

Article

Optimizing Task Offloading for Power Line Inspection in Smart Grid Networks with Edge Computing: A Game Theory Approach

Xu Lu ¹, Sihan Yuan ^{2,*}, Zhongyuan Nian ³, Chunfang Mu ³ and Xi Li ²

¹ Power Dispatching Control Center of State Grid Inner Mongolia Eastern Power Co., Ltd., Hohhot 010020, China; luxu@md.sgcc.com.cn

² Pan Network Wireless Communication Laboratory, Beijing University of Posts and Telecommunications, Haidian District, Beijing 100876, China; lixi@bupt.edu.cn

³ Information and Communication Branch of State Grid Inner Mongolia Eastern Power Co., Ltd., Hohhot 010020, China; nianzhongyuan@md.sgcc.com.cn (Z.N.); muchunfang@md.sgcc.com.cn (C.M.)

* Correspondence: ysh01@bupt.edu.cn

Abstract: In the power grid, inspection robots enhance operational efficiency and safety by inspecting power lines for information sharing and interaction. Edge computing improves computational efficiency by positioning resources close to the data source, supporting real-time fault detection and line monitoring. However, large data volumes and high latency pose challenges. Existing offloading strategies often neglect task divisibility and priority, resulting in low efficiency and poor system performance. This paper constructs a power grid inspection offloading scenario using Python 3.11.2 to study and improve various offloading strategies. Implementing a game-theory-based distributed computation offloading strategy, simulation analysis reveals issues with high latency and low resource utilization. To address these, an improved game-theory-based strategy is proposed, optimizing task allocation and priority settings. By integrating local and edge computing resources, resource utilization is enhanced, and latency is significantly reduced. Simulations show that the improved strategy lowers communication latency, enhances system performance, and increases resource utilization in the power grid inspection context, offering valuable insights for related research.



Citation: Lu, X.; Yuan, S.; Nian, Z.; Mu, C.; Li, X. Optimizing Task Offloading for Power Line Inspection in Smart Grid Networks with Edge Computing: A Game Theory Approach. *Information* **2024**, *15*, 441. <https://doi.org/10.3390/info15080441>

Academic Editor: Lorenzo Mucchi

Received: 12 June 2024

Revised: 22 July 2024

Accepted: 23 July 2024

Published: 29 July 2024



Copyright: © 2024 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/>).

Keywords: power line inspection; inspection robots; edge computing; distributed offloading strategy; game theory

1. Introduction

In recent years, the convergence of Artificial Intelligence (AI) and Mobile Edge Computing (MEC) has catalyzed transformative advancements across various domains, including power grid management [1,2]. A particularly promising application of this convergence lies in the optimization of power grid inspection processes through the deployment of power line inspection robots. These robots, endowed with sophisticated sensors and AI algorithms, possess the capacity to autonomously traverse power grid infrastructures, identifying anomalies and executing routine maintenance tasks [3].

Conventional approaches to the deployment of power line inspection robots typically entail the utilization of on-board processing units responsible for executing computationally intensive tasks such as image processing, data analysis, and decision-making. However, this paradigm confronts inherent limitations pertaining to processing capacity, energy consumption, and real-time responsiveness [4]. To surmount these challenges, the concept of MEC emerges as a pivotal paradigm [5].

MEC represents a distributed computing architecture that extends cloud computing capabilities closer to the network edge, thereby enabling computation to be performed closer to the data source. In the context of power grids, MEC servers are strategically deployed at base stations, forming an edge computing infrastructure within the grid. This

infrastructure offers proximity to the physical assets and operational data of the power grid, facilitating low-latency data processing, real-time analytics, and rapid decision-making [6].

By leveraging the computational prowess and proximity of MEC servers, power line inspection robots can effectively offload demanding computational tasks, thereby augmenting their performance, mitigating latency, and conserving energy [7]. Furthermore, MEC facilitates seamless integration with existing communication networks, enabling efficient data exchange and coordination between inspection robots, base stations, and central control centers [8].

The primary contributions of this paper are as follows:

- **Proposing an Improved Game-Theory-Based Distributed Computation Offloading Strategy:** This paper improves upon existing game-theory-based distributed computation offloading strategies by optimizing task allocation and prioritization, enhancing resource utilization, and significantly reducing latency.
- **Constructing a Power Grid Inspection Offloading Scenario:** Utilizing Python, this paper constructs a power grid inspection offloading scenario to study and enhance various offloading strategies, revealing issues with high latency and low resource utilization in existing strategies through simulation analysis.
- **Integrating Local and Edge Computing Resources:** In this paper, we introduce a method for integrating local and edge computing resources, proposing an optimized task allocation strategy that enhances resource utilization and significantly reduces communication latency.
- **Validating Through Simulation Results:** Simulation results demonstrate that the proposed improved strategy effectively lowers communication latency, enhances system performance, and increases resource utilization in the power grid inspection context, providing valuable insights for related research.

The subsequent sections of this paper are structured as follows: Section 2 provides a brief overview of related studies. Section 3 delineates the proposed system Model for integrating power line inspection robots with MEC. Subsequently, Section 4 presents the statement of the problem, while Section 5 demonstrates the solution to this problem. Subsequently, Section 6 presents the simulation results and discussion. Finally, a summary of the paper is provided in Section 7.

