algorithm that takes $\mathcal{S}$, $\mathcal{A}$, H, $\mathcal{Q}$ as input, and knows that $\mathcal{Q}$ is a coherent hypothesis class, $\mathbb{M}(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$ is the set of all finite-horizon deterministic systems that are consistent with the algorithm's prior information.

We provide a constructive proof for Theorem 2 by considering a scenario in which an adversary adaptively chooses a deterministic system $\mathcal{M} \in \mathbb{M}(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$. Specifically, we assume that

- At the beginning of each episode j, the adversary adaptively chooses the initial state $x_{j,0}$.
- At period t in episode j, the agent first chooses an action $a_{j,t} \in \mathcal{A}$ based on some RL algorithm[2], and then the adversary adaptively chooses a set of state-action-time triples $\mathcal{Z}_{j,t} \subseteq \mathcal{Z}$ and specifies the rewards and state transitions on $\mathcal{Z}_{j,t}$, subject to the constraints that (1) $(x_{j,t}, a_{j,t}, t) \in \mathcal{Z}_{j,t}$ and (2) these adaptively specified rewards and state transitions must be consistent with the agent's prior knowledge and past observations.

We assume that the adversary's objective is to maximize the number of episodes in which the agent achieves sub-optimal rewards. Then we have the following lemma:

LEMMA 4. *$\forall H, K = 1, 2, \cdots$ and $\forall \overline{R} \geq 0$, there exist a state space $\mathcal{S}$, an action space $\mathcal{A}$ and a hypothesis class $\mathcal{Q}$ with $\dim_E[\mathcal{Q}] = K$ such that no matter how the agent adaptively chooses actions, the adversary can adaptively choose an $\mathcal{M} \in \mathbb{M}(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$ with $\sup_{(x,a,t)} |R_t(x,a)| \leq \overline{R}$ such that the agent will achieve sub-optimal rewards in at least K episodes, and $\sup_T \text{Regret}(T) \geq 2\overline{R}HK$.*

Since the fact that an adversary can adaptively choose a "bad" deterministic system simply implies that such a system exists, thus, Theorem 2 follows directly from Lemma 4.

[2] In general, the RL algorithm can choose actions randomly. If so, all the results in Section 5.2.2 hold on the realized sample path.

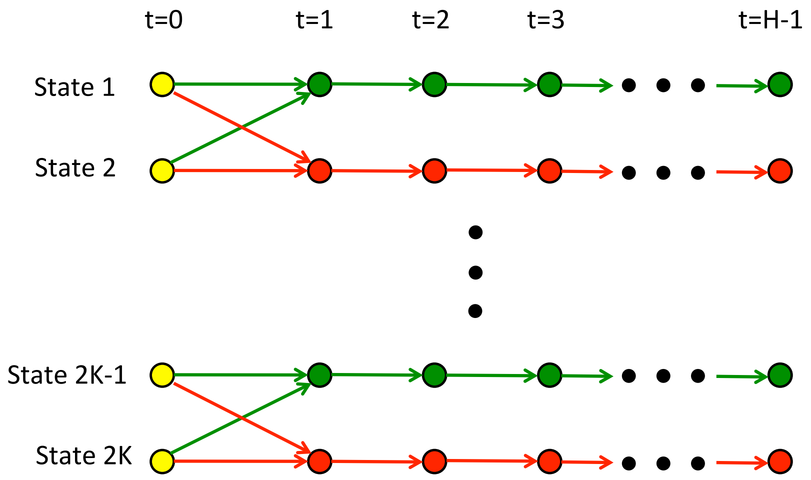


FIGURE 2. Illustration of the state transition

Proof for Lemma 4 We provide a constructive proof for Lemma 4. Specifically, $\forall H, K = 1, 2, \cdots$ and $\forall \overline{R} \geq 0$, we construct the state space as $\mathcal{S} = \{1, 2, \cdots, 2K\}$, and the action space as $\mathcal{A} = \{1, 2\}$. Recall that $\mathcal{Z} = \{(x, a, t) : x \in \mathcal{S}, t = 0, 1, \cdots, H-1, \text{ and } a \in \mathcal{A}\}$, thus, for $\mathcal{S}$ and $\mathcal{A}$ constructed above, we have $|\mathcal{Z}| = 4KH$. Hence, Q^*, the optimal Q-function, can be represented as a vector in $\Re^{4KH}$.

Before specifying the hypothesis class $\mathcal{Q}$, we first define a matrix $\Phi \in \Re^{4KH \times K}$ as follows. $\forall (x, a, t) \in \mathcal{Z}$, let $\Phi(x, a, t) \in \Re^K$ denote the row of Φ corresponding to the state-action-time triple (x, a, t), we construct $\Phi(x, a, t)$ as:

$$\Phi(x,a,t) = \begin{cases} (H-t)\mathbf{e}_k & \text{if } x = 2k-1 \text{ for some } k = 1, \cdots, K,\ a = 1, 2 \text{ and } t = 1, \cdots, H-1 \\ -(H-t)\mathbf{e}_k & \text{if } x = 2k \text{ for some } k = 1, \cdots, K,\ a = 1, 2 \text{ and } t = 1, \cdots, H-1 \\ H\mathbf{e}_k & \text{if } x = 2k-1 \text{ or } 2k \text{ for some } k = 1, \cdots, K,\ a = 1 \text{ and } t = 0 \\ -H\mathbf{e}_k & \text{if } x = 2k-1 \text{ or } 2k \text{ for some } k = 1, \cdots, K,\ a = 2 \text{ and } t = 0 \end{cases} \tag{1}$$

where $\mathbf{e}_k \in \Re^K$ is a (row) indicator vector with a one at index k and zeros everywhere else. Obviously, $\text{rank}(\Phi) = K$. We choose $\mathcal{Q} = \text{span}[\Phi]$, thus $\dim_{\mathrm{E}}[\mathcal{Q}] = \dim(\text{span}[\Phi]) = \text{rank}(\Phi) = K$.

Now we describe how the adversary adaptively chooses a finite-horizon deterministic system $\mathcal{M} \in \mathbb{M}(\mathcal{S}, \mathcal{A}, H, \mathcal{Q})$:

- For any $j = 0, 1, \cdots$, at the beginning of episode j, the adversary chooses the initial state in that episode as $x_{j,0} = (j \bmod K) \times 2 + 1$. That is, $x_{0,0} = x_{K,0} = x_{2K,0} = \cdots = 1$, $x_{1,0} = x_{K+1,0} = x_{2K+1,0} = \cdots = 3$, etc.
- Before interacting with the agent, the adversary chooses the following system function F[3]:

$$F_t(x,a) = \begin{cases} 2k-1 & \text{if } t = 0,\ x = 2k-1 \text{ or } 2k \text{ for some } k = 1, \cdots, K, \text{ and } a = 1 \\ 2k & \text{if } t = 0,\ x = 2k-1 \text{ or } 2k \text{ for some } k = 1, \cdots, K, \text{ and } a = 2 \\ x & \text{if } t = 1, \cdots, H-2 \text{ and } a = 1, 2 \end{cases}.$$

  The state transition is illustrated in Figure 2.
- In episode $j = 0, 1, \cdots, K-1$, the adversary adaptively chooses the reward function R as follows. If the agent takes action 1 in period 0 in episode j at initial state $x_{j,0} = 2j+1$, then the adversary set $R_0(2j+1, 1) = R_0(2j+2, 1) = R_t(2j+1, 1) = R_t(2j+1, 2) = -\overline{R}$ and $R_0(2j+1, 2) = R_0(2j+$

[3] More precisely, in this constructive proof, the adversary does not need to adaptively choose the system function F. He can choose F beforehand.

$2,2) = R_t(2j+2,1) = R_t(2j+2,2) = \overline{R}$, $\forall t = 1,2,\cdots,H-1$. Otherwise (i.e. if the agent takes action 2 in period 0 in episode j), then the adversary set $R_0(2j+1,1) = R_0(2j+2,1) = R_t(2j+1,1) = R_t(2j+1,2) = \overline{R}$ and $R_0(2j+1,2) = R_0(2j+2,2) = R_t(2j+2,1) = R_t(2j+2,2) = -\overline{R}$. Notice that the adversary completes the construction of the deterministic system $\mathcal{M}$ at the end of episode $K-1$.

Note that for the constructed deterministic system $\mathcal{M}$, we have $Q^* \in \mathcal{Q}$. Specifically, it is straight forward to see that $Q^* = \Phi\theta^*$, where $\theta^* \in \Re^K$, and θ^*_k, the kth element of θ, is defined as $\theta^*_k = -\overline{R}$ if $a_{k-1,0} = 1$ and $\theta^*_k = \overline{R}$ if $a_{k-1,0} = 2$, for any $k = 1,2,\cdots,K$. Thus, the constructed deterministic system $\mathcal{M} \in \mathbb{M}(\mathcal{S},\mathcal{A},H,\mathcal{Q})$.

Finally, we show that the constructed deterministic system $\mathcal{M}$ satisfies Lemma 4. Obviously, we have $|R_t(x,a)| \leq \overline{R}$, $\forall (x,a,t) \in \mathcal{Z}$. Furthermore, note that the agent achieves sub-optimal rewards in the first K episodes, thus, he will achieve sub-optimal rewards in at least K episodes. In addition, the cumulative regret in the first K episodes is $2KH\overline{R}$, thus, $\sup_T \text{Regret}(T) \geq 2KH\overline{R}$.

5.3. Agnostic Learning in State Aggregation Case As we have discussed in Section 4, OCP can also be applied in agnostic learning cases, where Q^* may not lie in $\mathcal{Q}$. For such cases, the performance of OCP should depend on not only the complexity of $\mathcal{Q}$, but also the distance between $\mathcal{Q}$ and Q^*. In this subsection, we present results when OCP is applied in a special agnostic learning case, where $\mathcal{Q}$ is the span of pre-specified indicator functions over disjoint subsets. We henceforth refer to this case as the state aggregation case.

Specifically, we assume that for any $t = 0,1,\cdots,H-1$, the state-action space at period t, $\mathcal{Z}_t = \{(x,a,t) : x \in \mathcal{S}, a \in \mathcal{A}\}$, can be partitioned into K_t disjoint subsets $\mathcal{Z}_{t,1}, \mathcal{Z}_{t,2}, \cdots, \mathcal{Z}_{t,K_t}$, and use $\phi_{t,k}$ to denote the indicator function for partition $\mathcal{Z}_{t,k}$ (i.e. $\phi_{t,k}(x,a,t) = 1$ if $(x,a,t) \in \mathcal{Z}_{t,k}$, and $\phi_{t,k}(x,a,t) = 0$ otherwise). We define $K = \sum_{t=0}^{H-1} K_t$, and $\mathcal{Q}$ as

$$\mathcal{Q} = \text{span}\left\{\phi_{0,1}, \phi_{0,2}, \cdots, \phi_{0,K_0}, \phi_{1,1}, \cdots, \phi_{H-1,K_{H-1}}\right\}. \tag{2}$$

Note that $\dim_E[\mathcal{Q}] = K$. We define the distance between Q^* and the hypothesis class $\mathcal{Q}$ as

$$\rho = \min_{Q \in \mathcal{Q}} \|Q - Q^*\|_\infty = \min_{Q \in \mathcal{Q}} \sup_{(x,a,t)} |Q_t(x,a) - Q^*_t(x,a)|. \tag{3}$$

The following result establishes that with $\mathcal{Q}$ and ρ defined above, the performance loss of OCP is larger than $2\rho H(H+1)$ in at most K episodes.

