DOI: 10.24996/ijs.2022.63.2.33

ISSN: 0067-2904

A Multi-Objective Task Offloading Optimization for Vehicular Fog Computing

Sura Kh. Abdullah*, Adnan J. Jabir

Department of Computer Science, College of Science, University of Baghdad, Baghdad, Iraq

Received: 1/2/2021 Accepted: 5/4/2021

Abstract

Internet of Vehicle (IoV) is one of the most basic branches of the Internet of Things (IoT), which provides many advantages for drivers and passengers to ensure safety and traffic efficiency. Most IoV applications are delay-sensitive and require resources for data storage and computation that cannot be afforded by vehicles. Thus, such tasks are always offloaded to more powerful nodes, like cloud or fog. Vehicular Fog Computing (VFC), which extends cloud computing and brings resources closer to the edge of the network, has the potential to reduce both traffic congestion and load on the cloud. Resources management and allocation process is very critical for satisfying both user and provider needs. However, the strategy of task offloading to fog node in constraints of energy and latency is still an open issue. Several research works have tackled the resource scheduling problem in the field of VFC; however, the recent studies have not carefully addressed the transmission path to the destination node, nor has it considered the energy consumption of vehicles. This paper aims to optimize the task offloading process in the VFC system in terms of latency and energy objectives while taking the deadline constraint into considerations by adopting a Multi-Objective Evolutionary Algorithm (MOEA). Four different execution/transmission models are proposed where vehicle resources are utilized for tasks execution and transmission, and the well-known Dijkstra's algorithm is adopted to find the minimum path between each two nodes. The simulation results show that the models which involve the vehicles in the transmission process have reduced the latency and the total energy for the VFC system significantly in comparison with other models and the current state of the art methods.

Keywords: VFC; IoV; MOEA; MOP, Task Offloading Optimization.

تحسين متعدد الاهداف لعملية تفريغ المهام في حوسبة ضباب المركبات

سرى خيري عبدالله*، عدنان جمعة جابر

قسم علوم الحاسوب، كلية العلوم، جامعة بغداد، بغداد، العراق

الخلاصة

يوفر إنترنيت المركبات (IoV) العديد من الخدمات للسائقين والركاب لضمان سلامتهم وكفاءة حركة المرور. معظم تطبيقات إنترنيت المركبات حساسة للتأخير وتتطلب موارد قد لا تستطيع المركبات توفيرها، لذلك يتم تفريغ مثل هذا النوع من المهام الى محطات ذات قدرة حسابية عالية مثل السحابة أو الضباب. تمتلك حوسبة ضباب المركبات

*Email: sura.kh.abdullah@gmail.com

(VFC) والتي تقرب موارد الحوسبة السحابية الى حافة الشبكة، القدرة على تقليل الازدحام المروري والحمل على السحابة. تعد إدارة الموارد وتخصيصها أمرًا بالغ الأهمية لتلبية رغبات المستخدمين ومزودي الخدمة. ومع ذلك، فإن استراتيجيات تفريغ المهام إلى عقدة الضباب في ظل قيود الطاقة والوقت لا تزال قيد المناقشة. عالجت العديد من الأعمال البحثية مشكلة جدولة الموارد في مجال VFC. الا ان الدراسات الحديثة لم تأخذ في الاعتبار استهلاك الطاقة للمركبات، ولم تتناول بعناية مسار النقل إلى عقدة الوجهة. يهدف هذا البحث إلى تحسين عملية تفريغ المهام في نظام VFC لتقليل زمن الاستجابة والطاقة المستهلكة مع أخذ قيود الموعد النهائي في الاعتبار حيث تم اعتماد خوارزمية تطورية متعددة الأهداف (MOEA). بالاضافة الى ذلك, تم اقتراح أربعة نماذج مختلفة للتنفيذ والإرسال حيث يتم استغلال قدرة السيارات لتنفيذ المهام ونقلها. واعتماد خوارزمية ديكسترا المعروفة لإيجاد المسار الاقصر بين عقدتين. حيث اظهرت نتائج المحاكاة أن النماذج التي تتضمن المركبات في عملية النقل والتنفيذ قد قللت بشكل كبير من زمن الانتقال والطاقة الإجمالية لنظام **VFC** مقارنةً بالطرازات الأخرى و احدث الاعمال ذات صلة.

I. Introduction

The rapid growth in wireless and embedded computing has led to the emergence of the Internet of Things (IoT) computing. IoT is a technology that connects billions of smart devices to the internet, so these devices work cooperatively to sense and disseminate the useful information required for certain systems [1],[2]. IoT has been successfully invested in various life fields that are considered as the backbone of smart cities and their economies such as home automation systems, Intelligent Transportation Systems (ITS) or Internet of Vehicle (IoV), systems of surveillance seismic vibrations in buildings, tracking levels systems of pollution and radiation in the city, trash management systems, and much more [3]. These systems diverge in their architecture depending on their application context and aim for the same purpose; to provide convenience and safety for users.

Vehicular Ad-hoc Networks (VANETs) encompasses intelligent vehicles where several devices are installed like wireless systems and sensing devices to increase road safety and user luxury. VANET provides two kinds of communications, vehicle to vehicle (V2V) and vehicle to infrastructure (V2I), where the latter allows vehicles to communicate with the Road Side Unit (RSU) that is equipped with high computing capabilities[4]. The VANETs applications delay-sensitive and the low computations capabilities of vehicles have imposed several challenges on the VANETs systems. These challenges involve the need for large storage space and high-speed processing devices to provide systems to take correct decisions in proper time [5].

Cloud computing, which is characterized by its high storage and computation capabilities, has been utilized for various applications including vehicular networks [6]. However, cloud computing is subject to several limitations related to network bandwidth and also the lengthy distance between the cloud and end-users [7] [8]. So, offloading to the remote central cloud may not be an idealistic solution. To conquer these drawbacks, fog computing has emerged to provide computation and storage services near the data sources [9]. Fog can reduce the congestion in the network and latency, due to its proximity to IoT devices and/or the end-user devices.

Recently, researchers are interested in studying the employment of a large number of parked and slow-moving vehicles to improve system performance and reduce response time, by applying the vehicular fog computing (VFC) architecture [10]. VFC consists of a large number of vehicles whose underutilized computation resources can be harnessed and provisioned to other users.

According to the task specifications, the simple tasks can be executed by vehicles while the delay-sensitive ones are offloaded to the high capabilities RSU servers. In addition, when a task requires intensive computation and cannot be executed by fog nodes within a deadline time, it is better to offload it to the unlimited capabilities cloud servers [11]. The task-offloading in the VFC system aims to select suitable resources for executing the vehicles' tasks by considering several parameters and constraints to satisfy the user needs.

The task-offloading in the VFC environment has been considered as an NP-hard optimization problem, where determining the best task offloading decision requires an efficient algorithm that can deal with high complexity and a large size problem. In recent literature, there have been several research works focused on utilizing the heuristic and meta-heuristic algorithm to find the best task offloading solution by making the best tradeoff between different conflicting optimization objectives like latency and energy consumptions under deadline and budget constraints. Different studies have formulated task-offloading as a single objective optimization problem e.g. [12],[13]. On the other hand, several others formulated task-offloading as a multi-objective problem to reduce both task execution latency and energy consumption in the cloud and fog nodes [14]. However, the recent studies that considered the task execution in the cloud and RSUs, did not consider the energy consumption of the vehicles, and have not carefully addressed the transmission path to the destination node. The task transmission path to the destination node, where the task must be executed, has a very high impact on the offloading performance. Thus, this paper investigates most of the possible transmission paths taking into consideration the required time and energy consumption required for task transmission.

