A Simulated Annealing-Based Task Offloading and Delay Optimization Method for Vehicular Fog Computing

Lin Chai, Jun Wang, Yumei Yang, and Wu Wang
School of Mathematics and Computer Science, Yunnan Minzu University, Kunming 650504, China

With the rapid increase in real-time computational demands from in-vehicle applications, traditional cloud computing is often unable to meet the millisecond-level response requirements of the Internet of Vehicles due to transmission delays. Vehicular fog computing, which integrates edge infrastructures and idle resources from nearby vehicles, presents an effective paradigm for enabling low-latency services. However, optimizing task offloading in such dynamic environments remains challenging due to the high mobility of vehicles, time-varying wireless channel states, and the highly heterogeneous computational capabilities of available fog nodes. This paper addresses the delay minimization problem in task offloading for Vehicular Fog Computing (VFC) by constructing a fine-grained system model. Tasks are modeled as divisible and heterogeneous subtasks, with factors such as distance, channel state, and computational resources comprehensively considered. The problem is formulated as a mixed-integer nonlinear programming problem. To solve it, an intelligent offloading algorithm based on Simulated Annealing (SA) is proposed. The algorithm regulates the search process using temperature parameters—conducting extensive exploration of the solution space at high temperatures to avoid local optima, and gradually refining the search as temperature decreases to converge to a near-globally optimal task allocation strategy. Simulation experiments are conducted under various task scale scenarios, and the results demonstrate that the SA-based algorithm consistently achieves lower latency across different configurations. Compared with greedy and random algorithms, the proposed method significantly reduces delay and offers greater improvements over local execution. In the scenario with a task data size of 100 Mb and a computational workload of 250 TFLOPS, the latency of the SA algorithm is only 59.0% of that of the greedy algorithm and 44.8% of that of the random algorithm. This study validates the effectiveness of the SA algorithm for task offloading optimization in VFC environments, providing a low-latency solution for real-time Internet of Vehicles applications and offering meaningful insights for the development of intelligent transportation systems.

***Index Terms*—Internet of Vehicles, Vehicular Fog Computing, Task Offloading, Delay Minimization, Simulated Annealing Algorithm.**

I. Introduction

In recent years, with the deep integration of the Internet of Things (IoT) and Intelligent Transportation Systems (ITS), the Internet of Vehicles (IoV) has developed into a critical infrastructure for enabling smart cities and autonomous driving [1]. Modern vehicles are no longer merely means of transportation but have evolved into intelligent terminals that integrate environmental perception, data fusion, and real-time decision-making, giving rise to a series of computationally intensive and latency-sensitive applications, such as real-time high-definition map updates, collaborative collision warning, and autonomous driving path planning. Such applications typically impose stringent requirements on data processing capability and response time, where millisecond-level latency variations can directly affect driving safety and user experience [2]. Although traditional cloud computing architectures provide abundant computational and storage resources, their centralized processing model necessitates the long-distance transmission of data to remote data centers, inevitably introducing transmission delays and network congestion risks. As a result, they struggle to meet the pressing demands of IoV applications for ultra-low latency and localized processing.

Against this backdrop, edge computing and fog computing have emerged as enabling paradigms, with the core idea of extending computational, storage, and networking capabilities to the network edge, in proximity to data sources and end users. VFC, as a concrete instantiation of this concept within the Internet of Vehicles domain, establishes a dynamic, distributed, and hierarchical collaborative computing network by integrating static edge infrastructures—such as Roadside Units (RSUs) and cellular base stations—with the idle computational resources of vehicles [3]. In this network, vehicles assume dual roles as both service consumers and providers: they can offload complex tasks to nearby resources, and when idle, contribute their computational capacity to support other vehicles, thereby enabling efficient utilization of computing resources. Task offloading, as a core technology in VFC, is designed to address the fundamental tension between the limited onboard computational capacity of vehicles and their continuously growing computational demands [4]. Essentially, this constitutes a dynamic resource allocation and scheduling problem, which involves the intelligent decomposition of computationally intensive tasks generated by vehicles—such as image recognition, path planning, and collaborative decision-making—and their assignment to the most suitable edge nodes or neighboring vehicles for execution, subject to multiple constraints including task completion time, energy consumption, and reliability.

In the context of the IoV, safety-critical applications such as autonomous driving, collision warning, and real-time navigation impose stringent requirements on response latency, where millisecond-level latency variations can directly impact driving safety [5]. By optimizing task processing latency, real-time transmission and processing of emergency data between vehicles and roadside units, neighboring vehicles, or fog

Manuscript received January 15, 2026; revised March 31, 2026. Corresponding author: Wu Wang (email: wangwu@ymu.edu.cn).

nodes can be ensured, thereby effectively avoiding decision-making errors caused by communication delays. Furthermore, latency optimization contributes to enhancing the Quality of Experience (QoE) for value-added services such as in-vehicle infotainment and high-definition map updates [6], while providing theoretical support for the dynamic scheduling and load balancing of network resources. This holds significant engineering guidance for promoting the large-scale deployment of Intelligent Transportation Systems. Therefore, optimizing latency through efficient task offloading strategies becomes particularly critical.

