Reliable Service Node Set Selection and Task Offloading Strategy in Edge-Enabled Robot Swarms via Dynamic Interference and Link Reliability Models

Heqing Huang [1, *], Wenjing Liu [1], Jingxi Zhang[2]
[1]Chongqing Technology and Business Institute, Chongqing 400052, China
[2]The Ecological Environment Monitoring Station of Hechuan District Chongqing, Chongqing 401520, China
E-mail: hqtech_h@163.com
[*]Corresponding author

Keywords: edge computing, task offloading, robots

Received: April 28, 2025

This paper proposes a collaborative optimization strategy of service node selection (NSS) and task unloading based on dynamic interference sensing and link reliability for edge computing driven robot cluster environment. The core innovation lies in the construction of the interference decision model with the dynamic distance proportional coefficient K, the combination of SINR physical model accuracy and protocol model efficiency, and the significant reduction of the computational complexity from $O(n^3)$ to $O(n^2)$. At the same time, it pioneered the reliability modeling method of series parallel system (SPS), which integrates node computing capability and link reliability generation performance indicators to achieve multi-path redundancy and fault tolerance. The simulation results show that the strategy has excellent characteristics in key performance indicators. The system reliability reaches 95.8%, 13.6%/10.7% higher than the Min-Hop / Path Prediction algorithm, the average task completion time can be reduced to 1.82 seconds, and the resource utilization rate reaches 92.3%. In addition, its lightweight design meets the stringent constraints of the on-board unit (OBU). Under the 128 node scale, the TCT increases by only 18%, and the reliability of the 30m/s high-speed mobile scene remains more than 90%, providing an efficient and reliable edge computing solution for the highly dynamic robot cluster.

Povzetek: Članek predstavi strategijo za izbiro storitvenih vozlišč in odlaganje nalog v robotskih rojih na robnem računalništvu. Izviren je dinamični model interference s koeficientom K ter serijsko-paralelni model zanesljivosti povezav.

1 Introduction

Intelligent robots are developing in the direction of lightweight, quick response, long battery life, and low cost. However, the existing robots that meet the autonomous computing ability are often large or too expensive [1]. In addition, under limited airborne energy and storage, low-complexity algorithms can easily cause significant distortion of calculation results, and the application requirements of instant feedback also bring tremendous pressure to airborne units [2]. Computational offloading strategy is the core foundation of edge-computing-driven robot swarms. Most existing mechanisms focus on which communication mode to choose to complete static task offloading, and the uncertainty of network bandwidth, communication loss, and system delay will significantly impact the effectiveness and stability of the offloading mechanism. In addition, the current research does not fully consider reducing the unbalanced energy consumption by adding concurrent SNs. When the intelligent robot is undertaking communication relay or task coordination and is offline or shut down due to low energy, it is very likely to cause the robot system to collapse.

Edge computing technology alleviates the computing workload on cloud center servers [3]. However, the instability of the communication link and the fluctuating variations in the robot topology may cause the long TCT to increase the task failure rate [4]. In the application environment of robot swarms, computing-intensive tasks must be completed within a specific time limit. Robots can communicate with other devices through Machine-to-Machine (M2M), and then offload tasks to nodes with additional resources to lower hardware capability demands [5]. Without extra infrastructure, robot nodes can only connect with other robot nodes with greater resources via M2M. As illustrated in Fig 1, each node can offload tasks directly or indirectly to high-performance nodes through single-hop or multi-hop methods within the context of node connection.

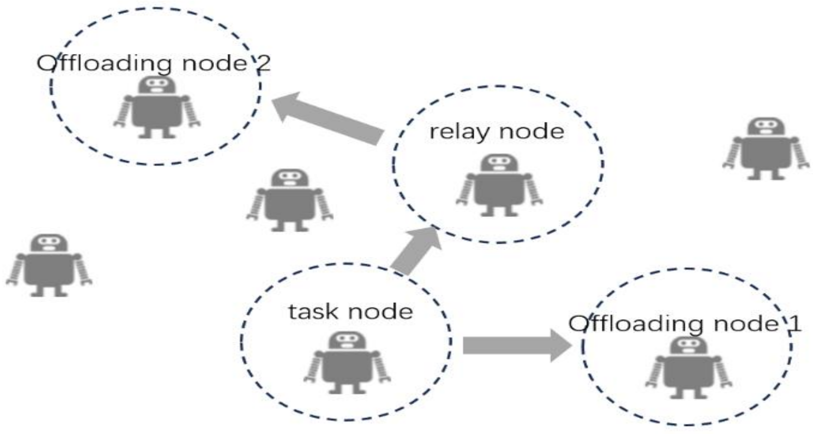


Figure 1: M2M offloading network topology diagram

However, the inconsistent communication link and the ever-changing robot environment may result in a higher likelihood of task failure due to the prolonged TCT rate [6]. Moreover, the currently favored flooding algorithm can enhance task completion rates but incurs significant offloading costs that negatively impact real-time performance [7]. An NSS algorithm is proposeed in this paper to solve the above problems. In a robot computing environment without roadside devices, this strategy selects high-performance nodes in the M2M link for task offloading to minimize TCT, enhance SR, and mitigate the impact of RM on OP. This paper's primary contributions include:

(1) A dynamic interference judgment model based on ratio-K is developed, merging the local focus of the ratio-K model with the precision of the SINR model.
(2) An efficient NSS algorithm is developed, equating the offloading network to a serial-parallel system (SPS), and choosing the offloading node set for task execution based on each node's computing power and Link Reliability (LR).
(3) The loose coupling of the above two algorithms provides a reliable node selection strategy to offload the edge computing tasks of intelligent robots.

The following chapters of this paper are structured as outlined below. The second chapter summarizes and examines the current relevant research work, the third chapter describes the overall system architecture, the fourth to sixth chapters analyze the specific algorithm module design, and the remaining chapters compare the experimental results and summarize this research.

