*Fuzzy Systems and Data Mining VI*
*A.J. Tallón-Ballesteros (Ed.)*


*doi:10.3233/FAIA200744*

# Reward-Free Reinforcement Learning Algorithm Using Prediction Network

Zhen Yu[a,b,1], Yimin Feng[a] and Lijun Liu[a,b]
[a] *School of Aerospace Engineering, Xiamen University, Xiamen 361005, China*
[b] *Shenzhen Research Institute of Xiamen University, Shenzhen 518000, China*

**Abstract.** In general reinforcement learning tasks, the formulation of reward functions is a very important step in reinforcement learning. The reward function is not easy to formulate in a large number of systems. The network training effect is sensitive to the reward function, and different reward value functions will get different results. For a class of systems that meet specific conditions, the traditional reinforcement learning method is improved. A state quantity function is designed to replace the reward function, which is more efficient than the traditional reward function. At the same time, the predictive network link is designed so that the network can learn the value of the general state by using the special state. The overall structure of the network will be improved based on the Deep Deterministic Policy Gradient (DDPG) algorithm. Finally, the algorithm was successfully applied in the environment of FrozenLake, and achieved good performance. The experiment proves the effectiveness of the algorithm and realizes rewardless reinforcement learning in a class of systems.

**Keywords.** Reinforcement learning, no reward value, prediction network, RFPG

## 1. Introduction

Reinforcement learning is an important machine learning method that has many applications in the fields of intelligent control, robotics, analysis, and prediction [1]. Such as using fuzzy reinforcement learning approach for efficient and reliable solution to the unit commitment problem [2], using fuzzy reinforcement learning propose an efficient solution to the Economic thermal power dispatch [3], using fuzzy reinforcement learning to increase efficiency of the traffic light control system [4]. Reinforcement learning overlaps with many disciplines. Compared with traditional machine learning, it has the following advantages: Firstly, because it does not require a sample labeling process, it can solve special control problems more effectively; Secondly, the entire system as a whole can enhance its robustness; thirdly, reinforcement learning can relatively easily learn some tasks that cannot be completed by traditional control methods [5]. Therefore, as a new type of control method, reinforcement learning can play its unique role in many fields.

Deep reinforcement learning [6] (DRL) is a kind of reinforcement learning using deep learning [7] methods, such as DQN algorithm. However, DQN has certain limitations which can only deal with discrete problems [8], and most of the systems that

[1] Corresponding Author: LiJun Liu, Room 485, Aerospace Building, Xiang'an Campus of Xiamen University, Xiamen, Fujian, China; E-mail: liulijun@xmu.edu.cn.

need to be controlled are high-dimensional continuous and must output continuous control quantities. The discretization of continuous quantities will cause the problem of dimensional explosion [9]. In response to this situation, the DeepMind team proposed the deep deterministic strategy gradient algorithm DDPG in 2015 [10-11]. Traditional reinforcement learning algorithms have a common drawback, which relies too much on artificially designing reward functions. For the agent in an unknown environment, the reward function can provide a goal and an improvement direction for the reinforcement learning algorithm [12]. If the system has too many state variables, the reward value function will be very complicated [13]. The artificially designed reward value function may deviate from human intuition value, resulting in the inconsistent direction of Agent learning and people's cognition [14-15]. This drawback limits the application scenarios of reinforcement learning. Inverse reinforcement learning [16] is an exploration of the reinforcement learning algorithm with no reward value function, which uses human expert data to reverse the reward value function. However, the workload of expert data collection is large, which is not suitable for a wide range of applications.

Based on the above situation, this paper proposes an algorithm model without reward function. Design a new evaluation function by defining the characteristics of a class of controlled systems. In a system that satisfies these characteristics, an algorithm without reward value function is proposed. This method can avoid the design of reward value function and reduce the design workload. In order to evaluate the effectiveness of the algorithm, Gym [17] is used to build a simulation environment. Use Tensorflow [18] to build an algorithm environment. This paper defines the concept of state quantity, and designs the Reward Free Policy Gradient (Reward Free Policy Gradient) RFPG algorithm using predictive network learning state transition relations. The algorithm performed well in various experiments and proved its effectiveness.