Currently, research on task offloading in VFC has achieved notable progress, primarily focusing on offloading decision-making, resource allocation, and path planning. Toopchinezhad et al. employed a deep reinforcement learning (DRL) method based on Proximal Policy Optimization (PPO) to address the task offloading decision problem in VFC environments [7]. Cho et al. proposed a fully decoupled learning algorithm based on adversarial multi-armed bandits to address the problem of distributed task offloading decision-making for multiple task vehicles in unknown dynamic environments in vehicular fog computing [8]. Reference [9][10][11] provided a comprehensive review, classification, and synthesis of resource allocation, task offloading, and security issues in VFC. Gao et al. proposed a Transformer-based Policy-Decoupled Multi-Agent Actor–Critic method (TPDMAAC) to address dynamic task offloading and resource allocation in heterogeneous VFC environments [12]. Wu et al. introduced a joint task assignment and resource allocation optimization method based on mobility prediction information to minimize task execution latency in dynamic vehicular networks [13]. Bi et al. developed a hierarchical learning whale optimization algorithm with adaptive Lévy flight (LWH) to optimize task offloading decisions from vehicles to fog servers or idle vehicles, aiming to minimize overall task latency subject to energy consumption constraints [14]. Maleki et al. presented a machine learning prediction method based on Matrix Completion, combined with sampling dynamic programming (S-OAMC) and a greedy algorithm (G-OAMC), to minimize application turnaround time—including offloading, migration, and execution time [15]. Zeng et al. in [16] proposed a vehicle trajectory prediction method based on a Bidirectional Long Short-Term Memory network (Bi-LSTM), integrated with candidate server backup and a look-ahead scheduling strategy (MEPT). By predicting the future positions of vehicles and proactively staging required data on multiple candidate edge servers, this approach minimizes task execution preparation time and thereby reduces overall offloading latency. Zhang et al. proposed a delay-optimized resource allocation scheme for fog-based vehicular networks to mitigate the high network transmission latency caused by spectrum resource contention and backhaul constraints in high-density vehicle scenarios, through joint optimization of user association and radio resource allocation to minimize the system average latency [17].

However, most existing studies treat computational tasks as indivisible wholes during offloading, and thus fail to fully account for the inherent decomposability of tasks and the heterogeneity among subtasks (i.e., variations in data volume and computational demands). Moreover, the highly dynamic nature of vehicular networks, characterized by rapid vehicle mobility, time-varying channel conditions, and the stochastic joining and leaving of resources, together with the heterogeneity among nodes in terms of computational capability, location, and workload, renders conventional static or simply greedy optimization methods inadequate for achieving stable, globally near-optimal offloading strategies. Therefore, exploring an efficient optimization algorithm that simultaneously accommodates task partitioning flexibility and adaptability to dynamic environments is of critical importance for fully realizing the potential of VFC and ensuring the Quality of Service (QoS) of vehicular applications.

The SA algorithm is a classical metaheuristic global optimization method, whose concept originates from the physical annealing process of solids. It simulates the gradual cooling of a solid material from a high-temperature, disordered state to a low-temperature, stable crystalline structure [18]. By introducing a temperature parameter and the Metropolis acceptance criterion, the algorithm can accept inferior solutions with a certain probability during the early search stage, thereby avoiding entrapment in local optima and enabling extensive exploration of the solution space. As the iteration proceeds, the temperature gradually decreases, allowing the algorithm to focus on refined local search within promising solution regions [19]. This characteristic renders SA particularly suitable for solving discrete, nonlinear, and multi-modal complex combinatorial optimization problems such as task offloading.

The main contributions of this paper are as follows:

- A system model is constructed that accounts for task divisibility, subtask heterogeneity, dynamic wireless channels, and varying computational capabilities of fog nodes. The problem is formulated as a delay minimization-oriented mixed-integer nonlinear programming (MINLP) problem, thereby establishing a solid theoretical foundation for subsequent algorithm design.
- A SA algorithm framework tailored for solving this problem is proposed, incorporating well-designed solution encoding, neighborhood generation mechanisms, and cooling strategies. This framework enables the acquisition of high-quality task allocation schemes within acceptable computational time, achieving near-global optimal search in dynamic environments.
- Extensive simulation experiments are conducted under various task scales and network configurations. The proposed algorithm is compared with greedy algorithms, random algorithms, and purely local processing strategies. The effectiveness and superiority of the proposed method are validated in terms of total processing delay.

The remainder of this paper is organized as follows. Section II presents the system model in detail and provides a formal formulation of the optimization problem. Section III elaborates on the specific design and implementation details of the proposed SA algorithm. Section IV describes the simulation experiment settings, along with the analysis and discussion of the results. Section V concludes the paper and outlines future research directions.

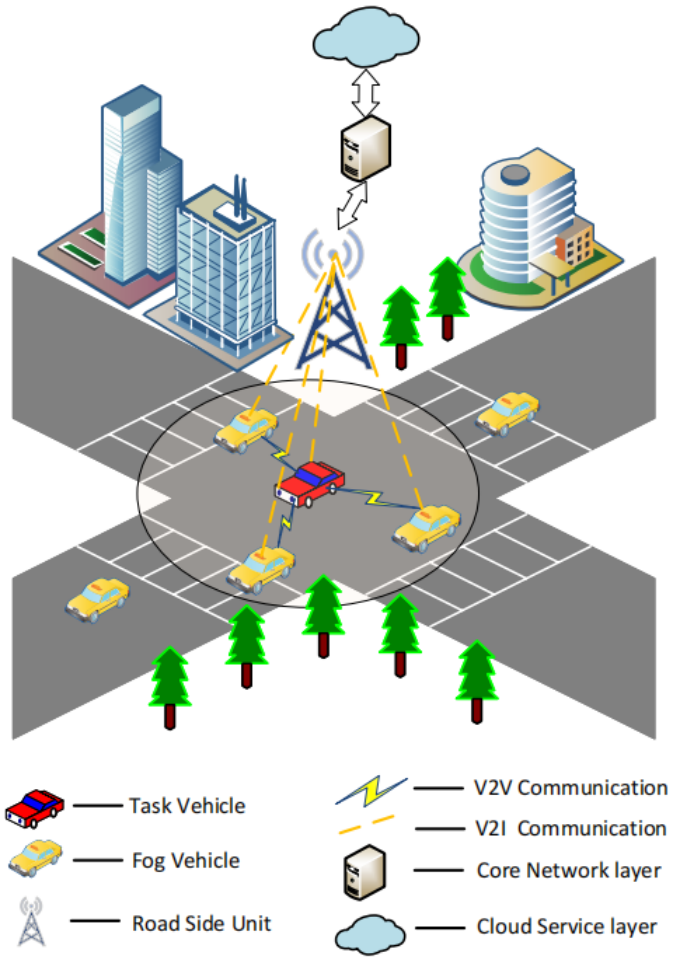

Fig. 1. The Architecture of VFC.

