An online task offloading method based on improved starfish optimization and blockchain

Lujie Tao[1], Zhaoyu Su[2,*,†], Yujue Wang[3]
{szyguet@guet.edu.cn}

[1] School of Information and Communication, Guilin University of Electronic Technology, Guilin, China
[2] Guangxi Engineering Research Center of Industrial Internet Security and Blockchain, Guilin University of Electronic Technology, Guilin, China
[3] Hangzhou Innovation Institute of Beihang University, Hangzhou, China

Abstract. Vehicular Edge Computing (VEC) is a key enabler of low-latency intelligent transportation applications. However, designing effective task offloading strategies in VEC faces challenges such as resource variability, dynamic network topology, and data isolation among distributed nodes. To address these issues, this paper proposes an Online Intelligent Blockchain-Enhanced Task Offloading Optimization System (OIBTO). The system employs a lightweight Proof-of-Authority (PoA) consensus within a two-tier architecture consisting of a vehicle layer and an edge layer. Edge nodes act as validators, jointly maintaining a distributed ledger to enable secure and efficient sharing of task dependencies and offloading decisions, effectively eliminating data silos. Additionally, we propose an Improved Starfish Optimization Algorithm (ISFOA) that utilizes Tent chaotic mapping and genetic mutation to optimize offloading decisions and task partitioning ratios, aiming to minimize a priority-weighted combination of latency and energy consumption. Theoretical analysis confirms the convergence of ISFOA. Simulation results show that the proposed framework improves average task latency and energy consumption by approximately 10% compared to state-of-the-art algorithms, demonstrating its effectiveness, security, and superiority in dynamic vehicular environments.

Keywords: Internet of Vehicles, edge computing, task offloading, task dependency, starfish optimization algorithm

1 Introduction

The Internet of Vehicles (IoV) facilitates real-time data exchange through V2V, V2I, and V2C communications, enabling applications such as autonomous driving and traffic optimization [1]. However, the growing volume of sensor data and real-time tasks exposes the limitations of cloud computing, whose centralized architecture introduces high latency and insufficient reliability. Edge computing overcomes these issues by deploying computational resources closer to data sources, reducing transmission delays and enhancing real-time processing capabilities [2]. This approach

IIKI 2025, December 18-21, Chengdu, People's Republic of China

DOI 10.4108/eai.18-12-2025.2365286

accelerates critical functions including environmental perception and path planning, thereby improving overall system performance. Task offloading plays a vital role in IoV edge computing by dynamically allocating computation tasks to local or edge resources to optimize latency, energy consumption, and resource utilization [3]. Yet, existing methods struggle with resource heterogeneity, task dependencies, and dynamic network conditions [4, 5, 6]. Most approaches lack effective dependency management, fail to prioritize tasks appropriately, and rely on static heuristics that cannot adapt to changing environments. Moreover, they often focus on isolated objectives without achieving multi-objective trade-offs. To address these challenges, this paper proposes the OIBTO system, which provides an integrated framework and practical mechanisms for dynamic resource management and real-time optimization. The main contributions are as follows:

- To address resource heterogeneity and dynamic allocation, this paper proposes the OIBTO system, integrating vehicular terminals, edge nodes, and blockchain. Its distributed architecture reduces end-to-end latency and overcomes limitations of static partitioning. Under heterogeneous and high-load conditions, the system balances latency and energy consumption through multi-objective optimization.
- To ensure reliable and transparent management of task dependencies, a lightweight Proof of Authority (PoA) consensus mechanism is introduced. Edge nodes serve as validators to maintain a distributed ledger recording task dependency graphs and offloading decisions, ensuring data consistency across nodes. The system supports asynchronous writes and offline operation, allowing graceful degradation during network instability and batch synchronization upon recovery. This enhances usability and robustness in IoV environments, mitigating single-point-of-failure risks and security issues.
- For efficient subtask partitioning, an enhanced Starfish Optimization Algorithm is proposed. It employs a fitness function to evaluate latency and energy tradeoffs, incorporating random factors and a learning rate to iteratively improve solutions. The algorithm reduces computational complexity and adapts well to task dependencies and resource constraints. Within OIBTO, it improves processing of high-priority tasks and demonstrates strong convergence and robustness, especially in dynamic subtask proportion adjustment.

