Prescriptive Dirichet Task Allocation Policy with Deep Reinforcement Learning

Ali Fuat Sahin *
Institute of Mechanical Engineering
École Polytechnique Fédérale de Lausanne
Lausanne, VD 1015
alifuat.sahin@epfl.ch

Emre Gursoy †
Institute of Mechanical Engineering
École Polytechnique Fédérale de Lausanne
Lausanne, VD 1015
mustafa.gursoy@epfl.ch

Matthias Schuller ‡
Institute of Mathematics
École Polytechnique Fédérale de Lausanne
Lausanne, VD 1015
matthias.schuller@epfl.ch

Abstract

Prescribing effective operations based on system conditions can significantly improve the availability, maintenance, and costs of complex systems. Reinforcement learning (RL) algorithms are ideal for task allocation problems, but traditional Gaussian policies in RL introduce bias and do not support constrained action spaces. This study implements the Dirichlet policy for continuous allocation tasks, showing it to be bias-free, faster in convergence, better in performance, and more robust than Gaussian-softmax policies. Using a UAV simulation for search and rescue missions, we aim to maximize the cumulative search distance before any UAV fails, considering random mission distances and varying health states. The Soft Actor-Critic (SAC) algorithm optimizes actions based on these factors. Results demonstrate that integrating the Dirichlet policy with the SAC framework reduces bias and variance, outperforming traditional task allocation strategies and extending UAV operational time. This approach offers significant improvements for dynamic task allocation and applies to various domains.

1 Introduction

Effective task allocation based on system conditions can significantly enhance the performance, availability, and cost management of complex systems. This involves making sequential decisions using precise dynamic models or robust learning capabilities. Recently, reinforcement learning (RL) algorithms, especially model-free RL, have shown remarkable success in sequential decision-making tasks, determining optimal policies without detailed environmental models. This has led to breakthroughs in gaming, control systems, prescriptive maintenance, and automated machine learning.

Task allocation is a critical application of prescriptive operations, where tasks are dynamically assigned to optimize system performance and longevity. This problem is often complicated by

*Student ID: 374463
†Student ID: 376825
‡Student ID: 315028

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

constraints on the action space, such as simplex constraints, which traditional RL methods struggle to handle effectively, leading to bias and variance issues.

This study focuses on task allocation within the context of unmanned aerial vehicle (UAV) simulations for search and rescue missions. Our objective is to maximize the cumulative search distance before any UAV fails, considering the random nature of mission distances and varying health states of UAV components. Traditional methods, such as Gaussian-softmax policies, often inadequately address these challenges due to inherent bias and variance.

We propose using a Dirichlet policy within the Soft Actor-Critic (SAC) algorithm to overcome these limitations. The Dirichlet policy offers a stable and unbiased probabilistic framework for task assignment, better suited to handle the uncertainties and dynamic conditions of search and rescue missions.

2 Related Work

In reinforcement learning (RL), agents observe the environment and take actions to maximize cumulative expected future rewards. The action space in RL tasks can be discrete, continuous, or a combination of both. Traditional methods like Q-learning Watkins, Dayan (1992) and its extensions, such as deep Q-networks (DQN) Mnih et al. (2015) and double-DQN Van Hasselt et al. (2016), have been effective for discrete action-space tasks. However, continuous action-space problems require different approaches, leading to the development of policy-based algorithms.

Several algorithms have been proposed to handle continuous action spaces effectively. Notable examples include Proximal Policy Optimization (PPO) Schulman et al. (2017), Trust Region Policy Optimization (TRPO) Schulman et al. (2015), and Soft Actor-Critic (SAC) Haarnoja et al. (2018a), which use Gaussian distributions to represent stochastic policies. Deep Deterministic Policy Gradient (DDPG) Lillicrap et al. (2015) employs a deterministic policy approach but tends to be less robust in complex environments. To improve efficiency under physical constraints, the beta policy has been introduced.

