mathematics

MDPI

Article

Joint Task Offloading and Resource Allocation in Mobile Edge Computing-Enabled Medical Vehicular Networks

Chuangchuang Zhang [1,*], Siquan Liu [1], Hongyong Yang [1], Guanghai Cui [1], Fuliang Li [2] and Xingwei Wang [2]

[1] School of Information and Electrical Engineering, Ludong University, Yantai 264025, China
[2] College of Computer Science and Engineering, Northeastern University, Shenyang 110169, China
* Correspondence: zhangcc_neu@163.com

Abstract: In medical vehicular networks, medical vehicles can serve as efficient mobile medical service points to provide necessary and critical medical services for patients while in motion. The delay requirement is very vital for medical services to guarantee service quality and save the lives of patients. Mobile Edge Computing (MEC), as an emerging network paradigm, enables the computation extensive tasks to be offloaded to edge servers, efficiently reducing the delay and bandwidth demands. MEC technology is a promising solution to provide high-quality medical services for users in medical vehicular networks. However, task offloading and resource allocation incurs additional service delay and energy consumption, affecting the overall service performance and Quality of Experience (QoE) of users. Thus, realizing the optimal task offloading and resource allocation in MEC-enabled medical vehicular networks, to reduce task completion time and energy consumption, becomes a potential challenge. To address the challenge, we investigate the joint task offloading and resource allocation problem in MEC-enabled medical vehicular networks to improve the QoE of users. Considering the resource requirements and QoS constraint, we formulate a multi-objective optimization model, with the target of average task completion time and average energy consumption minimization. On this basis, we propose a MOEAD-based task offloading and resource allocation (IMO) algorithm to solve it. Furthermore, in order to obtain the optimal solution and speed up the algorithm convergence, we design a greedy strategy-based population initialization algorithm. The extensive simulations demonstrate that compared to existing algorithms, our proposed IMO algorithm can obtain a smaller average completion time, and achieve better tradeoff between task completion time and energy consumption.

Keywords: medical vehicular networks; MEC; task offloading; resource allocation; multi-objective optimization

MSC: 68M10

check for updates

Academic Editor: Andrea Scozzari

Received: 25 November 2024
Revised: 24 December 2024
Accepted: 25 December 2024
Published: 27 December 2024

Citation: Zhang, C.; Liu, S.; Yang, H.; Cui, G.; Li, F.; Wang, X. Joint Task Offloading and Resource Allocation in Mobile Edge Computing-Enabled Medical Vehicular Networks. *Mathematics* **2025**, *13*, 52. https://doi.org/10.3390/math13010052

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the rapid advancements of 5G and Internet of Medical Things (IoMT) technologies, medical vehicular networks have been paid significant attention in recent years [1]. In medical vehicular networks, medical vehicles can serve as efficient mobile medical service points to automatically provide necessary and critical medical services for patients while in motion. Medical vehicular services can provide real-time medical assistance for users in the vehicles, especially in emergency situations. The delay requirement is very vital for medical services to guarantee service quality and save the lives of patients. As an emerging network paradigm, Mobile Edge Computing (MEC) has been regarded as a

promising solution to enhance the reliability and scalability of medical vehicular services [2]. In MEC-enabled medical vehicular networks, MEC is combined with roadside units to transfer efficient computing power from remote cloud to network edge close to medical vehicular users [2]. Thereby, it reduces service latency and the demand for bandwidth resource. Offloading the medical vehicular tasks to nearby edge servers can significantly reduce the data transmission time and improve response speed, enabling an immediate response to the conditions of patients. Through efficient task offloading and resource allocation, the medical vehicular network can provide efficient and reliable medical services in a mobile environment, significantly improving the medical service quality.

Figure 1 depicts a simple model diagram of MEC-enabled medical vehicular network scenarios. The medical vehicular network system mainly consists of different types of vehicles, and cloud service center and edge servers. The medical vehicles can communicate with cloud service center and edge servers by leveraging 5G networks. For example, when a user in the vehicle requests the emergency treatment due to a sudden illness, the vehicle automatically collects the vital sign data of the patient. Then, the vehicle transmits the collected information to the expert side of the receiving hospital through 5G networks, and obtains the remote guidance from the expert. The service request sent from the vehicle can be processed by the edge server close to it, providing a fast response to the patient. Through the MEC paradigm, the medical vehicular network can monitor the physiological data of patients in real-time and make an immediate response, such as automatically adjusting the interior environment of vehicles or directly contacting the nearest medical facilities. Through offloading medical vehicular tasks to edge servers, it can assist doctors in analyzing the condition and developing treatment plans as quickly as possible, as well as accelerating life rescue efforts. Fast response to service requests is crucial to save the lives of patients and improve service quality.

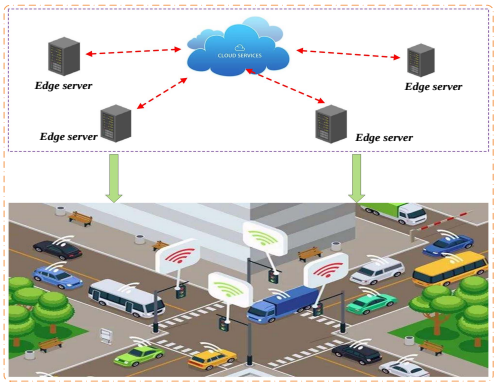


Figure 1. A diagram of the MEC-enabled medical vehicular network.

An efficient task offloading and resource allocation mechanism can effectively offload the medical vehicular tasks to vehicular edge servers, thereby reducing the computational load on medical vehicles and improving the processing efficiency. The optimization strategy can ensure the continuity and stability of critical medical services [3]. It is particularly important for responding to emergency medical situations. However, the limited computational capacity of vehicles and limited resource capacity of vehicular edge servers make task offloading and resource allocation complex. The inefficient task offloading and resource allocation method in medical vehicular networks incurs additional service delay and energy consumption. It inevitably affects the service performance and Quality of Experience (QoE) of users [4]. To guarantee QoE, realizing the optimal task offloading and resource allocation in MEC-enabled medical vehicular networks to reduce task completion time and energy consumption becomes a potential challenge.

In recent years, many efforts have been paid to optimize task offloading and resource allocation to improve service performance [5–39]. The majority of existing works focus on reducing delay, improving energy efficiency, and enhancing Quality of Service (QoS) by establishing various optimization models. However, there are a few research works to study the joint task offloading and resource allocation problem in medical vehicular network scenarios. Thus, it is desirable to design an efficient task offloading and resource allocation mechanism to improve the service performance of medical vehicular networks. Different from previous works, our work focuses on task offloading and resource allocation in MEC-enabled medical vehicular networks to shorten the average task completion time of medical vehicular services while reducing the average energy consumption. On the other hand, the optimization goals of completion time and energy consumption are conflicts to each other. In such a case, it is very difficult to determine the optimal solution. A variety of evolutionary algorithms have been widely used in the field of solving multi-objective optimization. Among them, the Multi Objective Evolutionary Algorithm based on Decomposition (MOEAD) is a widely used multi-objective evolutionary algorithm [40]. MOEAD can effectively find multiple non-dominated solutions by decomposing the multi-objective problem into multiple single objective sub-problems, and employ the evolutionary algorithm. Therefore, we design a MOEAD-based task offloading and resource allocation algorithm to solve the multi-objective optimization model.

In this paper, we investigate the joint task offloading and resource allocation problem in MEC-enabled medical vehicular networks. We consider resource requirements and QoS constraints to formulate a multi-objective optimization model, with the target of average task completion time and average energy consumption. To solve the problem, we design a MOEAD-based task offloading and resource allocation algorithm. The main contributions of this paper are summarized as follows.

- We formulate the problem of joint task offloading and resource allocation in MEC-enabled medical vehicular networks as a multi-objective optimization model to minimize the average task completion time and average energy consumption while satisfying resource requirements and QoS constraints.
- We design a MOEAD-based task offloading and resource allocation (IMO) algorithm to solve the problem. Furthermore, in order to obtain the optimal solution and speed up the convergence of the IMO algorithm, we develop a greedy strategy-based population initialization algorithm.
- We conduct simulation experiments for performance evaluations. Simulation experiments demonstrate that, compared to existing algorithms, the IMO algorithm can obtain smaller average task completion time, and achieve a better tradeoff between the average task completion time and average energy consumption. By measurement, the IMO algorithm can at least save 5% average task completion time.

The rest of this paper is organized as follows. We review the related works in Section 2, and the system model is presented in Section 3. We introduce the problem formulation in Section 4, and the detailed design of the IMO algorithm is presented in Section 5. We conduct performance evaluations in Section 6, and finally conclude this paper in Section 7.