## 2. Reinforcement learning model

The basic structure of reinforcement learning includes the controlled object Agent, environment, state, reward, and action [19]. The purpose of reinforcement learning is to learn certain strategies so that Agent can reach the target state. The tasks of reinforcement learning are usually described by Markov processes [20]. We abstractly represent the trajectory of a Markov process as $(s_1, a_1, r_1, s_2, a_2, r_2)$.The reward at $s_t$ is $r_t = R(s)$ , $R(s)$ is the reward function. The expression is as equation (1):

$$r_t = R(s_t) = \sum_{i=0}^{n} b_i \left\| x_{ti} - x_{gi} \right\|_m \tag{1}$$

Where $s_t = (x_{t1}, x_{t2}, \ldots, x_{tn})$ is the current state, $s_g = (x_{g1}, x_{g2}, \ldots, x_{gn})$ is the target state, $b_i$ is the empirical parameter, and $m$ is a certain norm. The reward function is very critical in reinforcement learning. The cumulative reward value $V_t$ is expressed as:

$$V_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{2}$$

Where $\gamma$ is the reward discount value, Agent and the goal of adopting strategies are to maximize the cumulative reward value $V_t$ From this, the value-based method represented by Q-learning [21] and the strategy-based method represented by strategy gradient were born. The DDPG algorithm combines the advantages of the two.

DDPG is mainly composed of strategy network, action network and experience pool. The strategy network is responsible for outputting actions, and the evaluation network is responsible for estimating value. And the two groups of networks have an online network and a target network, respectively, and these four networks are denoted as Actor-online, Actor-target, Critic-online, Critic-target. The experience pool is a data structure that can store basic units. The four networks interact to update the parameters. The DDPG algorithm also relies on the reward function design.

## 3. RFPG Algorithm

### *3.1. First-class application scenarios*

In reinforcement learning algorithms such as DQN and DDPG, we use the reward value provided by the environment as the basis for the correction of the evaluation network and the optimization goal of the strategy network. Researches have found that certain reinforcement learning scenarios do not require specific reward value functions. Summarizing such conditions into the following two, this paper is dedicated to solving the control problems of this type of system: (1) The purpose of Agent is to achieve a certain target state. (2) Agent has a probability to reach the target state when outputting random actions. The set of target states is recorded as $I$. The state that is not allowed to exist is called the failure state, and the failure state is recorded as $E$. The environments that meet the above conditions include FrozenLake, etc.

### *3.2. RFPG algorithm design*

This paper designs a reinforcement learning algorithm with no reward value called RFPG (Reward Free Policy Gradient) algorithm. The RFPG algorithm includes three networks: prediction network P, action network A, and value network G. Value network: divided into online value network G-online and target value network G-target, where the role of online value network is to fit the value of each state; the role of target value network is to provide value targets for online action network. Action network: divided into online action network A-online and target action network A-target, where the role of online action network is to output action values during the training phase; the role of target action network is to provide action output for the prediction network. Prediction network: The prediction network represents the Agent's understanding of the external environment, and its role is to predict the next state obtained by the action under the action.

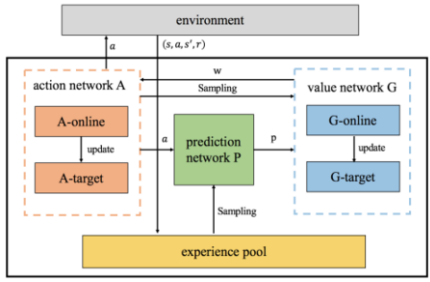


**Figure 1.** RFPG basic structure

A basic experience unit in DDPG is $(s, a, s', r)$, where $r$ is the reward value among them. This reward value is usually a customized function of human expert experience. The RFPG algorithm replaces the reward value $r$ with a simpler state quantity $w$. The definition of $w$ is shown in equation (3), which is simpler than equation (1).

$$w_s = W(s) = \begin{cases} 1 & s \in I \\ 0 & s \notin I \cup E \\ -1 & s \in E \end{cases} \tag{3}$$

In the definition of observation $w$, the state quantity function $W(s)$ is different from the value function $R(s)$, which only defines the target state $I$ and the failure state $E$. The ideal value of other general conditions $w$ is 0, and its value is learned by the G network, not provided by the environment. The basic experience unit of the RFPG experience pool is $(s, a, s', r)$. This process avoids the design of traditional reward functions. State quantity $w$ is simpler, intuitive and easier to use than traditional reward functions, reducing design workload. The relationship between the various networks is shown in Figure 2.