THEOREM 3. *For any system $\mathcal{M} = (\mathcal{S},\mathcal{A},H,F,R,S)$, if OCP is applied with $\mathcal{Q}$ defined in Eqn(2), then $|\{j : R^{(j)} < V^*_0(x_{j,0}) - 2\rho H(H+1)\}| \leq K$, where K is the number of partitions and ρ is defined in Eqn(3).*

That is, Theorem 3 bounds the $2\rho H(H+1)$-suboptimal sample complexity of OCP in the state aggregation case. Similar to Theorem 1, this theorem also follows from an "exploration-exploitation lemma" (Lemma 7), which asserts that in each episode, OCP either delivers near-optimal reward (exploits), or approximately determines $Q^*_t(x,a)$'s for all the (x,a,t)'s in a disjoint subset (explores). We outline the proof for Theorem 3 at the end of this subsection, and the detailed analysis is provided in the appendix. An immediate corollary bounds regret.

COROLLARY 2. *For any $\overline{R} \geq 0$, any system $\mathcal{M} = (\mathcal{S},\mathcal{A},H,F,R,S)$ with $\sup_{(x,a,t)} |R_t(x,a)| \leq \overline{R}$, and any time T, if OCP is applied with $\mathcal{Q}$ defined in Eqn(2), then* $\text{Regret}(T) \leq 2\overline{R}KH + 2\rho(H+1)T$, *where K is the number of partitions and ρ is defined in Eqn(3).*

Note that the regret bound in Corollary 2 is $O(T)$, and the coefficient of the linear term is $2\rho(H+1)$. Consequently, if Q^* is close to $\mathcal{Q}$, then the regret will increase slowly with T. Furthermore, the regret bound in Corollary 2 does not directly depend on $|\mathcal{S}|$ or $|\mathcal{A}|$.

We further notice that the threshold performance loss in Theorem 3 is $O(\rho H^2)$. The following proposition provides a condition under which the performance loss in one episode is $O(\rho H)$.

PROPOSITION 1. *For any episode j, if $\mathcal{Q}_\mathcal{C} \subseteq \{Q \in \mathcal{Q} : L_{j,t} \leq Q_t(x_{j,t}, a_{j,t}) \leq U_{j,t}\}$, $\forall t = 0, \cdots, H-1$, then we have $V_0^*(x_{j,0}) - R^{(j)} \leq 6\rho H = O(\rho H)$.*

That is, if all the new constraints in an episode are redundant, then the performance loss in that episode is $O(\rho H)$. Note that if the condition for Proposition 1 holds in an episode, then $\mathcal{Q}_\mathcal{C}$ will not be modified at the end of that episode. Furthermore, if the system has a fixed initial state and the condition for Proposition 1 holds in one episode, then it will hold in all the subsequent episodes, and consequently, the performance losses in all the subsequent episodes are $O(\rho H)$.

It is worth mentioning that the sample complexity bound and the regret bounds in this subsection are derived under the assumption that the partitions of the state-action spaces are given. An important problem in practice is how to choose the optimal number K of the state-action partitions. There are many approaches to choose K, and one approach is to formulate it as a regret bound optimization problem. Specifically, assume that for any $K \geq H$, $\mathcal{Q}(K)$ is the hypothesis class the agent constructs with K partitions. Let $\rho(K)$ be a known upper bound on the distance $\min_{Q \in \mathcal{Q}(K)} \|Q - Q^*\|_\infty$. Then from Corollary 2 , Regret$(T) \leq 2\overline{R}KH + 2\rho(K)(H+1)T$. Hence, the problem of choosing an optimal K can be formulated as

$$\min_{K \geq H} 2\bar{R}KH + 2\rho(K)(H+1)T,$$

which can be efficiently solved by line search. Notice that whether or not the optimal K depends on $|\mathcal{S}||\mathcal{A}|$, and/or how it grows with $|\mathcal{S}||\mathcal{A}|$, depends on if and how $\rho(K)$ depends on $|\mathcal{S}||\mathcal{A}|$. That is, it depends on the agent's capability to construct a good hypothesis class $\mathcal{Q}(K)$ for a given K, which in turn might depend on the agent's prior knowledge about the problem.

5.3.1. Sketch of Proof for Theorem 3 and Proposition 1 We start by briefly describing how constraint selection algorithm updates $\mathcal{Q}_\mathcal{C}$'s for the function class $\mathcal{Q}$ specified in Eqn(2). Specifically, let $\theta_{t,k}$ denote the coefficient of the indicator function $\phi_{t,k}$, $\forall (t,k)$. Assume that (x,a,t) belongs to partition $\mathcal{Z}_{t,k}$, then, with $\mathcal{Q}$ specified in Eqn(4.1), $L \leq Q_t(x,a) \leq U$ is a constraint on and only on $\theta_{t,k}$, and is equivalent to $L \leq \theta_{t,k} \leq U$. By induction, it is straightforward to see in episode j, $\mathcal{Q}_{\mathcal{C}_j}$ can be represented as

$$\mathcal{Q}_{\mathcal{C}_j} = \left\{ \theta \in \Re^K : \underline{\theta}_{t,k}^{(j)} \leq \theta_{t,k} \leq \overline{\theta}_{t,k}^{(j)}, \, \forall (t,k) \right\}, \tag{4}$$

for some $\underline{\theta}_{t,k}^{(j)}$'s and $\overline{\theta}_{t,k}^{(j)}$'s. Note that $\underline{\theta}_{t,k}^{(j)}$ can be $-\infty$ and $\overline{\theta}_{t,k}^{(j)}$ can be ∞, and when $j=0$, $\overline{\theta}_{t,k}^{(0)} = \infty$ and $\underline{\theta}_{t,k}^{(0)} = -\infty$. Furthermore, from the constraint selection algorithm, $\overline{\theta}_{t,k}^{(j)}$ is monotonically non-increasing in j, $\forall (t,k)$. Specifically, if OCP adds a new constraint $L \leq \theta_{t,k} \leq U$ on $\theta_{t,k}$ in episode j, we have $\overline{\theta}_{t,k}^{(j+1)} = \min\{\overline{\theta}_{t,k}^{(j)}, U\}$; otherwise, $\overline{\theta}_{t,k}^{(j+1)} = \overline{\theta}_{t,k}^{(j)}$. Thus, if $\overline{\theta}_{t,k}^{(j)} < \infty$, then $\overline{\theta}_{t,k}^{(j')} < \infty$, $\forall j' \geq j$.

For any episode j, we define $Q_j^\uparrow$, the optimistic Q-function in episode j, as

$$Q_{j,t}^\uparrow(x,a) = \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x,a), \quad \forall (x,a,t) \in \mathcal{Z}. \tag{5}$$

Similarly, $Q_j^\downarrow$, the pessimistic Q-function in episode j, is defined as

$$Q_{j,t}^\downarrow(x,a) = \inf_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x,a), \quad \forall (x,a,t) \in \mathcal{Z}. \tag{6}$$

Clearly, if $(x,a,t) \in \mathcal{Z}_{t,k}$, then we have $Q_{j,t}^\uparrow(x,a) = \overline{\theta}_{t,k}^{(j)}$, and $Q_{j,t}^\downarrow(x,a) = \underline{\theta}_{t,k}^{(j)}$. Moreover, (x,a,t)'s in the same partition have the same optimistic and pessimistic Q-values.

It is also worth pointing out that by definition of ρ, if (x,a,t) and (x',a',t) are in the same partition, then we have $|Q^*_t(x,a)-Q^*_t(x',a')|\le 2\rho$. To see it, let $\tilde{Q}\in\arg\min_{Q\in\mathcal{Q}}\|Q-Q^*\|_\infty$, then we have $|\tilde{Q}_t(x,a)-Q^*_t(x,a)|\le\rho$ and $|\tilde{Q}_t(x',a')-Q^*_t(x',a')|\le\rho$. Since $\tilde{Q}\in\mathcal{Q}$ and (x,a,t) and (x',a',t) are in the same partition, we have $\tilde{Q}_t(x,a)=\tilde{Q}_t(x',a')$. Then from the triangular inequality, we have $|Q^*_t(x,a)-Q^*_t(x',a')|\le 2\rho$.

The following lemma states that if $Q^{\uparrow}_{j,t}(x,a)<\infty$, then it is "close" to $Q^*_t(x,a)$.

Lemma 5. *$\forall(x,a,t)$ and $\forall j=0,1,\cdots$, if $Q^{\uparrow}_{j,t}(x,a)<\infty$, then $|Q^{\uparrow}_{j,t}(x,a)-Q^*_t(x,a)|\le 2\rho(H-t)$.*

Please refer to the appendix for the detailed proof of Lemma 5. Based on this lemma, we have the following result:

Lemma 6. *$\forall j=0,1,\cdots$, if $Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})<\infty$ for any $t=0,1,\cdots,H-1$, then we have $V^*_0(x_{j,0})-R^{(j)}\le 2\rho H(H+1)=O(\rho H^2)$. Furthermore, if the conditions of Proposition 1 hold, then we have $V^*_0(x_{j,0})-R^{(j)}\le 6\rho H=O(\rho H)$.*

Please refer to the appendix for the detailed proof of Lemma 6. Obviously, Proposition 1 directly follows from Lemma 6.

For any $j=0,1,\cdots$, we define t^*_j as the last period t in episode j s.t. $Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})=\infty$. If $Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})<\infty$ for all $t=0,1,\cdots,H-1$, we define $t^*_j=\text{NULL}$. We then have the following lemma:

Lemma 7. *$\forall j=0,1,\cdots$, if $t^*_j\neq\text{NULL}$, then $\forall j'\le j$, $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j},a_{j,t^*_j})=\infty$, and $\forall j'>j$, $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j},a_{j,t^*_j})<\infty$* (Exploration)*. Otherwise, if $t^*_j=\text{NULL}$, then $V^*_0(x_{j,0})-R^{(j)}\le 2\rho H(H+1)$* (Exploitation)*. Furthermore, $\sum_{j=0}^{\infty}\mathbf{1}[t^*_j\neq\text{NULL}]\le K$, where K is the number of partitions.*

Again, please refer to the appendix for the proof of Lemma 7. Note that Theorem 3 directly follows from Lemma 7.

6. Computational Efficiency of Optimistic Constraint Propagation

We now briefly discuss the computational complexity of OCP. As typical in the complexity analysis of optimization algorithms, we assume that basic operations include the arithmetic operations, comparisons, and assignment, and measure computational complexity in terms of the number of basic operations (henceforth referred to as operations) per period.

First, it is worth pointing out that for a general hypothesis class $\mathcal{Q}$ and general action space $\mathcal{A}$, the per period computations of OCP can be intractable. This is because:

- Computing $\sup_{Q\in\mathcal{Q}_\mathcal{C}}Q_t(x_{j,t},a)$, $U_{j,t}$ and $L_{j,t}$ requires solving a possibly intractable optimization problems.
- Selecting an action that maximizes $\sup_{Q\in\mathcal{Q}_\mathcal{C}}Q_t(x_{j,t},a)$ can be intractable.

Further, the number of constraints in $\mathcal{C}$, and with it the number of operations per period, can grow over time.

