electronics

Article

Computational Resources Allocation and Vehicular Application Offloading in VEC Networks

Fan Gu [1], Xiaoying Yang [2], Xianwei Li [3,*] and Haiquan Deng [4]

[1] Department of Electronic Commerce, Anhui Institute of International Business, Hefei 230000, China; gufan7811@163.com
[2] School of Information Engineering, Suzhou University, Suzhou 234000, China; yangxiaoying@ahszu.edu.cn
[3] School of Computer and Information Engineering, Bengbu University, Bengbu 233000, China
[4] China Mobile Group Hunan Company Limited Chenzhou Branch, Chenzhou 423000, China; dhq231@139.com
* Correspondence: lixianwei163@163.com

Abstract: With the advances in wireless communications and the Internet of Things (IoT), various vehicular applications such as image-aided navigation and autonomous driving are emerging. These vehicular applications require a significant number of computation resources and a lower processing delay. However, these resource-limited and power-constrained vehicles may not meet the requirements of processing these vehicular applications. By offloading these vehicular applications to the edge cloud, vehicular edge computing (VEC) is deemed a novel paradigm for improving vehicular performance. However, how to optimize the allocation of computation resources of both vehicles and VEC servers to reduce the energy and delay is a challenging issue when deploying the VEC systems. In this article, we try to address this issue and propose a vehicular application offloading and computational resources allocation strategy. We formulate an optimization problem and present an efficient offloading scheme for vehicular applications. Extensive simulation results are offered to analyze the performances of the proposed scheme. In comparison with the benchmark schemes, the proposed scheme can outperform them in terms of computation cost.

Keywords: IoT; vehicular applications; VEC; MINP

check for updates

Citation: Gu, F.; Yang, X.; Li, X.; Deng, H. Computational Resources Allocation and Vehicular Application Offloading in VEC Networks. *Electronics* **2022**, *11*, 2130. https://doi.org/10.3390/electronics11142130

Academic Editors: Shinichi Yamagiwa and Taeshik Shon

Received: 24 May 2022
Accepted: 6 July 2022
Published: 7 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 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

In recent years, due to the developments of the communication protocols [1] as well as the advancements in the Internet of Things (IoT) and intelligent transportation system (ITS) [2], varieties of applications such as autonomous driving and image-aided navigation are emerging [3]. These applications covering the aspects of driving safety and information entertainment demand a lot of computation resources and have strict requirements for processing time. Vehicles are equipped with computing resources and sensors to handle the generated data of these applications [4]. However, due to the limited physical spaces, the local resources that are provided by the vehicles can not satisfy the Quality of Service (QoS) requirements of the vehicle applications [3,5].

To address the above issue, vehicular edge computing (VEC) is deemed a novel paradigm and technology in the 5G networks [3,6,7]. Vehicular cloud computing (VCC) was introduced to provide computing resources for vehicles to reduce power consumption and improve service performance. However, one challenge that the VCC has to face is the high latency, which makes the VCC unsuitable for the delay-sensitive vehicle applications [8]. In VEC networks, the computational resources are deployed at the roadside units, which are much closer to the vehicles. Therefore, VEC can migrate the computing resources to the network edge, which reduces the transmission delay and relieves the computing pressure on vehicles [3]. Vehicle applications can benefit a lot from the advantages of VEC. Thus, a safe and efficient transportation system can be provided [9].

Due to the development of intelligent transportation systems, the study of resource allocation and vehicular application offloading in the VEC networks has attracted a significant amount of attention, such as [3,6,7,9]. Nevertheless, jointly allocating the computing resources of both vehicles and edge servers is not thoroughly investigated in the current works.

In this paper, a joint study of allocating computing resources of vehicles and a VEC server for vehicular application offloading in VEC networks is investigated. Each vehicle user can decide its vehicular application offloading strategy by judging the computation cost caused in the local vehicles and that of the VEC server. The objective is to minimize the computation cost of all vehicles, which is calculated by the processing time of vehicular applications and the consumed energy of vehicles. The studied problem is formulated as a mixed-integer nonlinear programming (MINP) optimization problem, which is known as an NP-hard problem and is not convex. We design an efficient algorithm that can optimally solve the formulated problem.

In summary, we have the following contributions.

- We study vehicular application offloading in a VEC network system by jointly allocating computational resources of vehicles and the VEC servers. We try to minimize the consumed energy and time cost of executing the vehicle applications. We formulated the studied problem as an optimization problem.
- By analyzing the structure of the problem, we know that it is MINP and well-known as non-convex, which is NP-hard. The objective problem is solved by decomposing it into three subproblems. The optimal solution for each subproblem is obtained.
- Extensive simulation results are provided to prove that the proposed offloading strategy achieves better performances than the three benchmark algorithms.

The remaining study of this work is structured as follows. An overview of the vehicular application offloading and computational resources allocation in VEC networks is presented in Section 2. The system model and problem formulation are given in Section 3. The solution approach to the formulated problem is presented in Section 4. Simulation results are shown in Section 5. Section 6 concludes this work and provides some prospective aspects of research.

2. Related Work

Extensive research has been conducted on resource allocation and vehicular application offloading in VEC networks and its counterpart vehicular fog computing (VFC) networks in recent years. Some works assumed that the allocated resources are fixed when studying vehicular application offloading. Hou et al. proposed the infrastructure of VFC and studied the communication and computational resource utilization of each vehicle to provide communication and computation services [10]. In [11], Zhou et al. studied service provisioning for workload offloading in VEC networks. The workload offloading problem was formulated to minimize the overall energy consumption of all vehicle users while considering the total energy consumption and latency. An ADMM-based solution method was proposed to solve the workload offloading problem. In [5], Zhao et al. proposed a computation offloading framework in an SDN-enabled VEC system assisted by UAV to minimize the system costs. A sequential game was applied to solve the multi-player computing offloading problem. In [3], Wang et al. studied resource competition among vehicles for computation offloading in VEC networks. They proposed a multi-user non-cooperative game to maximize the utility of each vehicle. To overcome the limitation of performance gain caused by the overhead when vehicles process their tasks on the same edge server, Dai et al. studied the integration of load balancing and task offloading in the VEC networks [6]. They formulated a MINP problem to maximize the system utility. In [12], Zhu et al. studied service latency and quality loss trade-off in VFC by optimizing task allocation. In [13], Du et al. studied the application offloading of vehicular terminals in VEC networks, which was formulated as a dual-side optimization problem to minimize the costs of VTs and the MEC server simultaneously. In [4], Sun et al. studied task replication in a VEC system, where tasks of vehicles can offload their tasks to multiple vehicles, such that the offloading delay can be minimized. However, this work did not analyze computing resource allocation. In [14], Kang et al. studied data sharing security in the VEC networks by utilizing blockchain technologies. Motivated by [14], the authors proposed a consortium blockchain for securing computing resource sharing among vehicles. An incentive mechanism based on the contract was designed to motivate the vehicles to share their computing resources. In [15], Wang et al. studied computation task offloading in a VEC system where vehicles offload tasks by using the computing resources of the available neighboring VEC clusters. Their aim is to minimize the system energy consumption while the task delay constraint is satisfied. An imitation learning algorithm was proposed to schedule the tasks of vehicles. In [16], Shine et al. studied optimizing delay and energy consumption considering the design of federated learning and the computation offloading process. Their problem formulation is solved by proposing an evolutionary genetic-based algorithm. However, in the above two works, the computing resources of VEC servers are assumed to be fixed.