2. Related Works

Inspection robots for power transmission lines have been the focus of extensive research due to their potential to improve the efficiency and safety of power grid maintenance. LinBot, detailed in [9], is an innovative robot designed for high-voltage transmission lines, capable of overcoming various ground wire obstacles such as warning balls and tower tips, thanks to its active and passive mechanisms. Its stability and feasibility were validated through simulations and field experiments. Similarly, the single-arm inspection robot system discussed in [10] addresses conventional robots' limitations by employing a large-load UAV for cooperative operations, enhancing live line inspections' precision and control. The dynamics of this robot were analyzed using the Lagrangian method and simulated in ADAMS software, confirming its practicality. In the context of smart grids, an autonomous high-voltage transmission inspection robot was introduced in [11], featuring a hierarchical control structure for remote management and autonomous decision-making using image recognition and laser sensors. The application of 5G mobile edge computing (MEC) in power robotic inspections, as proposed in [3], enhances efficiency by addressing multi-dimensional entity heterogeneity and environment dynamics. This framework uses an AI-enabled optimization algorithm for route planning and task offloading, significantly reducing latency. Another study [12] focuses on the mechanism design and kinematics of a robot equipped with sensors and cameras, controlled by a Raspberry Pi for real-time visualization and data transmission, ensuring continuous inspection and maintenance. Inspired by gibbons, a two-arm swinging inspection robot described in [13] uses a bionic design to swing and cross obstacles, with kinematic analysis and simulations verifying its

stability. Addressing slow obstacle crossing, a robot using asymmetrical driving wheels was developed in [14], enabling quick and efficient movement over spacers and counterweights, verified by simulations and prototype experiments. These advancements in robotics, edge computing, and innovative design principles significantly enhance inspection robots' capabilities for power transmission lines, providing a foundation for more efficient, reliable, and autonomous power grid maintenance solutions.

According to an overview of relevant research, it can be observed that previous studies on power line inspection robots have not yet considered the computation offloading scheme in practical smart grid scenarios. Furthermore, the problem of efficiently offloading computations through the comprehensive distributed utilization of computational resources remains unresolved for inspection robots.

Our work "Optimizing Task Offloading for Power Line Inspection in Smart Grid Networks with Edge Computing: A Game Theory Approach" introduces a game-theory-based dynamic optimization approach that ensures fair resource allocation among multiple agents while minimizing task offloading costs. Table 1 provides a comparative overview of key indicators and methodologies of our work against the aforementioned studies.

Table 1. Comparison of Key Indicators and Methodologies.

Indicator/Method	Our Paper: Optimizing Task Offloading for Power Line Inspection	PPO-Based Computation Offloading and Resource Allocation [3]	5G MEC-Based Intelligent Computation Offloading in Power Robotic Inspection [6]
Objective	Minimize task offloading cost and improve system utility	Minimize energy consumption and delay	Optimize task offloading and route planning to reduce delay
Approach	Multi-agent dynamic optimization based on game theory	Proximal Policy Optimization (PPO) algorithm	AI-enabled multi-dimensional collaborative optimization algorithm
Methodological Novelty	<ul style="list-style-type: none"> Combines game theory and edge computing for dynamic task offloading Considers multi-agent competition to ensure fair resource allocation Optimizes system utility through intelligent game models 	<ul style="list-style-type: none"> Uses PPO to avoid dimensionality curse Provides long-term optimization strategies 	<ul style="list-style-type: none"> Optimizes task offloading and route planning using AI and MEC Enhances low-latency task processing performance

This comparative analysis highlights the novel contributions of our work in ensuring fair resource allocation among multiple agents and dynamically optimizing task offloading strategies to enhance overall system utility.

3. System Model

This paper presents a scenario of distributed computation offloading involving multiple power line inspection robots and MEC servers within power grid infrastructure. In this scenario, the power line inspection robots are tasked with offloading computational tasks to servers stationed at base stations, a process known as computation offloading. Each base station is equipped with a dedicated MEC server. The scenario of multiple power line inspection robots and multiple MEC servers for computation offloading is depicted in Figure 1.

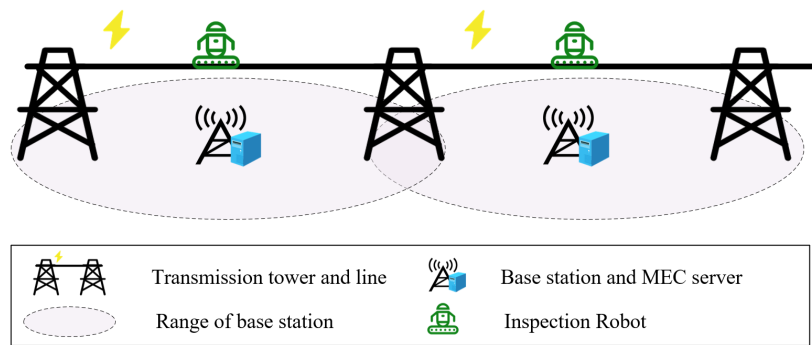


Figure 1. Power grid power line inspection scenario.

Within this context, computation offloading entails addressing the following issues:

- **Computation resource allocation:** When multiple power line inspection robots concurrently submit computation tasks to MEC servers, the allocation of resources becomes crucial. Inspection robots must consider factors such as latency, energy consumption, as well as the load and availability of each MEC server to determine the most suitable server for task offloading.
- **Task partitioning:** Each power line inspection robot's computational task can be partitioned into distinct segments, which are then assigned for processing either locally or offloaded to MEC servers. In this study, the tasks of each power line inspection robot are divided into 100 sub-tasks, and decisions are made regarding the allocation of these sub-tasks to optimize processing efficiency. This approach aims to maximize the utilization of local computational resources while alleviating the computational burden on power line inspection robots, thereby enhancing offloading efficiency.
- **Task prioritization:** In the power grid context, tasks exhibit varying degrees of urgency and priority. For example, tasks related to emergency fault detection take precedence over routine inspection tasks. Therefore, assigning priorities to tasks and considering them during task allocation are essential in determining the optimal task scheduling strategy within the power line inspection robots and MEC integration scenario.

In the scenario of multiple power line inspection robots and multiple MEC servers, the primary challenge in computation offloading lies in selecting the most appropriate MEC nodes to achieve optimal coordination between power line inspection robots and MEC servers. Addressing this challenge involves identifying an optimal solution that balances system resource allocation and task computation overhead, constituting a multi-player dynamic game. Subsequent sections will delve into modeling this problem.

