


Article

Policy Optimization of the Power Allocation Algorithm Based on the Actor–Critic Framework in Small Cell Networks

Haibo Chen ^{1,†} , Zhongwei Huang ^{1,†}, Xiaorong Zhao ¹, Xiao Liu ¹, Youjun Jiang ¹, Pinyong Geng ¹, Guang Yang ², Yewen Cao ^{1,*} and Deqiang Wang ^{1,*}¹ School of Information Science and Engineering, Shandong University, Qingdao 266237, China² College of Electronic and Information Engineering, Shandong University of Science and Technology, Qingdao 266590, China

* Correspondence: ycao@sdu.edu.cn (Y.C.); wdq_sdu@sdu.edu.cn (D.W.)

† These authors contributed equally to this work.

Abstract: A practical solution to the power allocation problem in ultra-dense small cell networks can be achieved by using deep reinforcement learning (DRL) methods. Unlike traditional algorithms, DRL methods are capable of achieving low latency and operating without the need for global real-time channel state information (CSI). Based on the actor–critic framework, we propose a policy optimization of the power allocation algorithm (POPA) for small cell networks in this paper. The POPA adopts the proximal policy optimization (PPO) algorithm to update the policy, which has been shown to have stable exploration and convergence effects in our simulations. Thanks to our proposed actor–critic architecture with distributed execution and centralized exploration training, the POPA can meet real-time requirements and has multi-dimensional scalability. Through simulations, we demonstrate that the POPA outperforms existing methods in terms of spectral efficiency. Our findings suggest that the POPA can be of practical value for power allocation in small cell networks.

Keywords: power allocation; deep reinforcement learning; actor–critic**MSC:** 94-10

Citation: Chen, H.; Huang, Z.; Zhao, X.; Liu, X.; Jiang, Y.; Geng, P.; Yang, G.; Cao, Y.; Wang, D. Policy Optimization of the Power Allocation Algorithm Based on the Actor–Critic Framework in Small Cell Networks. *Mathematics* **2023**, *11*, 1702. <https://doi.org/10.3390/math11071702>

Academic Editors: Mohamed Abouhawwash and K. Venkatachalam

Received: 6 February 2023
Revised: 31 March 2023
Accepted: 1 April 2023
Published: 2 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

As one of the key ways to achieve next-generation communication networks, ultra-dense small cell networks (UDSCNs) can increase network capacity and coverage, and reduce energy consumption. However, UDSCNs have serious interference problems due to complicated environments. Interference control and resource allocation for UDSCNs are current hot research topics, including user association, spectrum resource allocation, and power control. To improve the overall spectral efficiency of the system, real-time power allocation for UDSCNs is an NP-hard problem. Traditional algorithms (such as fractional programming (FP) [1] and the weighted minimum mean squared error (WMMSE) algorithm [2]) are effective but require global CSI; they have very high computational complexity, which is not feasible in practice.

For power allocation problems, deep Q-learning (DQL)-based algorithms [3], such as [4,5], are only suitable for discrete action spaces and have low accuracy. Reinforcement learning optimization algorithms based on the deterministic policy gradient (DDPG) [6], such as [7], are an improvement of DQL for continuous action spaces. References [8,9] use centralized training and distributed execution frameworks, and [10] designed a multiple-actor-shared-critic method to synchronize the actor-network parameters between local and central controllers, which effectively solves the latency problem and has good scalability and realistic feasibility. However, the DDPG algorithm uses action noise for the action space exploration and the convergence is usually not stable due to the dependence on many hyperparameters [11]. For a specific environment, it needs to do a lot of work for tuning

and is difficult to be directly used in another environment, i.e., the fixed hyperparameters can only have a noticeable effect in a specific environment.

The focus of this study is to present a detailed investigation of the mathematical principles involved in power allocation and propose a new algorithm that we believe could be of practical value. In this paper, we propose a deep reinforcement learning optimization framework based on actor–critic architecture for power allocation in UDSCNs, named the policy optimization of the power allocation (POPA) algorithm, which helps to achieve high performance, is scalable, and can be used for various resource allocation problems. In our proposed POPA scheme, the proximal policy optimization (PPO) algorithm [12] is used for power allocation in small cell networks to mitigate interference, improve overall capacity, and meet QoS requirements. The main contributions are summarized as follows:

1. Based on the actor–critic framework, we propose a novel policy optimization algorithm for power allocation (POPA) in small cell networks with distributed execution and centralized exploration training, which is suitable for continuous action spaces and can reduce the computational complexity and storage of SBSs, and ensure that SBSs can obtain power policy in real time.
2. The actor–critic-based policy optimization framework proposed in this paper can work effectively without the need for real-time global CSI; the training convergence effects of networks are fast with high robustness.
3. The optimization framework proposed in this paper is scalable, not only for increasing or decreasing the number of base stations in an environment, but also for deployment in different environments and other wireless networks without adjusting most of the hyperparameters.

