
Reinforcement learning in the presence of rare events

Jordan Frank

JORDAN.FRANK@CS.MCGILL.CA

Department of Computer Science, McGill University, Montreal, Quebec, Canada

Shie Mannor

SHIE.MANNOR@MCGILL.CA

Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec, Canada

Doina Precup

DPRECUP@CS.MCGILL.CA

Department of Computer Science, McGill University, Montreal, Quebec, Canada

Abstract

We consider the task of reinforcement learning in an environment in which rare significant events occur independently of the actions selected by the controlling agent. If these events are sampled according to their natural probability of occurring, convergence of conventional reinforcement learning algorithms is likely to be slow, and the learning algorithms may exhibit high variance. In this work, we assume that we have access to a simulator, in which the rare event probabilities can be artificially altered. Then, importance sampling can be used to learn with this simulation data. We introduce algorithms for policy evaluation, using both tabular and function approximation representations of the value function. We prove that in both cases, the reinforcement learning algorithms converge. In the tabular case, we also analyze the bias and variance of our approach compared to TD-learning. We evaluate empirically the performance of the algorithm on random Markov Decision Processes, as well as on a large network planning task.

1. Introduction

We consider a practically important class of control tasks, in which rare (potentially catastrophic) events might take place. For example, in a computer network, links and nodes can fail, causing traffic to be undelivered and large penalties to be incurred. A robot exploring a rugged terrain may be caught by a sudden gust of wind which rolls it over. An investment agent may be faced with a market that is in tur-

moil due to a sudden unforeseen event. In such cases, the rare events occur *independently* of the actions of the agent, with some small probability. However, such rare events can have a disproportionate effect on the agent's utility. If such events are sampled on-line, as is the case in most reinforcement learning (RL) applications, they may not occur often enough to obtain an accurate estimate of the value function.

In this paper, we formalize this problem and propose solution algorithms. We assume that learning will be done in a simulation environment in which the probability of the rare event can be set to desired levels. In most safety-critical applications, training in a simulated environment is a common approach. In this case, we can sample rare events more often, and use importance sampling corrections similar to (Precup et al. 2000, 2001) to evaluate a given policy. However, importance sampling can cause high variance in the learning updates. We propose to use an adaptive algorithm in which the sampling rate for the rare event is adjusted in such a way as to minimize variance. For the case in which the value function is represented as a table, we show that the algorithm converges and provide a bias-variance analysis, based on (Mannor et al., 2007). For the case of linear function approximation, we prove convergence. We note that a bias-variance analysis for this case is not even available for TD-learning without importance sampling. We illustrate the performance of our approach on two domains: random Markov Decision Processes (MDPs), and a large network planning task. Our approach proves quite successful when compared to on-line TD-learning.

The literature on simulation of rare events is vast; see (Bucklew, 2004; Asmussen & Glynn, 2007) for comprehensive reviews. There are many Markov (or Markov-like) models that have been studied in the simulation community including queues, inventory control problems, call centers, communication systems, etc. The main objective of these works is to estimate the probability of a rare event by simulating the system under an alternative probability mea-

sure, and then use importance sampling to unbiased the results. The search for the optimal change of measure can be done in several ways, including the cross-entropy method (Rubinstein & Kroese, 2004) and stochastic approximation. Variance reduction has also been studied within the RL community. In particular, (Baxter & Bartlett, 2001) considered adaptive control-variates for policy gradient algorithms.

The explicit modeling of rare events in reinforcement learning-style algorithms was studied in (Bhatnagar et al., 2006). Their objective is to find an optimal control policy conditioned on the occurrence of a rare event. A model that closely resembles our approach is presented in (Ahamed et al., 2006). However, they assume that the model of the transition probabilities is known, and can be arbitrarily modified. We make a less restrictive assumption: the only parameter of the simulator that can be modified is the rate at which the rare events occur. Also, the bias-variance analysis and discussion of the function approximation case are novel.

The rest of the paper is organized as follows. In Section 2 we provide the essential background on RL and MDPs. In Section 3 we formally describe the rare events model used in this paper. We review RL algorithms that use importance sampling in Section 4. The learning algorithm we propose is described in Section 5. In Section 6 we present bias-variance results for learning in MDPs with rare events. Section 7 presents a learning algorithm with function approximation and a proof of convergence in this case. The empirical results of our approach are presented in Section 8. Finally, Section 9 presents conclusions and avenues for future work.

2. Background

We use the standard RL framework (Sutton & Barto, 1998) in which an agent interacts with its environment at discrete time steps $t = 0, 1, 2, \dots$. At time t , the agent finds itself in a state $s_t \in S$, chooses an available action $a_t \in A_{s_t}$, and then receives a numerical reward, $r_{t+1} \in \mathbb{R}$ and observes the next state s_{t+1} . We denote $A = \bigcup_{s \in S} A_s$. If the environment is modeled as an MDP, its dynamics are characterized by the stationary transition probability distribution:

$$p(s'|s, a) = \Pr\{s_{t+1} = s' | s_t = s, a_t = a\},$$

and a bounded, real-valued reward function $r(s, a, s')$ with $|r(s, a, s')| \leq R_{\max} < \infty, \forall s, s' \in S, a \in A$.