In this paper, a multi-objective evolutionary algorithm (MOEA) for task offloading optimization in VFC systems is adopted to reduce the energy consumption and the task completion time. This is achieved by considering the task transmission time and energy under the time deadline constraint. The vehicles' abilities for task execution and transmission are exploited, so the well-known Dijkstra's algorithm is adopted to find the shortest route for task transmission over vehicles. The major contributions of this paper are summarized as follows:

A three layers architecture is proposed, consisting of the 1) Vehicular layer, 2) The RSUs layer which contains several RSUs distributed along the road, 3) The top layer where the Macro Base station (MBS) that has high capabilities and located in the center of the road where this station works to find the optimum solutions for offloading [15].

Different models for determining the entities involved in the task transmission and execution is investigated. These models are analyzed and evaluated to find the best method that fulfills the requirements of both users and service providers in terms of latency and energy consumption.

The MOEA/D algorithm is utilized to achieve task offloading optimization in fog computing by minimizing the consumed energy and the completion time for the generated tasks. This in turn is achieved by utilizing both RSUs and vehicles for task computation and transmission.

The well-known Dijkstra algorithm is adopted in order to find the best path for task transmission in the vehicles layer, taking into consideration both the required time and energy consumption,.

The reminder of this paper is as follows: Section II covers the literature related to task offloading optimization. Section III describes the system architecture, exhibits the proposed task offloading models. Section IV described the problem formulation details and MOEA/D algorithm. Section V discusses the results obtained by the simulation. The paper is concluded in section VI.

II. Related Works

Task offloading is considered as one of the most important aspects in VFC systems, due to its significance in making decisions concerning where to run the vehicle tasks and how to allocate the resources for computation. A survey of the literature related to task offloading was presented in [16]. They discussed different types of algorithms used to distribute and offload programs to save energy or improve performance and described why computation offloading is important for limited resources devices. In [17], the authors introduced a detailed definition of edge computing and mentioned the reasons behind its efficiency for some computing services. The authors in [18] discussed the importance of determining whether a task offloading is useful or not and introduced a fog computing model and an offloading policy. A fog framework and an offloading policy was proposed to offload IoT requests to the appropriate destination, which may be either the fog node or the cloud, aiming to reduce the delay of service [19].

Energy-saving is a very important aspect in the VFC systems; therefore, several research works have addressed reducing the energy consumption in the IoV systems. In [20], the authors focused on improving energy consumption and achieving the user Quality of Service (QoS) to enhance the performance of the fog system. They proposed a balanced energy-delay solution by employing Evolutionary Algorithms (EA) to scrutinize delay and consumed power in both fog-cloud computing and conventional cloud computing. A fog-cloud computational offloading algorithm in IoV was proposed for minimizing the vehicles power consumption and the power of computational facilities. The proposed model excelled in the cloud-only model and also in the fog-only model [21]. The authors in [22] concentrated on the control issues of energy consumption for mobile edge computing (MEC)-enabled RSUs. They constructed an energy-efficient scheduling framework for MEC-enabled IoV to reduce the energy consumption of RSUs taking into consideration the task latency constraint.

On the other hand, several research works have been conducted to reduce the latency and achieve a real-time response. In [23], the authors proposed a framework of MEC-based vehicular networks and discussed the consumed time and the offloading cost in various transmission modes. The authors demonstrated the importance of fog computing for time-sensitive applications and proposed a framework for allocating resources to reduce the latency using the genetic algorithm (GA) [12]. In [13], Fair Task Offloading (FTO) scheme, which selects an appropriate fog node for task offloading according to the fairness metric and a rule, was proposed for minimizing the task delay.

The authors in [24] designed an offloading algorithm for response time reduction in fog-based IoV systems. They used both moving and parked vehicles nearby RSUs as fog nodes. likewise in [25], vehicles near RSUs were used to act as fog nodes in VFC systems for better connectivity and promoting computational performance.

In [26], a multi-objective problem with various constraints was formulated and addressed by using of Interior Point Method(IPM)-based algorithm. A scheme was proposed to optimize both offloading probability and transmit power for mobile devices to minimize the energy consumption, delay performance, and payment cost. In [27], the authors formulated a problem of energy efficient optimization with the aim of reducing energy consumption giving regard to restrictions of execution delay. They proposed an algorithm of alternating direction method of multipliers (ADMM)-based distributed to address this problem. The authors in [14] proposed a multi-objective computation offloading method (MOC) to minimize the energy consumption and the task execution time with the constraints of the load balancing and the ensuring of the IoV data trustworthiness. Although the proposed method achieved improvement in energy consumption, it was unsuccessful in reducing time of transmission.

In [28], the authors designed a method of computation offloading for IoV under the 5G networks architecture to solve the problem of multi-objective optimization in order to reduce the delay and cost of offloading. This method determines which edge node is appropriate to be the destination of a vehicle application, taking into consideration the load balance in the edge nodes.

Although the current research work achieved acceptable performance in the field of task offloading optimization, the best transmission path (which ensures low time and energy consumption) still requires further investigation. This article implements and analyzes most of the possible transmission paths to formulate idea better understanding on how to choose the best path for a specific application. This work's target is to reduce the energy consumed and the response time required to complete the generated tasks by offloading and balancing the tasks among RSU and vehicles. Four transmission/processing models for the VFC system, named RSUs-Vs, RSUs-Vs/Vs, RSUs-Vs/RSUs, and RSUs-Only, are proposed and analyzed to find the best offloading that guarantees the low latency and energy.

III. System Architecture and Proposed Models

In this section, the layers of the system model are presented, then, the proposed execution/transmission models are described in detail. The proposed system model consists of three layers; the central management layer, the distributed RSUs layer, and the layer of the vehicular network. These layers are described as follows:

1) Central management layer: consists of MBS, which is positioned at the midst of the road, such that its coverage area is large enough that all vehicles can have access to it. The MBS can determine the appropriate strategy for task offloading based on two databases that are updated periodically. The first database contains information about RSUs (RSU's ID, coverage area, computational capability, ready time, channel state, etc.). The second database contains information about vehicles (vehicle's ID, position, velocity, processing ability, etc.).

2) Distributed RSUs layer: a group of RSUs $R = \{rsu_1, \dots, rsu_m , \dots, rsu_M\}$ with similar coverage areas and different specifications are distributed along the road. Periodically, they send information messages to the MBS to update their states. Based on the coverage area of RSUs, the road is divided into M segments with similar areas where one RSU is positioned at the center of each segment.