II. System Model and Problem Formulation

A. VFC Architecture

VFC is a distributed computing paradigm that integrates cloud computing, fog computing, and vehicular networks [20]. Its core concept lies in "sinking" computational, storage, and networking resources from the remote centralized cloud to the network edge—closer to vehicles and roadside infrastructure (e.g., Roadside Units, traffic lights)—thereby forming a hierarchical collaborative processing system. Fig. 1 illustrates the system architecture of VFC, which consists of four layers: the terminal layer, the fog node layer, the core network layer, and the cloud service layer.The circular area around the vehicle represents its communication coverage.

- Terminal layer: Composed of vehicles capable of environmental sensing, data collection, and task generation, this layer constitutes the fundamental component of the overall system.
- Fog node layer: Consists of roadside units (RSUs), base stations, and other infrastructure deployed at the network edge, providing proximate computational and storage resources.
- Core network layer: Serves as the intermediary layer connecting the fog node layer and the cloud service layer, ensuring reliable and manageable wide-area connectivity among vehicles, fog nodes, and the cloud.
- Cloud service layer: Refers to remote data centers equipped with massive storage and powerful computational capabilities, suitable for executing non-real-time tasks such as big data analytics and model training.

VFC, as a promising task offloading paradigm, effectively leverages the computational resources of idle vehicles. Within the VFC framework, vehicles serve as both providers and consumers of fog services [21]. On one hand, when a vehicle is unable to independently complete its computational tasks, it can request services from neighboring nodes; on the other hand, when its resources are idle, it can act as a fog node to provide communication and computational support for other vehicles. It is assumed that each vehicle is equipped with an onboard computing unit and a dedicated short-range communication (DSRC) module to enable reliable communication with neighboring vehicles and roadside infrastructure [22]. VFC supports multiple communication modes, such as vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication. Vehicles requiring computation offloading are defined as task vehicles, while those providing resources to other vehicles are defined as fog vehicles.

B. System Model

In intelligent transportation systems, vehicles can generate a variety of tasks, which typically differ in data size and computational workload [23]. Therefore, in our simulations, the values of these variables are generated independently. During the task offloading process, complex tasks need to be partitioned into multiple subtasks and offloaded to different fog vehicles for processing. Under our assumption, tasks are not simply divided equally; hence, the data sizes and computational workloads of the subtasks are not identical. This assumption aligns with practical application scenarios.

Consider a task offloading model in which a task vehicle generates a computationally intensive task. Due to its limited onboard computational resources, the task vehicle cannot complete this task independently within the required deadline. Under this scenario, the task vehicle regards multiple fog vehicles within its communication range as an available pool of distributed computing resources, and seeks to concurrently complete the task through task partitioning and offloading. To fully leverage the advantages of parallel processing in distributed resources and to accommodate the potential heterogeneity among subtasks, we adopt a fine-grained task partitioning model. It is assumed that the task vehicle generates a composite task with data size D and computational workload C. This task can be dynamically partitioned into N subtasks, denoted as the set $\mathcal{N} = \{1, \ldots, N\}$. The profile of each subtask n is expressed as $\text{subtask}_n = \{D_n, C_n, \tau_n\}$, where D_n represents the data size of subtask n, C_n denotes the computational workload of subtask n (i.e., the number of CPU cycles required to complete the subtask) and τ_n is the maximum tolerable delay for subtask n. The task partitioning must satisfy the following resource conservation constraints:

$$\sum_{n=1}^{N} D_n = D, \tag{1}$$

$$\sum_{n=1}^{N} C_n = C. \tag{2}$$

It is worth noting that the data sizes and computational workloads of subtasks are not allocated equally, which better reflects the heterogeneous nature of task modules in practical applications. For instance, image recognition subtasks and path planning subtasks differ in their reliance on computational resources and data dependencies.

Within the communication range of the task vehicle, there are M ($M > N$) fog vehicles with idle computational resources, denoted as the set $\mathcal{M} = \{1, \ldots, M\}$. Let F_m denote the computing capability of fog vehicle m. The computational capabilities of individual fog vehicles are typically heterogeneous and may vary dynamically over time for the same vehicle. Accordingly, the values of F_m are randomly generated during the simulation process to reflect this variability. In general, a higher computational capability results in shorter computation time for completing a given task. In vehicular environments, many tasks are closely related to computer vision and are highly dependent on GPU acceleration [24]. Therefore, this paper adopts TFLOPS (Tera floating-point operations per second) as the metric for computational capability, which is a commonly used standard for evaluating GPU performance. For ease of calculation, the computational workload of tasks is measured in TFLO (Tera floating-point operations) [13].

We assume that within a single time slot, each subtask can only be offloaded to one fog vehicle, and each fog vehicle can only serve one subtask, thereby establishing a one-to-one correspondence. A binary matrix $\mathbf{X}$ is employed to represent the task assignment policy, where each column of the matrix corresponds to the assignment strategy of a respective subtask. Specifically, let x_{mn} denote the binary decision variable:

$$x_{mn} = \begin{cases} 1, & \text{if subtask } n \text{ is offloaded to fog vehicle } m, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The task allocation constraints can be expressed as follows:

$$\sum_{m=1}^{M} x_{mn} = 1, \quad \forall n \in \{1, \ldots, N\}, \tag{4}$$

$$\sum_{n=1}^{N} x_{mn} \leq 1, \quad \forall m \in \{1, \ldots, M\}, \tag{5}$$

$$x_{mn} \in \{0, 1\}, \quad \forall m \in \{1, \ldots, M\}, \forall n \in \{1, \ldots, N\}, \tag{6}$$

Equation (4) ensures each subtask is assigned to exactly one fog vehicle. Equation (5) ensures each fog vehicle processes at most one subtask. Equation (6) denotes the binary constraint for task allocation.

To calculate the transmission rate between the task vehicle and fog vehicle m, we employ Shannon's theorem. The data transmission rate R_m can be expressed as follows:

$$R_m = B \log_2 \left(1 + \frac{P d_m^{-\alpha} h^2}{N_0}\right), \tag{7}$$

where, B is the channel bandwidth, P denotes the transmission power, d_m represents the transmission distance between the task vehicle and fog vehicle m, h represents the Rayleigh channel coefficient following a complex Gaussian distribution, N_0 is the power of Additive White Gaussian Noise (AWGN), α denotes the path loss exponent.

