*International Journal of Sensors and Sensor Networks*
2024, Vol. 12, No. 1, 2024, pp. 1-12
https://doi.org/10.11648/j.ijssn.20241201.11

Research Article

# Duty Cycle Scheduling in Wireless Sensor Networks Using an Exploratory Strategy-Directed MADDPG Algorithm

**Liangshun Wu[1], Peilin Liu[1], Junsuo Qu[2], Cong Zhang[2], Bin Zhang[3, 4, 5, *]**

[1]School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China
[2]Xi'an Key Laboratory of Advanced Control and Intelligent Process, School of Automation, Xi'an University of Posts & Telecommunications, Xi'an, China
[3]Political Science and Law, Xinjiang University, Tumushuke, China
[4]Department of Computer, City University of Hong Kong, Hongkong, China
[5]School of Remote Sensing and Information Engineering, Wuhan University, Wuhan, China

## Abstract

This paper presents an in-depth study of the application of Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithms with an exploratory strategy for duty cycle scheduling (DCS) in the wireless sensor networks (WSNs). The focus is on optimizing the performance of sensor nodes in terms of energy efficiency and event detection rates under varying environmental conditions. Through a series of simulations, we investigate the impact of key parameters such as the sensor specificity constant α and the Poisson rate of events on the learning and operational efficacy of sensor nodes. Our results demonstrate that the MADDPG algorithm with an exploratory strategy outperforms traditional reinforcement learning algorithms, particularly in environments characterized by high event rates and the need for precise energy management. The exploratory strategy enables a more effective balance between exploration and exploitation, leading to improved policy learning and adaptation in dynamic and uncertain environments. Furthermore, we explore the sensitivity of different algorithms to the tuning of the sensor specificity constant α, revealing that lower values generally yield better performance by reducing energy consumption without significantly compromising event detection. The study also examines the algorithms' robustness against the variability introduced by different event Poisson rates, emphasizing the importance of algorithm selection and parameter tuning in practical WSN applications. The insights gained from this research provide valuable guidelines for the deployment of sensor networks in real-world scenarios, where the trade-off between energy consumption and event detection is critical. Our findings suggest that the integration of exploratory strategies in MADDPG algorithms can significantly enhance the performance and reliability of sensor nodes in WSNs.

## Keywords

Multi-Agent Systems, Deep Reinforcement Learning, MADDPG, Wireless Sensor Networks, Energy Efficiency, Event Detection, Exploratory Strategy

*Corresponding author: bzhang48-c@my.cityu.edu.hk (Bin Zhang)

**Received:** 18 January 2024; **Accepted:** 29 January 2024; **Published:** 28 February 2024

## 1. Introduction

Wireless sensor networks (WSNs) are pivotal in various sectors such as agriculture, environmental monitoring, healthcare, and intelligent transportation, thanks to their ability to monitor and collect environmental data. However, the limited resources of sensor nodes, particularly energy, pose significant challenges to network efficiency and longevity.

Effective energy management is crucial for maintaining network performance, as sensor nodes, typically battery-powered, have constrained energy supplies affecting their lifespan and functionality. Efficient node scheduling strategies are therefore essential to enhance network performance and extend node lifetimes. While much research has focused on energy-efficient communication, energy-efficient sensing is less explored, despite its comparable energy consumption. Since sensing often occurs more frequently than communication, it's vital to develop energy-saving strategies for sensing activities. Dynamic events, which are unpredictable and require continuous monitoring, present additional challenges for energy efficiency. Duty cycle scheduling, which alternates between sleep and active states for nodes, is a promising approach to balance energy efficiency with event responsiveness. An adaptive duty cycle scheduling strategy, based on the additive increase/multiplicative decrease (AIMD) rule, can dynamically adjust node activity in response to real-time event detection and energy usage, optimizing both energy consumption and detection capabilities. The AIMD-based duty cycle scheduling strategy is significant as it ensures efficient energy management and network longevity while maintaining detection capabilities. It also offers adaptability to dynamic environments and node distributions, enhancing the effectiveness of node scheduling strategies.

In practical WSNs, the vast and varied distribution of sensing nodes adds complexity to communication and cooperation management. This complexity is intensified by the dynamic nature of node distribution, which can shift with time, environmental changes, and task requirements. Consequently, dynamic node scheduling and optimization are critical. Traditional scheduling algorithms, often static, assume fixed node locations and struggle to adapt to dynamic distributions. In real-world scenarios, such as outdoor or mobile sensing applications, node locations can change due to external factors, necessitating scheduling algorithms that are both flexible and adaptive.

Reinforcement learning algorithms, known for their autonomous learning and performance improvement capabilities, are gaining traction for their adaptability. Their application in WSNs presents new avenues for dynamic node scheduling and optimization. Through an agent-environment interaction model, sensor nodes can autonomously learn and optimize scheduling policies based on environmental feedback, leading to more efficient and adaptable node scheduling. Multi-agent reinforcement learning algorithms are particularly beneficial for addressing node cooperation and communication challenges. Treating each sensor node as an individual agent, these nodes can share information and collaborate within a multi-agent system to enhance overall performance. This approach leverages information sharing for more globally optimized scheduling decisions, boosting network efficiency and stability. The introduction of multi-agent reinforcement learning algorithms enables sensor nodes to make more intelligent cooperation and scheduling decisions in dynamic environments, optimizing WSN performance. The flexibility and intelligence of multi-agent reinforcement learning position it as a leading technique for addressing dynamic node scheduling challenges. Further exploration and optimization of this algorithm are necessary to navigate the complexities of WSNs, enhancing sensing efficiency, extending node lifetimes, and advancing the application of wireless sensor technology across various domains.

The multi-agent deep deterministic policy gradient (MADDPG) method, an advancement of the DDPG algorithm for multi-agent contexts, has shown significant improvements in cooperative and competitive tasks. In WSNs, MADDPG with an exploratory strategy offers several advantages:

1. Adaptability to dynamic environments: WSNs are inherently dynamic, with fluctuating channel conditions, data traffic, and network topologies. MADDPG with exploratory strategies enables nodes to adapt proactively, optimizing performance amidst these changes.
2. Enhanced local information sharing: While nodes in WSNs typically access only local information, exploratory strategies facilitate the sharing of detected information, improving collaborative decision-making and network performance.
3. Self-organization: WSNs are self-organizing, with nodes autonomously joining or leaving. MADDPG with exploratory strategies allows nodes to better adjust to these network dynamics.
4. Robustness in adversarial environments: WSNs may encounter adversarial conditions such as jamming or attacks. Exploratory strategies equip nodes to adapt to such uncertainties, bolstering network robustness.