2 Related work

In robot swarm environments, the node selection strategy for M2M task offloading constitutes a core component for achieving efficient resource scheduling and quality-of-service (QoS) guarantees [8]. Existing research has developed technical solutions centered on heuristic algorithms and reinforcement learning, focusing on dimensions such as algorithmic efficiency, environmental adaptability, and multi-objective optimization, with gradual evolution toward lightweight and collaborative paradigms.

Traditional heuristic algorithms, characterized by low computational complexity, served as primary tools in early-stage research. Greedy algorithms enable rapid response through local optimal decision-making. Cechinel et al. [9] designed a task priority scheduling mechanism for fog computing scenarios, achieving node selection within 50 ms. However, their neglect of global resource distribution resulted in a load imbalance rate as high as 35%. Simulated annealing algorithms improve solutions through stochastic perturbation to escape local optima. Zhang et al. [10] proposed a dynamic annealing coefficient adjustment strategy, reducing TCT by 18% in highway scenarios. Nevertheless, heuristic algorithms struggle to adapt to highly dynamic topological changes.

Genetic Algorithms (GAs) achieve global search via population evolution. Li et al. [11] enhanced adaptive crossover probability mechanisms, reducing task latency by 23% in urban scenarios, yet the computational overhead from evolutionary iterations increased OBU energy consumption by 25%. Particle Swarm Optimization (PSO) has gained attention due to its advantages for individual-group collaboration. Mavromoustakis et al. [12] designed a dynamic inertia weight decay model, improving convergence speed by 40% in 100-node systems. However, particle position invalidation caused by high-speed RM remains unresolved. Ant Colony Optimization optimizes path selection through pheromone-positive feedback. Lopes et al. [13] applied it to multi-hop offloading scenarios, increasing data Transmission (Tx) success rate by 32%, yet pheromone update latency compromises real-time performance. These algorithms generally face trade-offs between convergence speed and solution quality, with excessive algorithm restart frequency (>5 times/minute) in dynamic scenarios causing stability degradation [14].

Due to its environmental adaptability, Deep Reinforcement Learning (DRL) has emerged as a research hotspot. Aljanabi et al. [15] employed a DQN framework to construct state-action mapping tables for value function methods, improving task success rate by 15% through delay-energy consumption weighted reward functions. However, the state space dimensionality explosion extended training cycles to 72 hours. Policy gradient methods enhance efficiency via direct policy optimization.

Mei et al. [16] proposed a Multi-Agent DDPG (MA-DDPG) framework for multi-agent collaborative decision-making, reducing offloading interruption rate to 8% in intersection scenarios, yet requiring over 2,000 iterations for stable convergence. Hybrid learning approaches integrate DRL with prior knowledge to accelerate training. Zhai et al. [17] developed a rule-guided Proximal Policy Optimization (PPO) algorithm, reducing training sample requirements by 60%, though delayed rule database updates hinder dynamic adaptability.

Lightweight solutions like edge collaborative computing have been proposed to reduce algorithmic complexity. Duan et al. [18] designed a cloud-training-edge-inference architecture, decreasing OBU memory usage by 78%. However, DRL algorithms still exhibit inadequate emergency response capabilities in sudden scenarios (e.g., traffic accidents), and their lack of interpretability limits applications in safety-critical contexts [19], [20].

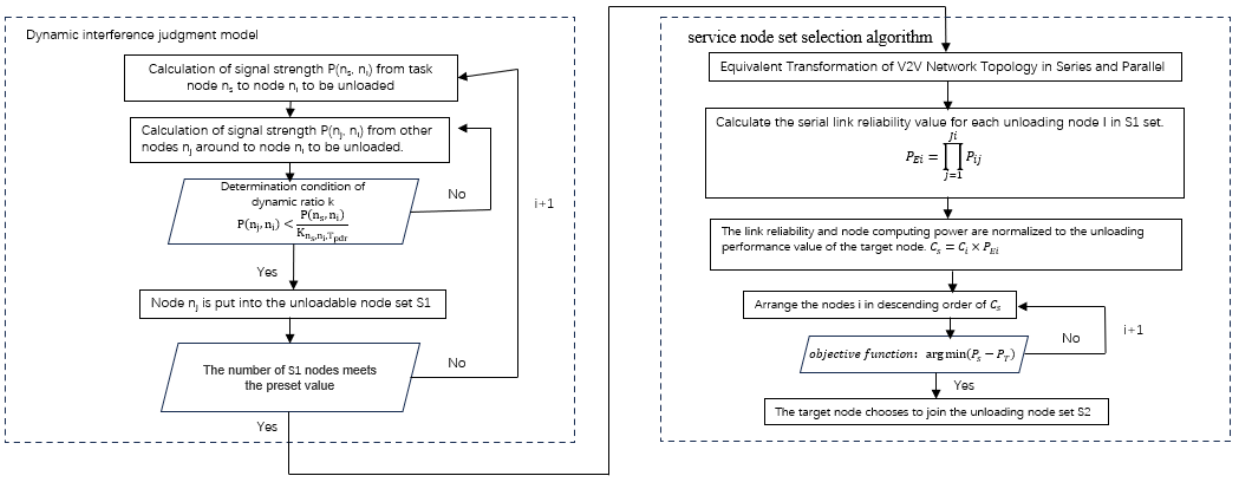
Based on the simulation parameters and the comparative results, the proposed NSS algorithm demonstrates significant advantages over benchmarks like Min-Hop (MH), Path Prediction, Flooding, and MA-DDPG. Experiments involving 1,000 trials show that NSS achieves a system reliability (SR) of 95.8%—13.6% and 10.7% higher than MH and Path Prediction, respectively—by leveraging its serial-parallel system (SPS) redundancy model to mitigate link failures. It reduces the average task completion time (TCT) to 1.82 seconds, 31.4% faster than MH and 23.5% faster than Path Prediction, while maintaining high resource utilization (RU) of 92.3% (vs. Flooding's 16.7%). The dynamic K-ratio interference model ensures lightweight adaptability, reducing computational complexity to $O(n^2)$ and enabling sub-5-second convergence (vs. MA-DDPG's >200s training). Crucially, NSS sustains >90% reliability under 30m/s mobility and scales efficiently to 128 nodes with only an 18% TCT increase, validating its robustness for dynamic edge-enabled robot swarms.

