Designing Non-greedy Reinforcement Learning Agents with Diminishing Reward Shaping

Fan-Yun Sun, Yen-Yu Chang, Yueh-Hua Wu, and **Shou-De Lin**
Department of Eletrical Engineering and Computer Science, National Taiwan University
{b04902045, b03901138, d06922005}@ntu.edu.tw, sdlin@csie.ntu.edu.tw

Abstract

This paper intends to address an issue in RL that when agents possessing varying capabilities, most resources may be acquired by stronger agents, leaving the weaker ones "starving". We introduce a simple method to train non-greedy agents in multi-agent reinforcement learning scenarios with nearly no extra cost. Our model can achieve the following goals in designing the non-greedy agent: *non-homogeneous equality*, *only need local information*, *cost-effective*, *generalizable and configurable*. We propose the idea of diminishing reward that makes the agent feel less satisfied for consecutive rewards obtained. This idea allows the agents to behave less greedy without the need to explicitly coding any ethical pattern nor monitor other agents' status. Given our framework, resources can be distributed more equally without running the risk of reaching homogeneous equality. We designed two games, *Gathering Game* and *Hunter Prey* to evaluate the quality of the model.

1 Introduction

Multi-agent reinforcement learning normally assumes that all agents are equally capable, which seems to be unrealistic as in real applications it is not guaranteed all the agents are designed under the same framework nor trained using the same data. Given agents with different levels of capability, one serious concern is that the stronger agents may dominate too many resources for the weaker ones to survive. For instance, considering heavy traffic with autonomous cars, the ones with faster decision making process and better acceleration devices may overtake the others and make the less-capable vehicle stuck or barely move in this situation. Instead of designing a specific protocol to solve the problem, we aim to provide a low-cost and general solution to alleviate the harm to the weaker agents in such competitive environments.

The most relevant work might be the one proposed by (Lerer and Peysakhovich 2017) which modifies the reward function for each agent as the sum of the rewards from all agents. In this case, the agents can optimize the global reward instead of its individual rewards to avoid damaging the overall rewards through acting greedily. However, to deploy such solution, one needs to assume the individual rewards are visible to all agents, which might be very costly to implement in practice. Our design aims at making the stronger agents less greedy given sufficient resources (or rewards) are available, thus the chance of the 'starving weaker agents' becomes smaller. It does not demand each agent to be omniscience, as only local reward information is needed to realize this strategy. In addition to that, we also point out that we need to avoid homogeneous equality, meaning that the resources are equally distributed across agents regardless of their capability. Our goal is to ensure the stronger agents still have higher chance to earn more resources, while at the same time be less greedy to leave some resources for the weaker ones. Below we summarize our design criteria:

- *Non-greedy*: stronger agents try not to grasp as much resources as they can and leave some for the weaker ones, given that resources are sufficient.
- *Non-homogeneous equality*: Stronger agents can still obtain more resources than the weaker ones.
- *No global observability required*: Agents do not have to know other agents' internal information, such as the rewards received.
- *Cost-effective*: simple and computationally efficient.
- *Universal and configurable*: Applicable to most reinforcement learning models, and the levels of non-greediness can be easily configured.

Our solution integrates the idea of *Diminishing Marginal Utility* in economics into a reward shaping from (Ng, Harada, and Russell). *Diminishing Marginal Utility* states that the marginal utility of each homogenous unit decreases as the supply of units increases and vice versa. Reward shaping is a method that aims to accelerate the convergence rate by transforming agents' reward function.

The proposed method works as below. Given the normal design of a reinforcement learning (RL) agent, apply an reward transformation function to the regular reward function. The adjusted rewards acts as the *mental reward* that the agents believed it has obtained. It takes the output of the original reward function and the historic information of the reward received by the agent as an input, and output the diminished reward as the mental reward. Note that the adjustment function (as shown in Figure 1)can be easily generalized without having to design a customized function for ev-

ery scenarios. The intuition is simple: our diminishing function allows the agent to feel that 'less reward' can be earned for consecutive success. Thus it would be less attractive for agents to pursue additional rewards within a short period, which leads to the non-greedy behavior.