However, if $|\mathcal{A}|$ is tractably small and $\mathcal{Q}$ has some special structures (e.g. $\mathcal{Q}$ is a finite set or a linear subspace or, more generally a polytope), then by discarding the "redundant" constraints in $\mathcal{C}$, OCP with a variant of the constraint selection algorithm will be computationally efficient, and the sample efficiency results developed in Section 5 will still hold. Due to space limitations, we only discuss the scenario where $\mathcal{Q}$ is a polytope of dimension d. Note that the finite state/action tabula rasa case, the linear-quadratic case, and the state aggregation case are all special cases of this scenario. Moreover, as we have discussed before, for the finite state/action tabula rasa case and the linear-quadratic case, $Q^*\in\mathcal{Q}$.

Specifically, if $\mathcal{Q}$ is a polytope of dimension d (i.e., within a d-dimensional subspace), then any $Q\in\mathcal{Q}$ can be represented by a weight vector $\theta\in\Re^d$, and $\mathcal{Q}$ can be characterized by a set of linear inequalities of θ. Furthermore, the new constraints of the form $L_{j,t} \leq Q_t(x_{j,t}, a_{j,t}) \leq U_{j,t}$ are also linear inequalities of θ. Hence, in each episode, $\mathcal{Q}_{\mathcal{C}}$ is characterized by a polyhedron in $\Re^d$, and $\sup_{Q \in \mathcal{Q}_{\mathcal{C}}} Q_t(x_{j,t}, a)$, $U_{j,t}$ and $L_{j,t}$ can be computed by solving linear programming (LP) problems. If we assume that each observed numerical value can be encoded by B bits, and LPs are solved by Karmarkar's algorithm [11], then the following proposition bounds the computational complexity.

Proposition 2. *If $\mathcal{Q}$ is a polytope of dimension d, each numerical value in the problem data or observed in the course of learning can be represented with B bits, and OCP uses Karmarkar's algorithm to solve linear programs, then the computational complexity of OCP is $O\left(\left[|\mathcal{A}| + |\mathcal{C}|\right] |\mathcal{C}| d^{4.5} B\right)$ operations per period.*

Proof Note that OCP needs to perform the following computation in one period:

1. Construct $\mathcal{Q}_{\mathcal{C}}$ by constraint selection algorithm. This requires sorting $|\mathcal{C}|$ constraints by comparing their upper bounds and positions in the sequence (with $O\left(|\mathcal{C}| \log |\mathcal{C}|\right)$ operations), and checking whether $\mathcal{Q}_{\mathcal{C}} \cap \mathcal{C}_\tau \neq \varnothing$ for $|\mathcal{C}|$ times. Note that checking whether $\mathcal{Q}_{\mathcal{C}} \cap \mathcal{C}_\tau \neq \varnothing$ requires solving an LP feasibility problem with d variables and $O\left(|\mathcal{C}|\right)$ constraints.
2. Choose action $a_{j,t}$. Note that $\sup_{Q \in \mathcal{Q}_{\mathcal{C}}} Q_t(x_{j,t}, a)$ can be computed by solving an LP with d variables and $O\left(|\mathcal{C}|\right)$ constraints, thus $a_{j,t}$ can be derived by solving $|\mathcal{A}|$ such LPs.
3. Compute the new constraint $L_{j,t} \leq Q_t(x_{j,t}, a_{j,t}) \leq U_{j,t}$. Note $U_{j,t}$ can be computed by solving $|\mathcal{A}|$ LPs with d variables and $O\left(|\mathcal{C}|\right)$ constraints, and $L_{j,t}$ can be computed by solving one LP with d variables and $O\left(|\mathcal{C}| + |\mathcal{A}|\right)$ constraints.

If we assume that each observed numerical value can be encoded by B bits, and use Karmarkar's algorithm to solve LPs, then for an LP with d variables and m constraints, the number of bits input to Karmarkar's algorithm is $O\left(mdB\right)$, and hence it requires $O\left(mBd^{4.5}\right)$ operations to solve the LP. Thus, the computational complexities for the first, second, third steps are $O\left(|\mathcal{C}|^2 d^{4.5} B\right)$, $O\left(|\mathcal{A}||\mathcal{C}| d^{4.5} B\right)$ and $O\left(|\mathcal{A}||\mathcal{C}| d^{4.5} B\right)$, respectively. Hence, the computational complexity of OCP is $O\left(\left[|\mathcal{A}| + |\mathcal{C}|\right] |\mathcal{C}| d^{4.5} B\right)$ operations per period. **q.e.d.**

Notice that the computational complexity is polynomial in d, B, $|\mathcal{C}|$ and $|\mathcal{A}|$, and thus, OCP will be computationally efficient if all these parameters are tractably small. Note that the bound in Proposition 2 is a worst-case bound, and the $O(d^{4.5})$ term is incurred by the need to solve LPs. For some special cases, the computational complexity is much less. For instance, in the state aggregation case, the computational complexity is $O\left(|\mathcal{C}| + |\mathcal{A}| + d\right)$ operations per period.

As we have discussed above, one can ensure that $|\mathcal{C}|$ remains bounded by using variants of the constraint selection algorithm (Algorithm 1) that only use a subset of the available constraints. For instance, in the coherent learning case discussed in Section 5.2, we can use a constraint selection algorithm that only chooses the constraints that will lead to a strict reduction of the eluder dimension of the hypothesis class. Obviously, with this constraint selection algorithm, $|\mathcal{C}| \leq |\mathcal{C}_{-1}| + \dim_E\left(\mathcal{Q}\right)$ always holds, where $\mathcal{C}_{-1}$ is the set of constraints defining $\mathcal{Q}$. Similarly, in the state aggregation case considered in Section 5.3, we can use a constraint selection algorithm that only chooses the constraints that reduce the optimistic Q-values of disjoint subsets from from infinity to finite. Obviously, with this constraint selection algorithm, $|\mathcal{C}| \leq |\mathcal{C}_{-1}| + K$ always holds, where K is the number of partitions. Based on our analysis, it can be shown that with these constraint selection algorithms, the performance bounds derived in Section 5 will still hold. Finally, for the general agnostic learning case, one naive approach is to maintain a time window W, and only constraints observed in episode $j - W, \cdots, j-1$ are used to construct $\mathcal{Q}_{\mathcal{C}}$ in episode j.

7. Experiment Results

In this section, we present computational results involving two illustrative examples: a synthetic deterministic Markov chain discussed in Van Roy and Wen [35] and the inverted pendulum problem considered in Lagoudakis et al. [14]. We compare OCP against least-squares value iteration (LSVI), a classical reinforcement learning algorithm.

7.1. Learning in a Deterministic Chain First, we consider the following example described in Van Roy and Wen [35] for which least-squares value iteration (LSVI) with Boltzmann exploration or ϵ-greedy exploration (see [35] for algorithm description) requires exponentially many episodes to learn an optimal policy, even in a coherent learning context with a small number of basis functions.

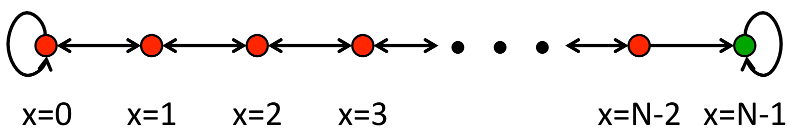


FIGURE 3. Deterministic system for which LSVI with Boltzmann/ϵ-greedy exploration is inefficient.

EXAMPLE 2. Consider the deterministic system illustrated in Figure 3. Each node represents a state, and each arrow corresponds to a possible state transition. The state space is $\mathcal{S} = \{0, 1, \cdots, N-1\}$ and the action space is $\mathcal{A} = \{a^{(1)}, a^{(2)}\}$. If the agent takes action $a^{(1)}$ at state $x = 0, 1, \cdots, N-2$ (the red nodes), the the state transitions to $y = [x-1]^+$. On the other hand, if the agent takes action $a^{(2)}$ at state $x = 0, 1, \cdots, N-2$, the state transitions to $y = x+1$. State $N-1$ (the green node) is absorbing. We assume a reward of 0 is realized upon any transition from a red node and a reward of 1 is realized upon any transition from the green node. We take the horizon H to be equal to the number of states N. The initial state in any episode is 0.

Let $\phi_{t,k}$ be a feature mapping $\mathcal{S} \times \mathcal{A}$ to $\Re$ for any $t = 0, 1, \cdots, H-1$ and any $k = 1, 2, \cdots, K$. We choose $\mathcal{Q}_t = \text{span}(\phi_{t,1}, \ldots, \phi_{t,K})$ and $\mathcal{Q} = \mathcal{Q}_0 \times \cdots \times \mathcal{Q}_{H-1}$, and consider the coherent learning case with $Q^* \in \mathcal{Q}$. Van Roy and Wen [35] shows that when LSVI with Boltzmann/ϵ-greedy exploration is applied to this problem, in expectation, it will take the agent at least $2^{|\mathcal{S}|-1}$ episodes to first reach the green node. In addition, its expected[4] regret Regret(T) is lower bounded by

$$\text{Regret}(T) \geq \left(2^{|\mathcal{S}|-1} - 1\right) \left(1 - \left[1 - 2^{-(|\mathcal{S}|-1)}\right]^{\lfloor T/H \rfloor}\right),$$

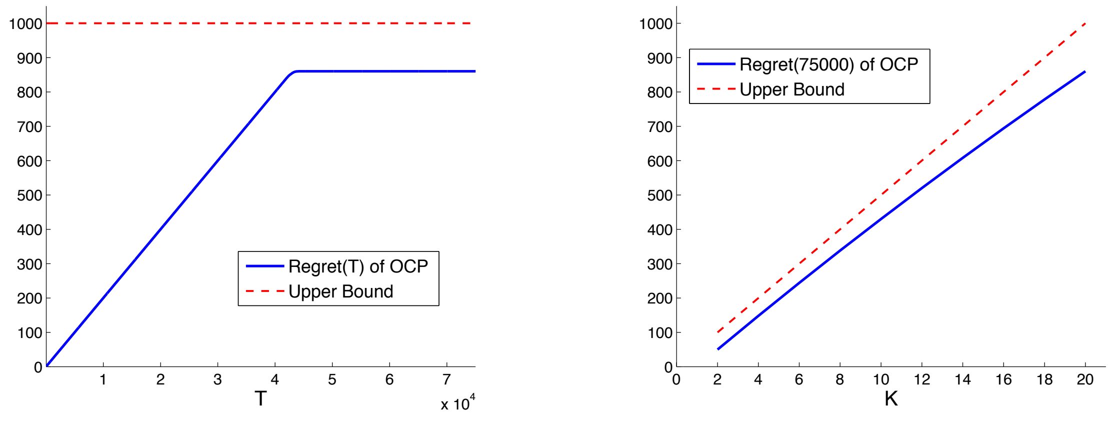
which implies that $\liminf_{T\to\infty} \text{Regret}(T) \geq 2^{|\mathcal{S}|-1} - 1$. These lower bounds hold for any choice of K and any choice of features.

In this experiment, we choose $N = |\mathcal{S}| = H = 50$, and run the simulation for 75000 time steps (i.e. 1500 episodes). Obviously, for this choice of N and T, the worst-case Regret(75000) for any reinforcement learning algorithm is 1500. From the above lower bounds, if we apply LSVI with Boltzmann/ϵ-greedy exploration to this problem, for any choice of features, in expectation it will take the agent 5.63×10^{14} episodes (2.81×10^{16} time steps) to first reach the green node, and Regret$(75000) \geq 1500 - 3 \times 10^{-12}$, which is extremely close to the worst-case regret. This shows that LSVI with Boltzmann/ϵ-greedy exploration is highly inefficient in this case.