The main contribution of this work is the application of MADDPG with exploratory strategies in WSNs enhancing adaptability, cooperation, self-organization, and robustness, making it a promising solution for the complex challenges of dynamic node scheduling.

## 2. Related Work

Duty cycle scheduling, which orchestrates active, sleep, and idle listening times, is crucial in sensor networks as it significantly impacts energy consumption. Most scheduling strategies, such as ELECTION [1] and AIMD [2], exploit the energy-saving benefits of sleep mode, with AIMD being a popular rule in congestion control [3, 4]. DANCE [5] refines

AIMD by considering the actions of neighboring nodes, allowing sensors to conserve energy by ceasing activity if the task is already completed nearby. Additionally, controlling the data sampling rate through Kalman filtering can manage the computational and communication demands on the central server [6]. However, current duty-cycle optimization algorithms are predominantly static, with limited research on dynamic optimization.

Multi-agent reinforcement learning (MARL) represents an innovative shift in reinforcement learning. Stone et al. introduced TPOT-RL in 1999 [7], a paradigm that, despite its nonflatness making it incompatible with deep Q-learning's experience replay, has been influential. Foerster et al. [8] addressed this by using importance sampling and fingerprint-modulated value functions to integrate MARL with experience replay. Nguyen et al. [9] explored various MARL challenges, such as nonstationarity and partial observability, while Samvelyan et al.'s StarCraft Multi-Agent Challenge (SMAC) [10] provides a benchmark for progress in the field. Chu et al. [11] developed a scalable, decentralized MARL algorithm for the actor-critic model, and MADDPG, [12] an advancement of DDPG, enables cooperative decision-making among multiple agents in a shared environment.

In the context of WSNs, MARL has been proposed to enhance the intelligence and autonomy of nodes in resource allocation, as demonstrated by Wang et al. [13] in Cognitive Radio-WSNs (CR-WSNs). Zhang et al. [14] provided a selective review of MARL, emphasizing theoretically grounded algorithms. The integration of MARL with WSNs promises to advance the field by enabling more dynamic, efficient, and intelligent sensor networks [15-18]. The investigation into active exploration MADDPG and other Multi-Agent Reinforcement Learning (MARL) methods is anticipated to contribute significantly to duty cycle scheduling [19, 20].

## 3. System Model

Table 1 lists the essential notations used in the model.

***Table 1****. T Notations and symbols.*

| Symbol | Description |
|---|---|
| $W$ | Total working duration |
| $S$ | Total sleep duration |
| $W_k$ | The $k$-th working interval |
| $S_k$ | The k-th sleep interval |
| $n$ | The number of intervals |
| $S_0$ | The initial sleep interval (fixed) |
| $s$ | The average sleep duration per duty cycle |
| $w$ | The average working duration per duty cycle |
| $M$ | The interested 2-D square region |
| $M_x$ | The length of M |
| $M_y$ | The width of $M$ |
| $C(t)$ | The counting process of events in $[0, t]$ |
| $\hat{q}\|[0,t]$ | The expected number of events processed in $[0, t]$ |
| $\Gamma_S$ | The average number of missed events in sleep duration |
| $\Gamma_W$ | The average number of missed events in working duration |
| $\Gamma$ | Event space |
| $\lambda$ | Poisson rate |
| $d$ | The distance between the sensor and the event |
| $r$ | The sensing radius |
| $r_{min}, r_{max}$ | The minimum and maximum sensing radius |
| $\mu$ | The detection accuracy |
| $\xi_1, \xi_2$ | The metrics of detection ability |
| $e$ | The energy consumption for sensing |
| $e_S$ | The energy consumption of sensing in sleep duration |
| $e_W$ | The energy consumption of sensing in working duration |
| $\varepsilon_S$ | The static energy consumption in sleep mode |
| $\varepsilon_W$ | The static energy consumption in working mode |
| $\gamma$ | The coefficient of energy depletion growing with distance |
| $\alpha$ | The changeable power scaling parameter for the sensing circuit |
| $\delta$ | Step-size increasing factor |
| $\theta$ | Step-size diminishing factor |
| $N$ | The number of sensors |
| $\mathrm{Sgn}(\cdot)$ | The sign function |
| $\sigma(\cdot)$ | The logistic sigmoid function |

### 3.1. Basic Assumptions

The model functions within a two-dimensional grid where numerous devices perform various activities. The architecture of the network is non-hierarchical, with each node possessing an identical energy allowance and being of the same type. The placement of the sensor nodes is uniform. The range of their detection can be adjusted. The sensors operate on an independent schedule, with each one deciding its own operational intervals without the need for coordinated timing. While the sensor nodes have the capability to move, their positions and the distances between them are not ascertainable.

## 3.2. Node Distribution

It is presumed that the nodes are evenly dispersed across the area. The sensor can adjust its communication range, denoted by r, which falls within the minimum and maximum limits $[r_{min}, r_{max}]$. This adjustment to the sensing radius is typically achieved by changing the power of the transmission.

## 3.3. Event Arrival

It is assumed that occurrences within the set M are random and independent, conforming to a Poisson distribution with a rate of λ. The expectation is that measurements will be precise, allowing for the immediate identification of all incidents. The likelihood of a sensor with a detection radius $r$ processing q events over a time period t is as follows:

$$\Pr(C(t) = q) = \frac{\exp\{-\lambda_0 \pi r^2 t\}(\lambda_0 \pi r^2 t)^q}{q!} = \frac{\exp\{-\lambda t\}(\lambda t)^q}{q!} \quad (1)$$

where C(t) is the counting process and $\lambda = \lambda_0 \pi r^2$ represents the mean Poisson rate within the peripheral area covered by the sensor. In line with the characteristics of the Poisson process, it is established that

$$E[C(t)] = \hat{q}|_{[0,t]} = \lambda t \quad (2)$$

where $\hat{q}|_{[0,t]}$ denotes the expected number of events handled in $[0, t]$.

