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

Kouta Oguni[1], Kazuyuki Narisawa[1] and Ayumi Shinohara[1]
[1]*Graduate 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 position 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 $R : S \times$

Oguni K., Narisawa K. and Shinohara A..
Reducing Sample Complexity in Reinforcement Learning by Transferring Transition and Reward Probabilities.
DOI: 10.5220/0004915606320638
In *Proceedings of the 6th International Conference on Agents and Artificial Intelligence* (ICAART-2014), pages 632-638
ISBN: 978-989-758-015-4

$A \to [0,1]$. M denotes a finite MDP in the sequel.

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

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

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

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

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

Definition 3 ((Kakade, 2003))**.** *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 $V_M^{\mathcal{A}_t}(s_t) < V_M^{\pi^*}(s_t) - \varepsilon$. 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 ((Kakade, 2003))**.** *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 $V_M^{\mathcal{A}_t}(s_t) < V_M^{\pi^*}(s_t) - \varepsilon$. 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 $select(s,a)$ and the number of transitions $trans(s,a,s')$ for any state $s, s' \in S$ and action $a \in A$, and calculate the *empirical transition probability* $T(s'|s,a)$ 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 $select(s,a) \geq m$. The sample complexity of R-MAX is analyzed by (Strehl et al., 2009) as follows.

Theorem 1 ((Strehl et al., 2009))**.** *Let $0 < \varepsilon < \frac{1}{1-\gamma}$ and $0 < \delta < 1$ be any real numbers, and assume that the two inputs m and ε_1 of R-MAX in M satisfy $m = O\left(\left(|S| + \ln\frac{|S|\,|A|}{\delta}\right)X^2\right)$ where $X = V_{\max}/(\varepsilon(1-\gamma))$ and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$. Let $\mathcal{A}_t$ denote the policy of R-MAX at timestep t and s_t denote the state at timestep t. Then, $V_M^{\mathcal{A}_t}(s_t) \geq V_M^*(s_t) - \varepsilon$ is true for all but*

$$O\left(|S||A|\left(|S| + \ln\frac{|S||A|}{\delta}\right)X^3 Y\right),$$
$$\text{where } Y = \ln\frac{1}{\delta}\ln\frac{1}{\varepsilon(1-\gamma)}$$

timesteps t with probability at least $1-\delta$.

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(λ) (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* $T_h : S \times A \times S \to [0,1]$ and a *heuristic reward function* $R_h : S \times A \to [0,1]$. 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*

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

This means that the agent never transitions to any $s' \in Z_{s,a}$ from state s by action a, and this fact is

[1] The R-MAX algorithm is obtained from Algorithm 1 by omitting lines 7-16, because our TR-MAX algorithm is an extension of the R-MAX algorithm.

known to the agent via T_h. We simply denote $|Z_{\min}| = \min_{(s,a)\in S\times A}|Z_{s,a}|$ in the sequel.

Definition 6. *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 $0<\varepsilon<\frac{1}{1-\gamma}$, we define the set $P_\varepsilon \subset S\times A$ of* useful state-action pairs *by*

$$P_\varepsilon := \left\{ (s,a)\in S\times A \;\middle|\; |R(s,a)-R_h(s,a)| \le \frac{\varepsilon(1-\gamma)}{V_{\max}}, \right. \\ \left. \sum_{\bar{s}\in S}\left|T(\bar{s}|s,a)-T'(\bar{s}|s,a)\right| \le \frac{\varepsilon(1-\gamma)}{V_{\max}} \right\}.$$

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 $(s,a)\in P_\varepsilon$.

Algorithm 1 describes our algorithm TR-MAX, which is based on R-MAX: the input arguments T_h, R_h, Z and P_ε, and lines 7-16 are added to R-MAX, to utilize the prior knowledge for M; $\hat{T}(s'|s,a)$ 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 7-13); and $Q(s,a)$ is initialized to reflect them (lines 14-16). 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* $(s,a)\notin P_\varepsilon$, owing to the prior knowledge. We show the sample complexity of TR-MAX as follows.