2 Related work

Task offloading in mobile edge computing has been extensively studied through three primary approaches: optimization algorithms, game-theoretic mechanisms, and blockchain-based methods. Heuristic and metaheuristic strategies offer flexibility but often lack adaptability to dynamic network conditions, as seen in multi-cycle heuristics [7] and artificial fish swarm methods with location prediction [8], which struggle with sudden congestion or mobility-induced topology changes. Game-theoretic approaches [9, 10] enable fast convergence but oversimplify task and resource models, while Lyapunov optimization [11] proves parameter-sensitive and computationally intensive. These methods generally overlook critical aspects such as task dependencies, environmental dynamics, and multi-objective coordination. Blockchain-based solutions enhance security and integrity but

introduce performance limitations. While decentralized verification [12] and smart contracts [13] improve reliability and privacy, they suffer from high overhead and synchronization delays. Consortium blockchain mechanisms [14] and digital twin integrations [15] address malicious behavior and mobility but face trust assumptions and synchronization challenges. Methods combining blockchain with DDQN [16] or reputation systems [17] reduce latency and energy consumption yet remain constrained by single-hop communication or vulnerable reputation updates. Although blockchain strengthens security, these approaches generally incur substantial computational and communication costs while adapting poorly to IoV instability. To bridge these gaps, we propose ISFOA, which integrates explicit task dependency and priority modeling within an enhanced metaheuristic framework, enabling efficient and robust multi-objective optimization in dynamic vehicular environments [18, 19, 20].

3 System model and problem formulation

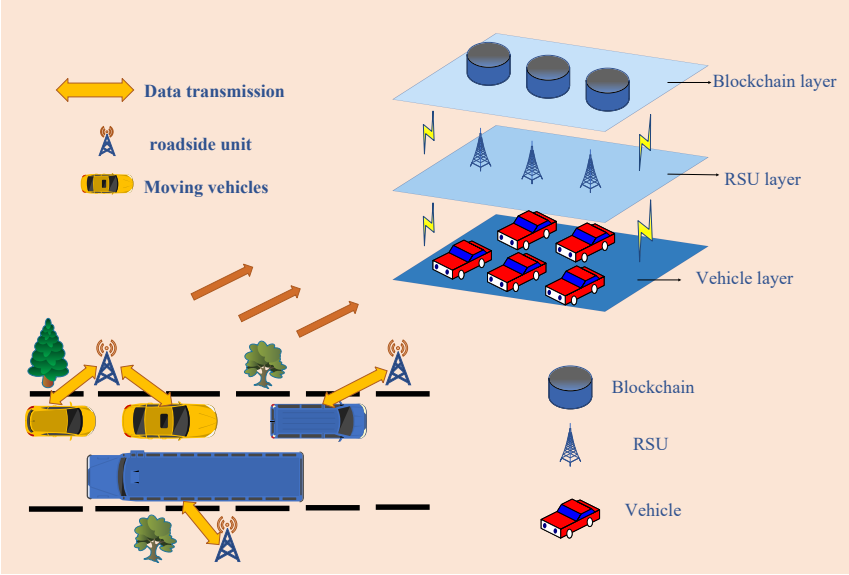

Fig. 1. System Architecture

The OIBTO system comprises four core components: vehicle model, edge node model, task model, and blockchain mechanism. The task model introduces a priority calculation method that considers inherent task priority, number of dependent subsequent tasks, and longest dependency chain length. This formulation establishes optimization objectives and constraints to tackle resource heterogeneity, task dependencies, and real-time requirements, while overcoming limitations like ignored task priorities, static partitioning, and single-objective optimization. As shown in Fig. 1,

vehicles offload tasks to fixed RSUs acting as edge nodes during movement, which then return processed results to the vehicles.

3.1 Vehicle model

In the IoV, vehicles serve as task generators, represented by the set $\mathscr{I} = \{I_1, I_2, \ldots, I_N\}$, where N denotes the total number of vehicles. Each vehicle I_i possesses limited computational resources characterized by total computing capacity c_i^{tot} and transmission bandwidth b_i^{tot}. Resource heterogeneity emerges from varying computational and communication capabilities across different vehicle models. Vehicle mobility follows the Gauss-Markov model, yielding Rayleigh-distributed instantaneous channel gain $h_{i,j}(t)$ between vehicle I_i and edge node F_j. The time-varying effective transmission bandwidth is expressed as:

$$b_{i,j}^{\text{tot}}(t) = B_{\text{total}} \cdot \log_2 \left(1 + \frac{P_t \cdot |h_{i,j}(t)|^2}{N_0}\right) \tag{1}$$