This scenario encompasses two primary components: power line inspection robots and base stations. The set of power line inspection robots is denoted as $\{Robot_1, Robot_2, \dots, Robot_N\}$, while the set of MEC servers is denoted as $\{MEC_1, MEC_2, \dots, MEC_M\}$. At an initial time point, power line inspection robots are assumed to be distributed uniformly and randomly within the power grid area, while MEC servers are strategically placed at base stations. For simplification purposes, it is assumed that power line inspection robots move at a constant speed within the power grid. Based on the coordinates of power line inspection robots and MEC servers, the distance between them can be calculated. At any given moment, each power line inspection robot selects the optimal MEC server for task offloading and computation processing, necessitating distributed decision-making.

Each power line inspection robot has a unique task represented by a triplet $Task_n = (\alpha_n, \beta_n, P_n)$, where α_n denotes the data volume of the task, β_n represents the required CPU cycle count, and P_n signifies the task's priority. The task offloading decision for each power line inspection robot is represented by the vector $d_n = (n, d_n^M)$, where $d_n^M \in M = \{1, 2, \dots, M\}$ indicates the base station to which the task is offloaded. Additionally, the task allocation decision for each power line inspection robot is represented by the

vector $f_n \in \{1, 2, \dots, 100\}$, indicating the number of sub-tasks offloaded to the MEC server for processing.

3.1. Communication Model

Each power line inspection robot needs to make a distributed decision on whether to offload its computation tasks to the corresponding MEC server and how many tasks to keep for local processing. For robot n , it needs to make an offloading decision d_n , i.e., choosing which MEC server to offload tasks to. When all robots complete their distributed decisions, the overall offloading vector d can be determined, and the data transmission rate $R_n^m(d)$ between robot n and the m th base station can be computed. This can be calculated using the Shannon formula, which demonstrates that the maximum information transmission rate C in the channel, subject to Gaussian white noise, is given by:

$$C = W \log_2(1 + \text{SINR}) \tag{1}$$

Here, W represents the channel bandwidth (in Hz), and SINR stands for Signal-to-Interference-plus-Noise Ratio, which is the ratio of the average signal power to the noise power in the channel. In the context of power line inspection in power grids, the SINR for robot n is the ratio of its data transmission power to the Gaussian white noise power and can be expressed as:

$$\text{SINR} = \frac{g_n p_n}{\sigma^2} \tag{2}$$

Here, g_n denotes the channel gain for robot n , σ^2 is the power of Gaussian white noise, and p_n is the transmission power of robot n , i.e., the transmit power of the robot device. The channel gain g_n is given by $g_n = D^{-r} \zeta_n$, where ζ_n represents the small-scale fading of the channel, and D^{-r} accounts for the path loss between the robot and the base station, with r being the path loss exponent.

Using Equations (1) and (2), we can compute the data transmission rate as:

$$R_n^m(d) = B \log_2\left(1 + \frac{g_n p_n}{\sigma^2}\right) \tag{3}$$

Here, B represents the bandwidth of the uplink channel between the robot and the MEC server.

The size of the input data for tasks and the data transmission rate determine the communication time. The uplink transmission rate $R_n^m(d)$ of the robot can be used to calculate the task transmission delay for the robot, which is expressed as:

$$T_n^{\text{tran}}(d) = \frac{a_n^{\text{off}}}{R_n^m(d)} \tag{4}$$

Here, a_n^{off} denotes the amount of data offloaded to the MEC server for computation, satisfying the following relation:

$$a_n^{\text{off}} + a_n^{\text{loc}} = \alpha_n \tag{5}$$

Based on the robot's upload power p_n , we can obtain the task transmission energy consumption $E_n^{\text{tran}}(d)$ of the robot device:

$$E_n^{\text{tran}}(d) = \frac{g_n p_n}{R_n^m(d)} \tag{6}$$

3.2. Computing Model

The utility function represents the quantitative relationship between the utility obtained and the combined resources consumed in the offloading system, reflecting robot satisfaction in the offloading system. Currently, offloading decision schemes often use delay cost and system utility as offloading indicators. Offloading decisions need to consider task processing delays because offloading decisions must ensure execution when grid line

inspection robots are within the communication range, ensuring that robots remain within the specified coverage area throughout the offloading process. Additionally, reducing device energy consumption is also an important issue, as excessive energy consumption can lead to rapid depletion of robot battery power and reduced operational capabilities. Therefore, formulating a reasonable offloading strategy requires a comprehensive consideration of both delay and energy consumption factors. This paper will weight and sum the delay and energy consumption factors to obtain the utility function for each smart grid robot, and decision-making aims to minimize the value of the weighted sum to achieve a balance between delay and energy consumption.

Tasks can be partially or fully computed locally or offloaded for computation, and the costs of delay and energy consumption will differ depending on the computation mode. The following analysis discusses these two computation modes.

(1) Local Computation

When power line inspection robots perform local computation, they need to bear the delay of their own tasks and the energy consumption of robot equipment, which are task processing costs, denoted by $T_n^{\text{loc}}(d)$ and $E_n^{\text{loc}}(d)$, respectively. Assuming that in this communication scenario, the computational capability of each robot device is the same, denoted by c_{loc} , and the number of CPU cycles required for local task computation is β_n^{loc} , then the delay and energy consumption incurred during local computation are given by:

$$T_n^{\text{loc}}(d) = \frac{\beta_n^{\text{loc}}}{c_{\text{loc}}} \quad (7)$$

$$E_n^{\text{loc}}(d) = \epsilon_n \beta_n^{\text{loc}} \quad (8)$$

Here, ϵ_n represents the energy consumption coefficient of the robot equipment during local computation. Let β_n^{off} represent the number of CPU cycles required for data offloaded to the central server for computation, satisfying:

$$\beta_n^{\text{off}} + \beta_n^{\text{loc}} = \beta_n \quad (9)$$

Then, the total cost for power line inspection robot n to perform local computation is:

$$C_n^{\text{loc}}(d) = w_n^t T_n^{\text{loc}}(d) + w_n^e E_n^{\text{loc}}(d) \quad (10)$$

subject to:

$$w_n^t, w_n^e \in [0, 1] \quad (11)$$

$$w_n^t + w_n^e = 1 \quad (12)$$

Here, w_n^t and w_n^e are weighting coefficients, corresponding to the weights of delay and energy consumption in the total cost of task execution. Robots have different levels of concern for delay and energy consumption, which can be dynamically adjusted by changing the values of these two parameters. Additionally, it is necessary to satisfy the constraints in Equations (11) and (12). In the scenario set in this paper, there are three types of power line inspection robots, categorized based on their task priorities: high delay-sensitive, medium delay-sensitive, and low delay-sensitive robots. The weighting coefficients for delay and energy consumption differ for these three types of robots.