The rest of this paper is organized as follows. We provide the system model and the formulation of the power allocation problem in Section 2. We describe the proximal policy optimization algorithm in Section 3 and the proposed power allocation algorithm in Section 4. The simulation results are presented in Section 5. Conclusions and discussion are given in Section 6.

2. System Model and Problem Formulation

In this paper, we consider a typical network structure with the power allocation problem, consisting of n areas with an SBS in the center of each area serving the user equipment (UE), as shown in Figure 1. The system model considers downlink interference, the SBSs share the same spectrum among themselves, and the UE is equipped with a single antenna. The SBSs are connected via backhaul links to a centralized controller with high computational and storage capacity.

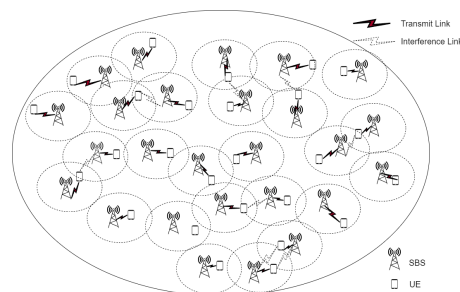


Figure 1. Ultra-dense small cell network environment.

2.1. Channel Model

Let $N = \{1, 2, \dots, n\}$ be the set of area indices; the indices of SBSs and UEs are the same as the area to which they belong. The channel gain g_{ij}^t , $i, j \in N$ from SBS_i to UE_j in time slot t consists of large-scale fading β_{ij}^t and small-scale fading h_{ij}^t :

$$g_{ij}^t = \beta_{ij}^t |h_{ij}^t|^2 \tag{1}$$

where large-scale fading accounts for path loss and shadow fading and does not vary over long time slots. The small-scale fading is a first-order complex Gauss–Markov process according to Jakes’ channel model [13]:

$$h_{ij}^t = \rho h_{ij}^{t-1} + \sqrt{1 - \rho^2} \delta_{ij}^t \tag{2}$$

The variables h_{ij}^0 and δ_{ij}^t are independently and identically distributed complex Gaussian random variables with unit variance, i.e., they follow $\mathcal{CN}(0, 1)$. The correlation factor $\rho = J_0(2\pi f_d T_s)$, where $J_0(\cdot)$ is the first class zero-order Bessel function, f_d is the maximum Doppler frequency, and T_s is the period of the time-slotted system.

2.2. Problem Formulation

The signal-to-interference-plus-noise ratio (SINR) of the link from SBS_i to UE_i in time slot t is a function of the power allocation policy $P^t = \{p_1^t, p_2^t, \dots, p_n^t\}$:

$$SINR_i^t = \frac{g_{ii}^t p_i^t}{\sum_{j \in N, j \neq i} g_{ji}^t p_j^t + \sigma^2} \tag{3}$$

where σ^2 is the variance of additive Gaussian white noise (AGWN).

The spectral efficiency of SBS_i in time slot t is

$$C_i^t(P^t) = \log_2(1 + SINR_i^t) \tag{4}$$

and the total spectral efficiency of the system is

$$C^t(P^t) = \sum_{i \in N} C_i^t(P^t) \tag{5}$$

With the limit of the power range of each SBS, the optimization problem is

$$\begin{aligned} & \max_{P^t} C^t(P^t) \\ & s.t. 0 \leq p_i^t \leq p_{max}, \forall i \in N \end{aligned} \tag{6}$$

where p_{max} is the maximum power of the SBSs. The optimization problem is non-convex and NP-hard [14], and each SBS must receive the power allocation scheme at the beginning of each time slot t . Model-based optimization algorithms, such as FP and WMMSE, have extremely high computational complexity and require real-time global CSI, which is not feasible in practice.