Despite these advancements, the application of RL algorithms to allocation tasks remains under-explored. Allocation tasks, such as computational resource allocation and reliability redundancy allocation, involve distributing limited resources to achieve goals while adhering to constraints. These tasks are crucial in domains like industrial automation, blockchain, and UAV operations.

Handling simplex constraints in RL problems is still a developing area. Previous work by Tian et al. (2022) implemented the Dirichlet policy with SAC for power allocation tasks, achieving promising results. Additionally, Ale et al. (2022) used the Dirichlet policy with DDPG for optimizing task partitioning and computational power allocation in dynamic environments. Our approach builds on Tian et al. (2022)'s work for more comprehensive environments and introduces a target entropy formulation for the exploration-exploitation tradeoff in RL algorithms with Dirichlet policies.s

3 Methodology

In this section, we present our method for efficiently assigning tasks in a multi-agent system using a reinforcement learning framework. Our approach addresses the limitations of the Gaussian-Softmax policy, which often introduces bias and variance, by integrating the Dirichlet policy into the Soft Actor-Critic (SAC) algorithm. This combination is particularly effective in handling the complexities of continuous action spaces with simplex constraints.

Traditional Gaussian-Softmax policies in reinforcement learning are prone to bias and variance issues, which can lead to suboptimal performance. The Dirichlet policy, however, provides a more stable and unbiased framework for task assignment. One of the key challenges we address is the formulation of entropy targets during training for the Dirichlet policy. Unlike Gaussian policies, the entropy of the Dirichlet policy is bounded, and traditional entropy formulations derived for Gaussian policies do not apply. Therefore, we introduce a novel target entropy function that optimizes performance within the maximum entropy learning framework.

3.1 Implications of Gaussian-Softmax Policy

In reinforcement learning (RL), the policy gradient is used to update policy parameters to maximize the expected cumulative reward. The policy gradient can be expressed as:

$$E_g(\theta) = \mathbb{E}_s \left[\int_{-\infty}^{\infty} \pi_\theta(\mathbf{x}|s) \nabla_\theta \log \pi_\theta(\mathbf{x}|s) Q^\pi(s, \mathbf{x}) \, d\mathbf{x} \right] \tag{1}$$

Here, $\pi_\theta(\mathbf{x}|s)$ is the policy distribution over actions $\mathbf{x}$ given state s, and $Q^\pi(s, \mathbf{x})$ is the action-value function.

To handle continuous action spaces, the Gaussian-Softmax policy is often used, which involves sampling from a Gaussian distribution and applying the softmax function. This can introduce bias due to unaccounted inner integration over the translation parameter c Tian et al. (2022). Mathematically, this is expressed as:

$$E_g(\theta) = \mathbb{E}_s \int_{-\infty}^{\infty} \pi_\theta(\mathbf{x} + c\mathbf{1}|s) \nabla_\theta \log \pi_\theta(\mathbf{x} + c\mathbf{1}|s) Q^\pi(s, \sigma(\mathbf{x})) \, d\mathbf{x} \, dc \tag{2}$$

where $\mathbf{x} \sim \mathcal{N}(\mu, \Sigma)$ and $\sigma(\mathbf{x})$ is the softmax function. The policy gradient estimator $\hat{E}_g(\theta)$ does not account for the inner integration over c, leading to bias.

Additionally, the variance of the policy gradient estimator is proportional to $1/\sigma^2$, where σ is the standard deviation of the Gaussian distribution. As $\sigma \to 0$, the variance increases, causing instability. Even with the natural policy gradient, which adjusts learning rates, the Fisher information matrix for Gaussian-Softmax policies is proportional to $1/\sigma^2$, resulting in increased variance and instability as the policy becomes more deterministic.

3.2 Implications of Dirichlet Policy

The Dirichlet policy offers a bias-free and low-variance alternative for allocation tasks. It inherently satisfies the simplex constraint, allowing direct sampling of allocation actions without further constraints. This ensures the policy gradient estimator is free from bias.