(2) Central Server Computation Offloading

When robots offload computation to the central server, they need to transfer task data to the server, incurring transmission delay and transmission energy consumption, and the computational delay of tasks is determined by the computational capability of the central server. Assuming that each central server has the same computational capability, denoted by c_{off} , and central servers can simultaneously provide computing and storage services to multiple robots, but the computational resources allocated to each robot offloading tasks

are limited. All power line inspection robots compete fairly for server computing resources, i.e., server computing resources are evenly distributed among each task offloaded to this node. Then, for task n , the computing capability obtained from the central server is:

$$c_n^{\text{off}}(d) = \frac{c_{\text{off}}}{\sum_{i \in N, d_i^M = d_n^M} 1} \quad (13)$$

Here, $\sum_{i \in N, d_i^M = d_n^M} 1$ represents the total number of robots offloading to the same central server. The CPU cycle required for task n is β_n , and the delay incurred during task offloading consists of two parts: transmission delay and processing delay. Task transmission energy consumption is calculated using Equation (6). Thus, the delay and device energy consumption incurred during task n offloading are:

$$T_n^{\text{off}}(d) = T_n^{\text{tran}}(d) + T_n^{\text{proc}}(d) = \frac{a_n^{\text{off}}}{R_n^m(d)} + \frac{\beta_n^{\text{off}}}{c_n^m(d)} \quad (14)$$

$$E_n^{\text{off}}(d) = E_n^{\text{tran}}(d) = \frac{a_n^{\text{off}} p_n}{R_n^m(d)} \quad (15)$$

Here, $T_n^{\text{tran}}(d)$ and $T_n^{\text{proc}}(d)$ represent the transmission delay and processing delay incurred during task n offloading, respectively. In this offloading scenario, the total energy consumption caused by power line inspection robot computation offloading includes both transmission energy consumption and computation energy consumption, as distributed offloading only considers the transmission energy consumption of smart grid robots. Additionally, because the amount of data for calculation results is much smaller than the input data for tasks, the energy required for robot equipment to receive task computation results can be neglected. Given the offloading decision vector d for a certain moment, the total cost, i.e., utility function, for task n , for MEC computation offloading, is expressed as:

$$C_n^{\text{off}}(d) = w_n^t T_n^{\text{off}}(d) + w_n^e E_n^{\text{off}}(d) \quad (16)$$

In summary, the total cost of task processing for smart grid robot n is the sum of the local computation cost and the MEC computation offloading cost, i.e.:

$$C_n^{\text{all}}(d) = C_n^{\text{loc}}(d) + C_n^{\text{off}}(d) \quad (17)$$

This concludes the computational model section, detailing the local computation and MEC computation offloading strategies for smart grid robots in the context of utility function optimization considering delay and energy consumption factors.

4. Problem Formulation

Using game-theory-related theories and techniques, we can construct a complete information, dynamic, non-cooperative game model to optimize the computational offloading process in the smart grid multi-access edge computing (MEC) network architecture. This game model is represented as $G = \{V, D, F, C(d)\}$, where players sequentially make computational offloading decisions. Before making decisions, players are aware of other players' actions and can make optimal offloading decisions based on this information to maximize their own benefits. Solving the Nash equilibrium of this game model optimizes the computational offloading decisions in the smart grid MEC network architecture. In G , V represents the finite set of players in the game, D represents the set of robot offloading decisions, F represents the set of robot task allocation decisions, and $C(d)$ represents each player's cost function, which in this game model denotes the computational offloading cost. Each player will choose an optimal computational offloading decision $d_n \in D$ to minimize their own cost function $C(d_n, f_n, d_{-n}, f_{-n})$, where $d_{-n} = (d_1, d_2, \dots, d_{n-1}, d_{n+1}, \dots, d_N)$ denotes the offloading decisions of other robots, and $f_{-n} = (f_1, f_2, \dots, f_{n-1}, f_{n+1}, \dots, f_N)$ denotes the task allocation decisions of other robots. In this scenario, the cost function

$C(d_n, f_n, d_{-n}, f_{-n})$ represents each player's task processing delay and energy consumption cost, mathematically expressed as:

$$\min_{d_n \in D} C_n^{\text{all}}(d_n, f_n, d_{-n}, f_{-n}), \quad \forall n \in N \quad (18)$$

where $C_n(d_n, f_n, d_{-n}, f_{-n})$ is the utility function of the inspection robot, representing the total task processing delay and energy consumption cost for local computation and MEC computational offloading in this scenario.

According to the Nash equilibrium theorem, if the strategy set of players in a dynamic game is finite, then the number of action steps in this game is also finite. Additionally, if this dynamic game is a complete information game, there must be a pure strategy Nash equilibrium. Therefore, the established computational offloading game model has a Nash equilibrium solution and can be obtained within a finite number of steps. Thus, an appropriate algorithm can be designed to obtain the optimal computational offloading strategy combination.

5. Proposed Algorithm

This section implements a game-theoretic distributed computational offloading strategy improvement algorithm, which allows each participant to select the optimal computational offloading strategy based on the current environment and reach the Nash equilibrium after a finite number of iterations. The algorithm consists of two phases: offloading environment perception and offloading decision and task allocation decision update phases.