Vehicles can process tasks locally or offload them to edge nodes based on computational delay, transmission delay, and energy consumption considerations. The available computational and communication resources vary dynamically with task load:

$$c_i^{\text{avl}}(t) = c_i^{\text{tot}} - c_i^{\text{local}}(t) \tag{2}$$

$$b_i^{\text{avl}}(t) = b_i^{\text{tot}} - b_i^{\text{local}}(t) \tag{3}$$

Offloading decisions must satisfy the constraints $x_{k,i,j} \cdot w_k \leq c_i^{\text{avl}}(t)$ and $x_{k,i,j} \cdot d_k \leq b_i^{\text{avl}}(t)$, where $x_{k,i,j}$ is a binary decision variable, w_k represents computational requirement, and d_k denotes data size. The computational delay for locally processed task T_k is:

$$t_{k,i}^{\text{comp}} = \frac{w_k}{c_i^{\text{avl}}(t)} \tag{4}$$

Using TDMA mechanism, the transmission delay for offloading is:

$$t_{i,j}^{\text{tran}} = \frac{d_k}{b_{i,j}^{\text{tot}}(t)} + \tau_{i,j}^{\text{prop}} \tag{5}$$

The local energy consumption model is given by:

$$e_{k,i}^{\text{comp}} = \kappa w_k (c_i^{\text{avl}}(t))^2 \tag{6}$$

Task generation follows a Poisson process with average arrival rate λ_s, reflecting dynamic task load.

3.2 Edge node model

The OIBTO system employs edge nodes (RSUs or base stations) as primary processing units, denoted by $\mathscr{F} = \{F_1, F_2, \ldots, F_M\}$, where M represents the number of edge nodes. Each node F_j possesses substantial computational capacity c_j^{edge} and transmission bandwidth b_j^{edge}, significantly exceeding vehicle capabilities ($c_j^{\text{edge}} \gg c_i^{\text{tot}}$, $b_j^{\text{edge}} \gg b_i^{\text{tot}}$). Each edge node maintains a multi-priority task queue with non-preemptive priority scheduling, consistently executing the highest-priority task first to minimize processing delays for critical tasks. The computational delay and energy consumption for processing task T_k at edge node F_j are:

$$t_{k,j}^{\text{comp}} = \frac{w_k}{c_j^{\text{edge}}}, \quad e_{k,j}^{\text{comp}} = \kappa w_k (c_j^{\text{edge}})^2 \tag{7}$$

Transmission energy consumption is modeled as:

$$e_{i,j}^{\text{tran}} = \rho d_k \tag{8}$$

where ρ denotes the transmission energy coefficient. Edge nodes employ dynamic resource allocation through task queue management and collaborate via a blockchain network to share task dependency and offloading information.

3.3 Task model

Tasks in the OIBTO system are generated by vehicle terminals and represented as the set $\mathscr{T} = \{T_1, T_2, \ldots, T_K\}$. Each task T_k is defined as a quadruple $T_k = (d_k, w_k, p_k^{\text{base}}, D_k)$, where d_k denotes the data size of the task, indicating the amount of data required for transmission. w_k represents the computational workload, specifying the computational resources needed to execute the task. $p_k^{\text{base}} \in [1, P_{\max}]$ is the initial priority assigned based on the task type. D_k denotes the deadline for task completion.

To comprehensively account for task importance and dependency structure, the composite priority of a task is calculated as a weighted combination of three factors: the task's inherent priority, the number of directly dependent subsequent tasks, and the length of the longest dependency chain:

$$p_k = w_p p_k^{\text{base}} + w_d |\mathscr{R}_k| + w_l L_k, \tag{9}$$

In the OIBTO system, the composite priority of a task T_k is calculated by considering the task's inherent attributes and its dependency structure. Specifically, $|\mathscr{R}_k|$ denotes the size of the set $\mathscr{R}_k = \{T_n | T_k \prec T_n\}$, which includes all tasks directly dependent on task T_k. The term L_k represents the length of the longest dependency chain involving task T_k, computed using Depth-First Search. The weights w_p, w_d, and w_l are coefficients , used to balance the influence of the task's inherent priority, the number of directly dependent tasks, and the length of the dependency chain on its urgency. The dependency set $D_k = \{T_m | T_m \prec T_k\}$ represents the tasks that task T_k must wait for, where $T_m \prec T_k$ indicates that T_k depends on the completion of T_m. A task T_k can be divided into S subtasks $\{T_{k,1}, T_{k,2}, \ldots, T_{k,S}\}$. The task division occurs along two dimensions:

1. Computational Load Division: The total computational load w_k of a task is divided such that $\sum_{s=1}^{S} w_{k,s} = w_k$.

2. Dependency Relationship Division: Subtasks inherit all external dependencies of the parent task. Furthermore, internal dependencies can be introduced among subtasks, forming a new subtask DAG. The ISFOA algorithm simultaneously optimizes both the division ratio of subtasks and their scheduling order.