2. Related Works

Many efforts have been made to optimize task offloading in vehicular network scenarios by considering different goals, including delay optimization, energy efficiency improvement, and QoS enhancement.

2.1. Delay Optimization

To optimize delay, Nan et al. [5] formulated the task offloading and resource allocation problem in vehicular edge computing as a Mixed Integer Non-Linear Programming (MINLP) model to minimize result feedback delay. And they designed an approximation algorithm to solve it. Qi et al. [6] designed a traffic aware task offloading algorithm in vehicular edge computing to minimize response time by optimizing task and wireless bandwidth ratios. Yan et al. [7] designed a Deep Reinforcement Learning (DRL)-based task offloading algorithm to reduce system delay, and used a long short-term memory network to obtain the optimal offloading policy by leveraging the attention mechanism and DDPG algorithm. To address the load imbalance problem in vehicular edge computing, Fan et al. [8] established an M/M/1 queue to model task queuing and the task computation process. A game-based task offloading and resource allocation algorithm was designed to minimize the task process delay. Cao et al. [9] formulated a joint task offloading and resource allocation optimization model in vehicular networks to minimize system delay, and developed a relay hopping and differentiated task prioritization-based algorithm to solve it.

Wang et al. [10] designed a K-neighbor-based joint task offloading and resource allocation algorithm to minimize delay, while guaranteeing the differential privacy security of vehicles. Fan et al. [11] designed a generalized bender decomposition and reformulation linearization algorithm, to minimize the task processing delay of vehicles, by jointly considering task diversity, vehicle classification, and task process flexibility. Sun et al. [12] presented an adaptive learning-based task offloading algorithm to minimize offloading delay. The proposed algorithm works in a distributed manner with input awareness and occurrence awareness, in order to adapt to a dynamic environment. To realize load balancing and cope with the dynamics of vehicular edge computing networks, Zhang et al. [13] designed an SDN-based load-balancing task-offloading algorithm to minimize task process delay. Raza et al. [14] designed a mobility-aware partial task offloading algorithm to reduce the system delay and computing cost by considering the task allocation ratio. Liu et al. [15] formulated the multi-hop task offloading problem as an optimization model to minimize the weighted sum of execution time and computation cost. And they devised a mobility aware task offloading algorithm to solve it. Wu et al. [16] designed a semi-Markov Decision Process (MDP)-based offloading strategy to minimize offloading delay and maximize long-term reward, and presented a robust task offloading strategy. Yin et al. [17] designed a hybrid task offloading model to offload tasks to roadside units or vehicles, to cope with the dynamic changes of vehicles and uncertainty of resource allocation.

Different from the above works that only consider to reduce delay in vehicular edge computing, our work focuses on reducing task completion time and energy consumption in MEC-enabled medical vehicular networks simultaneously to improve QoE.

2.2. Energy Optimization

To improve energy efficiency, Qin et al. [18] designed a learning-based, energy-efficient task-offloading algorithm to use an idle and redundant resource, to minimize energy consumption. Tan et al. [19] designed a decentralized task offloading and resource allocation algorithm for vehicular edge computing system, to reduce response time and energy consumption, by leveraging the dual decomposition method. Dai et al. [20] formulated an Unmanned Aerial Vehicle (UAV) assisted vehicular task offloading optimization model, to minimize task delay. To solve the model, they designed a Lyapunov optimization-based task offloading algorithm. Zhao et al. [21] formulated the task offloading problem in vehicular edge computing as a combinatorial optimization model to minimize the average energy consumption and response time by considering the data dependency of tasks. Meanwhile,

a DRL-based mobility aware dependent task-offloading algorithm is devised to solve it. Cao et al. [22] established a five-objective optimization model including energy consumption, downlink delay, computation delay, load balancing, and user satisfaction to adapt to the mobility of vehicles, and designed a guiding point-based offloading algorithm to solve it.

Although the above task offloading solutions in vehicular edge computing attempt to reduce energy consumption, the joint optimization of task offloading and resource allocation is not considered. Our work tries to jointly optimize task offloading and resource allocation in MEC-enabled medical vehicular networks to improve the average completion time and average energy consumption.

2.3. QoS Optimization

To improve QoS, Chen et al. [23] designed a multi-vehicle task offloading algorithm to obtain the tradeoff between system cost and task queue length. Xue et al. [24] established a task offloading and resource allocation Stackelberg game model to determine the optimal offloading and price solution. A gradient-based resource allocation iteration algorithm and a reverse auction-based task scheduling algorithm were designed to obtain the tradeoff. Sun et al. [25] designed a vehicular task offloading and job scheduling algorithm to improve the QoS of Internet of Vehicles based on vehicle location. They further applied the ant colony optimization algorithm to realize multi-objective optimization. Bute et al. [26] designed a distributed task-offloading algorithm to determine the optimal vehicles by jointly considering link reliability, distance, available resources and relative velocity. In order to adapt to fast moving and frequent handover by leveraging SDN technology, Guo et al. [27] designed a deep Q learning-based mobility aware task offloading to realize information collection and network management. Li et al. [28] formulated the collaborative task offloading and service caching replacement problem as a mixed-integer programming model, to minimize the computing cost and delay. And they designed a DRL-based iterative algorithm to solve it. Wei et al. [29] formulated the many-to-many task offloading problem as a partially observable MDP model, and designed a multi-agent gated actor attention critic algorithm to realize stable offload and server cooperation among vehicles. Gao et al. [30] designed a DDPG-based multi-agent task offloading algorithm to address the task dependency and resource competition problem. Xu et al. [31] formulated the cooperative task offloading and resource optimization problem to maximize the service ratio, and designed a multi-agent distributed DDPG algorithm to obtain the Nash equilibrium. Wu et al. [32] developed an elastic federated and multi-agent DRL-based cooperative edge caching algorithm to optimize the cost.

The above research studies attempted to optimize task offloading or resource allocation in vehicular edge computing scenarios. However, there have been a few research studies on task offloading and resource allocation in medical vehicular networks recently. To provide efficient medical services in healthcare networks, some efforts have been made. Zhou et al. [33] designed a learning-based task offloading algorithm in Internet of Health Things (IoHT) to balance URLLC constraints and energy consumption. Dong et al. [34] designed a multi-agent reinforcement learning-based mobility aware task offloading algorithm to adapt to the mobility change. Ren et al. [35] designed a healthcare task-offloading algorithm to realize low-latency secure and reliable task offloading, by integrating SDN, blockchain and fog-assisted healthcare IoT. He et al. [36] designed a blockchain-based medical data-offloading mechanism, to achieve the optimal medical resource allocation and minimize offloading cost. Wang et al. [37] designed a multi-agent-based URLLC-constrained task offloading and resource allocation algorithm, to maximize throughput and minimize energy consumption. Ren et al. [38] designed a trust aware task-offloading algorithm to securely perform smart contracts via blockchain. Gao et al. [39] devised a

multi-agent DDPG-based task-offloading algorithm to make offloading and power control decisions independently.

Despite numerous attempts to realize effective task offloading and resource allocation, the joint optimization of average task completion time and average energy consumption in medical vehicular networks is not considered. Different from previous studies, we formulate the problem of joint task offloading and resource allocation in medical vehicular networks as a multi-objective optimization model to minimize average task completion time and average energy consumption. To solve the multi-objective optimization problem, we design a MOEAD-based task offloading and resource allocation algorithm. Furthermore, to obtain the optimal solution and speed up the algorithm convergence, we design a greedy strategy-based population initialization algorithm. Moreover, we evaluate the tradeoff between the two optimization objectives.

3. System Model

The medical vehicular tasks can be processed by the local medical vehicles, or edge servers. When a task is offloaded, it will be to edge servers through the network. Therefore, in this section, we describe the system model including the network model, task model, and computation and communication model. The main notations are listed in Table 1.

Table 1. The main notations.