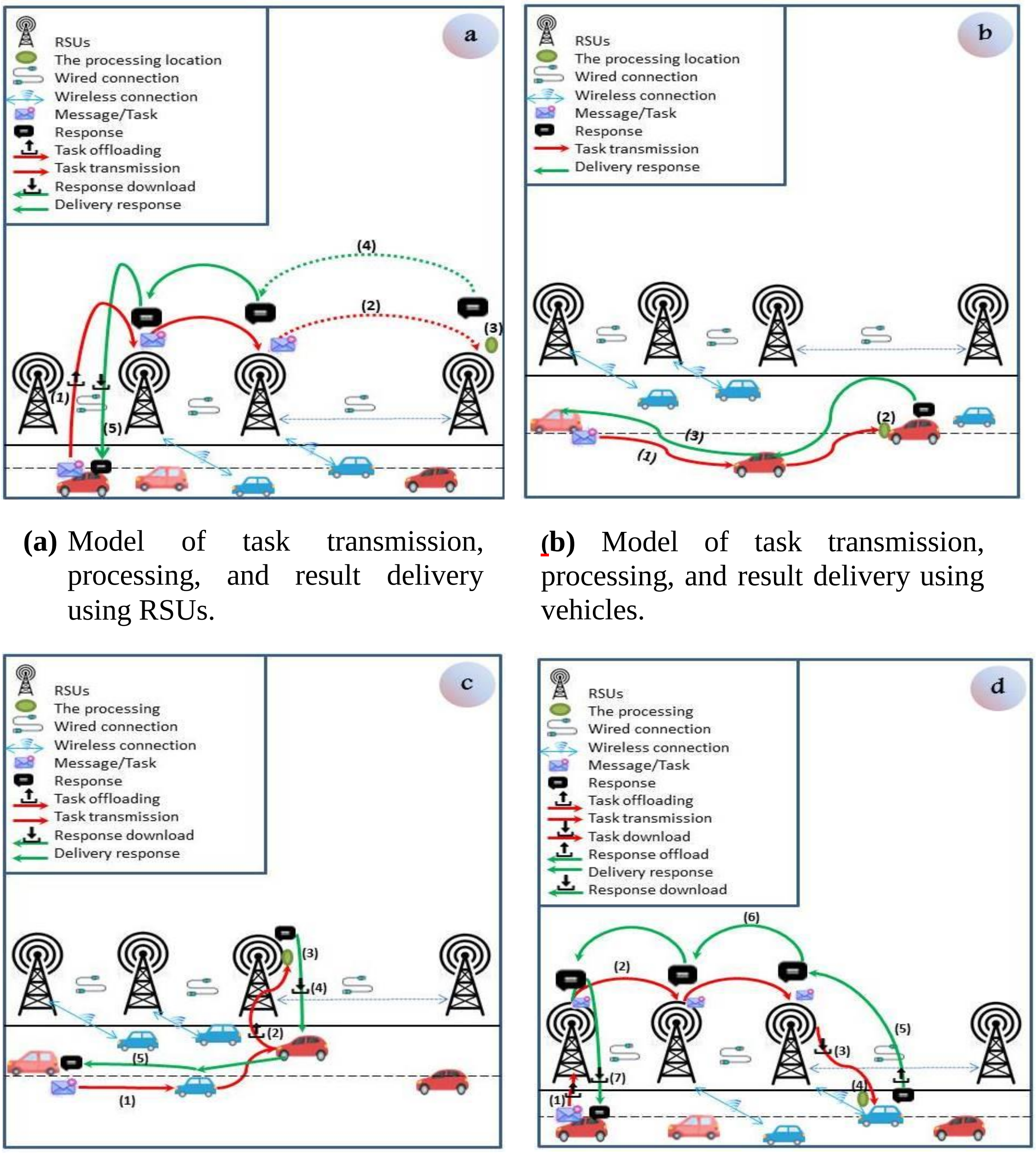
3) Vehicular network layer: in this layer, the tasks are generated, it encompasses a set of vehicles $V = \{v_1 \dots, v_n, \dots, v_N\}$ with different specifications distributed along the uni-direction road. All vehicles send information about their current state to MBS periodically. Each vehicle v_n generates a task t_n, which is characterized by I_n and D_n, where I_n denotes the data size of task (bits), and D_n is the maximum tolerable delay (Sec.).

In the proposed system, when a task t_n is generated by a vehicle v_n, the latter asks the MBS for the best execution node and the best transmission path to the destination node by sending the computational requirement of the task ($I_n$and D_n). Then, MBS assigns the task either to an appropriate RSU (*target*) or to a proximate vehicle (*goal*) as the task is processed with minimum consumed energy and minimum latency. After that, the MBS will inform the vehicle v_n with the decision of task offloading by sending the ID of the destination node (*target* or *goal*) and the path for delivering the task to the destination, in addition to the path for delivering the result to concerned vehicle v_n. In the proposed system, both RSUs and vehicles are involved in the task transmission and processing, which leads to generating various transmission and execution models These models will be described in detail below:

- **RSUs-Vs model:** using RSUs and vehicles for tasks execution, the path to transfer the task is chosen according to the place of its execution. If the destination is an RSU, the task will be transmitted over RSUs as shown in Figure 1 (a), while if the destination is a vehicle, the task will be transmitted over the vehicles as shown in Figure 1 (b).
- **RSUs-Vs/Vs model:** using RSUs and vehicles for tasks execution, while the tasks are transmitted over vehicles as shown in Figures 1 (c) and (b).

The Dijkstra's algorithm is used to find the path for task transmission over vehicles with minimum latency and energy. Considering the vehicle velocity and the time required to complete the task, the MBS determines the vehicles' locations after the task execution is finished for sending the result and informs the destination node the path for delivering the result.

- **RSUs-Vs/RSUs model:** using RSUs and vehicles for tasks execution, while tasks are transmitted over RSUs as shown in Figures 1 (a) and (d).
- **RSUs-Only model:** using RSUs for tasks execution and transmission as shown in Figure 1 (a).



(a) Model of task transmission, processing, and result delivery using RSUs.

(b) Model of task transmission, processing, and result delivery using vehicles.

(c) Model of task and result transmission using vehicles, task processing using RSUs.

(d) Model of task and result transmission using RSUs, task processing using vehicles.

Figure 1-Models of task transmission and result delivery.

IV. Task Offloading Problem Formulation and MOEA/D Algorithm

Generally, real-world problems require satisfying multiple objectives at the same time. However, the optimization of one objective leads predominantly to the retrogradation of one extra objective at least. The introduced problem of task offloading optimization in this paper combines two antithetic objective functions to provide the VFC system with a set of solutions. Each solution can perfectly map the tasks to the proper vehicles and RSU nodes, so that the required objectives are satisfied. This section presents how the MOEA/D algorithm is adopted for the task offloading problem. The important notations of this paper are included in Table-1.

Table 1- Important notations of MOEA/D algorithm and the system model

Notation	Description
Ɲ	Individual
Þ	Initial population
Ç	Crossover Probability
ɱ	Mutation Probability
Ŕ	Probability of choosing the fog node for processing
IL	Individual length
Z	Population size
M	The number of RSUs in the system
N	The number of vehicles and tasks
$\mathcal{R}$	The group of RSUs $\mathcal{R} = \{rsu_1, \dots, rsu_m, \dots, rsu_M\}$
T	The set of tasks $T = \{t_1, \dots, t_n, \dots, t_N\}$
$\mathcal{V}$	The set of vehicles $\mathcal{V} = \{v_1, \dots, v_n, \dots, v_N\}$
B^{I2V}, B^{V2I}	Bandwidth of I2V channel, Bandwidth of V2I channel
B^{RSU}, B^{V}	Bandwidth of RSUs channel, Bandwidth of vehicles channel
C_m^{RSU}, C_n^{V}	Computational capacity of rsu_m, Computational capacity of v_n
I_n, D_n	Data size of the task t_n, Maximum tolerable delay of t_n
I'_n	The result data size of the task t_n
$E^{download}$	The downloading energy consumption of a task
$E^{offload}$	The offloading energy consumption of a task
TE^{RSU}	The transmission energy consumption of an RSU
PE_m^{RSU}	The processing energy consumption of rsu_m
TE^{V}	The transmission energy consumption of a vehicle
PE_n^{V}	The processing energy consumption of v_n
W^{RSU}, W^{V}	Wireless communication of RSUs, wireless communication of vehicles

Recently, MOEA/D as a general-purpose algorithm framework has achieved magnificent success in the field of evolutional multi-objective optimization and has attracted a lot of interest. It decomposes a multi-objective optimization problem (MOP) into a set of sub-problems of single-objective optimization (or problems of simple multi-objective optimization) using uniformly distributed weight vectors of aggregation. It then uses a search heuristic to simultaneously and cooperatively optimize these sub-problems/problems [29]. The MOEA/D algorithm is utilized for the task offloading problem, such that, each individual in the meta-algorithm adopted in this study is represented as a vector with a length equals to the number of generated tasks. For simplicity, each vehicle generates one task at a time, thus the total number of generated tasks equals the total number of vehicles, and they are used interchangeably in the coming sections. Each gene represents a task generated by a vehicle, while its content identifies where this task will be executed.