Some works studied the allocation of computational resources of VEC servers and vehicular application offloading. In [17], NG et al. studied resource allocation of VEC servers for coded distributed computing (CDC) task offloading in VEC. They proposed a double auction mechanism for allocating computing resources of edge servers in order to complete the CDC tasks. In [18], Ning et al. presented an intelligent offloading framework for VEC systems, where the vehicular applications and resource allocation strategy problem formulation were given, and an algorithm based on a two-sided matching scheme and DRL was developed to obtain the solution with the aim of maximizing the vehicles' quality of experience. In [9], Khayyat et al. studied computational offloading and resource allocation in multi-vehicle edge-cloud networks to minimize the entire system costs in terms of energy and time. They proposed a deep-learning-based algorithm to solve the NP-Hard problem. In [19], Wang et al. investigated the transmission power and computation resources of a vehicle for vehicular application offloading in a single-user VEC system. Due to the complexity of the studied problem, a low-complexity algorithm was proposed to solve it. In [20], Tan and Hu studied the joint allocation of communication, caching and the computation resource problem in VEC networks considering the vehicles' mobility. A deep reinforcement learning (DRL) framework was proposed to address the formulated problem. In [21], Zhou et al. studied computation resource allocation for optimizing the task assignment in VFC networks. They proposed a contract-matching method to minimize the network delay. In [22], Li et al. studied bandwidth and computation resources allocation of edge servers in order to reduce the offloading delay. In [23], the authors proposed an offloading scheme to complete delay and computation intensive tasks in VEC networks. They jointly considered link reliability and the allocation of available computation resources of vehicles. In [24], Wu and Yan studied multi-user computation offloading in vehicle-aware MEC networks. They considered computing resources and bandwidth distribution and proposed an algorithm based on deep reinforcement learning to minimize the system energy consumption and delay. In [25], Cui et al. proposed an intelligent resource allocation strategy based on RL. They combined the allocation of communication and computation resources so that the low latency and the reliability can be addressed. In this work [26], Li et al. studied a resource allocation scheme considering bandwidth allocation to minimize the total costs of energy and time. However, these works did not jointly analyze the computing resources of vehicles and VEC servers. In the work of [27], computation resource allocation and crowd sensing data offloading in the VEC systems are studied for system latency minimization without considering the energy consumption. In reference [28], Li et al. studied the computation resource allocation of both devices and the edge server. They aimed to solve the minimization problem of the energy consumption for all the devices.

For the existing works mentioned above, they either considered the allocation of the resources of vehicles or the edge server but did not jointly consider the allocation of the computation resources of both the vehicles and the edge server. Although, in [29], Wang et al. studied computation resource allocation of the MEC server, and the objective of this work is different from ours. In this paper, computation resource allocation of the vehicles and the VEC server for vehicular application offloading are investigated. Energy consumption and time cost are taken into account.

3. System Model and Problem Formulation

In this section, we first overview the system model and then show the formulated optimization problem. Consider a vehicular VEC system, which is composed of an VEC server and the number of N vehicles, as illustrated in Figure 1. The VEC server provides computational resources to process the generated data from the vehicular applications of these vehicles. We suppose that each vehicle has a vehicular application to be processed. The VEC server is deployed at a roadside unit (RSU), through which each vehicle's application can be transmitted and processed by the edge server. All vehicles are assumed to be within the range of the RSU. The application of vehicle i is denoted by $Q_i = (L_i, I_i)$ [30], where L_i denotes the application's data size in bits, and I_i is the computing intensity, which means the required computing CPU cycles to finish computing one bit of the data. Let the binary value o_i denote the application offloading strategy of the vehicle i. If the vehicle i processes its application by way of offloading using edge cloud resources, $o_i = 1$. Otherwise, $o_i = 0$.

Remark 1. *It should be pointed out that we considered the case that all vehicles arrive at the range of the RSU simultaneously. However, the vehicles may arrive at the range of the RSU sequentially. In this case, the computation resources will be reallocated sequentially. This case corresponds to the scenario in which computation resources are allocated according to the number of vehicles.*

It should also be noted that the vehicles may take vehicle-to-everything (V2X) communication, which allows the vehicular applications of some vehicles to be processed by using the computation resources of other vehicles. In this case, the offloading strategies are almost independent of the number of vehicles and the computation capacity of the VEC server. For ease of analysis, we only considered the scenario of offloading applications to the VEC server.

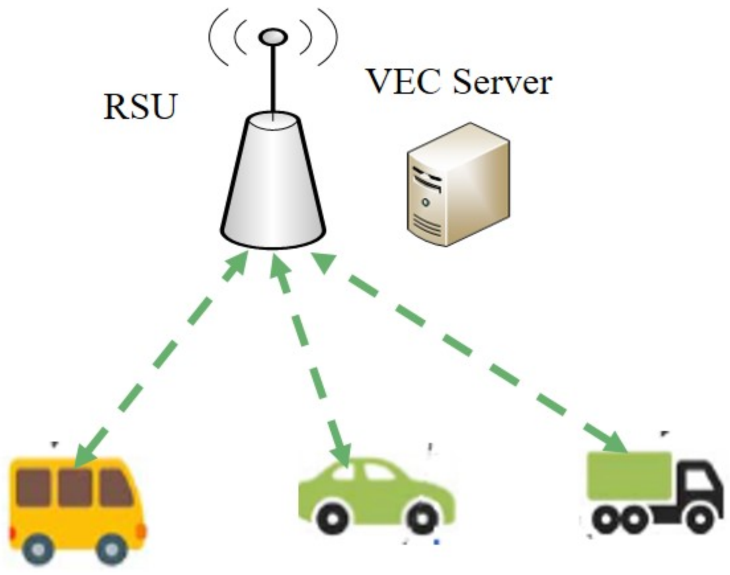


Figure 1. System model.

