Estimation of agent's rewards using multi-agent maximum discounted causal entropy inverse reinforcement learning

Keiichi Namikoshi
Chiba University, Japan
acka2158@chiba-u.jp

Sachiyo Arai
Chiba University, Japan
arai@tu.chiba-u.ac.jp

ABSTRACT

Multi-agent simulation can reproduce real behavior of a multi-agent system in which people can interact with each other. To reproduce a diverse behavior, it is necessary to design specific action rules for each agent from a plethora of rules. For this problem, imitation learning is used to automatically design rules from action logs of agent's behavior. However, the simulation designer should be able to interpret the purpose of rules designed for each agent to explain the validity of the simulation conducted. Therefore, we propose a novel entropy-based multi-agent inverse reinforcement learning. Multi-agent inverse reinforcement learning is a framework for estimating reward representing the agent's purpose. In this paper, we extend maximum discounted causal entropy inverse reinforcement learning to a Markov game environment. We propose the approach of decomposing the overall problem into a tractable problem for each agent. Compared with previous methods, the proposed method can be applied to Markov game of general-sum and infinite-horizon problems. Experimental results showed that the proposed method can estimate valid rewards in both deterministic and probabilistic transition environments.

KEYWORDS

multi-agent simulation, inverse reinforcement learning

1 INTRODUCTION

Environment containing people that are interested in interacting with each other, e.g., a crowd of people [18], and traffic flow [3] is referred to as a multi-agent system. Studies have been conducted to predict an agent's behavior or understand the purpose of each agent in a multi-agent system.

Multi-agent simulation is a method to reproduce a multi-agent system. In the multi-agent simulation people are treated as agents, an agent's decision-making is represented as behavior rules, i.e., function to determine the agent's action from agent's observation. Interpreting the decision-making process and purpose of an agent is easy because of multi-agent simulation's abovementioned characteristics. However, multi-agent simulation is a bottom-up approach for reproducing behavior of an agent from behavior rules. There is a possibility of individual differences occurring in an agent's action rules, and the combination of rules is enormous. Hence, designing of action rules requires a trial and error approach.

Therefore, we propose a framework that can automatically estimate heterogeneous action rules from the action log [9]. Action log represents trajectories that are a sequence of coordinates and actions, and can be obtained by observing the behavior of an agent using sensors. However, even if it is possible to reproduce the behavior, it is necessary to analyze the estimated action rules to understand the agent's purpose in the system.

Thus, we use multi-agent inverse reinforcement learning for estimating the agent's purpose. Multi-agent inverse reinforcement learning is a framework that can estimate an agent's reward in Markov game [8], which is an extension of the Markov decision process for single-agent environment. Because reward represents the value of a state-action pair, the agent's purpose could be interpreted as a state-action pair with high estimated reward value, i.e., multi-agent inverse reinforcement learning can directly estimate the agent's purpose as a reward.

Here, we propose a novel multi-agent inverse reinforcement learning that is based on entropy maximization principle. Particularly, we extend the maximum discounted causal entropy inverse reinforcement learning to an infinite-horizon problem in Markov game, and decompose this problem to a tractable problem for each agent. In the experiments performed, we showed that proposed method can estimate valid reward from action log generated using deterministic policy of Nash equilibrium in simple grid world.

2 PROBLEM DEFINITION

Markov game represents $< \mathcal{N}, \mathcal{S}, \{\mathcal{A}_n\}_{n\in\mathcal{N}}, T, \{R_n\}_{n\in\mathcal{N}} >$ where $\mathcal{N}$ is a set of agents ($|\mathcal{N}| \geq 2$), $\mathcal{S}$ is a finite set of states, $\mathcal{A}_n$ is a finite set of actions of agent n, $T : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_{|\mathcal{N}|} \times \mathcal{S} \rightarrow [0, 1]$ is state transition probability, and $R_n : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is reward of agent n. The action of agent n is $a_n \in \mathcal{A}_n$, i.e., the joint action of all agents is $a \in \mathcal{A}$. We assume that all agents are able to access the state $s \in \mathcal{S}$.

