Reducing Sample Complexity in Reinforcement Learning by Transferring Transition and Reward Probabilities

Kouta Oguni1, Kazuyuki Narisawa1 and Ayumi Shinohara1

1Graduate School of Information Sciences, Tohoku University, Aramaki aza Aoba 6-3-09, Aoba-ku, Sendai, Miyagi 980-8579, Japan

Keywords: Reinforcement Learning, Transfer Learning, Sample Complexity, PAC-MDP.

Abstract: Most existing reinforcement learning algorithms require many trials until they obtain optimal policies. In this study, we apply transfer learning to reinforcement learning to realize greater efficiency. We propose a new algorithm called TR-MAX, based on the R-MAX algorithm. TR-MAX transfers the transition and reward probabilities from a source task to a target task as prior knowledge. We theoretically analyze the sample complexity of TR-MAX. Moreover, we show that TR-MAX performs much better in practice than R-MAX in maze tasks.

1 INTRODUCTION

Reinforcement learning is one of the useful methods for providing control rules for machines, where an agent learns its own policy for an unknown environment autonomously. Programmers do not have to provide explicit rules for machines if the learning is powerful enough. In reinforcement learning, learners consume many trials until they obtain optimal rules, however it is difficult in real world learning. Recent studies (Tan, 1993; Miyazaki et al., 1997; Kretchmar., 2002) have attempted to reduce the number of trials for efficient learning.

Some researchers have applied transfer learning to reinforcement learning (Konidaris and Barto, 2006; Taylor and Stone, 2009; Taylor et al., 2007). Transfer learning applies the knowledge obtained in some source tasks to a target task to solve it efficiently. Some studies (Konidaris and Barto, 2006; Taylor et al., 2007) experimentally showed that applying transfer learning to reinforcement learning is effective in reducing the number of trials.

However, the theoretical aspects of transfer learning for reinforcement learning are not yet fully known. In (Mann and Choe, 2012), the authors defined an α-weak admissible heuristic and showed theoretically that transferring some action values yields efficient reinforcement learning.

In this paper, we outline our approach to transfer learning for reinforcement learning: we try to transfer the transition probability and reward probability as knowledge, because they should be useful for efficient learning.

On the basis of the theoretical framework in (Kakade, 2003; Strehl and Littman, 2005) we propose the efficient learning algorithm TR-MAX that is an extension of R-MAX (Brafman and Tennenholtz, 2003). From a theoretical viewpoint, we show the sample complexity of TR-MAX, which is smaller than that of R-MAX, and it implies that TR-MAX is PAC-MDP (Kakade, 2003). From a practical viewpoint, we compare TR-MAX with R-MAX in maze problems and verify that our TR-MAX algorithm is indeed efficient.

2 PRELIMINARIES

The reinforcement learning framework assumes that its task satisfies the Markov property. This task is called a Markov decision process (MDP) (Sutton and Barto, 1998).

Definition 1. A finite Markov Decision Process is a five tuple \( \langle S, A, T, R, \gamma \rangle \): \( S \) is a finite set called the state space, and \( A \) is a finite set called the action space, \( T : S \times A \times S \to [0, 1] \) is the transition probability, and \( R : S \times A \to \mathbb{R} \) is the reward function. \( 0 \leq \gamma < 1 \) is the discount factor.

According to the related work (Strehl et al., 2009; Mann and Choe, 2012), we assume that rewards take a value in the interval \([0, 1]\), so that we obtain \( R : S \times
A → [0, 1]. M denotes a finite MDP in the sequel.

**Definition 2.** For any policy in M, let

$$V_M^\pi(s) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r_{t+1+i} | s_t = s \right]$$

denote the policy (or state) value function from state $s_t = s$ at timestep $t$, and let

$$Q_M^\pi(s, a) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r_{t+1+i} | s_t = s, a_t = a \right]$$

denote the action value function from state $s_t = s$ and action $a_t = a$. Note that $s_t$ denotes the state at timestep $t$, and $a_t$ denotes the action at timestep $t$.

The policy (or state) value function indicates how good it is to be in a particular state for the agent. Moreover, the action value function indicates how good it is to perform the action in that state for the agent.

A policy $\pi$ is called optimal if it satisfies $V_M^\pi(s) \geq V_M^\pi(s)$ for any policy $\pi$ and state $s \in S$.