We are concerned with the problem of policy evaluation in discounted infinite horizon problems with discount factor $\gamma \in (0, 1)$. The agent chooses its actions according to a stationary policy $\pi(s, a) = \Pr\{a_t = a | s_t = s\}$. We are interested in computing the state-value function $V^\pi : S \rightarrow \mathbb{R}$ for

the given policy π . This value function is the solution to the well-known Bellman equations

$$V^\pi(s) = \sum_{a \in A} \pi(s, a) \sum_{s' \in S} p(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]. \quad (1)$$

In RL, this value function is often estimated on-line using the well-known TD-learning algorithm (Sutton, 1988). If the actions are chosen using the desired policy π , after observing transition (s, a, r, s') , the estimate of the value function, V , can be updated as:

$$V(s) \leftarrow V(s) + \alpha [r + \gamma V(s') - V(s)]. \quad (2)$$

In control tasks, the objective is to find the policy that maximizes $V^\pi(s)$ at all states s .

3. Rare Events

We are concerned with problems involving rare, significant events that occur as a result of environmental factors, and which are independent of the current action taken by the agent. We model this using a mixture of two separate transition probability distributions: $f(s'|s, a)$, which captures the environment dynamics during “normal” operating conditions, and $g(s'|s)$, which is the “rare event” transition distribution. We assume that at every state $s \in S$, there is a small probability, $\epsilon(s)$, that an unusual event might occur from this state. In this case, the transition to the next state is determined exclusively by g . If such an event does not occur, the next state is drawn according to f , and depends on both the current state and the agent’s action. Hence, the transition probability in the environment can be re-written as:

$$p(s'|s, a) = (1 - \epsilon(s))f(s'|s, a) + \epsilon(s)g(s'|s). \quad (3)$$

Without loss of generality, we will assume that the “normal” states (reachable by f) and the “rare event” states (reachable by g) are disjoint. Hence, the transition probability distribution can be re-written as:

$$p(s'|s, a) = \begin{cases} (1 - \epsilon(s))f(s'|s, a) & \text{if } s' \notin T, \\ \epsilon(s)g(s'|s) & \text{if } s' \in T, \end{cases}$$

where $T \subseteq S$ is the set of “rare event” states.

We are concerned with rare events that have a significant impact on the state-value function for a given policy. Therefore we define the rare events state set as follows.

Definition 3.1. *A subset of states $T \subseteq S$ is called a rare events state set if the following three properties hold:*

1. *For all $s \in S, a \in A, s' \in T, f(s'|s, a) = 0$ (i.e., T is not reachable from any state s using the agent’s actions).*

2. *There exists $s \in S, s' \in T$ such that $g(s'|s) > 0$ (i.e., T can be forced by the environment)*
3. *Let V_f^π denote the value function obtained by replacing p with f in (1). Then, for the given policy π ,*

$$\exists s \in S \text{ s.t. } |V_f^\pi(s) - V^\pi(s)| \gg 0.$$

The last condition means that the states in the rare event state set must (collectively) have a large impact on the state-value function. We define *rare events* to be transitions into the rare event state set. For convenience, we will refer to the states that are not in the rare event state set, $S \setminus T$, as the normal states.

We note that we use the term ‘‘rare event’’ loosely from the point of view of the simulation community (Bucklew, 2004), because our definition is not based solely on the probability of the event. We deviate from the typical definition due to the fact that there may be events that occur infrequently but do not have a noticeable effect on our value function estimates, and we are not concerned with these events.

4. Importance Sampling for Reinforcement Learning

The TD update (2) is based on the idea that the right-hand side of the Bellman equations (1) can be approximated using samples of the next transition, $r(s, a, s') + \gamma V(s')$, where $a \sim \pi(s, \cdot)$ and $s' \sim p(\cdot|s, a)$. However, in an environment with rare events, if $\varepsilon(s)$ is very small, a very large number of samples will be needed in order for the rare events to be averaged properly in the value function estimates. Instead, we investigate a sampling distribution which allows these events to be sampled preferentially, and then we use importance sampling corrections to account for this in the TD updates.