We estimate the reward of all agents $\{R_n\}_{n\in\mathcal{N}}$ from action log, which is a variable length of the trajectory set $\mathcal{D} = \{\{s_t, a_t\}_{t=0}^{t_d}\}_{d=1}^{D}$, in a Markov game that does not include a reward. Herein, experts mean the agents generating action log and expert's trajectories means $\mathcal{D}$. Furthermore, the estimator does not know T and expert's policy ($\boldsymbol{\pi}^E$), but can access the simulation of the environment.

3 RELATED WORK

3.1 Inverse reinforcement learning

Imitation learning is a framework for reproducing the expert's behavior from expert's trajectories. Imitation learning is mainly divided into two approaches [12]. First, behavioral cloning in which the expert's behavior is directly mimicked. Implementation of behavioral cloning is easy. However, if the number of the expert's trajectories is insufficient, imitating actions in states that are not included in the trajectory is dubious. Second, inverse reinforcement learning (IRL) in which reward is estimated to imitate the

expert's behavior. However, there are challenges such as ill-posed problem and high computation cost for inner-loop, which is internal calculations for policy of estimated reward using reinforcement learning [17].

IRL can be divided into reward learning for estimating the expert's reward and policy learning for estimating the expert's policy [12]. Reward learning is further divided into max-margin and max-entropy approaches. The former's objective is to maximize the margin between expert's policy and other policies [1, 11], and the latter's objective is to maximize the entropy constrain in matching expert with corresponding estimated behavior [19–21]. For policy learning, there is a generative adversarial approach based on the maximum-entropy approach [4].

3.2 Multi-agent inverse reinforcement learning

Table 1 shows the classification of multi-agent inverse reinforcement learning (MAIRL). MAIRL can be classified with respect to the structure of reward and the objective of approach that is mentioned in Section 3.1.

Table 1: Classification of multi-agent inverse reinforcement learning

Reward structure	Objective		
	Max-margin	Max-entropy	others
homogeneous	Šošić et al. [15]		
zero-sum			Lin et al. [7] Wang et al. [16]
others	Natarajan et al.[10] Reddy et al. [13]	Ziebart et al. [20] Bogert et al. [2] Song et al. [14]	Le et al.[6]

Šošić et al. [15] proposed a method for estimating homogeneous reward in swarm system, whereas Lin et al. [7] and Wang et al. [16] proposed MAIRL for a zero-sum game. However, these method [7, 15, 16] cannot be applied to a general-sum Markov game because it assumes a specific reward. MAIRL based on max-margin in general-sum Markov game [10, 13] is an extension of [11]. Natarajan et al. [10] is assumed that there is a common centralized policy for all agent. Reddy et al. [13] is assumed that there are decentralized policy for each agent. However, the former method [10] requires a transition probability while the latter method [13] requires Nash Q-learning for determining the Nash equilibrium in the inner-loop. MAIRL of Ziebart et al. [20] and Bogert et al. [2] based on maximum entropy. Ziebart et al. [20] formulated a maximum causal entropy IRL method and demonstrated the effectiveness of the proposed method in pursuit-evasion of three agents. Bogert et al. [2] proposed MAIRL for occlusion environment in which a part of the expert's trajectory is hidden. However, while the former method [20] is limited to the finite-horizon environment, the latter [2] requires a reward matrix for any state in which the agent interacts. Additionally, Le et al. [6] and Song et al. [14] presented a policy learning method to not explicitly estimate expert's reward.

Here, we assume reward function is represented by a linear sum of feature f vector and weight θ, to propose a novel MAIRL based on maximum discounted causal entropy. Unlike the method proposed in [2, 20], the proposed method does not require a reward matrix and can be applied to the infinite-horizon environment, and does not include the non-stationary problem in Markov game.

4 MULTI-AGENT MAXIMUM DISCOUNTED CAUSAL ENTROPY IRL

4.1 Formulation of the problem

In this section, we present a multi-agent maximum discounted causal entropy (M-MDCE) IRL problem and propose a novel MAIRL algorithm for solving a decomposed problem for each agent. M-MDCE IRL problem is an extension of maximum discounted causal entropy IRL [19] to Markov game. The M-MDCE IRL problem is represented as follows:

$$\max_{\boldsymbol{\pi}_t, t\geq 0} \sum_{n\in N} H_{\pi_{t,n},\pi^E_{-n}}(\pi_{t,n}) \tag{1}$$

$$\text{s.t.} \quad \overline{f}_{n,\boldsymbol{\pi}^E} = \overline{f}_{n,\pi_{t,n},\pi^E_{-n}} \quad \forall n \in \mathcal{N}, t \geq 0 \tag{2}$$