Finally, we conducted experiments on two simulated scenarios *Gathering Game*, *Hunter-Prey* to verify whether the trained non-greedy agents indeed can satisfy the above design criteria. We have released the source code here: https://github.com/petwill/diminishing-reward-shaping.

2 Preliminaries

2.1 Reinforcement Learning

Reinforcement learning (RL) is an effective approach to solve real-world decision-making problems, which is usually represented by a Markov Decision Process (MDP). The MDP is parameterized by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ is a set of actions, $\mathcal{T}$ is a transition function $T(s, a, s')$ indicating the transition probability of $P(s'|s, a)$, $\mathcal{R}$ is the reward function that returns a number to encourage/prohibit action a under state s, and we use discount factor $\gamma \in [0, 1)$ to represent different influences of long-term and short-term reward. To derive a policy $\pi(a|s)$ that optimize the discount cumulative reward $\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)$, an elaborate way is to utilize SARSA (Rummery and Niranjan 1994), where an estimator of expected future discounted reward $Q(s, a)$ is obtained based on the concept of Bellman's equation (Watkins and Dayan 1992):

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)] \quad (1)$$

where $\alpha \in [0, 1)$ is the learning rate. The optimal policy indicated by SARSA is to select the state-action pair (s, a) that has the maximal Q-value $Q(s, a)$.

Deep Q-learning Network (Mnih et al. 2015) has been a very popular model in reinforcement learning for Q-value approximation. At every iteration t, the model use the experience tuple (s, a, r, s') from replay buffer to update DQN model parameters θ to minimize the loss $L_t(\theta_t) = \mathbb{E}_{(s,a,r,s')}[(r + \gamma \max_{a'} Q(s', a', \hat{\theta}_t) - Q(s, a, \theta_t))^2)]$. The replay buffer is a queue only containing the set of the latest experience tuple, and the $\hat{\theta}_t$ is the parameters of the target network at time t.

2.2 Multi-agent Reinforcement Learning

Multi-agent reinforcement learning (MARL) is related to a multi-agent system with several autonomous, interacting agents sharing a common environment and each agent have to maximize their individual rewards (Busoniu, Babuška, and De Schutter 2010).

The notation of a multi-agent system with n agents is represented as a tuple $(n, \mathcal{S}, \mathcal{A}^1, ..., \mathcal{A}^n, \mathcal{R}^1, ..., \mathcal{R}^n, \mathcal{P})$ where $\mathcal{S}$ is the set of states, $\mathcal{A}^i$ is the action set of agent i, $\mathcal{R}^i$ is the reward of agent i and $\mathcal{P}$ is the state transition function. The reward function $\mathcal{R}^i : \mathcal{S} \times \mathcal{A}^1 \times \cdot \times \mathcal{A}^n \mapsto \mathbb{R}$ maps the joint action $a = (a^1, ..., a^n)$ to an immediate reward to agent i. The transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A}^1 \times \cdot \times \mathcal{A}^n \mapsto \Delta(\mathcal{S})$ determines the probabilistic state change to the next state s_{t+1}.

Q-learning algorithm can applied to multi-agent system by making each agent i learn an independently optimal Q-value function with its own Q-network. However, there are some challenges in MARL (Matignon, Laurent, and Le Fort-Piat 2012; Busoniu, Babuška, and De Schutter 2010). Q-learning algorithm estimates values of every possible discrete state and state-action pair, which can lead to exponential computational complexity. The case also applies in every agent since each agent has its own variables to the joint environment. This makes the suffering of dimensionality of MARL even severe than single-agent scenario. Also, non-stationarity is an issue in multi-agent system since all the agents are learning simultaneously under the same environment. Thus, each agent faces a moving-target learning problem: the optimal policy changes if other agents' policies change.

2.3 Reward Shaping

Most value-based reinforcement learning algorithms are rather slow because they need to explore state-action pairs uniformly at random in the early stage. Only going through enough explorations and then updated by associated rewards have been observed can the agent start to exploit the experience by biasing its action selection towards what it estimates to be good.