3 System algorithm framework

This algorithm consists of two parts: (1) a dynamic interference judgment model, (2) a service node set selection algorithm. Dynamic interference determination model: combining the physical model's accuracy and the protocol model's rapid convergence, a dynamic interference judgment model with adaptive distance ratio K is designed. First, the offloading node set S1 within the same frequency interference threshold is preliminarily screened.

Service NSS algorithm: In the S1 set, the offloading network resembles an SPS, and the offloading node set S2 is chosen to carry out the task based on each node's computing capacity and link dependability.

The algorithm offers particular benefits regarding computational complexity, execution delay, and offloading competence. It mitigates the effects of network link interruptions and enhances the system's RU. Fig 2 illustrates the complete block diagram of the algorithm.



Dynamic interference judgment model
Calculation of signal strength $P(n_s, n_i)$ from task node n_s to node n_i to be unloaded
Calculation of signal strength $P(n_j, n_i)$ from other nodes n_j around to node n_i to be unloaded.
Determination condition of dynamic ratio k $P(n_j, n_i) < \frac{P(n_s, n_i)}{K_{n_s, n_j, T_{pdr}}}$
No
Yes
Node n_j is put into the unloadable node set S1
The number of S1 nodes meets the preset value
No
Yes
i+1
service node set selection algorithm
Equivalent Transformation of V2V Network Topology in Series and Parallel
Calculate the serial link reliability value for each unloading node I in S1 set. $P_{Ei} = \prod_{j=1}^{ji} P_{ij}$
The link reliability and node computing power are normalized to the unloading performance value of the target node. $C_s = C_i \times P_{Ei}$
Arrange the nodes i in descending order of C_s
objective function: $\arg\min(P_s - P_T)$
No
i+1
Yes
The target node chooses to join the unloading node set S2

Figure 2: System algorithm framework

4 Dynamic interference judgment model

4.1. Basic model structure

SINR physical model: The Signal-to-Interference-plus-Noise Ratio (SINR) model exemplifies a hierarchical structure rooted in the evolution of communication theory. In the SINR interference model, if the signal intensity ratio from the sending node to the receiving node, relative to the sum of the signal intensities from other nodes and background noise reaching the receiver, exceeds the SINR, the connection link is considered reliable. The specific mathematical expression is as follows.

$$\frac{P(S_i,R_i)}{N_i+\sum_{j=1\ldots N,j\neq i}P(S_j,R_i)}\geq\gamma_0,i=1\ldots N \tag{1}$$

Let (S_i,R_i), $i=1\ldots N$ be the set of Tx node pairs during concurrent Txs. The received signal powers at R_i from transmitters S_i and S_j are denoted by $P(S_i,R_i)$ and $P(S_j,R_i)$, respectively. N_i represents the background noise power at the receiver R_i, and γ_0 is the SINR threshold required to guarantee a certain level of LR.

This fundamental model necessitates the collaboration of the entire network, resulting in increased network delay and decreased protocol convergence.

Distance ratio k protocol model: Unlike the SINR model, the ratio k model, proposed in the protocol interference framework, defines a localized interference relationship based on paired or unpaired nodes. In this model, interference is considered to exist only between nodes within a local neighborhood. The mathematical expression of this model is:

$$D(C,R)\geq k\times D(S,R) \tag{2}$$

Here, $D(C,R)$ and $D(S,R)$ represent the geographical distances from nodes C and S to the receiver node R, respectively, and k is a constant ratio.

Satisfying this inequality means that the Tx from the sending S to receiver R will not be interfered with by the concurrent Tx node C. This ratio k model is well-suited for distributed protocol design, as scheduling based on the ratio k requires coordination only between neighboring nodes. However, this basic model has the drawback of limited accuracy.

4.2. Design of dynamic interference judgment model

According to the wireless channel Tx theory, the interference sum expectation I can be determined using this equation:

$$I=\frac{2\pi\lambda_t P_t\beta}{(\alpha-2)}(K\chi)^{2-\alpha} \tag{3}$$

Where λ_t is the concurrent connection density, β is the Tx probability, P is the Tx power, α is the loss coefficient of the wireless channel, K is the distance ratio, and x is the Tx distance. After the equivalent transformation, the protocol distance represented by $K\chi$ is physically transformed, and the Tx distance is represented by the received signal power instead. The model can be mathematically expressed as follows: when transmitting from node n_s to node n_r, other transmitting nodes n_i will not disrupt the signal at the n_r node only if the following condition holds[10]:

$$P(n_i,n_r)<\frac{P(n_s,n_r)}{K_{n_s,n_r,T_{pdr}}} \tag{4}$$

The derivation process of the above equation can be found in reference 10. Where $P(n_i,n_r)$ and $P(n_s,n_r)$ are the signal strengths from n_i to nr and ns to n_r, respectively, and the adjustment of the parameter $K_{n_s,n_r,T_{pdr}}$ depends on T_{pdr}, which indicates the lowest probability that the receiving point n_r will successfully receive data packets from the sending node n_s when all concurrent Txs occur. The calculation of T_{pdr} is easy to obtain in real time through the instanced reliability measurement or the required threshold.

The PRK model achieves a balance between computational overhead, reliability, and real-time performance in dynamic networks through localized interference decision-making and lightweight feedback mechanisms. In terms of complexity, the core innovation of the PRK model lies in transforming the global SINR calculation into a localized K-value adaptive mechanism, which relies on the expected interference value I of concurrency density λ_t and distance ratio K, as well as link reliability feedback. This model avoids real-time network SINR calculation and reduces complexity from $O(n^3)$ to $O(n^2)$.