3. Proximal Policy Optimization Algorithm

3.1. Overview of Deep Reinforcement Learning

The agent and environment are key elements of reinforcement learning. The agent observes the environment and decides on an action to take at each step of the interaction. The environment changes when the agent acts upon it and may also change on its own. The outcome of the agent’s action on the environment is a reward, which measures the positive or negative effect of the agent’s action on the environment. DRL [15] combines deep learning and reinforcement learning to adapt to complex environments and achieve better results.

Most current deep reinforcement learning-based power allocation algorithms for wireless networks use DQL and DDPG [6] algorithms, which belong to the Q-learning branch. They require training to obtain an approximation of the action value function, which indirectly optimizes the agent’s performance. They are mostly off-policy, meaning that data from any time during training can be used at each update, regardless of how the agent chooses the action when acquiring the data. Various Q-learning-based policy

optimization algorithms optimize the parameters of the policy directly by gradient ascent on the performance optimization objective, and are on-policy, using only the data collected during the last operations. The core idea of the policy gradient algorithm is to increase the probability of actions that bring more rewards and decrease the probability of actions that bring fewer rewards until the best policy is reached.

Q-learning-based algorithms can only indirectly optimize agent performance by training a value network. It is also noteworthy that the state of the wireless network environment is generally not strongly correlated with the choice of action, i.e., user movement, changes in channel state, etc., are largely unrelated to power allocation. This rapidly changing and complicated environment often makes it difficult to estimate the action value function, and old historical training data is not guaranteed to optimize the training. It is worth noting that the DDPG algorithm is an algorithm that combines Q-learning with policy optimization by obtaining actions through a deterministic policy network and can be regarded as a DQL algorithm that operates in the continuous action space and relies on the action value function in its core. As DDPG is an off-policy method, exploration noise can cause sudden failures in unstable environments [11,16]. In comparison, the main advantage of policy optimization methods is that they optimize directly for the optimization objective and are usually more stable and reliable [17].

3.2. Proximal Policy Optimization Algorithm

The proximal policy optimization (PPO) algorithm [12] is designed to optimize the policy network as much as possible while using the current data in a short period of time, reducing the complexity of the operation, and it ensures that each policy update has a relatively small deviation from the previous one. The PPO algorithm strikes a balance between pattern complexity, ease of implementation, and tuning.

The PPO consists of an actor network and a critic network. The input of the actor network in time slot t is s^t , which is the state of the environment, and the output is the power allocation policy. The input of the critic network is the same as the actor network and the output value $V(s^t)$ is an estimate of the cumulative reward R^t for the last T time slots. The cumulative reward is used for the training of the critic network, which is computed as

$$R^t = r^t + \gamma r^{t+1} + \gamma^2 r^{t+2} + \dots + pow(\gamma, T - t + 1) r^{T-1} \tag{7}$$

where γ is the discount factor and r^t is the reward of the current action.

The advantage function represents the performance of the current action relative to the old policy average. Actions must be executed in an environment of T time slots (usually very short) using the old actor-network parameters for each policy update step. The data during each period are collected to compute the estimate of the policy advantage function. The advantage function is estimated using the widely used method proposed in [18], which is

$$A^t = \chi^t + (\gamma\lambda)\chi^{t+1} + (\gamma\lambda)^2\chi^{t+2} + \dots + pow(\gamma\lambda, T - t + 1)\chi^{T-1} \tag{8}$$

where $\chi^t = r^t + \gamma V(s^{t+1}) - V(s^t)$ is the action advantage value in time slot t and λ is the generalized advantage estimation parameter.

The PPO algorithm calculates the gradient of the policy gradient objective $L_{PPO}(\theta)$ and uses a gradient ascent algorithm for policy optimization. In this paper, we use the policy gradient objective of the clip variants [12]:

$$L_{PPO}(\theta) = E_t[\min(\varphi^t(\theta)A^t, clip(\varphi^t(\theta), 1 - \epsilon, 1 + \epsilon)A^t)] \tag{9}$$

where

$$clip(\varphi^t(\theta), 1 - \epsilon, 1 + \epsilon) = \min(\max(\varphi^t(\theta), 1 - \epsilon), 1 + \epsilon) \tag{10}$$

and

$$\varphi(\theta) = \frac{\pi_{\theta}(a^t|s^t)}{\pi_{\theta_{old}}(a^t|s^t)} \tag{11}$$

where a^t is the action for the environment in time slot t , $\pi_{\theta}(a^t|s^t)$ is the probability of obtaining action a^t from the current actor network, and $\pi_{\theta_{old}}(a^t|s^t)$ is the probability of obtaining the same action from the old actor network.