Reward shaping (Ng, Harada, and Russell), motivated from behavioral psychology , is an efficient way of including prior knowledge in the learning problems so as to enhance the convergence rate. Additional intermediate rewards are provided to enrich a sparse base reward signal, giving the agent with useful gradient information. Reward shaping can be easily incorporated with a variety of resources such as demonstration and verbal feedback. The shaping reward $\mathcal{H}$ is usually integrated with the original reward in the form of addition:

$$\mathcal{R}^{'}(s_t, a_t, s_{t+1}) = \mathcal{R}(s_t, a_t, s_{t+1}) + \mathcal{H}(s_t, a_t, s_{t+1}). \quad (2)$$

To ensure policy invariance, potential-based shaping method is proposed. Rewards are modified to the following form

$$\mathcal{R}^{'}(s_t, a_t, s_{t+1}) = \mathcal{R}(s_t, a_t) - \Phi(s_t) + \gamma\Phi(s_{t+1}). \quad (3)$$

The potential functions Φ encodes how good it is to be in a state. The relation of the value functions before and after the transformation is just a constant shift.

$$\mathcal{Q}^{'}(s_t, a_t) = \mathcal{Q}(s_t, a_t) - \Phi(s_t). \quad (4)$$

3 Diminishing Reward Shaping

Here we adopt the idea of diminishing reward shaping in training an RL agent. That is, each agent maintains a *mental-reward* that is different from the actual rewards obtained. To determine the mental-reward, the agent has to maintain the information $\mathcal{I}_t$ about the total reward received in the past $\mathcal{W}$ time frames.

Thus, we add one more variable to the state representation $s_t^{'}$ as:

$$s_t^{'} = s_t \cup \mathcal{I}_t. \quad (5)$$

Note that all agents observe the same state s_t, but $\mathcal{I}_t$ is agent-specific information and will not be shared with any other agents. With the adjustment function $\mathcal{F}$ (Figure 1) and the original reward function $\mathcal{R}$, the shaped reward function $\mathcal{R}'$ becomes

$$\mathcal{R}'(s_t, a_t, \mathcal{I}_t) = \mathcal{R}(s_t, a_t) \times \mathcal{F}(\mathcal{I}_t) \tag{6}$$

$$\mathcal{F}(\mathcal{I}_t) = \sum_{i=1}^{\mathcal{W}} \mathcal{R}(s_{t-i}, a_{t-i}) \tag{7}$$

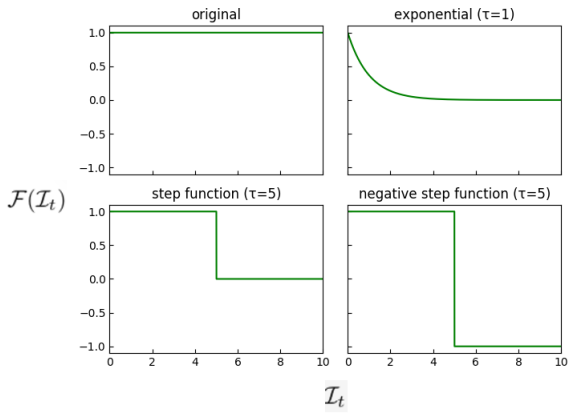

Figure 1: The possible form of the adjustment function $\mathcal{F}$. Exponential denotes the function $y = e^{\frac{-x}{\tau}}$.

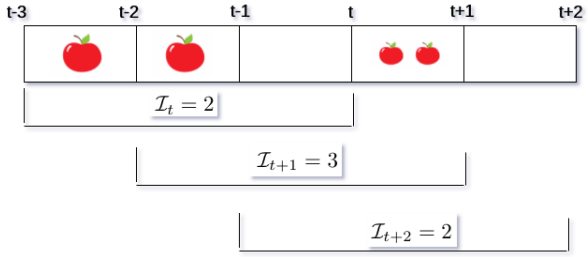

Figure 2: An illustration of our method with $\mathcal{W} = 3$.