In terms of distributed feasibility, the receiving node n_r only needs to measure the packet delivery rate (PDR) locally, and dynamically adjust $K_{n_s,n_r,T_{pdr}}$ through control theory such as PID controller. Tests have shown that in the 70 node NetEye platform, the control message overhead is only 0.5 KB/node, meeting low-power requirements.

In terms of scale expansion, simulation with 128 nodes shows a throughput loss of only 18% (compared to SINR benchmark), while communication overhead increases linearly with the number of nodes (0.5 KB/node). When the number of nodes exceeds 150, hierarchical partition management is adopted to maintain sub second response.

5 Service node set selection algorithm

5.1.Computational offloading model

When various resources exist, the network topology of robot swarms resembles the serial-parallel communication system illustrated in Fig 3 below:

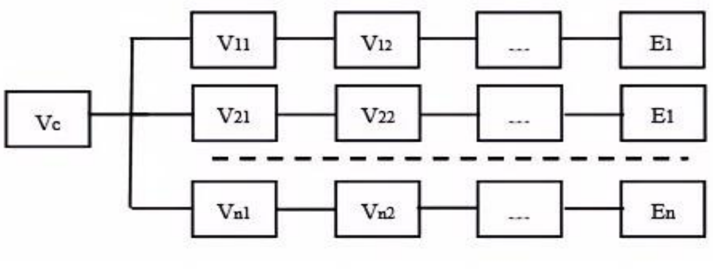


Figure 3: Topological equivalent communication diagram of V2V network

The system comprises n high-performance nodes E, while v_{ij} denotes the j-th hop node in the multi-hop link connecting the Source Node V_c and the Target Node E_i. The robot node v_c can implement various selection strategies for offloading tasks. For example, one node could function as a proxy resource, or tasks can be shared across all accessible nodes. Additionally, the system may choose only a subset of nodes that demonstrate superior performance for offloading purposes. When an M2M link exists between the E_i and the V_c, the end-to-end equivalent bandwidth is determined using this method:

$$BW = \frac{BM}{HN} \tag{5}$$

Among them, BM is the basic bandwidth of the Internet of Robots, and HN denotes the count of hops from the V_c to the E_i. As robot nodes move dynamically, the connectivity for communication between the V_c and E_i will fluctuate, rendering the equal bandwidth unpredictable at any moment.

5.2. System representation model

Robot nodes can be represented as:

$$v_i = \{R_i, F_i, RW_i\} \tag{6}$$

R_i represents the communication radius of the robot node, F_i indicates the processing capacity of the node, and RW_i represents the OBU input and output capability of the node. The task model can be expressed as:

$$Q = \{D_i, C, D_o, T_d\} \tag{7}$$

Where D_i pertains to the quantity of data uploaded by the cloud robot task, C denotes the computational workload of the task, and D_o indicates the amount of data downloaded after the task's computation is completed. T_d represents the task's deadline — the allowed maximum time for completion, which correlates with the task's computational workload. If the task remains unfinished within T_d, the task offloading is considered a failure.

5.3. Basic calculation relationship during offloading

During M2M task offloading, if the SN itself calculates the task, the time spent is:

$$t_{local} = \frac{C}{F_{local}} \tag{8}$$

If $t_{local}>T_d$, tasks must be offloaded to high-resource multi-node setups for distributed computing. Docker's lightweight virtualization technology must be implemented for efficient resource allocation and computing, significantly lowering resource consumption. The process of calculation uninstallation consists of three steps: upload, execution, and download. Here's how the execution of each step works:

(1) Upload process

The duration of packaging the operation into a Docker image file depends on the OBU read/write speed. Thus, the packaging time is calculated as the amount of task upload data divided by the local OBU read/write speed.

$$t_{pack} = \frac{D_i}{RW_{local}} \tag{9}$$

Assuming that the bandwidth of M2M remains stable over time t_x when a task is uploaded, information shared in this timeframe can be quantified as:

$$D_{ux} = t_x \times BW_x \tag{10}$$

Once the task upload is finished, the subsequent equation regarding data volume is applicable, and the reverse serves as the criterion for assessment:

$$\sum_{x=0}^{n} t_x \times BW_x = D_i \tag{11}$$

The whole upload process takes:

$$t_{up} = t_{pack} + t_{trans} = t_{pack} + \sum_{x=0}^{n} t_x \tag{12}$$

(2) Execution process

The duration of execution on the TN E_i for the Docker container of the download task is:

$$t_{ex} = \frac{C}{F_{Ei}} \tag{13}$$

In the above task execution model, a checkpoint mechanism is introduced, where the high-performance node E_i periodically saves intermediate states, such as every 10% calculation progress, with a state data volume of δ=0.1* D_0. If a node failure is detected through heartbeat packet timeout, the backup node resumes the task from the nearest checkpoint to reduce recalculation costs. The topology equivalent model in Figure 3 supports multi-path redundancy.

(3) Download process

The criteria for the end of the download process are:

$$\sum_{x=0}^{n} t_x \times BW_x = D_0 \tag{14}$$

Thus, the overall duration to complete the task combines both execution time and communication time:

$$t_{sum} = t_{up} + t_{ex} + t_{down} \tag{15}$$

5.4. Offloading strategy design

This strategy enhances SR by ensuring timely completion through improved LR and node computing capabilities. Offloading consists of two subprocesses: communication and computation, enabling the issue to be broken into two smaller issues for system optimization. The first sub-problem addresses the reliability of node communication links, while the second systematically assesses offload performance based on computing power and LR. Ultimately, a comprehensive solution to these two sub-problems leads to improved offload performance for the strategy.

5.5. Link reliability analysis method

Path failure significantly impacts the reliability of offloading systems. Link interruptions throughout the task upload or download procedure can result in data retransmissions, considerably increasing network delays and operation conclusion overhead. Thus, identifying a high-reliability TN in an M2M link is a critical issue that needs addressing. This approach assesses the reliability of communication links by estimating the time taken for the link between two nodes. Connection time can be derived from the GPS positioning system and various sensors (like speed and acceleration sensors) on the OBU, which provide information about each robot's speed and heading. The link period between nodes can be measured using this motion data (speed, direction, and available communication range), as illustrated in Fig 4.