A. Initial Population

In the RSUs-Only model, the formation of the initial population can be derived as shown in Eq. 1:

$$\text{Þ}_{\boldsymbol{p},\boldsymbol{g}} = \boldsymbol{rand}_{[\boldsymbol{1},\boldsymbol{M})} \;, \tag{1}$$

where $p \in \{1,2,\dots,Z\}$, $g \in \{1,2,\dots,IL\}$, and Þ ={ Ɲ^1, Ɲ^2,…, Ɲ^Z}.

Ɲ^{z} represents the zth chromosome in the initial population. In RSUs-Vs, RSUs-Vs/Vs, and RSUs-Vs/Vs models, the formation of initial population can be derived as shown in Eq. 2:

$$Þ_{\boldsymbol{p},\boldsymbol{g}} = \boldsymbol{rand}_{[\mathbf{1},\boldsymbol{\mu})} \ , \tag{2}$$

where $\mu = M + N \ , \ \mu \neq M + g$.

A random value is generated for each gene, If the random value is less or equal to the probability of choosing the fog node for processing Ŕ, the gene content will be the ID of an RSU. If the generated random value is greater than Ŕ, the gene content will be the ID of a vehicle.

B. Objective Function Evaluation

In general, the meta-heuristic algorithms satisfy single or multi-objective requirements based on the case at hand. As previously indicated, the target of the proposed VFC system is to minimize both latency and the total energy consumed . The objective function gauges the quality of each individual as follows:

B.1 Latency Measurement

The latency can be defined as the total amount of time required for the task completion, which consists of task transmission, task waiting time in the queue, task execution, and task result feeding back [14]. Before deriving these terms, it is necessary to define the following identifiers:

vsrc: is the source vehicle that generates the task.
vdst: is the vehicle where the task is executed.
rdst: is the RSU where the task is executed.
snrst: is the nearest RSU to *vsrc*.
rdnrst: is the nearest RSU to *vdst*.
vdnrst: is the nearest vehicle to *rdst*.

Using these identifiers, the main terms of the latency objective are derived as follows:

The offloading time ($Toffl$) is the time required to offload the task t_n from the source vehicle (*vsrc*) to the nearest RSU (*snrst*), and can be expressed as shown in Eq. 3:

$$\boldsymbol{Toffl}^{\boldsymbol{t_n}} = \frac{\boldsymbol{I_n}}{\boldsymbol{B^{V2I}}} \tag{3}$$

If the destination node (*rdst* or *vdst*) is far from *vsrc*, the task should be transmitted, over RSUs or vehicles, based on the selected VFC model. The transmission time ($Ttrans$) can be generally expressed as shown in Eq. 4:

$$\boldsymbol{Ttrans}^{\boldsymbol{t_n}}_{\boldsymbol{src2dst}} = \sum_{\mathbf{1}}^{\boldsymbol{H}} \frac{\boldsymbol{I_n}}{\boldsymbol{BW}} \tag{4}$$

where *src* and *dst* are either RSUs or vehicles, H represents the total hop count between *src* and *dst* nodes, and BW is the bandwidth of RSUs (B^{RSU}) or vehicles (B^{V}) that are responsible for transmitting the task.

The required time for the task completion encompasses the waiting time ($Twait$), which is the time that a task must wait until the resources get ready, plus the time consumption for the task execution, which are expressed as shown in Eq. 5:

$$\boldsymbol{Texec}^{\boldsymbol{t_n}}_{\boldsymbol{dst}} = \frac{\boldsymbol{I_n}}{\boldsymbol{C_{dst}}} \tag{5}$$

$$\boldsymbol{Twait}^{\boldsymbol{t_n}}_{\boldsymbol{dst}} = \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{td}} \boldsymbol{Texec}^{\boldsymbol{t_i}}_{\boldsymbol{dst}} \tag{6}$$

Where $\boldsymbol{C_{dst}}$ represents the computation capability of the destination node, which is either an RSU ($\boldsymbol{C^{RSU}_{rdst}}$) or a vehicle ($\boldsymbol{C^{V}_{vdst}}$), and $\boldsymbol{td}$ represents the number of tasks in the queue of the destination node that are waiting for their turn to be executed.

The task result $\boldsymbol{t'_n}$ should be transmitted to *vsrc* over RSUs or vehicles. Before that, the new location of vehicle after $\boldsymbol{t_n}$ is finished should be found to determine the transmission path of $\boldsymbol{t'_n}$, which may be the same path of $\boldsymbol{t_n}$ transmitting or may change. The vehicle location is determined depending on its location when it offloaded the task, the task transmission time, the task waiting time, the task execution time, the task result feeding back time, and the vehicle velocity.

If t_n is executed by an RSU, t'_n should be transmitted to the nearest RSU for the new location of *vsrc*. The downloading time to the *vsrc* can be expressed as shown in Eq.7. If t_n is executed by a vehicle, t'_n is transmitted normally to *vsrc* without any need for downloading process.

$$\boldsymbol{Tdown^{t'_n} = \frac{I'_n}{B^{I2V}}} \tag{7}$$

Accordingly, the total time for the proposed models can be obtained as follows:

a) RSUs-Vs model

In this model, if the task t_n is executed by an RSU, the total time consists of offloading t_n to *snrst*, transmitting t_n to *rdst*, waiting for resources to get ready, executing t_n, transmitting t'_n to *snrst*, downloading t'_n to *vsrc*. This can be expressed as shown in Eq. 8.

$$\boldsymbol{Ttotal^{t_n}_{rdst} = Toffl^{t_n} + Ttrans^{t_n}_{snrst2rdst} + Twait^{t_n}_{rdst} + Texec^{t_n}_{rdst} + Ttrans^{t'_n}_{rdst2snrst} + Tdown^{t'_n}}. \tag{8}$$

While if t_n is executed by a vehicle the total time consists of transmitting t_n to *vdst*, waiting for resources to get ready, executing t_n, and transmitting t'_n back to *vsrc* as shown in Eq. 9.

$$\boldsymbol{Ttotal^{t_n}_{vdst} = \; Ttrans^{t_n}_{vsrc2vdst} + Twait^{t_n}_{vdst} + Texec^{t_n}_{vdst} \; Ttrans^{t'_n}_{vdst2snrst}} \tag{9}$$

b) RSUs-Vs/Vs model

In this model, if the task t_n is executed by an RSU, the total time consists of transmitting t_n to *vdnrst*, offloading t_n to *rdst*, waiting for resources to get ready, executing t_n, downloading t'_n to *vdnrst*, and transmitting t'_n to *vsrc*. This can be expressed as shown in Eq. 10:

$$\boldsymbol{Ttotal^{t_n}_{rdst} = Ttrans^{t_n}_{vsrc2vdnrst} + Toffl^{t_n} + Twait^{t_n}_{rdst} + Texec^{t_n}_{rdst} + Tdown^{t'_n} + Ttrans^{t'_n}_{vdnrst2vsrc}} \tag{10}$$