Note that the major difference between our shaping function and the original reward shaping is that the rewards are adjusted through multiplication, instead of addition. As will be shown in our second experiments, diminishing reward shaping can be combined with the original shaping-by-addition to yield a more satisfiable outcome. Finally, reward transformation shall be applied to all agents, regardless whether they are strong or weak. However, weaker agents are less affected by the shaping since they probably do not obtain reward frequently.

4 Experiments

4.1 Game Setup

We conducted two map-wandering experiments to evaluate the proposed model. Both experiments are implemented in a 2D grid-world. Agents can observe the whole map, including all other agents' positions. However, the accumulated rewards for each agent is not known to others. All agents start the game from corner grids. There are five agent-centered actions: step forward, step backward, step left, step right, and stand still. Each episode lasts for 1000 rounds, and in each round (time frame), faster agents can make more moves than slower ones, which is where the game introduces inequality. In a two-player game, stronger agents can move two steps at a time while the weaker agent can move only one step. Likewise, the strongest agent can move three consecutive steps while the weakest one can only move one step in a three-player game. Below we describe the design of the games.

Gathering Game This game is modified from the *Gathering Game* mentioned in (Leibo et al. 2017). In this game, several agents move on a 5×5 grids table. Apples may only appear in some central grids to increase interaction between agents. Apples re-spawn after N_{apple} time frames, which is 5 by default. When an agent collects an apple, one reward is given. $\mathcal{W}$ is set to 5 in *Gathering Game*.

Hunter Prey This is a popular game in reinforcement learning scenarios (Tan 1993; Doya et al. 2002). Basically, multiple agents, also known as hunters, chase over a single prey that could appear anywhere on the map. The prey reoccurs only when the previous one is caught. To accelerate the training procedure, state-based reward shaping is applied, which allows us to reward the approaching behavior and punish the opposite. Let the position of the prey and the agent be (x_{prey}, y_{prey}) and (x_t, y_t), we have the following equation.

$$\mathcal{R}'(s_t, a_t, \mathcal{I}_t) = (\mathcal{R}(s_t, a_t) - \Phi(s_t) + \gamma\Phi(s_{t+1})) \times \mathcal{F}(\mathcal{I}_t) \tag{8}$$

$$\Phi(s_t) = \frac{1}{\mid x_{prey} - x_t \mid + \mid y_{prey} - y_t \mid} \tag{9}$$

This is an example to show how the diminishing reward can be combined with the original reward shaping function. $\mathcal{W}$ is set to 10 in *Hunter Prey*.

Simulation The i-th agent stores a function $Q_i : O_i \times A_i \to \mathbb{R}$ represented by deep Q-network (DQN). Note that each learning agent is "independent" of other agents and regards other agents as part of the environment. The state observed by each agent is different, denoted as $o_i = O(s, i)$. For convenience, we use $Q_i(s, a) = Q_i(O(s, i), a)$. During the training, epsilon-greedy policy is used for exploration, with epsilon ϵ decaying linearly over time (from 1 to 0.1). The i-th agent's policy has the following form.

$$\pi_i(s) = \begin{cases} \arg\max_{a \in A_i} Q_i(s, a) \text{ with probability 1 - } \epsilon \\ \mathcal{U}(A_i) \text{ with probabilty } \epsilon \end{cases} \tag{10}$$

$\mathcal{U}(A_i)$ denotes a sample from the uniform distribution over $\mathcal{A}_i$. The neural networks have two hidden layers with 32 units, interleaved with a rectified linear layer that project to the output layer with 5 units, one for each action. The default per-time-step discount rate is 0.99.

Evaluation Metrics Note that here we would like to evaluate how many resources are obtained (the actual reward), not the mental-reward of each agent. Since we want to achieve non-greedy in a global sense. We adopt entropy as our measure of equality. Let A_{total} denote the sum of resources obtained by all agents and A_i be the total reward agent i received in a single game. Entropy is calculated as follow

$$-\sum_i \frac{A_i}{A_{total}} \times \log_2(\frac{A_i}{A_{total}}). \tag{11}$$

However, measuring only entropy can be deceiving. If the goal is to simply maximize the entropy, a trivial optimal solution can be achieved by letting each agent take equal number of resources. It is not desirable since every agent should still attempt to maximize its own reward sum. As a result, resource consumption rate has to be evaluated as well. The total resources obtained by all agents is taken into consideration as the second metric. Since resources only re-spawn after the previous one is consumed, the sum of resources A_{total} after diminishing reward shaping is applied tends to be lower.