Importance sampling is a variance-reduction technique commonly used in statistics, as well as in the simulation community (Bucklew, 2004). The main idea is that instead of obtaining samples from the true distribution p , they will be drawn from a different distribution q , called the *proposal distribution*, in which events of interest occur more frequently. If q is devised well, then using these samples will reduce the variance of the estimator. Precup, Sutton & Singh (2000) extended this approach to TD-learning. They studied the case in which a target policy π is evaluated based on data generated by a different behavior policy. In this case, they showed that a TD-learning algorithm can still be used, in which the TD targets are adjusted by using the appropriate importance sampling weights: $w(s, a, s')(r(s, a, s') + \gamma V(s'))$, where:

$$w(s, a, s') = \frac{p(s'|s, a)}{q(s'|s, a)}.$$

In their case, the change of measure is induced by the behavior policy, and the importance sampling weights are the likelihood ratios of the probabilities of action a under the two policies.

Ahamed, Borkar & Juneja (2004) use the same idea but with the goal of changing the next-state probabilities in a discrete-time finite-state Markov chain with positive costs. They assume that the transition probabilities are known and can be modified at will, and propose an adaptive importance sampling algorithm (ASA) which finds an alternative set of transition probabilities in order to minimize the variance of the value function estimator. They provide a convergence proof (assuming a tabular representation of the value function), a discussion of convergence rates, and simulation results.

5. Learning in the presence of rare events

The ASA algorithm assumes that we have full knowledge of the transition model, and can completely control the transition probabilities, so all the transition probabilities can be tilted towards the zero-variance importance sampling distribution. In this paper, we relax this assumption because it is difficult to achieve in practical applications. We assume that the true rare event probability ε is known (e.g., as the mean of a Poisson process that generates failures in a network, or the weight of the tail of a distribution in which rare events occur). We assume that the system dynamics, f and g are unknown and cannot be modified, but that the probability with which rare events are generated can be changed as the simulation proceeds. In general, with only this parameter at our disposal, we cannot achieve the zero-variance importance sampling distribution; however, we can tilt the transition probability distribution p towards the zero-variance distribution, and therefore reduce the variance of our estimates.

We define $\hat{\varepsilon} : S \rightarrow [0, 1]$ to be the probability of a rare event occurring from every state during the simulation. Hence, the next states will be sampled from a proposal distribution given by:

$$q(s'|s, a) = (1 - \hat{\varepsilon}(s))f(s'|s, a) + \hat{\varepsilon}(s)g(s'|s), \quad (4)$$

where f and g remain unchanged. By considering that the state space S is separated into disjoint normal and rare event subsets of states, we note that the importance sampling corrections $w(s, a, s')$ can be computed by:

$$w(s, a, s') = \begin{cases} \varepsilon(s)/\hat{\varepsilon}(s) & \text{if } s \in T, \\ (1 - \varepsilon(s))/(1 - \hat{\varepsilon}(s)) & \text{if } s \notin T. \end{cases} \quad (5)$$

Following a similar argument as in the development of the ASA algorithm, we can determine the following optimal

form for the rare event sampling distribution:

$$\varepsilon^*(s) = \varepsilon(s) \frac{\sum_{s' \in T} g(s'|s) [(\sum_{a \in A} \pi(s, a) r(s, a, s')) + \gamma V^\pi(s')]}{V^\pi(s)}, \quad (6)$$

Fortunately, the values $\varepsilon^*(s)$ can be estimated on-line using samples.

Algorithm 1 is our proposed approach for learning in the presence of rare events. We call this algorithm rare events adaptive stochastic approximation (REASA). It is based on the observation that we can rewrite $\varepsilon^*(s)$ as follows:

$$\varepsilon^*(s) = \frac{T^*(s)}{T^*(s) + U^*(s)}, \text{ where}$$

$$T^*(s) = \varepsilon(s) \sum_{s' \in T} g(s'|s) [(\sum_{a \in A} \pi(s, a) r(s, a, s')) + \gamma V^\pi(s')]$$

is the contribution to the value of s by the rare event state set T , and

$$U^*(s) = (1 - \varepsilon(s)) \sum_{a \in A} \pi(s, a) \sum_{s' \notin T} f(s'|s, a) [r(s, a, s') + \gamma V^\pi(s')]$$

is the contribution to $V^\pi(s)$ from the normal states.

In the algorithm, $T(s)$ is an unbiased estimator of $T^*(s)$ and $U(s)$ is an unbiased estimator of $U^*(s)$. It follows that as $t \rightarrow \infty$, from Equation (6),

$$\hat{\varepsilon}(s) = \frac{T(s)}{T(s) + U(s)} \rightarrow \varepsilon^*(s),$$

for every state $s \in S$. Since we use importance sampling to calculate $\hat{V}^\pi(s)$, we also have that as $t \rightarrow \infty$, $\hat{V}^\pi(s) \rightarrow V^\pi(s)$ from standard stochastic approximation arguments under some mild assumptions on the MDP structure. We summarize the result in the following proposition.

Proposition 1. *Using Algorithm 1 and assuming that the MDP is unichain for $\varepsilon = \delta^1$ we have that:*

$$\hat{V}^\pi(s) \rightarrow V^\pi(s) \text{ almost surely.}$$

Moreover, $\forall s$ s.t. $\varepsilon^*(s) \in (\delta, 1 - \delta)$ we have that $\hat{\varepsilon}(s) \rightarrow \varepsilon^*(s)$ almost surely.