Consequently, the processing time for subtask n is given by:

$$t_n = \frac{D_n}{R_m} + \frac{C_n}{F_m}, \tag{8}$$

where D_n is the data size of subtask n, C_n is the computational workload of subtask n, R_m is the communication rate from the task vehicle to fog vehicle m, and F_m is the computing capability of fog vehicle m.

Since all subtasks are processed in parallel, the total processing time of the task is determined by the maximum processing time among all subtasks:

$$t = \max_{n \in \{1, \ldots, N\}} t_n \tag{9}$$

C. Formulation of the latency minimization problem in a min–max form

Our optimization objective is to minimize the total completion time t of the composite task by intelligently selecting the task assignment matrix $\mathbf{X}$, subject to a set of system constraints. The total completion time t is determined by the maximum completion time among all subtasks. Therefore, from a min–max perspective, the problem is formulated as the following constrained optimization problem.

$$\begin{aligned} P: &\min \max_{n \in \{1, \ldots, N\}} t_n, \\ \text{s.t. } &C1: x_{mn} \in \{0, 1\}, \quad \forall m \in \{1, \ldots, M\}, \forall n \in \{1, \ldots, N\}, \\ &C2: \sum_{m=1}^{M} x_{mn} = 1, \quad \forall n \in \{1, \ldots, N\}, \\ &C3: \sum_{n=1}^{N} x_{mn} \leq 1, \quad \forall m \in \{1, \ldots, M\}, \\ &C4: t_n \leq \tau_n, \quad \forall n \in \{1, \ldots, N\}. \end{aligned} \tag{10}$$

Here, C1 represents the binary constraint imposed on task assignment. C2 and C3 ensure that each subtask is offloaded to exactly one fog vehicle and that each fog vehicle handles at most one subtask. C4 stipulates that the processing time of each subtask must not exceed its maximum tolerable delay, τ_n denotes the maximum tolerable delay of subtask n.

Problem P is formulated as a MINLP problem. Due to the presence of binary decision variables and the nonlinear terms in both the objective function and constraints, this problem is classified as NP-hard and cannot be solved to exact optimality within polynomial time. Therefore, it is both necessary and

practical to pursue efficient heuristic or metaheuristic algorithms to obtain high-quality approximate solutions. Therefore, we employ the SA algorithm for solving the problem.

III. SA Algorithm Design

This paper proposes a task offloading method based on the SA algorithm to minimize time delay. The core idea of the algorithm is to draw upon the physical laws governing the annealing process of solids, thereby significantly increasing the probability of locating the global optimal solution, namely the optimal offloading strategy—within a complex solution space. Algorithm 1 presents the pseudocode for the proposed method.

A. Algorithm Principle

The SA algorithm is a metaheuristic optimization method inspired by the physical annealing process of solids. Its fundamental principle lies in simulating the stochastic thermal motion of particles at high temperatures and the trend of energy minimization during gradual cooling, thereby approaching the global optimum through probabilistic search [25]. The algorithm commences with a sufficiently high initial temperature. During each iteration, a neighboring solution is randomly generated from the current solution, and inferior solutions are accepted with a certain probability according to the Metropolis criterion. This mechanism ensures extensive exploration of the solution space at high temperatures, effectively avoiding entrapment in local optima. As the temperature parameter progressively decreases in accordance with a predefined cooling schedule, the algorithm gradually refines its focus toward promising regions of the solution space and ultimately stabilizes at a high-quality solution upon reaching low temperatures [26]. The probabilistic acceptance mechanism of SA enables it to effectively escape local optima, while the gradual cooling process ensures the stability of the final solution. Compared with exact solution methods, SA is capable of providing high-quality approximate solutions to NP-hard problems within reasonable computational time; in contrast to deterministic heuristic approaches such as greedy algorithms, it exhibits superior global search capability. Therefore, employing SA to address the task offloading delay minimization problem in VFC not only guarantees solution quality but also demonstrates favorable robustness and scalability.

B. Design of Core Algorithm Components

In the code implementation, a matrix encoding scheme is adopted to represent the task assignment solution. A binary assignment matrix $\mathbf{X}$ is defined, where $x_{mn} = 1$ indicates that subtask n is offloaded to fog vehicle m. This representation directly corresponds to the problem model and facilitates constraint verification and objective value computation. To satisfy constraints C2 and C3, the algorithm consistently maintains a feasible solution, ensuring that each column of the matrix contains exactly one entry of 1 and each row contains at most one entry of 1. This encoding scheme reduces the solution space from $2^{M\times N}$ to $\frac{M!}{(M-N)!}$, thereby significantly improving search efficiency. The initial solution is generated using a random assignment method, with the specific steps outlined as follows:

Step 1: Randomly select N distinct fog vehicles from the total M fog vehicles to form the initial set of fog vehicles.

Step 2: Assign the N subtasks to these N fog vehicles in a one-to-one manner, ensuring that each subtask is processed by a unique fog vehicle.

Algorithm 1 SA-based Task Offloading and Latency Optimization in VFC

Input: data size D, computation size C, number of subtasks N

Output: Task assignment matrix X, minimum total completion time t_{total}

Initialization
Set current temperature T_0, cooling coefficient a, iterations per temperature L, maximum iterations max_iter, minimum temperature T_{min}
Generate initial feasible solution $X_{current}$
Compute objective value of current solution: $t_{current} \leftarrow \max t_n$
Initialize historical best solution: $X_{best} \leftarrow X_{current}$, $t_{best} \leftarrow t_{current}$
Annealing Iteration
while $T > T_{min}$ **and** iteration count $< max\_iter$ **do**
  for $i = 1$ to L **do**
    Generate neighbor solution $X_{neighbor}$ from $X_{current}$:
      Randomly select two distinct subtasks n_1 and n_2
      Swap their assigned fog vehicles to obtain new assignment
      Verify if new solution satisfies constraints; if not, regenerate neighbor solution
    Compute objective value of neighbor: $t_{neighbor} \leftarrow \max t_n$
    Compute objective difference: $\Delta t = t_{neighbor} - t_{current}$
    if $\Delta t < 0$ **then**
      Accept new solution: $X_{current} \leftarrow X_{neighbor}$, $t_{current} \leftarrow t_{neighbor}$
      if $t_{current} < t_{best}$ **then**
        Update historical best: $X_{best} \leftarrow X_{current}$, $t_{best} \leftarrow t_{current}$
      end if
    else
      Accept worse solution with probability $p = \exp(-\Delta t/T)$
      if random number $rand() < p$ **then**
        $X_{current} \leftarrow X_{neighbor}$, $t_{current} \leftarrow t_{neighbor}$
      end if
    end if
  end for
  Cooling: $T \leftarrow a \cdot T$
  Record current temperature T, current cost $t_{current}$, historical best cost t_{best}
end while
Output optimal assignment: $X \leftarrow X_{best}$, minimum total completion time: $t_{total} \leftarrow t_{best}$