An alternative form of the policy gradient objective is

$$L_{PPO}(\theta) = E_t[\min(\varphi^t(\theta)A^t, f(\varepsilon)A^t)] \tag{12}$$

where

$$f(\varepsilon) = \begin{cases} 1+\varepsilon, & A \geq 0 \\ 1-\varepsilon, & A < 0 \end{cases} \tag{13}$$

The hyperparameter ε is typically 0.2 or 0.1, which limits the range of probability ratios. The PPO algorithm ensures that each policy update is controlled within a certain range by restricting the optimization objective, and the low complexity of the algorithm can meet the conditions for use in environments with high real-time requirements.

4. The Proposed Power Allocation Algorithm

4.1. Framework Architecture

The policy optimization of the power allocation algorithm (POPA) proposed in this paper is shown in Figure 2 with a distributed execution and centralized exploration training architecture, which ensures that the SBSs can obtain the policy in real time. The computation of the reward function is not required in the distributed execution phase, which reduces the computational pressure on the SBSs. The POPA can achieve the global optimization effect while each SBS only uses the local CSI.

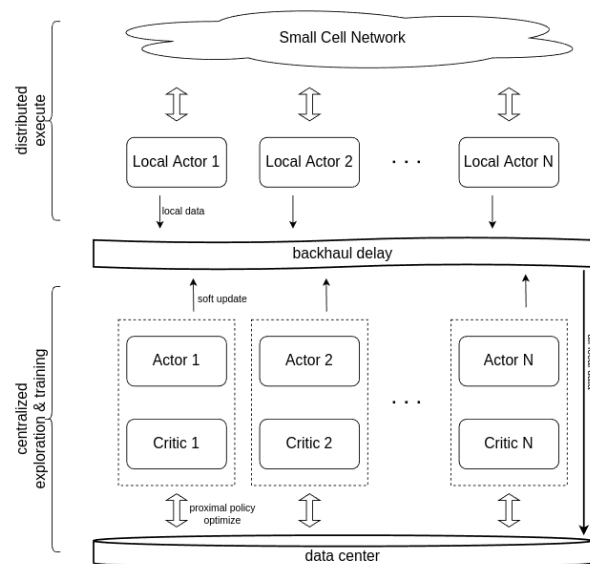


Figure 2. Policy optimization of the power allocation (POPA) algorithm architecture.

Figure 3 is an illustration of the actor–critic in the POPA. The actor network is used as a policy network for the PPO algorithm, where the input is state s^t of the environment and the output is the power allocation policy. The centralized controller contains a data center to store the historical data of the ultra-dense network and sets up a critic network for each SBS and an actor network with the same parameters as the local actor network. For the critic network, the input is state s^t of the environment and the output is the value estimate, i.e., the cumulative reward value estimate $V(s^t)$.

For the discrete action space, the output of the actor network is the probability of each possible action. However, quantifying power as a discrete value would reduce the

accuracy of the power allocation for the power allocation problem. In order to make the actor network operate on the continuous action space, in this paper, the output of the actor network is set to the mean *mean* and variance *var* of the two-parent normal distribution of the truncated normal distribution *mean*. The lower and upper bounds of this truncated normal distribution are 0 and 1, respectively, and the power allocation policy is determined by multiplying the maximum power p_{max} of the SBS by the value chosen from the distribution.

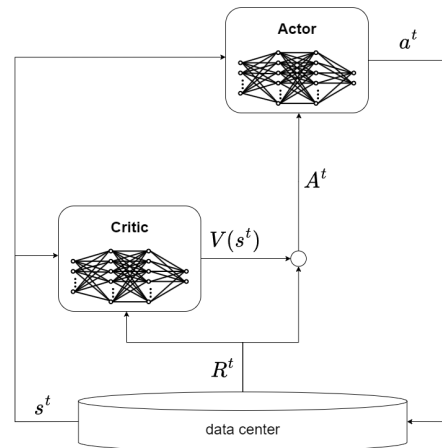


Figure 3. An illustration of the actor–critic in the POA.