We note that we guarantee that we have enough persistent exploration by requiring that $\hat{\varepsilon}(s)$ is bounded from below by δ and from above by $1 - \delta$ (step 5h in Algorithm 1).

Although the treatment above is assuming positive rewards (for ease of notation), our algorithm is actually formulated for the general case in which rewards can be both positive and negative, which is an extension of the ASA algorithm.

¹The unichain assumption is needed to invoke the stochastic approximation argument; see (Bertsekas & Tsitsiklis, 1996). Also note that if the MDP is unichain for one value of $\varepsilon \in (\delta, 1 - \delta)$ it is unichain for all values.

Algorithm 1 Rare-event Adaptive Importance Sampling

Input: Rare event set $T \subset S$, true rare-event probabilities $\varepsilon(s)$, and parameter $\delta > 0$, used to keep the sampling distribution non-zero everywhere.

1. Initialize \hat{V}^π arbitrarily.
2. Initialize the rare-event sampling distribution: $\hat{\varepsilon}(s) \leftarrow 1/2, \forall s$.
3. Initialize the variables $T(s)$, $U(s)$ (which measure the contribution of T and $S \setminus T$ to V^π) to 0.
4. Initialize eligibility traces: $e(s) = 0, \forall s$.
5. Select the initial state s_0 .
6. Repeat for $t = 0, 1, \dots$:

- (a) Update the eligibility trace of the current state

$$e(s_t) = e(s_t) + 1.$$

- (b) Select an action $a_t \sim \pi(s_t, \cdot)$.
- (c) Select whether a rare event happens, according to $\hat{\varepsilon}(s_t)$, and sample s_{t+1} from f or g accordingly. Observe the reward r_{t+1} .
- (d) Compute the importance sampling weight w_t according to Equation (5).
- (e) Compute the importance-sampling TD-error:

$$\Delta_t = w_t(r_{t+1} + \gamma \hat{V}^\pi(s_{t+1})) - \hat{V}^\pi(s_t).$$

- (f) Update the value estimates:

$$\hat{V}^\pi(s) \leftarrow \hat{V}^\pi(s) + \alpha e(s) \Delta_t, \forall s,$$

where $\alpha \in [0, 1]$ is a learning rate.

- (g) If $s_{t+1} \in T$, then:

$$T(s_t) \leftarrow (1 - \alpha_T) T(s_t) + \alpha_T \varepsilon(s_t) (r_{t+1} + \hat{V}^\pi(s_{t+1})),$$

else

$$U(s_t) \leftarrow (1 - \alpha_U) U(s_t) + \alpha_U (1 - \varepsilon(s_t)) (r_{t+1} + \hat{V}^\pi(s_{t+1})),$$

where $\alpha_T, \alpha_U \in (0, 1)$ are learning rates. In the experiments, we use the inverse of the number of times a transition from s_t has been observed to T and $S \setminus T$ respectively.

- (h) Update the rare event probabilities:

$$\hat{\varepsilon}(s_t) \leftarrow \min \left(\max \left(\delta, \frac{|T(s_t)|}{|T(s_t)| + |U(s_t)|} \right), 1 - \delta \right).$$

- (i) Update eligibility traces:

$$e(s) \leftarrow \gamma \lambda w_t e(s), \forall s.$$

6. Bias and Variance of Reinforcement Learning with Rare Events

For simplicity, let us assume that $\varepsilon(s) = \varepsilon$ for all states $s \in \mathcal{S}$ (all the analysis can be done without this assumption, but becomes more tedious). Let R^π denote the vector of immediate rewards for every state, with entries:

$$R_s^\pi = \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \pi(s, a) p(s' | s, a) r(s, a, s'),$$

and P^π be an $|\mathcal{S}| \times |\mathcal{S}|$ transition matrix under π , with entries:

$$P_{ss'}^\pi = \sum_a \pi(s, a) p(s' | s, a).$$

From (3), we can re-write P^π as:

$$P^\pi = (1 - \varepsilon)F^\pi + \varepsilon G,$$

where F^π is the transition matrix corresponding to staying in the normal states, and G is the matrix corresponding to transiting into the rare event states. Note that according to our assumptions, G does not depend on π . Similarly, the reward vector can be decomposed into two components, R_F^π and R_G^π . We use two sequences, $\{X_k\}_{k=1}^\infty$ and $\{Y_k\}_{k=1}^\infty$, of geometrically distributed random variables, with means $(1 - \varepsilon)^{-1}$ and ε^{-1} respectively, to represent the amount of time between transitions from the normal states and the rare event states respectively. We also assume that the initial state is a normal state. Hence, the simulation starts in some normal state and stays in the set of normal states for X_1 time steps, at which point it transitions to a state in the rare event set, where it stays for Y_1 time steps, then transitions back to the normal set for X_2 time steps, etc. We make two further simplifications. First, we assume that after each excursion into the rare event state set, the system ‘‘jumps back’’ to the normal state in which it was before entering; that is, $(F^\pi)^i G^j (F^\pi)^k \approx (F^\pi)^{i+k}$. Second, we assume that the rewards for transitioning to states in the normal set are similar regardless of the origin, that is that $GR_F^\pi \approx F^\pi R_F^\pi$. The analysis can be done without these assumptions, but it becomes more tedious. These assumptions are reasonable because in general the rare events model failures in the system, such as a failed link in a network, and when the failure is no longer present, the system resumes from the state prior to the failure. We define $\tau(k) = \sum_{i=1}^{k-1} X_i$ and $\upsilon(k) = \sum_{i=1}^k Y_i$. The value function estimate, V^π , can be re-written as:

$$V^\pi \approx \mathbb{E} \left[\sum_{k=1}^{\infty} \gamma^{\tau(k)+\upsilon(k)} (F^\pi)^{\tau(k)-1} \cdot \left(\sum_{i=0}^{X_k-1} \gamma^i (F^\pi)^i R_F^\pi + \gamma^{X_k} (F^\pi)^{X_k-1} \sum_{i=0}^{Y_k-1} \gamma^i G^i R_G^\pi \right) \right].$$

When the value function is estimated from data using TD-learning, we can analyze the bias and variance of this es-

timate by considering that all the model component estimates are affected by noise components (Mannor et al., 2007). The estimate of the value function, \hat{V}^π can be broken up into two components; the first component ignores rare events, and the second takes rare events into account. The first component is:

$$\mathbb{E}[\hat{V}_F^\pi] = \sum_{k=1}^{\infty} \mathbb{E} \left[\gamma^{\tau(k)+\upsilon(k)} (F^\pi + \tilde{F}^\pi)^{\tau(k)-1} \cdot \sum_{i=0}^{\infty} (1 - \varepsilon)^i \varepsilon \gamma^i (F^\pi + \tilde{F}^\pi)^i (R_F^\pi + \tilde{R}_F^\pi) \right]$$

where \tilde{R}_F^π , \tilde{F}^π represent the noise estimates in the normal part of the model. The bias and variance of this estimate can be derived directly as in (Mannor et al., 2007), noting that $\tau(k)$ and $\upsilon(k)$ are sums of independent geometrically distributed variables, and are therefore distributed according to a negative binomial distribution.

The second component is:

$$\mathbb{E}[\hat{V}_G^\pi] = \sum_{k=1}^{\infty} \mathbb{E} \left[\gamma^{\tau(k+1)+\upsilon(k)} (F^\pi + \tilde{F}^\pi)^{\tau(k+1)-1} \cdot \sum_{i=0}^{\infty} (1 - \varepsilon)^i \varepsilon^i \gamma^i (G + \tilde{G})^i (R_G^\pi + \tilde{R}_G^\pi) \right]$$

Note that the noise components \tilde{G} , \tilde{F}^π , \tilde{R}_G^π depend on the number of transitions observed in the environment. If we observe N transitions, then the expected number of transitions observed in the normal state set is $(1 - \varepsilon)N$ and the expected number of transitions observed in the rare event state set is εN . Hence, we assume that the noise component \tilde{F}^π is negligible compared to \tilde{G} , and \tilde{R}_G^π . Hence, to establish bias-variance estimates for \hat{V}_G^π , we need to look at $\mathbb{E}[(G + \tilde{G})(R_G^\pi + \tilde{R}_G^\pi)]$. Similarly to (Mannor et al., 2007), we assume that $\mathbb{E}[\tilde{G}] = 0$ and $\mathbb{E}[\tilde{R}_G^\pi] = 0$. Hence, the remaining term which will determine the bias and variance is $\mathbb{E}[\tilde{G}\tilde{R}_G^\pi]$, which captures the correlations between the transition and model estimates, due to the fact that they are estimated from the same samples. This expectation can be derived directly from the formulas in (Mannor et al., 2007).

We would like to point out that we could also have applied the analysis of (Mannor et al., 2007) directly to P^π . However, this would lead to very loose bounds, because their results depend on the inverse of the minimum number of samples obtained for any transition, and we expect that there will be very few transitions into the rare event set. In our analysis, only the second term depends on numbers of transitions into the rare events states, so we can focus our analysis on the effect of the rare events on the bias and variance in our estimates.

Also, note that the purpose of the algorithm is to sample rare events proportionately to their contribution to the value function for all states. Hence, intuitively, it will reduce bias and variance in the second component by oversampling the rare events, and thus decreasing the noise components \tilde{G} and \tilde{R}_G^π . Given the same amount of data, the errors in \tilde{F} and \tilde{R}_F^π , but not by much.

