Episodic Return Decomposition by Difference of Implicitly Assigned Sub-trajectory Reward

Haoxin Lin[1,2,3], **Hongqiu Wu**[1,2], **Jiaji Zhang**[1,2], **Yihao Sun**[1,2], **Junyin Ye**[1,2,3], **Yang Yu**[1,2,3,4*]

[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[2]School of Artificial Intelligence, Nanjing University, Nanjing, China
[3]Polixir Technologies, Nanjing, China
[4]Peng Cheng Laboratory, Shenzhen, 518055, China
{linhx, wuhq, zhangjj, sunyh, yejy}@lamda.nju.edu.cn, yuy@nju.edu.cn

Abstract

Real-world decision-making problems are usually accompanied by delayed rewards, which affects the sample efficiency of Reinforcement Learning, especially in the extremely delayed case where the only feedback is the episodic reward obtained at the end of an episode. Episodic return decomposition is a promising way to deal with the episodic-reward setting. Several corresponding algorithms have shown remarkable effectiveness of the learned step-wise proxy rewards from return decomposition. However, these existing methods lack either attribution or representation capacity, leading to inefficient decomposition in the case of long-term episodes. In this paper, we propose a novel episodic return decomposition method called Diaster (**D**ifference of **i**mplicitly **a**ssigned **s**ub-**t**rajectory **re**ward). Diaster decomposes any episodic reward into credits of two divided sub-trajectories at any cut point, and the step-wise proxy rewards come from differences in expectation. We theoretically and empirically verify that the decomposed proxy reward function can guide the policy to be nearly optimal. Experimental results show that our method outperforms previous state-of-the-art methods in terms of both sample efficiency and performance. The code is available at https://github.com/HxLyn3/Diaster.

Introduction

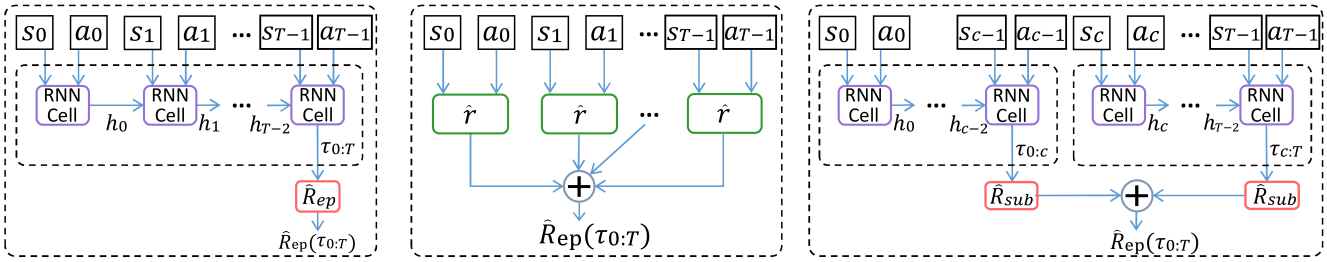
Reinforcement Learning (RL) has made empirical successes in several simulated domains and attracted close attention in recent years. Sparse and delayed reward, usually facing a range of real-world problems such as industrial control (Hein et al. 2017), molecular design (Olivecrona et al. 2017), and recommendation systems (Chen et al. 2018), is one of the critical challenges hindering the application of RL in reality. The trouble with the delayed reward function is that it affects RL's sample efficiency when using it to estimate the value function. Specifically, delayed rewards introduce numerous noneffective updates and fitting capacity loss (Lyle, Rowland, and Dabney 2022) into Temporal-Difference (TD) Learning and result in a high variance in Monte-Carlo (MC) Learning (Arjona-Medina et al. 2019). The problem is more severe in the highly delayed case that the feedback appears only at the end of the episode.

*Corresponding Author

It is essential to design a proxy Markovian reward function as the substitute for the original delayed one since current standard RL algorithms prefer instant feedback at each environmental step. The apparent solution is designing the required proxy reward function by handcraft, but it depends on domain knowledge and is not general for all tasks. Reward shaping (Mataric 1994; Randløv and Alstrøm 1998; Ng, Harada, and Russell 1999), one kind of automatical reward design method, is widely observed that it can accelerate RL. However, the reshaped rewards are coupled with the original rewards and are ineffective in environments where only an episodic reward is fed back. Recent work (Arjona-Medina et al. 2019; Gangwani, Zhou, and Peng 2020; Efroni, Merlis, and Mannor 2021; Ren et al. 2022) proposes to decompose the episodic reward into a string of step-wise rewards through the trajectory, which seems promising to deal with the long-term delay.

Previous return decomposition methods can be sorted into two classes, as shown in Figure 1(a) and 1(b). One pays attention to precisely predicting the long-term episodic return, thinking little of reward redistribution. The representative work is RUDDER (Arjona-Medina et al. 2019), which utilizes an LSTM (Hochreiter and Schmidhuber 1997) network to make a return prediction and assigns rewards to each pair of state-action via contribution analysis from a backward view. Another focuses on step-wise return decomposition without considering temporal structural representations. A little work (Efroni, Merlis, and Mannor 2021; Liu et al. 2019; Ren et al. 2022) aimed at the least-squares-based return decomposition objective falls into this class. Nevertheless, the above work suffers from a lack of either attribution or representation capacity, leading to inefficient episodic return decomposition in the case of long-term episodes.

In this paper, we propose to cut an entire trajectory at any time step and decompose the episodic return into two sub-trajectory rewards, as demonstrated in Figure 1(c). The step-wise reward can be straightforwardly obtained by differencing the implicitly assigned sub-trajectory reward in expectation. Based on this key idea, we formulate a simple yet effective episodic return decomposition method called Diaster (**D**ifference of **i**mplicit **a**ssigned **s**ub-**t**rajectory **re**ward), which can be embedded in any model-free RL algorithm. This new class of return decomposition not only introduces



(a) Attribute a return to the full trajectory. (b) Attribute a return to each step. (c) Attribute a return to any two cut sub-trajectories.