$$\pi_{t,n}(a_n|s) \geq 0 \quad \forall a_n \in \mathcal{A}_n, s \in \mathcal{S}, n \in \mathcal{N}, t \geq 0 \tag{3}$$

$$\sum_{a_n\in\mathcal{A}_n} \pi_{t,n}(a_n|s) = 1 \quad \forall s \in \mathcal{S}, n \in \mathcal{N}, t \geq 0 \tag{4}$$

$$\pi_{t,n}(a_n|s) = \pi_{t',n}(a_n|s) \quad \forall s \in \mathcal{S}, a_n \in \mathcal{A}_n, n \in \mathcal{N}, t, t' \geq 0 \tag{5}$$

where Eq. (1) denotes the maximization of the sum of causal entropy for each agent, defined in Eq. (6). Eq. (2) represents constraints of the expected feature vector matching defined in Eq. (7). Eq. (3), Eq. (4), and Eq. (5) represent constraints of the stationary policy. $f_n : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^k$ is the feature vector of agent n; and $\overline{f}_n$ is the expected feature vector.

$$H_{\pi_{t,n},\pi^E_{-n}}(\pi_{t,n}) = \mathbb{E}\left[\sum_{t=0}^{\infty} -\gamma^t \log \pi_{t,n}\left(A_{t,n}|S_t\right)\right] \tag{6}$$

$$\overline{f}_{n,\pi_n,\pi_{-n}} = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}\left[f_n\left(S_t, A_t\right)\right] \tag{7}$$

$$\boldsymbol{\pi}\left(A_t|S_t\right) = \prod_{n\in\mathcal{N}} \pi_n\left(A_{t,n}|S_t\right) \tag{8}$$

4.2 Decomposing the problem into a tractable problem for each agent

Herein, we consider two approaches for solving the M-MDCE problem. The first approach involves implementing a multi-agent reinforcement learning for the inner-loop procedure, a way for solving the M-MDCE problem in Markov game directly. However, multi-agent reinforcement learning requires dealing with non-stationary environment, thereby increasing the possibility that the learning does not converge.

The second approach involves decomposing the M-MDCE problem with respect to each agent, as a way of solving each agent's MDCE problem in a Markov decision process. The MDCE problem

for agent n is represented as follows:

$$\max_{\pi_{t,n}, t\geq 0} H_{\pi_{t,n},\pi^E_{-n}}(\pi_{t,n}) + \theta_n(\overline{f}_{n,\boldsymbol{\pi}^E} - \overline{f}_{n,\pi_n,\pi^E_{-n}})$$
$$\text{s.t.} \quad \pi_{t,n}(a_n|s) \geq 0 \quad \forall a_n \in \mathcal{A}_n, s \in \mathcal{S}, t \geq 0$$
$$\sum_{a_n\in\mathcal{A}_n} \pi_{t,n}(a_n|s) = 1 \quad \forall s \in \mathcal{S}, t \geq 0$$
$$\pi_{t,n}(a_n|s) = \pi_{t',n}(a_n|s) \quad \forall s \in \mathcal{S}, a_n \in \mathcal{A}_n, t, t' \geq 0$$

This problem is a decomposition of the Lagrangian relaxation of the original M-MDCE problem. Because the behavior of other agents is fixed by the expert policy, the state transition of each decomposed problem is stationary. However, the expert policy cannot be obtained simply from the assumption of Section 2.