Notation	Description
$H = (V, S, E)$	MEC-enabled medical vehicular network with a set V of medical vehicles, a set S of edge servers and a set E of links
c_v^i	CPU capacity of medical vehicle v_i
c_s^i	CPU capacity of edge server s_i
$t_{i,j}$	j-th task generated by v_i
$D_{i,j}$	Data size of $t_{i,j}$
$\tau_{i,j}$	Maximum tolerant delay of $t_{i,j}$
$x_{i,j}^k$	Binary decision variable indicating whether $t_{i,j}$ is offloaded to s_k
$c_{i,j}$	Number of CPU cycles required to process a bit of task
$f_{i,j}^l$	CPU resource allocated by v_i to $t_{i,j}$
$R_{i,j,k}$	Data transmission rate of $t_{i,j}$ from v_i to s_k
$P_{i,j,k}$	Transmission power of $t_{i,j}$ from v_i to s_k with the maximum threshold $p_{i,j,k}^{max}$
B	Transmission channel bandwidth
$f_{i,j,k}^s$	CPU resource allocated by s_k to $t_{i,j}$
$\tau_{i,j}^l$	Local computation delay of $t_{i,j}$
$\tau_{i,j,k}^u$	Data transmission delay of $t_{i,j}$ from v_i to s_k
$\tau_{i,j,k}^s$	Execution delay of $t_{i,j}$ on s_k
$\tau_{i,j,k}^o$	Offloading computation delay of $t_{i,j}$ on s_k
$\tau_{i,j}'$	Computation delay of $t_{i,j}$
$E_{i,j}^l$	Local computation energy consumption of $t_{i,j}$
$E_{i,j,k}^u$	Transmission energy consumption of $t_{i,j}$ from v_i to s_k
$E_{i,j,k}^s$	Computation energy consumption of $t_{i,j}$ on s_k
$E_{i,j}^o$	Offloading computation energy consumption of $t_{i,j}$
$E_{i,j}$	Total computation energy consumption of $t_{i,j}$
ξ_i	Local computation energy consumption coefficient of v_i
ζ_k	Computation energy consumption coefficient of s_k

3.1. Network Model

An MEC-enabled medical vehicular network is composed of medical vehicles and vehicular edge servers. The medical vehicles are responsible for generating medical service tasks. The vehicular edge servers are responsible for processing the offloaded tasks. Thus, we model the MEC-enabled medical vehicular network as an undirected graph

$G = (V, S, E)$, where V is the set of medical vehicles, S is the set of vehicular edge servers, and E is the set of physical links. In this work, we assume that each vehicular edge server can be associated with a set of medical vehicles. The medical service requested tasks can be processed by medical vehicles locally, or vehicular edge servers. Therefore, medical vehicles and vehicular edge servers have a certain amount of resources, such as computing resource and storage resource, to perform the local computation and offloading computation, respectively. For the simplicity, we only consider the computing resource in this work. We denote the total computing resource capacity (i.e., CPU cycles) of medical vehicle $v_i \in V$ and vehicular edge server $s_i \in S$ by c_v^i and c_s^i, respectively.

3.2. Task Model

In this work, we assume that each medical vehicle can generate a different number of medical service tasks. At the same time, medical service tasks can be offloaded to different vehicular edge servers. We use $t_{i,j}$ to represent the j-th medical service task generated by medical vehicle v_i. Furthermore, we define a binary decision variable $x_{i,j}^k$ to identify whether medical service task $t_{i,j}$ generated by medical vehicle $v_i \in V$ is offloaded to vehicular edge server $s_k \in S$. If $x_{i,j}^k = 0$, medical service task $t_{i,j}$ will be processed locally by medical vehicle v_i. Otherwise, medical service task $t_{i,j}$ will be offloaded to vehicular edge server s_k:

$$x_{i,j}^k = \begin{cases} 1, & if\ t_{i,j}\ is\ offloaded\ to\ s_k \\ 0, & otherwise \end{cases} \tag{1}$$

We define the data size and maximum tolerant delay of medical service task $t_{i,j}$ generated by medical vehicle v_i by $D_{i,j}$ and $\tau_{i,j}$, respectively.

3.3. Computation and Communication Model

The medical service tasks generated by medical vehicles can be processed either on the local medical vehicles or on vehicular edge servers.

3.3.1. Local Computation

In the case of local computation, the medical service task is processed by the local medical vehicle. Therefore, we can obtain the local computation delay of medical service task $t_{i,j}$ as follows:

$$\tau_{i,j}^l = \frac{c_{i,j} \cdot \left(1 - x_{i,j}^k\right) \cdot D_{i,j}}{f_{i,j}^l} \tag{2}$$

where $c_{i,j}$ is the number of CPU cycles required to process a bit of task, $f_{i,j}^l$ is the computing resources allocated by medical vehicle v_i to the task $t_{i,j}$ for local computation, i.e., local CPU cycle frequency of the medical vehicle.

Correspondingly, the local computation energy consumption $E_{i,j}^l$ of the medical service task $t_{i,j}$ generated by medical vehicle v_i can be expressed as follows:

$$E_{i,j}^l = \xi_i \cdot {f_{i,j}^l}^3 \cdot \tau_{i,j}^l \tag{3}$$

where ξ_i is the local computation energy consumption coefficient of medical vehicle v_i.

3.3.2. Offloading Computation

When a medical service task is offloaded to a vehicular edge server, the medical vehicle first uploads it to the target vehicular edge server. It is assumed that task transmission

is completed on the orthogonal channels. Let $p_{i,j,k}$ be the wireless transmission power of task $t_{i,j}$ generated by medical vehicle v_i to vehicular edge server s_k, with the maximum value $P_{i,j,k}^{max}$. Let σ^2 be the background noise power. We further define the channel gain between medical vehicle v_i and vehicular edge server s_k by $g_{i,k} = G_{i,k}^{-l}$, where $G_{i,k}^{-l}$ is the distance between medical vehicle v_i and vehicular edge server s_k, and $-l$ is the path loss index. The data transmission delay of the medical service task $t_{i,j}$ from medical vehicle v_i to vehicular edge server s_k can be expressed as follows:

$$\tau_{i,j,k}^{u} = \frac{x_{i,j}^{k} \cdot D_{i,j}}{R_{i,j,k}} \tag{4}$$

$$R_{i,j,k} = B\log\left(1 + \frac{p_{i,j,k} \cdot g_{i,k}}{\sigma^2}\right) \tag{5}$$

where $R_{i,j,k}$ is the data transmission rate of medical service task $t_{i,j}$ from medical vehicle v_i to vehicular edge server s_k, and B is the channel bandwidth.

On the other hand, we can calculate the computation execution delay of task $t_{i,j}$ on vehicular edge server s_k, expressed as follows:

$$\tau_{i,j,k}^{s} = \frac{c_{i,j} \cdot x_{i,j}^{k} \cdot D_{i,j}}{f_{i,j,k}^{s}} \tag{6}$$

where $f_{i,j,k}^{s}$ is the computing resources allocated by vehicular edge server s_k to task $t_{i,j}$ generated by medical vehicle v_i. It is assumed that computing resources of edge servers are allocated in a time-sharing manner.

The returned result from vehicular edge server to medical vehicle is generally much smaller than the uploaded data from the medical vehicle to the vehicular edge server. Therefore, the transmission time of the returned result can be ignored. Regarding task offloading, the overall delay is involved with data transmission delay and task execution delay in the vehicular edge server. Therefore, the overall computation delay of medical service task $t_{i,j}$ offloaded to vehicular edge server s_k can be expressed as follows:

$$\tau_{i,j,k}^{o} = \tau_{i,j,k}^{u} + \tau_{i,j,k}^{s} \tag{7}$$

Delay requirement guaranteeing is critical for medical vehicular services. The medical service tasks can be executed on the local medical vehicles or on vehicular edge servers. We expect to satisfy the delay requirement of tasks to improve the QoE of users. For medical vehicle v_i, the total completion time of its task $t_{i,j}$ can be expressed as follows:

$$\tau_{i,j}^{'} = (1 - x_{i,j}^{k}) \cdot \tau_{i,j}^{l} + x_{i,j}^{k} \cdot \tau_{i,j,k}^{o} \tag{8}$$

The overall energy consumption of the medical service task involves the transmission energy consumption from the medical vehicle to the vehicular edge server, and the offloading computation energy consumption of the vehicular edge server. The transmission energy consumption $E_{i,j,k}^{u}$ of medical service task $t_{i,j}$ from medical vehicle v_i to vehicular edge server s_k can be calculated as follows:

$$E_{i,j,k}^{u} = p_{i,j,k} \cdot \tau_{i,j,k}^{u} \tag{9}$$

Similarly, we can obtain the offloading computation energy consumption $E_{i,j}^{s}$ of vehicular edge server s_k processing the medical service task $t_{i,j}$, expressed as follows:

$$E^s_{i,j,k} = \zeta_k \cdot {f^s_{i,j,k}}^3 \cdot \tau^s_{i,j,k} \tag{10}$$

where ζ_k is the computation energy consumption coefficient of vehicular edge server s_k.

Thus, we can obtain the overall computation energy consumption of offloading medical service task $t_{i,j}$ as follows:

$$E^o_{i,j} = E^u_{i,j,k} + E^s_{i,j,k} \tag{11}$$

The total computation energy consumption of the task generated by medical vehicle v_i involves three parts, i.e., local computation energy consumption, transmission energy consumption from medical vehicle to vehicular edge server, as well as edge server computation energy consumption. Therefore, the total computation energy consumption of medical service task $t_{i,j}$ can be calculated as follows:

$$E_{i,j} = (1 - x^k_{i,j}) \cdot E^l_{i,j} + x^k_{i,j} \cdot E^o_{i,j,k} \tag{12}$$

4. Problem Formulation

In this section, we present the problem formulation for joint task offloading and resource allocation in MEC-enabled medical vehicular networks.