An event at a distance of d has the following probability of being detected:

$$\Pr(d) = \begin{cases} 1, & \text{if } d < r_{min} \\ \exp\{-\xi_1(d - r_{min})^{\xi_2}\}, & \text{if } r_{min} < d < r_{max} \\ 0, & \text{if } d > r_{max} \end{cases} \quad (3)$$

where the sensor's physical features determine $\xi_1$ and $\xi_2$, which reflects the feature of attenuation with distance. All events during SLEEP mode are regarded as absent. Let $\Gamma_S$ be the average number of missed events, then

$$\Gamma_S = E[C(S)] = \lambda S \quad (4)$$

The average number of missed events during working hours, $\Gamma_W$, is

$$\Gamma_W = E[C(W)](1 - \Pr(d)) = \lambda W(1 - \Pr(d)) \quad (5)$$

The event space is

$$\Gamma = E[C(W + S)] = \lambda(W + S) \quad (6)$$

The detection accuracy $\mu$ is now defined as a measurable value

$$\mu = 1 - \frac{\Gamma_S + \Gamma_W}{\Gamma} \quad (7)$$

It follows from (3) to (6) that

$$\mu = \begin{cases} \frac{W}{S+W}, d < r_{min} \\ \frac{W}{S+W} \exp\{-\xi_1(d - r_{min})^{\xi_2}\}, r_{min} < d < r_{max} \\ 0, d > r_{max} \end{cases} \quad (8)$$

## 3.4. Energy Depletion

We introduce the model in [16] that:

$$e_W = \gamma(d - r_{min})^{\alpha} + \varepsilon_W \quad (9)$$

where γ and α are sensor-specific constants; in exemplary cases, $\alpha \in [2,4]$; $\varepsilon_W$ specifies the static power, and $\varepsilon_W$ is intrinsically linked to scheduling techniques. Eq. (9) indicates that the operational energy consumption of the sensors is influenced by their distance from each other. When in sleep mode, only the timer and other critical components remain active, viz.

$$e_S = \varepsilon_S \quad (10)$$

$E_S$ represents the static energy consumption. Regular interruptions for waking up and entering sleep mode can influence the static energy consumption, thus $\varepsilon_S$ should also be considered a variable dependent on the scheduling strategy employed.

It is presumed that $\varepsilon_S$ and $\varepsilon_W$ remain fixed, signifying that the static energy usage does not fluctuate. The static energy expenditure while active is generally considered to be higher than during dormant periods, meaning $\varepsilon_W > \varepsilon_S$. This is due to the necessity of running the watchdog and other system applications when the device is operational. The total energy consumption of the sensing circuit is calculated by the weighted sum of these two constants.

$$e = e_S S + e_W W \quad (11)$$

where W and S represent working and sleeping durations. Then

$$E[e] = e_S \frac{S}{S+W} + e_W \frac{W}{S+W} \quad (12)$$

Assign (9) and (10) to (11), we have:

$$E[e] = \varepsilon_S \frac{S}{S+W} + (\gamma(d - r_{min})^{\alpha} + \varepsilon_W) \frac{W}{S+W} \quad (13)$$

### 3.5. Scheduling Model

In DCS, the length of time the sensor remains in sleep mode is adjusted dynamically, taking into account the frequency of events from the previous cycle. Upon transitioning to an active state, the sensor activates its detection unit and then deactivates it when it's time to enter sleep mode again. Although DCS is commonly implemented, fine-tuning the duration of these states remains a complex challenge.

We propose an adaptive approach that AIMD strategy, where the time intervals are adjusted dynamically in response to the observed event data. The equation for this adjustment is as follows:

$$S_k = \begin{cases} S_{k-1} + \delta S_{k-1}, \mathrm{Sgn}(W_{k-1}) = 0 \\ S_{k-1} - \theta S_{k-1}, \mathrm{Sgn}(W_{k-1}) = 1 \end{cases} \tag{14}$$

where $\delta$ and $\theta$ denote the step-size increasing/diminishing factor, $0 \le \delta \le 1$, $0 \le \theta \le 1$, and $\mathrm{Sgn}(\cdot)$ is defined as:

$$\mathrm{Sgn(t)} = \begin{cases} 1, \text{If an event is detected} \\ 0, \text{Otherwise} \end{cases} \tag{15}$$

There is a maintainable balance between event detection and energy usage. For instance, increasing $\delta$ can lead to greater energy conservation, yet it also heightens the risk of failing to detect an event.

### 3.6. Problem Formulation

An optimization problem serves as the final form of the model:

$$\min J = \min\left\{\frac{1}{2}(1-\mu) + \frac{1}{2}\sigma\left(\frac{E[e]}{1000}\right)\right\} \tag{16}$$

We list all the constraints:

s.t.

$$\begin{cases} S = \sum_{k=1}^{n} S_k \\ S_k = \begin{cases} S_{k-1} + \delta S_{k-1}, \mathrm{Sgn}(W_{k-1}) = 0 \\ S_{k-1} - \theta S_{k-1}, \mathrm{Sgn}(W_{k-1}) = 1 \end{cases} \\ W = \sum_{k=1}^{n} W_k \\ \mu = \begin{cases} \frac{\mathrm{W}}{\mathrm{S+W}}, \mathrm{d} < \mathrm{r_{min}} \\ \frac{\mathrm{W}}{\mathrm{S+W}} \exp\{-\xi_1(\mathrm{d} - \mathrm{r_{min}})^{\xi_2}\}, \mathrm{r_{min}} < \mathrm{d} < \mathrm{r_{max}} \\ 0, \mathrm{d} > \mathrm{r_{max}} \end{cases} \\ E[e] = \varepsilon_S \frac{S}{S+W} + (\gamma(d - r_{min})^{\alpha} + \varepsilon_W)\frac{W}{S+W} \\ \varepsilon_W = Const. \\ \varepsilon_S = Const. \\ \varepsilon_W > \varepsilon_S \\ d \le M_x \\ d \le M_y \\ 2 \le \alpha \le 4 \\ 0 \le \delta \le 1 \\ 0 \le \theta \le 1 \end{cases} \tag{17}$$

The optimization objective has shifted from maximizing $\mu$ to minimizing $1 - \mu$. $\sigma(\cdot)$ denotes the logistic sigmoid function, $\sigma(x)$, which reduces the entire number line to a narrow range $[0,1]$.