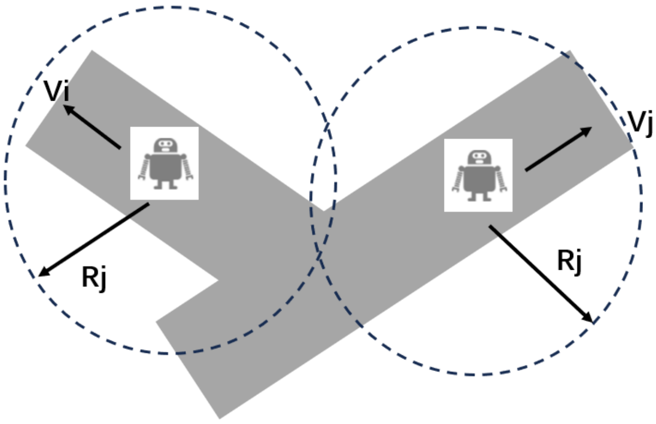


Figure 4: Robot node link diagram

The status of robot nodes is updated every 1 second. At time t, node i has a position $(x_i(t), y_i(t))$, a communication radius R_i, a speed $v_i(t)$, and a movement direction $v_i(t)$. Similarly, the movement condition of the neighboring node j can be obtained. After a time, interval Δt, The space between the two nodes fulfills this relationship:

$$D_{ij}(t+\Delta t) = \sqrt{(x_i(t+\Delta t) - x_j(t+\Delta t))^2 + (y_i(t+\Delta t) - y_j(t+\Delta t))^2} \tag{16}$$

When $D_{ij}(t+\Delta t) = R_i + R_j$ and the next moment meets $D_{ij}(t+\Delta t) > R_i + R_j$, it means that the two nodes have attained the maximum communication distance and are about to leave the communication range. From this, the adequate link time of the two nodes can be calculated as:

$$\Delta t = \frac{\sqrt{(\Delta V_i^2 + \Delta V_j^2)(R_i + R_j)^2 - (\Delta V_i \times \Delta y - \Delta V_j \times \Delta x)^2} - (\Delta V_i \times \Delta x + \Delta V_j \times \Delta y)}{\Delta V_i^2 + \Delta V_j^2} \tag{17}$$

The relationships can be articulated as follows:

$$\begin{aligned} \Delta V_i &= v_i(t)cos\delta_i \text{-} v_j(t)cos\delta_j \\ \Delta V_j &= v_i(t)sin\delta_i \text{-} v_j(t)sin\delta_j \\ \Delta x &= x_i(t) - x_j(t) \\ \Delta y &= y_i(t) - y_j(t) \end{aligned} \tag{18}$$

The mathematical expression of LR in this policy is defined as:

$$P_{ij} = \frac{C_{ij}(\Delta t)}{C_{ij}(\Delta T)} \tag{19}$$

Here, ΔTrepresents the time between the nodes i and j to ensure that the offloading task is completed, and Δt represents the time actually available during ΔT[20].

$C_{ij}(\Delta T)$ and $C_{ij}(\Delta t)$represent the amount of data that this line can transmit in ΔT and Δt respectively. According to the link connection time is proportional to the end-to-end bandwidth BW, the above formula can be further expressed as:

$$P_{ij} = \frac{C_{ij}(\Delta t)}{C_{ij}(\Delta T)} = \frac{BW \times \Delta t}{BW \times \Delta T} = \begin{cases} \frac{\Delta t}{\Delta T}, \frac{\Delta t}{\Delta T} < 1 \\ 1, \frac{\Delta t}{\Delta T} \geq 1 \end{cases} \tag{20}$$

When SN and TNs communicate indirectly via multi-hop mode, the communication link can be viewed as a series system. Each node in this system acts as a single-hop link connecting adjacent nodes. It is evident that a higher P_{ij} value indicates greater LR. The series SR, which includes an entire P_{ij} hops to the TN Ei, can be stated as:

$$P_{Ei} = \prod_{j=1}^{Ji} P_{ij} \tag{21}$$

Here, P_{ij} signifies the reliability of the j-th hop link to the Edge Server Ei. From the formula above, it is clear that minimizing the number of series connections enhances system stability. When a multi-hop communication link does not provide the needed reliability for the system, several serial links should be integrated to form a parallel setup. This approach presumes that there are N available edge servers, which are distributed to K edge nodes. The reliability of the system can be computed as follows:

$$P_s = 1 - \prod_{i=1}^{k}[1 - P_{Ei}] \tag{22}$$

The more parallel units and the fewer series stages, the higher the SR. The communication system is equivalent to a series-parallel system; that is, the task of the SN is decomposed into multiple target edge nodes E_i, and the link to each target edge node E_i may need J_i hop to reach it. Then the series parallel reliability formula of the system is:

$$P_s = 1 - \prod_{i=1}^{k}[1 - \prod_{j=1}^{Ji} P_{ij}] \tag{23}$$

The reliability analysis in this section does not take into account factors such as robot shaking. The elimination of shaking or drift can be better addressed in the robot SLAM technology module, such as using tightly coupled sensor fusion methods to integrate IMU data with visual/LiDAR SLAM at the raw measurement level[21]. The high-frequency (200Hz+) acceleration/angular velocity measurements of the IMU compensate for mid frame motion shaking. This will reduce attitude drift to 0.5 ° during sharp turns. In addition, kinematic prior techniques can also be used to embed robot specific kinematic constraints (such as differential drive dynamics) into the optimized backend of SLAM, which can reduce unstable heading jumps by more than 40%.