Figure 1: Comparison of different episodic return decomposition. (a) introduces a temporal structure to represent the episodic return function $\hat{R}_{\text{ep}}(\tau_{0:T})$. The step-wise proxy rewards can be assigned via contribution analysis from a backward view. (b) directly decomposes the episodic return into the step-wise proxy rewards $\{\hat{r}(s_i, a_i)\}_{i=0}^{T}$. (c) decomposes the episodic return into two sub-trajectory rewards, $\hat{R}_{\text{sub}}(\tau_{0:c})$ and $\hat{R}_{\text{sub}}(\tau_{c:T})$, for any cut point c.

temporal structural representations into the episodic reward function but also properly attributes the return to different parts of a given trajectory. Hence, a practical proxy reward function can be learned to guide the policy to be optimal.

In general, our contributions are summarized as follows:

- We elaborate on the formulation of our new episodic return decomposition method, *i.e.*, Diaster, and present its practical neuron-based implementation.
- We theoretically verify that the step-wise proxy reward function learned through Diaster is return-equivalent to the original MDP in expectation and capable of guiding the policy to be nearly optimal.
- We empirically show that Diaster can provide effective proxy rewards for RL algorithms and outperform previous state-of-the-art return decomposition methods in terms of both sample efficiency and performance.
- We conduct an ablation study to show that setting the number of cut points for Diaster can achieve a trade-off between long-term representation and attribution.

Related Work

Delayed Reward

The setting of delayed rewards (Walsh et al. 2009; Bouteiller et al. 2021; Han et al. 2022) usually accompanies real-world RL problems, resulting in a high bias in TD Learning and high variance in MC Learning (Arjona-Medina et al. 2019). This paper considers the extremely delayed case in which only one episodic feedback can be obtained at the end of an episode. The problem is how to automatically find a proxy dense reward function to encourage an efficient value estimation in any task with the episodic-reward setting.

Reward Shaping

Reward shaping is one of the popular methods of replacing the original rewards with more frequent feedback to accelerate RL agents' learning. Early work (Dorigo and Colombetti 1994; Mataric 1994; Randløv and Alstrøm 1998) focuses on designing the shaping reward function and shows an improved learning efficiency. However, these methods need more consideration for the consistency of the optimal policy. Potential-based reward shaping (PBRS) (Ng, Harada, and Russell 1999) proposes to guarantee the policy invariance property by restricting the shaping function to a difference of potential functions defined over states. Several variants (Wiewiora, Cottrell, and Elkan 2003; Devlin and Kudenko 2012; Harutyunyan et al. 2015) of PBRS introduce more information into the potential function to provide additional expressive power for proxy rewards and improve the sample efficiency. These potential-based methods do not correspond to an optimal return decomposition as their proxy rewards are additively coupled with the original rewards, according to (Arjona-Medina et al. 2019).

Return Decomposition

Episodic return decomposition is promising to tackle the rewards with an extreme delay. RUDDER (Arjona-Medina et al. 2019) uses an LSTM network (Hochreiter and Schmidhuber 1997) to predict the return of an episode and decompose the predicted return into contributions of each pair of state-action in the sequence. (Efroni, Merlis, and Mannor 2021) directly optimizes the least-squares-based decomposition objective, *i.e.*, the expected square error between the sum of the state-action pairs' proxy rewards and the corresponding trajectory-wise reward. This direct return decomposition method is extended by (Liu et al. 2019) to decompose the episodic return into the rewards of all time intervals. IRCR (Gangwani, Zhou, and Peng 2020) considers a uniform reward redistribution of the return to each constituent state-action pair, which is an intuitive solution of the optimal reward problem (ORP) (Barto, Lewis, and Singh 2009; Singh et al. 2010). To bridge direct return decomposition and IRCR, RRD (Ren et al. 2022) lets the proxy rewards from a random subsequence of the trajectory to fit the episodic return, achieving a trade-off between the decomposition complexity and the estimation accuracy.

Preliminaries

A finite-horizon Markov Decision Process (MDP) with an episodic-reward setting is characterized by a tuple

$\langle \mathcal{S}, \mathcal{A}, P, \rho_0, r, R_{\text{ep}}, T \rangle$. The episodic horizon is limited by the scalar T. Each episode starts with an environmental state $s_0 \in \mathcal{S}$ sampled from the initial state distribution ρ_0. At each time step $t \in \{0, \dots, T-1\}$, after agent observing the state s_t and performing a $a_t \in \mathcal{A}$, the environment transitions to the next state $s_{t+1} \in \mathcal{S}$ according to the unknown transition function $P(s_{t+1}|s_t, a_t)$. The step-wise reward $r(s_t, a_t)$ is unobservable to the agent, while the only feedback called episodic reward $R_{\text{ep}}(\tau_{0:T})$ is accessible at the end of the trajectory $\tau_{0:T} = (s_0, a_0, s_1, a_1, ..., s_{T-1}, a_{T-1})$. In this paper, we consider an assumption that the episodic reward comes from the sum of the step-wise rewards, *i.e.*, $R_{\text{ep}}(\tau_{0:T}) = \sum_{t=0}^{T-1} r(s_t, a_t)$, which is satisfied in most cases. The goal under this setting is to find the optimal policy that maximizes the expectation of episodic return (*i.e.*, episodic reward): $\mathbb{E}_{\rho^\pi}[R_{\text{ep}}(\tau_{0:T})]$, where ρ^π induced by the dynamics function $P(s_{t+1}|s_t, a_t)$ and the policy $\pi(a_t|s_t)$ is the on-policy distribution over states.

