electronics

MDPI

Article

Against Jamming Attack in Wireless Communication Networks: A Reinforcement Learning Approach

Ding Ma *, Yang Wang and Sai Wu

China Electric Power Research Institute, Beijing 100192, China; yangw@epri.sgcc.com.cn (Y.W.); wusai@epri.sgcc.com.cn (S.W.)
* Correspondence: martin_6653@163.com

Abstract: When wireless communication networks encounter jamming attacks, they experience spectrum resource occupation and data communication failures. In order to address this issue, an anti-jamming algorithm based on distributed multi-agent reinforcement learning is proposed. Each terminal observes the spectrum state of the environment and takes it as an input. The algorithm then employs Q-learning, along with the primary and backup channel allocation rules, to finalize the selection of the communication channel. The proposed algorithm designs primary and backup channel allocation rules for sweep jamming and smart jamming strategies. It can predict the behavior of jammers while reducing decision conflicts among terminals. The simulation results demonstrate that, in comparison to existing methods, the proposed algorithm not only enhances data transmission success rates across multiple scenarios but also exhibits superior operational efficiency when confronted with jamming attacks. Overall, the anti-jamming performance of the proposed algorithm outperforms the comparison methods.

Keywords: wireless communication; anti-jamming; reinforcement learning; channel allocation; multi-agent

Citation: Ma, D.; Wang, Y.; Wu, S. Against Jamming Attack in Wireless Communication Networks: A Reinforcement Learning Approach. *Electronics* **2024**, *13*, 1209. https://doi.org/10.3390/electronics13071209

Academic Editor: Dimitra I. Kaklamani

Received: 12 February 2024
Revised: 7 March 2024
Accepted: 11 March 2024
Published: 26 March 2024

1. Introduction

Wireless communication networks are widely used in electric power scenarios such as information acquisition, command transmission, drone patrol, construction site communication, and emergency communication. Due to the inherent openness of the wireless channel, jamming attacks on spectrum resources have a serious impact on the communication security and reliability of these scenarios [1,2]. Jamming attacks disrupt the wireless signals of terminals, making it impossible to recover accurate information [3]. Smart jammers, which have emerged in recent years, can launch even more threatening attacks by learning terminal communication strategies [4]. Many studies on anti-jamming resource allocation methods have achieved certain results and have shown good performance in dealing with conventional jamming attacks. However, most of the considered scenarios are idealized, and there is a large gap between the actual application scenarios and the idealized scenarios [5–7]. There are fewer studies on anti-jamming methods for practical scenarios, and it is necessary to conduct further research on anti-jamming spectrum resource allocation schemes.

With the rapid integration of artificial intelligence technology and wireless communication, significant breakthroughs have been made in applying AI technology to communication anti-jamming. Intelligent anti-jamming methods have emerged as an important research direction, leveraging techniques such as machine learning, deep learning, and reinforcement learning to identify jamming attacks and make anti-jamming decisions [8–10]. The author in [11] proposes a software and hardware platform based on long short-term memory networks and Bayesian network models to identify the types of jamming attacks on 5G systems. The author in [12] proposes a jamming attack method based on generative

adversarial networks to accelerate attack efficiency and also points out that the terminal can deliberately make erroneous spectrum selections to deal with such attacks. In practical communication scenarios, there are often dynamic changes in the environmental spectrum and difficulties in predicting the strategies of jammers, which lead to challenges in training and applying traditional machine learning and deep learning methods. Since reinforcement learning does not require prior knowledge or labeled data, it has the advantage of autonomous optimization through interaction with the environment. It is more suitable for dynamic scenarios, where data acquisition is difficult. The characteristics of reinforcement learning have made it an important method in the field of anti-jamming research in recent years. Reference [13] is based on reinforcement learning, which allows each terminal to not only predict its own spectrum selection but also predict the spectrum selection of other terminals and then reselect actions to avoid interference between terminals. Reference [14] achieves the maximization of the communication rate in the jamming attack scenario in two steps. First, it uses multi-agent deep reinforcement learning to obtain the optimal strategy for power control and beamforming and then uses federated deep reinforcement learning to make decisions on the maximum communication rate in the jamming attack scenario. In reference [15], the anti-jamming problem of the non-contiguous orthogonal frequency division multiplexing system is studied. To solve the problem of large state–action space, a Q-learning algorithm based on confidence intervals is proposed to solve the exploration–exploitation dilemma. The algorithm perceives the wireless environment and comprehensively considers the jamming and deep fading effects during transmission. It ultimately selects the channels and transmission rates to be used. Reference [16] establishes a non-zero-sum game model between transmitters and smart jammers. The author proposes two anti-jamming schemes: the DQN-based scheme aims to select the transmission channel and power, while the hierarchical learning scheme is used to obtain the optimal strategies for the Nash equilibrium. In reference [17], the electronic radar jamming countermeasure is studied, and an intelligent radar anti-jamming decision-making method is proposed. The decision is generated based on DDPG and MADDPG algorithms. The two algorithms perform hierarchical decision-making and joint optimization, which speeds up the convergence speed and enhances the anti-jamming effect. The use of neural networks to obtain Q values in DDPG and MADDPG means that the parameters of the neural network need to be stored in memory, and a large number of forward and backward propagation calculations are required during the training process. Therefore, algorithm implementation is difficult and requires a large amount of computing and storage resources. However, most of the existing algorithms mentioned have complex models and high hardware requirements. To ensure good algorithm performance, not only a large amount of time is needed for training, but also the communication equipment needs to have strong computing and storage capabilities. Most power communication terminals cannot meet the needs of complex algorithms in terms of performance, making it challenging for existing algorithms to be applied in power communication scenarios.