4.2 Results

Non-greedy and Configurability Three shaping function $\mathcal{F}$ below are adopted in our experiments.

- Exponential

$$\mathcal{R}^{'}(s_t, a_t, \mathcal{I}_t) = \mathcal{R}(s_t, a_t) \times e^{-\frac{\mathcal{I}_t}{\tau}} \tag{12}$$

- Step function

$$\mathcal{R}^{'}(s_t, a_t, \mathcal{I}_t) = \begin{cases} \mathcal{R}(s_t, a_t) & x \leq \tau \\ 0 & \tau \leq x \end{cases} \tag{13}$$

- Negative step function

$$\mathcal{R}^{'}(s_t, a_t, \mathcal{I}_t) = \begin{cases} \mathcal{R}(s_t, a_t) & x \leq \tau \\ -1 & \tau \leq x \end{cases} \tag{14}$$

Experiment results are shown in Figure 3. We notice significant improvement on the resources obtained by the weaker agent (and hence higher entropy). Note that except the exponential diminishing function, increasing τ can generally make stronger agent less greedy. Exponential diminishing seems to be not as configurable as others. It's probably because adjusting τ in such function is less straightforward.To sum up, the step function is recommended over other functions since it can tolerate a larger range of τ value.

Non-homogeneous Equality As mentioned previously, it is undesirable to achieve homogeneous equality that resources are equally distributed. To show that our method achieves non-homogeneous equality, the difference between agents' capabilities are reduced but preserved. In order to show that, we conducted the following experiments (Figure 5). It is shown that the domination is alleviated only when resources are abundant (e.g. 1/5). That is, stronger agents only act non-greedily when it is "satisfied". On the other hand, when the resources are scarce (e.g. spawn rate 1/20), the diminishing reward cannot significantly alter the behavior of the agent as the stronger ones still have higher chance to obtain the resources, which is important in our design.

Extending to 3 agents To show that our method can be easily generalized, experiments to environments with three agents are conducted (see Figure 4). Similar results are obtained.

5 Related Work

Machine ethics (Anderson and Anderson 2011) is a field that dedicate to make machines follow ethical principles, or to resolve ethical dilemmas. Some works propose frameworks with machine learning and reinforcement learning to make ethical decision (Abel, MacGlashan, and Littman 2016; Arnold, Kasenberg, and Scheutz 2017). Since real-world scenarios involve multiple agents, many research have been conducted on multi-agent systems to better approximate interactions in our daily life. The natural interaction between multiple agents can be cooperative or competitive. Many studies focus on how to make agents behave in cooperative manners through improving learning algorithms (Lauer and Riedmiller 2000; Matignon, Laurent, and Le Fort-Piat 2007) and model design (Gupta, Egorov, and Kochenderfer 2017). (Leibo et al. 2017; Lerer and Peysakhovich 2017) perform detailed observation about whether agents behave cooperatively or greedily under a variety of social dilemmas, and our work provides a solution to achieve fair resource distribution under similar situations. The followings are previous works that aim to design cooperative multi-agent systems and multi-agent RL with social dilemmas.

5.1 Machine Ethics

Making artificial agents learn about ethical objective function while making decisions is challenging. (Armstrong 2015) deals with the problem by utilizing Bayesian learning to update the true ethical objective function. (Abel, MacGlashan, and Littman 2016) adapts the same concept on utility function, which considers the ethical learning problem as learning an utility function as the part of hidden states of *Partially Observable Markov Desicion Process* (POMDP).