Step 3: Verify whether the generated assignment scheme satisfies the delay constraints for all subtasks.

Step 4: If the constraints are not satisfied, repeat steps 1–3 until a feasible initial solution is obtained.

This strategy ensures both the randomness and feasibility of the initial solution, thereby avoiding futile searches within the infeasible solution space. The design of the neighborhood structure directly influences the search efficiency and the quality of the final solution. This paper adopts a task-swapping-based neighborhood generation mechanism, which operates as follows: two distinct subtasks are randomly selected from the current solution, and their assigned fog vehicles are exchanged. The resulting new solution is then verified against all constraints. If the new solution violates any constraint, the original solution is retained; otherwise, the new solution is accepted as a neighboring solution.

The objective function is defined as the completion time in the problem model, i.e., the maximum processing time among all subtasks. Due to the parallel execution of subtasks, the total completion time is determined by the slowest subtask, which reflects the "bottleneck effect" inherent in task offloading.

C. Annealing Strategy

The performance of the SA algorithm is largely contingent upon the scheduling strategy of the temperature parameter. This paper adopts the classical proportional cooling strategy:

$$T_{k+1} = a \times T_k, \tag{11}$$

where T_k is the temperature at iteration k and a $(0 < a < 1)$ represents the cooling coefficient.

The ability of the SA algorithm to avoid entrapment in local optima stems primarily from its distinctive probabilistic acceptance criterion—the Metropolis criterion. During the iteration process, this criterion not only accepts new solutions that decrease the objective function value (i.e., reduce latency) but also accepts inferior solutions that increase the objective function value with a certain probability. At temperature T, for the current solution X_{current} and the neighbor solution X_{neighbor}, with corresponding objective function values t_{current} and t_{neighbor}, the time cost difference is defined as:

$$\Delta t = t_{\text{neighbor}} - t_{\text{current}}. \tag{12}$$

Then, the probability P is defined as follows:

$$P = \begin{cases} 1, & \Delta t < 0, \\ \exp\left(-\frac{\Delta t}{T}\right), & \text{otherwise.} \end{cases} \tag{13}$$

If the new solution is superior to the current solution, it is always accepted. If the new solution is inferior to the current solution, it is accepted with a probability of $\exp\left(-\frac{\Delta t}{T}\right)$, which decreases as the temperature lowers. This probabilistic acceptance mechanism is key to the SA algorithm's ability to escape local optima.

TABLE I
SIMULATION PARAMETERS.

Parameters	Value
Number of Vehicles	5–20
Transmission power of vehicles	30 dBm
Noise power	-114 dBm
Bandwidth	30 MHz
Path loss exponent α	3
Radius of vehicles' communication coverage	200 m
Computation size	50–250 TFLO
Computing capacity of each vehicle	5–15 TFLOPs
Data size of tasks	100–500 Mb

IV. PERFORMANCE EVALUATION

To validate the effectiveness of the proposed SA algorithm, systematic experimental tests were conducted within a simulation environment. The algorithm was implemented in the Python programming language, utilizing scientific computing and visualization libraries such as NumPy and Matplotlib. In the transmission model, the distance between the task vehicle and each fog vehicle remains within the communication coverage radius, thereby ensuring a stable and continuous communication link. The main parameters and their respective values are listed in Table I.

To comprehensively evaluate the performance of the proposed SA algorithm, we compare it against the following four baseline strategies:

- Two-stage Scheme: Reference [13] proposes a two-stage approach that first quickly assigns subtasks to fog vehicles using a matching algorithm, and then solves the bandwidth resource allocation via convex optimization based on the fixed task assignment, thereby reducing task processing latency.
- Greedy Algorithm: This strategy assigns each subtask to the nearest available fog vehicle in sequence, with distance as the sole priority. It serves as a baseline for local optimization.
- Random Algorithm: This strategy generates task assignment schemes that satisfy the basic constraints in a random manner, serving as a baseline for random search.
- Local Processing: All subtasks are executed locally on the task vehicle, with the onboard computational capability assumed to be 5 TFLOPs. This strategy serves as the worst-case performance lower bound.

We conducted multiple sets of experiments to evaluate the performance of the proposed SA algorithm. Since the data sizes and computational workloads of subtasks are randomly partitioned, each delay result is obtained by averaging over 100 independent simulation runs, thereby providing a realistic assessment of the actual performance of each scheme. Statistical analysis of the results from 100 independent simulation runs shows that the coefficient of variation (CV) for the SA algorithm remains consistently below 5% across all scenarios. This result indicates that the SA-based approach is highly stable and can reliably converge to a near-global optimal solution, despite the stochastic nature of subtask partitioning and node resource distribution.