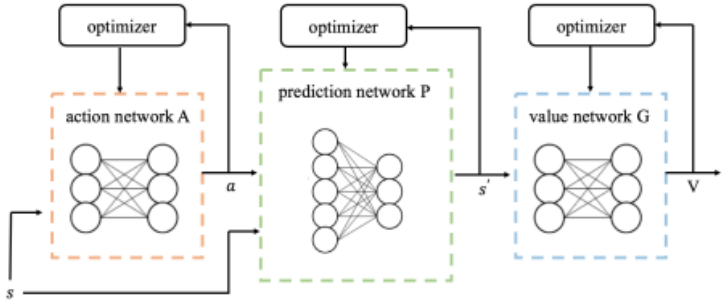


**Figure 2.** RFPG network relationships

Each network parameter is indicated by $\theta$, $\theta_G$ is a G network parameter, $\theta_u$ is an A network parameter, and $\theta_p$ is a G network parameter.

The purpose of the A-online output action is to maximize value, so the action network loss function is equation (4). A-target obtains parameter values from A-online in soft update mode.

$$loss\_a = J(\theta_\mu) = -\frac{1}{m}\sum_{j=1}^{m} G(p_i \mid \theta_\mu) \tag{4}$$

The value network G is the learning of the value function. The value of the G network is obtained through learning. The objective function of fitting is equation (5).

$$V_t = \begin{cases} 1 & s_t \in I \\ \gamma V_{t+1} & s_t \notin I \cup E \\ -1 & s_t \in E \end{cases} \tag{5}$$

Where $\gamma$ is the attenuation coefficient, which is generally around 0.9. The loss function of G-online is shown in equation (6), which $p_i$ is the output of the prediction network. G-target obtains parameter values from G-online in soft update mode.

$$loss\_g = J(\theta_G) = \frac{1}{m}\sum_{j=1}^{m} (w_i - \gamma G(p_i, \theta_G))^2 \tag{6}$$

The function of the prediction function is based on $s$, $a$ to predict the next state $s'$. The samples $s'$ in the sample can be regarded as labels to do training to minimize the mean square error. The loss function corresponds to equation (7).

$$loss\_p = J(\theta_P) = \frac{1}{m}\sum_{j=1}^{m} (s_{i+1} - P(s_i, a_i))^2 \tag{7}$$

In practical applications, the network infrastructure of each part is a typical double-layer neural network. The network hyperparameters can be adjusted according to the application scenario. The algorithm steps are shown in Algorithm 1 and the flowchart of RFPG are shown in Figure 3.

**Table 1.** RFPG algorithm flow

| **Algorithm 1 RFPG algorithm flow** |
|---|
| 1.Initialize action network A, predict network P, evaluate network G, and assign random weights $\theta_u, \theta_p$, $\theta_G$ |
| 2. Initialize the target network $\theta_{G'} \leftarrow \theta_G, \theta_{\mu'} \leftarrow \theta_\mu$, initialize the experience pool R. |
| 3. **for** j = 1 to M **do** |
| 4. Randomly initialize the environment to get the initial state S |
| 5. **for** t = 1 to T **do** |
| 6. Use the ε-greedy strategy to output action $a$ |
| 7. Agent performs action $a$ and obtains sample $(s, a, s', r)$ and deposit into experience pool |
| 8. Take n samples $(s_i, a_i, s_i', r_i)$ from the experience pool, forecasting $p_i$ with prediction network |
| 9. Update the value network with minimized variance G : $Lg = \frac{1}{n}\sum (w_i - \gamma G(p_i, \theta_g))^2$ |
| 10. Update the action network with a strategy gradient A. $La = -\frac{1}{n}\sum G(p_i, \theta_\mu)$ |