7. Learning with Rare Events and Function Approximation

If the state space is very large or continuous, function approximation must be used to estimate the value function. Here, we are concerned with the case of linear function approximation, in which the value of a state is estimated as:

$$V^\pi(s) \approx \theta\phi(s), \quad (7)$$

where θ is a parameter vector that needs to be estimated and $\phi(s)$ is a set of features corresponding to state s . In this case, the eligibility traces are also represented as a vector \mathbf{e} of the same size as θ . We now extend the REASA algorithm to deal with this case. First, note that in this case, we may not be able to have a state-dependent probability of obtaining a rare event state, because specifying this on a state-by-state basis would be too expensive. Hence, we will assume for the moment, without loss of generality, that the true rare event probability ε is constant over the entire state space. We discuss possible extension to this in Section 9. The algorithm will estimate a parameter $\hat{\theta}$ by taking the view that, at a high level, the agent switches between the normal states $S \setminus T$ and the rare-event states T . These are now treated as two states in a high-level MDP, and $\hat{\theta}$ is estimated like in REASA, on this 2-state system.

Algorithm 2 presents the approach, which adapts the algorithm of Precup et al. (2001). Unlike in the tabular case, here importance sampling corrections have to be made to account for the difference in the distribution of observed features, as well as for the difference in the TD target. As explained in Precup et al. (2001), these corrections, which are collected in the trajectory weight c , can result in high variance. However, since we assume that the sets of normal and rare event states are disjoint, we can assume, without loss of generality, that they are represented by disjoint features as well. In this case, step 7i of Algorithm 2 can be eliminated, and variance will be greatly improved.

Proposition 2. *Under standard stochastic approximation conditions, Algorithm 2 converges in the limit, with probability 1, to the same estimates as the on-policy TD-learning algorithm.*

Algorithm 2 Rare-event Adaptive Importance Sampling with Function Approximation

Input: Rare event set $T \subset S$, true rare-event probability ε , and parameter $\delta > 0$, used to keep the sampling distribution non-zero everywhere.

1. Initialize parameter vector θ arbitrarily.
2. Initialize the rare-event sampling parameter: $\hat{\varepsilon} \leftarrow 1/2$.
3. Initialize $\hat{T} \leftarrow 0, \hat{U} \leftarrow 0$.
4. Initialize eligibility vector: $\mathbf{e} \leftarrow 0$.
5. Initialize the total importance sampling trajectory weight: $c \leftarrow 1$.
6. Select the initial state s_0 .
7. Repeat for $t = 0, 1, \dots$:

- (a) Update the eligibility trace of the current state

$$\mathbf{e} = \mathbf{e} + c\phi(s_t).$$

- (b) Select an action $a_t \sim \pi(s_t, \cdot)$.
- (c) Select whether a rare event happens, according to $\hat{\varepsilon}$, and sample s_{t+1} from f or g accordingly. Observe the reward r_{t+1} .
- (d) Compute the importance sampling weight w_t according to (5).
- (e) Compute the importance-sampling TD-error:

$$\Delta_t = w_t(r_{t+1} + \gamma\hat{V}^\pi(s_{t+1})) - \hat{V}^\pi(s_t),$$

where \hat{V}^π is computed according to (7).

- (f) Update the parameter vector: $\theta \leftarrow \theta + \alpha\mathbf{e}\Delta_t$, where $\alpha \in [0, 1]$ is a learning rate.
- (g) If $s_{t+1} \in T$, then:

$$\hat{T} \leftarrow ((1 - \alpha_T)\hat{T} + \alpha_T\varepsilon(r_{t+1} + \gamma\hat{V}^\pi(s_{t+1}))),$$

else

$$\hat{U} \leftarrow (1 - \alpha_U)\hat{U} + \alpha_U(1 - \varepsilon)(r_{t+1} + \gamma\hat{V}^\pi(s_{t+1})).$$

- (h) Update the rare event probabilities:

$$\hat{\varepsilon} \leftarrow \min\left(\max\left(\delta, \frac{|\hat{T}|}{|\hat{T}| + |\hat{U}|}\right), 1 - \delta\right).$$

- (i) Update the trajectory weight: $c \leftarrow cw_t$.
- (j) Update eligibility traces:

$$\mathbf{e} \leftarrow \gamma\lambda w_t \mathbf{e}.$$

8. Experimental Results

8.1. Random MDPs

We first compare the performance of REASA to on-line TD(λ) and to ASA on a testbed of randomly generated

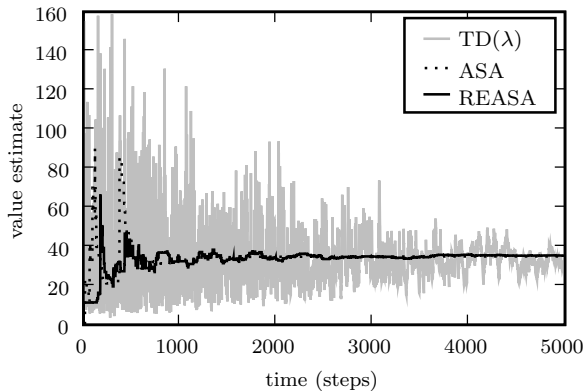


Figure 1. Value function estimate for state 0.