Theorem 2. *Let $0<\varepsilon<\frac{1}{1-\gamma}$ and $0<\delta<1$ be any real numbers, and T_h (reps. R_h) be a heuristic transition (resp. reward) function for M. Suppose that two inputs m and ε_1 of TR-MAX in M satisfy $m = O\left(\left(|S|-|Z_{\min}|+\ln\frac{|S||A|-|P_\varepsilon|}{\delta}\right)X^2\right)$, where $X = V_{\max}/(\varepsilon(1-\gamma))$, and $\frac{1}{\varepsilon_1} = O\left(\frac{1}{\varepsilon}\right)$. Then, $V_M^{\mathcal{A}_t}(s_t) \ge V_M^*(s_t)-\varepsilon$ is true for all but*

$$O\left((|S||A|-|P_\varepsilon|)\left(|S|-|Z_{\min}|+\ln\frac{|S||A|-|P_\varepsilon|}{\delta}\right)X^3Y\right) \quad (1)$$

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

timesteps t 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 $V_M^{\mathcal{A}_t}(s_t)\ge V_M^*(s_t)-\varepsilon$ is true for all but

$$O\left((|S||A|-|P_\varepsilon|)\ln\frac{|S||A|-|P_\varepsilon|}{\delta}X^3Y\right) \quad (2)$$

timesteps t 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 $|P_\varepsilon|$ is non-zero. When they are bigger, a smaller sample complexity is realized.

Algorithm 1: TR-MAX.

Input: $S, A, \gamma, m, \varepsilon_1, T_h, R_h, Z$ and P_ε

1 **forall the** $(s,a)\in S\times A$ **do**
2 $Q(s,a)\leftarrow\frac{1}{1-\gamma}$; $select(s,a)\leftarrow 0$;
3 **forall** $s'\in S$ **do** $trans(s,a,s')\leftarrow 0$;
4 $reward(s,a)\leftarrow 0$;
5 **forall** $s'\in Z_{s,a}$ **do** $\hat{T}(s'|s,a)\leftarrow T_h(s'|s,a)$;
6 **if** $(s,a)\in P_\varepsilon$ **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** $i=1,2,3,\ldots,\left\lceil\frac{\ln(1/\varepsilon_1(1-\gamma))}{1-\gamma}\right\rceil$ **do**
10 **forall the** $(s,a)\in P_\varepsilon$ **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** $t=1,2,3,\ldots$ **do**
13 Let s denote the state at timestep t;
14 Choose action $a:=\operatorname{argmax}_{a\in A}Q(s,a)$;
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** $i=1,2,3,\ldots,\left\lceil\frac{\ln(1/\varepsilon_1(1-\gamma))}{1-\gamma}\right\rceil$ **do**
22 **forall the** $(\bar{s},\bar{a})\in S\times A$ **do**
23 **if** $select(\bar{s},\bar{a})\ge m$ **then**
24 $Q(\bar{s},\bar{a})\leftarrow\hat{R}(\bar{s},\bar{a})+\gamma\sum_{s'\in S}\hat{T}(s'|\bar{s},\bar{a})\max_{a'\in A}Q(s',a')$;

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

Definition 7. *An algorithm $\mathcal{A}$ is* greedy *if the action a_t of $\mathcal{A}$ is $a_t=\operatorname{argmax}_{a\in A}Q(s_t,a)$ 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 $0<\varepsilon<\frac{1}{1-\gamma}$ and $0<\delta<1$ be any real numbers. Let $m=m(M,\varepsilon,\delta)$ be an integer determined by M, ε, and δ. 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 $\xi_{s,a}\in S_{\text{new}}$ corresponds to an* unknown *state-action pair $(s,a)\notin K_t$. We then define the* known state-action MDP *$M_{K_t}=\langle S\cup S_{\text{new}}, A, T_{K_t}, R_{K_t}, \gamma\rangle$, where T_{K_t} and R_{K_t} are defined as follows:*

- *For $s\in S_{\text{new}}$,*

$T_{K_t}(\xi_{s,a} \mid \xi_{s,a}, \bar{a}) = 1$ *for each* $\bar{a} \in A$,
$R_{K_t}(\xi_{s,a}, \bar{a}) = Q(s,a)(1-\gamma)$ *for each* $\bar{a} \in A$.