(Russell, Dewey, and Tegmark 2015) suggests that *Inverse Reinforcement Learning* (IRL) is possibly viable in machine ethic issues since a system infers preferences of other rational or nearly rational actors by observing their actions. Instead of formulating rules, laws or utility functions in the beginning of the training process, the system could learn from modeled what other agents trying to do and what behaviors are being observed. However, (Arnold, Kasenberg, and Scheutz 2017) claims that systems that only applies IRL is insufficient for agents to infer norms. Consequently, they propose an approach to combine RL and logical representations: the agents maximize their reward function over the state-action pairs and meanwhile maximally satisfy the norms.

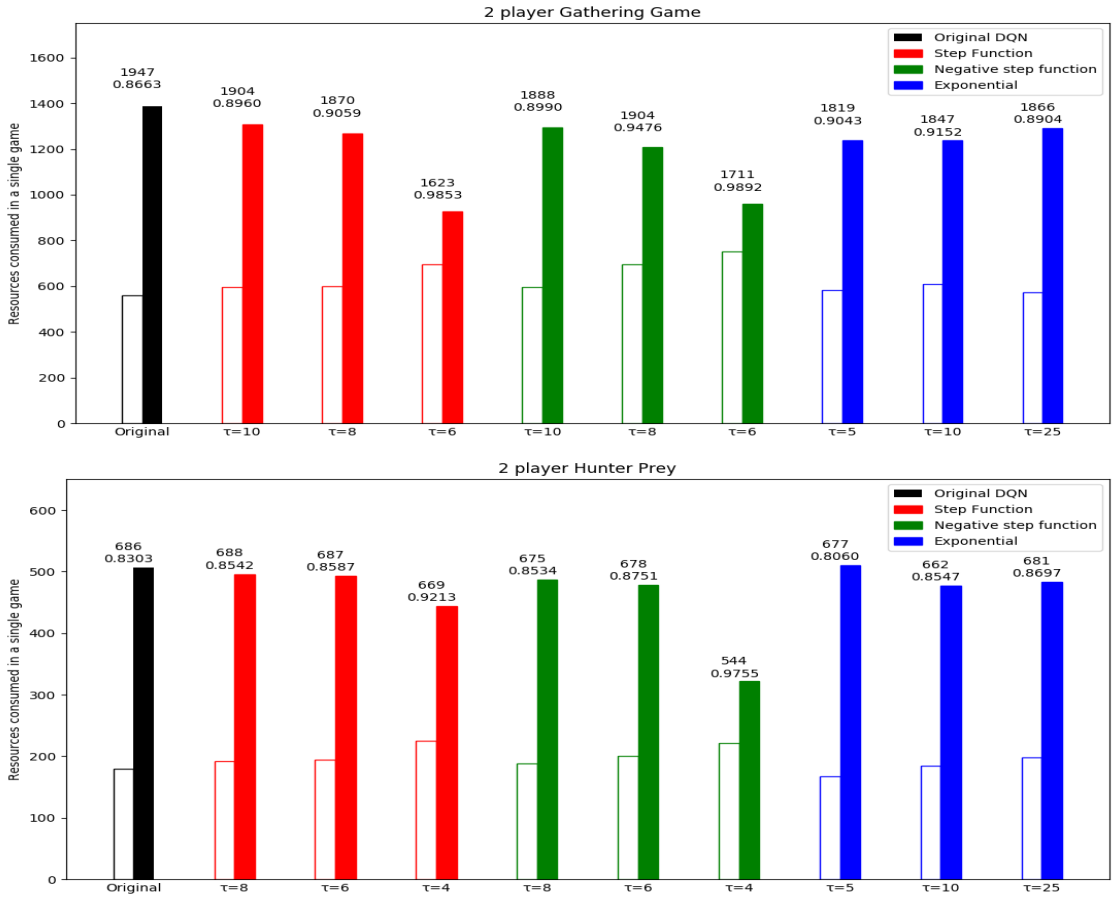

Figure 3: Experiment results for 2-player *Gathering Game* and *Hunter Prey*. For two near-by bars, the left one denotes the weaker agent while the one on the right is the stronger agent. Different functions applied are distinguished by color.