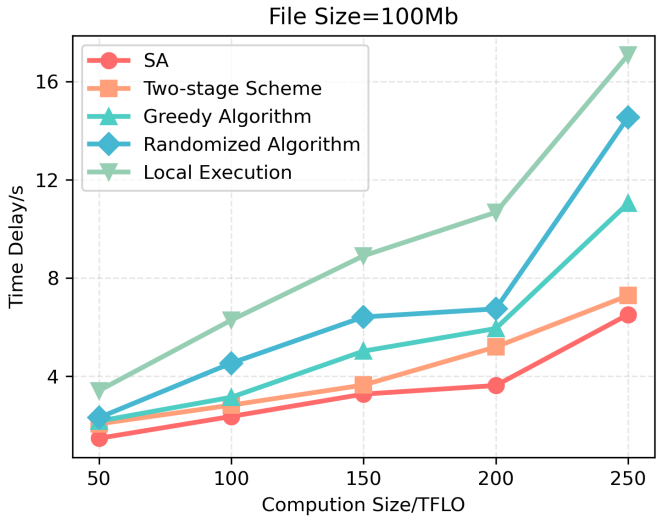

Fig. 2. Comparison of time delays of five algorithms under different computational sizes when the task size is 100 Mb.

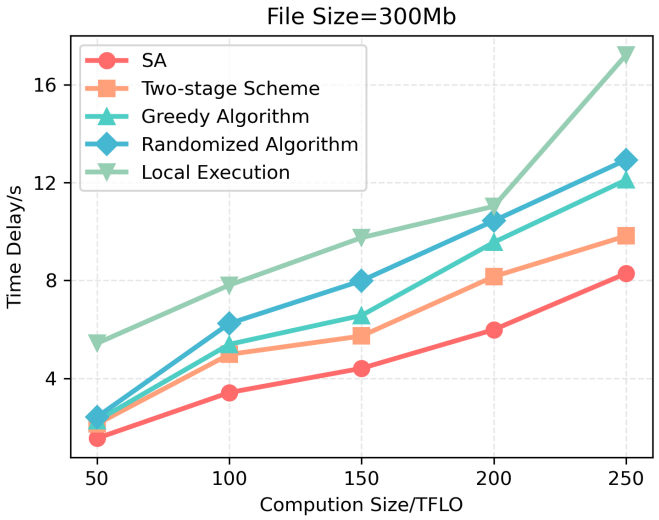

Fig. 4. Comparison of time delays of five algorithms under different computational sizes when the task size is 300 Mb.

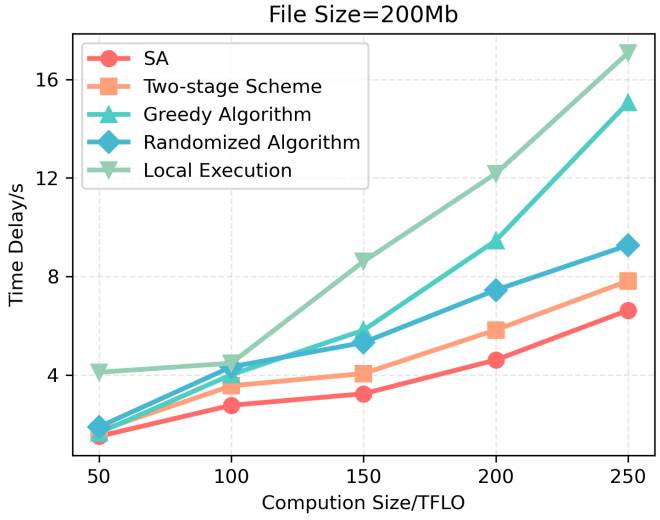

Fig. 3. Comparison of time delays of five algorithms under different computational sizes when the task size is 200 Mb.

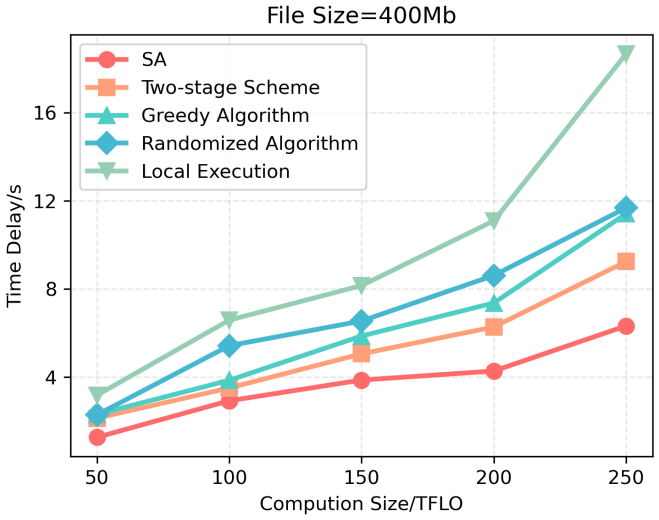

Fig. 5. Comparison of time delays of five algorithms under different computational sizes when the task size is 400 Mb.

In terms of computational overhead, the proposed SA algorithm demonstrates favorable efficiency for deployment in VFC environments. For the tested scenarios with up to 20 subtasks, the average runtime of the algorithm ranges within one second, depending on the problem scale. This level of computational cost is acceptable for task offloading planning and meets the sub-second decision requirements in many fog computing scenarios.

Fig. 2–5 illustrate the system time delay variations of the five schemes under different data volumes (100-400 Mb) as the task computational workload increases. Comprehensive analysis of the experimental data reveals that the proposed SA algorithm achieves the lowest total processing delay across all tested scenarios, significantly outperforming the greedy algorithm, random algorithm, and local processing strategy. This substantiates the global optimization capability of the SA algorithm. As the task computational scale increases (from 50 TFLOPS to 250 TFLOPS), the performance advantage of SA over the greedy and random algorithms becomes more pronounced. Specifically, when processing computationally intensive tasks (e.g., at a computational scale of 250 TFLOPS), the superiority of SA is particularly evident. For instance, in the scenario with a task data size of 100 Mb and a computational workload of 250 TFLOPS, the latency of SA is merely 89.4% of that of the two-stage method, 59.0% of that of the greedy algorithm, and 44.8% of that of the random algorithm. In the scenario with a task data size of 300 Mb and a computational workload of 250 TFLOPS, the latency of simulated annealing is reduced by approximately 15.6%, 31.5% and 35.9% compared to the two-stage method, greedy algorithm and random algorithm, respectively. This indicates that when faced with high computational loads, SA is capable