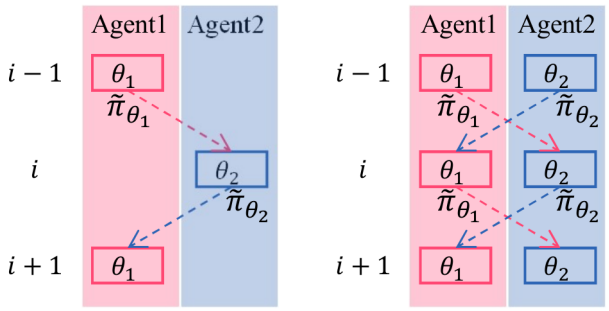
Hence, we propose a multi-agent maximum discounted causal entropy IRL, which can be used to decompose the M-MDCE problem into each agent's MDCE problem and replaces the expert policy π^E_{-n} with alternative policy $\tilde{\pi}_{-n}$. Algorithm 1 describes the M-MDCE IRL approach. First, in each iteration we select a set of agents $\tilde{\mathcal{N}}$ to update the reward's weight and the alternative policy. Next, the reward's weight θ_n is updated using MDCE IRL. Termination of reward's update is done when the difference between the expert's expected feature vector and the estimated one is very small, or the number of updates has reached the maximum value. Finally, we obtain the policy from the estimated reward's weight using Soft Q-Learning [19], and an alternative policy is calculated from the obtained policy using complement procedure. Complement procedure is to close the gap between expert's policy π^E_{-n} distribution and alternative policy $\tilde{\pi}_{-n}$ distribution. Details of complement procedure are described in subsection 4.2.2.

Algorithm 1 Multi-agent MDCE IRL

Require: Markov Game$\backslash\{R_n\}_{n\in\mathcal{N}}$
Require: Expert trajectories $\mathcal{D}$
Ensure: Reward weight $\{\theta_n\}_{n\in N}$
Initialize alternative policies $\{\tilde{\pi}_n\}_{n\in N}$ and $\{\theta_n\}_{n\in N}$
for iteration $= 1, 2 \ldots$ **do**
  $\tilde{\mathcal{N}} \leftarrow$ Selector($\mathcal{N}$)
  $\theta_n \leftarrow$ MDCE($\mathcal{D}^n, \tilde{\pi}_{-n}$) $\quad \forall n \in \tilde{\mathcal{N}}$
  $\pi_n \leftarrow$ SoftQ($\theta_n, \tilde{\pi}_{-n}$) $\quad \forall n \in \tilde{\mathcal{N}}$
  $\tilde{\pi}_n \leftarrow$ Complement($\pi_n, \mathcal{D}$) $\quad \forall n \in \tilde{\mathcal{N}}$

In the following subsection, we present a procedure selecting a set of agents and calculating the alternative policy.

4.2.1 Slecting set of agents. Herein, we discuss two kinds of selection methods referred to as cyclic and parallel. Figure 1 presents the flow of the procedure in case the number of agents is two. The cyclic procedure selects one agent with respect to its order in the iteration. The parallel procedure updates all agents in parallel during the iteration and regularly exchange the alternative policy between agents.



(a) Cyclic **(b) Parallel**

Figure 1: Procedure of the proposed method (the number of agents is two ($|\mathcal{N}| = 2$). $i-1, i, i+1$ is index of iteration)

4.2.2 Calculating alternative policy. In MDCE IRL, policy for reward is known to satisfy the following Soft Bellman equation:

$$Q^{\text{soft}}_\theta(s,a) = \theta^\top f(s,a) + \gamma \sum_{s'\in\mathcal{S}} T(s'|s,a) V^{\text{soft}}_\theta(s') \tag{9}$$
$$V^{\text{soft}}_\theta(s) = \text{softmax}_{a\in\mathcal{A}} Q^{\text{soft}}_\theta(s,a) \tag{10}$$
$$\pi(a|s) = \exp\left(Q^{\text{soft}}_\theta(s,a) - V^{\text{soft}}_\theta(s)\right) \tag{11}$$

where $\text{softmax}_{a\in\mathcal{A}} Q^{\text{soft}}_\theta(s,a)$ represents $\log \sum_{a\in\mathcal{A}} \exp\left(Q^{\text{soft}}_\theta(s,a)\right)$. Thus, an alternative policy can be learned using Soft Q-Learning, presented in Algorithm 2. Next, we apply a complement to the alternative policy, presented in Algorithm 3. Complement procedure modifies the alternative policy so that the probability of state and action that is included in expert trajectory is one. This procedure is intended to approximate the probability distribution of the expert policy.