## 4. MADDPG Solution with Exploratory Strategy

### 4.1. Markov Decision Process

To design a Markov Decision Process (MDP) for optimizing DCS in WSNs, we must define the state space, action space, and reward function for each sensor node agent. Here's a refined description of these components:

#### 4.1.1. State Space

The state space encapsulates the variables that characterize the current environment and system conditions pertinent to AIMD scheduling decisions. For a sensor node, the state space includes:

1) Current energy level of the sensor node $e_t$
2) Event space at the time step $\Gamma_t$
3) Distance to the event $d_t$
4) Time step index, indicating the current decision epoch

Thus, the state space is defined as $S = \{e_t, \Gamma_t, d_t, t\}$.

#### 4.1.2. Action Space

The action space defines the set of possible actions an agent can take at each time step. For sensor nodes employing AIMD rules, the action space comprises the step increase/decrease factors, ($\delta$) and ($\theta$), which adjust sleep and work times. The action space is thus represented as $A = \{\delta, \theta\}$, where $0 \le \delta \le 1$ and $0 \le \theta \le 1$.

#### 4.1.3. Reward Function

The reward function assesses the quality of an agent's actions within a given state. The optimization objective function, $J$, integrates two components: energy consumption and event detection rate. The reward function is designed to balance the trade-off between these two aspects and is expressed as:

$$R(e_t, \Gamma_t, d_t, t|\delta, \theta) = \lambda * \Delta\mu - (1-\lambda) * \Delta e \tag{18}$$

Here, $\Delta e$ represents the gain of energy consumption, $\Delta\mu$ denotes the gain in event capture rate due to the agent's action, and $\lambda$ is a weighting factor that prioritizes one objective over the other. This weight can be dynamically adjusted to ensure the agent learns an effective policy for the specific optimization challenge at hand.

With these definitions, the MDP can be modeled, and reinforcement learning algorithms such as Q-learning, DDPG, or MADDPG can be employed to discover the optimal policy. The goal is to maximize the reward function, identifying the values of $\delta$ and $\theta$ that best balance energy efficiency with a high event capture rate.

## 4.2. MADDPG Algorithm

MADDPG is an algorithm designed for solving problems in multi-agent environments. It is an extension of the DDPG algorithm tailored for multiple interacting agents. MADDPG combines ideas from actor-critic architectures and centralized training with decentralized execution.

### 4.2.1. Critic Network Update

The critic network aims to estimate the action-value function. The update rule for the critic network is given by:

$$L(\theta_i) = \mathbb{E}\left[\left(y_i - Q(s_i, a_i \mid \theta_i)\right)^2\right] \tag{19}$$

where $y_i$ is the target value and $Q(s_i, a_i \mid \theta_i)$ is the critic's prediction.

### 4.2.2. Actor Network Update

The actor network is responsible for determining the optimal policy. The update rule for the actor network is based on the deterministic policy gradient:

$$\nabla_{\theta_i} J(\theta_i) \approx \mathbb{E}\left[\nabla_a Q(s, a \mid \theta_Q)\big|_{s=s_i, a=\mu(s_i \mid \theta_i)} \nabla_{\theta_i}\mu(s \mid \theta_i)\right] \tag{20}$$

where $J(\theta_i)$ is the expected return, $\mu(s \mid \theta_i)$ is the actor's output, and $\theta_Q$ are the parameters of the critic network.

### 4.2.3. Target Network Updates

Both the actor and critic networks use target networks to stabilize training. Target networks are updated with a soft update rule:

$$\theta_{\text{target}} \leftarrow \tau\theta + (1-\tau)\theta_{\text{target}} \tag{21}$$

where $\tau$ is a small constant.

### 4.2.4. Action Selection

The action selection is done by adding exploration noise to the output of the actor network:

$$a_i = \mu(s_i \mid \theta_i) + \mathcal{N} \tag{22}$$

## 4.3. MADDPG Algorithm with an Exploratory Strategy

A MADDPG algorithm with an exploratory strategy involves incorporating mechanisms that allow agents to explore their environment effectively while still exploiting their learned policies (see Figure 1). The steps are as follows:

### 4.3.1. Initialize the Multi-Agent Environment

Define the number of agents, state space, and action space. Initialize the critic and actor networks for each agent (WSN node) with random weights.

### 4.3.2. Exploratory Strategies

We provide two exploratory strategies:

Deterministic Policy with Noise: Each agent's action is determined by its current policy plus some noise for exploration. This noise can be reduced over time as the agent learns. For example, add Ornstein-Uhlenbeck noise to the action output for exploration, which provides temporally correlated exploration suitable for physical control problems.

$\varepsilon$-greedy Policy: With a probability ε, select a random action for exploration, and with a probability 1- $\varepsilon$, select the best action according to the current policy. The value of $\varepsilon$ can be set to decay over episodes to shift from exploration to exploitation gradually.

### 4.3.3. Training Loop

For each episode:

Initialize the environment and get the initial state.

For each time step:

Each agent selects an action based on the current policy and exploration strategy.

Execute actions in the environment and observe the next state and reward.

Store the experience (state, action, reward, next state) in a replay buffer.

Sample a random minibatch of experiences from the replay buffer.

For each agent:

Update the critic by minimizing the loss with the action from the target actor network for the next state.

Update the actor using the sampled policy gradient.

Update the target networks for both the actor and critic using soft updates.

Reduce the exploration noise (for deterministic policy) or ε (for ε-greedy policy) as the agent learns to balance exploration and exploitation.

Continue training for a predetermined number of episodes or until the performance metric reaches a satisfactory level.

### 4.3.4. Post-Training

Evaluate the performance of the trained policy with reduced or no exploration. Fine-tune the policy if necessary.