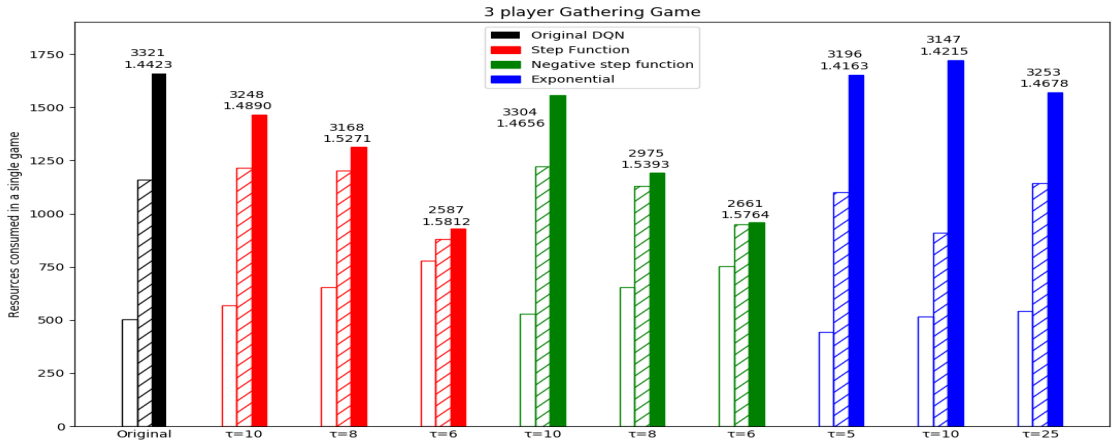

Figure 4: Experiment results for 3-player *Gathering Game*.

5.2 Cooperative Multi-agent System

In multi-agent systems, a agent's policy depends on the actions chosen by the team. Thus, an agent would be punished even if it makes the optimal decision since bad decisions are made by teammates. In (Lauer and Riedmiller 2000), they design *Distributed Q-Learning* algorithm to make optimal agents neglect the penalties from non-coordinate agents in the update procedure. However, an issue is that the op-

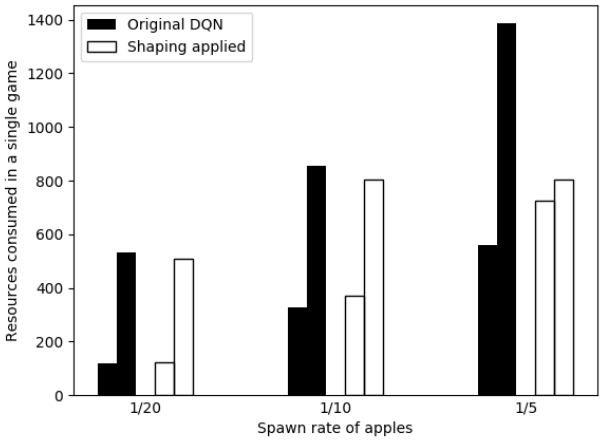

Figure 5: Results on a two-player *Gathering Game* with different resource spawn rate. Lower spawn rate means that resources are more scarce. The shaping function used in this experiment is a step function with $\tau = 6$. For every two adjacent bars, the left one denotes resources consumed by the stronger agent while the right bar represented the weaker agent.

timistic agents do not manage to achieve the coordination between multiple joint optimal actions. (Matignon, Laurent, and Le Fort-Piat 2007) designs the *Hysteretic Q-learning* algorithm to solve the coordination issue. Both approaches make agents to improve collective reward.

In (Gupta, Egorov, and Kochenderfer 2017), they focus on the *Parameter Sharing* method. The approach indirectly achieves cooperation behavior through sharing policy parameters. However, this approach requires agents to be homogeneous and it is hard to implement in real-world scenarios.

5.3 Multi-agent Social Dilemma

In social dilemmas, individuals tempt to increase their payoffs in the short run at a cost to the long run total welfare. Several algorithms and analyses have been developed for the two-player zero-sum case (Littman 1994), but the general-sum case is significantly challenging (Zinkevich, Greenwald, and Littman 2006). Most algorithms require to track several different potential equilibrium for each agent (Hu, Wellman, and others 1998; Greenwald, Hall, and Serrano 2003), or to pose restrictions to agents to simplify the problem (Littman 2001). (Leibo et al. 2017) aims to answer "what social effects emerge when each agent uses a particular learning rule?". Their purpose is to study and characterize the resulting learning dynamics, rather than designing new learning algorithms. Analysis is also studied on the dynamics of policies learned by multiple self-interested independent learning agents using its own deep Q network. They also characterize how the learned behavior in each domain changes as a function of environmental factors.