The state-action value function defined as

$$Q^\pi(s_t, a_t) = \mathbb{E}_{P,\pi}\left[\sum_{i=0}^{T-t-1} r(s_{t+i}, a_{t+i}) \middle| s_t, a_t\right] \tag{1}$$

cannot be estimated since the immediate reward isn't fed back from the environment. Directly using the delayed episodic reward to learn the state-action value function leads to the optimal policy consistently, as proven by (Arjona-Medina et al. 2019). However, it isn't an appropriate choice as it introduces high bias in TD Learning and high variance in MC Learning. The promising way to solve any MDP with an episodic-reward setting is to find a proxy dense reward function $\hat{r}(s, a)$ that nearly satisfies the sum-from decomposition,

$$R_{\text{ep}}(\tau_{0:T}) \approx \hat{R}_{\text{ep}}(\tau_{0:T}) = \sum_{t=0}^{T-1} \hat{r}(s_t, a_t), \tag{2}$$

which is considered in several previous work (Arjona-Medina et al. 2019; Efroni, Merlis, and Mannor 2021; Liu et al. 2019; Ren et al. 2022; Raposo et al. 2021).

Diaster: Difference of Implicitly Assigned Sub-Trajectory Reward

In this section, we will first elaborate on our return decomposition method Diaster (**D**ifference of **i**mplicit **a**ssigned **s**ub-**t**rajectory **re**ward), and next verify the effectiveness of this method theoretically. Then we will present a practical neuron-based implementation of Diaster.

Description of Diaster

Rather than directly decomposing the episodic reward into the step-wise rewards, we consider finding a proxy sub-trajectory reward function $\hat{R}_{sub}$ that satisfies

$$\hat{R}_{\text{sub}}(\varnothing) = 0, \tag{3}$$

$$\hat{R}_{\text{sub}}(\tau_{0:T}) \approx R_{\text{ep}}(\tau_{0:T}), \tag{4}$$

$$\hat{R}_{\text{sub}}(\tau_{0:c}) + \hat{R}_{\text{sub}}(\tau_{c:T}) \approx R_{\text{ep}}(\tau_{0:T}), \forall c \in \{1, \dots, T-1\}, \tag{5}$$

for any episode $\tau_{0:T}$, where $\tau_{0:c}$ is the former c-step sub-trajectory, $\tau_{c:T}$ is the latter $\{T-c\}$-step sub-trajectory, and $\hat{R}_{\text{sub}}(\varnothing)$ is the supplementary definition of the empty sub-trajectory $\tau_{0:0} = \varnothing$. Compared to the step-wide decomposition (2), this formulation of decomposition can ease the difficulty of credit assignment for several reasons:

- Only two components need to be assigned the reward after cutting an episode at any cut-point c.
- T cut-points to cut an episode provides more relations between the proxy reward function and the episodic reward function, increasing the training samples for episodic return decomposition.
- The step-wise decomposition (2) can be regarded as a special case of the subtrajectory-wise decomposition (5) that lets $\hat{R}_{\text{sub}}(\tau_{0:c}) = \sum_{t=0}^{c-1} \hat{r}(s_t, a_t)$ and $\hat{R}_{\text{sub}}(\tau_{c:T}) = \sum_{t=c}^{T-1} \hat{r}(s_t, a_t)$. However, a sub-trajectory reward does not have to come from the sum of its state-action pairs' rewards in practice, while it can be any other function of the sub-trajectory. The general formulation can relax the assumption that decides how the episodic reward comes.
- This form of return decomposition introduces some temporal structural representations into the episodic reward function, encouraging the utilization of RNNs (Elman 1990) to take advantage of the sequential information.

Given any episode $\tau_{0:T}$ sampled by the policy π, the proxy reward connected with its preceding sub-trajectory is obtained by difference as

$$\hat{R}_d(s_0, a_0) = \hat{R}_{\text{sub}}(\tau_{0:1}) - \hat{R}_{\text{sub}}(\varnothing), \tag{6}$$

$$\hat{R}_d(\tau_{0:t}, s_t, a_t) = \hat{R}_{\text{sub}}(\tau_{0:t+1}) - \hat{R}_{\text{sub}}(\tau_{0:t}), \tag{7}$$

for any (s_t, a_t) belonging to $\tau_{0:T}$. The method of difference is also considered by (Arjona-Medina et al. 2019) to guarantee the equivalent episodic return since

$$\sum_{t=0}^{T-1} \hat{R}_d(\tau_{0:t}, s_t, a_t) = \hat{R}_{\text{sub}}(\tau_{0:T}) \approx R_{\text{ep}}(\tau_{0:T}). \tag{8}$$

However, the above reward function $\hat{R}_d$ is still delayed and non-Markovian. Thus we define the step-wise proxy reward function as

$$\hat{r}^\pi(s_t, a_t) = \mathbb{E}_{\tau_{0:t} \sim \Gamma_t^\pi(\cdot|s_t)}\left[\hat{R}_d(\tau_{0:t}, s_t, a_t)\right], \tag{9}$$

where

$$\Gamma_t^\pi(\tau_{0:t}|s_t) = \frac{\mathcal{T}_t^\pi(\tau_{0:t}) P(s_t|s_{t-1}, a_{t-1})}{\rho^\pi(s_t)} \tag{10}$$

is the distribution conditioned on the state s_t over the t-step sub-trajectory space, while $\mathcal{T}_t^\pi(\tau_{0:t})$ is the unconditioned distribution that comes from

$$\mathcal{T}_t^\pi(\tau_{0:t}) = \rho_0(s_0)\pi(a_0|s_0) \prod_{i=1}^{t-1} P(s_i|s_{i-1}, a_{i-1})\pi(a_i|s_i). \tag{11}$$

Analysis of Diaster

Our analysis in this subsection focuses on determining whether the proxy reward function and the corresponding proxy state-action value function are effective for policy optimization. We start by presenting a simple theorem, offering some insights about the relationship between the step-wise reward $\hat{r}^\pi$ and the sub-trajectory reward $\hat{R}_{\text{sub}}$.