We now describe our experiment setup for OCP, for which we need to specify how to choose features. We are interested in how the performance of OCP scales with K, the number of features at each period t, and vary $K = 2, 4, 6, \cdots, 20$. For a given K, we construct the features as follows: for each period $t = 0, 1, \cdots, H-1$, we choose $\phi_{t,1} = Q_t^*$[5], $\phi_{t,2} = \mathbf{1}$, a vector of all ones, and if $K > 2$, we sample $\phi_{t,3}, \cdots, \phi_{t,K}$ i.i.d. from the Gaussian distribution $N(0, I)$. For $K = 2$, we apply OCP to Example 2 with the above-specified features. Notice that in this case, one simulation is sufficient since the features, the OCP algorithm, and the dynamic system are all deterministic. On the other

[4] Since Boltzmann exploration and ϵ-greedy exploration are randomized exploration schemes, we should measure the performance of LSVI with Boltzmann/ϵ-greedy exploration with expected regret. We use the same symbol Regret(T) for the expected regret since the regret defined in this paper can be viewed as a special case of the expected regret.

[5] Notice that this ensures that $\mathcal{Q}$ is a coherent hypothesis class.



(a)Regret vs. T for $K = 20$ (b)Regret(75000) vs. K

FIGURE 4. Experiment Results in Example 2

hand, for $K > 2$, we apply OCP to Example 2 for 100 times, and each time we resample feature $\phi_{t,3}, \cdots, \phi_{t,K}$ for all t. We then average the results of these 100 simulations.

The experiment results are demonstrated in Figure 4. Specifically, in Figure 4(a), we fix $K = 20$ and vary $T = 50, 100, \cdots, 75000$, and plot Regret($T$) as a function of T. In Figure 4(b), we fix $T = 75000$ and vary $K = 2, 4, \cdots, 20$, and plot Regret(75000) as a function of K. From Theorem 1, in this problem, the $O(1)$ bound on Regret(T) of OCP is $HK = 50K$. We also plot this $O(1)$ upper bound in the figures.

We now briefly discuss the experiment results. Note that in this problem, the *realized regret* in an episode is either 0 or 1, depending on whether or not the agent reaches the green node in that episode. Figure 4(a) shows that for $K = 20$, it takes the agent about 900 episodes to learn how to reach the green node. Based on our discussion above, this result demonstrates the dramatic efficiency gains of OCP over LSVI with Boltzmann/ϵ-greedy exploration in this problem. On the other hand, Figure 4(b) shows that Regret(75000) scales linearly with K. The experiment results also indicate that the $O(1)$ upper bound derived in Theorem 1 is not tight in this problem, but the gap is small.

7.2. Inverted Pendulum We will now show that OCP significantly outperforms LSVI with ϵ-greedy exploration in a reinforcement learning formulation of an inverted pendulum problem. The system dynamics of an inverted pendulum on a cart are described in Equation (18) of Wang et al. [36], which is

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{g\sin(x_1) - \alpha m l x_2^2 \sin(2x_1)/2 - \alpha\cos(x_1)u}{4l/3 - \alpha m l \cos^2(x_1)} \end{aligned} \tag{7}$$

where x_1 is the angular position (in radians) of the pendulum from the vertical, x_2 is the angular velocity, $g = 9.8\text{m/s}^2$ is the gravity constant, $m = 2\text{kg}$ is the mass of the pendulum, $M = 8\text{kg}$ is the mass of the cart, $l = 0.5\text{m}$ is the length of the pendulum, $\alpha = 1/(m+M) = 0.1\text{kg}^{-1}$, and u is the force applied to the cart (in Newtons). Note that $\dot{x}_1$ and $\dot{x}_2$ are respectively the derivatives of x_1 and x_2 with respect to time. Similarly as Lagoudakis et al. [14], we simulate this nonlinear system with a step size 0.1s. The action space $\mathcal{A} = \{-50, 0, 50\}$, but the actual input to the system can be noisy. Specifically, when action $a \in \mathcal{A}$ is selected, the actual input to the system is $u = a + \xi_a$, where ξ_a is a random variable independently drawn from the uniform distribution unif$(-\delta, \delta)$ for some $\delta \geq 0$. The initial state of the system is $(x_1 = 0, x_2 = \xi_0)$, where ξ_0 is also independently drawn from unif$(-\delta, \delta)$. Notice that this dynamic system is deterministic if $\delta = 0$.

We consider a reinforcement learning setting in which an agent learns to control the inverted pendulum such that it does not fall for one hour while repeatedly interacting with it for 1000 episodes. The reward in each episode j is the length of time until the inverted pendulum falls, capped at one hour. We also assume that the agent does not know the system dynamics or the reward function. We apply OCP and LSVI with the same form of state aggregation to this problem. In particular, the state space of this problem is

$$\mathcal{S} = \{(x_1, x_2) : x_1 \in (-\pi/2, \pi/2),\, x_2 \in \Re\} \bigcup \{\text{inverted pendulum is fallen}\}.$$

We grid the angular position space $(-\pi/2, \pi/2)$ uniformly into 31 intervals; and grid the angular velocity space as $(-\infty, -x_2^{\max})$, $(x_2^{\max}, \infty)$ and 29 uniform intervals between $-x_2^{\max}$ and $x_2^{\max}$, where $x_2^{\max}$ is the maximum angular velocity observed when the initial state is $(0,0)$ and $u = 50$ for all the time steps. $\mathcal{S}$ is partitioned as follows: the first partition only includes the special state "inverted pendulum is fallen", and all the other 961 partitions are Cartesian products of intervals of x_1 and x_2 described above. We choose the basis functions as the indicator functions for each action-(state space partition) pair,[6] hence there are 2886 basis functions.

We demonstrate the experiment results for two cases: $\delta = 0$ and $\delta = 2.5$. For each case, we apply OCP and LSVI with exploration rate $\epsilon = 0.05, 0.1, 0.15$ to it. We also show the performance of a purely randomized policy as a baseline, under which each action in $\mathcal{A}$ is chosen uniformly randomly at each time. The experiment results are averaged over 100 simulations. Figure 5 plots the cumulative reward as a function of episode. Notice that the cumulative reward in the first J episodes is bounded by J hours since the per-episode reward is upper bounded by one hour.

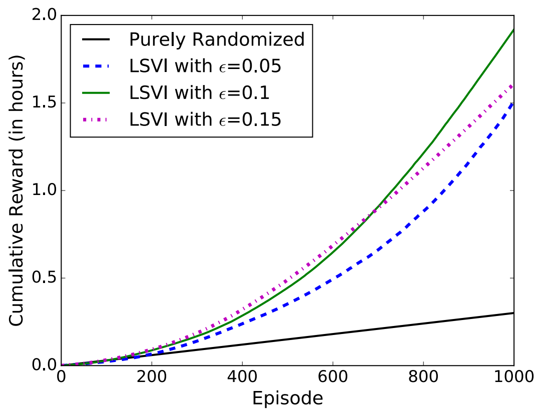
Figure 5(a) and 5(c) compare LSVI with ϵ-greedy exploration with the purely randomized policy. Notice that though LSVI significantly outperforms the purely randomized policy, its performance is unsatisfactory since in both cases its cumulative reward at 1000 episodes is less than 2 hours, indicating that in the first 1000 episodes the average time length until the pendulum falls is less than 7.2 seconds. Figure 5(b) and 5(d) compare OCP with the best LSVI ($\epsilon = 0.1$ in both cases). We observe that in both cases, the performance of OCP is orders of magnitude better than that of the LSVI. We also note that the performances of both OCP and LSVI are worse in the case with $\delta = 2.5$ than the case with $\delta = 0$, since the stochastic disturbances make the inverted pendulum problem more challenging.

Finally, we would like to emphasize that the system dynamics are stochastic in the case with $\delta = 2.5$. However, the magnitude of the stochastic disturbances, 2.5, is small relative to the magnitude of the control, 50. Thus, though OCP is motivated and developed in the framework of reinforcement learning in deterministic systems, it might also perform well in some reinforcement learning problems with stochastic environments (e.g. reinforcement learning in MDPs), especially when the magnitude of the stochastic disturbances is small.

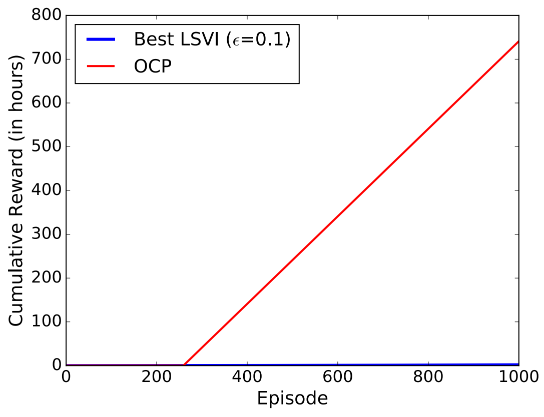
8. Conclusion

We have proposed a novel reinforcement learning algorithm, called optimistic constraint propagation (OCP), that synthesizes efficient exploration and value function generalization for episodic reinforcement learning in deterministic systems. We have shown that when the true value function Q^* lies in the given hypothesis class $\mathcal{Q}$ (the coherent learning case), OCP selects optimal actions over all but at most $\dim_E[\mathcal{Q}]$ episodes, where $\dim_E[\mathcal{Q}]$ is the eluder dimension of $\mathcal{Q}$. We have also established sample efficiency and asymptotic performance guarantees for the state aggregation case, a special agnostic learning case where $\mathcal{Q}$ is the span of pre-specified indicator

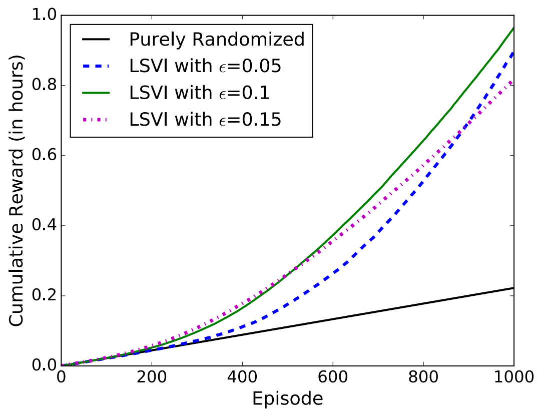
[6] The inverted pendulum problem is time-homogenous if it is not stopped by the time one hour. This motivates us to use basis functions independent of the period t.



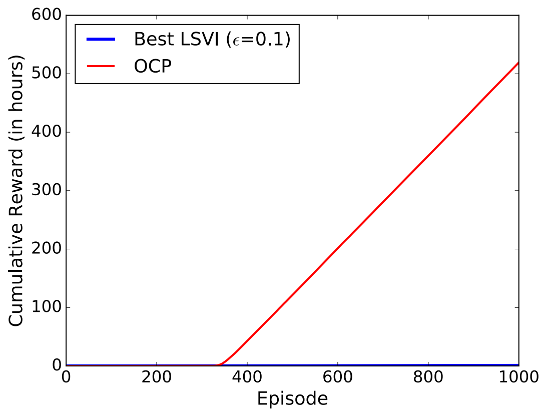
(a)$\delta = 0$: LSVI vs Purely Randomized



(b)$\delta = 0$: OCP vs Best LSVI



(c)$\delta = 2.5$: LSVI vs Purely Randomized



(d)$\delta = 2.5$: OCP vs Best LSVI

Figure 5. Experiment Results in Inverted Pendulum Example

functions over disjoint sets. We have also discussed the computational complexity of OCP and presented computational results involving two illustrative examples. Our results demonstrate dramatic efficiency gains enjoyed by OCP relative to LSVI with Boltzmann or ϵ-greedy exploration.