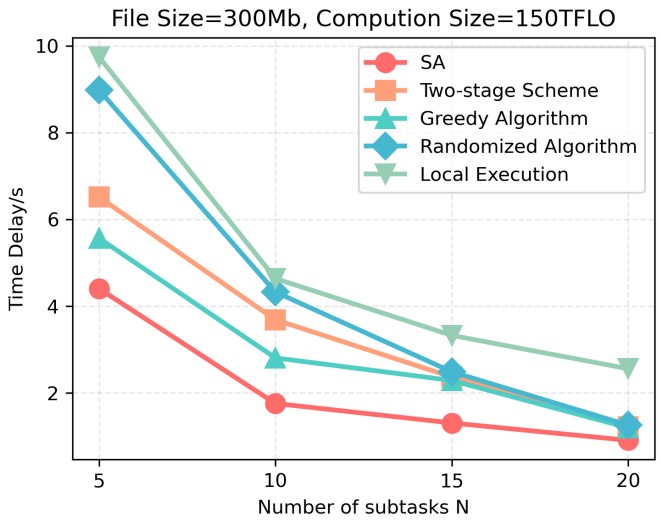

Fig. 6. Comparison of time delays of five algorithms with different N values.

of more intelligently balancing transmission and computation, preferentially assigning computationally intensive subtasks to fog vehicles with higher computational capabilities, whereas the distance-based decision-making of the greedy algorithm performs poorly in such scenarios.

Furthermore, the results demonstrate that as the overall computational demand of tasks increases, the delays of all algorithms exhibit an upward trend; however, the delay curve of SA shows the most gradual increase, reflecting its favorable scalability and robustness in complex scenarios. The local processing strategy consistently yields the highest delay, which negatively validates the necessity of implementing collaborative task offloading in VFC.

Fig. 6 illustrates the relationship between the number of subtasks and the total processing delay under a fixed task data volume (300 Mb) and computational workload (150 TFLOPS). The experimental results demonstrate that as the number of subtask partitions N increases (from 5 to 20), the total processing delay under all five strategies exhibits a significant downward trend. Specifically, when the task is partitioned into $N = 5$ subtasks, the delay is relatively high across all algorithms, with the SA achieving approximately 4.41s. As N increases to 10 and 15, the advantages of parallel execution become more pronounced, and the delay of the SA drops significantly to approximately 1.76s and 1.31s, respectively. Ultimately, at the finest granularity of $N = 20$, the SA achieves its lowest delay of 0.91s, maintaining a clear advantage over the baseline strategies.

This phenomenon is primarily attributed to the parallel processing advantages brought about by fine-grained task partitioning. As N increases, the data size D_n and computational workload C_n allocated to each subtask decrease correspondingly, thereby reducing the transmission time and computation time of individual subtasks. Since subtasks can be executed in parallel on different fog vehicles, the overall system completion time is determined by the execution time of the ”slowest” subtask; consequently, the reduction in individual subtask processing time directly contributes to lower total delay. Among all strategies, the SA algorithm demonstrates the strongest optimization capability, achieving the lowest delay at $N = 20$ and significantly outperforming the other strategies. This indicates that SA can more effectively exploit the scheduling flexibility afforded by task partitioning, achieving more balanced and efficient resource matching. These results indirectly confirm that, in VFC, appropriate fine-grained task partitioning combined with intelligent optimization algorithms can effectively enhance processing efficiency and system parallelism. The experimental findings demonstrate that the SA algorithm proposed in this paper exhibits excellent performance in solving the task offloading problem in VFC.

The above model assumes that the network environment (including channel states, fog node availability, etc.) remains relatively stable during one offloading decision cycle. In actual VFC scenarios, these parameters may evolve dynamically due to vehicle mobility and resource contention. However, the proposed SA algorithm is designed with adaptability in mind: its fast convergence characteristics enable rapid re-optimization when the system state deviates from the initial assumptions. Specifically, the algorithm can be executed periodically or triggered upon significant environmental changes (e.g., a fog node moving out of communication range or a sudden fluctuation in computational load) to generate a new offloading scheme based on the updated system state. This periodic re-optimization strategy ensures the effectiveness of the solution in dynamic environments without requiring explicit mobility prediction. Extending the model to incorporate mobility prediction and proactive rescheduling mechanisms (such as trajectory prediction or handover strategies) can be considered a direction for future work.

The current formulation focuses on a single task vehicle offloading its composite task to a set of fog nodes. Although this setting captures the core complexity of task partitioning and resource allocation, a more realistic VFC scenario involves multiple task vehicles simultaneously competing for the same pool of fog resources. In such a multi-agent scenario, constraints C2 and C3, which ensure a one-to-one mapping between subtasks and fog nodes, would be insufficient to capture resource contention among vehicles. A three-dimensional assignment variable $x_{m,n,k}$ can be introduced to indicate whether subtask n of task vehicle k is offloaded to fog node m, along with additional constraints ensuring that each fog node serves at most one subtask across all task vehicles at any given time. The resulting problem becomes a multi-task-vehicle collaborative offloading problem, which remains within the scope of metaheuristic optimization. The proposed SA framework can be adapted to this extended scenario by redesigning the solution encoding to accommodate the additional dimension and modifying the neighborhood generation operations accordingly, while preserving the core annealing mechanism.

V. Conclusion

This paper investigates the task offloading delay optimization problem in VFC and proposes an intelligent offloading

method based on the SA algorithm. By establishing a system model that supports fine-grained task partitioning, the problem is formulated as a MINLP problem, and a simulated annealing solver with task-swapping-based neighborhood operations is designed accordingly. Simulation results demonstrate that, compared with the greedy algorithm, random algorithm, and purely local processing strategy, the proposed algorithm effectively balances transmission and computational resources, achieving superior offloading decisions across various task scales and network scenarios. These findings validate its effectiveness and superiority in enabling low-latency Internet of Vehicles services. The current model focuses on a single-task-vehicle scenario. Future work can extend this framework to multi-agent collaborative offloading scenarios involving multiple task vehicles competing for resources. Exploring the integration of SA with artificial intelligence approaches, such as deep learning, to enable adaptive parameter tuning and more efficient large-scale problem solving represents a promising research direction.