To address the problem of collaborative anti-jamming in power multi-terminal communication scenarios, taking into account the computational and processing capabilities of power communication terminals, this article proposes a low-complexity anti-jamming algorithm. The main contributions of this work are summarized as follows:

- We take a power emergency communication business as the scenario and study the anti-jamming communication problem from the perspective of channel selection. Then, a system communication model that includes multiple terminals and jammers is established. We assume terminals and jammers do not know each other's communication strategies. We also discuss the characteristics of sweep jamming attacks and smart jamming attacks, which serve as the basis for the design of the anti-jamming algorithm.
- To counter sweep and smart jamming attacks, we propose a distributed multi-agent primary and backup channel allocation Q-learning algorithm (DMPBQ). Terminals using DMPBQ can predict jammer strategies and negotiate channel selection with each other by following the primary and backup channel allocation rules. Consequently,

terminals are able to select adequate communication channels. Through theoretical analysis, the complexity of the proposed algorithm is found to be comparable to independent Q-learning. The simulation results show that compared with existing anti-jamming algorithms, the proposed algorithm has faster operational efficiency, better adaptability to different scenarios, and superior anti-jamming performance.

The rest of this article is organized as follows: In Section 2, we consider the scenario of a power emergency communication business and construct an ad hoc network model composed of multiple terminals and jammers with spectrum sensing capabilities. Subsequently, based on the Q-learning algorithm, we propose a distributed multi-agent primary and backup channel allocation Q-learning algorithm to address the anti-jamming problem. Section 4 shows the specifics of simulations and discussions. Finally, this work is summarized in Section 5.

2. System Model

Power emergency communication is used to establish temporary communication networks when the communication infrastructure is nonexistent or damaged, providing communication services for power emergency repair, operation and maintenance, etc. It often adopts wireless ad hoc network technology. The application environment of the network is complex, and the communication effectiveness is related to the safety of life and property, so it needs to have strong communication anti-jamming ability. Figure 1 shows the system model for the power emergency communication scenario considered in this article, which includes a wireless ad hoc network composed of multiple terminals and jammers.

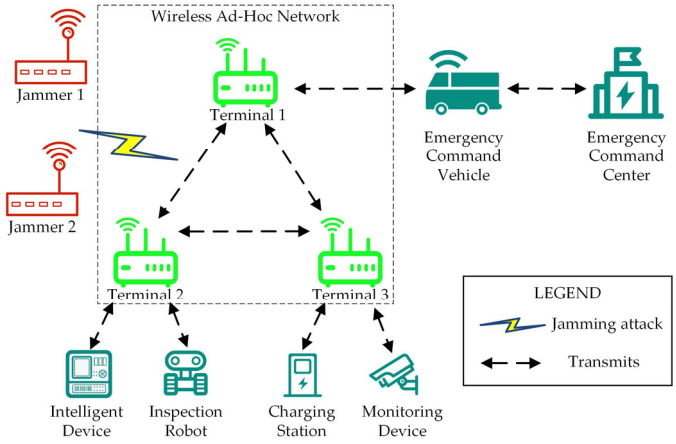

Figure 1. The considered system model.

In wireless ad hoc networks, each terminal selects an available channel for communication based on the current spectrum state and its own strategy. Jammers will launch jamming attacks on the communication network by deploying jamming devices. Each jammer independently selects the channel to jam based on its own strategy and sends jamming signals on that channel to disrupt normal communication. The terminal communication process will fail in two cases. The first is interference, which occurs when different terminals choose the same channel to transmit data at the same time, causing signals from each terminal to interfere with each other, resulting in failed communication for all terminals on that channel. The second is jamming, when a terminal selects a channel that is also chosen by a jammer, resulting in successful jamming and a failed communication attempt. The channel selection strategy of the terminal directly affects the communication success rate.

A system model is established, taking the jamming attack scenario shown in Figure 1 as an example. The system model contains N terminals $\mathcal{N} = \{1, 2, \ldots, N\}$ and M jammers $\mathcal{M} = \{1, 2, \ldots, M\}$. The available spectrum resources in the system are divided into C channels $\mathcal{C} = \{1, 2, \ldots, C\}, C \geq M + N$, which are shared among terminals and jammers.

The system's operation time is divided into equal time slots. The terminal senses the usage of the spectrum in every time slot and then determines whether the channel is occupied by the terminal or the jammer. To better evaluate the performance of different anti-jamming strategies, we assume that each terminal needs to transmit data in every time slot, and it only selects one channel for transmission. Furthermore, assuming that jammers are energy-constrained, each jammer only chooses one channel as the target of jamming attacks in the same time slot. Each terminal will record whether the data transmission is successful in every time slot, and if interference or jamming is detected, it will record the failure of that data transmission attempt.