Moreover, the variance of the Dirichlet policy is bounded, even as the policy becomes deterministic. The Dirichlet distribution maintains stability in deterministic policies, keeping the variance of the policy gradient estimator low. This leads to stable and efficient optimization of allocation tasks.

By adopting the Dirichlet policy, we mitigate the bias and variance issues of the Gaussian-Softmax policy, enhancing overall performance and convergence speed in RL settings.

3.3 Reinforcement Learning Framework

In this study, we leverage the Soft Actor-Critic (SAC) Haarnoja et al. (2018a), Haarnoja et al. (2018b) algorithm to address the task allocation problem in a multi-agent system. SAC is a state-of-the-art off-policy reinforcement learning (RL) algorithm known for its ability to handle continuous action spaces and ensure stable learning.

We extend SAC by parameterizing the policy using the Dirichlet distribution, which naturally satisfies the simplex constraint. This makes it well-suited for allocation tasks where actions must be probabilistic and constrained to a simplex. By using the Dirichlet policy, we ensure adherence to allocation constraints while optimizing efficiently in continuous action spaces.

The SAC algorithm with a Dirichlet policy offers several advantages for our task allocation problem. It provides stability and convergence guarantees in learning optimal policies, allows for efficient exploration of the action space, and ensures compliance with the simplex constraint, facilitating seamless integration into real-world allocation scenarios.

3.4 Maximum Entropy Reinforcement Learning with Dirichlet Policy

The standard reinforcement learning objective is to learn a policy that maximizes the expected sum of rewards. The maximum entropy objective Ziebart et al. (2008), generalizes the standard objective by

augmenting it with an entropy term such that the objective is to maximize the policy entropy at each visited state and the trade-off between reward and entropy is controlled by a temperature parameter α.

$$\pi^* = \arg\max_{\pi} \sum_{t} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\big[r(s_t,a_t) + \alpha H(\pi(.|s_t))\big] \tag{3}$$

The maximum entropy approach offers both conceptual and practical advantages. Firstly, the policy is encouraged to explore a wider range of options. Secondly, in situations where multiple options appear equally attractive, the policy can assign equal probabilities to those actions.

The SAC algorithm aims to find a stochastic policy with a maximal expected return that satisfies a minimum entropy constraint Haarnoja et al. (2018b).

$$\max_{\pi_{0:T}} \sum_{t} \mathbb{E}_{\rho_\pi}\big[r(s_t,a_t)\big] \quad s.t. \quad \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\big[-\log(\pi(a_t|s_t))\big] \geq H \quad \forall t \tag{4}$$

Where H denotes the target entropy term of auto-entropy tuning in SAC. Usually, this constraint is tight to encourage exploration during training. However, target entropy, especially in discrete environments, could be very environment-specific. In general, traditional target entropy functions implemented with Gaussian policies do not converge with Dirichlet policy implementations. Here, we propose a target entropy function for SAC with Dirichlet policy based on the upper bound of entropy of Dirichlet distribution.

Dirichlet distribution's entropy can be defined as follows Lin (2016).

$$H(Y^k) = \log(B(\alpha^K)) + (\alpha_0 - K)\psi(\alpha_0) - \sum_{i=1}^{K}(a_i - 1)\psi(\alpha_i) \tag{5}$$

Where, $\psi(x)$ is the digamma function, α^K is the vector of concentration parameters of the distribution, K is the cardinality of α, $\alpha_0 = \sum_{t=1}^{k} \alpha_t$ and $B(\alpha_i) = \frac{\prod_{t=1}^{k}\Gamma(\alpha_t)}{\Gamma(\sum_{t=1}^{k}\alpha_t)}$ where $\Gamma(x)$ is the gamma function. When $\alpha_i = 1\ \forall i \in K$, the Dirichlet distribution is uniform over all points in its support Kotz et al. (2019). This particular distribution is known as flat Dirichlet distribution and the entropy is maximum in this special case. This upper bound on the entropy of the entropy can be simply calculated by letting $\alpha_i = 1\ \forall i \in K$ as follows.

$$\max H(Y^k) = -log(\Gamma(K)) \tag{6}$$

Here, Eqn. (6) defines the upper bound for Dirichlet policy target entropy.