**Definition 3.** Let $\{s_1, a_1, r_1, s_2, a_2, r_2, \ldots\}$ be a random path generated by executing an algorithm $\mathcal{A}$ in an MDP $M$. For any fixed $\varepsilon > 0$, the sample complexity of $\mathcal{A}$ is the number of timesteps $t$ such that the policy $\mathcal{A}_t$ at time $t$ satisfies $V_M^\pi(s_t) < V_M^\pi(s_t) - \varepsilon$. The algorithm $\mathcal{A}$ is called Probably Approximately Correct in Markov Decision Processes (PAC-MDP) if the sample complexity of $\mathcal{A}$ is bounded by some polynomial in $|S|$, $|A|$, $\frac{1}{\varepsilon}$, $\frac{1}{\delta}$, $\frac{1}{1-\gamma}$ with probability at least $1 - \delta$, for any $0 < \varepsilon < 1$ and $0 < \delta < 1$.

If a reinforcement learning algorithm is PAC-MDP, the algorithm can efficiently learn in any MDP.

**Definition 4.** Let $\{s_1, a_1, r_1, s_2, a_2, r_2, \ldots\}$ be a random path generated by executing an algorithm $\mathcal{A}$ in an MDP $M$. For any fixed $\varepsilon > 0$, the sample complexity of $\mathcal{A}$ is the number of timesteps $t$ such that the policy $\mathcal{A}_t$ at time $t$ satisfies $V_M^\pi(s_t) < V_M^\pi(s_t) - \varepsilon$. The algorithm $\mathcal{A}$ is called Probably Approximately Correct in Markov Decision Processes (PAC-MDP) if the sample complexity of $\mathcal{A}$ is bounded by some polynomial in $|S|$, $|A|$, $\frac{1}{\varepsilon}$, $\frac{1}{\delta}$, $\frac{1}{1-\gamma}$ with probability at least $1 - \delta$, for any $0 < \varepsilon < 1$ and $0 < \delta < 1$.

If a reinforcement learning algorithm is PAC-MDP, the algorithm can efficiently learn in any MDP.

Brafman et al. (Brafman and Tennenholtz, 2003) proposed an efficient model-based algorithm called R-MAX $^1$. Model-based algorithms maintain not only the rewards but also the number of selections $\text{select}(s, a)$ and the number of transitions $\text{trans}(s, a, s')$ for any state $s, s' \in S$ and action $a \in A$, and calculate the empirical transition probability $\text{empirical transition probability}$ and the empirical expectation of the reward distribution $\hat{R}(s, a)$. In R-MAX, the action value is updated only for the state-action pairs $(s, a)$ satisfying $\text{select}(s, a) \geq m$. The sample complexity of R-MAX is analyzed by (Strehl et al., 2009) as follows.

**Theorem 1.** Let $0 < \varepsilon < \frac{1}{\sqrt{m}}$ and $0 < \delta < 1$ be any real numbers, and assume that the two inputs $m$ and $\varepsilon$ of R-MAX in M satisfy $m = O\left(\frac{|S| + \ln|S|}{\varepsilon^2}\right)$, where $X = \max_{s} |\pi'(1 - \gamma)|$ and $\frac{1}{\delta} = O\left(\frac{1}{\varepsilon}\right)$. Let $\mathcal{A}$ denote the policy of R-MAX at timestep $t$ and $s_t$ denote the state at timestep $t$. Then, $V_M^\pi(s_t) \geq V_M^\pi(s_t) - \varepsilon$ is true for all but

$$O\left(\frac{|S|}{\varepsilon} \left[\frac{|S|}{\varepsilon} + \ln\frac{|S|}{\varepsilon}\right]^2\right),$$

where $Y = \ln\frac{1}{\delta} \ln\frac{1}{\varepsilon(1-\gamma)}$.

**Corollary 1.** R-MAX is PAC-MDP.

By corollary 1, in theory, R-MAX performs efficiently for any MDP. In practice, however, it is very slow compared with other well-known algorithms such as Sarsa($\lambda$) (Rummery and Niranjan, 1994), because R-MAX requires many trials in order to guarantee the theoretical bound for all states and actions. We note that R-MAX uses no prior knowledge for the reinforcement learning problem.

3 TR-MAX

In this section, we propose the learning algorithm TR-MAX, which uses prior knowledge in order to speed up learning.