Task generation follows a Poisson process with an average arrival rate of λ, reflecting the dynamic task load.

3.4 Blockchain mechanism

In the IoV, each vehicle I_i is equipped with constrained computational resources, characterized by its total computing capacity c_i^{tot} and total transmission bandwidth b_i^{tot}.

$$\text{Tx}_k = (\text{ID}_k, \text{Node}_k, \text{Time}_k, p_k, D_k, \text{Status}_k) \tag{10}$$

In the OIBTO system, each blockchain transaction records essential task-related information to ensure reliable and transparent task management, with the transaction format structured as a tuple that includes the unique identifier of task T_k denoted as ID_k, the node assigned to process the task represented by Node_k, the timestamp indicating when the transaction is recorded as Time_k, the composite priority of the task p_k calculated based on its inherent priority, the number of dependent tasks, and the length of the longest dependency chain, the dependency set D_k specifying the tasks that T_k depends on, and the current status of the task indicated by Status_k.

In contrast to traditional blockchains, OIBTO employs a lightweight Proof of Authority (PoA) consensus where authorized edge nodes act as validators, drastically reducing computational and communication overhead. This PoA mechanism utilizes an efficient message protocol and high-speed local networking to maintain a consistently low consensus latency $\tau_{\text{consensus}}$. The system features a layered architecture—with a vehicle layer for task submission and an edge layer for validation—to enhance security and scalability. To handle network instability, an asynchronous batch writing mechanism and intelligent offline mode are implemented. This ensures real-time recording during stable connectivity and local storage with batch synchronization upon recovery, guaranteeing eventual data consistency.

3.5 Problem modeling

The objective of task offloading is to optimize system performance while satisfying real-time, dependency, and priority constraints. The offloading decision variable for a task T_k is defined as:

$$x_{k,i,j} = \begin{cases} 1, & \text{if task } T_k \text{ is assigned to node } (i,j) \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

where $(i, j) \in \mathscr{I} \cup \mathscr{F}$. The total task delay includes both transmission delay and computational delay:

$$t_k = \sum_{i \in \mathscr{I}} x_{k,i,i} t_{k,i}^{\text{comp}} + \sum_{i \in \mathscr{I}, j \in \mathscr{F}} x_{k,i,j} (t_{i,j}^{\text{tran}} + t_{k,j}^{\text{comp}}). \tag{12}$$

The energy consumption model considers both computational energy consumption and transmission energy consumption:

$$e_k = \sum_{i \in \mathscr{I}} x_{k,i,i} e_{k,i}^{\text{comp}} + \sum_{i \in \mathscr{I}, j \in \mathscr{F}} x_{k,i,j} (e_{i,j}^{\text{tran}} + e_{k,j}^{\text{comp}}). \tag{13}$$

To balance delay and energy consumption while prioritizing the performance requirements of high-priority tasks and ensuring task dependency relationships and real-time requirements, the optimization objective adopts a priority-weighted multi-objective function:

$$\min_{\mathbf{x}} \quad \sum_{k \in T} p_k (w_t t_k + w_e e_k) \tag{14a}$$

$$\text{s.t.} \quad \sum_{i \in T} x_{k,i,j} + \sum_{\substack{i \in T \\ j \in F}} x_{k,i,j} = 1, \qquad \forall k \in T, \tag{14b}$$

$$s_k \geq \max_{T_m \in D_k} (s_m + t_m), \qquad \forall k \in T, \tag{14c}$$

$$t_k \leq t_k^{\max}, \qquad \forall k \in T, \tag{14d}$$

$$\sum_{k \in T} x_{k,i,j} w_k \leq c_j^{edge}, \qquad \forall j \in F, \tag{14e}$$

$$\sum_{k \in T} x_{k,i,j} d_k \leq b_j^{edge}, \qquad \forall i \in I,\ j \in F, \tag{14f}$$

$$x_{k,i,j} \in \{0, 1\}, \qquad \forall k \in T,\ (i, j) \in L \cup F. \tag{14g}$$

4 Proposed algorithm

This section proposes an algorithmic framework for optimizing task offloading in IoV environments, integrating the Improved Starfish Optimization Algorithm (ISFOA) with a blockchain-supported priority scheduling algorithm to achieve efficient task allocation, resource management, and data security assurance. ISFOA minimizes latency and energy consumption by optimizing task offloading decisions and partitioning ratios; the blockchain scheduling algorithm ensures scheduling transparency, traceability, and priority management via a private blockchain network. The schematic flowchart is shown in Fig. 2.