To compare the effectiveness of our anti-jamming algorithm with reference [13], two types of jamming strategies are considered in this article, namely sweep jamming and smart jamming. The sweep jamming strategy considered in this article switches the channels to be jammed at each time slot and cyclically jams each available channel in the system for a certain period of time. A jammer using the smart jamming strategy, which is based on the Q-learning algorithm, has the ability to learn. It outputs the target channel to jam based on that strategy. The algorithm state is a set containing all terminals' selected channels. The algorithm action is to select a target channel to jam. The reward function is defined as follows:

$$R_j(t) = \begin{cases} 1, \text{ jamming successful} \\ 0, \text{ jamming fails} \end{cases}. \quad (1)$$

3. Algorithm Design

3.1. Preliminaries

Reinforcement learning, as a paradigm of machine learning, focuses on using reward mechanisms to guide agents in learning how to take effective actions in a specific environment to achieve predefined goals. Within this framework, an agent refers to a machine entity that performs learning and decision making, while the environment encompasses all interacting objects external to the agent. At each discrete time slot, the environment can be abstracted as a state representation. After observing the current state, the agent selects and executes an action based on its policy, which triggers a transition in the environment and receives a reward. This reward constitutes the core objective for the agent to optimize its policy, namely to maximize the accumulated rewards over the long-term interaction. The mathematical foundation and modeling tool for reinforcement learning is the Markov decision process. When the future state of the environment only depends on the current state and the action taken by the agent, without any dependence on past states and actions, it satisfies the Markov property. We can formalize the interaction process between the agent and the environment as MDP.

A standard MDP model is represented as $< S, A, R, P, \gamma >$, which consists of the state space, action space, reward function, state transition probability function, and discount factor. The state space refers to the set of all possible states of the environment. The action space represents the collection of all possible actions that the agent can take. The reward function calculates the numerical value returned to the agent by the environment after the agent performs an action. The state transition function describes the process and probabilities of transitioning from the current state to a future state, often assumed to be stochastic due to the randomness inherent in the environment. The discount factor is used to discount future rewards, balancing the importance of immediate rewards with those in the future.

Multi-agent reinforcement learning refers to the scenario where multiple agents interact independently with a shared environment. Each agent aims to improve its own policy by leveraging the rewards provided by the environment, ultimately seeking to achieve higher rewards. In multi-agent systems, the policies of individual agents cannot solely rely on their own observed states and actions but must also take into account the states and actions observed by other agents. There exist four primary relationships among agents in multi-agent systems: cooperation, competition, mixed cooperation–competition, and ego-

ism. Cooperation implies that agents share a common goal and collaborate to maximize the overall reward. Competition, on the other hand, involves agents with conflicting interests, where the gain of one agent results in a loss for others. Mixed cooperation–competition occurs when agents are organized into groups, with cooperation within groups but competition between groups. Lastly, egoism denotes agents that solely focus on their own interests, seeking to maximize their own rewards regardless of the benefits or losses incurred by other agents.

3.2. Algorithm Model

The behavior of terminals and jammers is shown in Figure 2. Terminals are the only entities regarded as agents, and all terminals and jammers are in the same environment. They obtain the channel state information of the environment through spectrum sensing. After observing the state of the environment, terminals and jammers will choose the communication channel to use or attack according to their own strategy. Then, all the actions taken by terminals and jammers will interact with the environment, causing changes in the environment. Finally, the feedback from the environment will indicate whether the data transmission or jamming attack is successful. It should be noted that, in this article, we divide the selection of terminal communication channels into two issues for research, namely policy learning and channel allocation. Policy learning enables terminals to learn the attack strategies of jammers and predict attack behaviors based on the state of the environment. Channel allocation is based on policy learning and further selects the communication channels based on prediction results.

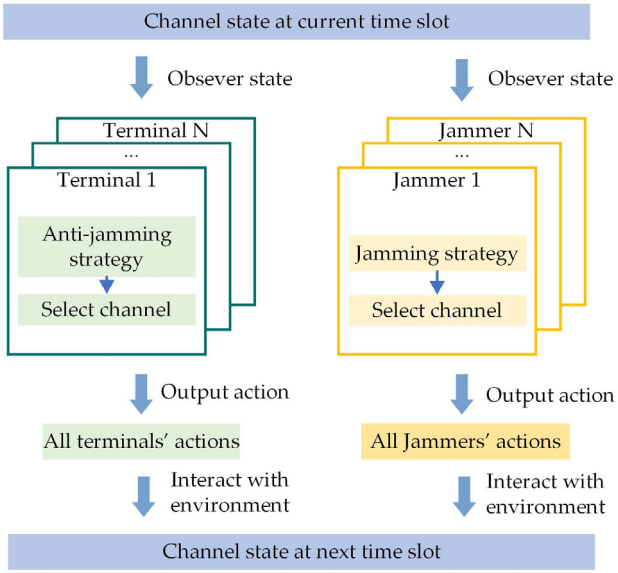

Figure 2. The behavior of terminals and jammers.

The environmental state in the system model shown in Figure 2 depends only on the current time slot and the actions taken by all terminals, which is consistent with the Markov property, so the problem studied in this article is consistent with the Markov decision process theory. Due to the existence of multiple agents in the system, each agent needs to observe the state and take actions to collaboratively counteract the jamming attacks. Since this system model aligns with the cooperative paradigm in multi-agent systems, a multi-agent reinforcement learning algorithm should be used to address the problem. The algorithm model is shown below:

- State space S: In slot t, the observed state of the terminal i is defined as $s_i(t) = \{u_1(t), u_2(t), \ldots, u_N(t), j_1(t), j_2(t), \ldots, j_M(t)\}, s_i(t) \in S$, where $u_k(t)$, $k \in \mathcal{N}$ represents the channel currently selected by the terminal k, and $j_l(t)$, $l \in \mathcal{M}$ represents the channel currently selected by the jammer l.