In our experiments, we have changed the target entropy function a couple of times to find a suitable formulation for each environment. We have seen that the target entropy of the policy is very sensitive to environmental change and it is hard to tune in general. Therefore, we have implemented target entropy scheduling Xu et al. (2021) to relax the entropy constraint as the training progresses. Xu et al. (2021) tested this approach only with discrete environments. In our case, we have implemented and tested the algorithm's performance in continuous action spaces.

With this approach, we do not have to tune the target entropy or the temperature parameter for each case. Initially, when the entropy constraint is high, the policy explores a wide range of actions, which can lead to discovering high-reward regions of the environment. As you anneal the entropy constraint, the policy becomes more deterministic and focuses more on exploiting known high-reward actions.

4 Implementation Details

4.1 Algorithm Implementation

The algorithm is implemented in Python utilizing the PyTorch framework. For scenarios involving a small number of UAVs, a neural network architecture with hidden layers of size `256-256` is employed.

In contrast, larger fleets are managed using a more complex `256-512-256` network structure. Both policy and critic networks exclusively employ Leaky ReLU activation functions.

In our setup, we utilize double Q-learning, employing two separate Q-functions represented by neural networks with their respective parameters, ν_1 and ν_2. When it comes to learning the policy, we rely on the Q-function that yields the lower value. This approach aids in minimizing the impact of value estimation bias, thus improving overall performance.

To ensure non-negative outputs in the Dirichlet policy, a Softplus layer is incorporated. When α values are less than one, the Dirichlet policy tends to produce highly deterministic actions, avoiding actions corresponding to α values below one. To mitigate this issue, a constant of 1 is added to the policy output.

Target entropy tuning is performed using a target entropy annealing algorithm Xu et al. (2021), which iteratively adjusts the target entropy so that it is achievable in the given timeframe.

The hyperparameters used for the training of 4 UAVs can be seen in Table 1.

Table 1: Hyperparameters used to train 4 UAVs

Hyperparameter	**Value**
Hidden Dimensions	[256, 256]
Batch Size	256
Alpha	0.01
Gamma	0.99
Tau	0.005
Learning Rate	0.0001
Update Interval	1
Initial Target Entropy	$-\Gamma(4)$

4.2 Case Studies

4.2.1 Simplex Regression Task

Simplex regression revolves around reconstructing and sequencing a 4-dimensional simplex from a 3-dimensional vector.

The problem begins with a randomly generated 4-dimensional simplex vector representing a geometric shape in a multi-dimensional space. Subsequently, one dimension is randomly removed from this original vector, resulting in a 3-dimensional vector. For instance, if the original simplex vector reads [0.4, 0.2, 0.3, 0.1], a dimension is selectively eliminated, yielding an input vector such as [0.4, 0.3, 0.1].

Our goal is twofold: first, to accurately predict the missing dimension from the 3-dimensional input vector, and second, to correctly sequence and reconstruct the original 4-dimensional simplex. This entails not only discerning the missing dimension but also rearranging the remaining elements to form a ranked reconstructed simplex. For instance, if the missing dimension corresponds to 0.2 in the original vector, the reconstructed simplex should read [0.1, 0.2, 0.3, 0.4], with the elements appropriately rearranged in ascending order.

Schematic of the RL problem can be seen in Figure 1.

To quantify the performance of our algorithm, we employ the Mean Average Error (MAE) as our evaluation metric. Lower MAE values signify a higher degree of accuracy in reconstructing the original simplex.