5.1. Offloading Environment Perception Phase

In this phase, power line inspection robots first calculate the distance to each MEC node and select servers within the MEC coverage area for offloading. Then, the task processing delay $T_n(d)$ is calculated. Since local computation and computational offloading occur simultaneously, the task processing delay is the minimum of the two, i.e.,

$$T_n(d) = \min\{T_n^{\text{loc}}(d), T_n^{\text{off}}(d)\} \quad (19)$$

Next, the data transmission rate for the current task offloading is calculated. The data transmission rate for offloading to each MEC node within range is computed according to Equation (3). Based on this, the delay and energy consumption for offloading tasks to different MEC nodes are determined, which serves as the basis for robots to choose offloading strategies and task allocation strategies.

5.2. Offloading Decision and Task Allocation Decision Update Phase

In this phase, as the game is sequential, all robots update their strategies in order, with only one power line inspection robot able to update its offloading decision and task allocation decision in each time slot. Based on the computed computational offloading costs for all offloading strategies from the previous phase, robot n comprehensively considers offloading and task allocation decisions, choosing the optimal combination of offloading strategy and task allocation decision:

$$d_n^*(t) = \arg \min_{d_n(t) \in D} C_n(d_n(t), f_n(t), d_{-n}(t), f_{-n}(t)) \quad (20)$$

$$f_n^*(t) = \arg \min_{f_n(t) \in F} C_n(d_n(t), f_n(t), d_{-n}(t), f_{-n}(t)) \quad (21)$$

$d_n^*(t)$ and $f_n^*(t)$ are the optimal offloading strategy and task allocation strategy for inspection robot n at time slot t . If $d_n^*(t) \neq d_n^*(t-1)$ and $f_n^*(t) \neq f_n^*(t-1)$, meaning the obtained optimal strategy combination differs from the current offloading strategy combination, robot n will update their strategy combination in the next time slot, while other robots' strategy combinations remain unchanged. Upon reaching Nash equilibrium,

all robots' utility functions are minimized, and the weighted sum of delay and energy consumption is minimized. The computational offloading decision vector d and task allocation decision vector f remain unchanged.

The pseudocode of this improved algorithm is shown in Algorithm 1.

Algorithm 1: Game-Theoretic Distributed Computational Offloading Strategy Improvement Algorithm

Initialization: Game model $G = \{V, D, C(d)\}$, with all robots' initial offloading strategies $d_n = (n, 0)$ and task allocation strategies $f_n = (n, 0, 0)$, and the priority of each task P_j .

Output: Optimal offloading strategy combination and optimal task allocation.

```

1: while  $d^*(t) \neq d^*(t-1)$  and  $f^*(t) \neq f^*(t-1)$  do
2:   for each decision time slot  $t$  and each robot  $n$  do
3:     Calculate the distance to each MEC node within range.
4:     Compute the data upload rate  $R_n^m(d)$ .
5:     Calculate the transmission delay  $T_n^m(d)$  and exclude MEC nodes that are
      out of range.
6:     Compute the optimal computational offloading strategy  $d_n^*(t)$  and the
      optimal task allocation strategy  $f_n^*(t)$ .
7:     if  $d_n^*(t) \neq d_n^*(t-1)$  then
8:       Update the robot's offloading strategy  $d_n^*(t+1) = d_n^*(t)$ .
9:     end if
10:    if  $f_n^*(t) \neq f_n^*(t-1)$  then
11:      Update the task allocation strategy  $f_n^*(t+1) = f_n^*(t)$ .
12:    end if
13:  end for
14: end while

```

6. Simulation Results and Discussions

6.1. Simulation Experiment Settings

This experiment is conducted in a simulation environment built with Python 3.11.2. The inspection area is a 5 km segment of power lines. Power line inspection robots are initially distributed randomly along the segment. The robots move at a constant speed along the power lines. Base stations are deployed at regular intervals along the power lines. Table 2 shows the specific parameter settings.

Table 2. System parameters.

Parameter	Value	Description
M	5, 6, 7, ..., 10	Number of base stations
N	10, 15, 20, ..., 50	Number of inspection robots
V	1 or 2 m/s	Robot speed
R	3000 m	Base station coverage radius
w_h^t	0.7	High-latency-sensitive robot weight
w_m^t	0.5	Medium-latency-sensitive robot weight
w_l^t	0.3	Low-latency-sensitive robot weight
α_n	[5, 30] MB	Task data size
β_n	[500, 3000] Megacycles	CPU cycles required by task
B	5 MHz	Channel bandwidth
σ^2	-125 dBm	Gaussian white noise power
p_n	100 mW	robot device transmission power
c_{off}	10 GHz	MEC server computational capacity
c_{loc}	10 GHz	Local computational capacity

To compare the effectiveness of the game-theoretic distributed computational offloading strategy improvement algorithm with other schemes, this paper conducts comparative experiments. Specifically, the effectiveness of the proposed algorithm is compared with other schemes, and their performances are evaluated.

(1) Fully Local Computing

In this offloading scheme, power line inspection robots process their computational tasks on their local devices without offloading tasks. The computation cost includes the local computation delay and the energy consumption for CPU task execution.

(2) Binary Offloading Computation

In this offloading scheme, power line inspection robots can choose only between local computation and computational offloading. Computational tasks are either fully processed locally or fully offloaded, with no task allocation decisions made. The computation cost for robots involves two computation modes: local computation or MEC computational offloading.

6.2. Simulation Results

(1) Task Allocation Strategy

Figure 2 illustrates the average task offloading ratio of power line inspection robots under different robot population scenarios using the improved algorithm. This ratio represents the proportion of task data offloaded compared to the total task data. The scenarios of full local computation and binary offloading are not discussed here since they only consider a single offloading mode, while the improved algorithm considers partial offloading. The figure shows that the average offloading ratio remains constant initially but then decreases as the number of robots increases. When the number of inspection robots is less than 20, the offloading ratio is 100%, meaning every robot opts to offload all computational tasks. This is due to the ample computational resources available, allowing each robot to fully offload its tasks. However, when the number of inspection robots exceeds 20, the offloading ratio gradually decreases. This is primarily because computational resources become limited, base stations get busier, and robots start to increase the proportion of local computation.