The SBS obtains the best action by the power allocation policy of the local actor network at each time slot and acts in the environment before the next time slot starts to achieve the effect of real-time power control, while transmitting the local state information s^t (local data) to the centralized controller and storing it in the data center through the backhaul link. After receiving the state information, the centralized controller obtains the power allocation policy from the actor network and selects action a^t and its probability according to the distribution of the policy. Then, it calculates the reward value r^t based on the global information to complete the action space exploration and stores it in the data center for policy optimization. It is worth noting that the policy output by the actor implies the probability of power allocation, and the local actor can directly choose the action with the highest probability without random selection for the sake of action space exploration. The calculation of the exploration reward value for the action space is done by the central controller, which reduces the computational pressure on the local actor to ensure real-time performance. This also allows the reward calculation to have global state information without requiring cooperation between base stations.

The central controller performs PPO on the actor and critic networks based on the data from the data center with a time step T as a period. It then transmits the parameters of the actor network with the best reward effect to the corresponding local actor network for soft updating of the parameters every T_u period. This ensures that the local actor-network parameters are the same as the centralized training of the latest actor-network parameters, and are the same as those of the centrally trained actor network.

4.2. Process of Proximal Policy Optimization

The central controller needs to perform pre-training preparation after each local data acquisition and completion of action space exploration by feeding state s^t to the critic network to obtain the cumulative reward value estimate $V(s^t)$ and store it in the data center. After obtaining the closest T data sets according to the time t , the controller performs parameter training optimization for the actor network ($iter_a$) and the critic network ($iter_c$).

For the actor network, the advantage function A^t is computed first; in each training process, state s^t is fed into the latest actor network to obtain $\pi(a^t|s^t)$, from which the gradient of the PPO objective $L_{PPO}(\theta)$ is computed; according to this gradient, the gradient

ascent optimization algorithm is carried out. This paper uses the Adam optimizer [19] for training.

For the critic network, the cumulative reward value R^t is first calculated and used as the target value, and the estimated value $V(s^t)$ is trained $iter_c$ times for the critic network using the gradient descent algorithm to make its output close to the true cumulative reward value.

4.3. Algorithm Design

4.3.1. State and Action Space

For practical feasibility, the algorithm proposed in this paper only uses the local CSI as the environment state and does not require any information exchange between the SBSs. The SBSs must receive the power allocation action a_t from $actor_i$ before the arrival of the time t . Therefore, state s_i^t consists of the information of the environment in time slot $t - 1$:

$$s_i^t = \{p_i^{t-1}, g_{ii}^{t-1}, \sum_{j \in N, j \neq i} g_{ji}^{t-1} p_j^{t-1}, I_i^{t-1}\} \tag{14}$$

where I_i^t is the set of interfering powers, consisting of the top c interfering powers, ordered from highest to lowest according to the interfering power received by UE i . If the number of interfering base stations is less than c , the empty spaces are filled with zero (padding). Thus, state s_i^t consists of $3 + c$ elements. The action space is the power assigned to the SBSs in time slot t :

$$a_i^t = \{p_i^t\} \tag{15}$$

4.3.2. Reward Function

After the local data are transferred to the central controller's data center, the reward for each agent can be calculated based on the global information:

$$r_i^t = \frac{C^t(P^t)}{n} \tag{16}$$

4.3.3. Action Selection Distribution

In this paper, a truncated normal distribution is used as the action selection distribution, whose truncated allowed range is the power range $[0, p_{max}]$ for each SBS, and the power allocation action is selected according to this distribution. In time slot t , the output of $actor_i$ is the mean $mean_i^t$ and variance var_i^t of its bi-parental normal distribution. The probability density function of its action selection distribution is as follows; when $0 \leq x \leq p_{max}$:

$$pdf_i^t(x) = \frac{1}{\sqrt{var_i^t}} \frac{\phi(\frac{x - mean_i^t}{\sqrt{var_i^t}})}{\Phi(\frac{p_{max} - mean_i^t}{\sqrt{var_i^t}}) - \Phi(\frac{0 - mean_i^t}{\sqrt{var_i^t}})} \tag{17}$$

where

$$\phi(\zeta) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\zeta^2\right) \tag{18}$$

is the probability density function of the standard normal distribution, and $\Phi(\cdot)$ is its cumulative distribution function. For the other cases, $pdf_i^t(x) = 0$.