Through this case study, we aim to demonstrate the robustness and versatility of Dirichlet policy in tackling allocation tasks.

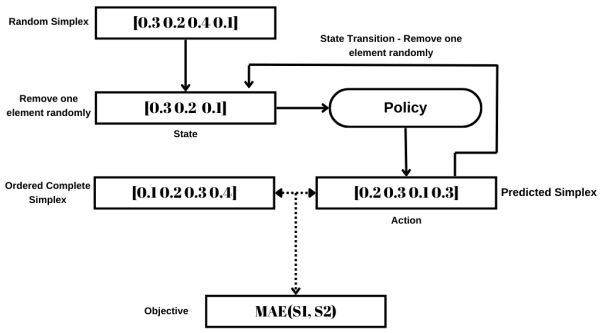


Figure 1: Simplex regression task RL problem.

4.3 UAV Fleet Task Allocation

In our simulation, we focus on monitoring the health status of critical components within Unmanned Aerial Vehicles (UAVs), specifically the bearings and coils of four hover motors and one pusher motor. To ensure consistency and reliability in training outcomes, we initialize the health status of each component within a predetermined range of 0.8 to 1, with the aggregate sum constrained to $0.9N$, where N is the number of UAVs. This normalization procedure effectively mitigates training variance.

Each component is endowed with a stochastic failure appearance parameter, initialized randomly between 0.45 and 0.6. Subsequent to the appearance of failure, the severity of degradation accelerates based on randomly assigned severity factors for each component. This dynamic accounts for varying rates of degradation, with faster deterioration correlated to higher severity factors.

Notably, the degradation dynamics of coils exhibit a nuanced pattern, initially deteriorating at a slower pace but frequently experiencing higher severity levels of failure than the bearings Garcia-Calva et al. (2022). This characteristic underscores the complexity of component degradation profiles within the UAV system and ensures a uniform failure distribution among the components, hampering the over-fitting to a certain component. In Table 2, we present the distribution of component failures across 1000 simulation runs within the UAV system.

Table 2: Percentage of failures for different components over 1000 runs.

Component	**Percentages of failure**
Hover Bearing 1	13%
Hover Bearing 2	10%
Hover Bearing 3	8%
Hover Bearing 4	11%
Hover Coil 1	7%
Hover Coil 2	6%
Hover Coil 3	11%
Hover Coil 4	8%
Pusher Bearing	16%
Pusher Coil	9%

Mission planning involves the random generation of total mission distances within an upper limit, which is then partitioned among the UAVs. Following allocation, individual UAVs are assigned random distances for both hover and cruise segments of the mission. Hover distances predominantly utilize hover motors, resulting in accelerated degradation of associated components, while the pusher motor predominates during the cruise, influencing degradation accordingly.

Total component failure is defined as any health status dropping below zero. Our neural network architecture incorporates inputs of all component health states, alongside hover and cruise distances.

Action selection within the network is represented by percentage allocations of distances for the UAVs.

5 Results

5.1 Simplex Regression Task

The result shows that the Dirichlet distribution performs better and is more robust to different learning rates. For the given learning rates, the Dirichlet policy performs two times better than the Gaussian policy for the simplex regression task, as can be seen in Figure 2.