Theorem 1. *Given any policy π, the following equation holds for any subsequence length $h \in \{2, \dots, T\}$ and time step $t \in \{0, \dots, T-h\}$, that is,*

$$\mathbb{E}_{\rho^\pi}\left[\sum_{i=t}^{t+h-1} \hat{r}^\pi(s_i, a_i)\right] = \mathbb{E}_{\rho^\pi}\left[\hat{R}_{sub}(\tau_{0:t+h}) - \hat{R}_{sub}(\tau_{0:t})\right]. \tag{12}$$

Proof. See Appendix A.1. □

This theorem states that the sum of any consecutive h-step sequence of $\hat{r}^\pi$ is the difference of two sub-trajectory rewards that differ by h steps, in expectation. In particular, letting $t = 0$ and $h = T$, it becomes

$$\mathbb{E}_{\rho^\pi}\left[\sum_{t=0}^{T-1} \hat{r}^\pi(s_t, a_t)\right] \approx \mathbb{E}_{\rho^\pi}\left[R_{\text{ep}}(\tau_{0:T})\right], \tag{13}$$

which reveals that $\hat{r}^\pi$ is one of the return-equivalent proxy rewards of the original MDP. Therefore, the goal of policy optimization does not change while using $\hat{r}^\pi$ as the immediate guidance.

Next we aim to show that using the proxy reward $\hat{r}^\pi(s_t, a_t)$ to learn the proxy state-action value function defined as

$$\hat{Q}^\pi(s_t, a_t) = \mathbb{E}_{P,\pi}\left[\sum_{i=0}^{T-t-1} \hat{r}^\pi(s_{t+i}, a_{t+i}) \middle| s_t, a_t\right], \tag{14}$$

nearly leads to the optimal policy in the original MDP. The necessary condition that $\hat{Q}^\pi(s_t, a_t)$ needs to satisfy for the optimal policy consistency is provided by Lemma 1.

Lemma 1. *For any finite MDP with a state-action value function $Q^\pi(s, a)$ defined based on the unobservable step-wise reward function $r(s, a)$, any proxy state-action value function $\hat{Q}^\pi(s, a)$ that satisfies*

$$\hat{Q}^\pi(s, a) = Q^\pi(s, a) + \delta(s), \tag{15}$$

where $\delta(s)$ is any function defined over the state space, will lead to the same optimal policy as Q^π, whether using value-based or policy gradient methods.

Proof. See Appendix A.2. □

This lemma indicates that once the difference of $\hat{Q}^\pi(s, a)$ and $Q^\pi(s, a)$ is only related to the state s, the original optimal policy can be guaranteed even though using $\hat{Q}^\pi(s, a)$ to optimize the policy. In order to verify that $\hat{Q}^\pi(s, a)$ can satisfy the condition (15), we further extend Theorem 1 to the conditional form.

Algorithm 1: Diaster

Input: Neural parameters ψ, ϕ, replay buffer $\mathcal{D}$, learning rate $\lambda_{\hat{R}}$, $\lambda_{\hat{r}}$, batch size B, and RL algorithm RL-ALGO containing the initial policy π_θ.
for N episodes **do**
  Collect a trajectory $\tau_{0:T}$ using the policy π_θ.
  Store $\tau_{0:T}$ and its feedback $R_{\text{ep}}(\tau_{0:T})$ in $\mathcal{D}$.
  for M batches **do**
    Sample B trajectories $\{\tau^i_{0:T}\}_{i=0}^{B-1}$ from $\mathcal{D}$.
    Sample B scalars $\{c^i\}_{i=0}^{B-1}$ from $\{0, \dots, T-1\}$.
    $\psi \leftarrow \psi - \nabla_\psi \frac{1}{B} \sum_{i=0}^{B-1} l_{\hat{R}}(\psi, \tau^i_{0:T}, c^i)$ by Eq.(19).
    $\phi \leftarrow \phi - \nabla_\phi \frac{1}{B} \sum_{i=0}^{B-1} l_{\hat{r}}(\phi, \tau^i_{0:c^i+1})$ by Eq.(21).
    Sample B transitions $\{(s_i, a_i, s_{i+1})\}_{i=0}^{B-1}$ from $\mathcal{D}$.
    Use RL-ALGO to optimize the policy π_θ with $\{(s_i, a_i, \hat{r}^\phi(s_i, a_i), s_{i+1})\}_{i=0}^{B-1}$.
  end for
end for

Theorem 2. *Given any policy π, the following equation holds for any $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$ at any time step $t \in \{0, \dots, T-1\}$, that is,*

$$\begin{aligned}&\mathbb{E}_{P,\pi}\left[\sum_{i=1}^{T-t-1} \hat{r}^\pi(s_{t+i}, a_{t+i}) \middle| s_t, a_t\right] \\ &\quad = \mathbb{E}_{P,\pi}\left[\hat{R}_{sub}(\tau_{0:T}) - \hat{R}_{sub}(\tau_{0:t+1}) \middle| s_t, a_t\right].\end{aligned} \tag{16}$$

Proof. See Appendix A.3. □

By applying Theorem 2, it is observed that

$$\begin{aligned}&\hat{Q}^\pi(s_t, a_t) \\ =&\hat{r}^\pi(s_t, a_t) + \mathbb{E}_{P,\pi}\left[\hat{R}_{\text{sub}}(\tau_{0:T}) - \hat{R}_{\text{sub}}(\tau_{0:t+1}) \middle| s_t, a_t\right] \\ \approx&\mathbb{E}_{P,\pi}\left[\sum_{i=0}^{T-1} r(s_i, a_i) - \hat{R}_{\text{sub}}(\tau_{0:t}) \middle| s_t, a_t\right] \\ \approx&Q^\pi(s_t, a_t) + \mathbb{E}_{P,\pi}\left[\sum_{i=0}^{t-1} r(s_i, a_i) - \hat{R}_{\text{sub}}(\tau_{0:t}) \middle| s_t\right],\end{aligned} \tag{17}$$

which conforms the form of the condition (15) with $\delta(s_t) = \mathbb{E}_{P,\pi}\left[\sum_{i=0}^{t-1} r(s_i, a_i) - \hat{R}_{\text{sub}}(\tau_{0:t}) \middle| s_t\right]$. Hence $\hat{Q}^\pi$ is an effective substitute of Q^π to optimize the policy, in the case of original reward r being inaccessible.