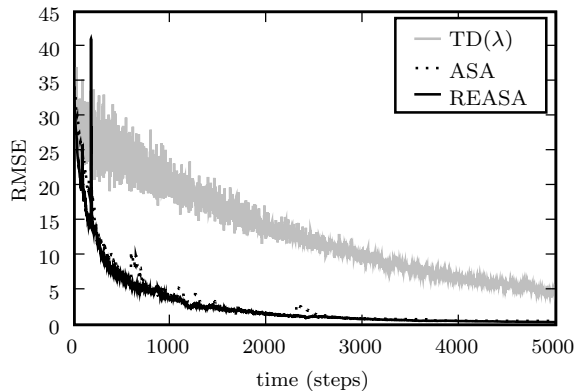


Figure 2. Root MSE for value function estimate for state 0.

Markov chains. Each environment contains 10 regular states and one rare event state. Each regular state can transition to seven other regular states (chosen randomly) with probabilities drawn from a uniform distribution, and to the rare event state with probability $\epsilon = 0.001$. The rewards for transitioning between the regular states and from the rare event state to the regular states are drawn from a normal distribution with mean 1.0 and standard deviation 0.5, with negative values being discarded (so that we can run ASA). The rewards for transitioning to the rare event state are drawn from a normal distribution with mean $10/\epsilon$ and standard deviation $1/\epsilon$. The initial state is state 0, and the discount factor is $\gamma = 0.7$.

In the following results, a step is considered to be one transition for both ASA and REASA, but for $\text{TD}(\lambda)$, a “step” actually consists of 2300 real time steps. We chose this number of steps so that the probability of observing at least one rare event transition in each episode is approximately 0.9. Therefore, we put $\text{TD}(\lambda)$ at a significant advantage in terms of the number of samples that it is provided. In Figure 1 we plot the estimate for the value function at the initial state over time, averaged across 70 independent runs. We use a value of $\lambda = 0.7$ and the learning rates are on decreasing schedules that have been tuned separately for each of the algorithms. Figure 2 shows the root mean squared error for the value function estimate at the initial state, again averaged across 70 independent runs.

The learning and error curves for REASA and ASA are nearly indistinguishable, and both outperform $\text{TD}(\lambda)$. We note that in the case of ASA, the original transition probability distribution is needed, and the algorithm has full control over the transition probabilities that are used in the simulation (an unlikely case in many practical applications). We observe that despite the fact that REASA can only know and control the rare event probability, it performs nearly as well as ASA.

8.2. Policy Evaluation for Network Planning

In order to demonstrate REASA in a practical setting with a large state space, we use a network planning task in which a reinforcement learning agent has to build and maintain a telecommunications network linking ten North American cities. Each pair of cities has a certain traffic demand, ranging from 3GBs² to 60GBs initially, and this demand grows stochastically at a rate of approximately 3% per year. The goal is to place links between the cities in order to deliver this data. Links consist of bundles of fiber optic cables, and each fiber can carry a specific unit of bandwidth. Building links between the cities incurs a large one-time cost of \$500k/mile. Once a link has been built, the capacity of the link can be increased by activating fibers, in units of 25GBs; this incurs a cost of \$30k/mile. The revenue from traffic is generated daily: traffic delivered generates a reward of \$1k/GBs/mile, and undelivered traffic is penalized at a rate of \$200k/GBs/mile every hour.

Link failures occur with a small probability, completely severing a link for a short period of time. Without considering link failures, a minimum spanning tree (MST) could be built, with enough activated fibers to carry the traffic. However, in such a network, any link failure would disconnect the network, which would lead to undelivered traffic and a high penalty. Hence, link failures in a network that lacks robustness are rare events according to our definition. On each day, each link goes down with probability $1/1460$, or approximately once every four years. When a link fails, it remains down for a random amount of time that is normally distributed with a mean of 12 hours and standard deviation of 2 hours. In a tree network with 9 links, this is equivalent to seeing at least one link fail with probability of approximately 0.00896 each day during the 10 year simulation period; this is our rare event probability.

²We use GBs to represent an average sustained traffic rate of 1 gigabyte per second; because the time interval under consideration is always roughly the same, we also use it as a unit of traffic, with an abuse of notation.