While if t_n is executed by a vehicle the total time consumption can be obtained from Eq.9.

c) RSUs-Vs/RSUs model

In this model, if the task t_n is executed by an RSU, the total time consumption can be obtained from Eq.9. If t_n is executed by a vehicle, the total time consists of offloading t_n to *snrst*, transmitting t_n to *rdnrst*, downloading t_n to *vdst*, waiting for resources to get ready, executing t_n, offloading t'_n to *rdnrst*, transmitting t'_n to *snrst*, and downloading t'_n to *vsrc*. This can be expressed as shown in Eq. 11:

$$\boldsymbol{Ttotal^{t_n}_{rdst} = Toffl^{t_n} + Ttrans^{t_n}_{snrst2Vrdnrst} + Tdown^{t_n} + Twait^{t_n}_{vdst} + Texec^{t_n}_{vdst} + Toffl^{t'_n} + Ttrans^{t'_n}_{rdnrst2snrst} + Tdown^{t'_n}} \tag{11}$$

d) RSUs-Only model

In this model, the total time consumption can be obtained from Eq.8.

To get the accurate minimum time consumption for one task, the delivery time, the execution time, and the feedback time for each fog node must be calculated whether it was an RSU or a vehicle. For RSUs-Vs, RSUs-Vs/V, and RSUs-Vs/RSUs models, the following objectives should be satisfied:

$$\boldsymbol{min \sum_{i=1}^{N} \sum_{j=1}^{u} Ttotal^{t_i}_{j}}, \tag{12}$$

$$s.t.\begin{cases} \boldsymbol{Eq.(8), Eq.(9)}\ \mathbf{RSUs-Vs\ model}, \\ \boldsymbol{Eq.(10), Eq.(9)}\ \mathbf{RSUs-Vs/Vs\ model}, \\ \boldsymbol{Eq.(8), Eq.(11)}, \mathbf{RSUs-Vs/RSUs\ model}, \\ \boldsymbol{u = N + M}, \\ \boldsymbol{Ttotal_j^{t_i} \leq D_i}. \end{cases} \tag{13}$$

While in the RSUs-Only model, the following objective should be satisfied:

$$\boldsymbol{min \sum_{i=1}^{N} \sum_{j=1}^{M} Ttotal_j^{t_i}}, \tag{14}$$

$$s.t.\begin{cases} \boldsymbol{Eq.(8), RSUs -}\ \mathbf{only\ model}, \\ \boldsymbol{Ttotal_j^{t_i} \leq D_i}. \end{cases} \tag{15}$$

B.2 Energy Measurement

The total energy consumption is the total amount of energy that is consumed for a task completion, which consists of transmission energy, execution energy and result feedback energy [14]. These terms can be generally expressed as follows:

The offloading energy ($Eoffl$) is the energy consumption for offloading t_n to the nearest RSU, and can be expressed as:

$$\boldsymbol{Eoffl^{t_n} = E^{offload} \times Toffl^{t_n}}. \tag{16}$$

The transmission energy ($Etrans$) is the energy consumption for transmitting t_n to the destination node over RSUs and vehicles, can be expressed as shown in Eq. 17:

$$\boldsymbol{Etrans_{src2dst}^{t_n} = TE \times Ttrans_{src2dst}^{t_n}} \tag{17}$$

where *src* and *dst* are the source and destination nodes which can be either an RSU or a vehicle,

and TE is the energy consumed by either RSUs (TE^{RSU}) or vehicles (TE^{V}) for transmission.

The execution energy ($Eexec$) is the energy consumption for executing t_n by RSUs or vehicles, can be expressed by Eq. 18:

$$\boldsymbol{Eexec_{dst}^{t_n} = EE \times Texec_{dst}^{t_n}} \tag{18}$$

where dst is the node where the task gets executed, and EE is the energy consumed by either RSU (EE_{rdst}^{RSU}) or vehicle (EE_{vdst}^{V}) for processing.

The downloading energy can be expressed as shown in Eq.19. If t_n is executed by a vehicle, t_n' is transmitted to *vsrc*.

$$\boldsymbol{Edown^{t_n'} = E^{download} \times Tdown^{t_n'}} \tag{19}$$

The total energy consumption for the proposed models can be obtained as follow:

a) RSUs-Vs model

In this model, the total amount of energy consumption for $\boldsymbol{t_n}$ execution and transmission by RSUs is calculated as shown in Eq. 20:

$$\boldsymbol{Etotal_{rdst}^{t_n} = offl^{t_n} + Etrans_{snrst2rdst}^{t_n} + Eexec_{rdst}^{t_n} + Etrans_{rdst2snrst}^{t_n'} + Edown^{t_n'}} \tag{20}$$

While the total amount of energy consumption for $\boldsymbol{t_n}$ execution and transmission by vehicle is calculated as shown in Eq. 21:

$$\boldsymbol{Etotal_{vdst}^{t_n} = Etrans_{vsrc2vdst}^{t_n} + Eexec_{vdst}^{t_n} + Etrans_{vdst2snrst}^{t_n'}} \tag{21}$$

b) RSUs-Vs/Vs model

In this model, the total energy consumption for $\boldsymbol{t_n}$ transmission to an RSU over vehicles is calculated by Eq. 22:

$$\boldsymbol{Etotal_{rdst}^{t_n} = Etrans_{vsrc2vdnrst}^{t_n} + Eoffl^{t_n} + Eexec_{rdst}^{t_n} + Edown^{t_n'} + Etrans_{vdnrst2vsrc}^{t_n'}} \tag{22}$$

While the total energy consumption for $\boldsymbol{t_n}$ execution and transmission by vehicles can be obtained from Eq.21.

c) RSUs-Vs/RSUs model

In this model, the total amount of energy consumption for $\boldsymbol{t_n}$ execution and transmission by RSUs can be obtained from Eq.2.20. The total energy consumption for $\boldsymbol{t_n}$ transmission by RSUs to a vehicle is calculated as shown in Eq. 23:

$$\boldsymbol{Etotal^{t_n}_{rdst}} = \boldsymbol{Eoffl^{t_n}} + \boldsymbol{Etrans^{t_n}_{snrst2rdnrst}} + \boldsymbol{Edown^{t_n}} + \boldsymbol{Eexec^{t_n}_{vdst}} + \boldsymbol{Eoffl^{t'_n}} + \boldsymbol{Etrans^{t'_n}_{rdnrst2snrst}} + \boldsymbol{Edown^{t'_n}} \qquad (23)$$

d) RSUs-Only model

In this model, the total energy consumption for t_n execution and transmission by RSUs can be obtained from Eq.20.