Algorithm 2 Soft Q-Learning for agent n

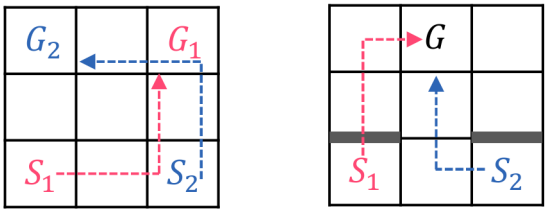
Require: Reward weight θ_n, Explore policy π, Alternative policy of other agent's $\tilde{\pi}_{-n}$
for $t = 0, 1, 2, \cdots$ **do**
  Generate sample (s_t, a_t, s_{t+1}) from $\pi, \tilde{\pi}_{-n}$
  $Q^{\text{soft}}_n(s_t, a_{t,n}) \leftarrow Q^{\text{soft}}_n(s_t, a_{t,n}) + \eta_t(s_t, a_{t,n}) \cdot$
    $\left[\theta_n^\top f_n(s_t, a_t) + \gamma V^{\text{soft}}_n(s_{t+1}) - Q^{\text{soft}}_n(s_t, a_{t,n})\right]$

Algorithm 3 Complement for agent n

Require: Policy π_n, Expert trajectories $\mathcal{D}$
Ensure: Alternative policy $\tilde{\pi}_n$
$\tilde{\pi}_n \leftarrow \pi_n$
for trajectory $\in \mathcal{D}$ **do**
  for $s, a \in$ trajectory **do**
    $\tilde{\pi}_n(a'|s) \leftarrow \begin{cases} 1 & (a' = a) \\ 0 & (otherwise) \end{cases}$

5 EXPERIMENTS

5.1 Experiments setting



(a) Deterministic transition environment (GW1) **(b) Probabilistic transition environment (GW2)**

Figure 2: Environments and expert's trajectories (S_1 and S_2 are the start coordinates for agent 1 and 2, respectively. G_1 and G_2 are goal coordinates while the bold line in GW2 represents barriers)

Figure 2 shows two experimental environments. Each environment is a 3×3 grid world, and the objective is that the two agents move from the start (S_1, S_2) to goal G along the shortest path. The set of states is a combination of the agent's coordinates while the set of actions of each agent is $A_1 = A_2 = \{\text{up}, \text{down}, \text{right}, \text{left}\}$. An agent can move to adjacent cells in any of the four directions with one step. If an agent moves on the wall or to the same cell as another agent, except when either agent moves toward the goal coordinate, the agent remains in the same coordinate where it was before the transition. In GW2 environment, there are barriers that can interfere with the transition from the start coordinate (S_1, S_2) to the coordinates adjacent to the north with $1/2$ probability. The states from which either agent reaches the goal is regarded as the absorbing state.

The expert trajectory of each environment represents a Nash equilibrium. Arrows in Figure 2 represent the expert trajectory. The experts take deterministic action in a coordinate along the arrow. This expert's trajectory is Nash equilibrium, the agent obtains +100 reward on reaching the goal, and -1 reward on moving in the same cell [5].

A feature vector is a binary vector for all state-action pairs. The expected feature vector can be calculated using Eq. (7) from 100 sampling trajectories, the maximum length of each trajectory is 50 steps. The initial reward's weight and policy is 0 vector and uniform distribution, respectively. The maximum number of updating with MDCE IRL and episodes of Soft Q-Learning is 100, respectively. The step size of MDCE IRL is fixed at 0.1. To evaluate the effect of L2 regularization at updating the reward's weight, we experiment with the two kinds of the weight of L2 regularization term, i.e., 0 and 0.01.

5.2 Experiments result and discussion

In the experiment, the estimation error of 10 trials is used to evaluate the differences of the expert's behavior and estimation results. The estimation error is the norm of differences between the expert's behavior and the estimated expected feature vector. If the estimated policy matches the expert's policy, then the estimation error is zero.