This MADDPG with exploratory strategy allows agents to learn from their environment while ensuring sufficient exploration. The balance between exploration (learning about the environment) and exploitation (using the learned knowledge) is crucial for the success of the algorithm.

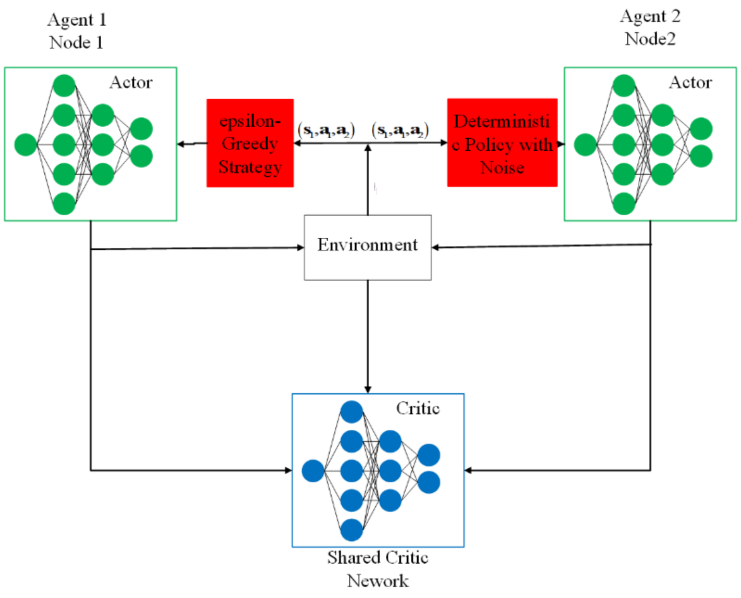

***Figure 1****. The diagram of MADDPG model with an exploratory strategy.*

## 5. Experiments

### 5.1. Experimental Setup

#### 5.1.1. Hardware Platform

The software environment for the simulation is configured with Python 3.7, PyCharm 2021, and TensorFlow 1.14. The hardware setup includes a 12th generation Intel(R) Core (TM) i9-12900K processor with a clock speed of 3.19 GHz and 32GB of RAM.

#### 5.1.2. Parameters Setting

The simulation assumes that sensor nodes adjust their sleep intervals using a duty-cycle scheduling model governed by AIMD rule. The parameters $\delta$ and $\theta$ are within the range [0,1]. Events are modeled to occur following a Poisson process, with the event located at a distance of $d = 5$ meters from the sensor node. The specific model parameters employed in the experiments are detailed in Table 2.

***Table 2****. Simulation parameters.*

| **Parameters** | **Value** |
|---|---|
| $\overline{w}$ | 30 s |
| $r_{\min}$ | 0.5 m |
| $r_{\max}$ | 10 m |
| $M_x$ | 100 m |
| $M_y$ | 100 m |
| $\Gamma$ | 0.7 |
| A | 3 |
| $\varepsilon_W$ | 50 mW |
| $\varepsilon_S$ | 30 mW |
| N | 4 |
| D | 5 m |

Table 3 shows the specific hyperparameter values used in the implementation of MADDPG algorithm. The parameters in question are pivotal in sculpting the learning trajectory and decision-making capabilities of the MADDPG algorithm. The values that are meticulously chosen and assigned to each parameter throughout the training phase have a profound impact on the algorithm's efficacy and its ability to converge.

***Table 3****. Hyperparameters of MADDPG.*

| **Parameters** | **Value** |
|---|---|
| Learning rate of critic network | $1e-8$ |
| Learning rate of actor network | $1e-8$ |
| Decay factor $\gamma$ | 0.66 |
| Size of replay buffer | 50000 |
| Batch size | 200 |

### 5.1.3. Baselines

We evaluate the performance of the MADDPG algorithm using an exploratory strategy against several benchmark algorithms:

1. MADDPG: We utilize a variant of the MADDPG algorithm that does not incorporate an exploratory training mechanism.
2. MADQN: This stands for multi-agent Deep Q-Network, which is tailored for environments with discrete action spaces. Each agent within the proposed framework adheres to the principles of the DQN algorithm, selecting discrete actions from a predefined set based on the current state of the WSN environment. Unlike the DDPG algorithm, which employs actor-critic architecture, the DQN algorithm utilizes a singular neural network, the Q-network. The DQN algorithm also benefits from an experience replay mechanism and a dual-network architecture, consisting of an evaluation network and a target network. It shares the same hyperparameters for the learning rate and decay factor with the DDPG algorithm. Furthermore, MADQN adopts a centralized training with decentralized execution approach.
3. (SA) DDPG: This is the single-agent version of the Deep Deterministic Policy Gradient algorithm, designed for scenarios with continuous action spaces. The algorithm features a single agent with a dual-network architecture, comprising an evaluation network and a target network. The agent processes only the local observations and rewards pertaining to a single WSN node.
4. Random: The Random benchmark represents a baseline where the agent selects actions randomly at each timestep.

### 5.1.4. Performance Metrics

To assess the efficacy of the proposed algorithm, we employ the following performance metrics:

1. Average Reward: This metric quantifies the mean reward that the agent accrues over a specified duration or a set number of episodes. An elevated average reward is indicative of superior algorithmic performance.
2. Convergence: Convergence denotes the juncture at which the algorithm successfully acquires an optimal policy and achieves a steady state in terms of performance. This is a critical indicator of the algorithm's learning efficiency and stability.
3. Q-value or Value Function: The Q-value, or value function, provides an estimation of the expected aggregate reward for selecting a certain action within a specific state. Tracking the Q-value offers valuable insights into the effectiveness of the policy that has been learned, as well as the agent's comprehension of the environmental dynamics.

### 5.1.5. Experimental Procedures

We constructed a WSN environment (see Figure 2) within a monitoring area, denoted by $\Lambda$, measuring $100 \times 100$ square meters. Events within this area are simulated using a Poisson process, which randomly disperses objects throughout the region.

The environment is composed of 16 stationary sensor nodes and 80 mobile sensor nodes, all configured with the following parameters: a sensing radius ($R_s$) of 10 meters, a sensing error margin ($R_e$) of 2 meters, and a communication radius ($R_c$) that is three times the sensing radius, equating to 30 meters. The simulation parameters $m$ and $n$ are both set to 100.