3.1. Local Execution

In local execution, each vehicle will use its own computing resources to complete its application. For the vehicle i, the time cost and the energy cost for this vehicular application completion are expressed respectively as:

$$t_{i,l} = \frac{L_i I_i}{f_i} \tag{1}$$

$$e_{i,l} = P_i \frac{L_i I_i}{f_i} \tag{2}$$

where f_i is the allocated computational resource of vehicle i in the CPU cycle numbers per second, and P_i denotes the consumed power per second.

The power consumed by vehicle i is denoted by

$$P_i = k_i f_i^3 \tag{3}$$

where k_i depends on the vehicle $i's$ chip architecture, and its value can be 10^{-26} [13].

The computing cost for the local execution is denoted as

$$\begin{aligned} C_{i,l} &= \gamma_{i,t} t_{i,l} + \gamma_{i,e} e_{i,l} \\ &= \gamma_{i,t} \frac{I_i L_i}{f_i} + \gamma_{i,e} P_i I_i L_i \end{aligned} \tag{4}$$

where $\gamma_{i,t}$ and $\gamma_{i,e} \in [0,1]$ represent the values of the weighting factors of time and energy costs for vehicle i, respectively. If $\gamma_{i,t} > \gamma_{i,e}$, vehicle i places a higher value on the execution time of vehicular application. Otherwise, if $\gamma_{i,t} < \gamma_{i,e}$, vehicle i pays more attention to the energy cost.

3.2. VEC Server Execution

In the VEC server execution, each vehicle will offload its vehicular application to the edge cloud. When the data of the vehicular applications are offloaded to the VEC server via wireless communications, extra time for transmitting these data and energy costs will be required. For the vehicle i, its data transmission rate in the uplink channel is

$$R_i = B_i \log_2(1 + \frac{p_i h_i}{\sigma_0^2}) \tag{5}$$

where B_i is the bandwidth allocation, p_i denotes the transmission power, h_i is the channel gain denoted as d_i^{-2}, where d_i is the distance between the vehicle i and the RSU, and σ_0^2 denotes the Gaussian noise.

Therefore, the time cost and energy cost for transmitting vehicle applications are respectively denoted as,

$$t_{i,t} = \frac{L_i}{R_i} \tag{6}$$

$$e_{i,t} = p_i t_{i,t} = p_i \frac{L_i}{R_i} \tag{7}$$

When the vehicular application of vehicle i is completed by using the computational resources of the VEC server, the time cost is

$$t_{i,e} = \frac{I_i L_i}{F_i} \tag{8}$$

where F_i denotes the computational resources allocated to vehicle i.

The computation costs for the VEC server execution are denoted as

$$\begin{aligned} C_{i,e} &= \gamma_{i,t} t_{i,e} + \gamma_{i,e} e_{i,e} \\ &= \gamma_{i,t} (\frac{L_i}{R_i} + \frac{I_i L_i}{F_i}) + \gamma_{i,e} p_i \frac{L_i}{R_i} \end{aligned} \tag{9}$$

For convenience analysis, a summary of the notations is shown in Table 1.

Table 1. Notations summary.

Notation	Description
N	the number of vehicles
f_i	the allocated computing resources of vehicle i
L_i	the vehicular application size of the vehicle i in bits
I_i	the needed number of CPU cycles to finish the vehicular application of vehicle i
$\gamma_{i,t}$	the weighting factor of the execution time cost for vehicle i
$\gamma_{i,e}$	the weighting factor the energy energy of vehicle i
λ	the Lagrange multiplier
B_i	the allocated bandwidth to vehicle i
p_i	the transmission power of vehicle i
h_i	the channel gain from vehicle i to the BS
σ_0^2	the Gaussian noise
R_i	the uplink rate for vehicle i
o_i	the vehicular application offloading strategy determined by vehicle i
F_i	the allocated VEC computation resources to vehicle i

3.3. Problem Formulation

The resource allocation for the vehicular application offloading problem is formulated with the objective of minimizing the time and energy costs of all vehicles.

Denote C_i as the computing cost for executing the vehicular application of vehicle i, which is given as

$$\begin{aligned} C_i &= (1-o_i)C_{i,l} + o_i C_{i,e} \\ &= (1-o_i)[\gamma_{i,t}\frac{I_i L_i}{f_i} + \gamma_{i,e} k_i f_i^2 I_i L_i] + \\ & o_i[\gamma_{i,t}(\frac{L_i}{R_i} + \frac{I_i L_i}{F_i}) + \gamma_{i,e}\frac{p_i L_i}{R_i}] \end{aligned} \tag{10}$$

Therefore, the optimization problem for time and energy cost minimization of all vehicles is expressed as,

Problem 1.

$$\begin{aligned} \min_{f_i, F_i, o_i} & \sum_{i=1}^{N} C_i \\ s.t. \quad & 0 < f_i \leq f_{i,m} \\ & \sum_{i=1}^{N} F_{i,e} \leq F \\ & a_i \in \{0, 1\} \end{aligned} \tag{11}$$

where the first constraint means that the computing resource of vehicle i should not exceed the maximum value $f_{i,m}$, the second one is the constraint of computation resources allocated to vehicle i, and o_i is the vehicular application offloading strategy of vehicle i.

Remark 2. *As the vehicular application offloading strategy is a binary value, the allocated computing resources of the vehicles and the VEC server are continuous values in the formulated problem. Therefore, Problem 1 is an MINP problem and non-convex, and it is hard to obtain the solution. In traditional methods, the optimal solutions to the MINP problems can be reached by applying the Dinkelbach, Branch-and-Bound, and Alternating Direction Method of Multipliers (ADMM). However, the time complexity is considered to be prohibitive [31]. In work [32], Yu et al. solved the formulated MINP problem by making use of the traditional methods, and their performances were compared. In the next section, an efficient method to solve Problem* **P** *is proposed.*

4. Methodology

As the formulated problem in the previous section is not a standard convex optimization problem, an efficient methodology is introduced to solve it in this section. Firstly, a lemma is shown, whose proof has been provided in [33]. Our proposed solution approach is given by referring to this lemma.

Lemma 1. *We always have*

$$\sup_{x,y} f(x,y) = \sup_{x} \tilde{f}(x),$$

where $\tilde{f}(x) = sup_y f(x,y)$*.*

This lemma shows that a function could be minimized by firstly optimizing some variables and optimizing the left ones later.

By referring to Lemma 1, the solution to the formulated problem, Problem 1, can be obtained by sequentially optimizing f_i, F_i and o_i. In other words, we can firstly optimize the computing resources of vehicles, and the VEC servers, assuming that the offloading strategies of vehicles are given. Therefore, Problem 1 can be decomposed into the following subproblems:

(1) Local execution problem;
(2) VEC server execution problem;
(3) Vehicular application offloading strategy problem.