4.1. Optimization Objective

To improve QoE, we explore the optimal task offloading and resource allocation solution, to minimize the average task completion time and average energy consumption, while satisfying the resource requirements and QoS constraints. Thus, the problem of joint task offloading and resource allocation in MEC-enabled medical vehicular networks can be formulated as a multi-objective optimization problem ($\mathcal{P}$), expressed as follows:

$$\mathcal{P}: \begin{array}{ll} \min & \tau = \frac{\sum_{i,j} \tau'_{i,j}}{\sum_i N^t_i} \\ \min & E = \frac{\sum_{i,j} Ei,j}{\sum_i N^t_i} \end{array} \tag{13}$$

where N^t_i is the total number of medical service tasks generated by medical vehicle v_i.

4.2. Constraints

To perform task offloading and resource allocation, some constraints on the offloading decision, resource allocation, and QoS requirements should be satisfied.

Regarding the task offloading decision, we assume that each medical vehicular task is either offloaded to one vehicular edge server or executed on the medical vehicle locally. Thus, the following task offloading constraint should be satisfied:

$$x^k_{i,j} \in \{0, 1\}, \ \ \forall v_i \in V, \ \forall j \in N^t_i, \ \forall s_k \in S \tag{14}$$

Formula (15) ensures that when executing the task offloading, the wireless transmission power must be less than or equal to the predefined maximum threshold. Similarly, Formula (16) specifies the constraint of the CPU cycle frequency of each medical vehicle.

$$0 \leq p_{j,j,k} \leq P^{max}_{i,j,k}, \quad \forall v_i \in V, \ \forall j \in N^t_i, \ \forall s_k \in S \tag{15}$$

$$0 \leq f^l_{i,j} \leq f^{l,max}_{i,j}, \quad \forall v_i \in V, \ \forall j \in N^t_i \tag{16}$$

To provide efficient medical vehicular services, the total CPU resources demanded by all local computation tasks $t_{i,j}$ on medical vehicle v_i cannot exceed its available CPU resource capacity, expressed as follows:

$$\sum_{j} \in V \, f_{i,j}^{l} \leq c_{v}^{i}, \quad \forall v_i \in V \tag{17}$$

For each vehicular edge server s_k, the total CPU resource demands of all offloaded tasks $t_{i,j}$ should be smaller or equal to its CPU resource capacity, expressed as follows:

$$\sum_{i,j} f_{i,j,k}^{s} \leq c_{s}^{j}, \quad \forall s_k \in S \tag{18}$$

To guarantee medical vehicular service quality, the total computation delay of task $t_{i,j}$ cannot exceed its maximum tolerant delay requirement, expressed as follows:

$$\tau_{i,j}^{'} \leq \tau_{i,j}, \quad \forall v_i \in V, \, \forall j \in N_i^t \tag{19}$$

To solve the multi-objective optimization models, a large number of evolutionary algorithms have been proposed. Among them, the MOEAD algorithm has been widely used. Compared to the other evolutionary algorithms, the MOEAD algorithm has a smaller time complexity and can obtain the approximate solution of the Pareto front. The MOEAD algorithm decomposes the multi-objective problem into multiple single objective sub-problems, and uses the evolutionary algorithm to effectively determine the multiple non-dominated solutions. Therefore, we design a MOEAD-based task offloading and resource allocation (IMO) algorithm to solve the above model. Moreover, to obtain the optimal solution and speed up algorithm convergence, we design a greedy strategy-based population initialization algorithm.

5. IMO Solution

In this section, we present the detailed design of the proposed IMO algorithm, and then analyze the algorithm complexity.

5.1. Solution Encoding

In the IMO algorithm, each chromosome represents a solution of our proposed multi-objective optimization model. Each medical vehicular task corresponds to a task offloading decision strategy and a resource allocation decision strategy. Therefore, in our work, each chromosome is composed of two parts, i.e., computation location ($L_{i,j}$) of each medical vehicular task $t_{i,j}$ and resource allocation ratio ($r_{i,j}$). For medical vehicular task $t_{i,j}$, it can be computed locally by medical vehicle v_i or vehicular edge server s_k. On the other hand, the resource allocation ratio refers to the ratio of the total amount of resources allocated to medical vehicular task $t_{i,j}$ to the total amount of resources own by medical vehicle v_i or vehicular edge server s_k. If the resource requirements and QoS constraints of task $t_{i,j}$ cannot be satisfied by the medical vehicle or vehicular edge server, the service request of task $t_{i,j}$ will be rejected. Therefore, we set different values to $L_{i,j}$ to identify the computation location of medical vehicular task $t_{i,j}$. If medical vehicular task $t_{i,j}$ is processed locally by medical vehicle $L_{i,j} = 0$. If medical vehicular task $t_{i,j}$ is offloaded to vehicular edge server s_k, $L_{i,j} = k \in (1, |S|)$. Figure 2 depicts a solution encoding example of task offloading and resource allocation. As illustrated in Figure 2, the numbers in the red area indicate the offloading location of medical vehicular tasks. And the numbers in the green area represent the ratio of resources allocated by the devices (i.e., local medical vehicles or vehicular edge servers) to medical vehicular tasks. For example, 1 indicates that the first medical vehicular task is offloaded to the first vehicular edge server; 0.27 indicates that the first vehicular edge server allocates 27% of its computing resource to process the medical vehicular task; 0 in the red area indicates that the medical vehicular task is processed on the local medical

vehicle; and 0.33 indicates that the local medical vehicle allocates 33% of its computing resource to process the medical vehicular task.

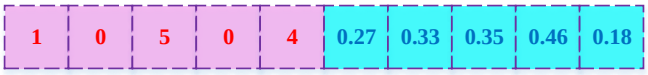


Figure 2. Solution encoding of the IMO algorithm.

5.2. Greedy Strategy-Based Population Initialization

The traditional MOEAD algorithm consists of four stages, i.e., population initialization, reproduction operation, evolution operation, and optimal solution selection. Population initialization has a great impact on the algorithm performance. In the MOEAD algorithm, the population is initialized randomly in order to obtain the diversity of the solution. However, in the realistic scenarios, the random population initialization method may result in serious deterioration of the optimization performance. To obtain the optimal solution and speed up algorithm convergence, a greedy strategy is used. Therefore, in this work, we design a greedy strategy-based population initialization (InGP) algorithm to improve the quality of solutions. The key of the InGP algorithm is to determine the task offloading and resource allocation solution based on the smallest task completion time and energy consumption. When determining the initial offloading location of a medical vehicular task, we compare the cost values of the local medical vehicle and a random edge server. If the cost value of the vehicular edge server is smaller than that of the medical vehicle, the task will be offloaded to that vehicular edge server. Otherwise, the task will be processed locally by the medical vehicle. The InGP algorithm keeps the diversity of the solution and, at the same time, improves the quality of the initial solution.

The main framework of the proposed InGP algorithm is shown in Algorithm 1. As illustrated in Lines 3–4, for each medical vehicular task $t_{i,j}$, we first randomly select a vehicular edge server, and then check whether it satisfies the resource demand of task $t_{i,j}$. If the resource demand cannot be satisfied by the vehicular edge server, we continue to search the other vehicular edge servers. If neither the vehicular edge servers or local medical vehicle can satisfy the resource demand, the service request of task $t_{i,j}$ will be rejected due to resource limitation. Otherwise, as illustrated in 5–11, we calculate the task completion time of $t_{i,j}$ on local medical vehicle v_i by Equation (2), and vehicular edge server s_k by Equation (7), respectively. If the task completion time on the vehicular edge server is smaller than that on the medical vehicle, medical vehicular task $t_{i,j}$ will be offloaded to vehicular edge server s_k. Otherwise, it will be processed locally by medical vehicle v_i.

5.3. Reproduction Operation

After generating the initial population, we execute the reproduction operation to generate the offspring populations. Algorithm 2 presents the main workflow of reproduction operation. The reproduction operation mainly consists of two stages, i.e., crossover operation and mutation operation. As illustrated in Lines 1–15 in Algorithm 2, the crossover operation aims to produce the offspring population with the smaller cost value. Specifically, we first generate a random number r and compare it with the set crossover probability P_c. If $r < P_c$, offspring individuals op_1 and op_2 will inherit the genes from parent individuals p_1 and p_2, respectively. Otherwise, offspring individuals op_1 and op_2 will inherit the genes from parent individuals p_2 and p_1, respectively. Then, we calculate the cost values of new offspring individuals of_1 and of_2, respectively. In this work, our optimization objective is to minimize the average completion time and average energy consumption. Therefore, we define the cost function F as follows:

$$F = \alpha \cdot \tau^* + \beta \cdot E^* \tag{20}$$

$$\alpha \in (0,1) \tag{21}$$

$$\beta \in (0,1) \tag{22}$$

$$\alpha + \beta = 1 \tag{23}$$

where τ^* and E^* are the normalized values of τ and E, and α and β are weighted coefficients.