- Action space A: The action space of the terminal i is defined as $A = \mathcal{C}$. In the slot t, the actions taken by the terminal i in the two stages of policy learning and channel selection are defined as $a_i(t) \subset A$, $a'_i(t) \in A$, where $a_i(t)$ represents which channels that the terminal i predicts will be attacked by jammers, and $a'_i(t) = u_i(t+1)$ represents the channel that terminal i ultimately selects for data transmission.
- Reward: In slot t, the terminal i predicts jamming attacks based on the observed state and receives a certain reward to adjust its policy learning process. There are two reward functions defined in this article. The first reward function is used to deal with sweep jamming, which can be expressed as

$$R_p = \begin{cases} 1, \text{ predict successful} \\ 0, \text{ predict fails} \end{cases}. \tag{2}$$

When the terminal successfully predicts all jammed channels, the reward value is 1, and otherwise, the reward value is 0. The second reward function is used to deal with smart jamming, which can be expressed as

$$R_s = \begin{cases} 1, \text{ channel is not jammed} \\ 0, \text{ channel is jammed} \end{cases}. \tag{3}$$

When the channel selected by the terminal is not jammed, the reward value is 1, and in other cases, the reward value is 0.

- State transition probability function P: The probability that the state of the environment transitions from $s_i(t)$ to $s_i(t+1)$. Due to the presence of jammers and the unknown jamming strategies, the state transition probability $P(s_i(t), s_i(t+1))$ cannot be known in advance.

3.3. Anti-Jamming Algorithm Based on Distributed Multi-Agent Reinforcement Learning

When using multi-agent reinforcement learning methods to study anti-jamming problems, it is necessary for each terminal in the environment to be able to quickly take action after observing the state of the environment. The Q-learning algorithm is simple to implement and does not consume excessive storage and computing resources, making it suitable for the problem scenario studied in this article. If all terminals in the environment independently use the standard Q-learning algorithm to select communication channels, there will be serious conflicts between terminals. The solution is to use the concept of communication in multi-agent reinforcement learning, allowing all terminals to share the same Q-table. However, in jamming scenarios, there is no absolutely secure communication channel, and the shared Q-table approach can significantly slow down the algorithm convergence rate. This article proposes a distributed multi-agent Q-learning algorithm for primary and backup channel allocation to address these issues.

3.3.1. Sweep Jamming Scenario

The sweep jamming considered in this article is a typical representative of conventional jamming. The DMPBQ algorithm in the sweep jamming scenarios is shown in Algorithm 1. Each terminal runs this algorithm independently. The core idea of Algorithm 1 is to allow each terminal to independently predict the channels to be jammed and to pre-formulate the channel allocation rule for all terminals. In Algorithm 1, the terminal i can determine the channel $a_{i,x}(t)$ that will be selected by the terminal with primary channel x. Therefore, each terminal can avoid jamming attacks and select a communication channel that is different from other terminals.

Algorithm 1: DMPBQ (Sweep Jamming)

Input state of terminal i as $s_i(t)$, total number of terminals as N, total number of jammers as M, primary channel of terminal i as c_i, backup channel resource pool as $\mathbb{P}$
Output the channel finally selected by the terminal i as $a'_i(t)$
Initialize total operating time of the system as T
While $t < T$ **do**
  For $i = 1, 2, \ldots, N$ **do**
    If random number $< \varepsilon$ **then**
      $a_i(t) = \underset{a}{\text{argmax}} Q_i(s_i(t), a)$
    Else
      Random choose an action
    End if
    If $c_i \notin a_i(t)$ **do**
      $a'_i(t) = c_i$
    Else
      Remove $a_i(t)$ from $\mathbb{P}$
      For $x \in a_i(t)$ **do**
        Select the channel $a_{i,x}(t) \notin a_i(t)$ with the smallest number from $\mathbb{P}$
        Remove $a_{i,x}(t)$ from $\mathbb{P}$
        If $i = x$ **then**
          $a'_i(t) = a_{i,x}(t)$
        End if
      End for
    End if
  End for
  $s_i(t+1) = [a'_1(t), a'_2(t), \ldots, a'_N(t), j_1(t), j_2(t), \ldots, j_M(t)]$
  For $i = 1, 2, \ldots, N$ **do**
    Update Q table
  End for
  $t = t + 1$
End while

The preconditions for the primary and backup channel allocation rules in Algorithm 1 are as follows: Each terminal that accesses the network obtains a terminal identification code, which is denoted as i for ease of discussion. It also obtains a primary channel $c_i = i$ and a backup channel resource pool $\mathbb{P}$. Terminals with different primary channels share the same available spectrum resources and the same backup channel resource pool.

In the jamming prediction stage, the terminal i takes action based on the ε-greedy strategy, observes the current state $s_i(t)$, queries the Q table to output the predicted action $a_i(t)$, and then enters the channel selection stage.

In the channel selection stage, the terminal needs to adopt the primary and backup channel allocation rules to determine the final selected channel $a'_i(t)$. The rule is implemented through the following steps:

1. The terminal i compares the predicted jammed channels with the allocated primary channel c_i. If the primary channel is not subject to the predicted jammed channels, then it selects the primary channel for the next data transmission. If the primary channel is determined to be subject to predicted jammed channels during the next data transmission, proceed with steps 2–4;
2. If the predicted jammed channels of the terminal i are in the backup channel resource pool, then the corresponding channels are removed from the backup channel resource pool;
3. We can infer that any terminal using one of the predicted jammed channels as its primary channel will suffer from jamming attacks, naming them jammed terminals. The terminal i sorts all the jammed terminals according to their terminal identification codes from small to large and allocates the channel with the smallest number from

the backup channel resource pool to every jammed terminal in turn. Each channel can only be assigned to one jammed terminal until the terminal i is allocated to an available channel, or there are no remaining available channels in the backup channel resource pool.