4.1. Local Execution Problem

From Problem 1, when $o_i = 0$, vehicles will complete their application executions by using their own computing resources. Therefore, the local execution problem is,

Problem 2.

$$\begin{aligned} &\min_{f_i} C_{i,l}(f_i) \\ &s.t. \quad 0 < f_i \le f_{i,m} \end{aligned} \tag{12}$$

where $C_{i,l}(f_i)$ *is denoted as*

$$\begin{aligned} C_{i,l} &= \gamma_{i,t} t_{i,l} + \gamma_{i,e} e_{i,l} \\ &= \gamma_{i,t} \frac{I_i L_i}{f_i} + \gamma_{i,e} k_i f_i^2 I_i L_i \end{aligned} \tag{13}$$

According to the second-order derivative of Equation (13),

$$\frac{\partial^2 C_{i,l}}{\partial {f_i}^2} = \frac{2\gamma_{i,t} I_i L_i}{f_i^3} + 2 I_i L_i \gamma_{i,e} \tag{14}$$

it is easily verified that $C''_{i,l}$ is positive in the domain of f_i, which means that $C_{i,l}$ is convex in its domain. The first derivative of $C_{i,l}(f_i)$ with respect to f_i,

$$\frac{\partial C_{i,l}}{\partial f_i} = -\frac{\gamma_{i,t} I_i L_i}{(f_i)^2} + 2\gamma_{i,e} k_i f_i L_i I_i = 0 \tag{15}$$

From Equation (15), the optimal solution is

$$f_i^* = \sqrt[3]{\frac{\gamma_{i,t}}{2 k_i \gamma_{i,e}}} \tag{16}$$

It can be readily observed that $C_{i,l}(f_i)$ is a monotonously increasing function when $f_i > f_i^*$ and a monotonously increasing function when $f_i < f_i^*$. From this conclusion, the solution to the local execution problem P2 can be expressed as

$$C_{i,l}(f_i) = \begin{cases} C_{i,l}(f_{i,m}), f_i^* \geq f_{i,m} \\ C_{i,l}(f_i), f_i^* < f_{i,m} \end{cases} \tag{17}$$

Consequently, the minimum computing costs in the local execution problem can be denoted as

$$C_{i,l}(f_i) = \gamma_{i,t}\frac{I_iL_i}{f_i^*} + \gamma_{i,e}k_i(f_i^*)^2 I_iL_i \tag{18}$$

4.2. VEC Server Execution Problem

Based on Problem 1, when $o_i = 1$, the vehicles will execute the vehicular applications by using the computing resources in the VEC server. Therefore, the VEC server execution problem is,

Problem 3.

$$\begin{gathered} \min_{F_i} \sum_{i=1}^{N} C_{i,e}(F_i) \\ s.t. \sum_{i=1}^{N} F_i \leq F \\ F_i > 0 \end{gathered} \tag{19}$$

where $C_{i,e}(F_i)$ is denoted as

$$C_{i,e}(F_i) = \gamma_{i,t}\frac{I_iL_i}{F_i} \tag{20}$$

From the objective function of Problem 3, we can obviously observe that it monotonously decreases with f_i. Therefore, the optimal solution of this function with respect to f_i is $f_{i,m}$. As the domain of the objective function is convex, and from

$$\frac{\partial^2 C_{i,e}}{\partial F_i{}^2} = \frac{2\gamma_{i,t}I_iL_i}{F_i^3} > 0 \tag{21}$$

we have the conclusion that $C_{i,e}$ is a convex function [33]. We can formulate the following Lagrangian function

$$L(F_i, u) = \sum_{i=1}^{N}[\frac{\gamma_{i,t}I_iL_i}{F_i} + u(\sum_{i=1}^{N} F_i - F)] \tag{22}$$

where $u \geq 0$ denotes the Lagrange multiplier.

We can obtain the following dual problem of Problem 3 as

$$\varphi(u) = \underset{u\geq 0}{max}\ \underset{F_i>0}{min}\ L(F_i, u) = \sum_{i=1}^{N}[\frac{\gamma_{i,t}I_iL_i}{F_i} + u(\sum_{i=1}^{N} F_i - F)] \tag{23}$$

From

$$\frac{\partial L}{\partial F_i} = \frac{-\gamma_{i,t}I_iL_i}{F_i^2} + u = 0 \tag{24}$$

we know that $u > 0$, and

$$F_i^* = \sqrt{\frac{\gamma_{i,t}I_iL_i}{u}} \tag{25}$$

Substituting Equation (25) into Equation (23), the Lagrangian multiplier value is

$$u = \left(\frac{\sum_{i=1}^{N} \sqrt{\gamma_{i,t} I_i L_i}}{F}\right)^2 \tag{26}$$

Substituting Equation (26) into Equation (25), the optimal computational resource allocation of the VEC server is

$$F_i^* = \frac{\sqrt{\gamma_{i,t} I_i L_i}}{\sum_{i=1}^{N} \sqrt{\gamma_{i,t} I_i L_i}} F \tag{27}$$

Substituting Equation (27) into Equation (9), the optimal computation costs of the VEC server execution can be obtained and expressed as

$$C_{i,e}^* = \gamma_{i,t}\left(\frac{L_i}{R_i} + \frac{I_i L_i}{F_i^*}\right) + \gamma_{i,e} p_i \frac{L_i}{R_i} \tag{28}$$

4.3. Offloading Problem

An efficient computing resource allocation for the vehicular application offloading algorithm in the VEC system is put forward, which is illustrated in Algorithm 1. Vehicles determine to process vehicular applications by applying the computational resources that the VEC server provides if and only if the computation costs of the edge server execution are less than the computation costs of the local execution. Vehicle i determines the vehicular application offloading strategy by making a comparison of the computation costs caused by the local execution model and the VEC server execution model,

$$o_i = \begin{cases} 1, C_{i,l} \geq C_{i,e} \\ 0, C_{i,l} < C_{i,e} \end{cases} \tag{29}$$

When the minimum computation costs for all vehicles are obtained, the system computation costs are denoted as

$$C^* = \sum_{i=1}^{N} ((1 - o_i) C_{i,l}^* + o_i C_{i,e}^*) \tag{30}$$

The computation resource allocation for vehicular application offloading strategy of each vehicle is summarized in Algorithm 1.