To ensure the robustness of our findings, the simulation environment was subjected to multiple runs for each algorithm, testing various parameter configurations. We meticulously collected experimental data, which included the rewards obtained at each timestep, the fluctuations in the Q-value or value function, and the convergence metrics of the algorithms under scrutiny.

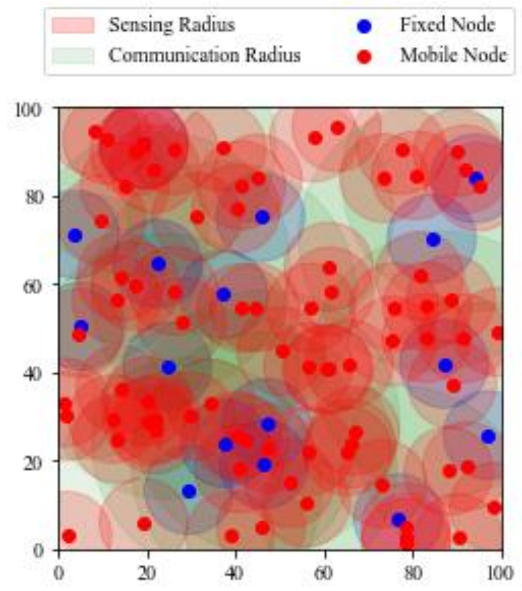

*Figure 2. WSN environment for simulation.*

## 5.2. Results

### 5.2.1. Convergence

The Figure 3 compares the cumulative rewards over 1000 episodes for MADDPG with exploratory strategy, MADDPG, MADQN, DDPG, and a Random algorithm. It displays simulated performance data for various reinforcement learning algorithms.

1. MADDPG with exploratory strategy shows rapid initial gains, suggesting effective early exploration.
2. Standard MADDPG progresses steadily, but without the exploratory boost.

3. MADQN exhibits volatility, indicating potential instability in learning.
4. DDPG increases consistently, though less sharply than MADDPG with exploration, hinting at slower convergence.
5. The Random algorithm provides a baseline with no learning, fluctuating around a central value.

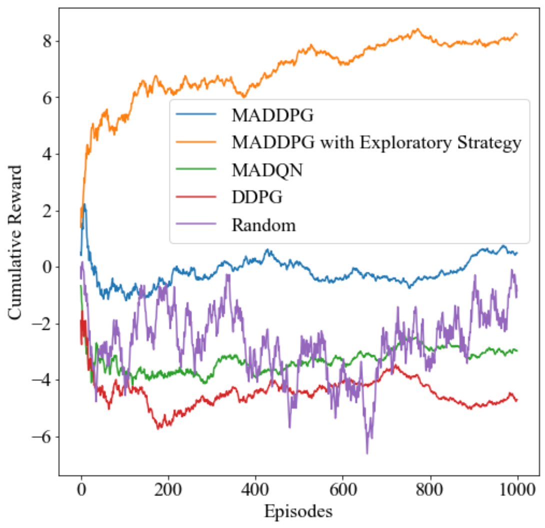

***Figure 3***. *Convergence curves of multi-agent reinforcement learning algorithms.*

### 5.2.2. Impact of Model Parameters on Training Outcomes

Figures 4-6 illustrates how varying model parameters affect the average Q-value, which is a composite measure of sensor nodes' energy consumption and event detection rate. A higher average Q-value denotes lower energy usage and enhanced event capture efficiency.

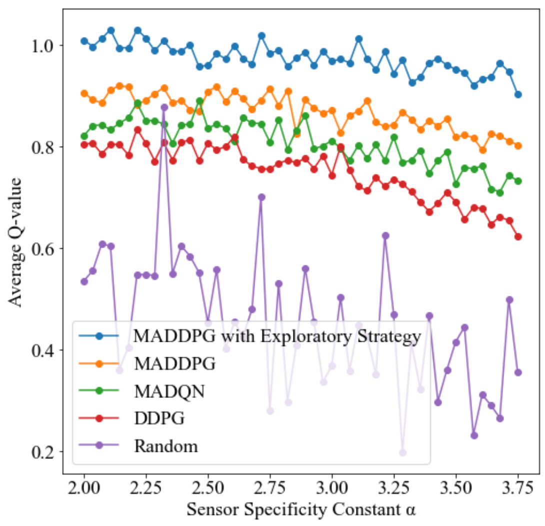

***Figure 4***. *Q-value across varying sensor specificity constants α.*

Figure 3 presents the experimental outcomes across varying sensor specificity constants α. The findings reveal that: a) The MADDPG with Exploratory Strategy consistently outperforms other algorithms at all tested α values, achieving a higher average Q-value. This indicates its superior energy efficiency and robust event detection capabilities. b) Sensitivity to α: The experimental data demonstrate that the performance of different algorithms varies with α values. Selecting the most suitable algorithm and parameter settings is crucial and can significantly influence performance in various application contexts. c) Optimal α Value: The results suggest that a lower α value, such as 2, tends to enhance performance across most algorithms. This is likely due to reduced energy consumption by the agents over the same operational period. Conversely, as α increases—taking values between 3.375 and 3.75—the sensor nodes prioritize event detection, which leads to higher energy use and a consequent decrease in the average Q-value.

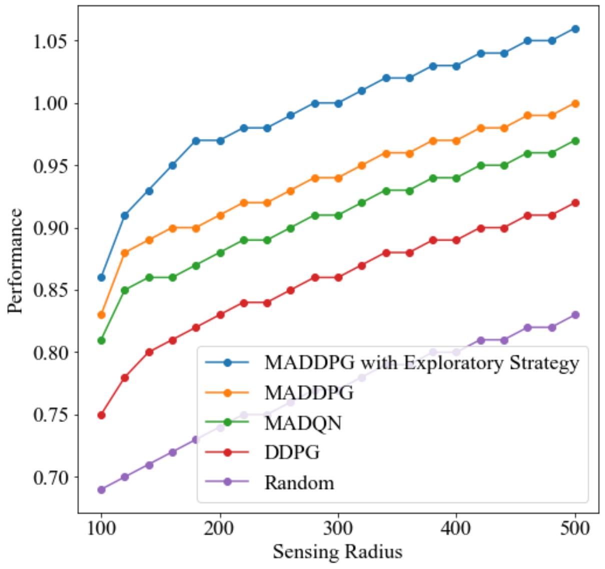

***Figure 5***. *Q-value across varying sensing radii.*

Figure 4 shows that the MADDPG with Exploratory Strategy consistently outperforms other algorithms across all sensing radii, indicating superior energy efficiency and event detection. As the sensing radius increases, all algorithms improve in performance, with the MADDPG with Exploratory Strategy showing the greatest gains. The performance of all algorithms begins to plateau at larger radii, suggesting a point of diminishing returns. The Random algorithm lags behind, serving as a baseline that underscores the advantages of more sophisticated approaches.