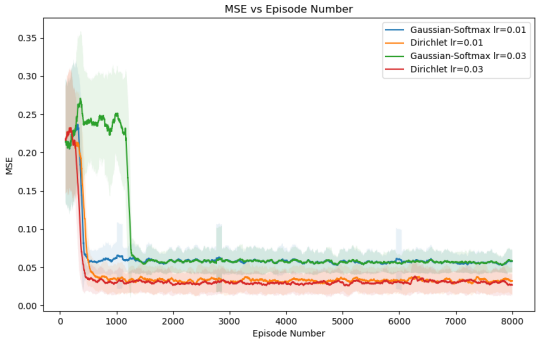
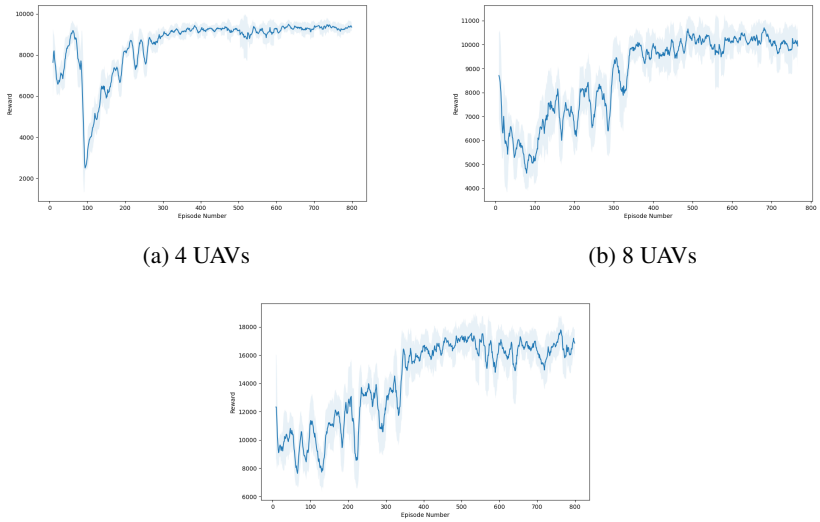


Figure 2: Gaussian-Softmax vs. Dirichlet policy training curves for the simplex regression task.

5.2 UAV Fleet Task Allocation

5.2.1 Training Performance

Training performance and convergence plots can be seen in Figure 3.



(a) 4 UAVs

(b) 8 UAVs

(c) 12 UAVs

Figure 3: Training curves for different numbers of UAVs. Rewards were taken as a rolling average for a window of 10 episodes, and the shaded areas show the standard deviation.

5.2.2 Performance Analysis

To evaluate the performance of our agent, we compared it against two baselines. The first baseline involves equally distributing the missions among all UAVs. The second baseline allocates missions in direct proportion to the health state of each UAV. The number of missions accomplished for different fleet sizes is presented in Table 3.

Table 3: Completed mission comparison for baselines and RL agent

Fleet Size	Baseline 1	Baseline 2	RL Agent
4	55	54	62
8	187	185	205
12	250	251	275

5.3 Case Study Insights

For the simplex regression task, the results revealed that the Dirichlet policy outperformed the Gaussian-softmax policy in terms of accuracy, as measured by the Mean Average Error (MAE). This improvement is attributed to the inherent properties of the Dirichlet distribution, which better captures the probabilistic nature of simplex data. Additionally, the Gaussian-softmax policy exhibited higher bias and variance, further validating the robustness of the Dirichlet policy.

The study on task allocation for the UAV fleet showed that our method effectively balances exploration and exploitation, adapting to changes in the environment and the health of the UAVs. By using adaptive entropy annealing, the policy initially focused on exploration to discover optimal strategies, and then gradually shifted towards exploitation as training progressed. Gradually reducing the entropy constraint helped maintain a balance between exploration and exploitation, preventing premature convergence to suboptimal policies and ensuring continued exploration of the state-action space.

6 Conclusion

6.1 Implications

In general, our study presents a prescriptive, fully autonomous, scalable, and transferable framework with Dirichlet policy. Also, giving the target entropy and scheduling algorithm for the framework. This framework improves the sustainability and efficiency of multi-agent systems. Also, the Dirichlet policy framework for continuous allocation tasks mitigates bias and reduces variance, offering significant improvements in efficiency and reliability. This approach applies to various continuous control reinforcement learning algorithms.

6.2 Future Work

All project-related folders can be found at `https://github.com/alifuatsahin/UAV-Fleet-Task-Allocation-RL`.