Algorithm 1 The Proposed Vehicular Application Offloading Algorithm

```
Input:
 1: $N$ applications of vehicles;
Output:
 2: the computation resources allocation, vehicular application offloading strategy, and
    computation costs;
 3: Solving subproblem Problem2;
 4: Obtain the optimal allocated computational resource for each vehicle based on
    Equation (17);
 5: Calculate the minimum computation costs for each vehicle in the local execution
    problem according to Equation (18);
 6: Solving subproblem Problem3;
 7: Obtain the optimal allocated computational resources of the VEC server based on
    Equation (25);
 8: Obtain the minimum computation costs in the execution model of the VEC server
    execution problem from Equation (28);
 9: if $C_{i,l} \geq C_{i,e}$ then
10:     $o_i = 1$;
11: else
12:     $o_i = 0$;
13: end if
```

5. Experimental Evaluation

The performance gain from the proposed vehicular offloading strategy is quantified by using the numerical experimental results. In addition, the performances are confirmed by comparing our offloading scheme with the following two baseline schemes:

Local Scheme: In this scheme, all vehicles complete vehicular applications by using their computational resources.

VEC Scheme: In this scheme, all vehicles complete vehicular applications by applying the computational resources of the VEC server.

5.1. Simulation Settings

We assume that there are $N = 5$ vehicles and one VEC server in a VEC network system. The computation capacity of the VEC server is $F = 10$ GHz, and the computation capacity of each vehicle is 1 GHz. Each vehicle has a vehicular application that needs to be executed. These vehicular applications can be the traffic efficiency applications, which focus on planning the route for vehicles or sharing the information of geographical location and road conditions, and the infotainment applications, which provide the location of car rental services or video streaming services. For the vehicular application i, its data size L_i is chosen from the range $[0.2, 1]$ Mbits, computing intensity is set as 1000 cycles per bit, transmission power p_i is set as 0.2 W and bandwidth B_i is set as 0.36 WHz. The Gaussian noise is set as 10^{-13} W. The distance between the vehicles and the RSU ranges from 0 to 1000 m. The default parameter values are shown in Table 2. We set these values by mainly referring to [15,25,34]. These values have been verified in real-world datasets.

Table 2. Parameter values.

Parameters	Values
N	5
B_i	0.36 MHz
L_i	[0.2, 1] Mbits
I_i	1000 Cycles/bit
p_i	0.2 W
d_i	$[0, 1000]$ m
σ_0^2	10^{-7} W
f_i	[0, 1] GHz
F_i	10 GHz

5.2. Impacts of the Values of Weighting Factor

The impacts of the values of weighting factors on the offloading strategies and the time cost and energy of vehicular applications are analyzed. We set the vehicle number as 5, the capacity of the VEC server as 5 GHz, and $\gamma_{i,t}$ as 0.2, 0.5, and 0.8, respectively. The experimental results are shown in Figure 2a–c.

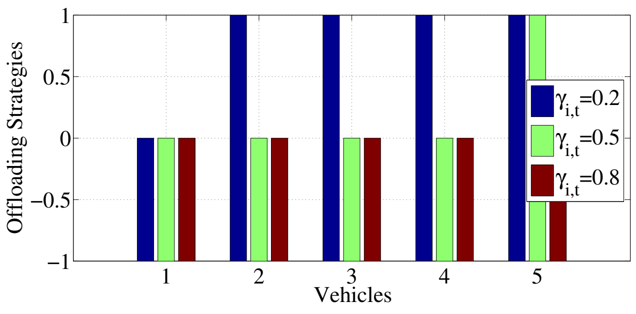
Figure 2a plots the impacts of values of weighting factors on the offloading strategies of vehicle users. In Figure 2a, when $\gamma_{i,t} = 0.2$, it can be observed that vehicle users 2, 3, 4, and 5 execute their vehicular applications in the VEC server while vehicle user 1 executes its vehicular application locally. One reason is because of the fact that the data size of the vehicular application of vehicle user 1 is smaller than other vehicle users. Another reason is that vehicle users consider execution time the main factor in this situation. In Figure 2a, when $\gamma_{i,t} = 0.5$, it can be found that only vehicle user 5 executes its vehicular application in the VEC server. Especially when $\gamma_{i,t} = 0.8$, we find that all vehicle users execute their vehicular applications locally.

In Figure 2b, it is easily found that the time cost decreases with the values of the weighting factor $\gamma_{i,t}$ increasing. Conversely, it is seen from Figure 2c that the energy cost increases as the value of the weighting factor increases. By comparing Figures 2b and 2c, we see that the weighting factor plays a vital role in the costs of time and energy.

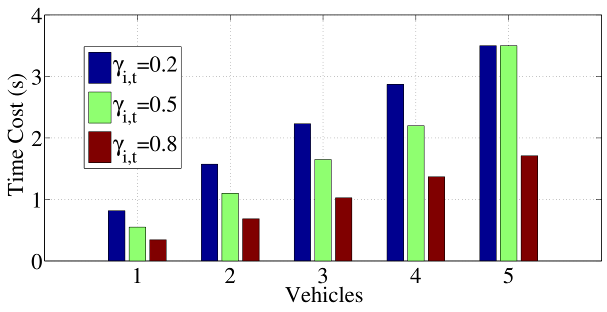
5.3. Impacts on the Capacity of the VEC Server

The impacts of the capacity of the VEC server on the offloading strategies, the time and energy cost, and the computation cost are analyzed, respectively. We set the vehicle number as 5, varied the capacity of the VEC server from 4 to 8 GHz, and show the simulation results in Figures 3–6.

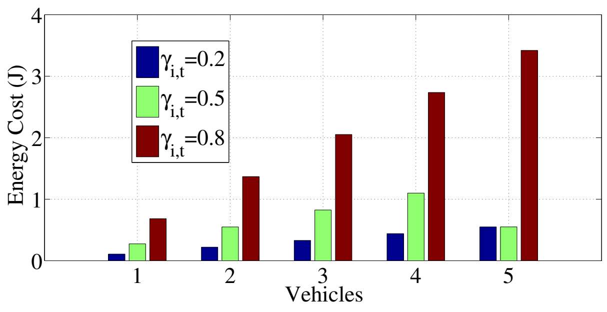
Figure 3a–c plots the impacts of the capacity of the VEC server on the offloading strategies that vehicles use, considering different values of the weighting factor. We set the vehicle number as 5 and varied the capacity of the VEC server from 4 to 8 GHz. In Figure 3a, it is observed that all vehicle users will offload their vehicular applications in case the capacity of the VEC server increases. In particular, for the vehicular applications of the vehicle users whose data sizes are larger than others, they are executed in the VEC server when $\gamma_{i,t} = 0.2$. However, in Figure 3b, when $\gamma_{i,t} = 0.5$, only some vehicle users will take the offloading strategies. In Figure 3c, when $\gamma_{i,t} = 0.8$, all vehicle users will execute their vehicular applications in the local model. This is because the local execution time is smaller than the time cost in the offloading and execution.