First, we define a heuristic transition function $T_h: S \times A \times S \rightarrow [0, 1]$ and a heuristic reward function $R_h: S \times A \rightarrow [0, 1]$. They are regarded as prior knowledge of $M$. Next, we define the useful knowledge for efficient reinforcement learning.

**Definition 5.** For a state-action pair $(s, a) \in S \times A$, we define the set $Z_{s,a} \subset S$ of zero-transitioned states by

$$Z_{s,a} := \{ s' \in S | T(s'|s, a) = T_h(s'|s, a) = 0 \}.$$  

This means that the agent never transitions to any $s' \in Z_{s,a}$ from state $s$ by action $a$, and this fact is
known to the agent via \(T_h\). We simply denote \(|Z_{min}| = \min_{(s,a) \in S \times A} |Z_{a}|\) in the sequel.

**Theorem 2.** Let \(M\) be a finite MDP whose optimal action value is upper bounded by \(V_{max}\), and \(T_h\) (resp. \(R_h\)) be a heuristic transition (resp. reward) function for \(M\). For any \(0 < \varepsilon < \frac{1}{\gamma T_h} \), we define the set \(P_k \subset S \times A\) of useful state-action pairs by

\[
P_k := \left\{ (s,a) \in S \times A \left| |R(s,a) - R_h(s,a)| \leq \frac{\varepsilon (1 - \gamma)}{V_{max}}, \sum_{s' \in S} T(s'|s,a) - T'(s'|s,a) \leq \frac{\varepsilon (1 - \gamma)}{V_{max}} \right. \right\}
\]

This means that the heuristic values provided by \(T_h\) and \(R_h\) are close enough to the true probabilities for the state-action pairs \((s,a) \in P_k\).

Algorithm 1 describes our algorithm TR-MAX, which is based on R-MAX. the input arguments \(T_h\), \(R_h\), \(Z\) and \(P_k\), and lines 7-16 add to R-MAX, to utilize the prior knowledge for \(M\); \(T'(s'|s,a)\) and \(\hat{R}(s,a)\) are initialized by using \(T_h(s'|s,a)\) and \(R_h(s,a)\) respectively if they are useful (lines 13-13), and \(Q(s,a)\) is initialized to reflect them (lines 18-18). Most model-based algorithms consume a considerable number of timesteps to make the empirical probabilities close to the true probabilities for all pairs \((s,a) \in S \times A\). whereas TR-MAX does it only for \((s,a) \notin P_k\) owing to the prior knowledge. We show the sample complexity of TR-MAX as follows.

**Theorem 2.** Let \(0 < \varepsilon < \frac{1}{\gamma T_h} \) and \(0 < \delta < 1\) be any real numbers, and \(T_h\) (resp. \(R_h\)) be a heuristic transition (resp. reward) function for \(M\). Suppose that two inputs \(m\) and \(\varepsilon_1\) of TR-MAX in \(M\) satisfy \(m = O\left(\frac{|S| - |Z_{min}| + \ln\frac{|S| |A| - |P_e|}{\delta}}{\varepsilon (1 - \gamma)}\right)^2\), where \(X = V_{max}/(\varepsilon (1 - \gamma))\), and \(\delta = O\left(\frac{1}{X}\right)\). Then, \(V_M(s_t) \geq V_{\hat{M}}(s_t) - \varepsilon\) is true for all but

\[
o\left(\frac{|S| |A| - |P_e|}{\delta}\left| |S| - |Z_{min}| + \ln\frac{|S| |A| - |P_e|}{\delta}\right| X^2 Y \right)
\]

where \(Y = \ln\frac{1}{\delta} + \frac{1}{\varepsilon (1 - \gamma)}\) timesteps with probability at least \(1 - \delta\).

As a special case, if \(T'(s'|s,a) = T_h(s'|s,a)\) holds for all \((s,a,s') \in S \times A \times S\), then \(V_M(s_t) \geq V_{\hat{M}}(s_t) - \varepsilon\) is true for all but

\[
o\left(\frac{|S| |A| - |P_e|}{\delta}\ln\frac{|S| |A| - |P_e|}{\delta}\right) X^2 Y \right)
\]

timesteps with probability at least \(1 - \delta\).

By Theorem 2, the sample complexity of TR-MAX is less than that of R-MAX if either \(|Z_{min}|\) or \(|P_e|\) is non-zero. When they are bigger, a smaller sample complexity is realized.