4.1 Improved starfish optimization algorithm

ISFOA enhances the Starfish Optimization Algorithm[21], a nature-inspired metaheuristic that mimics starfish behaviors like movement, foraging, and regeneration, to optimize task offloading in IoV scenarios. The original algorithm suffers from limited initial population diversity, susceptibility to local optima, inadequate handling of task priorities p_k, and poor adaptability to edge node

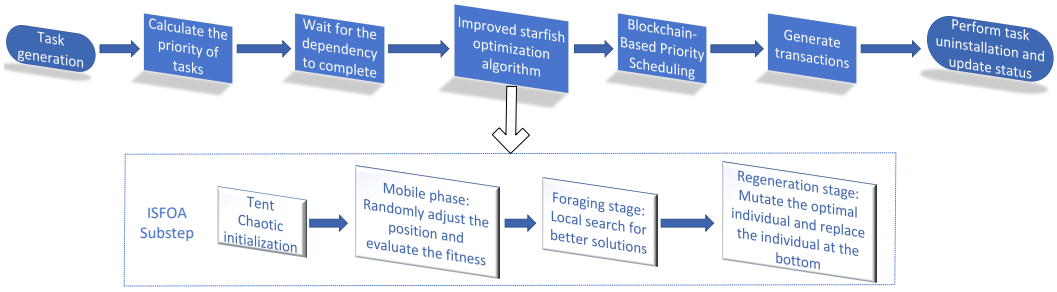

Fig. 2. Schematic flowchart

resource heterogeneity and dynamic IoV network conditions, leading to suboptimal task scheduling and resource allocation. To overcome these limitations, ISFOA introduces Tent chaotic mapping for improved initial population diversity, genetic variation mechanisms to escape local optima, and priority-weighted fitness functions to align with IoV requirements. It optimizes task offloading decisions $x_{k,i,j}$ and partitioning ratios α_k through iterative search, simulating starfish foraging. The objective function balances load, minimizes task latency t_k, reduces energy consumption e_k, and prioritizes tasks based on p_k, ensuring efficient and practical resource allocation in dynamic IoV environments.

$$\min \sum_{k \in T} p_k (w_t t_k + w_e e_k), \tag{15}$$

where t_k denotes the total latency of the task, including transmission latency $t^{\text{tran}}i, j = \frac{d_k}{b^{\text{iot}}{}_{i,j}} + \tau^{\text{prop}}i, j$ and computation latency $t^{\text{comp}}k, j = \frac{w_k}{c^{\text{fog}}{}_j}$ or $t^{\text{comp}}k, i = \frac{w_k}{c^{\text{iot}}{}_i}$; e_k represents the total energy consumption, which consists of computation energy $e^{\text{comp}}k, j = \kappa w_k (c^{\text{fog}}\, j)^2$ or $e^{\text{comp}}k, i = \kappa w_k (c^{\text{iot}}i)^2$, and transmission energy $e^{\text{tran}}i, j = \rho d_k$. The regeneration phase incorporates a mutation mechanism from genetic algorithms, where new individuals are generated by mutating the best individual to escape local optima.

4.2 Blockchain-supported priority scheduling algorithm

The blockchain-supported priority scheduling algorithm utilizes a private blockchain network maintained by edge nodes $F_j \in F$ to ensure transparent, traceable, and secure task scheduling, working collaboratively with ISFOA to optimize task offloading and resource allocation. Task priority p_k is calculated using Equation (10), which integrates base priority p_k^{base}, the number of dependent tasks $|R_k|$, and the longest dependency chain L_k. Each task T_k is formed into a transaction Tx_k containing its ID, assigned node, timestamp, dependency set, and status, which is then verified by edge nodes via a consensus protocol. Smart contracts automatically schedule tasks based on priority and resource availability C_j^{fog}, $B_{i,j}^{\text{fog}}$. The process begins by initializing the blockchain and deploying smart contracts. When a vehicle I_i issues a task request, it creates a signed transaction Tx_k, which is broadcast to edge nodes for verification of priority, dependencies, resources, and signature. If consensus is

reached, the transaction is recorded on the blockchain and the task is assigned by the smart contract; otherwise, it is returned to ISFOA for re-optimization. In the collaborative workflow, ISFOA first optimizes the offloading decision $x_{k,i,j}$ and partitioning ratio α_k, and the blockchain scheduler then validates the resulting allocation scheme. Successful consensus leads to smart contract-based assignment, while failure triggers ISFOA re-optimization. This integration enables OIBTO to achieve efficient, fair, and secure task offloading—with ISFOA optimizing performance in latency and energy, and the blockchain ensuring prioritized and trustworthy execution.