REFERENCES

[1] P. Mishra and G. Singh, "Internet of vehicles for sustainable smart cities: Opportunities, issues, and challenges," *Smart Cities (2624-6511)*, vol. 8, no. 3, 2025.

[2] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 36–44, 2017.

[3] S. Zhou, Y. Sun, Z. Jiang, and Z. Niu, "Exploiting moving intelligence: Delay-optimized computation offloading in vehicular fog networks," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 49–55, 2019.

[4] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys Tutorials*, vol. PP, no. 99, pp. 1–1, 2017.

[5] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2017.

[6] C. Quadros, A. Santos, M. Gerla, and E. Cerqueira, "Qoe-driven dissemination of real-time videos over vehicular networks," *Computer Communications*, vol. 91-92, no. oct.1, pp. 133–147, 2016.

[7] M. P. Toopchinezhad and M. Ahmadi, "Deep reinforcement learning for delay-optimized task offloading in vehicular fog computing," in *2025 29th International Computer Conference, Computer Society of Iran (CSICC)*. IEEE, 2025, pp. 1–6.

[8] B. Cho and Y. Xiao, "A repeated unknown game: Decentralized task offloading in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 10, pp. 13 430–13 446, 2023.

[9] S. Bharathi and P. Prakasam, "A systematic review on resource allocation, task offloading, and security issues in vehicular fog computing: research challenges and future directions," *Engineering Research Express*, vol. 7, no. 2, p. 022303, 2025.

[10] O. Nazih, N. Benamar, H. Lamaazi, and H. Chaoui, "Toward secure and trustworthy vehicular fog computing: A survey," *IEEE Access*, vol. 12, pp. 35 154–35 171, 2024.

[11] C. Sneha, A. S. Chakravarthy, and T. Veni, "A comprehensive review of task offloading methods in vehicular fog computing," *Computers and Electrical Engineering*, vol. 129, p. 110847, 2026.

[12] Z. Gao, L. Yang, and Y. Dai, "Fast adaptive task offloading and resource allocation via multiagent reinforcement learning in heterogeneous vehicular fog computing," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 6818–6835, 2022.

[13] X. Wu, S. Zhao, and H. Deng, "Joint task assignment and resource allocation in vfc based on mobility prediction information," *Computer Communications*, vol. 205, pp. 24–34, 2023.

[14] J. Bi, X. Xue, H. Yuan, and J. Zhang, "Latency-minimized computation offloading in vehicle fog computing with improved whale optimization algorithm," in *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2023, pp. 5003–5008.

[15] E. F. Maleki, L. Mashayekhy, and S. M. Nabavinejad, "Mobility-aware computation offloading in edge computing using machine learning," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 328–340, 2021.

[16] F. Zeng, Z. Zhang, and J. Wu, "Task offloading delay minimization in vehicular edge computing based on vehicle trajectory prediction," *Digital Communications and Networks*, vol. 11, no. 2, pp. 537–546, 2025.

[17] K. Zhang, M. Peng, and Y. Sun, "Delay-optimized resource allocation in fog-based vehicular networks," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1347–1357, 2020.

[18] F. Kosanoglu, M. Atmis, and H. H. Turan, "A deep reinforcement learning assisted simulated annealing algorithm for a maintenance planning problem," *Annals of Operations Research*, pp. 1–32, 2022.

[19] A. S. Mustafa, S. Yussof, and N. A. M. Radzi, "Multi-objective simulated annealing for efficient task allocation in uav-assisted edge computing for smart city traffic management," *Access, IEEE*, vol. 13, no. 000, pp. 24 251–24 275, 2025.

[20] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 87–93, 2019.

[21] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE transactions on vehicular technology*, vol. 65, no. 6, pp. 3860–3873, 2016.

[22] O. Dokur, G. Olenscki, and S. Katkoori, "An edge computing approach forautonomous vehicle platooning," *IFIP Advances in Information and Communication Technology*, pp. 332–349, 2022.

[23] Q. Wu, H. Liu, R. Wang, P. Fan, and Z. Li, "Delay sensitive task offloading in the 802.11p based vehicular fog computing systems," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2019.

[24] S. Shi, J. Cui, Z. Jiang, Z. Yan, G. Xing, J. Niu, and Z. Ouyang, "Vips: Real-time perception fusion for infrastructure-assisted autonomous driving," *GetMobile: Mobile Computing and Communications*, vol. 27, pp. 28 – 33, 2023.

[25] C. Liang, Z. Gao, B. Wang, K. Cheng, and Y. Zhao, "Qeclo: A novel qos-aware joint optimization of energy and latency for vfc task offloading," *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pp. 1814–1819, 2024.

[26] C. Liang, Y. Zhao, Z. Gao, K. Cheng, B. Wang, and L. Huang, "Ealso: joint energy-aware and latency-sensitive task offloading for artificial intelligence of things in vehicular fog computing," *Wireless Networks (10220038)*, vol. 31, no. 1, 2025.

Lin Chai graduated with a Bachelor of Science degree in Mathematics and Applied Mathematics from Lyuliang University in 2022. She is currently pursuing a Master of Science degree in Probability Theory and Mathematical Statistics at Yunnan Minzu University. Her research interests focus on Vehicular Ad-hoc Networks.

Jun Wang graduated with a Bachelor of Science degree in Mathematics and Applied Mathematics from Boda College of Jilin Normal University in 2023. She is currently pursuing a Master of Science degree in Probability Theory and Mathematical Statistics at Yunnan Minzu University. Her research interests focus on Vehicular Ad-hoc Networks and Fog Networks.

Yumei Yang graduated with a Bachelor of Science degree in Mathematics and Applied Mathematics from Yunnan Minzu University in 2024. She is currently pursuing a Master of Science degree in Probability Theory and Mathematical Statistics at Yunnan Minzu University. Her research interests focus on Vehicular Ad-hoc Networks.

Wu Wang received his B.S. and M.S. degrees from Yunnan University, China, in 2003 and 2007, respectively. Received his Ph.D. degree from Future University Hakodate, Japan, in 2018. He is currently an associate professor at the School of Mathematics and Computer Science at Yunnan Minzu University. His research interest includes ad hoc networks, neural network, and network security.