Finally, we briefly discuss some possible directions for future research. One possible direction is to propose a variant of OCP for reinforcement learning in infinite-horizon discounted deterministic systems. Note that for an infinite-horizon discounted problem with bounded rewards, its *effective horizon* is $\frac{1}{1-\gamma}$, where $\gamma \in (0,1)$ is the discount factor. We conjecture that with this notion of effective horizon, similar sample complexity/regret bounds can be derived for the infinite-horizon discounted problems. Another possible direction is to design provably sample efficient algorithms for the general agnostic learning case discussed in this paper. A more important problem is to design efficient algorithms for reinforcement learning in MDPs. Though many provably efficient algorithms have been proposed for the tabula rasa case of this problem (see [6, 30, 19, 21, 35] and references therein), however, how to design such algorithms with value function generalization is currently still open. Thus, one interesting direction for future research is to extend OCP, or a variant of it, to this problem.

Appendix A: Equivalence of OCP and Q-Learning in the Tabula Rasa Case We prove that in the finite state/action tabula rasa case, OCP is equivalent to Q-learning with learning rate 1 and initial Q-value $Q_t(x,a)=\infty$. To see it, notice that in this setting, the OCP algorithm imposes constraints on individual Q-values of all the state-action-period triples. Moreover, if we define the optimistic Q-function in an arbitrary episode j as

$$Q_t^{\uparrow}(x,a)=\sup_{Q\in\mathcal{Q}_{\mathcal{C}}} Q_t(x,a) \quad \forall (x,a,t),$$

then $a_{j,t}\in\arg\max_{a\in\mathcal{A}} Q_t^{\uparrow}(x_{j,t},a)$. Thus, the lower bound $L_{j,t}$'s do not matter in this setting since there is no value function generalization across (x,a,t)'s.

Note that by definition of $Q^{\uparrow}$, $U_{j,t}=R_t(x_{j,t},a_{j,t})+\sup_{a\in\mathcal{A}} Q_{t+1}^{\uparrow}(x_{j,t+1},a)$. Moreover, since $Q^*\in\mathcal{Q}$ in this case, as we will prove in Lemma 1, there are no conflicting constraints. Hence, in the next episode (episode $j+1$), the optimistic Q-function is updated as

$$Q_t^{\uparrow}(x,a)\leftarrow\begin{cases}\min\left\{Q_t^{\uparrow}(x,a),U_{j,t}\right\} & \text{if } (x,a,t)=(x_{j,t},a_{j,t},t)\\ Q_t^{\uparrow}(x,a) & \text{otherwise}\end{cases}$$

Notice that the above equation implies that $Q^{\uparrow}$ is a non-increasing function in episode j. Thus, to show OCP is equivalent to Q-learning, we only need to prove that $U_{j,t}\le Q_t^{\uparrow}(x_{j,t},a_{j,t})$ in episode j. Obviously, we only need to consider the case when $Q_t^{\uparrow}(x_{j,t},a_{j,t})<\infty$. Notice that this holds trivially for $t=H-1$, since when $t=H-1$, $U_{j,t}=R_t(x_{j,t},a_{j,t})$ always holds, and if $Q_t^{\uparrow}(x_{j,t},a_{j,t})<\infty$ then $Q_t^{\uparrow}(x_{j,t},a_{j,t})=R_t(x_{j,t},a_{j,t})$. On the other hand, if $t<H-1$ and $Q_t^{\uparrow}(x_{j,t},a_{j,t})<\infty$, then $Q_t^{\uparrow}(x_{j,t},a_{j,t})=U_{j',t}$ for some $j'<j$. Note that by definition of $U_{j,t}$, we have $Q_t^{\uparrow}(x_{j,t},a_{j,t})=U_{j',t}\ge U_{j,t}$ since $Q_{t+1}^{\uparrow}$ is a non-increasing function in j.

Appendix B: Eluder Dimension for the Sparse Linear Case We start by defining some useful terminologies and notations. For any $\theta\in\Re^K$, any $l\le K$ and any index set $\mathcal{I}=\{i_1,i_2,\cdots,i_l\}\subseteq\{1,2,\cdots,K\}$ with $i_1<i_2<\cdots<i_l$ and $|\mathcal{I}|=l\le K$, we use $\theta_{\mathcal{I}}$ to denote the subvector of θ associated with the index set $\mathcal{I}$, i.e. $\theta_{\mathcal{I}}=\left[\theta_{i_1},\theta_{i_2}\cdots,\theta_{i_l}\right]^T$.

For a sequence of vectors $\theta^{(1)},\theta^{(2)},\cdots\in\Re^K$, we say $\theta^{(k)}$ is linearly l-independent of its predecessors if there exists an index set $\mathcal{I}$ with $|\mathcal{I}|=l$ s.t. $\theta_{\mathcal{I}}^{(k)}$ is linearly independent of $\theta_{\mathcal{I}}^{(1)},\theta_{\mathcal{I}}^{(2)},\cdots,\theta_{\mathcal{I}}^{(k-1)}$. Let $N=|\mathcal{S}||\mathcal{A}|$, and use Φ_j^T to denote the jth row of Φ. For any $l\le K$, we define rank$[\Phi,l]$, the l-rank of Φ, as the length d of the longest sequence of Φ_j's such that every element is linearly l-independent of its predecessors. Recall that $\mathcal{Q}_0=\{\Phi\theta:\theta\in\Re^K,\|\theta\|_0\le K_0\}$, we have the following result:

PROPOSITION 3. *If $2K_0\le K$, then $\dim_{\mathrm{E}}[\mathcal{Q}_0]=\mathrm{rank}[\Phi,2K_0]$.*

Proof We use $y=(x,a)$ to denote a state-action pair, and use $\Phi(y)^T$ to denote the row of matrix Φ associated with y. Based on our definitions of eluder dimension and l-rank, it is sufficient to prove the following lemma:

LEMMA 8. *For any state-action pair y and for any set of state-action pairs $Y=\left\{y^{(1)},y^{(2)},\cdots,y^{(n)}\right\}$, y is independent of Y with respect to $\mathcal{Q}_0$ if and only if $\Phi(y)$ is linearly $2K_0$-independent of $\left\{\Phi(y^{(1)}),\Phi(y^{(2)}),\cdots,\Phi(y^{(n)})\right\}$.*

We now prove the above lemma. Note that based on the definition of independence (see Section 5.1), y is independent of Y with respect to $\mathcal{Q}_0$ if and only if there exist $Q_1,Q_2\in\mathcal{Q}_0$ s.t. $Q_1(y^{(i)})=Q_2(y^{(i)})$, $\forall i=1,2,\cdots,n$, and $Q_1(y)\neq Q_2(y)$. Based on the definition of function space $\mathcal{Q}_0$, there exist two K_0-sparse vectors $\theta^{(1)},\theta^{(2)}\in\Re^K$ s.t. $Q_1=\Phi\theta^{(1)}$ and $Q_2=\Phi\theta^{(2)}$. Thus, y is independent of Y with respect to $\mathcal{Q}_0$ if and only if there exist two K_0-sparse vectors $\theta^{(1)},\theta^{(2)}\in\Re^K$ s.t.

$$\begin{aligned}\Phi(y^{(i)})^T(\theta^{(1)}-\theta^{(2)})&=0 \quad \forall i=1,2,\cdots,n\\ \Phi(y)^T(\theta^{(1)}-\theta^{(2)})&\neq 0\end{aligned}$$

Based on the definition of K_0-sparsity, the above condition is equivalent to there exists a $2K_0$-sparse vector $\theta \in \Re^K$ s.t.

$$\begin{aligned} \Phi(y^{(i)})^T\theta &= 0 \quad \forall i = 1,2,\cdots,n \\ \Phi(y)^T\theta &\neq 0 \end{aligned}$$

To see it, note that if $\theta^{(1)}, \theta^{(2)}$ are K_0-sparse, then $\theta = \theta^{(1)} - \theta^{(2)}$ is $2K_0$-sparse. On the other hand, if θ is $2K_0$-sparse, then there exist two K_0-sparse vectors $\theta^{(1)}, \theta^{(2)}$ s.t. $\theta = \theta^{(1)} - \theta^{(2)}$.

Since θ is $2K_0$-sparse, there exists a set of indices $\mathcal{I}$ s.t. $|\mathcal{I}| = 2K_0$ and $\theta_i = 0$, $\forall i \notin \mathcal{I}$. Thus, the above condition is equivalent to

$$\begin{aligned} \Phi(y^{(i)})_{\mathcal{I}}^T\theta_{\mathcal{I}} &= 0 \quad \forall i = 1,2,\cdots,n \\ \Phi(y)_{\mathcal{I}}^T\theta_{\mathcal{I}} &\neq 0, \end{aligned}$$

which is further equivalent to $\Phi(y)_{\mathcal{I}}$ is linearly independent of $\Phi(y^{(1)})_{\mathcal{I}}, \Phi(y^{(2)})_{\mathcal{I}}, \cdots, \Phi(y^{(n)})_{\mathcal{I}}$. Since $|\mathcal{I}| = 2K_0$, from the definition of linear l-dependence, this is equivalent to $\Phi(y)$ is linearly $2K_0$-independent of $\Phi(y^{(1)}), \Phi(y^{(2)}), \cdots, \Phi(y^{(n)})$. **q.e.d.**

We now show that if Φ satisfies a technical condition, then $\text{rank}[\Phi, l] = l$. Specifically, for any $l \leq \min\{N, K\}$, we say Φ is l-full-rank if any submatrix of Φ with size $l \times l$ has full rank. Based on this notion, we have the following result:

Proposition 4. *For any $l \leq \min\{N, K\}$, if Φ is l-full-rank, then we have* $\text{rank}[\Phi, l] = l$.

Proof Consider any sequence of matrix rows $\Phi^{(1)}, \Phi^{(2)}, \cdots, \Phi^{(l+1)}$ with length $l+1$, and any index set $\mathcal{I}$ with $|\mathcal{I}| = l$. Since Φ is l-full-rank, thus $\Phi_{\mathcal{I}}^{(1)}, \Phi_{\mathcal{I}}^{(2)}, \cdots, \Phi_{\mathcal{I}}^{(l)} \in \Re^l$ are linearly independent (hence forms a basis in $\Re^l$). Thus, $\Phi_{\mathcal{I}}^{(l+1)}$ is linearly dependent on $\Phi_{\mathcal{I}}^{(1)}, \Phi_{\mathcal{I}}^{(2)}, \cdots, \Phi_{\mathcal{I}}^{(l)} \in \Re^l$. Since this result holds for any $\mathcal{I}$ with $|\mathcal{I}| = l$, thus $\Phi^{(l+1)}$ is linearly l-dependent on $\Phi^{(1)}, \Phi^{(2)}, \cdots, \Phi^{(l)} \in \Re^K$. Furthermore, since this result holds for any sequence of matrix rows with length $l+1$, thus we have $\text{rank}[\Phi, l] \leq l$.