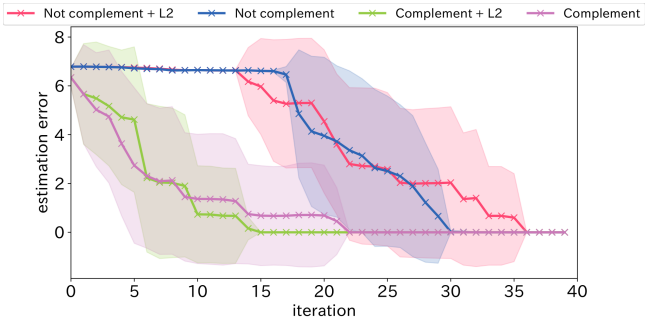


Figure 3: Cyclic procedure on GW1

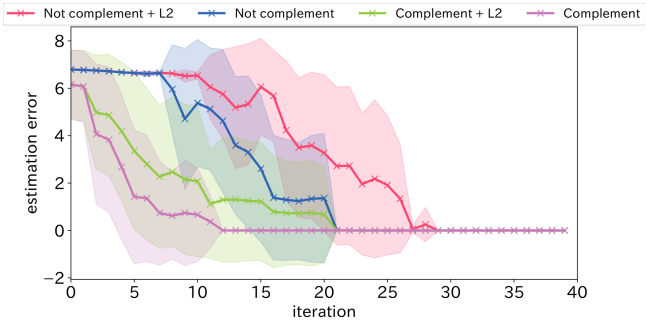


Figure 4: Parallel procedure on GW1

Figure 3 and 4 show changes in the average and standard deviation value of the estimation error in GW1 using cyclic and parallel procedures, respectively. The horizontal axis represents the iteration, whereas the vertical axis represents the estimation error. The label including L2 in Figure 3 and 4 is the result of using L2 regularization. From these results on the environment with a deterministic transition, the proposed method can estimate rewards that match the expert's behavior policy. Further, the complement procedure reduces the number of iteration until convergence and parallel procedure converge quickly than the cyclic procedure.

The reason that complement procedure is most effective is probably because imitating the expert behavior is easy. For example, consider the case of estimating reward of agent 1 in GW1. If the alternative policy of agent 2 is uniform distribution and agent 1 selects action of moving right in initial state, the probability of transition to the same state as expert is 1/4. In this case, the expected feature vector does not match the expert's vector for any policy of agent 1. However, if the alternative policy is calculated using complement procedure, the agents do not conflict. Therefore, it indicates that the complement procedure facilitates the estimation of reward. The parallel procedure is effective because the number of updating reward is two times the updating reward obtained using the cyclic procedure.

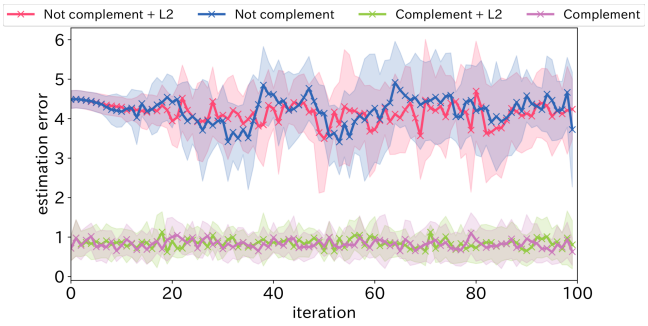


Figure 5: Cyclic procedure on GW2

Figure 5 shows the result obtained when the environment was performed in an environment that includes the probabilistic transition of GW2 using cyclic procedure. In this environment, even if complement is used, the estimation error with accurately zero reward was not estimated. It produced the same results as cyclic procedure irrespective of whether a parallel procedure is used or not. However, the estimated rewards at least reflect the expert's purpose. Figure 6 shows five state-action pairs with the highest estimated reward value. Because four state-action pairs of the proposed method are included in the expert's trajectory, it can be said that the proposed method can be used to estimate valid rewards. Therefore, the estimation error is not zero because sampling distribution is influenced by a probabilistic transition.

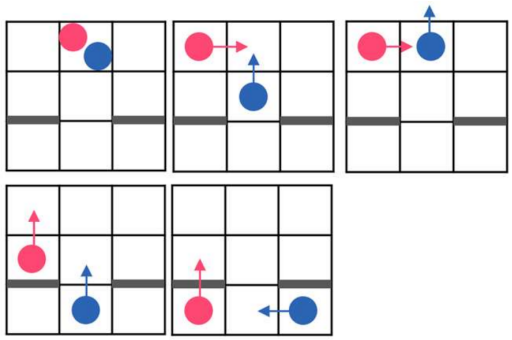


Figure 6: State-action pairs of the largest value of the estimated reward (GW2+Cyclic)

6 CONCLUSION AND FEATURE WORK

We proposed a new MAIRL approach that can be used to design an agent's action rules and to understand the purpose of the agent in the multi-agent simulation. Particularly, we formulated MAIRL problem based on maximum discounted causal entropy in infinite-horizon problem of Markov game. Further, we proposed M-MDCE IRL that can decompose the overall problem into tractable problems in an environment of a single agent. The experiments conducted on simple 3×3 grid world environment show that the proposed method can estimate rewards from deterministic expert's trajectory.