If the cost values of two offspring individuals are bigger than those of two parents, offspring individuals op_1 and op_2 will inherit the genes from two parents p_1 and p_2, respectively, illustrated in Lines 6–12.

After executing the crossover operation, we will execute the mutation operation on the new offspring population OP, illustrated in Lines 16–24. For each gene in the chromosome, i.e., offloading location and resource allocation ratio of each medical vehicular task, we decide whether it is altered based on mutation probability P_m, illustrated in Lines 19–22. As illustrated in Line 20, we will offload the medical vehicular task to vehicular edge server k with the smallest task completion time to reduce the delay and improve the QoE of users.

Algorithm 1 InGP algorithm

Input: Medical vehicular network G;
Set of tasks T;
Population size N_p;
Output: Initial solution P;
1: **for** $p = 1 : N_p$ **do**
2: **for** $t_{i,j} \in T$ **do**
3: $k = GenerateRandom(|S|)$;
4: **if** Server s_k meets the constraints (14)–(19) **then**
5: Calculate $\tau^l_{i,j}$ by Equation (2);
6: Calculate $\tau^o_{i,j}$ by Equation (7);
7: **if** $\tau^o_{i,j} \leq \tau^l_{i,j}$ **then**
8: $t_{i,j}$ is processed by s_k;
9: **else**
10: $t_{i,j}$ is processed by v_i;
11: **end if**
12: **else**
13: **if** None of devices meet the constraints (14)–(19) **then**
14: $t_{i,j}$ is rejected;
15: **else**
16: Goto Step 3;
17: **end if**
18: **end if**
19: Record the decision of $t_{i,j}$ into P;
20: **end for**
21: **end for**

5.4. Overall Framework

The overall workflow of the IMO algorithm is illustrated in Algorithm 3. It takes MEC-enabled medical vehicular network G, the set of tasks T, population size N_p, neighborhood size N_g, weight vector λ, the maximum iteration times N_{iter}, population crossover probability P_c, and mutation probability P_m as the algorithm input, and outputs the optimal task offloading and resource allocation solution P. As illustrated in Lines 1–6, we first randomly initialize the weight vector λ, neighborhood weight vector and reference point, and produce the initial population by executing Algorithm 1. As illustrated in Lines 7–12, we execute the crossover and mutation operations to produce new offspring population, and evaluate the cost value of each chromosome individual in the current population. As illustrated in Line 10, based on the weight vector and the neighborhood set, we select a better solution to update the original solution to improve the quality of solutions. When it reaches the set maximum times of iterations, the reproduction and evolution operations

end. Finally, we determine the optimal solution from the current population, illustrated in Lines 12–13.

Algorithm 2 ExReproduct algorithm.

Input: Original population P;
Population size N_p;
Crossover probability P_c;
Mutation probability P_m;
Output: New population $P1$;
1: **for** $i = 1 : N_p/2$ **do**
2: Select (p_1, p_2) from P;
3: **for** $j = 1 : |p_i|$ **do**
4: $r = GenerateRandom(0, 1)$;
5: **if** $r < P_c$ **then**
6: $(p_{1,j}, p_{2,j}) \longrightarrow (op_{2,j}, op_{1,j})$;
7: Calculate cost value (f_1, f_2, of_1, of_2);
8: **if** $of_1 > f_1$ and $of_2 > f_2$ **then**
9: $(p_{1,j}, p_{2,j}) \longrightarrow (OP_{2*i,j}, OP_{2*i-1,j})$;
10: **else**
11: $(p_{1,j}, p_{2,j}) \longrightarrow (OP_{2*i-1,j}, OP_{2*i,j})$;
12: **end if**
13: **end if**
14: **end for**
15: **end for**
16: **for** $i = 1 : N_p$ **do**
17: **for** $j = 1 : |p_i|$ **do**
18: $r = GenerateRandom(0, 1)$;
19: **if** $r < P_c$ **then**
20: Select server k with the smallest completion time;
21: Update OP;
22: **end if**
23: **end for**
24: **end for**
25: $OP \longrightarrow P1$;

Algorithm 3 IMO algorithm

Input: Medical vehicular network G;
Set of tasks T;
Population size N_p;
Neighborhood N_g;
Weight vector λ;
Maximum iteration times N_{iter};
Crossover probability P_c;
Mutation probability P_m;
Output: Solution P;
1: Initialize weight vector λ;
2: **for** $i = 1 : N_p$ **do**
3: Get N_g closest weight vectors for λ_i;
4: **end for**
5: $P0 = InGP(G, T, N_P)$;
6: Initialize reference point z;
7: **for** $i = 1 : N_{iter}$ **do**
8: $P1 = ExeReproduct(P0, N_p, P_c, P_m)$;
9: $F = CalculateCostValue(P1, N_p)$;
10: $P2 = ExecuteEvolution(P1, F, N_P)$;
11: **end for**
12: $P^{'} = GetApproximationSolution(P2, N_P)$;
13: $P = SelectOptimalSolution(P^{'})$;

5.5. Complexity Analysis

It is assumed that the total number of medical vehicular tasks is N_T, and the total number of vehicular edge servers is N_S. The InGP algorithm needs to take $O(N_p \cdot N_T \cdot N_S)$ time to perform the population initialization operation. The ExReproduct algorithm needs to take $O(N_p \cdot N_T)$ time to execute the crossover and mutation operations on the population P. Therefore, the IMO algorithm needs to take $O(N_p \cdot N_T \cdot N_S \cdot N_{der})$ to determine the optimal task offloading and resource allocation solution.

6. Simulation Experiments

We first present the simulation setups for simulation experiments, and then describe the simulation result analysis.

6.1. Simulation Setups

The simulation experiments are implemented by using Python 3.1, running on the PC equipped with 2.8 GHz, Intel i7 CPU, and 32 GB RAM.

Similar to [18,22], we set the simulation parameters, shown in Table 2. We vary the number of medical vehicles from 2 to 24, and the number of edge servers from 2 to 8. The number of tasks generated by medical vehicles varies from 4 to 24. We set the size of population to 100 and the maximum times of iterations to 100. During the population evolutionary process, the crossover probability and mutation probability are set to 0.5 and 0.05. The weight coefficients of two optimization objectives are set to 0.6 and 0.4.

Table 2. Simulation parameter settings.

Parameter	Value
Number of medical vehicles	2∼24
Number of edge servers	2∼8
CPU of medical vehicle	$[0.5\ \ 2\ ] \times 10^9$ Hz
CPU of edge server	$[2\ \ 2.2] \times 10^{10}$Hz
Number of medical tasks	4∼24
Data size of medical task	$[0.1\ \ 10] \times 10^6$Byte
Maximum delay of task	[0.001 0.1]s
Path loss index	1×10^{-3}
Background noise power	1×10^{-9}
Transmission power of task	$[1\ 10] \times 10^{-2}$ W
Transmission channel bandwidth	1×10^8
Vehicle energy consumption coefficient	$[100\ 100 + 0.1 * rand(M)] \times 10^{-30}$
Server energy consumption coefficient	$[100\ 100 + 0.1 * rand(S)] \times 10^{-30}$
Population size	100
Maximum iteration times	100
Crossover probability	0.5
Mutation probability	0.05

To comprehensively evaluate the performance results of the IMO algorithm, local computing based on the task offloading and resource allocation (LOC) algorithm [17], server computing-based task offloading and resource allocation (OFC) algorithm [17], and random offloading and resource allocation (RAN) algorithm [21], the NSGA-II based task offloading and resource allocation (NSG) algorithm [41] are selected as comparison algorithms as follows.

- LOC algorithm: All the tasks generated by the medical vehicles are processed on the local medical vehicles, rather than offloaded to the vehicular edge servers.
- OFC algorithm: All the tasks generated by the medical vehicles are offloaded to the vehicular edge servers, rather than processed locally by the medical vehicles.