To obtain the minimum energy consumption for the execution of one task. For each fog node (an RSU or a vehicle), the energy consumed during task delivery, the task execution, and the delivery of the result to the vehicle that generated the task must be calculated. For the RSUs-Vs, RSUs-Vs/V, and RSUs-Vs/RSUs models, the following objectives should be satisfied as shown in Eq. 24:

$$\boldsymbol{min \sum_{i=1}^{N} \sum_{j=1}^{u} Etotal_j^{t_i}}, \qquad (24)$$

$$\boldsymbol{s.t.} \begin{cases} \boldsymbol{Eq.(20), Eq.(21),} \textbf{RSUs} - \textbf{Vs model} \\ \boldsymbol{Eq.(22), Eq.(21),} \textbf{RSUs} - \textbf{Vs/Vs model} \\ \boldsymbol{Eq.(20), Eq.(23),} \textbf{RSUs} - \textbf{Vs/RSUs model} \\ \boldsymbol{u = N + M,} \\ \boldsymbol{Ttotal_j^{t_i} \le D_i.} \end{cases} \qquad (25)$$

While in the RSUs-Only model, the following objective should be satisfied as shown in Eq. 26:

$$\boldsymbol{min \sum_{i=1}^{N} \sum_{j=1}^{M} Etotal_j^{t_i}}, \qquad (26)$$

$$\boldsymbol{s.t.} \begin{cases} \boldsymbol{Eq.(20),} \textbf{RSUs} - \textbf{Only model} \\ \boldsymbol{Ttotal_j^{t_i} \le D_i.} \end{cases} \qquad (27)$$

Crossover Operation

This operation is the same for all models. It is responsible for forming a new individual (Child / $\text{Ɲ}^{\mathbf{z}}$) by integrating the genetic information of two elected solutions.

$$\text{Ɲ}^{\mathbf{z}} = \begin{cases} \text{Ɲ}^{\mathbf{1}} & \boldsymbol{rand_i} \le \text{Ç} \\ \text{Ɲ}^{\mathbf{2}} & \boldsymbol{otherwise} \end{cases}. \qquad (28)$$

C. Mutation Operation

The fundamental function that aids in reconnaissance of the entire search space and forbidding the population from falling in a local optimal solution. In this operation, a random modification to one or more gene values is made to generate a better solution from the original solution, which resulted from the crossover operation. In all proposed models, mutation operation depends on the value of the mutation probability ɱ. For each gene, a random value is generated firstly. After that, this generated value is compared with ɱ. Gene content will be replaced if and only if the random value is less than or equal to ɱ.

If the fog node, which processes the task, is an RSU, then the gene content is randomly replaced by another RSU. But if the fog node is a vehicle, then the gene content will be randomly replaced by another vehicle. After that, the generated solution is examined to confirm whether the gene replacing process had a positive affect and produced a better solution than the previous solution or not. If the resulting solution is better than the original solution and has achieved the deadline condition, the replacement will be approved. Otherwise, the original solution will be restored and the replacement process canceled.

V. Numerical Result

In order to validate the performance of the proposed models in the VFC system based on MOEA/D, these models and MOC model were implemented in MATLAB. Ten systems for each model with different entities and task specifications were used, such that, every system was executed ten times. It is necessary to mention that the population size was 100, the system generations was set also to 100, Ç, ɱ, and Ŕ were set to 0.3, 0.3, and 0.5 respectively. The simulation parameters are summarized in TABLE 2. The wireless coverage, bandwidth, and energy consumption of RSU and Vehicles are set according to the DSRC standard. Also, some other parameters are selected based on numerous experiences.

Table 2- Simulation parameters

Variable	Value	Variable	Value
Length of the road	3000 m	B^{V2I}	2 Mbps
M	6	C_n^V	0.5 – 1 GHz
N	20-60	$E^{offload}$	31.7 dBm
T	20-60	PE_n^V	36 – 40 dBm
B^{RSU}	27 Mbps	TE^{V}	14 dBm
C_m^{RSU}	1 – 4 GHz	W^V	250 m
$E^{download}$	31.7 dBm	$V_n^{velocity}$	30 - 60 Km/h
TE^{RSU}	33 dBm	I_n	10 Sec.
PE_m^{RSU}	43 – 49 dBm	D_n	1 – 256 KB
W^{RSU}	250 m	D_n'	≥ 0.5 KB
B^V	20 Mbps	B^{V2I}	2 Mbps

Figure 2 shows the average latency versus the number of vehicles for the proposed models and MOC model. It is noticeable that the average latency for RSUs-Vs and RSUs-Vs/Vs models are much lower than the other three models. The reason is that in RSUs-Vs model, the task is not offloaded to RSUs unless it is processed by an RSU, which led to a reduction in the required time and energy consumption for offloading and downloading. RSUs-Vs/Vs model exploits the computational capacity of vehicles for the task execution, so this model outperforms the MOC model. RSUs-Vs/RSUs and RSUs-Only models outperform the MOC model in terms of latency because the latter utilized vehicles for task transmission to RSUs, which led to increase the required time for the task delivering and result feeding back.

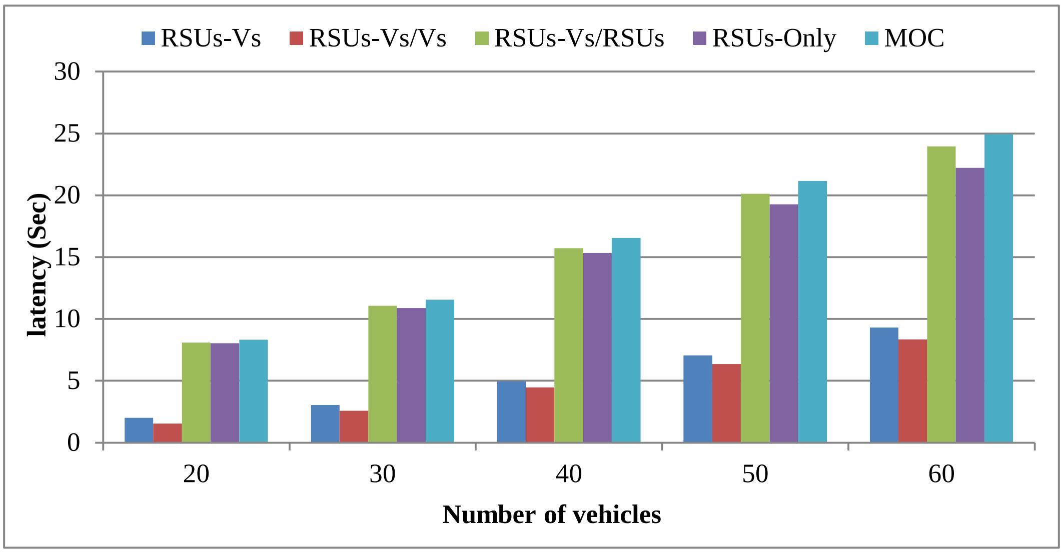


Figure 2-Comparison of the average latency of RSUs-Vs, RSUs-Vs/Vs, RSUs-Vs/RSUs, RSUs-Only and MOC models versus the number of vehicles

As mentioned previously, there are several different parts of time consumption Figure 3 and Figure 4 show that the time consumption for transmission and feeding back in RSUs-Vs and RSUs-Vs/Vs models is less than that in the other three models.

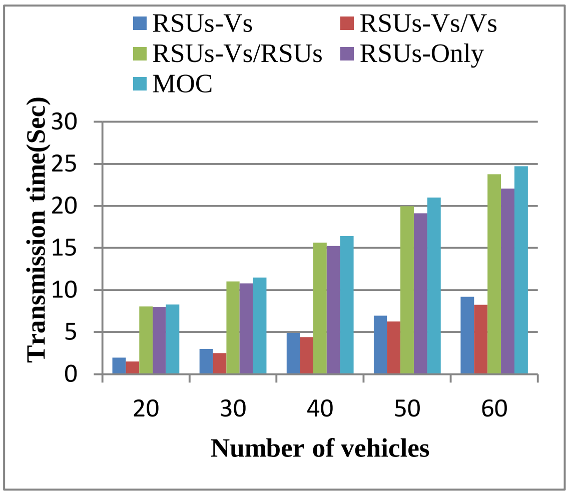


Figure 3-Time consumption of the task transmission.