We prove Theorem 2 after reviewing the existing theorems and lemmas and providing new lemmas.

**Definition 7.** An algorithm \(A\) is greedy if the action \(a_t^*\) of \(A\) is \(a_t^* = \arg\max_{a \in A} Q(s_t | a)\) at any timestep \(t\), where \(s_t\) is the \(i\)-th state reached by the agent.

**Definition 8.** Let \(M\) be a finite MDP, and let \(0 < \varepsilon < \frac{1}{\gamma T_h} \) and \(0 < \delta < 1\) be any real numbers. Let \(m = m(M, \varepsilon, \delta)\) be an integer determined by \(M\), \(\varepsilon\), and \(\delta\). During learning process, let \(K_t\) be the set of state-action pairs \((s,a)\) that have been experienced by the agent at least \(m\) times until timestep \(t\), and we call \(K_t\) a known state-action pair. For \(K_t\), we define a set \(S_{new}\) of new states, where each element \(s_{new} \in S_{new}\) corresponds to an unknown state-action pair \((s,a) \notin K_t\). We then define the known state-action MDP \(K_{K_t} = (S \cup S_{new}, A, T_{K_t}, R_{K_t}, \gamma)\), where \(T_{K_t}\) and \(R_{K_t}\) are defined as follows:

- For \(s \in S_{new}\),

**Algorithm 1:** TR-MAX.

```plaintext
Input: S, A, \gamma, m, \varepsilon, T_h, R_h, Z and P_e
1 forall the (s,a) \in S \times A do
2 Q(s,a) \leftarrow \frac{1}{|S|}; select(s,a) \leftarrow 0;
3 forall s' \in S do trans(s,a,s') \leftarrow 0;
4 reward(s,a) \leftarrow 0;
5 forall s \in Z do \hat{T}(s'|s,a) \leftarrow T_h(s'|s,a);
6 if (s,a) \in P_k then
7 forall s' \in S do \hat{T}(s'|s,a) \leftarrow T_h(s'|s,a);
8 \hat{R}(s,a) \leftarrow R_h(s,a); select(s,a) \leftarrow m;
9 for i = 1, 2, 3, ..., \left\lceil\frac{1}{\varepsilon^2} \right\rceil do
10 forall the (s,a) \in P_k do
11 Q(s,a) \leftarrow \hat{R}(s,a) + \gamma \sum_{s' \in S} \hat{T}(s'|s,a) max_{a' \in A} Q(s', a');
12 for i = 1, 2, 3, ... do
13 Let s denote the state at timestep t;
14 Choose action a := \arg\max_{a \in A} Q(s | a);
15 Execute a and obtain the next state s' and the reward r;
16 if select(s,a) < m then
17 select(s,a) \leftarrow select(s,a) + 1;
18 trans(s,a,s') \leftarrow trans(s,a,s') + 1;
19 reward(s,a) \leftarrow reward(s,a) + r;
20 if select(s,a) = m then
21 for i = 1, 2, 3, ..., \left\lceil\frac{1}{\varepsilon^2} \right\rceil do
22 forall the (s,a) \in S \times A do
23 if select(s,a) \geq n then
24 Q(s,a) \leftarrow \hat{R}(s,a) + \gamma \sum_{s' \in S} \hat{T}(s'|s,a) max_{a' \in A} Q(s', a');
```
Reducing Sample Complexity in Reinforcement Learning by Transferring Transition and Reward Probabilities

\[ T_{K_t}(\xi_{sa}, a) = 1 \text{ for each } a \in A, \]
\[ R_{K_t}(\xi_{sa}, a) = Q(s, a)(1 - \gamma) \text{ for each } a \in A. \]

- For every state \( s \in S \) and \((s, a) \in K_t\),
  \[ T_K(s, a) = T(s', s, a) \text{ for each } s' \in S, \]
  \[ R_K(s, a) = R(s, a). \]
- For every state \( s \in S \) and \((s, a) \notin K_t\),
  \[ T_K(s, a) = 1, \]
  \[ R_K(s, a) = Q(s, a)(1 - \gamma). \]