11. Train prediction network using state transition P
$$Lp = \frac{1}{n}\sum(S_{i+1} - P(S_i - a_i))^2$$
12. Use soft update to update the target network：
$$\theta_{G'} \leftarrow \tau\theta_G + (1-\tau)\theta_{G'}$$
$$\theta_{\mu'} \leftarrow \tau\theta_\mu + (1-\tau)\theta_{\mu'}$$
13. **end for**
14. **end for**

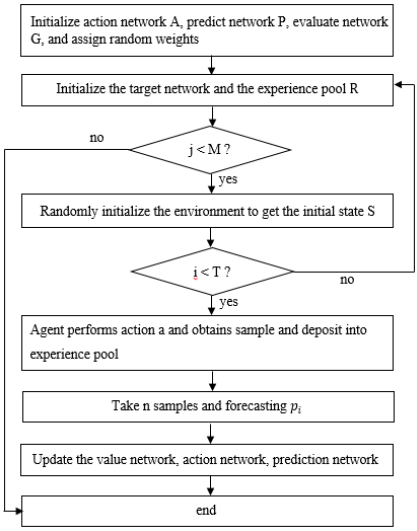


**Figure 3** flowchart of RFPG

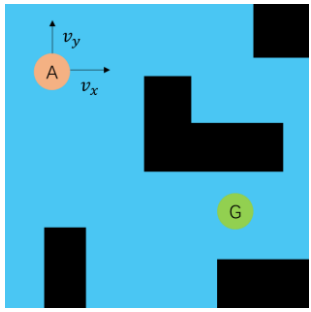


**Figure 4** FrozenLake environment

Compared with a series of traditional reinforcement learning methods such as DDPG, the RFPG algorithm has the following improvements: (1) The design of the reward value function is eliminated, and the workload is reduced. (2) Avoid situations where the network is sensitive to reward value functions. (3) Reduced dependence on the experience pool.

## 4. Simulation study

The purpose of this chapter is to verify the effectiveness of the algorithm and compare the performance of the DDPG algorithm. The main indicator is the average return value. From the previous analysis, it can be seen that the larger the average return value, the better the control performance. The following content analyzes the experimental results of FrozenLake and CartPole in detail. All networks in the experiment are fully connected neural networks.

### *4.1 Environmental introduction*

As shown in Figure 4, the environment is a continued state version of FrozenLake in Gym. The environment is a square ice surface with side length *n*, Agent glides on the ice surface. The starting point of the agent is marked A, and the target point is marked G.

Dangerous areas exist on the ice, which are represented as black areas in the figure. Entering the black area indicates that the operation failed. Agent has position $x, y$ on the horizontal and vertical coordinates, so the state value dimension is 2. The action output is speed $v_x$, $v_y$ ,and the action value dimension is 2. There is random interference in the environment, and there is a 10% chance that the action performed by the Agent cannot be executed smoothly, and the direction command is a random

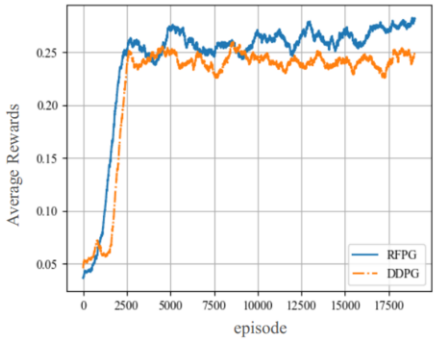


**Figure 5** Frozen Lake training process

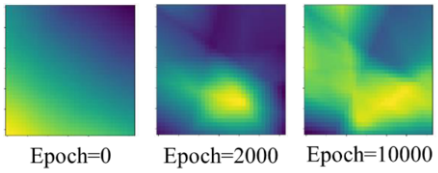


**Figure 6** Value network learning results

### *4.2 Experimental analysis*

The experiment first needs to define the state quantity $w$. The target state is the G position, which $w$ is 1 at this time; the failure state is the black area, which $w$ is -1 at this time, and the subtraction factor $\gamma$ is 0.9.

Figure 5 shows the trend of average returns in the Frozen Lake environment. The performance of the RFPG algorithm in the Frozen Lake environment is superior to the DDPG algorithm. It shows that the RFPG algorithm can still perform quite well without return value

For the environment in Figure 4, the learning results of the value network are shown in Figure 6. The colors in the figure represent the value learned. The brighter the color, the higher the value. It can be seen that the value area is basically consistent with the environmental characteristics. Explain that the value network has learned the characteristics of the environment.