(**a**)

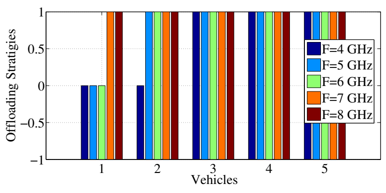


(**b**)

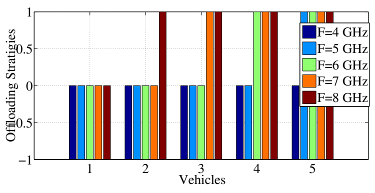


(**c**)

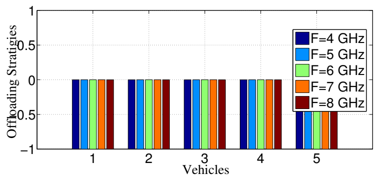
Figure 2. Impacts of the values of weighting factors. (**a**) Offloading strategies; (**b**) time cost; (**c**) energy cost.



(**a**)

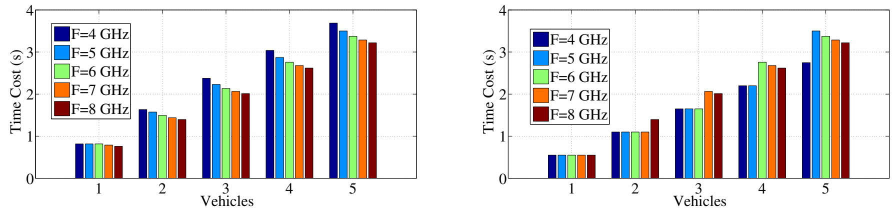


(**b**)



(**c**)

Figure 3. The impacts of the capacity of the VEC server on the offloading strategies. (**a**) $\gamma_{i,t} = 0.2$, $\gamma_{i,e} = 0.8$; (**b**) $\gamma_{i,t} = 0.5, \gamma_{i,e} = 0.5$; (**c**) $\lambda_{i,t} = 0.8, \gamma_{i,e} = 0.2$.

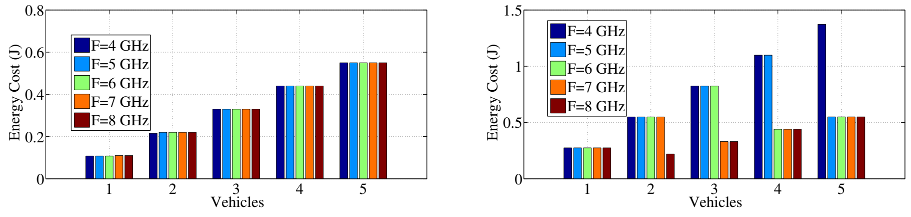


(**a**) (**b**)

Figure 4. The impacts of the capacity of the VEC server on the time cost. (**a**) $\gamma_{i,t} = 0.2, \gamma_{i,e} = 0.8$; (**b**) $\gamma_{i,t} = 0.5, \gamma_{i,e} = 0.5$.

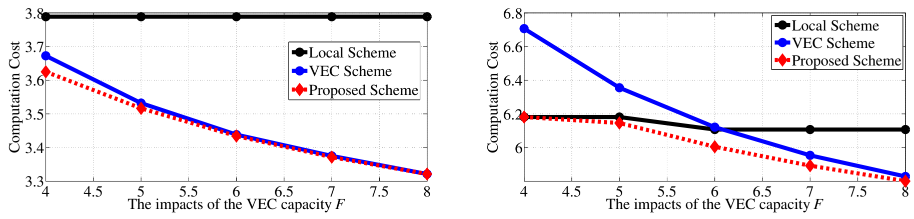
Figure 4a,b depicts the impacts of the capacity of the VEC server on the time cost considering different values of the weighting factor. We set the vehicle number as 5 and varied the capacity of the VEC server from 4 to 8 GHz. In the two figures, we see that the time cost of each vehicle user decreases as the capacity of the VEC server increases when $\gamma_{i,t} = 0.2$. However, when $\gamma_{i,t} = 0.5$, the time cost of each vehicle user is not necessarily decreased. This is because some vehicle users execute their vehicular applications locally, and the local time cost is more minor compared to the time cost in the VEC server.

Figure 5a,b depicts the impacts of the capacity of the VEC server on the energy cost considering different values of the weighting factor. In Figure 5a, we see that the energy cost of each vehicle user will decrease with the increase in the capacity of the VEC server when $\gamma_{i,t} = 0.2$. However, when $\gamma_{i,t} = 0.5$, the energy of some vehicle users, such as the 2, 3, 4, and 5, will increase. This is because some of these vehicle users execute their vehicular applications locally while others offload.



(**a**) (**b**)

Figure 5. The impacts of the capacity of the VEC server on the energy cost. (**a**) $\gamma_{i,t} = 0.2, \gamma_{i,e} = 0.8$; (**b**) $\gamma_{i,t} = 0.5, \gamma_{i,e} = 0.5$.



(**a**) (**b**)

Figure 6. The impacts of the capacity of the VEC server on the computation cost. (**a**) $\gamma_{i,t} = 0.2, \gamma_{i,e} = 0.8$; (**b**) $\gamma_{i,t} = 0.5, \gamma_{i,e} = 0.5$.

Figure 6a,b illustrates the impacts of the capacity of the VEC server on the computation cost considering different values of the weighting factor. In Figure 4, we obtain the observation that the computation cost of all vehicle users decreases under the **VEC Scheme** and the **Proposed Scheme** with the increase in the capacity of the VEC server. However, the computation cost of the **VEC Scheme** does not change versus the increase in the capacity of the VEC server. When $\gamma_{i,t} = 0.2$, the computation cost of the **VEC Scheme** is much higher than the computation cost of the **VEC Scheme**. From the two figures, we have the evident observation that the **Proposed Scheme** can outperform the two baseline schemes.

6. Discussions

In this paper, we only considered the Vehicle-to-RSU (V2R) case, namely, offloading vehicular applications to the VEC server. However, we can also consider the case of Vehicle-to-Vehicle (V2V) [35–37], which allows applications to be processed by other vehicles. Vehicle networks can apply the two communication models to support a series of vehicular applications, which mainly include the following categories [36,38]:

Autonomous/Cooperative Driving: This kind of vehicular application mainly adopts the V2V communication model and has a much higher requirement for latency, typically less than 10 ms.

Traffic Safety: This kind of application is to provide traffic safety services for the vehicle, such as the pre-sense crash warning, which requires 50 ms of round-trip latency.