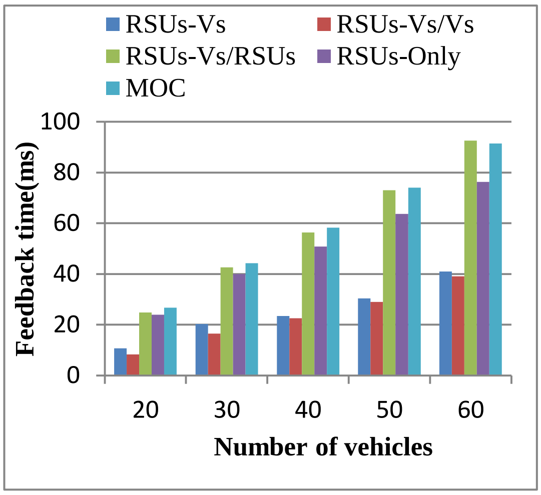
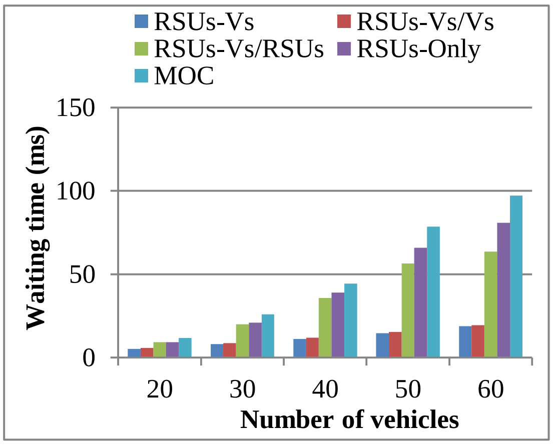


Figure 4-Time consumption of the result feeding back.

The time that is spent by a task in the queue is called waiting time. Figure 5 shows that the waiting time in RSUs-Only and MOC models increases with the number of vehicles. Due to the constant number of fog nodes (M), the increase of vehicles number leads to an increase in the load on RSUs. While the waiting time remains almost constant or change in a negligible small percentage in RSUs-Vs and RSUs-Vs/Vs models because the number of fog nodes increases with the number of vehicles $(M + N)$. Figure 6 shows that RSUs-Vs and RSUs-Vs/Vs models take more time than other models in task execution. This is because the computation capability of vehicles is less than that of RSUs. Therefore, a vehicle takes longer than the time taken by an RSU to execute the same task.



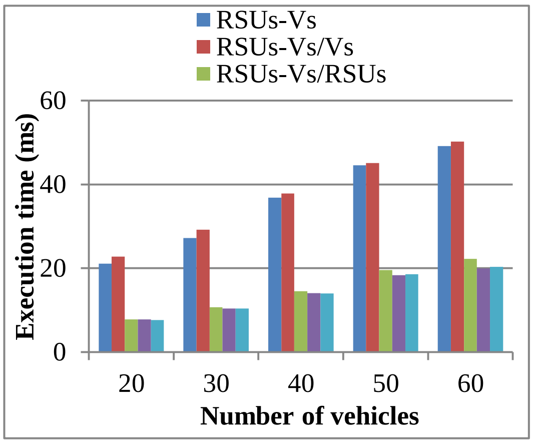


Figure 5-The amount of waiting time.

Figure 6-Time consumption for the task execution.

Figure 7 shows the average energy consumption versus the number of vehicles for the different five models. It is noticeable that the RSUs-Vs and RSUs-Vs/Vs consume much less energy than the other three models. As previously described, the total energy consumption consists of transmission energy consumption, execution energy consumption, and feeding back energy consumption.

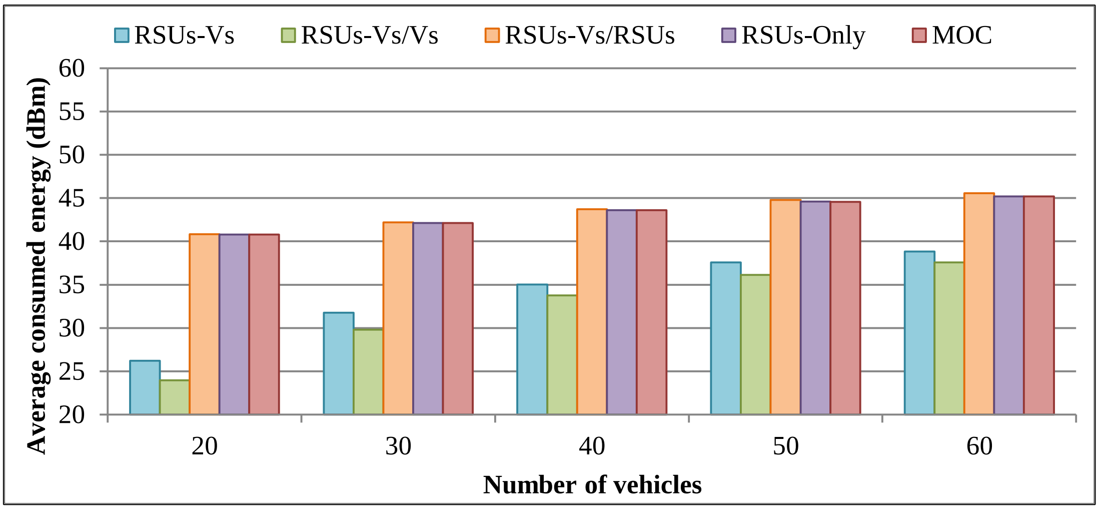
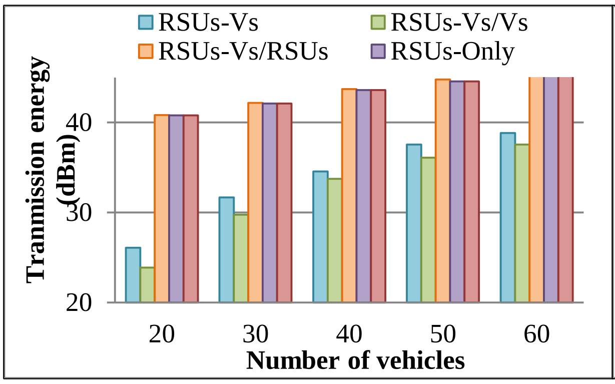


Figure 4-Comparison of the average energy of RSUs-Vs, RSUs-Vs/Vs, RSUs-Vs/RSUs, RSUs-Only and MOC models versus the number of vehicles

Figures 8,9, and 10 show that the energy consumption of transmission, execution, and feeding back increase monotonously with number of vehicles. However, it is clearly noticeable that RSUs-Vs and RSUs-Vs/Vs consume less energy than the other models during each of the three parts that compose the total energy consumption.



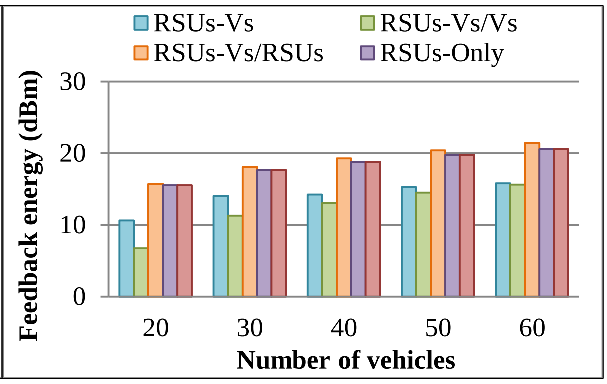


Figure 8-Transmission energy consumption.