**Theorem 3** (Strehl et al., 2009). Let \( \mathcal{A}(\varepsilon, \delta) \) be any \( \varepsilon \)-greedy algorithm, \( K_t \) be the known state-action pairs at timestep \( t \), and \( M_{K_t} \) be the known state-action MDP. Assume that \( Q(s, a) \leq V_{\text{max}} \) for all timestep \( t \) and \((s, a) \in S \times A \). Suppose that for any inputs \( \varepsilon \) and \( \delta \), with probability at least \( 1 - \delta \), the following conditions hold for all state \( s \), action \( a \), and timestep \( t \):

- **optimism:** \( V_t(s) \geq V^*_t(s) - \varepsilon \)
- **accuracy:** \( V_t(s) - V^*_t(s) \leq \varepsilon \)

**learning complexity:** the total number of update of the action-value estimates plus the number of times the agent experiences some state-action pair \((s, a) \notin K_t\) is bounded by \( \xi_{sa}(\varepsilon, \delta) \).

Note that \( V_t(s) \) denotes the estimated value of the state \( s \) at timestep \( t \). Then, when \( \mathcal{A}(\varepsilon, \delta) \) is executed on \( M \), the inequality \( V_{\text{max}}(s) \geq V^*_t(s) - 4\varepsilon \) holds for all but:

\[ O\left(\frac{V_{\text{max}}(\xi_{sa}(\varepsilon, \delta))}{\varepsilon(1 - \delta)}\ln\frac{1}{\delta}\ln\frac{1}{\varepsilon(1 - \delta)}\right) \quad (3) \]

timesteps \( t \) with probability at least \( 1 - 2\delta \).

**Definition 9.** We say that the \( \epsilon_1 \)-close event occurs if for every stationary policy \( \pi \), timestep \( t \) and state \( s \) during the execution of TR-MAX on some MDP \( M \), where \( |V_{\text{max}}(s) - V^*_t(s)| \leq \epsilon_1 \).

**Lemma 1.** Let \( M \) be a finite MDP whose optimal action value is upper bounded by \( V_{\text{max}} \). There exists a constant \( C \) such that if TR-MAX is executed on \( M \) for \( m \) satisfying:

\[ m \geq \frac{CV_{\text{max}}^2}{\epsilon_1^2(1 - \gamma)} \left( |S| - |Z_{\text{min}}| + \ln \frac{|S||A| - |P|}{\delta} \right), \quad (4) \]

then the \( \epsilon_1 \)-close event will occur with probability at least \( 1 - \delta \).

As a special case, suppose that \( T(s'|s, a) = T_h(s'|s, a) \) for any \((s, a, s') \in S \times A \times S \). If \( m \) satisfies:

\[ m \geq \frac{CV_{\text{max}}^2}{\epsilon_1^2(1 - \gamma)} \left( |S||A| - |P| \right) \delta \quad (5) \]

then the \( \epsilon_1 \)-close event will occur with probability at least \( 1 - \delta \).

**Proof.** By Lemma 12 in (Strehl et al., 2009), if we obtain \( \frac{CV_{\text{max}}(1 - \gamma)}{V_{\text{max}}} \)-approximate transition and reward probabilities, then for any policy \( \pi \) and any state-action pair \((s, a) \in S \times A \) we have \( |Q_{M_{K_t}}(s, a) - Q_{M_{K_t}}^*(s, a)| \leq \epsilon_1 \). That is, the \( \epsilon_1 \)-close event occurs.

Then, we choose large enough \( m \) to obtain \( \frac{CV_{\text{max}}(1 - \gamma)}{V_{\text{max}}} \)-approximate transition and reward probabilities. We now fix a state-action pair \((s, a) \in S \times A \) arbitrarily. By Lemma 13 and Lemma 14 in (Strehl et al., 2009), we obtain the following inequalities for the transition and reward probabilities:

\[ \sqrt{\frac{1}{2m} \ln \frac{2}{\delta}} \leq \frac{CV_{1}(1 - \gamma)}{V_{\text{max}}}, \quad (6) \]
\[ \frac{2\ln(2|S| - 2) - \ln \delta}{m} \leq \frac{CV_{1}(1 - \gamma)}{V_{\text{max}}}. \quad (7) \]

Let us focus on the quantity \(|S|\) on the left-hand of the inequality in (7), which represents the number of states that can be transitioned from state \( s \) by action \( a \). If there is no prior knowledge, the number of possible states is \(|S|\) indeed. However, we have prior knowledge for \( Z_{s,a} \). By the definition of \( Z_{s,a} \), the agent never transitions to the state \( s' \in Z_{s,a} \) from \((s, a) \). Therefore, we can replace \(|S|\) with \(|S| - |Z_{s,a}|\), and obtain a better inequality:

\[ \frac{2\ln(2|S| - |Z_{s,a}|) - \ln \delta}{m} \leq \frac{CV_{1}(1 - \gamma)}{V_{\text{max}}}. \quad (8) \]