- *For* $s \in S$ *and* $(s,a) \in K_t$,
  $T_{K_t}(\bar{s} \mid s,a) = T(\bar{s} \mid s,a)$ *for each* $\bar{s} \in S$,
  $R_{K_t}(s,a) = R(s,a)$.
- *For* $s \in S$ *and* $(s,a) \notin K_t$,
  $T_{K_t}(\xi_{s,a} \mid s,a) = 1$,
  $R_{K_t}(s,a) = Q(s,a)(1-\gamma)$.

Theorem 3 ((Strehl et al., 2009)). *Let* $\mathcal{A}(\varepsilon,\delta)$ *be any 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* $Q_t(s,a) \leq V_{\max}$ *for all timestep* t *and* $(s,a) \in S \times A$*. Suppose that for any inputs* ε *and* δ*, with probability at least* $1-\delta$*, the following conditions hold for all state* s*, action* a *and timestep* t*:*

optimism: $V_t(s) \geq V_M^{\pi^*}(s) - \varepsilon$

accuracy: $V_t(s) - V_{M_{K_t}}^{\pi_t}(s) \leq \varepsilon$

learning complexity: *the total number of updates of the action-value estimates plus the number of times that the agent experiences some state-action pair* $(s,a) \notin K_t$ *is bounded by* $\zeta(\varepsilon,\delta)$.

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* $V_M^{\mathcal{A}_t}(s_t) \geq V_M^*(s_t) - 4\varepsilon$ *holds for all but*

$$O\left(\frac{V_{\max}\zeta(\varepsilon,\delta)}{\varepsilon(1-\delta)} \ln\frac{1}{\delta} \ln\frac{1}{\varepsilon(1-\delta)}\right) \tag{3}$$

timesteps t *with probability at least* $1-2\delta$.

Definition 9. *We say that* the ε_1-close event occurs *if for every stationary policy* π*, timestep* t *and state* s *during the execution of TR-MAX on some MDP* M*, where* $\left|V_{M_{K_t}}^{\pi}(s) - V_{\hat{M}_{K_t}}^{\pi}(s)\right| \leq \varepsilon_1$ *holds.*

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

$$m \geq \frac{CV_{\max}^2}{\varepsilon_1^2(1-\gamma)^2}\left(|S| - |Z_{\min}| + \ln\frac{|S||A| - |P_\varepsilon|}{\delta}\right), \tag{4}$$

then the ε_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*

$$m \geq \frac{CV_{\max}^2}{\varepsilon_1^2(1-\gamma)^2} \ln\frac{|S||A| - |P_\varepsilon|}{\delta} \tag{5}$$

then the ε_1*-close event will occur with probability at least* $1-\delta$.

Proof. By Lemma 12 in (Strehl et al., 2009), if we obtain $\frac{C\varepsilon_1(1-\gamma)}{V_{\max}}$-approximate transition and reward probabilities, then for any policy π and any state-action pair $(s,a) \in S \times A$ we have $\left|Q_{M_{K_t}}^{\pi}(s,a) - Q_{\hat{M}_{K_t}}^{\pi}(s,a)\right| \leq \varepsilon_1$. That is, the ε_1-close event occurs. Then, we choose large enough m to obtain $\frac{C\varepsilon_1(1-\gamma)}{V_{\max}}$-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:

$$\sqrt{\frac{1}{2m}\ln\frac{2}{\delta'}} \leq \frac{C\varepsilon_1(1-\gamma)}{V_{\max}}, \tag{6}$$

$$\sqrt{\frac{2[\ln(2^{|S|}-2) - \ln\delta']}{m}} \leq \frac{C\varepsilon_1(1-\gamma)}{V_{\max}}. \tag{7}$$

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:

$$\sqrt{\frac{2[\ln(2^{|S|-|Z_{s,a}|}-2) - \ln\delta']}{m}} \leq \frac{C\varepsilon_1(1-\gamma)}{V_{\max}}. \tag{8}$$