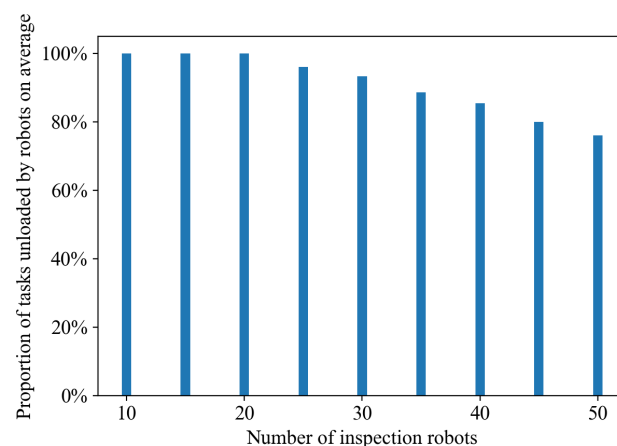


Figure 2. The proportion of tasks unloaded by robots on average under different numbers of robots.

(2) Task Computation Total Cost

Figure 3a presents the performance comparison of the all-local computation, binary offloading, and improved algorithm offloading schemes under different numbers of inspection robots. The figure shows that the improved algorithm has the lowest task computation cost, reducing it by 57.93% and 27.24% compared to the all-local computation and binary offloading schemes, respectively. This reduction is due to the improved algorithm's consideration of partial offloading, unlike the all-local and binary offloading schemes that

only consider a single offloading mode. The all-local computation scheme only uses local offloading, while the binary offloading scheme chooses between local computation and offloading, both of which fail to balance computational efficiency and energy efficiency. As the number of inspection robots increases, the total computation cost also increases. When the number of robots is small (less than 35), the total computation costs of the binary offloading and improved algorithm schemes are similar. This similarity is because the computational resources provided by the base stations are sufficient for both schemes. However, when the number of robots is large (greater than 35), the difference in total computation cost between the binary offloading and improved algorithm schemes becomes significant. The improved algorithm exhibits a lower rate of cost increase, demonstrating better performance. This is because the base stations become busier and computational resources more scarce, giving the partial offloading approach of the improved algorithm a greater advantage.

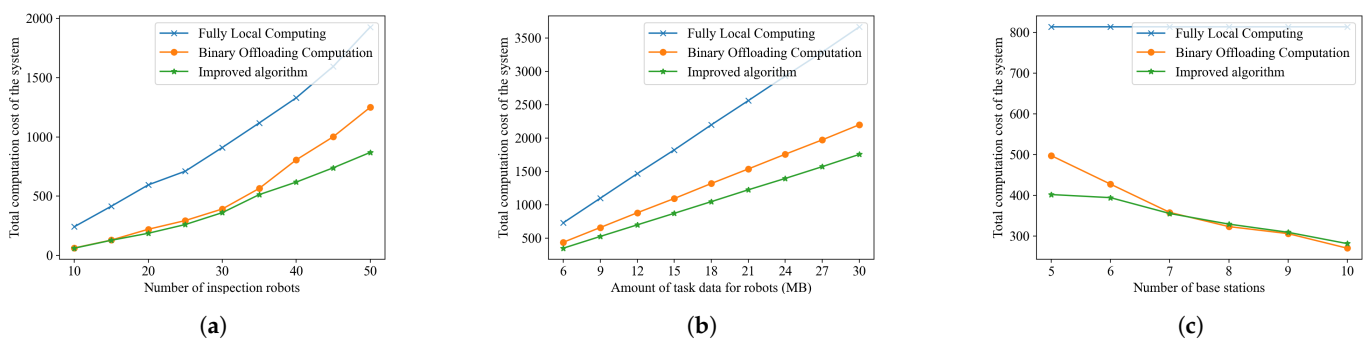


Figure 3. Comparison of the total computation cost of the system under different numbers of robots (a); different amounts of task data for robots (b); different numbers of base stations (c).

Figure 3b compares the total computational cost of different offloading schemes under varying task data sizes for all-local computation, binary offloading, and the improved algorithm. The task data size ranges from 6 MB to 30 MB. As shown in the figure, the total computational cost increases with the task data size. The all-local computation scheme incurs the highest total cost, while the improved algorithm incurs the lowest. Specifically, the total cost of the improved algorithm is 51.11% lower than the all-local computation scheme and 26.78% lower than the binary offloading scheme. Notably, the cost curves for all three schemes exhibit linear growth due to the linear relationship between task data size and both delay and energy consumption, thus their weighted sum also increases linearly.

Figure 3c shows the comparison of total task computation cost for all-local computation, binary offloading, and the improved algorithm under different numbers of base stations. The number of base stations is increased from 5 to 10, and the total task computation cost is analyzed. Results indicate that the total cost for all-local computation remains relatively constant, whereas the costs for binary offloading and the improved algorithm decrease as the number of base stations increases. This is because the additional base stations provide more computational resources, serving more inspection robots. With a constant number of robots, more base stations mean more allocated computational resources per robot, resulting in lower total computation costs. Additionally, when the number of base stations is less than 7, the improved algorithm has a lower total computation cost compared to binary offloading, due to its consideration of task allocation, which enhances cost reduction performance. Compared to all-local computation and binary offloading, the improved algorithm reduces total computation cost by 54.59% and 13.52%, respectively.

(3) Task Average Processing Delay

Figure 4a shows the task average processing delay under different task data sizes for the three computation offloading schemes: all-local computation, binary offloading, and the improved algorithm. The results indicate that the improved algorithm has the shortest task processing delay. However, the task processing delay for all-local computation is similar

to that of binary offloading, with binary offloading having the highest delay. In both the binary offloading and improved algorithm schemes, tasks can be offloaded to base stations for processing, with the delay comprising both computation and transmission components. In contrast, local computation only involves computation delay. In this scenario, the CPU resources are fully utilized for local tasks, while base stations must serve multiple robots, leading to resource competition. As the number of offloading robots increases, processing efficiency decreases. With increasing task data size, transmission delay also increases, and this increase rate surpasses that of computation delay. Therefore, although MEC servers have higher computational capacity than local devices, the average processing delay for all-local computation is lower than for binary offloading. The improved algorithm significantly reduces processing delay by adopting task allocation, where local computation and computation offloading occur simultaneously, resulting in the processing delay being the maximum of the two delays.