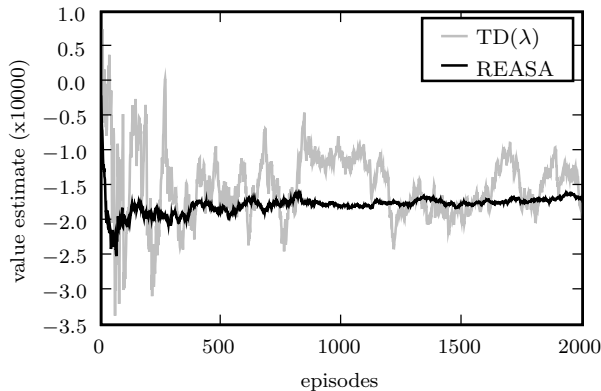


Figure 3. Value estimate for tree network.

We implemented a network planning agent with a simple heuristic policy, which first builds a tree network and then monitors the links, adding capacity when the utilization of a link reaches 90%. We use REASA and $\text{TD}(\lambda)$ to estimate the value of this policy. We represent the network state as a vector of binary features, and use linear function approximation to represent the value function. In order to cope with the high dimensionality, we use a fairly coarse state representation, consisting of: indicator variables regarding whether each of the possible links have been built; indicator variables for each link, which are true if the link is currently failing; and the percentage utilization of each link, partitioned into 4 bins: $[0]$, $(0, 0.6]$, $(0.6, 0.9]$, and $(0.9, 1.0]$. For our 10 node network, this corresponds to 270 binary features plus an additional bias feature.

We use a discount factor of 0.95 and we set $\lambda = 1.0$. We use a decaying schedule for the learning rate parameter α , starting with a value of $\alpha_0 = 2^{-15}$ for $T = 100$ episodes, then using $\alpha_0/2$ for $2T$ episodes, $\alpha_0/4$ for $4T$ episodes, etc. We note that α_0 is extremely small due to the fact that the rewards often have large magnitude and can vary between -10^7 and 10^5 . In the following results, an episode consists of a simulated 10-year time span.

In Figure 3, we show the value estimate for the initial tree network state. We see that REASA converges quickly, while the $\text{TD}(\lambda)$ estimates have high variance and converge quite slowly. On longer runs, the $\text{TD}(\lambda)$ estimates do converge to the same value as REASA. REASA estimates the optimal failure probability to be 0.155, which in a tree network with 9 links corresponds to each link going down approximately every 54 days; this is quite far from the original failure probability of once every 1460 days.

The rate of convergence is crucial for applications such as the network task. Here, each episode corresponds to a simulated 10 year period, and these simulations are computationally expensive to run, because on each day, a routing algorithm has to be run to determine the reward. Hence, the gains obtained by REASA are significant.

9. Conclusions and Future Work

We presented an approach for reinforcement learning in environments with rare events, aimed at reducing the variance of RL algorithms. Our algorithm modifies the sampling probability of the rare events, and makes minimal assumptions on the simulator available to the agent. The empirical results demonstrate the viability of our approach for solving large-scale problems. Future work will include measuring empirically the bias and variance of the algorithm. We would also like to lift the assumption that the rare event probability is constant for the function approximation case. Note that Algorithm 2 can be easily adapted to compute \hat{T} and \hat{U} as a function of the features available. Hence, if a representation of $\epsilon(s)$ as a function of the available features ϕ is given, we could estimate $\hat{\epsilon}$ as a function of features as well. It is possible also to learn the true rare event probabilities ϵ from data, but we anticipate that in practice this may be difficult.

Acknowledgments

The authors gratefully acknowledge the support of NSERC and the Canada Research Chairs program.

References

- Ahamed, T. P. I., Borkar, V. S., & Juneja, S. (2006). Adaptive importance sampling technique for Markov chains using stochastic approximation. *Oper. Res.*, 54, 489–504.
- Asmussen, S. & Glynn, P. (2007). *Stochastic Simulation: Algorithms and Analysis*. Springer.
- Baxter, J. & Bartlett, P. (2001). Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15, 319–350.
- Bertsekas, D. & Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*. Athena Scientific.
- Bhatnagar, S., Borkar, V. S., & Akarapu, M. (2006). A simulation-based algorithm for ergodic control of Markov chains conditioned on rare events. *Journal of Machine Learning Research*, 7, 1937–1962.
- Bucklew, J. (2004). *Introduction to Rare Event Simulation*. Springer.
- Mannor, S., Simester, D., Sun, P., & Tsitsiklis, J. (2007). Bias and variance approximation in value function estimates. *Management Science*, 53, 308.
- Precup, D., Sutton, R., & Dasgupta, S. (2001). Off-policy temporal-difference learning with function approximation. *Proc. 18th International Conf. on Machine Learning*, 417–424.
- Precup, D., Sutton, R., & Singh, S. (2000). Eligibility traces for off-policy policy evaluation. *Proc. 17th International Conf. on Machine Learning*, 759–766.
- Rubinstein, R. & Kroese, D. (2004). *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. Springer-Verlag.
- Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3, 9–44.
- Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning*. The MIT Press.