4. Repeat steps 1–3 for each terminal in the system.

After the above steps, the terminal i uses the channel $a'_i(t)$ for data transmission and obtains the corresponding reward according to the reward function R_p. After all terminals and jammers interact with the environment, the terminal i observes the state $s_i(t+1)$. The Q table is further updated by the following Equation (4), where α represents the learning rate, and γ represents the discount factor.

$$Q_i(s_i(t), a_i(t)) = (1-\alpha)Q_i(s_i(t), a_i(t)) + \alpha[r_i(t) + \gamma \max_{a_i(t)} Q_i(s_i(t+1), a_i(t))] \quad (4)$$

Further analyzing the complexity of Algorithm 1, the complexity of the jamming prediction stage depends on the size of the state space and action space, represented as $O(|S|\cdot|A|)$. The complexity of the channel selection stage depends on the number of jammed channels, which in this case is the number of jammers, represented as $O(M)$. Therefore, the complexity of algorithm 1 for every terminal is given by Equation (5).

$$G_1 = (O(|S|\cdot|A|) + O(M)) \quad (5)$$

Due to $O(M) \ll O(|S|\cdot|A|)$, the complexity of Algorithm 1 $G_1 \sim O(|S|\cdot|A|)$ is comparable to the standard Q-learning algorithm, with high operational efficiency.

3.3.2. Smart Jamming Scenario

The DMPBQ algorithm in the smart jamming scenario is shown in Algorithm 2. It is also run independently by each terminal without inter-terminal communication. The crux of Algorithm 2 lies in assigning non-overlapping channel resource pools to each terminal. Modeling jammers' behavior and introducing randomness to channel selection can enhance the difficulty for jammers to learn, thereby countering smart jamming attacks effectively. The algorithm proceeds as follows:

1. The terminal i will obtain a dedicated channel resource pool $\mathbb{P}'_i$ after accessing the network. At every slot, the terminal i independently maintains two Q tables, Q_i^1 and Q_i^2, where Q_i^1 is used to learn the jamming attack strategies of the jammers and output the predicted jamming channels. Q_i^2 is used to output the primary channel $c_i(t)$ for communication based on the observed state;
2. The terminal i compares the output results of the Q table mentioned above, and if $c_i(t) \notin a_i(t)$, the terminal i selects the channel $a'_i(t) = c_i(t)$ for data transmission; otherwise, it randomly selects a channel $a'_i(t)$ that does not belong to $a_i(t)$ from $\mathbb{P}'_i$ for data transmission.

After all terminals and jammers interact with the environment, the terminal i observes the state $s_i(t+1)$ and receives rewards based on R_p, R_s. Then, each terminal i uses Equation (5) to update its two Q tables.

Algorithm 2: DMPBQ (Smart Jamming)

Input state of terminal i as $s_i(t)$, total number of terminals as N, total number of jammers as M, primary channel of terminal i at t as $c_i(t)$, backup channel resource pool as $\mathbb{P}'_i$
Output the channel finally selected by the terminal i as $a'_i(t)$
Initialize total operating time of the system as T
While $t < T$ **do**
  For $i = 1, 2, \ldots, N$ **do**
    $a_i(t) = \underset{a}{\text{argmax}} Q_i^1(s_i(t), a)$ $c_i(t) = \underset{a}{\text{argmax}} Q_i^2(s_i(t), a)$
    If $c_i(t) \notin a_i(t)$ **do**
      $a'_i(t) = c_i(t)$
    Else
      Remove $a_i(t)$ from $\mathbb{P}'_i$
      Random select a channel from $\mathbb{P}'_i$
    End if
  End for
  $s_i(t+1) = [a'_1(t), a'_2(t), \ldots, a'_N(t), j_1(t), j_2(t), \ldots, j_M(t)]$
  For $i = 1, 2, \ldots, N$ **do**
    Update Q table
  End for
  $t = t + 1$
End while

To further analyze the complexity of Algorithm 2, the complexity of Algorithm 2 proposed for each terminal is given by Equation (6).

$$G_2 = 2 \times O(|S| \cdot |A|) \tag{6}$$

The algorithm complexity $G \sim O(|S| \cdot |A|)$ is comparable to the standard Q-learning algorithm, and it also has high operational efficiency.

4. Performance Evaluation

In this section, we evaluate the anti-jamming performance of the proposed DMPBQ in both sweep and smart jamming scenarios through simulation experiments. We compare it with the CCQ algorithm and RANDOM algorithm proposed in [13], as well as the independent Q-learning algorithm. The numerical settings for the learning rate and discount factor of the algorithm are the same as those in [13]. We used the packet transmission success rate μ to evaluate the anti-jamming performance of each algorithm, which is defined as the proportion of successfully transmitted packets out of the total transmitted packets. We assumed that each simulation lasts 10,000 time slots, and each terminal transmits one packet in every time slot. Every 20 packets form a block, and we calculate the packet transmission success rate of each block. The simulation experiments were independently performed 250 times, and the average packet transmission success rate μ' of each algorithm was calculated and evaluated. The following simulation experiment was conducted using Python 3.10 on a hardware device with an Intel Core i7-7700HQ 2.8 GHz processor and 8 GB of memory. This algorithm is implemented at the MAC layer of the terminal communication protocol and can be executed without the need for additional hardware equipment.