4.4. Framework Scalability

The framework proposed in this paper focuses on multidimensional scalability. Firstly, the state and action space of each agent are the same, which is still applicable to multi-layer heterogeneous networks, and the base stations of different layers can be uniformly indexed, making it scalable in terms of network heterogeneity. Second, when the number of base stations increases, similar to the scalability on the aspect of network heterogeneity, the

corresponding actor–critic network can be expanded, and the framework can also be applied by treating each user as an agent when base stations serve multiple users, which allows scalability on the aspect of the number of base stations and users with only a few modifications. Finally, for other similar environments, such as power allocation for cellular networks, software-defined networks, and other wireless networks, the power allocation algorithm of this policy optimization framework can be applied without much tuning, which is scalable in terms of multiple environments. Compared with DQL- and DDPG-based algorithms, the optimization of our proposed POPA is more stable and portable.

5. Simulation

5.1. Simulation Setup

Small cell networks with n SBSs were simulated. Unless otherwise stated, the parameters for the simulation are given in Table 1. For every SBS, the maximum power P_{max} is 30 dBm over the 10 MHz frequency band. In the simulation, we obtain the distance-dependent path loss by $120.9 + 37.6 \log_{10}(d)$ (in dB) following the LTE standard, where d is the SBS-to-UE distance in km. The AGWN power σ^2 is -144 dBm and the log-normal shadowing standard deviation is 8 dB. The SBS-to-SBS distance is set at 20 m, and we also define an inner area within a radius of 3 m from each SBS, where no UE is allowed to be placed. In addition to these parameters, we set the time-slotted system period T_s to 20 ms and the maximum Doppler frequency f_d to 10 Hz for all UEs.

As shown in Figure 4, we use the same architecture for the actor and critic networks, which have one input layer, two hidden layers, and one output layer. The input layers of both the actor and critic networks consist of $3 + c$ neurons, and both hidden layers contain 64 neurons. The difference lies in the fact that the output layer of the actor network has two units that represent the mean and variance of the action selection distribution, while the output layer of the critic network has only one neuron that represents the cumulative reward value estimate. The activation function of the output layer is sigmoid, and the rectified linear unit (ReLU) is adopted in the hidden layers. The output is linearly aligned to the power range. The Adam algorithm [19] is adopted as the optimizer; the learning rate for the actor network is lr_a and the critic network is lr_c . By default, the actor and critic networks are trained 50 times per 100 time slots, and networks that have the best average episodic reward ever will be used for distributed execution.

Table 1. Simulation Parameters.

Parameters	Values
n	16
c	5
ε	0.2
T	100
γ	0.99
λ	0.97
$iter_a$	50
$iter_c$	50
lr_a	0.001
lr_c	0.001
T_u	20 ms
hidden layers	(64, 64)

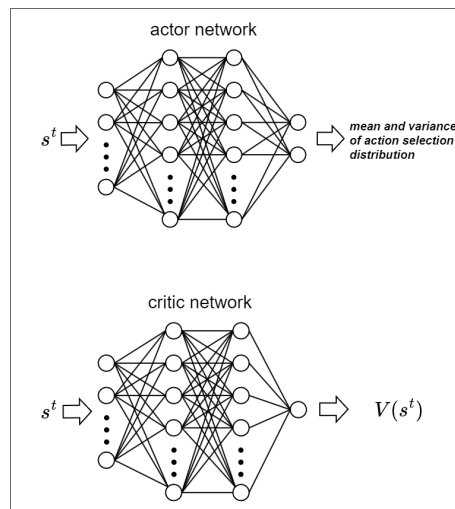


Figure 4. The actor and critic networks for the POPA.

5.2. Clip Range

The clip range ϵ in the POPA ensures that the actor network updates in a trust region and has better convergence during training. Figure 5 shows the episodic average rewards during training with different clip ranges. As shown in the results, a clip range that is too large can result in uncontrolled policy updates and trap the actor network in local optimization. On the other hand, a clip range that is too small may cause the actor network to plateau without significant improvement. We chose a default clip range of 0.2, which balances the convergence effect.