5 Performance analysis

Table 1: Experimental Parameter Settings

Parameter	Symbol	Value
Number of vehicles	N	20
Number of edge nodes	M	5
Training iteration	H	1000
Time slot base	T_{base}	100
Maximum delay	$\Delta_{\max}$	10
Total time slots	T	110
Time slot duration	τ	0.1 s
Task arrival probability	λ	0.25
Maximum bits arrival	$A_{\max}$	5 Mbits
Minimum bits arrival	$A_{\min}$	2 Mbits
Vehicle computing capability	c_i^{iot}	3 GHz * τ
Edge computing capability	c_j^{edge}	50 GHz * τ
Transmission bandwidth	$b_{i,j}^{\text{iot}}$	16 Mbps * τ

The experimental parameters primarily include vehicle and edge node configurations, task generation parameters, algorithm hyperparameters, and blockchain settings. The specific parameters are shown in Table 1.

Experiments were conducted on a workstation with an Intel® Core™ i5-12600KF CPU, NVIDIA GeForce RTX 4060 Ti GPU, 32GB DDR4 RAM, and a 2TB NVMe SSD, under Windows 10. The algorithm was implemented in Python 3.6.5 using TensorFlow 1.4.0 for training and inference. A local blockchain testnet was deployed with Ganache 2.5.4, and smart contracts were developed in Remix IDE. Numerical computations utilized NumPy and SciPy, and visualizations were generated with Matplotlib and Seaborn. The experiments compare the average energy consumption and latency of ISFOA against seven baseline algorithms, including SFOA as the original algorithm, Random , Local , Edge , HP [22], PDC [22], and Cluster [23].

As shown in Figures. 3 and 4, the performance of ISFOA is compared with PDC, HP, Cluster, and SFOA under varying vehicle counts in dynamic IoV environments. ISFOA achieves the lowest

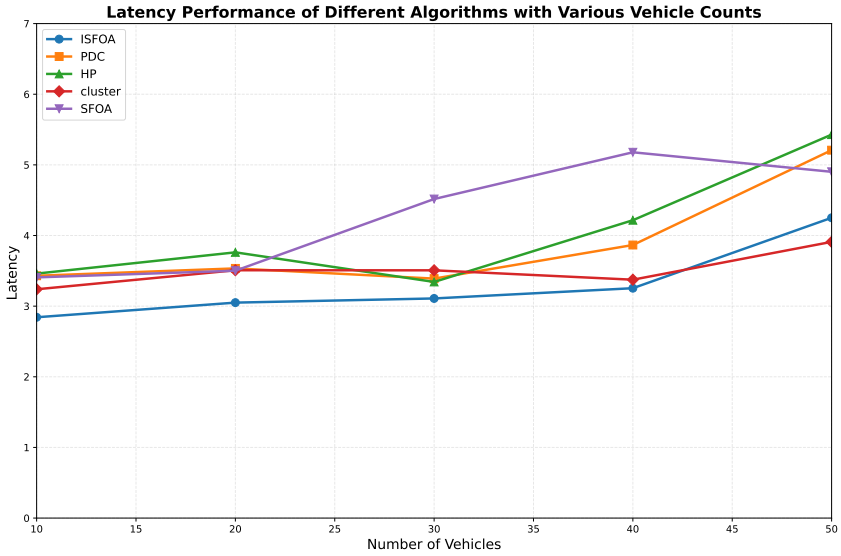

Fig. 3. Performance Comparison of Different Algorithms with Various Vehicle Counts

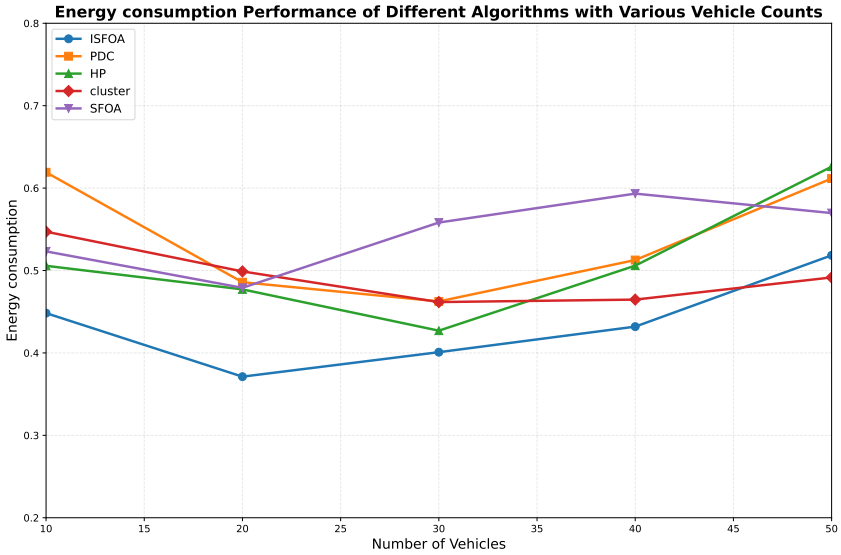

Fig. 4. Performance Comparison of Different Algorithms with Various Vehicle Counts