4.1. Evaluation under Sweep Jamming Scenarios

We considered the anti-jamming performance of Algorithm 1 and the comparison algorithms in a sweep jamming scenario with five channels, where different numbers of terminals and jammers were present. It was assumed that jammers adopt the sweep jamming strategy. We chose the learning rate $\alpha = 0.8$ and discount factor $\gamma = 0.6$. The epsilon in the ε-greedy strategy gradually increased from 0 to 1 as the training process progressed.

Figure 3 shows the anti-jamming performance in the scenario with two terminals and one jammer. The results showed that, after 50 blocks of packet transmission, the DMPBQ,

CCQ, and independent Q-learning algorithms all exhibited strong anti-jamming performance. Among them, DMPBQ had the fastest algorithm convergence speed, achieving an average packet transmission success rate of 0.99 after 25 blocks of transmission and remaining stable. When comparing the average packet transmission success rate for 500 blocks of packets, DMPBQ demonstrated a marginally superior anti-jamming performance than CCQ and independent Q-learning, and a significantly superior performance than RANDOM. The DMPBQ could achieve such a high average packet transmission success rate because each terminal could accurately learn the same jammer's strategy. Additionally, the primary and backup channel allocation rules ensured that each terminal did not experience channel selection conflicts.

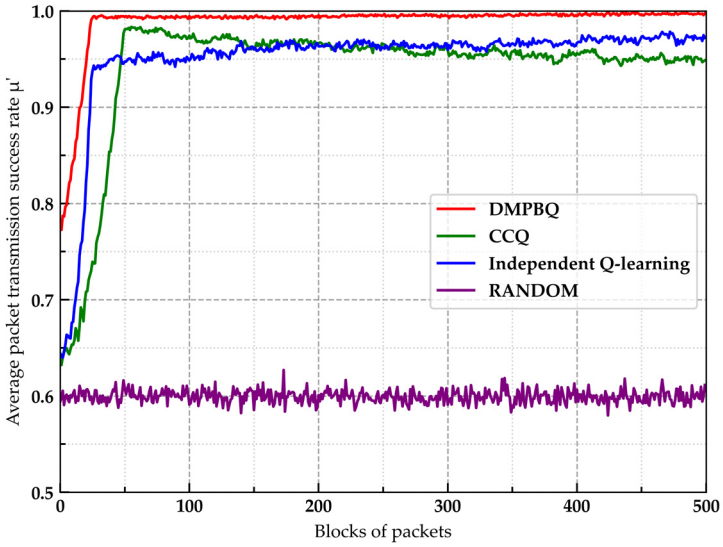

Figure 3. The anti-jamming performance of various algorithms in a sweep jamming scenario with 2 terminals and 1 jammer.

Figure 4 shows the anti-jamming performance in the scenario with four terminals and one jammer. When the number of available channels was limited, as the number of terminals increased, the probability of collision between terminals increased, leading to a significant drop in the anti-jamming performance of CCQ, independent Q-learning, and RANDOM algorithms. In this case, the average packet transmission success rate dropped below 0.7. All terminals using the DMPBQ reached a consensus in predicting the behavior of jammers. By using the primary and backup channel allocation rules, it was possible to prevent collisions between terminals, thus maintaining a stable average packet transmission success rate of 0.98. This demonstrates good anti-jamming performance in complex multi-terminal scenarios.

Figure 5 shows the anti-jamming performance in the scenario with three terminals and two jammers. When the number of available channels was limited, and there were multiple jammers, the average packet transmission success rates of all algorithms decreased. The reason for this is that multiple jammers increase the likelihood of terminals being jammed. After the transmission of 50 blocks of packets, DMPBQ and independent Q-learning stabilized the average packet transmission success rate at 0.9, with DMPBQ performing slightly better than independent Q-learning. The average packet transmission success rate of CCQ initially increased but then gradually decreased as the number of blocks transmitted increased, ultimately reaching approximately 0.8. In this scenario, the average packet transmission success rate of the RANDOM algorithm was only about 0.5.

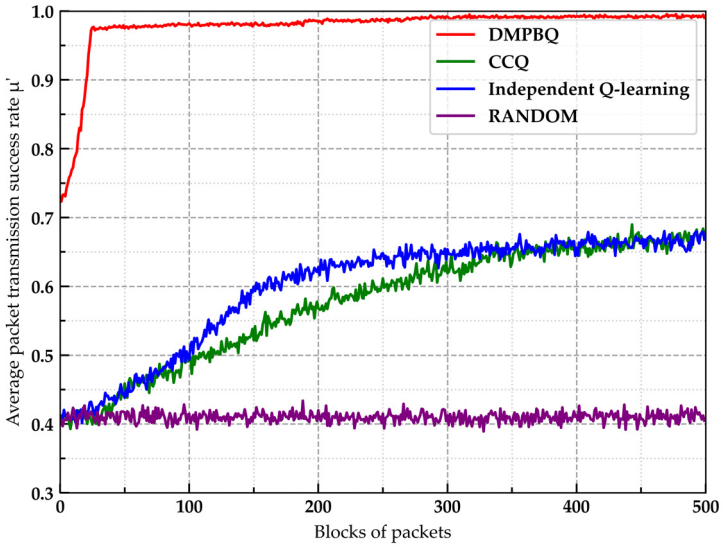

Figure 4. The anti-jamming performance of various algorithms in a sweep jamming scenario with 4 terminals and 1 jammer.