There are three future challenges to this research. The first is to apply the proposed method to different environments such as a zero-sum game. The second is to apply it to an application in a continuous state space, to validate the proposed method scalability. The third challenge is to identify the equilibrium point that can be acquired using the proposed method.

REFERENCES

[1] Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML '04)*. 1–.

[2] Kenneth Bogert and Prashant Doshi. 2018. Multi-robot inverse reinforcement learning under occlusion with estimation of state transitions. *Artificial Intelligence* 263 (2018), 46–73.

[3] Arnaud Doniec, René Mandiau, Sylvain Piechowiak, and Stéphane Espié. 2008. A behavioral multi-agent model for road traffic simulation. *Engineering Applications of Artificial Intelligence* 21, 8 (2008), 1443 – 1454.

[4] Jonathan Ho and Stefano Ermon. 2016. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems 29*. 4565–4573.

[5] Junling Hu and Michael P. Wellman. 2003. Nash Q-learning for General-sum Stochastic Games. *J. Mach. Learn. Res.* 4 (2003), 1039–1069.

[6] Hoang M. Le, Yisong Yue, Peter Carr, and Patrick Lucey. 2017. Coordinated Multi-agent Imitation Learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70 (ICML'17)*. 1995–2003.

[7] Xiaomin Lin, Peter A. Beling, and Randy Cogill. 2018. Multiagent Inverse Reinforcement Learning for Two-Person Zero-Sum Games. *IEEE Transactions on Games* 10, 1 (2018), 56–68.

[8] Michael L. Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*. 157 – 163.

[9] Keiichi Namikoshi and Sachiyo Arai. 2018. Estimation of the Heterogeneous Strategies from Action Log. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '18)*. 1310–1317.

[10] Sriraam Natarajan, Gautam Kunapuli, Kshitij Judah, Prasad Tadepalli, Kristian Kersting, and Jude Shavlik. 2010. Multi-Agent Inverse Reinforcement Learning. In *2010 Ninth International Conference on Machine Learning and Applications*. 395–400.

[11] Andrew Y. Ng and Stuart J. Russell. 2000. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00)*. 663–670.

[12] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. 2018. An Algorithmic Perspective on Imitation Learning. *Foundations and Trends® in Robotics* 7, 1-2 (2018), 1–179.

[13] Tummalapalli Sudhamsh Reddy, Vamsikrishna Gopikrishna, Gergely Zaruba, and Manfred Huber. 2012. Inverse reinforcement learning for decentralized non-cooperative multiagent systems. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 1930–1935.

[14] Jiaming Song, Hongyu Ren, Dorsa Sadigh, and Stefano Ermon. 2018. Multi-Agent Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems 31*. 7461–7472.

[15] Adrian Šošić, Wasiur R KhudaBukhsh, Abdelhak M Zoubir, and Heinz Koeppl. 2017. Inverse Reinforcement Learning in Swarm Systems. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. 1413–1421.

[16] Xingyu Wang and Diego Klabjan. 2018. Competitive Multi-agent Inverse Reinforcement Learning with Sub-optimal Demonstrations. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Vol. 80. 5143–5151.

[17] Shao Zhifei and Er Meng Joo. 2012. A review of inverse reinforcement learning theory and recent advances. In *2012 IEEE Congress on Evolutionary Computation*. 1–8.

[18] Suiping Zhou, Dan Chen, Wentong Cai, Linbo Luo, Malcolm Yoke-Hean Low, Feng Tian, Victor Su-Han Tay, Darren Wee Sze Ong, and Benjamin D. Hamilton. 2010. Crowd modeling and simulation technologies. *ACM Trans. Model. Comput. Simul.* 20 (2010), 20:1–20:35.

[19] Zhengyuan Zhou, Michael Bloem, and Nicholas Bambos. 2018. Infinite Time Horizon Maximum Causal Entropy Inverse Reinforcement Learning. *IEEE Trans. Automat. Control* 63, 9 (2018), 2787–2802.

[20] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. 2010. Modeling Interaction via the Principle of Maximum Causal Entropy. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. 1255–1262.

[21] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. 2008. Maximum Entropy Inverse Reinforcement Learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3 (AAAI'08)*. 1433–1438.