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

$$m \propto \frac{V_{\max}^2}{\varepsilon_1^2(1-\gamma)^2}\left(|S| - |Z_{s,a}| + \ln\frac{1}{\delta'}\right)$$

we then obtain $\frac{C\varepsilon_1(1-\gamma)}{V_{\max}}$-approximate transition and reward probabilities with probability at least $1-\delta'$. Then, in order to ensure a total failure probability of δ, we set δ'. Without prior knowledge, we have to set $\delta' = \frac{\delta}{|S||A|}$. We have, however, the prior knowledge for P_ε. By the definition of P_ε, we already have $\frac{C\varepsilon_1(1-\gamma)}{V_{\max}}$-approximate transition and reward probabilities for all $(s,a) \in P_\varepsilon$. Thus, we can set $\delta' = \frac{\delta}{|S||A|-|P_\varepsilon|}$, and if m satisfies the condition in (4) then the ε_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

$$m \propto \frac{V_{\max}^2}{\varepsilon_1^2(1-\gamma)^2} \ln\frac{1}{\delta'}.$$

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

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 ε_1-close event occurs, and then explain that the learning complexity is sufficient to cause the ε_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:

$$V_t(s) \geq V^{\pi^*}_{\hat{M}_{K_t}}(s) - \varepsilon_1 \quad (9)$$

$$\geq V^{\pi^*}_{M_{K_t}}(s) - 2\varepsilon_1 \quad (10)$$

$$\geq V^{\pi^*}_{M}(s) - 2\varepsilon_1. \quad (11)$$

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 ε_1-close to the true state value in the empirical known state-action MDP $\hat{M}_{K_t}$. The inequality in (10) comes from the assumption that the ε_1-close event occurs, and the inequality in (11) comes from the definition of the known state-pair MDP M_{K_t}. By setting $\varepsilon_1 = \frac{\varepsilon}{2}$, optimism is ensured.

Next, we verify accuracy. By the definition of M_{K_t}, the state value of M_{K_t} is more valuable than that of M for all state, such that $V_t(s) \leq V^{\pi_t}_{\hat{M}_{K_t}}(s)$. Because we assumed that the ε_1-close event occurs, $V^{\pi_t}_{\hat{M}_{K_t}}(s) - V^{\pi_t}_{M_{K_t}}(s) \leq \varepsilon_1$ holds. By $\varepsilon_1 = \frac{\varepsilon}{2}$, we have that $V_t(s) - V^{\pi_t}_{M_{K_t}} \leq V^{\pi_t}_{\hat{M}_{K_t}}(s) - V^{\pi_t}_{M_{K_t}}(s) \leq \varepsilon_1 < \varepsilon$ and accuracy is also ensured.

Finally, we explain that the learning complexity is sufficient to cause the ε_1-close event. If we set

$$m = \frac{CV^2_{\max}}{\varepsilon_1^2(1-\gamma)^2}\left(|S| - |Z_{\min}| + \ln\frac{|S||A| - |P_\varepsilon|}{\delta}\right),$$

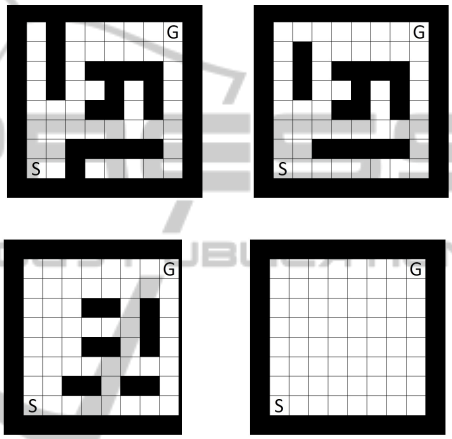
then the ε_1-close event occurs with probability at least $1-\delta$ by Lemma 1. Moreover, if we set $\zeta(\varepsilon,\delta) = m(|S||A| - |P_\varepsilon|)$, then $\zeta(\varepsilon,\delta)$ satisfies the condition of learning complexity, because the action value of the state-action pair $(s,a) \in K_t$ is never updated for any timestep t, and K_t consists of the state-action pair such that $select(s,a) \geq m$. In addition, by the definition of P_ε, the state-action pair $(s,a) \in P_\varepsilon$ is included in K_t 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 $|A| = 4$).