Figure 9-Feeding back energy consumption.

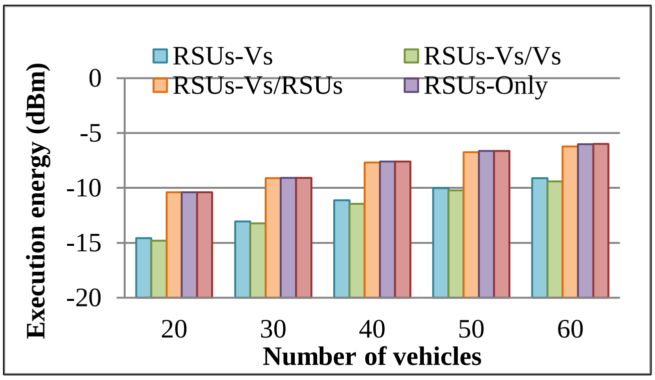


Figure 10-Execution energy consumption.

VI. Conclusions

In this paper, Multi-Objective Evolutionary Algorithm (MOEA) for task offloading optimization in Vehicular Fog Computing (VFC) systems is adopted to reduce the energy consumption and the task completion latency by considering the task transmission time and transmission energy. The vehicles resources are exploited for computation and transmission by employing these vehicles as fog nodes to reduce the total cost of energy consumed and achieve the response in a shorter time. Four different models of the VFC system are proposed. The numerical results prove that RSUs-Vs and RSUs-Vs/Vs models can effectively reduce task latency and the total amount of energy consumed . RSUs-Vs model can reduce latency and energy consumption by 72% and 22% on average respectively, while RSUs-Vs/Vs model is able to reduce latency and energy consumption by 90% and 25.8% respectively.

According to the obtained results, RSUs-Vs and RSUs-Vs/Vs models consume more execution and waiting time than the other models. On the other hand, RSUs-Vs/RSUs and RSUs-only models consume more energy than the other models. Enhancing the performance of such models in terms of latency and energy optimization is worth further investigation. A real dataset can be used to verify the performance of the proposed models and more criteria and constraints like cost and budget can be investigated.

References

[1] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on industrial informatics,* vol. 10, pp. 2233-2243, 2014.

[2] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of IoT: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things journal,* vol. 1, pp. 349-359, 2014.

[3] P. Suresh, J. V. Daniel, V. Parthasarathy, and R. Aswathy, "A state of the art review on the Internet of Things (IoT) history, technology and fields of deployment," in *2014 International conference on science engineering and management research (ICSEMR),* 2014, pp. 1-8.

[4] C.-M. Huang, M.-S. Chiang, D.-T. Dao, H.-M. Pai, S. Xu, and H. Zhou, "Vehicle-to-Infrastructure (V2I) offloading from cellular network to 802.11 p Wi-Fi network based on the Software-Defined Network (SDN) architecture," *Vehicular Communications,* vol. 9, pp. 288-300, 2017.

[5] S. Mehar, S. M. Senouci, A. Kies, and M. M. Zoulikha, "An optimized roadside units (RSU) placement for delay-sensitive applications in vehicular networks," in *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC),* 2015, pp. 121-127.

[6] J. W. Rittinghouse and J. F. Ransome, *Cloud computing: implementation, management, and security*: CRC press, 2016.

[7] I. M. Ali and M. I. Salman, "SDN-assisted Service Placement for the IoT-based Systems in Multiple Edge Servers Environment," *Iraqi Journal of Science,* pp. 1525-1540, 2020.

[8] M. N. Abdulredha, A. A. Bara'a, and A. J. Jabir, "Heuristic and Meta-Heuristic Optimization Models for Task Scheduling in Cloud-Fog Systems: A Review," *Iraqi Journal for Electrical And Electronic Engineering,* vol. 16, 2020.

[9] R. Z. Naeem, S. Bashir, M. F. Amjad, H. Abbas, and H. Afzal, "Fog computing in internet of things: Practical applications and future directions," *Peer-to-Peer Networking and Applications,* vol. 12, pp. 1236-1262, 2019.

[10] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Transactions on Vehicular Technology,* vol. 65, pp. 3860-3873, 2016.

[11] M. N. Abbas, A. A. Bara'a, and N. J. Kadhim, "Evolutionary Based Set Covers Algorithm with Local Refinement for Power Aware Wireless Sensor Networks Design," *Iraqi Journal of Science,* pp. 1959-1966, 2018.

[12] Y. Liu, J. E. Fieldsend, and G. Min, "A framework of fog computing: Architecture, challenges, and optimization," *IEEE Access,* vol. 5, pp. 25445-25454, 2017.

[13] G. Zhang, F. Shen, Y. Yang, H. Qian, and W. Yao, "Fair task offloading among fog nodes in fog computing networks," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1-6.

[14] X. Xu, R. Gu, F. Dai, L. Qi, and S. Wan, "Multi-objective computation offloading for internet of vehicles in cloud-edge computing," *Wireless Networks*, pp. 1-19, 2019.

[15] P. Liu, J. Li, and Z. Sun, "Matching-based task offloading for vehicular edge computing," *IEEE Access*, vol. 7, pp. 27628-27640, 2019.

[16] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile networks and Applications*, vol. 18, pp. 129-140, 2013.

[17] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, pp. 637-646, 2016.

[18] Q. Zhu, B. Si, F. Yang, and Y. Ma, "Task offloading decision in fog computing system," *China Communications*, vol. 14, pp. 59-68, 2017.

[19] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing IoT service delay via fog offloading," *IEEE Internet of Things Journal*, vol. 5, pp. 998-1010, 2018.

[20] A. Mebrek, L. Merghem-Boulahia, and M. Esseghir, "Efficient green solution for a balanced energy consumption and delay in the IoT-Fog-Cloud computing," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, 2017, pp. 1-4.

[21] X. Wang, X. Wei, and L. Wang, "A deep learning based energy-efficient computational offloading method in Internet of Vehicles," *China Communications*, vol. 16, pp. 81-91, 2019.

[22] Z. Ning, J. Huang, X. Wang, J. J. Rodrigues, and L. Guo, "Mobile edge computing-enabled Internet of vehicles: Toward energy-efficient scheduling," *IEEE Network*, vol. 33, pp. 198-205, 2019.

[23] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, vol. 12, pp. 36-44, 2017.

[24] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 4568-4578, 2018.

[25] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," *IEEE Wireless Communications*, vol. 26, pp. 87-93, 2019.

[26] [26]L. Liu, Z. Chang, X. Guo, S. Mao, and T. Ristaniemi, "Multiobjective optimization for computation offloading in fog computing," *IEEE Internet of Things Journal*, vol. 5, pp. 283-294, 2017.

[27] Z. Chang, Z. Zhou, T. Ristaniemi, and Z. Niu, "Energy efficient optimization for computation offloading in fog computing system," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, 2017, pp. 1-6.

[28] S. Wan, X. Li, Y. Xue, W. Lin, and X. Xu, "Efficient computation offloading for Internet of Vehicles in edge computing-assisted 5G networks," *The Journal of Supercomputing*, pp. 1-30, 2019.

[29] F. Wang, H. Zhang, Y. Li, Y. Zhao, and Q. Rao, "External archive matching strategy for MOEA/D," *Soft Computing*, vol. 22, pp. 7833-7846, 2018.