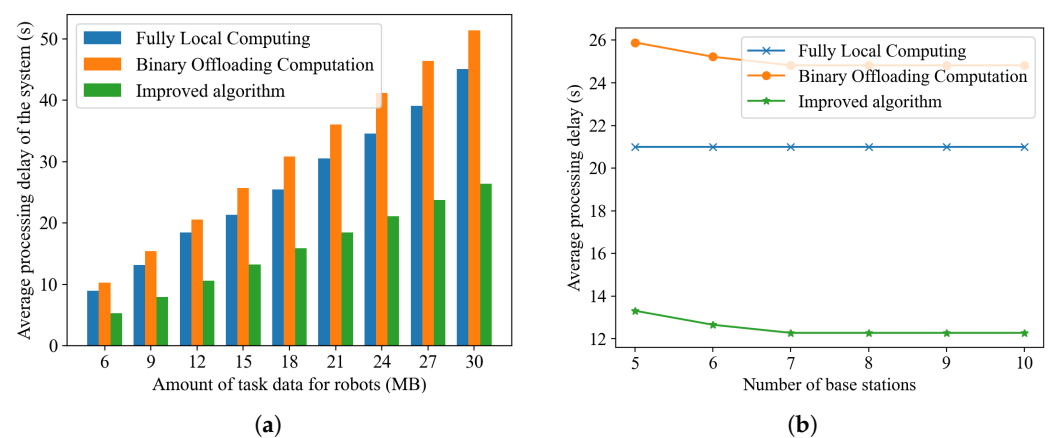


Figure 4. Comparison of the average processing delay of the system under different amounts of task data for robots (a); different numbers of base stations (b).

Figure 4b compares the average task processing delay for the three computation offloading schemes under different numbers of base stations. When the number of base stations is less than 7, the average processing delay for both binary offloading and the improved algorithm decreases with the increase in base stations. When the number of base stations exceeds 7, the average processing delay stabilizes. In the simulation setup with 10 inspection robots, the MEC servers are sufficient to meet low-delay offloading demands when the number of base stations ranges from 6 to 10, maintaining the lowest delay. Overall, the average delay for binary offloading is higher than for both all-local computation and the improved algorithm, indicating that transmission time cost is higher than computation cost. The improved algorithm reduces the average task delay by 52.15% and 38.56% compared to all-local computation and binary offloading, respectively.

(4) Task Average Processing Energy Consumption

Figure 5a shows the task average processing energy consumption for different task data sizes across the three computation offloading schemes: all-local computation, binary offloading, and the improved algorithm. The energy consumption cost for local computation is significantly higher than for computation offloading. When tasks are executed entirely on the robot's equipment, the energy consumption cost is determined by the data size and energy consumption coefficient. When tasks are offloaded to base station servers, the only energy consumption is during the transmission process, with negligible energy consumption while waiting for the base station to return the results. Numerically, the improved algorithm reduces the task average processing energy consumption by 75.63% and 15.18% compared to all-local computation and binary offloading, respectively. The improved algorithm optimizes the offloading strategy, saving energy consumption.

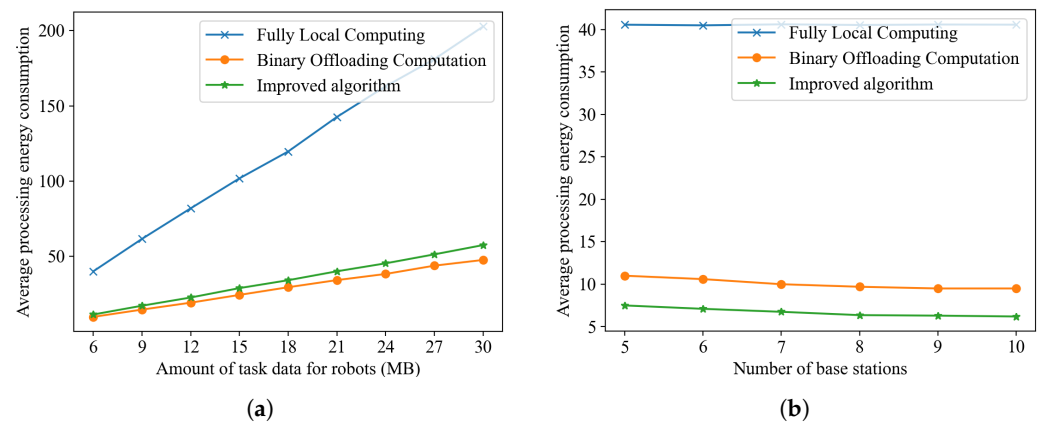


Figure 5. Comparison of the average processing energy consumption under different amounts of task data for robots (a); different numbers of base stations (b).

Figure 5b compares the task average processing energy consumption of the three computation offloading schemes—all-local computation, binary offloading, and the improved algorithm—under different numbers of base stations. Since the energy consumed while waiting for the base station to return the results is negligible, the task average processing energy consumption decreases with an increasing number of base stations in both binary offloading and the improved algorithm, as the average distance between inspection robots and base stations decreases. In contrast, the energy consumption for local computation remains constant as it only depends on the task data size. Specifically, compared to all-local computation and binary offloading, the improved algorithm reduces the task average processing energy consumption by 81.70% and 31.81%, respectively.

7. Discussion

The primary contribution of this paper lies in proposing an improved distributed computation offloading strategy based on game theory for power line inspection robots. This strategy enhances performance through task allocation and task prioritization, integrating local computing resources with edge computing resources. This integration effectively reduces latency and improves resource utilization, ultimately achieving Nash equilibrium and obtaining optimal offloading and task allocation strategies. The proposed algorithm demonstrates strong performance in edge computing scenarios for power grid inspection, providing significant theoretical reference and practical guidance for research on offloading strategies in such contexts. However, the algorithm presented in this paper still has some limitations and requires further exploration and improvement to meet the demands of more diverse scenarios.