In our tests, we have observed that the choice of temperature parameter and target entropy significantly affects the results and is generally environment-specific. Although we addressed this issue by implementing target entropy scheduling Xu et al. (2021), the hyperparameters of the target entropy scheduling algorithm should be theoretically derived before being applied to different environments. We believe that the standard deviation threshold and mean threshold parameters should be related to the action space size and the concentration parameter's norm (α). In future work, it is important to derive these relationships and conduct thorough comparative tests.

Future work also includes applying and deploying the proposed framework to more challenging and extensive real-world problems, evaluating the limitations of the proposed framework, and extending the algorithm for time-varying action space depending on the UAV availabilities.

References

Ale Laha, King Scott A, Zhang Ning, Sattar Abdul Rahman, Skandaraniyam Janahan. D3PG: Dirichlet DDPG for task partitioning and offloading with constrained hybrid action space in mobile-edge computing // IEEE Internet of Things Journal. 2022. 9, 19. 19260–19272.

Garcia-Calva Tomas, Morinigo-Sotelo Daniel, Fernandez-Cavero Vanessa, Romero-Troncoso Rene. Early detection of faults in induction motors—A review // Energies. 2022. 15, 21. 7855.

Haarnoja Tuomas, Zhou Aurick, Abbeel Pieter, Levine Sergey. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor // International conference on machine learning. 2018a. 1861–1870.

Haarnoja Tuomas, Zhou Aurick, Hartikainen Kristian, Tucker George, Ha Sehoon, Tan Jie, Kumar Vikash, Zhu Henry, Gupta Abhishek, Abbeel Pieter, others . Soft actor-critic algorithms and applications // arXiv preprint arXiv:1812.05905. 2018b.

Kotz Samuel, Balakrishnan Narayanaswamy, Johnson Norman L. Continuous multivariate distributions, Volume 1: Models and applications. 334. 2019.

Lillicrap Timothy P, Hunt Jonathan J, Pritzel Alexander, Heess Nicolas, Erez Tom, Tassa Yuval, Silver David, Wierstra Daan. Continuous control with deep reinforcement learning // arXiv preprint arXiv:1509.02971. 2015.

Lin Jiayu. On the dirichlet distribution // Department of Mathematics and Statistics, Queens University. 2016. 40.

Mnih Volodymyr, Kavukcuoglu Koray, Silver David, Rusu Andrei A, Veness Joel, Bellemare Marc G, Graves Alex, Riedmiller Martin, Fidjeland Andreas K, Ostrovski Georg, others . Human-level control through deep reinforcement learning // nature. 2015. 518, 7540. 529–533.

Schulman John, Levine Sergey, Abbeel Pieter, Jordan Michael, Moritz Philipp. Trust region policy optimization // International conference on machine learning. 2015. 1889–1897.

Schulman John, Wolski Filip, Dhariwal Prafulla, Radford Alec, Klimov Oleg. Proximal policy optimization algorithms // arXiv preprint arXiv:1707.06347. 2017.

Tian Yuan, Han Minghao, Kulkarni Chetan, Fink Olga. A prescriptive Dirichlet power allocation policy with deep reinforcement learning // Reliability Engineering & System Safety. 2022. 224. 108529.

Van Hasselt Hado, Guez Arthur, Silver David. Deep reinforcement learning with double q-learning // Proceedings of the AAAI conference on artificial intelligence. 30, 1. 2016.

Watkins Christopher JCH, Dayan Peter. Q-learning // Machine learning. 1992. 8. 279–292.

Xu Yaosheng, Hu Dailin, Liang Litian, McAleer Stephen, Abbeel Pieter, Fox Roy. Target entropy annealing for discrete soft actor-critic // arXiv preprint arXiv:2112.02852. 2021.

Ziebart Brian D, Maas Andrew L, Bagnell J Andrew, Dey Anind K, others . Maximum entropy inverse reinforcement learning. // Aaai. 8. 2008. 1433–1438.