On the other hand, since Φ is l-full-rank, choose any sequence of matrix rows $\Phi^{(1)}, \Phi^{(2)}, \cdots, \Phi^{(l)}$ with length l and any index set $\mathcal{I}$ with $|\mathcal{I}| = l$, $\Phi_{\mathcal{I}}^{(1)}, \Phi_{\mathcal{I}}^{(2)}, \cdots, \Phi_{\mathcal{I}}^{(l)}$ are linearly independent. Thus, $\Phi^{(1)}, \Phi^{(2)}, \cdots, \Phi^{(l)}$ is a sequence of matrix rows s.t. every element is linearly l-independent of its predecessors. Thus, $\text{rank}[\Phi, l] \geq l$. So we have $\text{rank}[\Phi, l] = l$. **q.e.d.**

Thus, if $2K_0 \leq \min\{N, K\}$ and Φ is $2K_0$-full-rank, then we have $\dim_{\mathrm{E}}[\mathcal{Q}_0] = \text{rank}\,[\Phi, 2K_0] = 2K_0$. Consequently, we have $\dim_{\mathrm{E}}[\mathcal{Q}] = \dim_{\mathrm{E}}[\mathcal{Q}_0^H] = 2K_0 H$.

Appendix C: Detailed Proof for Theorem 1

C.1. Proof for Lemma 1

Proof for Lemma 1 We prove this lemma by induction on j and choose the induction hypothesis as follows: $\forall j = 0, 1, \cdots$, we have (1) $Q^* \in \mathcal{Q}_{\mathcal{C}_j}$ and (2) $L_{j',t} \leq Q_t^*(x_{j',t}, a_{j',t}) \leq U_{j',t}$ for all $t = 0, 1, \cdots, H-1$ and all $j' = 0, 1, \cdots, j-1$.

First, we notice that the induction hypothesis is true for $j = 0$. To see it, notice that when $j = 0$, (2) holds trivially since $j - 1 < 0$; and (1) also holds since by definition $\mathcal{Q}_{\mathcal{C}_0} = \mathcal{Q}$, and hence $Q^* \in \mathcal{Q} = \mathcal{Q}_{\mathcal{C}_0}$. We now prove that if the induction hypothesis holds for episode j, then it also holds for episode $j+1$. We first show that (2) holds for episode $j+1$, which is sufficient to prove

$$L_{j,t} \leq Q_t^*(x_{j,t}, a_{j,t}) \leq U_{j,t} \quad \forall t = 0, 1, \cdots, H-1.$$

We prove the above inequality by considering two different cases. First, if $t = H-1$, then we have $U_{j,t} = L_{j,t} = R_t(x_{j,t}, a_{j,t}) = Q_t^*(x_{j,t}, a_{j,t})$, and hence the above inequality trivially holds. On the other hand, if $t < H-1$, then we have

$$\begin{aligned} U_{j,t} &= R_t(x_{j,t}, a_{j,t}) + \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t+1}(x_{j,t+1}, a) \\ &\geq R_t(x_{j,t}, a_{j,t}) + \sup_{a \in \mathcal{A}} Q_{t+1}^*(x_{j,t+1}, a) = Q_t^*(x_{j,t}, a_{j,t}), \end{aligned}$$

where the inequality follows from the induction hypothesis $Q^* \in \mathcal{Q}_{\mathcal{C}_j}$, and the last equality follows from the Bellman equation. Similarly, we also have

$$\begin{aligned} L_{j,t} &= R_t(x_{j,t}, a_{j,t}) + \inf_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t+1}(x_{j,t+1}, a) \\ &\leq R_t(x_{j,t}, a_{j,t}) + \sup_{a \in \mathcal{A}} Q^*_{t+1}(x_{j,t+1}, a) = Q^*_t(x_{j,t}, a_{j,t}). \end{aligned}$$

Hence, (2) holds for episode $j+1$. Since $Q^* \in \mathcal{Q}$ and (2) holds for episode $j+1$, then by definition of $\mathcal{Q}_{\mathcal{C}_{j+1}}$, we have $Q^* \in \mathcal{Q}_{\mathcal{C}_{j+1}}$. Thus, the induction hypothesis also holds for episode $j+1$. Hence, we have completed the proof for Lemma 1. **q.e.d.**

C.2. Proof for Lemma 2

Proof for Lemma 2 We prove this lemma by induction on j. First, notice that if $j = 0$, then from Algorithm 3, we have $\mathcal{Z}_0 = \varnothing$. Thus, Lemma 2(a) holds for $j = 0$.

Second, we prove that if Lemma 2(a) holds for episode j, then Lemma 2(b) holds for episode j and Lemma 2(a) holds for episode $j+1$. To see why Lemma 2(b) holds for episode j, notice that from Lemma 1, we have $Q^* \in \mathcal{Q}_{\mathcal{C}_j} \subseteq \mathcal{Q}$. Furthermore, from the induction hypothesis, $\forall z \in \mathcal{Z}_j$ and $\forall Q \in \mathcal{Q}_{\mathcal{C}_j}$, we have $Q(z) = Q^*(z)$. Since $(x_{j,t}, a_{j,t}, t)$ is dependent on $\mathcal{Z}_j$ with respect to $\mathcal{Q}$, then $\forall Q \in \mathcal{Q}_{\mathcal{C}_j} \subseteq \mathcal{Q}$, we have that $Q_t(x_{j,t}, a_{j,t}) = Q^*_t(x_{j,t}, a_{j,t})$. Hence we have $\sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x_{j,t}, a_{j,t}) = Q^*_t(x_{j,t}, a_{j,t})$, furthermore, from the OCP algorithm, we have $\sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x_{j,t}, a_{j,t}) \geq \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x_{j,t}, a)$, $\forall a \in \mathcal{A}$, thus we have

$$Q^*_t(x_{j,t}, a_{j,t}) = \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x_{j,t}, a_{j,t}) \geq \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_t(x_{j,t}, a) \geq Q^*_t(x_j, a), \quad \forall a \in \mathcal{A},$$

where the last inequality follows from the fact that $Q^* \in \mathcal{Q}_{\mathcal{C}_j}$. Thus, $a_{j,t}$ is optimal and $Q^*_t(x_{j,t}, a_{j,t}) = V^*_t(x_{j,t})$. Thus, Lemma 2(b) holds for episode j.

We now prove Lemma 2(a) holds for episode $j+1$. We prove the conclusion by considering two different scenarios. If $t^*_j = \text{NULL}$, then $\mathcal{Z}_{j+1} = \mathcal{Z}_j$ and $\mathcal{Q}_{\mathcal{C}_{j+1}} \subseteq \mathcal{Q}_{\mathcal{C}_j}$. Thus, obviously, Lemma 2(a) holds for episode $j+1$. On the other hand, if $t^*_j \neq \text{NULL}$, we have $\mathcal{Q}_{\mathcal{C}_{j+1}} \subseteq \mathcal{Q}_{\mathcal{C}_j}$ and $\mathcal{Z}_{j+1} = \left[\mathcal{Z}_j, (x_{j,t^*_j}, a_{j,t^*_j}, t^*_j)\right]$. Based on the induction hypothesis, $\forall z \in \mathcal{Z}_j$ and $\forall Q \in \mathcal{Q}_{\mathcal{C}_{j+1}} \subseteq \mathcal{Q}_{\mathcal{C}_j}$, we have $Q(z) = Q^*(z)$. Thus, it is sufficient to prove that

$$Q_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = Q^*_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}), \quad \forall Q \in \mathcal{Q}_{\mathcal{C}_{j+1}}. \tag{8}$$

We prove Eqn(8) by considering two different cases. First, if $t^*_j = H-1$, it is sufficient to prove that $Q_{H-1}(x_{j,H-1}, a_{j,H-1}) = R_{H-1}(x_{j,H-1}, a_{j,H-1})$, $\forall Q \in \mathcal{Q}_{\mathcal{C}_{j+1}}$, which holds by definition of $\mathcal{Q}_{\mathcal{C}_{j+1}}$ (see OCP algorithm, and recall that from Lemma 1, no constraints are conflicting if $Q^* \in \mathcal{Q}$). On the other hand, if $t^*_j < H-1$, it is sufficient to prove that for any $Q \in \mathcal{Q}_{\mathcal{C}_{j+1}}$, $Q_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = R_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) + V^*_{t^*_j+1}(x_{j,t^*_j+1})$. Recall that OCP algorithm add a constraint $L_{j,t^*_j} \leq Q_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) \leq U_{j,t^*_j}$ to $\mathcal{Q}_{\mathcal{C}_{j+1}}$ (and again, recall that no constraints are conflicting if $Q^* \in \mathcal{Q}$). Based on the definitions of L_{j,t^*_j} and U_{j,t^*_j}, it is sufficient to prove that

$$V^*_{t^*_j+1}(x_{j,t^*_j+1}) = \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a) = \inf_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a). \tag{9}$$

We first prove that $V^*_{t^*_j+1}(x_{j,t^*_j+1}) = \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a)$. Specifically, we have that

$$\sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} \sup_{a \in \mathcal{A}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a) = \sup_{a \in \mathcal{A}} \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a) = \sup_{Q \in \mathcal{Q}_{\mathcal{C}_j}} Q_{t^*_j+1}(x_{j,t^*_j+1}, a_{j,t^*_j+1}) = V^*_{t^*_j+1}(x_{j,t^*_j+1}),$$

where the second equality follows from the fact that $a_{j,t_j^*+1} \in \arg\max_{a\in\mathcal{A}} \sup_{Q\in\mathcal{Q}_{\mathcal{C}_j}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a)$ and the last equality follows from the definition of t_j^* and Part (b) of the lemma for episode j (which we have just proved above, and holds by the induction hypothesis). Specifically, since t_j^* is the last period in episode j s.t. $(x_{j,t}, a_{j,t}, t)$ is independent of $\mathcal{Z}_j$ with respect to $\mathcal{Q}$. Thus, $(x_{j,t_j^*+1}, a_{j,t_j^*+1}, t_j^*+1)$ is dependent on $\mathcal{Z}_j$ with respect to $\mathcal{Q}$. From Lemma 2(b) for episode j, we have $V^*_{t_j^*+1}(x_{j,t_j^*+1}) = Q_{t_j^*+1}(x_{j,t_j^*+1}, a_{j,t_j^*+1})$ for any $Q\in\mathcal{Q}_{\mathcal{C}_j}$. Thus, $\sup_{Q\in\mathcal{Q}_{\mathcal{C}_j}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a_{j,t_j^*+1}) = V^*_{t_j^*+1}(x_{j,t_j^*+1}) = \inf_{Q\in\mathcal{Q}_{\mathcal{C}_j}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a_{j,t_j^*+1})$. On the other hand, we have that

$$\inf_{Q\in\mathcal{Q}_{\mathcal{C}_j}} \sup_{a\in\mathcal{A}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a) \geq \sup_{a\in\mathcal{A}} \inf_{Q\in\mathcal{Q}_{\mathcal{C}_j}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a) \geq \inf_{Q\in\mathcal{Q}_{\mathcal{C}_j}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a_{j,t_j^*+1}) = V^*_{t_j^*+1}(x_{j,t_j^*+1}),$$

where the first inequality follows from the max-min inequality, the second inequality follows from the fact that $a_{j,t_j^*+1}\in\mathcal{A}$, and we have just proved the last equality above. Hence we have

$$V^*_{t_j^*+1}(x_{j,t_j^*+1}) = \sup_{Q\in\mathcal{Q}_{\mathcal{C}_j}} \sup_{a\in\mathcal{A}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a) \geq \inf_{Q\in\mathcal{Q}_{\mathcal{C}_j}} \sup_{a\in\mathcal{A}} Q_{t_j^*+1}(x_{j,t_j^*+1}, a) \geq V^*_{t_j^*+1}(x_{j,t_j^*+1}).$$

Thus, Eqn(9) holds. Hence, Lemma 2(a) holds for episode $j+1$, and by induction, we have proved Lemma 2. **q.e.d.**

C.3. Proof for Lemma 3

Proof for Lemma 3 Note that from Algorithm 3, if $t_j^* = \text{NULL}$, then $\forall t = 0,1,\cdots,H-1$, $(x_{j,t}, a_{j,t}, t)$ is dependent on $\mathcal{Z}_j$ with respect to $\mathcal{Q}$. Thus, from Lemma 2(b), $a_{j,t}$ is optimal $\forall t = 0,1,\cdots,H-1$. Hence we have $R^{(j)} = \sum_{t=0}^{H-1} R_t(x_{j,t}, a_{j,t}) = V_0^*(x_{j,0})$.