We foresee several exciting directions for future research. First, exploring advanced optimization techniques such as deep reinforcement learning could further enhance task offloading efficiency and resource allocation in smart grids. Second, integrating more sophisticated AI algorithms can improve real-time decision-making capabilities, enabling more responsive and adaptive power grid management. Third, investigating the scalability of our proposed methods in larger and more complex smart grid environments will be crucial for practical implementation. These future directions will help to build upon the foundations laid by our current work and drive further advancements in the field of smart grid technology.

Author Contributions: Investigation, X.L. (Xu Lu), Z.N. and C.M.; Methodology, S.Y.; writing—original draft preparation, Z.N.; writing—review and editing, S.Y.; visualization, X.L. (Xi Li); supervision, X.L. (Xu Lu); funding acquisition, X.L. (Xu Lu). All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the State Grid Corporation of China Science and Technology Project (No. SGMDXTOOJSJS2100034).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

Conflicts of Interest: Author Xu Lu was employed by the company Power Dispatching Control Center of State Grid Inner Mongolia Eastern Power Co., Ltd. Authors Zhongyuan Nian and Chunfang Mu was employed by Information and Communication Branch of State Grid Inner Mongolia Eastern Power Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Abbas, N.; Zhang, Y.; Taherkordi, A.; Skeie, T. Mobile Edge Computing: A Survey. *IEEE Internet Things J.* **2018**, *5*, 450–465. [\[CrossRef\]](#)
2. Ceselli, A.; Premoli, M.; Secci, S. Mobile Edge Cloud Network Design Optimization. *IEEE/ACM Trans. Netw.* **2017**, *25*, 1818–1831. [\[CrossRef\]](#)
3. Wang, W.; Qu, R.; Liao, H.; Wang, Z.; Zhou, Z.; Wang, Z.; Mumtaz, S.; Guizani, M. 5G MEC-Based Intelligent Computation Offloading in Power Robotic Inspection. *IEEE Wirel. Commun.* **2023**, *30*, 66–74. [\[CrossRef\]](#)
4. Li, Z.; Yan, F.; Xu, X.; Xia, W.; Shen, L. Model-Based Trajectory Planning of a Hybrid Robot for Powerline Inspection. *IEEE Robot. Autom. Lett.* **2024**, *9*, 3443–3450. [\[CrossRef\]](#)
5. Devi, K.V.R.; Koithyar, A.; Lakhnopal, S.; Parmar, A.; Sethi, V.A.; Ftaiet, A.A. Edge Computing and 5G Integration for Real-time Analytics in Interoperable Smart Grids. In Proceedings of the 2023 International Conference on Power Energy, Environment & Intelligent Control (PEEIC), Greater Noida, India, 19–23 December 2023; pp. 419–424.
6. Lin, X.; Yan, F.; Li, Z.; Xu, X.; Xia, W.; Shen, L. PPO Based Computation Offloading and Resource Allocation Algorithm in Smart Grids. In Proceedings of the 2023 9th International Conference on Computer and Communications (ICCC), Chengdu, China, 8–11 December 2023; pp. 243–248.
7. Liu, P.; Wang, J.; Ma, K.; Guo, Q. Joint Cooperative Computation and Communication for Demand-Side NOMA-MEC Systems With Relay-Assisted in Smart Grid Communications. *IEEE Internet Things J.* **2024**, *early access*. [\[CrossRef\]](#)
8. Saleh, T.; Anwar, A.; Elmusrati, M.; Kauhaniemi, K.; Välisuo, P. Virtualized Intelligent Relaying of Smart Grid Over 5G Network. In Proceedings of the 2024 International Workshop on Artificial Intelligence and Machine Learning for Energy Transformation (AIE), Vaasa, Finland, 20–22 May 2024; pp. 1–6.
9. Fakhari, A.; Mostashfi, A. LinBot—Design, Analysis, and Field Test of a Novel Power Transmission Lines Inspection Robot. In Proceedings of the 2019 7th International Conference on Robotics and Mechatronics (ICRoM), Tehran, Iran, 20–21 November 2019; pp. 132–137.
10. Zou, D.; Liu, L.; Jiang, Z.; Luo, B.; Quan, W. Design and Dynamic Simulation of Single-arm Inspection Robot System for Transmission Lines. In Proceedings of the 2023 International Conference on Advanced Robotics and Mechatronics (ICARM), Sanya, China, 8–10 July 2023; Volume 5, pp. 1210–1213.
11. Xiaojie, Z.; Yi, R.; Jian, Z. The research application of inspection robot for the smart grid. In Proceedings of the 2011 International Conference on Consumer Electronics, Communications and Networks (CECNet), Xianning, China, 16–18 April 2011; pp. 2776–2779.
12. Sumanth, S.; Srivastav, A.; Sagar, R.; Malik, M.A.; Vishwanath, M.N. Mechanism, Design and Kinematics for a Transmission Line Inspection Robot. In Proceedings of the 2021 2nd International Conference for Emerging Technology (INCET), Belagavi, India, 21–23 May 2021; pp. 1–7.
13. Zhu, P.; Zhu, A.; Zhang, Q.; Wang, Y.; Zhang, X.; Cao, G. Design of Gibbon-Like Crawling Robot for High Voltage Transmission Line Inspection. In Proceedings of the 2019 16th International Conference on Ubiquitous Robots (UR), Jeju, Republic of Korea, 24–27 June 2019; Volume 5, pp. 16–20.
14. Zhu, A.; Tu, Y.; Zheng, W.; Shen, H.; Zhang, X. Design and Implementation of High-Voltage Transmission Line Inspection and Foreign Bodies Removing Robot. In Proceedings of the 2018 15th International Conference on Ubiquitous Robots (UR), Honolulu, HI, USA, 26–30 June 2018; pp. 852–856.

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.