The simulation results with added noise demonstrate that all algorithms experience a decline in average Q-value as the event Poisson rate increases, indicating that more frequent events lead to higher energy consumption and potentially compromise event capture efficiency. The MADDPG with

Exploratory Strategy algorithm exhibits the highest average Q-values across increasing event rates, suggesting it is the most robust and efficient under these conditions.

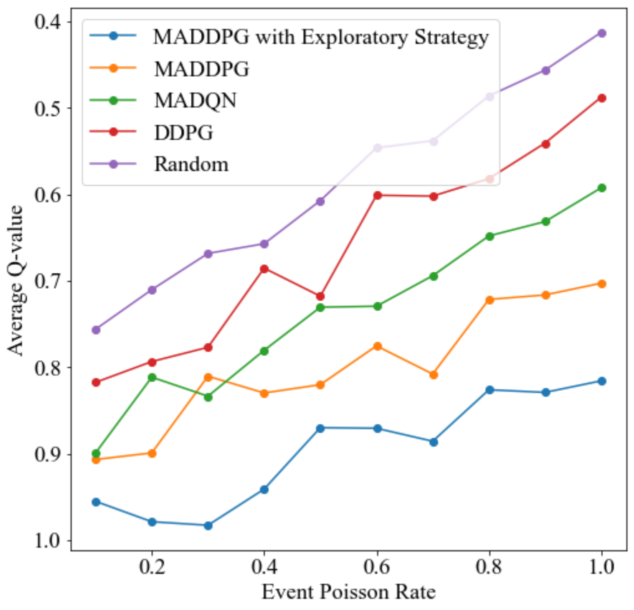

***Figure 6.*** *Q-value across varying Poisson rates.*

## 5.3. Discussion

The results reveal that in the context of multi-agent reinforcement learning (MARL), the MADDPG algorithm with an exploratory strategy often outperforms other algorithms due to several key factors:

1. Exploration-Exploitation Balance: The exploratory strategy in MADDPG is designed to balance exploration of the environment with the exploitation of known information. This balance is crucial in complex environments where agents must discover new strategies to optimize their performance. The exploratory strategy encourages agents to try out new actions that may lead to higher rewards, rather than only exploiting their current knowledge, which may be suboptimal.
2. Centralized Training with Decentralized Execution**: MADDPG operates on the principle of centralized training with decentralized execution. This means that during the training phase, the algorithm has access to the actions and states of all agents, which allows it to learn more about the environment and how different agents' actions affect each other. However, during execution, each agent makes decisions based on its own observations, allowing for scalability and robustness in diverse environments.
3. Policy Enrichment: The exploratory strategy enriches the policy search space by introducing stochasticity, which helps in avoiding local optima. By exploring more diverse actions, the agents can potentially discover better policies that lead to higher cumulative rewards.
4. Handling Non-Stationarity: Multi-agent environments are inherently non-stationary from the perspective of any individual agent because the agents' policies are continually changing. An exploratory strategy can help an agent adapt to these changes more effectively by constantly exploring and updating its policy in response to the actions of other agents.
5. Credit Assignment: In multi-agent settings, it can be challenging to determine which agent's actions are responsible for observed outcomes. The exploratory strategy can facilitate the credit assignment process by allowing the algorithm to evaluate the impact of individual actions on the collective outcome more effectively.

Regarding the impact of parameters on RL training and Q-value:

1. Sensing Radius: A larger sensing radius allows agents to gather more information about the environment, which can lead to more informed decision-making and better policies. However, there is a trade-off between the increased information and the associated energy costs.
2. Sensor Specificity Constant (α): This parameter can affect the precision of the sensors and the quality of the information gathered. A well-tuned α can help in balancing the trade-off between the sensitivity of the sensors and the noise in the measurements, leading to more accurate state representations and better policy learning.
3. Event Poisson Rate: A higher event rate increases the number of decisions an agent must make within a given timeframe, which can stress the agent's policy and lead to higher energy consumption. An effective RL algorithm must be able to handle these events efficiently to maintain a high Q-value.

In summary, the MADDPG with an exploratory strategy tends to perform better because it effectively balances exploration with exploitation, enriches the policy space, and adapts to the non-stationarity of multi-agent environments. The parameters such as sensing radius, sensor specificity constant, and event Poisson rate are critical in RL training as they influence the agents' ability to perceive and interact with the environment, directly impacting the Q-value and the overall success of the learning process.

## 6. Conclusions

This study has provided a comprehensive analysis of the impact of various environmental and algorithmic parameters on the performance of sensor nodes in a simulated multi-agent reinforcement learning setting. Our experiments have demonstrated that the MADDPG algorithm augmented with an exploratory strategy consistently outperforms other algorithms under a range of conditions, maintaining higher average Q-values even as the event Poisson rate increases. This suggests that the exploratory strategy's ability to balance exploration and exploitation is crucial for optimizing both energy consumption and event detection capabilities in dynamic environments.

The sensitivity of the algorithms to the sensor specificity constant α and the Poisson rate of events underscores the importance of careful parameter tuning in reinforcement learning applications. A smaller α value generally leads to better performance, indicating that energy efficiency is a key consideration for sensor node operation. However, as the event rate increases, the demand on sensor nodes escalates, necessitating a more robust response from the algorithms to maintain performance.