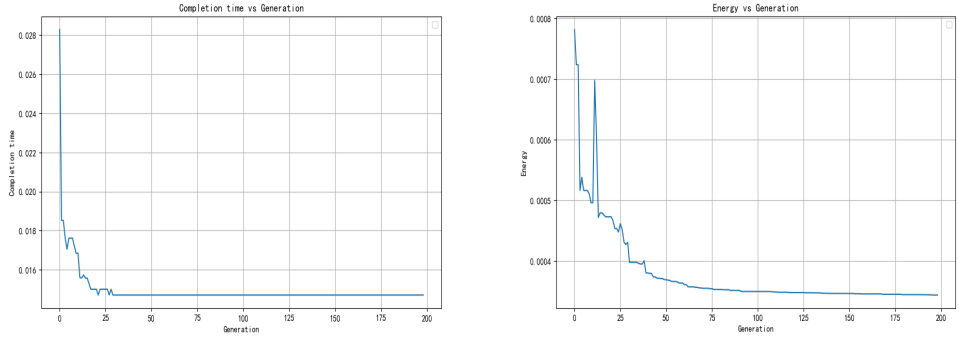
- RAN algorithm: All the tasks generated by the medical vehicles are randomly processed on the local medical vehicles or offloaded to the vehicular edge servers.
- NSG algorithm: All the tasks generated by the medical vehicles are determined based on the NSGA-II algorithm to be processed on the local medical vehicles or offloaded to the vehicular edge servers.

6.2. Simulation Results

To evaluate the performance of IMO algorithm comprehensively, we first analyze the convergence performance, and then verify the impacts of the number of medical vehicles, vehicular edge servers and medical vehicular tasks on the algorithm performance, respectively.

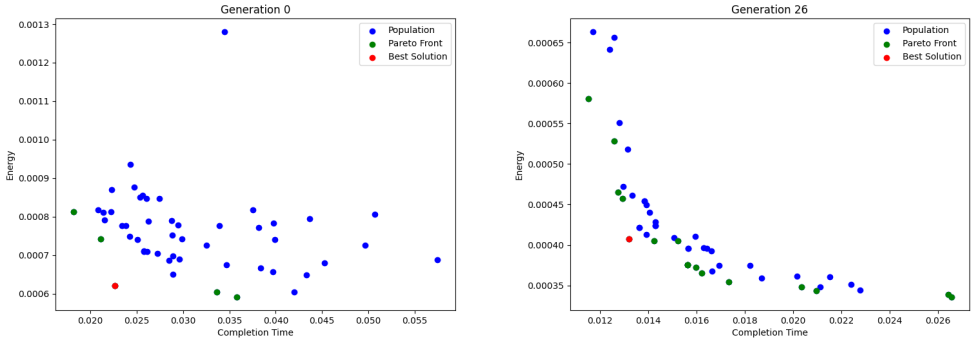
6.2.1. Convergence Evaluation

Figure 3 depicts the algorithm convergence performance results on the average completion time and average energy consumption, respectively. From Figure 3, we can observe that as the population generation increases, the average task completion time and average energy consumption first decrease and then tend to stabilize. In particular, when the population generation is about 98, the performance results of the IMO algorithm in terms of the average completion time and average energy consumption are almost stable. Figure 4 depicts the convergence performance of the proposed IMO algorithm under different generations. We can observe from Figure 4 that with the increases in the population generations, the solution obtained by the IMO algorithm is closer to the optimal solution.



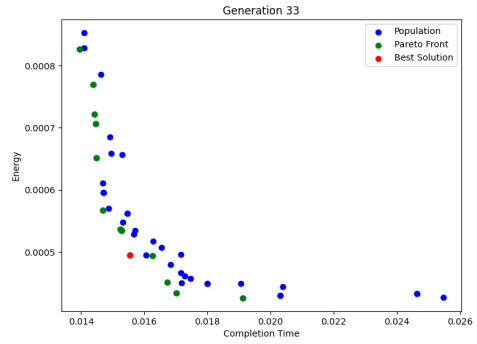
(**a**) Average completion time vs. Generation (**b**) Average energy consumption vs. Generation

Figure 3. Convergence performance.

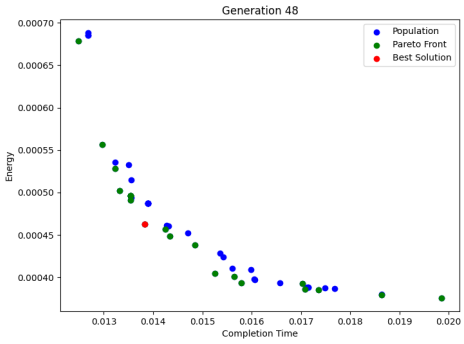


(**a**) Generation 0 (**b**) Generation 26

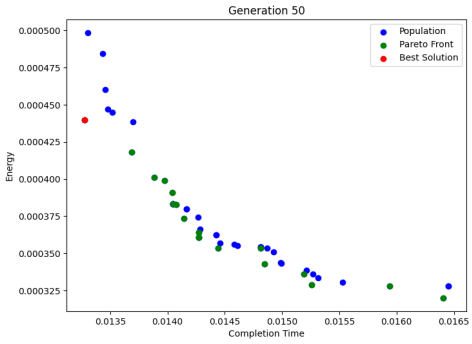
Figure 4. *Cont.*



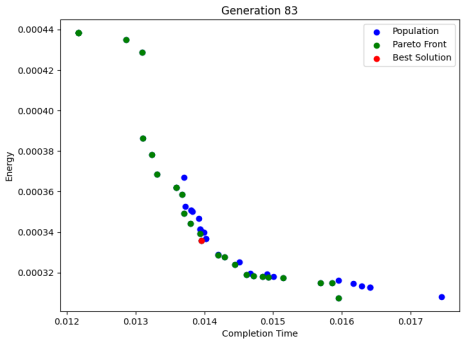
(**c**) Generation 33



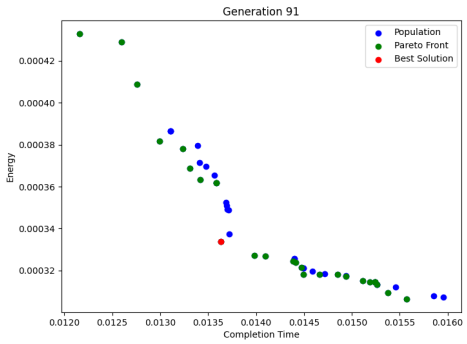
(**d**) Generation 49



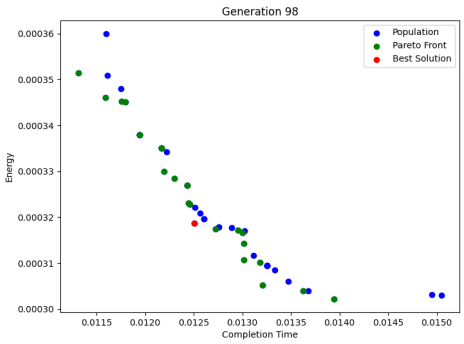
(**e**) Generation 50



(**f**) Generation 83



(**g**) Generation 91



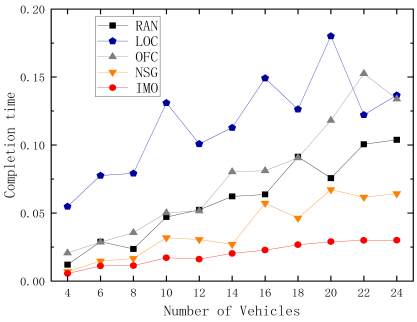
(**h**) Generation 98

Figure 4. Convergence performance under different generations.

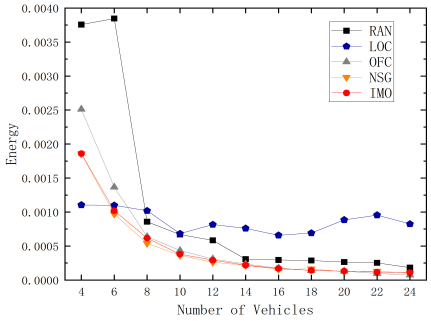
6.2.2. Effect of Number of Medical Vehicles

Figure 5 depicts the performance comparison results by varying the numbers of medical vehicles. From Figure 5a, we can observe that as the number of medical vehicles increases, the proposed IMO algorithm obtains the smallest average task completion time among five algorithms. This is because the proposed IMO algorithm uses the greedy strategy to obtain the better initial solution. And it selects medical vehicles or edge servers, which have a small task completion time and energy consumption, to perform the task computation. From Figure 5b, we can observe that the IMO algorithm obtains smaller average energy consumption. However, when the number of medical vehicles is 22 or 24, the energy consumption of the IMO algorithm is slightly bigger than that of the OFC algorithm. This is because the OFC algorithm offloads all tasks to the edge servers. Unlike the OFC algorithm, the IMO algorithm determines the offloading solution of each task

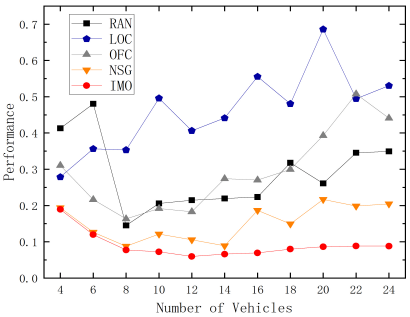
based on the overall performance. Moreover, the randomness of the IMO algorithm may affect the task offloading decision. Figure 5c shows the superiority of the IMO algorithm under different numbers of medical vehicles. The proposed IMO algorithm can obtain the best tradeoff between the average task completion time and average energy consumption among the five algorithms. This is because our proposed IMO algorithm attempts to balance the tradeoff between the average task completion time and average energy consumption.