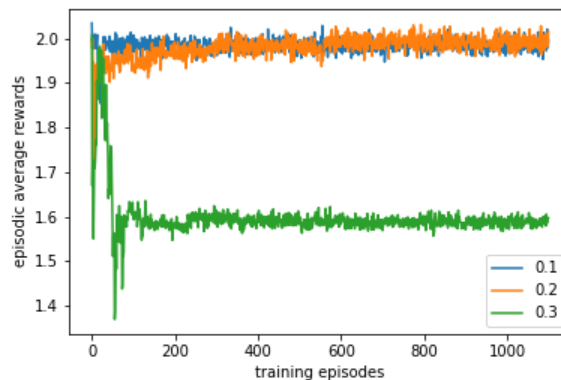


Figure 5. Episodic average rewards while training with different clip ranges.

5.3. Algorithm Comparison

The DDPG [6], ideal FP algorithm [1], random power, and maximum power are chosen as benchmarks, and the results are shown in Figure 6. We compare the performance of DDPG and PPO algorithms using identical settings, such as the same state and action space, reward function, model, and training parameters as specified in the simulation setup. Compared to random and maximum power approaches, DDPG demonstrates a significant improvement in performance. However, it falls behind when compared to the POPA, which is based on the PPO algorithm. The ideal FP algorithm has the best performance, but it needs full real-time CSI and has extremely high computational complexity, which is not feasible in practice. Our proposed POPA algorithm has shown good performance while only requiring local CSI; the execution time for an SBS to obtain power allocation is very short.

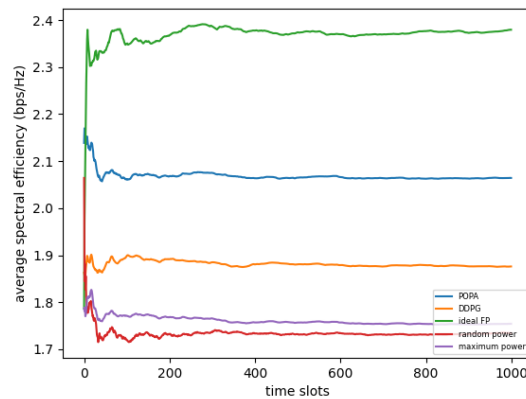


Figure 6. Average spectral efficiency (bps/Hz) in 1000 time slots.

Figure 7 shows the average executions per second for each algorithm in our simulation environment with an Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz, 16 GB RAM, and a GeForce RTX 3060 LHR GPU. The POPA has an average execution time of 1.46 ms, which means it can be executed approximately 684 times per second. In comparison, DDPG has an average execution time of 1.43 ms, which means it can be executed approximately 657 times per second. The maximum and random power have average execution times of 1.24 ms and 1.28 ms, respectively, which means they can be executed approximately 909 and 869 times per second, respectively. However, the ideal FP takes 15.60 ms for the execution and can be executed 64 times per second in the same condition. In our experiments, we found that—compared to the maximum power allocation—for every additional 1% of the execution time, the POPA can achieve about 1% improvement in spectral efficiency, which is a significant improvement in communication systems, while DDPG can achieve a 0.46% improvement and ideal FP can only have a performance improvement of 0.03%. The observation suggests that the POPA algorithm may outperform the other schemes in real experiments or practical scenarios.

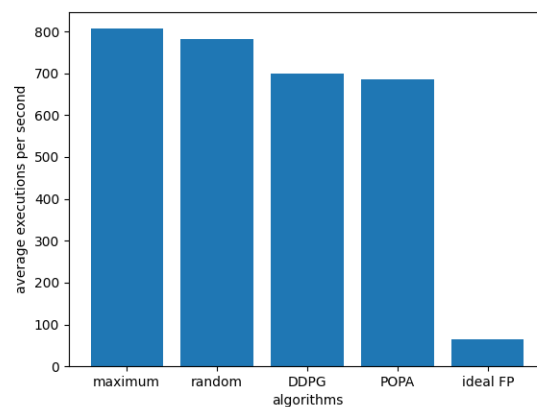


Figure 7. Comparison of the average executions per second.

6. Conclusions

In this paper, we propose a novel policy optimization of the power allocation algorithm (POPA) for small cell networks based on an actor–critic framework. The POPA requires only local CSI and low computational complexity, making it a practical solution for power allocation in small cell networks. Based on simulations, the POPA outperforms benchmark algorithms, such as DDPG, random power, and maximum power. Moreover, the POPA is able to achieve the same level of performance with a much lower computational complexity; only local CSI is required with a smaller amount of loss in terms of the network throughput compared to the ideal FP algorithm. Specifically, the POPA has an average execution time of

1.46 ms, which is close to the DDPG average execution time of 1.43 ms, and is comparable to the maximum power and random power (1.24 ms and 1.28 ms, respectively). Additionally, the POPA is approximately 91% faster than the ideal FP algorithm (15.60 ms).