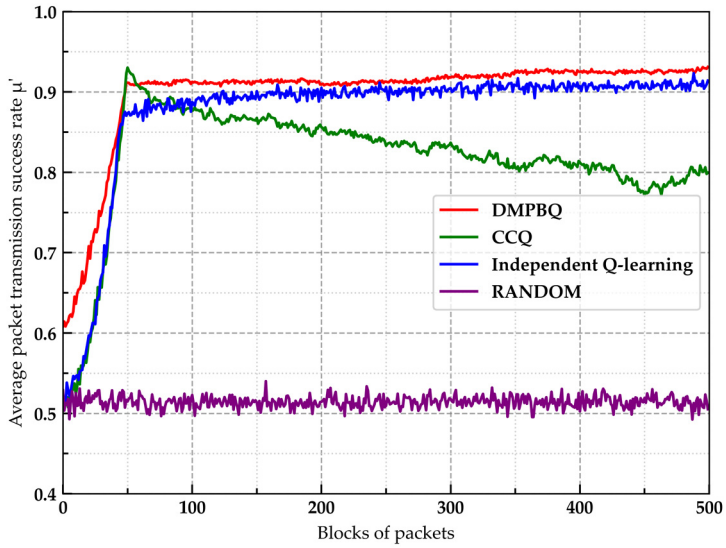

Figure 5. The anti-jamming performance of various algorithms in a sweep jamming scenario with 3 terminals and 2 jammers.

Table 1 shows the time required for each algorithm to transmit 5000 blocks of packets in the sweep jamming scenario. This running time is the sum of the time consumed by each terminal and jammer. Due to the impact of processor performance, we only compared the running time among algorithms. The results showed that the running speed from fast to slow was in the following order: RANDOM, independent Q-learning, DMPBQ, and CCQ. The running time of DMPBQ increased by 13% compared to independent Q-learning but decreased by 45% compared to CCQ, indicating a high operational efficiency of DMPBQ. In summary, DMPBQ had the best anti-jamming performance in sweep jamming scenarios. Both theoretical analysis and experiments demonstrate that the running time of the algorithm is comparable to independent Q-learning, which has high operational efficiency. Even with an increase in terminals, DMPBQ can still maintain good anti-jamming performance.

Table 1. The running time of each algorithm in a sweep jamming scenario with 2 terminals and 1 jammer.

Algorithm	Running Time (Seconds)
DMPBQ	127
CCQ	232
Independent Q-learning	112
RANDOM	1

4.2. Evaluation under Smart Jamming Scenario

We assumed that the smart jamming scenario included six channels, three terminals, and one jammer. We compared the anti-jamming performance of Algorithm 2 with other algorithms. Jammers adopted the jamming attack strategy based on Q-learning, and the rest of the parameters were set the same as those in the sweep jamming scenarios.

Figure 6 shows that when dealing with smart jammers, the anti-jamming performance of each algorithm is ranked from high to low as follows: DMPBQ, CCQ, independent Q-learning, and RANDOM. The average packet transmission success rate of DMPBQ was higher than that of the other three comparison algorithms, stabilizing at 0.84 from the outset. This is because DMPBQ divides each terminal's available channel to prevent conflicts during channel selection among them. It also uses the primary and backup channel allocation rules to increase randomness when the terminals choose the primary channel. DMPBQ poses a challenge for jammers in learning the strategy of the terminals, thereby reducing the effectiveness of the jammers' attack model.

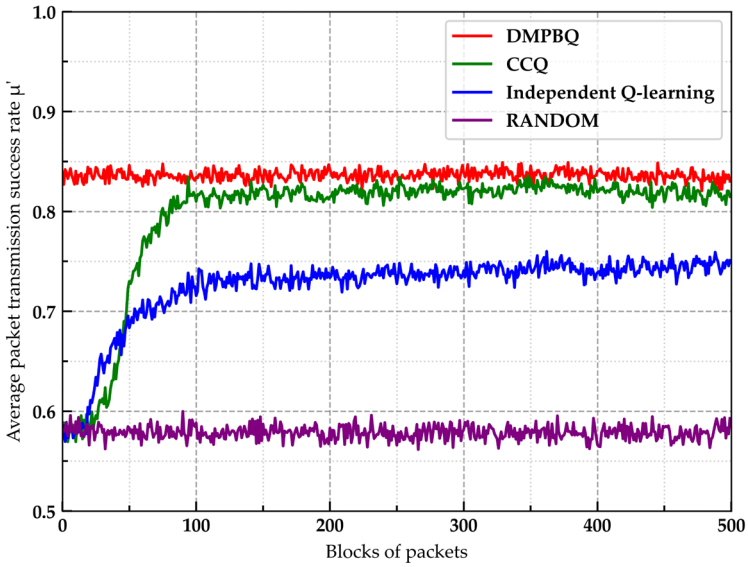

Figure 6. The anti-jamming performance of various algorithms in a smart jamming scenario with 3 terminals and 1 jammer.

Table 2 shows the time required for each algorithm to transmit 5000 blocks of packets in the smart jamming scenario. This time is also the sum of the time consumed by each terminal and jammer. It can be seen that the ranking of the algorithm running time is consistent with that in the sweep jamming scenario. The running time of DMPBQ increased by 12% compared to independent Q-learning but decreased by 48% compared to CCQ. DMPBQ demonstrated high operational efficiency even when confronted with smart jamming attacks. Through theoretical analysis and experiments, it can be demonstrated that the operational efficiency of DMPBQ in the smart jamming scenario is also comparable to independent Q-learning, and it has good anti-jamming performance.