On the other hand, $t_j^* \neq \text{NULL}$, then from Algorithm 3, $(x_{j,t_j^*}, a_{j,t_j^*}, t_j^*)$ is independent of $\mathcal{Z}_j$ and $|\mathcal{Z}_{j+1}| = |\mathcal{Z}_j|+1$. Note $(x_{j,t_j^*}, a_{j,t_j^*}, t_j^*) \in \mathcal{Z}_{j+1}$, hence from Lemma 2(a), $\forall Q\in\mathcal{Q}_{\mathcal{C}_{j+1}}$, we have $Q_{t_j^*}(x_{j,t_j^*}, a_{j,t_j^*}) = Q^*_{t_j^*}(x_{j,t_j^*}, a_{j,t_j^*})$. **q.e.d.**

C.4. Proof for Theorem 1 Based on Lemma 3

Proof for Theorem 1 Notice that $\forall j = 0,1,\cdots$, $R^{(j)} \leq V_0^*(x_{j,0})$ by definition. Thus, from Lemma 3, $R^{(j)} < V_0^*(x_{j,0})$ implies that $t_j^* \neq \text{NULL}$. Hence, for any $j = 0,1,\cdots$, we have $\mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right] \leq \mathbf{1}\left[t_j^* \neq \text{NULL}\right]$. Furthermore, notice that from the definition of $\mathcal{Z}_j$, we have $\mathbf{1}\left[t_j^* \neq \text{NULL}\right] = |\mathcal{Z}_{j+1}| - |\mathcal{Z}_j|$, where $|\cdot|$ denotes the length of the given sequence. Thus for any $J = 0,1,\cdots$, we have

$$\sum_{j=0}^{J} \mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right] \leq \sum_{j=0}^{J} \mathbf{1}\left[t_j^* \neq \text{NULL}\right] = \sum_{j=0}^{J} \left[|\mathcal{Z}_{j+1}| - |\mathcal{Z}_j|\right] = |\mathcal{Z}_{J+1}| - |\mathcal{Z}_0| = |\mathcal{Z}_{J+1}|, \tag{10}$$

where the last equality follows from the fact that $|\mathcal{Z}_0| = |\varnothing| = 0$. Notice that by definition (see Algorithm 3), $\forall j = 0,1,\cdots$, $\mathcal{Z}_j$ is a sequence of elements in $\mathcal{Z}$ such that every element is independent of its predecessors with respect to $\mathcal{Q}$. Hence, from the definition of eluder dimension, we have $|\mathcal{Z}_j| \leq \dim_{\mathrm{E}}[\mathcal{Q}]$, $\forall j = 0,1,\cdots$. Combining this result with Eqn(10), we have $\sum_{j=0}^{J} \mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right] \leq |\mathcal{Z}_{J+1}| \leq \dim_{\mathrm{E}}[\mathcal{Q}]$, $\forall J = 0,1,\cdots$. Finally, notice that $\sum_{j=0}^{J} \mathbf{1}\left[V_j < V_0^*(x_{j,0})\right]$ is a non-decreasing function of J, and is bounded above by $\dim_{\mathrm{E}}[\mathcal{Q}]$. Thus, $\lim_{J\to\infty} \sum_{j=0}^{J} \mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right] = \sum_{j=0}^{\infty} \mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right]$ exists, and satisfies $\sum_{j=0}^{\infty} \mathbf{1}\left[R^{(j)} < V_0^*(x_{j,0})\right] \leq \dim_{\mathrm{E}}[\mathcal{Q}]$. Hence we have $\left|\left\{j : R^{(j)} < V_0^*(x_{j,0})\right\}\right| \leq \dim_{\mathrm{E}}[\mathcal{Q}]$. **q.e.d.**

Appendix D: Detailed Proof for Theorem 3 and Proposition 1

D.1. Proof for Lemma 5

Proof for Lemma 5 We prove Lemma 5 by induction on j. Note that when $j=0$, $\forall(x,a,t)$, $Q^{\uparrow}_{j,t}(x,a)=\infty$. Thus, Lemma 5 trivially holds for $j=0$.

We now prove that if Lemma 5 holds for episode j, then it also holds for episode $j+1$, for any $j=0,1,\cdots$. To prove this result, it is sufficient to show that for any (x,a,t) whose associated optimistic Q-value has been updated in episode j (i.e. $Q^{\uparrow}_{j,t}(x,a)\neq Q^{\uparrow}_{j+1,t}(x,a)$), if the new optimistic Q-value $Q^{\uparrow}_{j+1,t}(x,a)$ is still finite, then we have $|Q^{\uparrow}_{j+1,t}(x,a)-Q^*_t(x,a)|\leq 2\rho(H-t)$.

Note that if $Q^{\uparrow}_{j,t}(x,a)\neq Q^{\uparrow}_{j+1,t}(x,a)$, then (x,a,t) must be in the same partition $\mathcal{Z}_{t,k}$ as $(x_{j,t},a_{j,t},t)$. Noting that $\sup_{Q\in\mathcal{Q}_{\mathcal{C}_j}}\sup_{b\in\mathcal{A}}Q_{t+1}(x_{j,t+1},b)=\sup_{b\in\mathcal{A}}Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)$, from the discussion in Section 5.3, we have

$$Q^{\uparrow}_{j+1,t}(x,a)=\overline{\theta}^{(j+1)}_{t,k}=\begin{cases}R_{H-1}(x_{j,H-1},a_{j,H-1}) & \text{if } t=H-1\\ R_t(x_{j,t},a_{j,t})+\sup_{b\in\mathcal{A}}Q^{\uparrow}_{j,t+1}(x_{j,t+1},b) & \text{if } t<H-1\end{cases}$$

We now prove $|Q^{\uparrow}_{j+1,t}(x,a)-Q^*_t(x,a)|\leq 2\rho(H-t)$ by considering two different scenarios. First, if $t=H-1$, then $Q^{\uparrow}_{j+1,t}(x,a)=R_{H-1}(x_{j,H-1},a_{j,H-1})=Q^*_{H-1}(x_{j,H-1},a_{j,H-1})$. From our discussion above, we have $|Q^*_t(x,a)-Q^*_{H-1}(x_{j,H-1},a_{j,H-1})|\leq 2\rho$, which implies that $|Q^*_t(x,a)-Q^{\uparrow}_{j+1,t}(x,a)|\leq 2\rho=2\rho(H-t)$. On the other hand, if $t<H-1$, then $Q^{\uparrow}_{j+1,t}(x,a)=R_t(x_{j,t},a_{j,t})+\sup_{b\in\mathcal{A}}Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)$. If $Q^{\uparrow}_{j+1,t}(x,a)<\infty$, then $Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)<\infty$, $\forall b\in\mathcal{A}$. Furthermore, from the induction hypothesis, $Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)<\infty$, $\forall b\in\mathcal{A}$, implies that $\forall b\in\mathcal{A}$, $\left|Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)-Q^*_{t+1}(x_{j,t+1},b)\right|\leq 2\rho(H-t-1)$. On the other hand, from the Bellman equation at $(x_{j,t},a_{j,t},t)$, we have that $Q^*_t(x_{j,t},a_{j,t})=R_t(x_{j,t},a_{j,t})+\sup_{b\in\mathcal{A}}Q^*_{t+1}(x_{j,t+1},b)$. Thus,

$$\begin{aligned}\left|Q^{\uparrow}_{j+1,t}(x,a)-Q^*_t(x_{j,t},a_{j,t})\right| &= \left|\sup_{b\in\mathcal{A}}Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)-\sup_{b\in\mathcal{A}}Q^*_{t+1}(x_{j,t+1},b)\right|\\ &\leq \sup_{b\in\mathcal{A}}\left|Q^{\uparrow}_{j,t+1}(x_{j,t+1},b)-Q^*_{t+1}(x_{j,t+1},b)\right|\leq 2\rho(H-t-1).\end{aligned}$$

Moreover, since (x,a,t) and $(x_{j,t},a_{j,t},t)$ are in the same partition, we have $|Q^*_t(x,a)-Q^*_t(x_{j,t},a_{j,t})|\leq 2\rho$, consequently, we have $\left|Q^{\uparrow}_{j+1,t}(x,a)-Q^*_t(x,a)\right|\leq 2\rho(H-t)$. Thus, Lemma 5 holds for episode $j+1$. By induction, we have proved Lemma 5. **q.e.d.**

D.2. Proof for Lemma 6

Proof for Lemma 6 Notice that from OCP algoriothm, $\forall t=0,1,\cdots,H-1$, we have $Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})\geq Q^{\uparrow}_{j,t}(x_{j,t},a)$, $\forall a\in\mathcal{A}$. Thus, if $Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})<\infty$ for any t, then $Q^{\uparrow}_{j,t}(x_{j,t},a)<\infty$, $\forall(a,t)$. Consequently, from Lemma 5, we have that $\forall(a,t)$, $\left|Q^*_t(x_{j,t},a)-Q^{\uparrow}_{j,t}(x_{j,t},a)\right|\leq 2\rho(H-t)$. Thus, for any t, we have

$$Q^*_t(x_{j,t},a_{j,t})+2\rho(H-t)\geq Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})\geq Q^{\uparrow}_{j,t}(x_{j,t},a)\geq Q^*_t(x_{j,t},a)-2\rho(H-t),\quad \forall a\in\mathcal{A},$$

which implies that $Q^*_t(x_{j,t},a_{j,t})\geq\sup_{a\in\mathcal{A}}Q^*_t(x_{j,t},a)-4\rho(H-t)=V^*_t(x_{j,t})-4\rho(H-t)$, $\forall t$.

We first prove that $V^*_0(x_{j,0})-R^{(j)}\leq 2\rho H(H+1)$. Note that combining the above inequality with Bellman equation, we have that $R_t(x_{j,t},a_{j,t})\geq V^*_t(x_{j,t})-V^*_{t+1}(x_{j,t+1})-4\rho(H-t)$ for any $t<H-1$ and $R_{H-1}(x_{j,H-1},a_{j,H-1})\geq V^*_{H-1}(x_{j,H-1})-4\rho$. Summing up these inequalities, we have $V^*_0(x_{j,0})-R^{(j)}\leq 2\rho H(H+1)$.