(**a**) Average completion time



(**b**) Average energy consumption

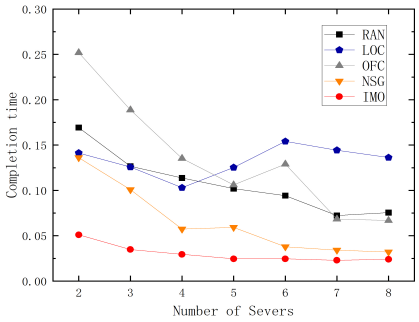


(**c**) Overall performance

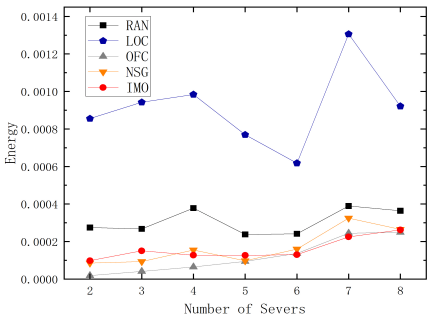
Figure 5. Performance comparison under different numbers of medical vehicles.

6.2.3. Effect of Number of Vehicular Edge Servers

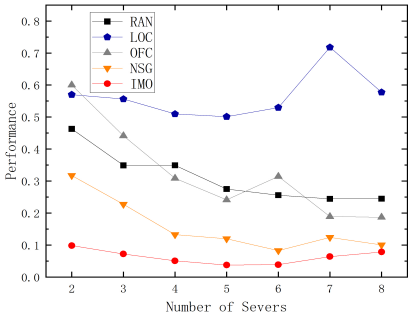
Figure 6 depicts the performance results of five task-offloading and resource allocation algorithms by varying the number of vehicular edge servers. We can observe from Figure 6 that the proposed IMO algorithm can obtain the smallest average completion time and smaller average energy consumption. The proposed IMO algorithm can also achieve the best tradeoff between average completion time and average energy consumption. This is because, different from comparison algorithms, the proposed IMO algorithm tries to reduce the average task completion time and average energy consumption while balancing the tradeoff. Moreover, as the number of vehicular edge servers increases, the average completion time of these algorithms becomes small. This is because as the number of vehicular edge servers increases, the total resource capacity becomes big. Each task can be allocated to more resources to perform the task computation. Consequently, the average completion time becomes small.



(**a**) Average completion time



(**b**) Average energy consumption



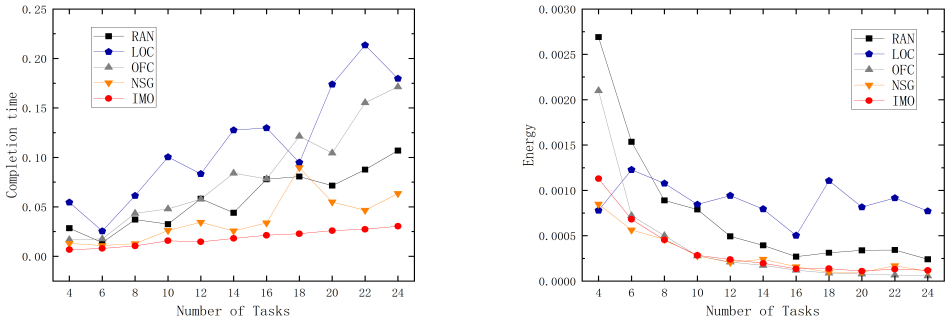
(**c**) Overall performance

Figure 6. Performance comparison under different numbers of vehicular edge servers.

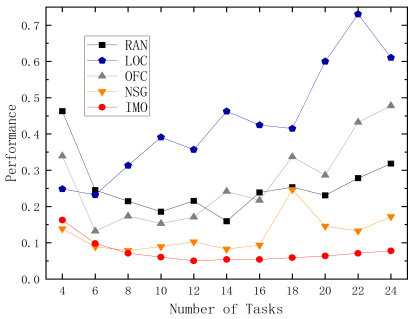
6.2.4. Effect of Number of Tasks

Figure 7 depicts the performance effects of the number of tasks on the average completion time, average energy consumption, and overall performance. From Figure 7, as the total number of tasks increases, the average completion time of the five task-offloading and resource allocation algorithms becomes big. This is because the increase in tasks makes resource contention more competitive. The resource allocated by medical vehicles or vehicular edge servers to each task becomes small. As a result, executing each task needs to take more time. We can also observe from Figure 7 that the proposed IMO algorithm can obtain the smallest average completion time and achieve the best tradeoff between the task completion time and energy consumption.

In summary, we can observe that under different numbers of vehicles, edge servers and tasks, the proposed IMO algorithm can obtain the smallest average completion time, and achieve the best tradeoff between the task completion time and energy consumption. By measurement, compared to comparison algorithms, our algorithm can at least save 5% of average task completion time. We can draw a conclusion that the proposed IMO algorithm has good robustness under different conditions.



(**a**) Average completion time (**b**) Average energy consumption



(**c**) Overall performance

Figure 7. Performance comparison under different numbers of tasks.

7. Conclusions

In this work, we study the joint optimization problem of task offloading and resource allocation in MEC-enabled medical vehicular networks in order to improve service performance and enhance QoE of users. We first establish a multi-objective optimization model, with the target of average completion time minimization and average energy consumption minimization. We further design a novel MOEAD-based task offloading and resource allocation algorithm, IMO, to solve the above optimization model. To speed up algorithm convergence and obtain the optimal solution, we design a novel greedy strategy-based population initialization algorithm. Final simulations demonstrate the effectiveness and efficiency of the IMO algorithm. Compared to the comparison algorithm, the IMO algorithm at least can reduce 5% of average task completion time.

Our proposed IMO algorithm can significantly reduce the average task completion time and achieve a better tradeoff between the task completion time and energy consumption. It is helpful to provide a fast response for patients in medical vehicles. However, the diversity of users' service requests and high dynamics of medical vehicular network are not considered in this work, such as the priority of critical patients, data security, network reliability, task success rate, and network latency fluctuations. In future works, we will design an online adaptive task offloading and resource allocation mechanism in large-scale medical vehicular networks to improve the service performance and satisfy the diversified service demands of users by considering the above important factors in medical situations.

Author Contributions: Conceptualization, C.Z., G.C., H.Y. and X.W.; methodology, C.Z., S.L., G.C. and F.L.; formal analysis, C.Z., S.L. and X.W; writing—original draft preparation, C.Z., S.L., H.Y. and G.C.; writing—review and editing, C.Z., S.L., H.Y. and F.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Natural Science Foundation of Shandong Province of China under Grants No. ZR2021QF086, ZR2022MF231 and ZR2022MF254.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: All authors declare no conflicts of interest.