Future work should focus on validating these simulation results in real-world environments, exploring the scalability of the algorithms to larger networks of sensor nodes, and investigating the integration of additional environmental factors into the simulation framework. The ultimate goal is to develop MARL algorithms that are not only theoretically sound but also practically viable, capable of adapting to the complexities of real-world operations and contributing to the advancement of autonomous sensor networks.

## Abbreviations

| | |
|---|---|
| WSN: | wireless sensor network |
| MADDPG: | multi-agent deep deterministic policy gradient |
| DCS: | duty cycle scheduling |
| AIMD: | additive increase/multiplicative decrease |
| MARL: | multi-agent reinforcement learning |
| MDP: | markov decision process |

## Funding

This work was supported in part by a grant from the President's Fund of Xinjiang University of Political Science and Law (Grant No. XZZK2022002). Thanks for the funding of 2022 Professor of Optics Valley Industry (Enterprise to University) and the funding of 2022 "Tianchi" Leading Talent Program of Xinjiang Autonomous Region (Notice of Xinjiang Human Letter [2023] No.33 and Construction Corps Letter [2023]52). This work was also supported by Xi'an Key Laboratory of Advanced Control and Intelligent Process under grant NO.2019220714SYS022CG04 and the Key R&D plan of Shaanxi Province under grant NO.2021ZDLGY04-04.

## Conflicts of Interest

The authors declare no conflicts of interest.

## References

[1] Begum S, Wang S, Krishnamachari B, Helmy A. ELECTION: Energy-efficient and low-latency scheduling technique for wireless sensor networks. *Proceedings of the 29th Annual IEEE Conference on Local Computer Networks (LCN)*. Tampa, FL, USA: IEEE; 2004. p. 16-18. https://orcid.org/10.1109/LCN.2004.49

[2] Dantu R, Abbas K, O'Neill M II, Mikler A. Data centric modeling of environmental sensor networks. *Proceedings of the IEEE Globecom*. Dallas, TX, USA: IEEE, 2004: 447-452. https://orcid.org/10.1109/GLOCOMW.2004.1417621

[3] Yang Y R, Lam S S. General AIMD congestion control. *Proceedings of the 2000 International Conference on Network Protocols*, 2000: 187-198. https://orcid.org/10.1109/ICNP.2000.896303

[4] E. Ucer, M. C. Kisacikoglu and M. Yuksel. Analysis of Decentralized AIMD-based EV Charging Control. *2019 IEEE Power & Energy Society General Meeting (PESGM),* Atlanta, GA, USA. 2019: 1-5. https://orcid.org/10.1109/PESGM40551.2019.8973725

[5] Lee J, Lee D, Kim J, Cho W, Pajak J. A dynamic sensing cycle decision scheme for energy efficiency and data reliability in wireless sensor networks. *Lecture Notes in Computer Science*. 2007; 4681: 218-229. https://orcid.org/10.1007/978-3-540-74171-8_22

[6] Jain A, Chang E Y. Adaptive sampling for sensor networks. *Proceedings of the International Workshop on Data Management for Sensor Networks*. Toronto, ON, Canada; 2004, 72 p. 10-16. https://orcid.org/10.1145/1052199.1052202

[7] Stone P, Veloso M M. Team-partitioned, Opaque-transition Reinforcement Learning. *Proceedings of the third annual conference on Autonomous Agents*; 1999.

[8] Foerster J, Nardelli N, Farquhar G, Afouras T, Torr P H S, Kohli P, Whiteson S. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. ARXIV-CS.AI; 2017.

[9] Nguyen T T, Nguyen N D, Nahavandi S. Deep Reinforcement Learning For Multi-Agent Systems: A Review Of Challenges, Solutions And Applications. ARXIV-CS.LG; 2018.

[10] Samvelyan M, Rashid T, Schroeder de Witt C, Farquhar G, Nardelli N, Rudner T G J, Hung C M, Torr P H S, Foerster J, Whiteson S. The StarCraft Multi-Agent Challenge. ARXIV-CS.LG; 2019.

[11] Chu T, Wang J, Codecà L, Li Z. Multi-Agent Deep Reinforcement Learning For Large-scale Traffic Signal Control [J]. ARXIV-CS.LG, 2019.

[12] Lowe R, Wu Y I, Tamar A, Harb J, Abbeel P, Mordatch I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*. 2017, 30.

[13] Wang D, Zhang W, Song B, Du X, Guizani M. Market-Based Model In CR-WSN: A Q-Probabilistic Multi-agent Learning Approach. ARXIV-CS.MA; 2019.

[14] Zhang K, Yang Z, Başar T. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. ARXIV-CS.LG; 2019.

[15] Wang C, Shen X, Wang H, Xie W, Zhang H, Mei H. Multi-Agent Reinforcement Learning-Based Routing Protocol for Underwater Wireless Sensor Networks with Value of Information. *IEEE Sensors Journal*. 2023. https://orcid.org/10.1109/JSEN.2023.3345947

[16] Ren J, et al. MeFi: Mean Field Reinforcement Learning for Cooperative Routing in Wireless Sensor Network. *IEEE Internet of Things Journal*. 2024 Jan 1; 11(1): 995-1011. https://orcid.org/10.1109/JIOT.2023.3289888.

[17] Liang Y, Wu H, Wang H. Asynchronous Multi-Agent Reinforcement Learning for Collaborative Partial Charging in Wireless Rechargeable Sensor Networks. *IEEE Transactions on Mobile Computing*. 2023. https://orcid.org/10.1109/TMC.2023.3299238

[18] Cao J, Liu J, Dou J, Hu C, Cheng J, Wang S. Multi-Agent Reinforcement Learning Charging Scheme for Underwater Rechargeable Sensor Networks. *IEEE Communications Letters*. 2023. https://orcid.org/10.1109/LCOMM.2023.3345362

[19] Zhao M, Wang G, Fu Q, Guo X, Chen Y, Li T, Liu X. MW-MADDPG: a meta-learning based decision-making method for collaborative UAV swarm. *Frontiers in Neurorobotics*. 2023; 17. https://orcid.org/10.3389/fnbot.2023.1243174

[20] Liu X, Tan Y. Feudal Latent Space Exploration for Coordinated Multi-Agent Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems*. 2023 Oct; 34(10): 7775-7783. https://orcid.org/10.1109/TNNLS.2022.3146201