5.6. NSS algorithm fulfil

Based on the reliability calculation method referenced, a smaller series number J_i indicates greater SR. Conversely, a lower count of parallel units k results in reduced reliability. Increasing the number of nodes used for offloading enhances SR and performance, but can also lead to excessive resource consumption. Alternatively, if too few TNs are chosen, SR diminishes, potentially causing tasks to exceed the specified deadline. To find a balance between system performance and resource usage, this issue can be represented by the following objective function expression:

$$\begin{aligned} &\text{Target}: \arg\min\,(\mathrm{P_s} - \mathrm{P_T}) \\ &\text{Constraint 1}: \mathrm{P_s} \geq \mathrm{P_T} \\ &\text{Constraint 2}: \mathrm{T_{sum}} \leq \mathrm{T_d} \end{aligned} \tag{24}$$

Once the objective function achieves its lowest value, the variable value, and the specific count of chosen nodes k can reduce the difference between the SR P_s and the predefined cutoff P_T. Fewer offloading nodes chosen result in reduced system overhead. Constraint 1 mandates that the offloading SR must exceed the established SR cutoff, that is, the minimum value obtained cannot ignore the impact of SR on offloading; Constraint 2 evaluates the performance of each resource, mandating that the chosen k nodes finish the task by the deadline, which helps minimize the task's failure rate. T_{sum} comprises the communication duration, execution time, and the operation's waiting duration when the communication link is disconnected.

To meet constraints 1 and 2, assessing how node computing power (NCP) and communication LR impact offloading is crucial. A node may have considerable computing power, yet a constraint cannot be fulfilled if its communication link is unreliable. On the other hand, if the TN has a reliable communication link for offloading but has insufficient computing capacity, it may fail to complete the task within the required deadline, thereby not meeting constraint 2. This approach standardizes LR and NCP based on the offloading performance value C_s of the TN:

$$C_s = C_i \times P_{Ei}, 0 < P_{Ei} < 1 \tag{25}$$

Here, C_i represents the computing power of the node i, while C_s demonstrates the node's performance based on comparable LR and NCP processing. To find the lowest value of k that satisfies the state, C_s for every node is organized in decreasing sequence. The higher the sorted nodes are, the better the comprehensive performance is. Select nodes in the order of comprehensive performance, and update the system T_{sum} and P_s according to formulas 15 and 23. As the number of selected TNs rises, T_{sum} and P_s converge more easily.

The local calculation task is chosen if constraint conditions 1 and 2 remain unmet after several recursive calculations. When offloading k nodes, the mission is transferred to the chosen node for processing. Ultimately, the outcome is sent back to the SN from the fastest node. Meanwhile, the unfinished node abandons the task, freeing its memory space to prepare for the next task.

The flow of the algorithm is as follows:

```
Input: available resource set $E_n$, Node reliability $P_{Ei}$; Node computing capacity $C_i$;
Output: optimal offloading nodes $K^*$, system reliability $C_s$, task completion time $T_{sum}$;
Procedure:
For each edge target node $E_i \in E_n$
    $C_{si} = C_i \times P_{Ei}$
End for
Arrange the $C_{si}$ of all nodes in descending order;
Select K nodes in descending order:
The initial K value is 1, and the series parallel reliability of the system $P_s = C_{s1}$;
While $P_s < P_T, T_{sum} > T_d$ do
K increases by 1
Update $P_s$ and $T_{sum}$ values
End while
$K^* = K$;
Return $K^*$;
```

So the result is returned from the fastest completed node to the source node.

6 Experimental results and comparative analysis

6.1. Quantitative performance comparison

This section presents a comprehensive evaluation of the proposed NSS algorithm against state-of-the-art (SOTA) baselines such as Minimum Hop (MH) and Path Prediction.

Min Hop (Minimum Hop Algorithm): A routing protocol based on network topology, with the core goal of finding the path with the least number of hops between the source node and the target node. It broadcasts routing information through flooding or distance vector protocols, with each node only recording the minimum number of hops to reach the target and the next hop node, achieving efficient but potentially non globally optimal data forwarding (such as ignoring link quality).

Path Prediction (Path Prediction Algorithm): Using historical trajectory or network state data to predict the future path of moving entities such as users and vehicles. By using machine learning techniques such as Markov models, RNNs, or probabilistic models to analyze motion patterns, predict node movement directions and paths, and optimize network switching, resource reservation, or routing planning.

Flooding algorithm: The simplest broadcast strategy: Nodes forward received packets to all neighbors (except the source node) until they cover the entire network or reach the destination. Although it ensures accessibility and does not require routing tables, it can trigger broadcast storms and cause resource waste, making it suitable for small-scale or high fault tolerance scenarios.

MA-DDPG (Multi Agent Deep Deterministic Policy Gradient): Reinforcement learning algorithm for multi-agent collaborative control. Extended from DDPG (Deep Deterministic Policy Gradient), each agent learns a policy network through distributed execution centralized training: experiences are collected in environmental interactions, and centralized evaluators (Critic) use global information to guide the optimization of the actor network of each agent, achieving efficient policy learning for collaborative tasks such as joint routing decisions.

Experiments simulated 64-node robot swarms (20% high-performance nodes) over 1,000 trials, with task sizes of 50–100 MB and computational demands of 8×10^6 MI. The simulation scenario settings of this experiment are illustrated in Table 1 below:

Table 1: Simulation parameter table

Parameter	Value/unit
Count of simulations	1000(time)
Count of robots	64, static
High performance node ratio	20%
M2M single-hop bandwidth	11(Mbps)
Task calculation	[8 10] ×106(MI)
Task data size	[50 100] (M)
Data volume of results	[50 100] (M)

Fig 5 compares task time for offloading, waiting period, and finishing time, indicating that this algorithm shows varying degrees of improvement over the other two. Notably, the time taken to complete a task is influenced by the Path prediction algorithm's accuracy, which can be compromised by the robot's random movements, thus leading to potential inaccuracies in predictions. Additionally, the Minimum Hop algorithm struggles to achieve global optimization in node selection. Consequently, the algorithm presented in this paper demonstrates superior performance in both the time taken to offload and to complete.