Practical Implementation of Diaster

The primary problem is how to obtain an implicitly assigned sub-trajectory reward that satisfies (5). The error of the approximation (5) directly decides the optimality of the policy learned from the proxy rewards. Considering to extract the temporal structures contained in sub-trajectories, we use an RNN with a GRU (Cho et al. 2014) cell, parameterized by ψ, to represent the sub-trajectory reward function $\hat{R}^\psi_{\text{sub}}$. We train $\hat{R}^\psi_{\text{sub}}$ by minimizing the expected loss of episodic return

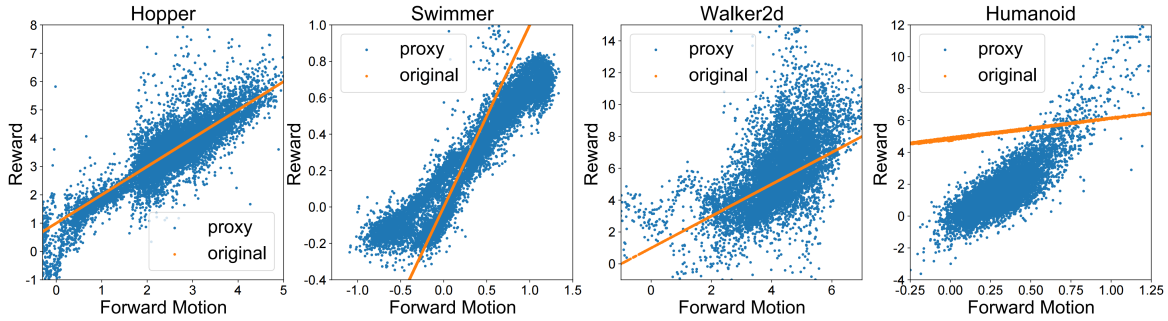


Figure 2: Proxy Reward Visualization. For each sampled pair of state-action (s, a), we show the forward motion of the agent along with the learned proxy reward $\hat{r}^{\phi}(s, a)$ (blue point) and the original unobservable reward $r(s, a)$ (orange point).

decomposition:

$$\mathcal{L}_{\hat{R}}(\psi) = \mathop{\mathbb{E}}_{\tau_{0:T}\in\mathcal{D}}\left[\mathop{\mathbb{E}}_{c\sim U(\{0,\dots,T-1\})}\left[l_{\hat{R}}(\psi, \tau_{0:T}, c)\right]\right], \quad (18)$$

with the replay buffer $\mathcal{D}$, and

$$l_{\hat{R}}(\psi, \tau_{0:T}, c) = \left(\hat{R}^{\psi}_{\text{sub}}(\tau_{0:c}) + \hat{R}^{\psi}_{\text{sub}}(\tau_{c:T}) - R_{\text{ep}}(\tau_{0:T})\right)^2. \quad (19)$$

The step-wise proxy reward function $\hat{r}^{\phi}$, with neural parameters ϕ, is trained to predict the difference of sub-trajectory rewards. The expected loss is

$$\mathcal{L}_{\hat{r}}(\phi) = \mathop{\mathbb{E}}_{\tau_{0:T}\in\mathcal{D}}\left[\mathop{\mathbb{E}}_{t\sim U(\{0,\dots,T-1\})}\left[l_{\hat{r}}(\phi, \tau_{0:t+1})\right]\right], \quad (20)$$

where

$$l_{\hat{r}}(\phi, \tau_{0:t+1}) = \left(\hat{r}^{\phi}(s_t, a_t) - \left(\hat{R}^{\psi}_{\text{sub}}(\tau_{0:t+1}) - \hat{R}^{\psi}_{\text{sub}}(\tau_{0:t})\right)\right)^2 \quad (21)$$

is the prediction loss for any given sub-trajectory $\tau_{0:t+1} = (s_0, a_0, \dots, s_t, a_t)$.

The complete algorithm of Diaster is described in Algorithm 1, where RL-ALGO can be any off-policy RL algorithm. It is efficient to apply Diaster in tasks with an episodic-reward setting due to its easy implementation and few hyper-parameters requiring tuning.

Experiments

In this section, we focus on four primary questions: 1) What is the step-wise proxy reward function of Diaster like? 2) How well does Diaster perform on RL benchmark tasks? 3) What will happen if decomposing the episodic return into implicit rewards of more than two sub-trajectories? 4) Is it necessary to learn the step-wise proxy reward function?

Experimental Setting

We conduct a series of experiments on MuJoCo (Todorov, Erez, and Tassa 2012) and PointMaze (Fu et al. 2020) benchmarks with the same episodic-reward setting as (Gangwani, Zhou, and Peng 2020; Ren et al. 2022). Specifically, we modify the original environment to prevent the agent from accessing the instant reward. Instead, zero signals will be received by the agent at non-terminal states. The only feedback is the episodic reward $R_{\text{ep}}(\tau_{0:T})$ returned at the end of an episode. Other environmental settings are kept as default.

In the following experiments, our Diaster is implemented based on SAC (Haarnoja et al. 2018), one of the state-of-the-art model-free RL algorithms in continuous control tasks. The hyper-parameters are given in Appendix B.

Proxy Reward Visualization

We aim to visualize the step-wise proxy rewards learned through our Diaster on four MuJoCo tasks with the episodic-reward setting, including Hopper, Swimmer, Walker2d, and Humanoid. The principal goal of these environments is to let the robot move in the forward direction as fast as possible in the limited time horizon. To coincide with the episodic goal, the proxy reward function has to be related to the forward motion at any time step. Therefore, revealing the relationship between the learned proxy rewards and the agent's forward motion can indicate the effectiveness of return decomposition. We sample several state-action pairs in the environment and show their proxy rewards along with the step-wise displacements of forward motion. What's more, we show the original environmental rewards for reference.

Figure 2 shows that the point with more distance of forward motion tends to have a greater proxy reward. The nearly monotonic tendency shows how the proxy rewards depend on the forward motion and indicates that Diaster can discover the latent attribution of the episodic reward function and make an effective decomposition. Intuitively, using such a proxy reward function can encourage the agent to step forward, consistent with the original goal of MuJoCo tasks.