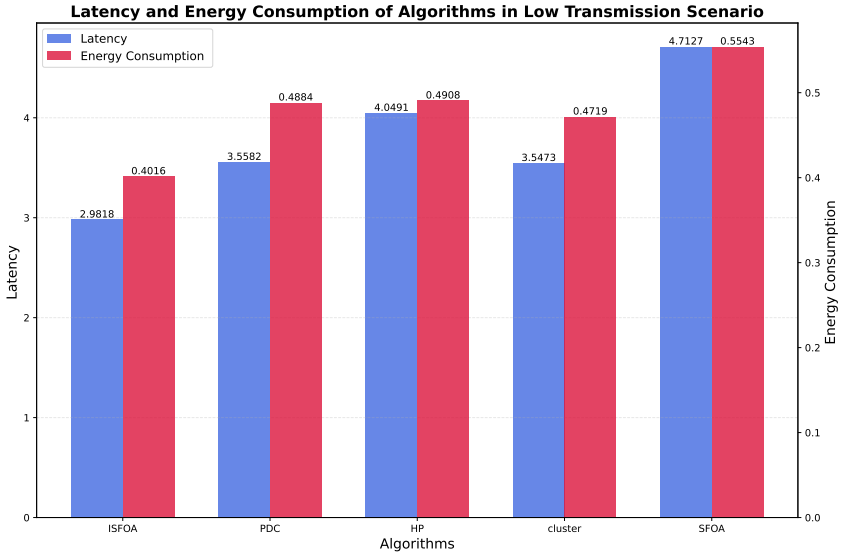

Fig. 5. Performance Comparison of Different Algorithms with Various Vehicle Counts

latency and energy consumption, with latency increasing only marginally and energy decreasing from 0.5 to 0.35 as vehicle density grows, while other algorithms exhibit significantly higher or more volatile values. This demonstrates ISFOA's superior scalability and efficiency, attributed to Tent chaotic initialization, genetic mutation, and priority-weighted optimization. Under constrained transmission conditions as shown in Fig. 5, ISFOA again maintains the lowest latency and energy use, confirming its robustness in resource-limited scenarios.

6 Conclusion

This paper proposed OIBTO, an online task offloading system that integrates an Improved Starfish Optimization Algorithm (ISFOA) with a lightweight blockchain to address key challenges in vehicular edge computing, including resource heterogeneity, network dynamics, and task dependencies. The system employs a comprehensive formal model for vehicles, edge nodes, and multi-priority tasks to reflect real-world IoV complexities. By introducing Tent chaotic mapping and genetic mutation, ISFOA significantly enhances the exploration and exploitation capabilities of the original SFOA, utilizing a priority-weighted objective to optimize the trade-off between task latency and energy consumption. Furthermore, the PoA-based blockchain mechanism provides a secure and transparent environment for managing task dependencies and offloading decisions with minimal consensus overhead.Extensive simulation results confirm that OIBTO outperforms existing state-of-the-art methods, achieving an average improvement of approximately 10

Acknowledgment

This work was supported in part by the Guangxi Key Research and Development Project (Guike AB25069120), in part by the Basic Ability Enhancement Program for Young and Middle-aged Teachers of Guangxi (2025KY0249).

Declaration on Generative AI

During the preparation of this work, the author used deepseek and Grok in order to: Grammar and spelling check. After using these tools, the author reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] Dong S, Tang J, Abbas K, Hou R, Kamruzzaman J, Rutkowski L, et al. Task offloading strategies for mobile edge computing: A survey. Computer Networks. 2024;254:110791.

[2] Mahapatra A, Mishra K, Pradhan R, Majhi SK. Next generation task offloading techniques in evolving computing paradigms: Comparative analysis, current challenges, and future research perspectives. Archives of Computational Methods in Engineering. 2024;31(3):1405-74.

[3] Xie M, Su X, Sun H, Zhang G. Online task offloading algorithm based on multi-objective optimization caching strategy. Computer networks. 2024;245:110400.

[4] Cao Z, Deng X, Yue S, Jiang P, Ren J, Gui J. Dependent Task Offloading in Edge Computing Using GNN and Deep Reinforcement Learning. IEEE Internet of Things Journal. 2024;11(12):21632-46.