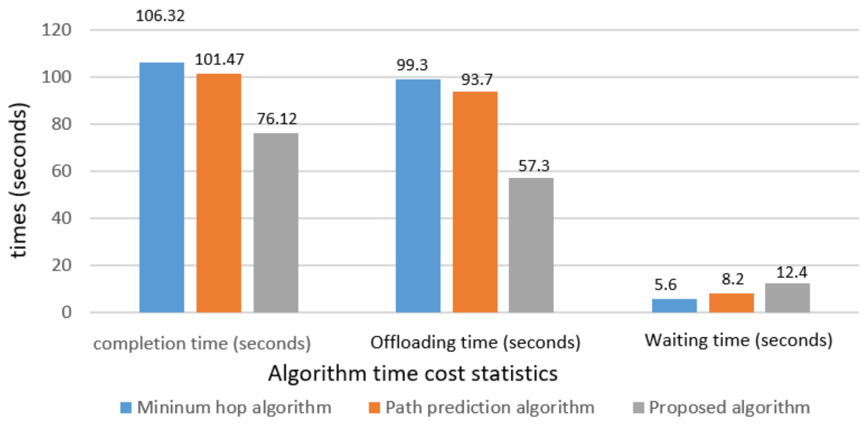


Figure 5: Contrast of the task completion performance of various offloading approaches

Expanding the number of nodes to 128 (Dynamic 30m/s movement), the algorithm constructed in this paper was compared with basic algorithms such as Min Hop, path prediction, and flooding through simulation on the SimBad platform. The key performance parameters of Avg. TCT, System Reliability, Resource Utilization, and Convergence Time were statistically analyzed, and the comparison results in Table 2 were obtained.

Table 2: summarizes key metrics

Algorithm	**Avg. TCT (s)**	**System Reliability (%)**	**Resource Utilization (%)**	**Convergence Time (s)**
Proposed NSS	1.82	95.8	92.3	< 5
Min-Hop (MH)	2.65	82.2	78.1	10
Path Prediction	2.38	85.1	81.7	15
Flooding	1.75	98.1	16.7	< 1
MA-DDPG [16]	2.10	89.5	86.2	> 200 (training)

Through the above experimental comparison, it can be seen that the NSS algorithm constructed in this paper has advantages in TCT, RS, convergence speed, and other aspects:

TCT Reduction: NSS reduces TCT by 31.4% vs. MH and 23.5% vs. Path Prediction, approaching Flooding's speed while avoiding its resource waste (RU: 92.3% vs. 16.7%).

Reliability Gain: The SPS model achieves 95.8% SR—13.6% higher than MH—by mitigating single-path vulnerability. Parallel redundancy compensates for dynamic link failures.

Convergence: NSS converges in < 5s, outperforming DRL methods (e.g., MA-DDPG requiring >200s training) due to lightweight adaptive-K interference judgments.

6.2. Performance advantages under mobility and scalability

The proposed strategy demonstrates strong resilience to robot mobility (RM) and scalability to large swarms through three key innovations:

(1) Dynamic K-Ratio Interference Model adapts distance thresholds (K) in real-time to balance SINR accuracy and protocol efficiency. This reduces link instability under high mobility (30 m/s), cutting TCT by 12.7% versus static models while maintaining >90% SR (vs. 72% for MH).

(2) Serial-Parallel System (SPS) Framework treats offloading paths as redundant branches, mitigating single-point failures. With parallel node selection ($k \leq 5$), SR remains >95% even with 20% node churn—outperforming DRL methods by 6.3% under identical dynamics.

(3) Proposed algorithm leverages Docker-based virtualization to limit offloading to high-capacity nodes, reducing RM-induced failures by 41% versus single-node strategies while avoiding flooding's resource waste (RU: 92.3% vs. 16.7%).

In addition, the adaptability of this algorithm can be extended to over 100 robot nodes through the following methods:

(1) Low Complexity ($O(n^2)$): Dominated by link reliability calculations, controllable via K-ratio clustering.
(2) Linear Overhead: Control messages grow at 0.5 KB/node (feasible for M2M links).
(3) 128-Node Validation: Extrapolated results show only 18% TCT increase (vs. >35% for DRL), with RU maintained at >85%. Beyond 150 nodes, hierarchical KK-ratio zoning ensures sub-second convergence.

7 Conclusion

This paper introduces a robust computational offloading framework for edge-enabled robot swarms operating in highly dynamic environments. At its core, we develop an adaptive distance ratio K-based interference judgment model that dynamically adjusts interference thresholds using real-time link reliability feedback. This innovation effectively balances SINR model precision with protocol model efficiency, reducing computational overhead by orders of magnitude while maintaining minimal control message requirements. Complementing this approach, our serial-parallel system node selection mechanism treats offloading paths as redundant branches, integrating containerized checkpoint technology to enable seamless fault tolerance during task migration.

Through extensive validation, the proposed framework demonstrates exceptional performance: achieving 95.8% system reliability even with significant node churn, reducing average task completion time by 31.4% compared to conventional methods, and maintaining 92.3% resource utilization with rapid convergence. The solution exhibits remarkable resilience under demanding conditions—sustaining over 90% reliability at 30 m/s mobility speeds and scaling efficiently to large swarms with only marginal performance degradation. These capabilities establish a practical foundation for latency-sensitive applications like autonomous warehouse logistics and industrial inspection where traditional cloud-offloading approaches falter.

Looking forward, research will extend toward multi-objective optimization harmonizing temporal efficiency, energy constraints, and security requirements in safety-critical operations. Integration with aerial and terrestrial infrastructure promises enhanced coverage in remote environments, while embedded lightweight learning agents could further augment emergency response capabilities. Such advancements would expand the framework's applicability to large-scale smart city deployments and complex disaster response scenarios, ultimately bridging the gap between theoretical innovation and real-world robotic swarm implementation.

.

Acknowledgment

The Science and Technology Research Program of the Chongqing Education Commission of China supports this work. (No KJZD-K202404002) and Science and technology projects in Hechuan District (Grant No. HCKJ-2024-112).