We now prove that $V^*_0(x_{j,0})-R^{(j)}\leq 6\rho H$ if the conditions of Proposition 1 hold. Note that the conditions of Proposition 1 imply that $U_{j,t}\geq Q^{\uparrow}_{j,t}(x_{j,t},a_{j,t})\geq Q^{\downarrow}_{j,t}(x_{j,t},a_{j,t})\geq L_{j,t}$ for any t. Note that by definition, $U_{j,H-1}=L_{j,H-1}=R_{H-1}(x_{j,H-1},a_{j,H-1})$, and for $t<H-1$, we have $U_{j,t}=R_t(x_{j,t},a_{j,t})+Q^{\uparrow}_{j,t+1}(x_{j,t+1},a_{j,t+1})$, and

$$L_{j,t}\geq R_t(x_{j,t},a_{j,t})+\sup_{a\in\mathcal{A}}Q^{\downarrow}_{j,t+1}(x_{j,t+1},a)\geq R_t(x_{j,t},a_{j,t})+Q^{\downarrow}_{j,t+1}(x_{j,t+1},a_{j,t+1}),$$

where the first inequality follows from the definition of $L_{j,t}$ and max-min inequality, and the second inequality follows from the fact that $a_{j,t+1} \in \mathcal{A}$. Combining the above inequalities, we have $Q^{\downarrow}_{j,0}(x_{j,0}, a_{j,0}) \geq \sum_{t=0}^{H-1} R_t(x_{j,t}, a_{j,t}) = R^{(j)} \geq Q^{\uparrow}_{j,0}(x_{j,0}, a_{j,0}) \geq Q^{\downarrow}_{j,0}(x_{j,0}, a_{j,0})$. Thus we have $Q^{\uparrow}_{j,0}(x_{j,0}, a_{j,0}) = Q^{\downarrow}_{j,0}(x_{j,0}, a_{j,0}) = R^{(j)} < \infty$. So from Lemma 5, $\left|R^{(j)} - Q^*_0(x_{j,0}, a_{j,0})\right| = \left|Q^{\uparrow}_{j,0}(x_{j,0}, a_{j,0}) - Q^*_0(x_{j,0}, a_{j,0})\right| \leq 2\rho H$. Thus, $R^{(j)} \geq Q^*_0(x_{j,0}, a_{j,0}) - 2\rho H$. Furthermore, from the above analysis, $Q^*_0(x_{j,0}, a_{j,0}) \geq V^*_0(x_{j,0}) - 4\rho H$. Thus we have $R^{(j)} \geq V^*_0(x_{j,0}) - 6\rho H$. **q.e.d.**

D.3. Proof for Lemma 7

Proof for Lemma 7 $\forall j = 0, 1, \cdots$, if $t^*_j = \text{NULL}$, then by definition of t^*_j and Lemma 6, we have $V^*_0(x_{j,0}) - R^{(j)} \leq 2\rho H(H+1)$. On the other hand, if $t^*_j \neq \text{NULL}$, then by definition of t^*_j, $Q^{\uparrow}_{j,t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \infty$. We now show that $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) < \infty$ for all $j' > j$, and $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \infty$ for all $j' \leq j$.

Assume that $(x_{j,t^*_j}, a_{j,t^*_j}, t^*_j)$ belongs to partition $\mathcal{Z}_{t^*_j,k}$, thus $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \overline{\theta}^{(j')}_{t^*_j,k}$, $\forall j'$. Based on our discussion above, $\overline{\theta}^{(j')}_{t^*_j,k}$ is monotonically non-increasing in j'. Thus, $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j})$ is monotonically non-increasing in j', and hence for any $j' \leq j$, we have $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \infty$. Furthermore, to prove that $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) < \infty$ for all $j' > j$, it is sufficient to prove that $Q^{\uparrow}_{j+1,t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) < \infty$.

From OCP, the algorithm will add a new constraint $L_{j,t^*_j} \leq Q_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) \leq U_{j,t^*_j}$. We first prove that $U_{j,t^*_j} < \infty$. To see it, notice that if $t^*_j = H - 1$, then $U_{j,t^*_j} = U_{j,H-1} = R_{H-1}(x_{j,H-1}, a_{j,H-1}) < \infty$. On the other hand, if $t^*_j < H - 1$, then by definition $U_{j,t^*_j} = R_{t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) + Q^{\uparrow}_{j,t^*_j+1}(x_{j,t^*_j+1}, a_{j,t^*_j+1})$. From the definition of t^*_j, $Q^{\uparrow}_{j,t^*_j+1}(x_{j,t^*_j+1}, a_{j,t^*_j+1}) < \infty$, thus $U_{j,t^*_j} < \infty$. Consequently, $Q^{\uparrow}_{j+1,t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \overline{\theta}^{(j+1)}_{t^*_j,k} = \min\{\overline{\theta}^{(j)}_{t^*_j,k}, U_{j,t^*_j}\} \leq U_{j,t^*_j} < \infty$. Thus, $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) < \infty$ for all $j' > j$.

Thus, if we consider $Q^{\uparrow}_{j',t^*_j}(x_{j,t^*_j}, a_{j,t^*_j}) = \overline{\theta}^{(j')}_{t^*_j,k}$ as a function of j', then this function transits from infinity to finite values in episode j. In summary, $t^*_j \neq \text{NULL}$ implies that $\overline{\theta}^{(j')}_{t^*_j,k}$ transits from infinity to finite values in episode j. Since other $\overline{\theta}^{(j')}_{t,k}$'s might also transit from ∞ to finite values in episode j, thus $\mathbf{1}[t^*_j \neq \text{NULL}]$ is less than or equal to the number of $\overline{\theta}^{(j')}_{t,k}$'s transiting from ∞ to finite values in episode j. Note that from the monotonicity of $\overline{\theta}^{(j')}_{t,k}$, for each partition, this transition can occur at most once, and there are K partitions in total. Hence we have $\sum_{j=0}^{\infty} \mathbf{1}[t^*_j \neq \text{NULL}] \leq K$. **q.e.d.**

References

[1] Abbasi-Yadkori, Yasin, Csaba Szepesvári. 2011. Regret bounds for the adaptive control of linear quadratic systems. *Journal of Machine Learning Research - Proceedings Track* **19** 1–26.

[2] Auer, Peter, Ronald Ortner. 2006. Logarithmic online regret bounds for undiscounted reinforcement learning. *NIPS*. 49–56.

[3] Azar, Mohammad Gheshlaghi, Alessandro Lazaric, Emma Brunskill. 2013. Regret bounds for reinforcement learning with policy advice. *Machine Learning and Knowledge Discovery in Databases*. Springer, 97–112.

[4] Bartlett, Peter L., Ambuj Tewari. 2009. REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI2009)*. 35–42.

[5] Bertsekas, Dimitri P., John Tsitsiklis. 1996. *Neuro-Dynamic Programming*. Athena Scientific.

[6] Brafman, Ronen I., Moshe Tennenholtz. 2002. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research* **3** 213–231.

[7] Gordon, Geoffrey. 1995. Online fitted reinforcement learning. *Advances in Neural Information Processing Systems 8*. MIT Press, 1052–1058.

[8] Ibrahimi, Morteza, Adel Javanmard, Benjamin Van Roy. 2012. Efficient reinforcement learning for high dimensional linear quadratic systems. *NIPS*.

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

[10] Kakade, Sham. 2003. On the sample complexity of reinforcement learning. Ph.D. thesis, University College London.

[11] Karmarkar, Narendra. 1984. A new polynomial-time algorithm for linear programming. *Combinatorica* **4**(4) 373–396.

[12] Kearns, Michael J., Daphne Koller. 1999. Efficient reinforcement learning in factored MDPs. *IJCAI*. 740–747.

[13] Kearns, Michael J., Satinder P. Singh. 2002. Near-optimal reinforcement learning in polynomial time. *Machine Learning* **49**(2-3) 209–232.

[14] Lagoudakis, Michail G, Ronald Parr, Michael L Littman. 2002. Least-squares methods in reinforcement learning for control. *Methods and Applications of Artificial Intelligence*. Springer, 249–260.

[15] Lattimore, Tor, Marcus Hutter, Peter Sunehag. 2013. The sample-complexity of general reinforcement learning. *ICML*.

[16] Li, Lihong, Michael Littman. 2010. Reducing reinforcement learning to kwik online regression. *Annals of Mathematics and Artificial Intelligence* .

[17] Li, Lihong, Michael L Littman, Thomas J Walsh, Alexander L Strehl. 2011. Knows what it knows: a framework for self-aware learning. *Machine learning* **82**(3) 399–443.

[18] Neu, Gergely, Andras Antos, András György, Csaba Szepesvári. 2010. Online markov decision processes under bandit feedback. *Advances in Neural Information Processing Systems*. 1804–1812.

[19] Neu, Gergely, András György, Csaba Szepesvári. 2012. The adversarial stochastic shortest path problem with unknown transition probabilities. *International Conference on Artificial Intelligence and Statistics*. 805–813.

[20] Ortner, Ronald, Daniil Ryabko. 2012. Online regret bounds for undiscounted continuous reinforcement learning. *NIPS*.

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

[22] Pazis, Jason, Ronald Parr. 2013. Pac optimal exploration in continuous space markov decision processes. *AAAI*. Citeseer.

[23] Powell, Warren. 2007. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience.

[24] Powell, Warren, Ilya Ryzhov. 2011. *Optimal Learning*. John Wiley and Sons.

[25] Rummery, G. A., M. Niranjan. 1994. On-line Q-learning using connectionist systems. Tech. rep.

[26] Russo, Daniel, Benjamin Van Roy. 2014. Learning to optimize via posterior sampling. *Mathematics of Operations Research* **39**(4) 1221–1243.

[27] Ryzhov, Ilya O, Warren B Powell. 2010. Approximate dynamic programming with correlated bayesian beliefs. *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*. IEEE, 1360–1367.

[28] Singh, Satinder P., Tommi Jaakkola, Michael I. Jordan. 1994. Reinforcement learning with soft state aggregation. *NIPS*. 361–368.

[29] Strehl, Alexander L. 2007. *Probably approximately correct (PAC) exploration in reinforcement learning*. ProQuest.

[30] Strehl, Er L., Lihong Li, Eric Wiewiora, John Langford, Michael L. Littman. 2006. PAC model-free reinforcement learning. *Proceedings of the 23rd international conference on Machine learning*. 881–888.

[31] Sutton, Richard, Andrew Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

[32] Szepesvári, Csaba. 2010. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers.

[33] Tsitsiklis, John N., Benjamin Van Roy. 1996. Feature-based methods for large scale dynamic programming. *Machine Learning* **22**(1-3) 59–94.

[34] Van Roy, Benjamin. 2006. Performance loss bounds for approximate value iteration with state aggregation. *Math. Oper. Res.* **31**(2) 234–244.

[35] Van Roy, Benjamin, Zheng Wen. 2014. Generalization and exploration via randomized value functions. *arXiv preprint arXiv:1402.0635* .

[36] Wang, Hua O, Kazuo Tanaka, Michael F Griffin. 1996. An approach to fuzzy control of nonlinear systems: stability and design issues. *Fuzzy Systems, IEEE Transactions on* **4**(1) 14–23.

[37] Wen, Zheng, Benjamin Van Roy. 2013. Efficient exploration and value function generalization in deterministic systems. *Advances in Neural Information Processing Systems*. 3021–3029.

[38] Whitehead, Steven D. 2014. Complexity and cooperation in q-learning. *Proceedings of the Eighth International Workshop on Machine Learning*. 363–367.