Table 2. The running time of each algorithm in a smart jamming scenario with 3 terminals and 1 jammer.

Algorithm	Running Time (Seconds)
DMPBQ	355
CCQ	684
Independent Q-learning	316
RANDOM	79

5. Conclusions

In this article, we considered the challenging problem of anti-jamming in wireless communication networks and proposed an anti-jamming algorithm called DMPBQ based on reinforcement learning. The proposed algorithm divides the available channels into primary channels and backup channels for selection. We designed channel selection rules for both sweep jamming and smart jamming to minimize the impact of jamming attacks. The simulation results show that, compared with existing algorithms, the packet transmission success rate of DMPBQ can reach up to 0.99 in the sweep jamming scenario and 0.84 in the smart jamming scenario. DMPBQ not only has good anti-jamming performance but also has high algorithm operational efficiency.

Author Contributions: D.M.: methodology, experiments, writing—original draft preparation. Y.W.: supervision, writing—review and editing. S.W.: methodology, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Project of SGCC (State Grid Corporation of China), grant number 5700-202218439A-2-0-ZN.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

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

Abbreviations

The following abbreviations are used in this manuscript:

MDP	Markov decision process
DMPBQ	Distributed multi-agent primary and backup channel allocation Q-learning
CCQ	Cross-check Q-learning

References

1. Hu, S.; Chen, X. Modeling and Analysis of Energy Harvesting and Smart Grid-Powered Wireless Communication Networks: A Contemporary Survey. *IEEE Trans. Green Commun.* **2020**, *4*, 461–496. [CrossRef]
2. Hamamreh, J.M.; Furqan, H.M. Classifications and Applications of Physical Layer Security Techniques for Confidentiality: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1773–1828. [CrossRef]
3. Pirayesh, H.; Zeng, H. Jamming Attacks and Anti-Jamming Strategies in Wireless Networks: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 767–809. [CrossRef]
4. Shi, Y.; Sagduyu, Y.E. How to Attack and Defend NextG Radio Access Network Slicing With Reinforcement Learning. *IEEE Open J. Veh. Technol.* **2023**, *4*, 181–192. [CrossRef]
5. Yao, F.; Jia, L. A Collaborative Multi-Agent Reinforcement Learning Anti-Jamming Algorithm in Wireless Networks. *IEEE Wirel. Commun. Lett.* **2019**, *8*, 1024–1027. [CrossRef]
6. Bi, Y.; Wu, Y.; Hua, C. Deep Reinforcement Learning Based Multi-User Anti-Jamming Strategy. In Proceedings of the 2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019.
7. Örnek, C.; Kartal, M. Securing the Future: A Resourceful Jamming Detection Method Utilizing the EVM Metric for Next-Generation Communication Systems. *Electronics* **2023**, *12*, 4948. [CrossRef]
8. Lu, X.; Xiao, L. Reinforcement Learning-Based Physical Cross-Layer Security and Privacy in 6G. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 425–466. [CrossRef]
9. Bout, E.; Loscri, V. How Machine Learning Changes the Nature of Cyberattacks on IoT Networks: A Survey. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 248–279. [CrossRef]

10. Han, D.; Li, A. Deep Learning-Guided Jamming for Cross-Technology Wireless Networks: Attack and Defense. *IEEE ACM Trans. Netw.* **2021**, *29*, 1922–1932. [CrossRef]
11. Wang, Y.; Jere, S.; Banerjee, S. Anonymous Jamming Detection in 5G with Bayesian Network Model Based Inference Analysis. In Proceedings of the 2022 IEEE 23rd International Conference on High Performance Switching and Routing (HPSR), Taicang, China, 6–8 June 2022.
12. Erpek, T.; Sagduyu, Y.E. Deep Learning for Launching and Mitigating Wireless Jamming Attacks. *IEEE Trans. Cogn. Commun.* **2019**, *5*, 2–14. [CrossRef]
13. Elleuch, I.; Pourranjbar, A. A Novel Distributed Multi-Agent Reinforcement Learning Algorithm Against Jamming Attacks. *IEEE Commun. Lett.* **2021**, *25*, 3204–3208. [CrossRef]
14. Sharma, H.; Kumar, N. Mitigating Jamming Attack in 5G Heterogeneous Networks: A Federated Deep Reinforcement Learning Approach. *IEEE Trans. Veh. Technol.* **2023**, *72*, 2439–2452. [CrossRef]
15. Yuan, X.; Yu, L. Joint Sub-Band and Transmission Rate Selection for Anti-Jamming Non-Contiguous Orthogonal Frequency Division Multiplexing System: An Upper Confidence Bound Based Reinforcement Learning Approach. *Electronics* **2023**, *12*, 4418. [CrossRef]
16. Li, Y.; Wang, J.; Gao, Z. Learning-Based Multi-Domain Anti-Jamming Communication with Unknown Information. *Electronics* **2023**, *12*, 3901. [CrossRef]
17. Wei, J.; Wei, Y.; Yu, L.; Xu, R. Radar Anti-Jamming Decision-Making Method Based on DDPG-MADDPG Algorithm. *Remote Sens.* **2023**, *15*, 4046. [CrossRef]