Performance Comparison

We evaluate our Diaster on seven MuJoCo tasks with the episodic-reward setting, including InvertedPendulum, Hopper, Swimmer, Walker2d, Ant, Humanoid, and HumanoidStandup. Three existing episodic return decomposition methods, RUDDER (Arjona-Medina et al. 2019), IRCR (Gangwani, Zhou, and Peng 2020) and RRD (Ren et al. 2022), are selected for comparison.

Figure 3 shows the learning curves of our Diaster (red) and the other three baselines. By contrast, RUDDER is no match for Diaster in all seven tasks. As for RRD and IRCR,

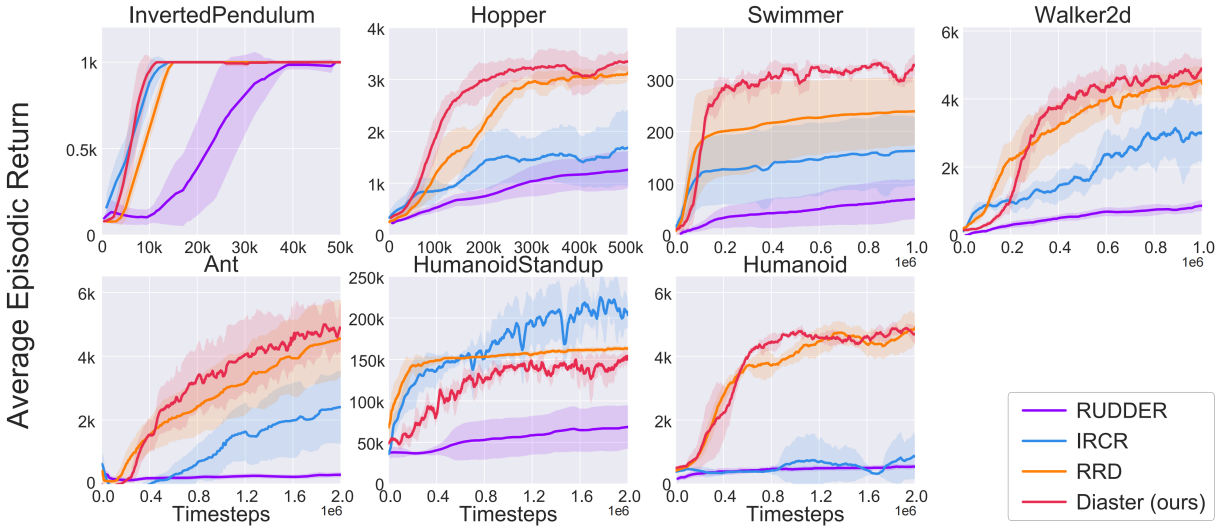


Figure 3: Learning curves of our Diaster (red) and other three baselines on MuJoCo continuous control tasks. The solid lines indicate the mean and shaded areas indicate the standard error over five different random seeds. Each evaluation, taken every 1,000 environmental steps, calculates the average return over ten episodes.

it can be observed that Diaster shows a better sample efficiency than these two algorithms in most of the environments. Only in the HumanoidStandup task, Diaster has a slight disadvantage compared to RRD and IRCR. Although unable to demonstrate an overwhelming superiority, these results show that Diaster has both high sample efficiency and competitive performance on MuJoCo benchmarks.

On Number of Cut Points

A natural question is what will happen if decomposing the episodic return into implicit rewards of more than two sub-trajectories. In this subsection, we conduct an ablation study to show how Diaster performs while changing the number of cut points.

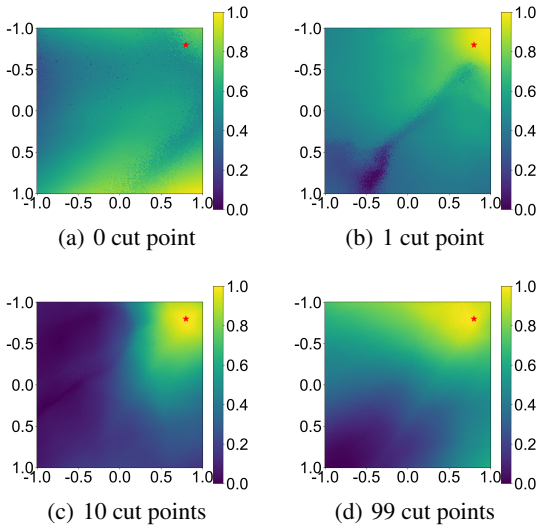
With any m sampled cut points $\{c_i\}_{i=0}^{m-1}$ satisfying that $0 < c_0 < c_1 < \cdots < c_{m-1} < T$, the condition (5) for the sub-trajectory reward function can be extended as

$$\hat{R}_{\text{sub}(\tau_{0:c_0})} + \sum_{i=1}^{m-1} \hat{R}_{\text{sub}(\tau_{c_{i-1}:c_i})} + \hat{R}_{\text{sub}(\tau_{c_{m-1}:T})} \approx R_{\text{ep}}(\tau_{0:T}), \tag{22}$$

while $\hat{R}_{\text{sub}}(\varnothing)$ and $\hat{R}_{\text{sub}}(\tau_{0:T}) \approx R_{\text{ep}}(\tau_{0:T})$ none the less hold. If $m = 0$, this formulation will just learn a function to fit $R_{\text{ep}}(\tau_{0:T})$, without any attribution of the episodic reward. If $m = T - 1$, this formulation will equal step-wise return decomposition (2), which is incapable of bringing in any temporal structural representation. Any m in $\{1, 2, \ldots, T-2\}$ can achieve a trade-off between representation and attribution of the episodic reward function.

First, we visualize the influence of m on the learned proxy value function. For the sake of explainability, the experiment is conducted on one PointMaze task (PointMaze_UMaze), where a 2-Degree-of-Freedom ball force-actuated in the cartesian directions x and y is aiming to reach a target goal in a closed maze. We plot the heatmap of the normalized expected cumulative proxy rewards in the future for different numbers of cut points, as shown in Figure 4. The red star is the goal on the map. The closer the position (x, y) is to the goal, the greater its expected proxy value tends to be. The tendency of 10 cut points is the clearest, indicating that a proper m between 0 and $T-1$ can yield reasonable proxy value estimation since the attribution and representation of the episodic return are both considered.