[5] Fofana N, Letaifa AB, Rachedi A. Intelligent Task Offloading in Vehicular Networks: A Deep Reinforcement Learning Perspective. IEEE Transactions on Vehicular Technology. 2025;74(1):201-16.

[6] Hao H, Xu C, Zhang W, Yang S, Muntean GM. Joint Task Offloading, Resource Allocation, and Trajectory Design for Multi-UAV Cooperative Edge Computing With Task Priority. IEEE Transactions on Mobile Computing. 2024;23(9):8649-63.

[7] Zhang R, Wu L, Cao S, Xiong NN, Li J, Wu D, et al. MPTO-MT: A multi-period vehicular task offloading method in 5G HetNets. Journal of Systems Architecture. 2022;131:102712.

[8] Zhang Z, Chen Z, Shen Y, Dong X, Xi N. A dynamic task offloading scheme based on location forecasting for mobile intelligent vehicles. IEEE Transactions on Vehicular Technology. 2024;73(6):7532-46.

[9] Li S, Li W, Liu H, Sun W. A two-stage service-oriented task offloading framework with edge-cloud collaboration: a game theory approach. Journal of Systems Science and Systems Engineering. 2024;33(5):521-51.

[10] Lu J, Li Q, Guo B, Li J, Shen Y, Li G, et al. A multi-task oriented framework for mobile computation offloading. IEEE Transactions on Cloud Computing. 2019;10(1):187-201.

[11] Chen Y, Zhao F, Lu Y, Chen X. Dynamic task offloading for mobile edge computing with hybrid energy supply. Tsinghua Science and Technology. 2022;28(3):421-32.

[12] Du J, Yu Z, Sun A, Jiang J, Zhao H, Zhang N, et al. Secure Task Offloading in Blockchain-Enabled MEC Networks With Improved PBFT Consensus. IEEE Transactions on Cognitive Communications and Networking. 2025;11(2):1225-43.

[13] Zuo L, Li Y, Xia S, Pan J. Blockchain-Based Collaborative Task Offloading Algorithm in Heterogeneous Edge Computing Networks. IEEE Transactions on Cognitive Communications and Networking. 2026;12:784-95.

[14] Xu Y, Li H, Zhang C, Tang Z, Zhong X, Ren J, et al. Blockchain-Enabled Multiple Sensitive Task-Offloading Mechanism for MEC Applications. IEEE Transactions on Mobile Computing. 2025;24(4):3241-55.

[15] Xu C, Zhang P, Xia X, Kong L, Zeng P, Yu H. Digital-Twin-Assisted Intelligent Secure Task Offloading and Caching in Blockchain-Based Vehicular Edge Computing Networks. IEEE Internet of Things Journal. 2025;12(4):4128-43.

[16] Moghaddasi K, Rajabi S, Gharehchopogh FS. Multi-objective secure task offloading strategy for blockchain-enabled IoV-MEC systems: a double deep Q-network approach. IEEe Access. 2024;12:3437-63.

[17] Jing J, Yang Y, Zhou X, Huang J, Qi L, Chen Y. Multi-UAV Cooperative Task Offloading in Blockchain-Enabled MEC for Consumer Electronics. IEEE Transactions on Consumer Electronics. 2025;71(1):2271-84.

[18] Liang G, Li C, Zhao F, Zhang C, Zhu L. Task Offloading for Vehicular Edge Computing Based on Improved Hotstuff Under Parking Assistance. IEEE Transactions on Vehicular Technology. 2025;74(9):14550-63.

[19] Chen Q, Li C, Chen M, Wu M, Zhang G. Digital twin assisted multi-task offloading for vehicular edge computing under SAGIN with blockchain. IET Communications. 2025;19(1):e70002.

[20] Li C, Chen Q, Chen M, Su Z, Ding Y, Lan D, et al. Blockchain enabled task offloading based on edge cooperation in the digital twin vehicular edge network. Journal of Cloud Computing. 2023;12(1):120.

[21] Zhong C, Li G, Meng Z, Li H, Yildiz AR, Mirjalili S. Starfish optimization algorithm (SFOA): a bio-inspired metaheuristic algorithm for global optimization compared with 100 optimizers. Neural Computing and Applications. 2025;37(5):3641-83.

[22] Chen H, Todd TD, Zhao D, Karakostas G. Task class partitioning for mobile computation offloading. IEEE Internet of Things Journal. 2023;11(2):2534-49.

[23] Moon S, Lim Y. Task migration with partitioning for load balancing in collaborative edge computing. Applied Sciences. 2022;12(3):1168.