Traffic Efficiency: The traffic efficiency applications mainly focus on planning the route for vehicles and sharing information on geographical location and road conditions. Such vehicular applications generally do not have strict requirements for tolerable latency, ranging from 100 to 500 ms.

Infotainment: The infotainment applications can offer the location of car rental services or video streaming services. These types of vehicular applications have the lowest requirements for tolerable latency compared with the other categories of vehicular applications.

According to the experimental results and analysis in Chapter 5 and the categories of the vehicular applications, the proposed model of this paper can be applied to traffic efficiency applications and infotainment applications. For the traffic safety applications and traffic efficiency applications, as they have extreme requirements for latency, the operator of the VEC system has to enlarge its cloud capacity to provide more computation resources.

Our study can be further investigated from the following aspects. First, the queueing model can be applied to analyze the performances of the VEC system [39]. Second, the Software Defined Networking (SDN) technology can be applied to the VEC system [2,5]. Third, the safety issue of the VEC systems needs to be further explored [40]. In addition, the digital twin [41], unmanned aerial vehicle (UAV) [42] and reconfigurable intelligent surfaces (RISs) [43] can be integrated into the VEC system. Other solution methods to the MINP problems, such as the dandelion algorithm (DA) [44] and Bat algorithm [45], can also be adopted to solve the formulated problem.

7. Conclusions

This paper has presented an investigation of computational resource allocation and vehicular application offloading in a VEC network system. In a specific manner, we analyzed the computational resource allocation of the VEC server and the vehicles to minimize the computation cost in terms of time and energy costs. We formulated an optimization problem by considering both the time and energy cost of vehicles. As the objective function for the optimization problem is not convex, the formulated problem is known as a MINP problem, which is NP-hard. To solve this problem, we decomposed it into three sub-problems, and the solution to each one was obtained. We have analyzed the impacts of different parameters, such as the values of the coefficient factors and the capacity of the VEC server on the time and energy costs as well as the computation cost. The simulation results have revealed that the proposed offloading scheme has the benefits of efficient allocation of computation resources and vehicular application processing in the VEC network system. The experimental results highlighted the fact that the values of coefficient factors and the capacity of the VEC server have a dramatic impact on the offloading strategies of vehicle users. In comparison to the two baseline schemes, the proposed scheme reveals its superiority in the cost of time and energy. Moreover, the computation cost can be significantly reduced.

Author Contributions: Conceptualization, F.G. and X.L.; methodology, F.G. and X.L.; software, X.Y. and H.D.; validation, X.Y. and H.D.; formal analysis, F.G. and X.L.; investigation, F.G. and X.Y.; resources, X.Y. and H.D.; data curation, X.Y.; writing—original draft preparation, F.G. and X.Y.; writing—review and editing, X.L.; visualization, F.G. and X.Y.; supervision, X.L.; project administration, X.L.; funding acquisition, X.Y. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by a funding project for the cultivation of outstanding talents in Colleges and Universities (gxyqZD2021135), the Natural Science Foundation of Anhui Provincial Education Department (KJ2020A0733), scientific research of high level talents of Bengbu University (BBXY2020KYQD02), and High Level Scientific Research and Cultivation Project of Benbu University (2021pyxm05).

Data Availability Statement: The data of this paper can be available from the corresponding author.

Acknowledgments: The authors of this paper would like to thank the academic editor and the anonymous reviewers for their precious comments, which contributed a lot to improving the quality of this paper.

Conflicts of Interest: The authors declare that they have no conflict of interest.

Abbreviations

The following abbreviations are used in this paper:

VEC	vehicular edge computing
MEC	mobile edge computing
IoT	Internet of Things
MINP	mixed-integer nonlinear programming

References

1. Zhao, L.; Li, X.; Gu, B. Vehicular Communications: Standardization and Open Issues. *IEEE Commun. Stand. Mag.* **2018**, *2*, 74–87. [CrossRef]
2. Zhao, L.; Zheng, T.; Lin, M.; Hawbani, A.; Shang, J.; Fan, C. SPIDER: A Social Computing Inspired Predictive Routing Scheme for Softwarized Vehicular Networks. *IEEE Trans. Intell. Transp.* **2021**, *1*, 1–12. [CrossRef]
3. Wang, Y.; Lang, P.; Tian, D. A game-based computation offloading method in vehicular multiaccess edge computing networks. *IEEE Internet Things* **2020**, *6*, 4987–4996. [CrossRef]
4. Sun, Y.; Zhou, S.; Niu, Z. Distributed task replication for vehicular edge computing: Performance analysis and learning-based algorithm. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 1138–1151. [CrossRef]
5. Zhao, L.; Yang, K.; Tan, Z.; Li, X.; Sharma, S.; Liu, Z. A novel cost optimization strategy for sdn-enabled uav-assisted vehicular computation offloading. *IEEE Trans. Intell. Transp.* **2021**, *22*, 3664–3674. [CrossRef]
6. Dai, Y.; Xu, D.; Maharjan, S. Joint load balancing and offloading in vehicular edge computing and networks. *IEEE Internet Things* **2019**, *6*, 4377–4387. [CrossRef]
7. Qiao, G.; Leng, S.; Maharjan, S. Deep Reinforcement Learning for Cooperative Content Caching in Vehicular Edge Computing and Networks. *IEEE Internet Things* **2020**, *7*, 247–257. [CrossRef]
8. Boukerche, A.; Grande, R.E.D. Vehicular cloud computing: Architectures, applications, and mobility. *Comput. Netw.* **2018**, *135*, 171–189. [CrossRef]
9. Khayyat, M.; Elgendy, I.A.; Muthanna, A. Advanced Deep Learning-Based Computational Offloading for Multilevel Vehicular Edge-Cloud Computing Networks. *IEEE Access* **2020**, *8*, 137052–137062. [CrossRef]
10. Hou, X.; Li, Y.; Liu, P. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Trans. Veh. Technol.* **2016**, *65*, 3860–3873. [CrossRef]
11. Hou, X.; Li, Y.; Liu, P. Energy-Efficient Edge Computing Service Provisioning for Vehicular Networks: A Consensus ADMM Approachs. *IEEE Trans. Veh. Technol.* **2019**, *68*, 5087–5099.
12. Zhu, C.; Tao, J.; Pastor, J. Folo: Latency and quality optimized task allocation in vehicular fog computing. *IEEE Internet Things* **2019**, *6*, 4150–4161. [CrossRef]
13. Du, J.; Yu, F.R.; Chu, X. Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. *IEEE Trans. Veh. Technol.* **2019**, *68*,1079–1092. [CrossRef]
14. Kang, J.; Yu, R.; Huang, X. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet Things* **2019**, *6*, 4660–4670. [CrossRef]
15. Wang, X.; Ning, Z.; Guo, S. Imitation Learning Enabled Task Scheduling for Online Vehicular Edge Computing. *IEEE Trans. Mobile Comput.* **2022**, *21*, 598–611. [CrossRef]