(Lerer and Peysakhovich 2017) proposes a method using modern reinforcement learning to generalize successful strategy in Prisoner's Dilemma: tit-for-tat. The learning agents get rewards from both their own payoff and the rewards other agents receive. They both show that agents can maintain cooperation in complex environments. However, it only works when the zero-sum games is considered and fails in positive-sum interaction. Moreover, the reward for the learning agent does not make much sense in real-world situations because getting other agents' information costs too much.

6 Conclusion

One key difference between our diminishing and reward shaping is that the regular reward shaping is proposed to improve the convergence rate, while diminishing reward is proposed to make sure reinforcement learners not only maximize their individual rewards but also behave in a non-greedy manner. Our method involves using a sliding window to collect historic information of rewards and this can be adopted to most reinforcement algorithms.

Future works include finding a better theoretical justification for the convergence of the model, as well as providing more empirical justification for a variety of diminishing functions.

References

Abel, D.; MacGlashan, J.; and Littman, M. L. 2016. Reinforcement learning as a framework for ethical decision making. In *AAAI Workshop: AI, Ethics, and Society*.

Anderson, M., and Anderson, S. L. 2011. *Machine ethics*. Cambridge University Press.

Armstrong, S. 2015. Motivated value selection for artificial agents. In *AAAI Workshop: AI and Ethics*.

Arnold, T.; Kasenberg, D.; and Scheutz, M. 2017. Value alignment or misalignment–what will keep systems accountable?

Busoniu, L.; Babuška, R.; and De Schutter, B. 2010. Multi-agent reinforcement learning: An overview. *Innovations in multi-agent systems and applications-1* 310:183–221.

Doya, K.; Samejima, K.; Katagiri, K.-i.; and Kawato, M. 2002. Multiple model-based reinforcement learning. *Neural computation* 14(6):1347–1369.

Greenwald, A.; Hall, K.; and Serrano, R. 2003. Correlated q-learning. In *ICML*, volume 3, 242–249.

Gupta, J. K.; Egorov, M.; and Kochenderfer, M. 2017. Cooperative multiagent control using deep reinforcement learning. In *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS 2017)*.

Hu, J.; Wellman, M. P.; et al. 1998. Multiagent reinforcement learning: theoretical framework and an algorithm. In *ICML*, volume 98, 242–250.

Lauer, M., and Riedmiller, M. 2000. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *In Proceedings of the Seventeenth International Conference on Machine Learning*. Citeseer.

Leibo, J. Z.; Zambaldi, V.; Lanctot, M.; Marecki, J.; and Graepel, T. 2017. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, 464–473. International Foundation for Autonomous Agents and Multiagent Systems.

Lerer, A., and Peysakhovich, A. 2017. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *arXiv preprint arXiv:1707.01068*.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning*, volume 157, 157–163.

Littman, M. L. 2001. Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, 322–328.

Matignon, L.; Laurent, G. J.; and Le Fort-Piat, N. 2007. Hysteretic q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 64–69. IEEE.

Matignon, L.; Laurent, G. J.; and Le Fort-Piat, N. 2012. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27(1):1–31.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Ng, A. Y.; Harada, D.; and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping.

Rummery, G. A., and Niranjan, M. 1994. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering.

Russell, S.; Dewey, D.; and Tegmark, M. 2015. Research priorities for robust and beneficial artificial intelligence. *Ai Magazine* 36(4):105–114.

Tan, M. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, 330–337.

Watkins, C. J., and Dayan, P. 1992. Q-learning. *Machine learning* 8(3-4):279–292.

Zinkevich, M.; Greenwald, A.; and Littman, M. L. 2006. Cyclic equilibria in markov games. In *Advances in Neural Information Processing Systems*, 1641–1648.