By the inequalities in (6) and (8), if \( m \) satisfies:

\[ m \geq \frac{CV_{\text{max}}^2}{\epsilon_1^2(1 - \gamma)} \left( |S| - |Z_{s,a}| + \ln \frac{1}{\varepsilon_1} \right) \]

we then obtain \( \frac{CV_{1}(1 - \gamma)}{V_{\text{max}}} \)-approximate transition and reward probabilities with probability at least \( 1 - \delta' \). Then, in order to ensure a total failure probability of \( \delta \), we set \( \delta' = \frac{\delta}{|S||A| - |P|} \). We have, however, the prior knowledge for \( P_e \). By the definition of \( P_e \), we already have \( \frac{CV_{1}(1 - \gamma)}{V_{\text{max}}} \)-approximate transition and reward probabilities for all \((s, a) \in P_e \). Thus, we can set \( \delta' = \frac{\delta}{|S||A| - |P|} \). If \( m \) satisfies the condition in (4) then the \( \epsilon_1 \)-close event occurs with probability at least \( 1 - \delta \).

Next, we consider the case that \( T(s'|s, a) = T_h(s'|s, a) \) for any \((s, a, s') \in S \times A \times S \). In this case, by Lemma 12 and Lemma 13 in (Strehl et al., 2009), \( m \) only has to satisfy the condition in (6). Then, we have:

\[ m \geq \frac{CV_{\text{max}}^2}{\epsilon_1^2(1 - \gamma)^2} \ln \frac{1}{\delta}. \]

Similar to the condition in (4), we have the condition in (5) by setting \( \delta' = \frac{\delta}{|S||A| - |P|} \).

We are now ready to prove Theorem 2.
Proof. (Theorem 2) We show that TR-MAX satisfies the three conditions in Theorem 3: optimism, accuracy, and learning complexity. We begin by showing that optimism and accuracy are satisfied if the $\varepsilon_1$-close event occurs, and then explain that the learning complexity is sufficient to cause the $\varepsilon_1$-close event.

First, we verify optimism. Similar to the proof of Theorem 11 in (Strehl et al., 2009) (we referred it as Theorem 1 in this paper), we obtain the inequalities as follows:

$$V_t(s) \geq V_{\hat{M}_s^s}(s) - \varepsilon_1 \quad (9)$$

$$\geq V_{\hat{M}_s^s}(s) - 2\varepsilon_1 \quad (10)$$

$$\geq V_{\hat{M}_s^s}(s) - 2\varepsilon_1. \quad (11)$$

By Proposition 4 in (Strehl et al., 2009), we have the inequality in (9); it shows that the estimated state value $V_t(s)$ obtained by the value iteration is $\varepsilon_1$-close to the true state value in the empirical known state-action MDP $\hat{M}_s^s$. The inequality in (10) comes from the assumption that the $\varepsilon_1$-close event occurs, and the inequality in (11) comes from the definition of the known state-pair MDP $M_{K_s}$. By setting $\varepsilon_1 = \frac{s}{2}$, optimism is ensured.

Next, we verify accuracy. By the definition of $M_{K_s}$, the state value of $M_{K_s}$ is more valuable than that of $M$ for all state, such that $V_t(s) \leq V_{\hat{M}_s^s}(s)$. Because we assumed that the $\varepsilon_1$-close event occurs, $V_{\hat{M}_s^s}(s) - V_{\hat{M}_s^s}(s) \leq \varepsilon_1$ holds. By $\varepsilon_1 = \frac{s}{2}$, we have that $V_t(s) - V_{\hat{M}_s^s}(s) - V_{\hat{M}_s^s}(s) \leq \varepsilon_1 < \varepsilon$ and accuracy is also ensured.