References

1. Fan, W.; Liu, J.; Hua, M.; Liu, Y. Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles. *IEEE Trans. Veh. Technol.* **2022**, *71*, 5314–5330. [CrossRef]
2. He, Q.; Feng, Z.; Chen, Z.; Nan, T.; Li, K.; Shen, H.; Yu, K.; Wang, X. Low-cost data offloading strategy with deep reinforcement learning for smart healthcare system. *IEEE Trans. Serv. Comput.* **2024**. [CrossRef]
3. Akhlaqi, M.Y.; Hanapi, Z.B.M. Task offloading paradigm in mobile edge computing-current issues, adopted approaches, and future directions. *J. Netw. Comput. Appl.* **2023**, *212*, 103568. [CrossRef]
4. Hamdi, A.M.A.; Hussain, F.K.; Hussain, O.K. Task offloading in vehicular fog computing: State-of-the-art and open issues. *Future Gener. Comput. Syst.* **2022**, *133*, 201–212. [CrossRef]
5. Nan, Z.; Zhou, S.; Jia, Y.; Niu, Z. Joint task offloading and resource allocation for vehicular edge computing with result feedback delay. *IEEE Trans. Wirel. Commun.* **2023**, *22*, 6547–6561. [CrossRef]
6. Qi, Y.; Zhou, Y.; Liu, Y.F.; Liu, L.; Pan, Z. Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing. *IEEE Internet Things J.* **2021**, *8*, 17762–17777. [CrossRef]
7. Yan, M.; Xiong, R.; Wang, Y.; Li, C. Edge computing task offloading optimization for a UAV-assisted Internet of vehicles via deep reinforcement learning. *IEEE Trans. Veh. Technol.* **2024**, *73*, 5647–5658. [CrossRef]
8. Fan, W.; Hua, M.; Zhang, Y.; Su, Y.; Li, X.; Tang, B.; Wu, F.; Liu, Y. Game-based task offloading and resource allocation for vehicular edge computing with edge-edge cooperation. *IEEE Trans. Veh. Technol.* **2023**, *72*, 7857–7870. [CrossRef]
9. Cao, D.; Wu, M.; Gu, N.; Sherratt, R.S.; Ghosh, U.; Sharma, P.K. Joint optimization of computation offloading and resource allocation considering task prioritization in ISAC-assisted vehicular network. *IEEE Internet Things J.* **2024**, *11*, 29523–29532. [CrossRef]
10. Wang, S.; Li, J.; Wu, G.; Chen, H.; Sun, S. Joint optimization of task offloading and resource allocation based on differential privacy in vehicular edge computing. *IEEE Trans. Comput. Soc. Syst.* **2021**, *9*, 109–119. [CrossRef]
11. Fan, W.; Su, Y.; Liu, J.; Li, S.; Huang, W.; Wu, F.; Liu, Y. Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 4277–4292. [CrossRef]
12. Sun, Y.; Guo, X.; Song, J.; Zhou, S.; Jiang, Z.; Liu, X. Adaptive learning-based task offloading for vehicular edge computing systems. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3061–3074. [CrossRef]
13. Zhang, J.; Guo, H.; Liu, J.; Zhang, Y. Task offloading in vehicular edge computing networks: A load-balancing solution. *IEEE Trans. Veh. Technol.* **2019**, *69*, 2092–2104. [CrossRef]
14. Raza, S.; Liu, W.; Ahmed, M.; Anwar, M.R.; Mirza, M.R.; Sun, Q.; Wang, S. An efficient task offloading scheme in vehicular edge computing. *J. Cloud Comput.* **2020**, *9*, 28. [CrossRef]
15. Liu, L.; Zhao, M.; Yu, M.; Jan, M.A.; Lan, D.; Taherkordi, A. Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 2169–2182. [CrossRef]
16. Wu, Q.; Wang, S.; Ge, H.; Fan, Q.; Letaieff, K.B. Delay-sensitive task offloading in vehicular fog computing-assisted platoons. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 2012–2026. [CrossRef]
17. Yin, L.; Luo, J.; Qiu, C.; Wang, C.; Qiao, Y. Joint task offloading and resources allocation for hybrid vehicle edge computing systems. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 10355–10368. [CrossRef]
18. Qin, P.; Fu, Y.; Tang, G.; Zhao, X.; Geng, S. Learning based energy efficient task offloading for vehicular collaborative edge computing. *IEEE Trans. Veh. Technol.* **2022**, *71*, 8398–8413. [CrossRef]
19. Tan, K.; Feng, L.; Dán, G.; Törngren, M. Decentralized convex optimization for joint task offloading and resource allocation of vehicular edge computing systems. *IEEE Trans. Veh. Technol.* **2022**, *71*, 13226–13241. [CrossRef]
20. Dai, X.; Xiao, Z.; Jiang, H.; Liu, J.C.S. UAV-assisted task offloading in vehicular edge computing networks. *IEEE Trans. Mob. Comput.* **2023**, *23*, 2520–2534. [CrossRef]
21. Zhao, L.; Zhang, E.; Wan, S.; Hawbani, A.; AI-dubai, A.Y.; Min, G. MESON: A mobility-aware dependent task offloading scheme for urban vehicular edge computing. *IEEE Trans. Mob. Comput.* **2023**, *23*, 4259–4272. [CrossRef]
22. Cao, B.; Li, Z.; Liu, X.; Lv, Z.; He, H. Mobility-aware multiobjective task offloading for vehicular edge computing in digital twin environment. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 3046–3055. [CrossRef]
23. Chen, Y.; Zhao, F.; Chen, X.; Wu, Y. Efficient multi-vehicle task offloading for mobile edge computing in 6G networks. *IEEE Trans. Veh. Technol.* **2021**, *71*, 4584–4595. [CrossRef]

24. Xue, J.; Hu, Q.; An, Y.; Wang, L. Joint task offloading and resource allocation in vehicle-assisted multi-access edge computing. *Comput. Commun.* **2021**, *177*, 77–85. [CrossRef]
25. Sun, Y.; Wu, Z.; Meng, K.; Zheng, Y. Vehicular task offloading and job scheduling method based on cloud-edge computing. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 14651–14662. [CrossRef]
26. Bute, M.S.; Fan, P.; Zhang, L.; Abbas, F. An efficient distributed task offloading scheme for vehicular edge computing networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13149–13161. [CrossRef]
27. Guo, H.; Liu, J.; Ren, J.; Zhang, Y. Intelligent task offloading in vehicular edge computing networks. *IEEE Wirel. Commun.* **2020**, *27*, 126–132. [CrossRef]
28. Li, Z.; Yang, C.; Huang, X.; Zeng, W.; Xie, S. CoOR: Collaborative task offloading and service caching replacement for vehicular edge computing networks. *IEEE Trans. Veh. Technol.* **2023**, *72*, 9676–9681. [CrossRef]
29. Wei, Z.; Li, B.; Zhang, R.; Cheng, X.; Yang, L. Many-to-many task offloading in vehicular fog computing: A multi-agent deep reinforcement learning approach. *IEEE Trans. Mob. Comput.* **2023**, *23*, 2107–2122. [CrossRef]
30. Gao, H.; Wang, X.; Wei, W.; AI-Dulaimi, A.; Xu, Y. Com-DDPG: Task offloading based on multiagent reinforcement learning for information-communication-enhanced mobile edge computing in the internet of vehicles. *IEEE Trans. Veh. Technol.* **2024**, *73*, 348–361. [CrossRef]
31. Xu, X.; Liu, K.; Dai, P.; Jin, F.; Ren, H.; Zhan, C.; Guo, S. Joint task offloading and resource optimization in noma-based vehicular edge computing: A game-theoretic DRL approach. *J. Syst. Archit.* **2023**, *134*, 102780. [CrossRef]
32. Wu, Q.; Wang, W.; Fan, P.; Fan, Q.; Zhu, H.; Letaief, K.B. Cooperative edge caching based on elastic federated and multi-agent deep re-inforcement learning in next-generation networks. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 4179–4196. [CrossRef]
33. Zhou, Z.; Wang, Z.; Yu, H.; Liao, H.; Mumtaz, S.; Oliveira, L. Learning-based URLLC-aware task offloading for internet of health things. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 396–410. [CrossRef]
34. Dong, C.; Sun, Y.; Shafiq, M.; Hu, N.; Liu, Y.; Tian, Z. Optimizing mobility-aware task offloading in smart healthcare for internet of medical things through multi-agent reinforcement learning. *IEEE Internet Things J.* **2024**, *11*, 13677–13691. [CrossRef]
35. Ren, J.; Li, J.; Liu, H.; Qin, T. Task offloading strategy with emergency handling and blockchain security in SDN-empowered and fog-assisted healthcare IoT. *Tsinghua Sci. Technol.* **2021**, *27*, 760–776. [CrossRef]
36. He, Q.; Feng, Z.; Fang, H.; Wang, X.; Zhao, L.; Yao, Y. A blockchain-based scheme for secure data offloading in healthcare with deep reinforcement learning. *IEEE/ACM Trans. Netw.* **2023**, *32*, 65–80. [CrossRef]
37. Wang, Y.; Wu, H.; Jhaveri, R.H.; Djenouri, Y. DRL-based URLLC-constraint and energy-efficient task offloading for Internet of Health Things. *IEEE J. Biomed. Health Inform.* **2024**, *28*, 3305–3316. [CrossRef] [PubMed]
38. Ren, J.; Qin, T. Decentralized blockchain-based and trust-aware task offloading strategy for Healthcare IoT. *IEEE Internet Things J.* **2023**, *11*, 829–847. [CrossRef]
39. Gao, J.; Xu, X.; Qi, L.; Dou, W.; Xia, X.; Zhou, X. Distributed computation offloading and power control for UAV-enabled internet of medical things. *ACM Trans. Internet Technol.* **2024**. [CrossRef]
40. Wang, X.; Zhao, Y.; Tang, L.; Yao, X. MOEA/D with Spatial-Temporal Topological Tensor Prediction for Evolutionary Dynamic Multiobjective Optimization. *IEEE Trans. Evol. Comput.* **2024**. [CrossRef]
41. Mousavi, S.; Mood, S.E.; Souri, A.; Javidi, M.M. Directed search: A new operator in NSGA-II for task scheduling in IoT based on cloud-fog computing. *IEEE Trans. Cloud Comput.* **2022**, *11*, 2144–2157. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.