16. Shinde, S.S.; Bozorgchenani, A.; Tarchi, D. On the Design of Federated Learning in Latency and Energy Constrained Computation Offloading Operations in Vehicular Edge Computing Systems. *IEEE Trans. Veh. Technol.* **2022**, *71*, 2041–2057. [CrossRef]
17. Ng, J.; Lim, W.; Xiong, Z. A double auction mechanism for resource allocation in coded vehicular edge computing. *IEEE Trans. Veh. Technol.* **2022**, *71*, 1832–1845.
18. Ning, Z.; Dong, P.; Wang, X. Deep reinforcement learning for vehicular edge computing: An intelligent offloading system. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–24. [CrossRef]
19. Wang, J.; Feng, D.; Zhang, S. Computation offloading for mobile edge computing enabled vehicular networks. *IEEE Access* **2019**, *7*, 62624–62632. [CrossRef]
20. Tan, L.; Hu, R.Q. Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning. *IEEE Trans. Veh. Technol.* **2018**, *67*, 10190–10203. [CrossRef]
21. Zhou, Z.; Liu, P.; Feng, J. Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3113–3125. [CrossRef]
22. Li, S.; Zhang, N.; Chen, H. Joint road side units selection and resource allocation in vehicular edge computing. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13190–13204. [CrossRef]
23. Bute, M.S.; Fan, P.; Zhang, L. An efficient distributed task offloading scheme for vehicular edge computing networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13149–13161. [CrossRef]
24. Wu, Z.; Yan, D. Deep reinforcement learning-based computation offloading for 5g vehicle-aware multi-access edge computing network. *China Commun.* **2021**, *18*, 26–41. [CrossRef]
25. Cui, Y.; Du, L.; Wang, H. Reinforcement learning for joint optimization of communication and computation in vehicular networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 13062–13072. [CrossRef]
26. Li, X.; Zhao, L.; Yu, K.; Aloqaily, M.; Jararweh, Y. A cooperative resource allocation model for IoT applications in mobile edge computing. *Comput. Commun.* **2021**, *173*, 183–191. [CrossRef]
27. Feng, W.; Zhang, N.; Li, S.; Lin, S.; Ning, R.; Yang, S.; Gao, Y. Latency Minimization of Reverse Offloading in Vehicular Edge Computing. *IEEE Trans. Veh. Technol.* **2022**, *71*, 5343–5357. [CrossRef]
28. Li, S.; Sun, W.; Sun, Y.; Huo, Y. Energy-Efficient Task Offloading Using Dynamic Voltage Scaling in Mobile Edge Computing. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 588–598. [CrossRef]
29. Wang, Q.; Li, Z.; Nai, K. Dynamic resource allocation for jointing vehicle-edge deep neural network inference. *J. Syst. Architect.* **2021**, *117*, 1–12. [CrossRef]
30. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]
31. Lyu, X.; Tian, H.; Ni, W.; Zhang, Y.; Zhang, P.; Liu, R.P. Energy-efficient admission of delay-sensitive tasks for mobile edge computing. *IEEE Trans. Commun.* **2018**, *6*, 2603–2616. [CrossRef]
32. Yu, Y.; Bu, X.; Yang, K. Green large-scale fog computing resource allocation using joint benders decomposition, dinkelbach algorithm, admm, and branch-and-bound. *IEEE Internet Thing* **2019**, *6*, 4106–4117. [CrossRef]
33. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
34. Sun, Y.; Guo, X.; Song, J. Adaptive learning-based task offloading for vehicular edge computing systems. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3061–3074. [CrossRef]
35. Masini, B.M.; Bazzi, A.; Natalizio, E. Radio Access for Future 5G Vehicular Networks. In Proceedings of the 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), Toronto, ON, Canada, 24–27 September 2017; pp. 1–7.
36. Liu, L.; Chen, C.; Pei, Q.; Maharjan, S.; Zhang, Y. Vehicular Edge Computing and Networking: A Survey. *Mob. Netw. Appl.* **2021**, *26*, 1145–1168. [CrossRef]
37. Zhang, K.; Mao, Y.; Leng, S.; He, Y.; Zhang, Y. Mobile-Edge Computing for Vehicular Networks: A Promising Network Paradigm with Predictive Off-Loading. *IEEE Veh. Technol. Mag.* **2017**, *12*, 36–44. [CrossRef]
38. Moubayed, A.; Shami, A.; Heidari, P.; Brunner, B. Edge-Enabled V2X Service Placement for Intelligent Transportation Systems. *IEEE Trans. Mobile Comput.* **2021**, *20*, 1380–1392. [CrossRef]
39. Edwan, T.A.; Tahat, A.; Yanikomeroglu, H.; Crowcroft, J. An Analysis of a Stochastic ON-OFF Queueing Mobility Model for Software-Defined Vehicle Networks. *IEEE Trans. Mobile Comput.* **2022**, *21*, 1552–1565. [CrossRef]
40. Zhao, L.; Chai, H.; Han, Y.; Yu, K.; Mumtaz, S. A Collaborative V2X Data Correction Method for Road Safety. *IEEE Trans. Reliab.* **2022**, *1*, 951–962. [CrossRef]
41. Zhao, L.; Wang, C.; Li, W.; Zhao, K.; Tarchi, D.; Wan, S.; Kumar, N. INTERLINK: A Digital Twin-Assisted Storage Strategy for Satellite-Terrestrial Networks. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, 1. [CrossRef]
42. Xu, Y.; Xie, H.; Wu, Q.; Huang, C.; Yuen, C. Robust Max-Min Energy Efficiency for RIS-Aided HetNets With Distortion Noises. *IEEE Wirel. Commun.* **2022**, *70*, 1457–1471.
43. Liu, Z.; Zhan, C.; Cui, Y.; Wu, C.; Hu, H. Robust Edge Computing in UAV Systems via Scalable Computing and Cooperative Computing. *IEEE Wirel. Commun.* **2021**, *28*, 36–42.
44. Li, X.-G.; Han, S.-F.; Zhao, L.; Gong, C.-Q.; Liu, X.-J. New dandelion algorithm optimizes extreme learning machine for biomedical classification problems. *Comput. Intell. Neurosci.* **2017**, *2017*, 4523754. [CrossRef] [PubMed]
45. Lin, N.; Tang, J.; Li, X.; Zhao, L. A novel improved bat algorithm in uav path planning, Computers. *Comput. Mater. Contin.* **2019**, *61*, 323–344.