Finally, we explain that the learning complexity is sufficient to cause the $\varepsilon_1$-close event. If we set

$$m = \frac{CV_2(1-\gamma)}{\varepsilon_1^2(1-\gamma^2)} \left( |S| - |Z_{min}| + \ln \frac{|S| |A| - |P_s|}{\delta} \right),$$

then the $\varepsilon_1$-close event occurs with probability at least $1 - \delta$ by Lemma 1. Moreover, if we set $\zeta(\varepsilon, \delta) = m(|S||A| - |P_s|)$, then $\zeta(\varepsilon, \delta)$ satisfies the condition of learning complexity, because the action value of the state-action pair $(s,a) \in K_s$ is never updated for any timestep $t$, and $K_s$ consists of the state-action pair such that $select(s,a) \geq m$. In addition, by the definition of $P_s$, the state-action pair $(s,a) \in P_s$ is included in $K_s$ for any timestep $t$.

We showed that the three conditions in Theorem 3 are satisfied, and obtained the sample complexity in (1) by applying $\zeta(\varepsilon, \delta)$ to (3).

Similarly, for the special case that $T(s'|s,a) = T_h(s'|s,a)$ for any $(s,a,s') \in S \times A \times S$, we obtain (2) by applying (5) in Lemma 1.

4 EXPERIMENTS

We compare TR-MAX with R-MAX for the reinforcement learning task of mazes, as illustrated in Figure 1. Maze tasks are popular in reinforcement learning (see e.g., (Sutton and Barto, 1998; Schuitema et al., 2010; Saito et al., 2012)). The agent observes its own position as a state, and an initial state is the cell marked “S” (the state space is of size $|S| = 100$). The goal of the agent is to reach the cell marked “G”, by selecting an action among UP, DOWN, RIGHT and LEFT at each step (thus, the action space is of size $|A| = 4$).

If the selected direction is blocked by a wall, the agent remains at the same position. Moreover, we assume that the environment is noisy. With a probability of $0.2$, the next state is randomly chosen from its neighbors of the current state, regardless of the selected action. The discount factor is $\gamma = 0.7$. The agent receives a reward 1 only when the agent reaches the goal; otherwise, the reward is 0. One trial consists of either “from start to reaching the goal state” or “from start to the timestep $t$ that exceeds a fixed time limit.” The aim of the agent is to minimize the number of timesteps per trial.

A target task is a task to be solved, and a source task is an accomplished task that may be similar to the target task. We illustrate target and source tasks in Figure 1. In the experiments, we provide the true transition (resp. reward) probability of each source task as the heuristic transition (resp. reward) probability for the target task. For $(s,a) \in P_s$, the empirical transition (resp. reward) probability is initialized to the heuristic transition (reps. reward) probability. $m$ is derived from $Z_{s,a}$ for each state-action pair.

In the experiments, we executed each algorithm
10 times for target task. In each execution, the agent experiences 240,000 trials. Then, we calculated the average steps that the agent consumes by reaching the goal for each trail. Figure 2 shows the results of the initial 60,000 trials because almost no change occurred after 60,000 trials. TR-MAX converged at about 70,000 to 20,000 trials, whereas R-MAX did not converge within 60,000 trials. Moreover, we observe that the similarity of the source task to the target task affects the convergence speed. For instance, among these three source tasks, Source task 1 in Figure 1(a) is the most similar to the target task in Figure 1(b), and “TR-MAX (Source task 1)” in Figure 2 converged the fastest among others. On the other hand, “TR-MAX (Source task 3)” converged very slowly, because Source task 3 in Figure 1(d) is not similar to the target. In this sense, we verified that TR-MAX could effectively utilize the prior knowledge.

Next, we compare the sample complexities of R-MAX and TR-MAX in Table 1. As expected, the sample complexities of TR-MAX are by far smaller than that of R-MAX, and they reflect the similarities. We also note that the size $|Z_{min}|$ of the minimum zero-transitioned sets does not.

### 5 CONCLUDING REMARKS

We proposed the TR-MAX algorithm that improved the R-MAX algorithm by using prior knowledge for a target task obtained from a source task. We proved that the sample complexity of TR-MAX is indeed smaller than that of R-MAX, and that TR-MAX is PAC-MDP. In computational experiments, we verified that TR-MAX could learn much more efficiently than R-MAX.

In our future work, we are interested in capturing “knowledge transfer in reinforcement learning” in another way. In this position paper, we captured this as a zero-transitioned state set and a useful state-action pair set defined by the heuristic transition and heuristic reward functions. In reality, however, these functions cannot be obtained without knowing the true transition and reward probabilities. In a real environment, we never know the true probabilities, which should be treated in future work.

### REFERENCES