(c) Source task 2 (d) Source task 3

Figure 1: Reinforcement learning tasks of a maze.

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 $(s,a) \in P_\varepsilon$, 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

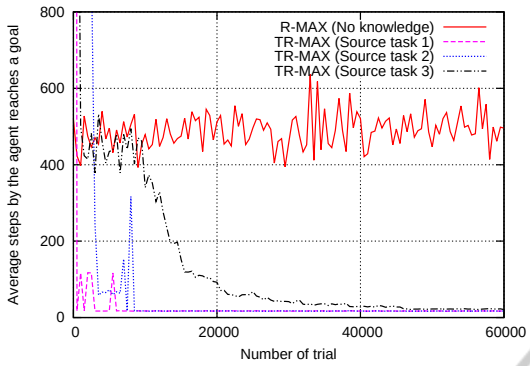


Figure 2: Average number of steps above first 60,000 trials in the target task for R-MAX and TR-MAX using each source task's knowledge.

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 10,000 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(b) is the most similar to the target task in Figure 1(a), 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.

Table 1: Sample complexities of R-MAX and TR-MAX using each source task's knowledge. $|Z_{\min}|$ is the minimum size of the zero-transitioned state set, and $|P_\varepsilon|$ is the size of the useful state-action pairs for the target task.

	R-MAX	TR-MAX		
		Source task 1	Source task 2	Source task 3
Sample complexity $\times 10^7$	4970	3.88	4.99	5.34
Ratio to R-MAX	100%	0.078%	0.100%	0.107%
m	46742738	36550	46956	50243
$\lvert Z_{\min}\rvert$	–	96	95	95
$\lvert P_\varepsilon\rvert$	–	376	304	196

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 the complexity of R-MAX, and they reflect the similarities. We also note that the size $|P_\varepsilon|$ of the useful state-action pairs P_ε significantly depends on the similarity, where 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

Brafman, R. I. and Tennenholtz, M. (2003). R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *The Journal of Machine Learning*, 3:213–231.

Kakade, S. M. (2003). *On the Sample Complexity of Reinforceent Learning*. PhD thesis, University College London.

Konidaris, G. and Barto, A. (2006). Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proc of ICML*, pages 489–496.

Kretchmar., R. M. (2002). Parallel reinforcement learning. In *Proc. of SCI*, pages 114–118.

Mann, T. A. and Choe, Y. (2012). Directed exploration in reinforcement learning with transferred knowledge. In *Proc. of EWRL*, pages 59–76.

Miyazaki, K., Yamamura, M., and Kobayashi, S. (1997). k-certainty exploration method: an action selector to identify the environment in reinforcement learning. *Artificial intelligence*, 91(1):155–171.

Rummery, G. A. and Niranjan, M. (1994). *On-line Q-learning using connectionist systems*. Cambridge University.

Saito, J., Narisawa, K., and Shinohara, A. (2012). Prediction for control delay on reinforcement learning. In *Proc. of ICAART*, pages 579–586.

Schuitema, E., Busoniu, L., Babuska, R., and Jonker, P. (2010). Control delay in reinforcement learning

for real-time dynamic systems: a memoryless approach. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3226–3231. IEEE.

Strehl, A. L., Li, L., and Littman, M. L. (2009). Reinforcement learning in finite mdps: Pac analysis. *The Journal of Machine Learning Research*, 10:2413–2444.

Strehl, A. L. and Littman, M. L. (2005). A theoretical analysis of model-based interval estimation. In *Proc. of ICML*, pages 857–864.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning*. MIT Press.

Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proc. of ICML*, pages 330–337.

Taylor, M. E. and Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, 10:1633–1685.

Taylor, M. E., Stone, P., and Liu, Y. (2007). Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(1):2125–2167.