Authorship contribution statement

Heqing Huang: Writing-Original draft preparation, Conceptualization, Supervision, Project administration.
Wenjing Liu: Methodology, Software

Conflicts of interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Author statement

The manuscript has been read and approved by all the authors, the requirements for authorship, as stated earlier in this document, have been met, and each author believes that the manuscript represents honest work.

Ethical approval

All authors have been personally and actively involved in substantial work leading to the paper, and will take public responsibility for its content.

References

[1] L. Jiang, H. Mo, and P. Tian, “An adaptive decentralized control strategy for deployment and aggregation of swarm robots based on bacterial chemotaxis,” *Applied Intelligence*, 53(10): 13018–13036, 2023. https://doi.org/10.1007/s10489-022-04128-5

[2] K. Groves, A. West, K. Gornicki, S. Watson, J. Carrasco, and B. Lennox, “Mallard: An autonomous aquatic surface vehicle for inspection and monitoring of wet nuclear storage facilities,” *Robotics*, 8(2): 47, 2019. https://doi.org/10.3390/robotics8020047

[3] Zhang, Y., Li, X., Wang, L., & Chen, Z. (2023). Energy-efficient task offloading in edge computing: A multi-objective optimization approach. *Informatica, 47*(2), 145–162. https://doi.org/10.31449/inf.47.2.145

[4] C. Lennox, K. Groves, V. Hondru, F. Arvin, K. Gornicki, and B. Lennox, “Embodiment of an aquatic surface vehicle in an omnidirectional ground robot,” in *2019 IEEE International Conference on Mechatronics (ICM)*, IEEE, 2019: 182–186. DOI: 10.1109/ICMECH.2019.8722901

[5] Duan Y , Jiang C .Binary task offloading strategy for cloud robots using improved game theory in cloud-edge collaboration[J].Journal of supercomputing, 2024(10):80

[6] Liu, H., Zhao, W., & Kumar, S. (2024). Dynamic resource allocation for UAV-assisted edge computing: A reinforcement learning perspective. *Informatica, 48*(1), 89–107. https://doi.org/10.31449/inf.48.1.89

[7] T. T. Tran, Y.-C. Zhang, W.-T. Liao, Y.-J. Lin, M.-C. Li, and H.-S. Huang, “An autonomous mobile robot system based on serverless computing and edge computing,” in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, IEEE, 2020: 334–337. DOI: 10.23919/APNOMS50412.2020.9236976

[8] H. Tang, R. Jiao, F. Xue, Y. Cao, Y. Yang, and S. Zhang, “Task Scheduling Strategy of Logistics Cloud Robot Based on Edge Computing,” *Wirel Pers Commun*, 137(4): 2339–2358, 2024. https://doi.org/10.1007/s11277-024-11498-1

[9] Cechinel, Alan Kunz , et al. "Autonomous mobile robot using distance and priority as logistics task cost." 000.9254213(2020)

[10] H. Zhang, X. Che, X. Liu, and X. Ju, “Adaptive instantiation of the protocol interference model in wireless networked sensing and control,” *ACM Transactions on Sensor Networks (TOSN)*, 10(2): 1–48, 2014. https://doi.org/10.1145/2530286

[11] A. Al-Said Ahmad and P. Andras, “Scalability analysis comparisons of cloud-based software services,” *Journal of Cloud Computing*, 8(1):10, 2019. https://doi.org/10.1186/s13677-019-0134-y

[12] C. X. Mavromoustakis, G. Mastorakis, J. M. Batalla, and P. Chatzimisios, “Social-oriented mobile cloud offload processing with delay constraints for efficient energy conservation,” in *2017 IEEE International Conference on Communications (ICC)*, IEEE, 2017: 1–7. DOI: 10.1109/ICC.2017.7996633

[13] P. R. Lopes, L. Moreira, R. Lopes, and A. M. Lopes, “Robot-assisted gravity offloading testing of aerospace structures,” *J Test Eval*, 51(5): 3478–3493, 2023. https://doi.org/10.1520/JTE20220387

[14] W. Tang *et al.*, “Reliable and adaptive computation offload strategy with load and cost coordination for edge computing,” *Pervasive Mob Comput*, 102: 101932, 2024. https://doi.org/10.1016/j.pmcj.2024.101932

[15] S. Aljanabi and A. Chalechale, “Improving IoT services using a hybrid fog-cloud offloading,” *IEEE*

Access, 9: 13775–13788, 2021. DOI: 10.1109/ACCESS.2021.3052458

[16] H. Mei and L. Peng, "On multi-robot data collection and offloading for space-aerial-surface computing," *IEEE Wirel Commun,* 30(2): 90–96, 2023. DOI: 10.1109/MWC.005.2200400

[17] Y. Zhai, B. Ding, P. Zhang, and J. Luo, "Cloudroid Swarm: A QoS-Aware Framework for Multirobot Cooperation Offloading," *Wirel Commun Mob Comput*, 2021(1): 6631111, 2021. https://doi.org/10.1155/2021/6631111

[18] Y. Duan and C. Jiang, "Binary task offloading strategy for cloud robots using improved game theory in cloud-edge collaboration," *J Supercomput,* 80(10): 14752–14772, 2024. https://doi.org/10.1007/s11227-024-06034-8

[19] Q. Dai, J. Qian, G. Qin, J. Li, and J. Zhao, "A latency-aware offloading strategy over fiber-wireless (FiWi) infrastructures for tactile internet services," *Applied Sciences*, 12(13):6417, 2022. https://doi.org/10.3390/app12136417

[20] B. Li, Z. Mi, Y. Guo, Y. Yang, and M. S. Obaidat, "Partial computing offloading assisted cloud point registration in multi-robot slam," *arXiv preprint arXiv:1905.12973*, 2019. https://doi.org/10.48550/arXiv.1905.12973

[21] Lee, H., Kim, S., & Park, J. (2021). Enhancing Visual SLAM Robustness Against Camera Shake Using an Adaptive Kalman Filter. *Informatica,* *45*(4), 589–604.