Next, we evaluate Diaster with a set of choices of the cut points' number, $m \in \{0, 1, 5, 10, 100, T-1\}$, on four MuJoCo tasks with the episodic-reward setting, including InvertedPendulum, Hopper, Swimmer, and Walker2d. The results are demonstrated in Figure 5. The performance in-



(a) 0 cut point (b) 1 cut point

(c) 10 cut points (d) 99 cut points

Figure 4: Heatmap of expected cumulative proxy rewards.

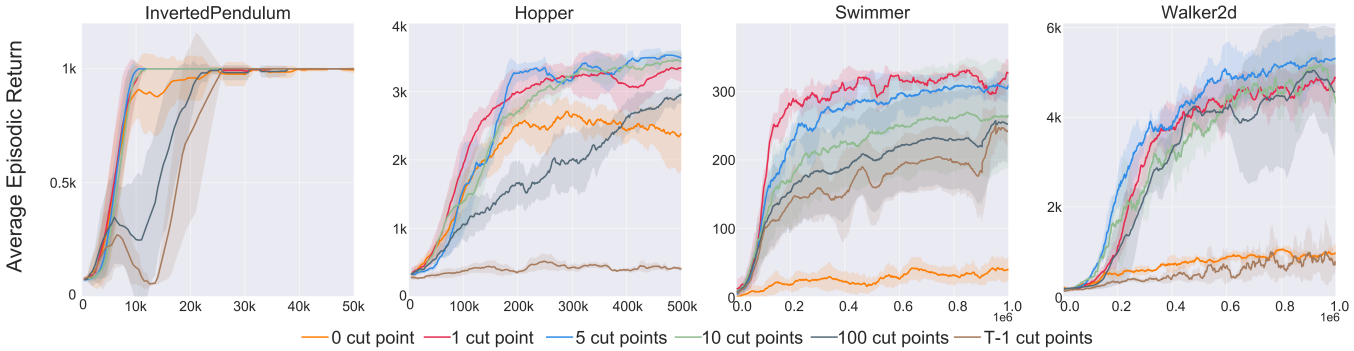


Figure 5: Learning curves of our Diaster with a set of choices of the cut points' number, $m \in \{0, 1, 5, 10, 100, T-1\}$, on MuJoCo continuous control tasks. The solid lines indicate the mean and shaded areas indicate the standard error over five different random seeds. Each evaluation, taken every 1,000 environmental steps, calculates the average return over ten episodes.

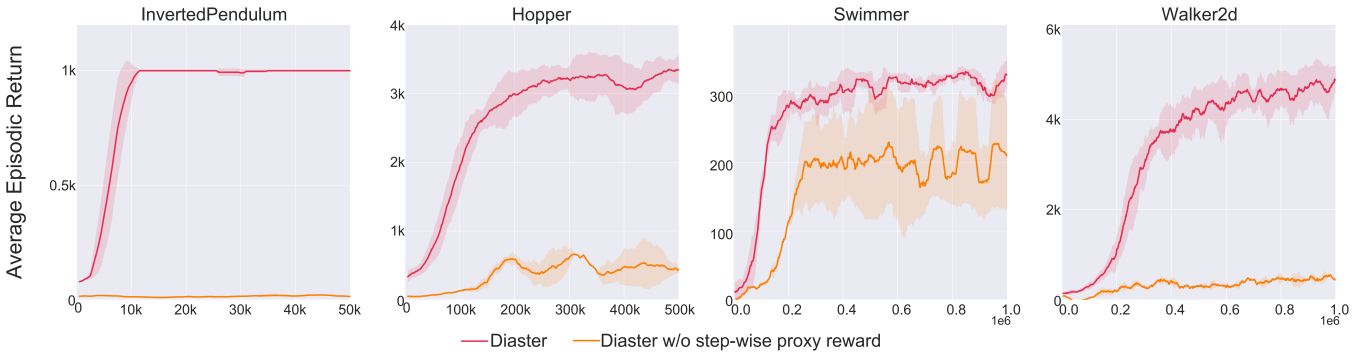


Figure 6: Learning curves of our Diaster, with and without learning the step-wise proxy reward function, respectively.

creases first and then decreases as the number of cut points m increases from 0 to $T-1$. $m=0$ and $m=T-1$ are always unable to achieve an acceptable performance, because $m=0$ restricts the attribution capability while $m=T-1$ restricts the long-term representation. Although the sensitivity of this hyper-parameter depends on the task, an appropriate integer m between 0 and $T-1$ can better approximate the return decomposition object and achieve better performance since it can balance attribution and representation of the episodic return. In all the experimental tasks, $m=1$ or $m=5$ performs the best, which suggests that a relatively small m is befitting.

In conclusion, attribution and representation are both necessary for long-term episodic return decomposition. The trade-off between attribution and representation achieved by our Diaster improves the sample efficiency of RL in the tasks with episodic-reward settings.

On Step-wise Proxy Reward Function Learning

We propose to learn the sub-trajectory reward function $\hat{R}^{\psi}_{\text{sub}}$ by minimizing (19) then learn the step-wise proxy reward function $\hat{r}^{\phi}$ by minimizing (21) and use $\hat{r}^{\phi}$ to provide incentives for the agent. It's also feasible to directly use the difference of $\hat{R}^{\psi}_{\text{sub}}$ to optimize the policy for each trajectory. In this subsection, we conduct an ablation study to show how Diaster performs without learning $\hat{r}^{\phi}$.

We evaluate Diaster, with and without learning the step-wise proxy reward function $\hat{r}^{\phi}$, respectively, on four MuJoCo tasks with the episodic-reward setting, including InvertedPendulum, Hopper, Swimmer, and Walker2d. The results are demonstrated in Figure 6. We point out that $\hat{R}^{\psi}_{\text{sub}}(\tau_{0:t})$ is conditioned on the sub-trajectory $\tau_{0:t}$ and the difference of $\hat{R}^{\psi}_{\text{sub}}$ is non-Markovian. Without a Markovian reward signal, the policy optimization becomes non-stationary. Therefore, the performance deteriorates after removing the step-wise rewards in all four tasks.