We believe that our proposed algorithm has the potential to significantly improve the performances of small cell networks and enable efficient and practical power allocation. Future work could explore the performance of the POPA in real-world experiments and its potential for further optimization. Moreover, additional research could investigate the scalability and adaptability of the POPA in various network scenarios and deployment scenarios.

In summary, the framework design of the POPA enables good performance and multi-dimensional scalability, making it a promising solution for power allocation in small cell networks. We hope that our research will inspire further investigations into the application of the actor–critic frameworks and policy optimization algorithms in the field of wireless communication networks.

Author Contributions: Conceptualization, H.C., Z.H., X.Z. and X.L.; Methodology, H.C., P.G., Y.J. and G.Y.; Software, H.C., Z.H. and X.Z.; Writing-original draft preparation, H.C.; Writing-review and editing, H.C. and Z.H.; Supervision, Y.C. and D.W.; Funding acquisition, Y.C. and D.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Natural Science Foundation of Shandong Province under grant ZR2020MF004 and in part by the National Key R&D Program of China under grant 2020YFC0833201.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the anonymous reviewers for the constructive and valuable comments, which helped us improve this paper to its present form.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shen, K.; Yu, W. Fractional Programming for Communication Systems—Part I: Power Control and Beamforming. *IEEE Trans. Signal Process.* **2018**, *66*, 2616–2630. [[CrossRef](#)]
2. Shi, Q.; Razaviyayn, M.; Luo, Z.Q.; He, C. An Iteratively Weighted MMSE Approach to Distributed Sum-Utility Maximization for a MIMO Interfering Broadcast Channel. *IEEE Trans. Signal Process.* **2011**, *59*, 4331–4340. [[CrossRef](#)]
3. 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. Human-Level Control through Deep Reinforcement Learning. *Nature* **2015**, *518*, 529–533. [[CrossRef](#)] [[PubMed](#)]
4. Zhang, Y.; Kang, C.; Ma, T.; Teng, Y.; Guo, D. Power Allocation in Multi-Cell Networks Using Deep Reinforcement Learning. In Proceedings of the 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), Chicago, IL, USA, 27–30 August 2018; pp. 1–6. [[CrossRef](#)]
5. Nasir, Y.S.; Guo, D. Multi-Agent Deep Reinforcement Learning for Dynamic Power Allocation in Wireless Networks. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 2239–2250. [[CrossRef](#)]
6. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous Control with Deep Reinforcement Learning. *arXiv* **2019**, arXiv:1509.02971.
7. Zhang, T.; Zhu, K.; Wang, J. Energy-Efficient Mode Selection and Resource Allocation for D2D-Enabled Heterogeneous Networks: A Deep Reinforcement Learning Approach. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 1175–1187. [[CrossRef](#)]
8. Meng, F.; Chen, P.; Wu, L.; Cheng, J. Power Allocation in Multi-User Cellular Networks: Deep Reinforcement Learning Approaches. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 6255–6267. [[CrossRef](#)]
9. Sinan Nasir, Y.; Guo, D. Deep Actor-Critic Learning for Distributed Power Control in Wireless Mobile Networks. In Proceedings of the 2020 54th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 1–5 November 2020; pp. 398–402. [[CrossRef](#)]
10. Zhang, L.; Liang, Y.C. Deep Reinforcement Learning for Multi-Agent Power Control in Heterogeneous Networks. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 2551–2564. [[CrossRef](#)]
11. Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; Meger, D. Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
12. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
13. Dent, P.; Bottomley, G.; Croft, T. Jakes Fading Model Revisited. *Electron. Lett.* **1993**, *29*, 1162. [[CrossRef](#)]

14. Luo, Z.Q.; Zhang, S. Dynamic Spectrum Management: Complexity and Duality. *IEEE J. Sel. Top. Signal Process.* **2008**, *2*, 57–73. [[CrossRef](#)]
15. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [[CrossRef](#)]
16. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
17. Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Liu, M.; Boning, D.; Hsieh, C.J. Robust deep reinforcement learning against adversarial perturbations on state observations. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21024–21037.
18. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous Methods for Deep Reinforcement Learning. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; pp. 1928–1937.
19. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.