## 5. Conclusion

Aiming at a class of reinforcement learning problems defined in Chapter 3, this paper designs a reinforcement learning algorithm RFPG with no reward value. This method uses state quantities to provide learning directions for the network. The state quantity can distinguish the target state, failure state and general state of the agent. Experiments in various environments, the experimental results show that in an environment that meets certain conditions, the reinforcement learning method without reward value can still achieve control of the system. In the future, the application scenarios can be further expanded in the research. The RFPG algorithm is not only used for solving fixed target problems, but also for solving optimal state problems.

## Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant 61304110, the Guangdong Natural Science Foundation under Grant 2018A030313124, the Natural Science Foundation of Shanghai under Grant 18ZR1443200, the Basic Research Program of Science and Technology of Shenzhen JCYJ20190809163009630, and the Open Foundation of Key Laboratory of Marine Navigation and Control Technology under Grant 202005.

## References

[1] Rubo Zhang, et al. Reinforcement learning theory, Algorithm and application[J]. Theory and Applications. 2000, 17(5):637-642.
[2] Navin N K , Sharma R . A fuzzy reinforcement learning approach to thermal unit commitment problem[J]. Neural Computing and Applications, 2017.
[3] Nandan Kumar Navin, Rajneesh Sharma, H. Malik. Solving nonconvex economic thermal power dispatch problem with multiple fuel system and valve point loading effect using fuzzy reinforcement learning. 2018, 35(5):4921-4931.
[4] Kumar N , Rahman S S , Dhakad N . Fuzzy Inference Enabled Deep Reinforcement Learning-Based Traffic Light Control for Intelligent Transportation System[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, PP(99):1-10.
[5] Guoyan Xu, Xiaopeng Zhong, Guizhen Yu, et al. Research on intelligent Obstacle Avoidance Method of Unmanned Vehicle based on DDPG[J]. Automotive Engineering. 2019, 41(02):90-96.
[6] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv:1312.5602, 2013.
[7] Yujian Li. Introduction to Deep Learning and Case Study [M]. China Machinery Industry Press, 2016.
[8] Šemrov D, Marsetič R, Žura M, et al. Reinforcement learn-ing approach for train rescheduling on a single-track rail-way[J]. Transportation Research Part B: Methodological, 2016, 86: 250-267.
[9] Heess N, Wayne G, Silver D, et al. Learning continuous control policies by stochastic value gradients[C]// Advances in Neural Information Processing Systems. 2015: 2944-2952.
[10] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.
[11] Lin L J. Reinforcement learning for robots using neural networks[R]. Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, 1993.
[12] Sutton R S, Barto A G. Reinforcement learning: An intro-duction[M]. MIT press, 2018.
[13] Christiano P F, Leike J, Brown T, et al. Deep reinforcement learning from human preferences[C]//Advances in Neural Information Processing Systems. 2017: 4299-4307.
[14] Bostrom N . Superintelligence[J]. Computer Science, 2016.
[15] Amodei D, Olah C, Steinhardt J, et al. Concrete problems in AI safety[J]. arXiv preprint arXiv:1606.06565, 2016.
[16] Ng A Y, Russell S J. Algorithms for inverse reinforcement learning[C]//Icml. 2000, 1: 2.
[17] Brockman G, Cheung V, Pettersson L, et al. Openai gym[J]. arXiv preprint arXiv:1606.01540, 2016.
[18] Abadi M, Agarwal A, Barham P, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems[J]. arXiv preprint arXiv:1603.04467, 2016.
[19] Quan Liu, Jianwei Zhai, Zongchang Zong, et al. Review of Deep Reinforcement learning [J].Journal of Computer Science, 2018, 41(1): 1-27.
[20] Yin Luo, Wenhu Qin, Jinfeng Zhai. Research on Vehicle Low-speed tracking Behavior Decision Based on improved DDPG Algorithm [J].Measurement and Control Technology, 2019 (9): 4.
[21] Watkins C J C H. Learning from delayed rewards[J]. 1989.