Conclusion

In this paper, we present a new episodic return decomposition approach for episodic-reward setting where the only feedback is the episodic reward obtained at the end of an episode. Specifically, we propose to cut an entire trajectory at any time step and decompose the episodic return into two sub-trajectory rewards. Then the step-wise reward can be obtained by differencing the implicitly assigned sub-trajectory reward in expectation. The theoretical results verify that the step-wise proxy reward function learned through Diaster is return-equivalent to the original MDP in expectation and capable of guiding the policy to be nearly optimal. The empirical results show that Diaster can provide effective proxy rewards for RL algorithms and outperform previous state-of-the-art return decomposition methods.

Acknowledgments

This work is supported by the National Science Foundation of China (61921006), and The Major Key Project of PCL (PCL2021A12).

References

Arjona-Medina, J. A.; Gillhofer, M.; Widrich, M.; Unterthiner, T.; Brandstetter, J.; and Hochreiter, S. 2019. RUDDER: Return Decomposition for Delayed Rewards. In *Advances in Neural Information Processing Systems 32 (NeurIPS'19)*. Vancouver, Canada.

Barto, A. G.; Lewis, R. L.; and Singh, S. 2009. Where Do Rewards Come From. *Proceedings of the 31st Annual Meeting of the Cognitive Science Society (CogSci'09)*.

Bouteiller, Y.; Ramstedt, S.; Beltrame, G.; Pal, C. J.; and Binas, J. 2021. Reinforcement Learning with Random Delays. In *9th International Conference on Learning Representations (ICLR'21)*. Virtual Event.

Chen, S.; Yu, Y.; Da, Q.; Tan, J.; Huang, H.; and Tang, H. 2018. Stabilizing Reinforcement Learning in Dynamic Environment with Application to Online Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'18)*. London, UK.

Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, (EMNLP'14)*. Doha, Qatar.

Devlin, S.; and Kudenko, D. 2012. Dynamic potential-based reward shaping. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS'12)*. Valencia, Spain.

Dorigo, M.; and Colombetti, M. 1994. Robot Shaping: Developing Autonomous Agents Through Learning. *Artificial Intelligence*, 71(2): 321–370.

Efroni, Y.; Merlis, N.; and Mannor, S. 2021. Reinforcement Learning with Trajectory Feedback. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI'21)*. Virtual Event.

Elman, J. L. 1990. Finding Structure in Time. *Cognitive Science*, 14(2): 179–211.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *CoRR*, abs/2004.07219.

Gangwani, T.; Zhou, Y.; and Peng, J. 2020. Learning Guidance Rewards with Trajectory-space Smoothing. In *Advances in Neural Information Processing Systems 33 (NeurIPS'20)*. Virtual Event.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*. Stockholmsmässan, Stockholm, Sweden.

Han, B.; Ren, Z.; Wu, Z.; Zhou, Y.; and Peng, J. 2022. Off-Policy Reinforcement Learning with Delayed Rewards. In *Proceedings of the 39th International Conference on Machine Learning (ICML'22)*. Baltimore, Maryland.

Harutyunyan, A.; Devlin, S.; Vrancx, P.; and Nowé, A. 2015. Expressing Arbitrary Reward Functions as Potential-Based Advice. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. Austin, Texas.

Hein, D.; Depeweg, S.; Tokic, M.; Udluft, S.; Hentschel, A.; Runkler, T. A.; and Sterzing, V. 2017. A benchmark environment motivated by industrial control problems. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI'17)*. Honolulu, Hawaii.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780.

Liu, Y.; Luo, Y.; Zhong, Y.; Chen, X.; Liu, Q.; and Peng, J. 2019. Sequence Modeling of Temporal Credit Assignment for Episodic Reinforcement Learning. *CoRR*, abs/1905.13420.

Lyle, C.; Rowland, M.; and Dabney, W. 2022. Understanding and Preventing Capacity Loss in Reinforcement Learning. In *The 10th International Conference on Learning Representations (ICLR'22)*. Virtual Event.

Mataric, M. J. 1994. Reward Functions for Accelerated Learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML'94)*. New Brunswick, NJ.

Ng, A. Y.; Harada, D.; and Russell, S. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the 16th International Conference on Machine Learning (ICML'99)*. Bled, Slovenia.

Olivecrona, M.; Blaschke, T.; Engkvist, O.; and Chen, H. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of Cheminformatics*, 9(1): 48:1–48:14.

Randløv, J.; and Alstrøm, P. 1998. Learning to Drive a Bicycle Using Reinforcement Learning and Shaping. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*. Madison, Wisconsin.

Raposo, D.; Ritter, S.; Santoro, A.; Wayne, G.; Weber, T.; Botvinick, M. M.; van Hasselt, H.; and Song, H. F. 2021. Synthetic Returns for Long-Term Credit Assignment. *CoRR*, abs/2102.12425.

Ren, Z.; Guo, R.; Zhou, Y.; and Peng, J. 2022. Learning Long-Term Reward Redistribution via Randomized Return Decomposition. In *10th International Conference on Learning Representations (ICLR'22)*. Virtual Event.

Singh, S.; Lewis, R. L.; Barto, A. G.; and Sorg, J. 2010. Intrinsically Motivated Reinforcement Learning: An Evolutionary Perspective. *IEEE Transactions on Autonomous Mental Development*, 2(2): 70–82.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12)*. Vilamoura, Portugal.

Walsh, T. J.; Nouri, A.; Li, L.; and Littman, M. L. 2009. Learning and planning in environments with delayed feedback. *Autonomous Agents and Multi-Agent Systems*, 18(1): 83–105.

Wiewiora, E.; Cottrell, G. W.; and Elkan, C. 2003. Principled Methods for Advising Reinforcement Learning Agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*. Washington, DC.