Hindawi
Journal of Sensors
Volume 2022, Article ID 4173346, 14 pages
https://doi.org/10.1155/2022/4173346

Research Article

AI-Enabled Energy-Efficient Fog Computing for Internet of Vehicles

Hira Tariq,[1] Muhammad Awais Javed,[1] Ahmad Naseem Alvi,[1] Mozaherul Hoque Abul Hasanat,[2] Muhammad Badruddin Khan,[2] Abdul Khader Jilani Saudagar,[2] and Mohammed Alkhathami[2]

[1]*Department of Electrical and Computer Engineering, COMSATS University Islamabad (CUI), Islamabad 45550, Pakistan*
[2]*Information Systems Department, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia*

Correspondence should be addressed to Mozaherul Hoque Abul Hasanat; mhhasanat@imamu.edu.sa

Received 16 March 2022; Revised 24 April 2022; Accepted 10 May 2022; Published 26 May 2022

Academic Editor: Han Wang

Future autonomous electric vehicles (EVs) are equipped with several IoT sensors, smart devices, and wireless adapters, thus forming an Internet of Vehicles (IoVs). These intelligent EVs are envisioned to be a promising solution for improving transportation efficiency, road safety, and driving experience. Vehicular fog computing (VFC) is an evolving technology that allows vehicular application-related tasks to be offloaded to nearby computing nodes and process them quickly. A major challenge in the VFC system is to design energy-efficient task offloading algorithms. In this paper, we propose an optimal energy-efficient algorithm for task offloading in a VFC system that maximizes the expected reward function which is derived using the total energy and time delay of the system for the computation of the task. We use parallel computing and formulate the optimization problem as semi-Markov decision process (SMDP). Bellman optimal equation is used in value iteration algorithm (VIA) to get an optimal scheme by selecting the best action for the current state that maximizes the energy-based reward function. Numerical results show that the proposed scheme outperforms the greedy algorithm in terms of energy consumption.

1. Introduction

Recently, autonomous and connected electric vehicles also known as the Internet of Vehicles (IoVs) have gained extensive attention and flourished as a promising technology by bringing convenience to society by solving the traffic issues like accidents [1], congestion, and environmental pollution [2]. By 2035, it is estimated that around 25% of autonomous vehicles will be on-road [3]. Nowadays, a large number of sensors, smart devices, and controllers are deployed in vehicles to facilitate the drivers and passengers for autonomous driving, infotainment, and natural language processing [4].

According to an estimate in 2020, every day around 4000 GB of data is produced by the vehicles [5]. Due to these smart sensors and controllers, the IoVs consume an enormous amount of power for the processing of data generated by the smart devices [6, 7]. As the vehicles have limited resources of energy and computational power, the computation of smart applications that cannot be managed locally needs to be offloaded to the helping nodes [8].

With the development in the technology of computation and communication [9], the vehicles can be both task producer and service provider nodes. This concept brings the computation capabilities near the proximity of task producer vehicles with the help of Vehicular to Vehicular (V2V) and Vehicular to Infrastructure (V2I) communication [10].

Vehicular fog computing is a novel technique for computing tasks and uses computational resources of both moving and parked vehicles [11]. The basic concept behind VFC is to install resource units (RUs) on connected vehicles so that these vehicles act as fog nodes and deliver their services of communication and computing, according to the

requirements [12]. To enable communication of tasks to the vehicular fog nodes, vehicles are armed with different types of network interface components. Vehicular fog nodes communicate with devices and the Internet via cellular network or IEEE 802.11p [13].

VFC is thus a proficient method for low latency tasks and smart IoV applications. However, vehicular fog nodes have limited computation capability of RU's and bandwidth of communication; hence, it is not possible to process all the tasks and satisfy computation latency requirements by the VFC. To solve this issue, vehicles are also connected to the cloud servers in the form of remote cloud (RC) using cellular communications [14]. Vehicles can transfer the tasks to the RC and avail the opportunity of powerful computational resources. However, long distance between RC and vehicles suffers from challenges of high latency and high-power consumption due to the transfer of tasks to the RCs. To assure reliable computing services to the EVs, the three-tiered VFC architecture is considered in this paper. This architecture includes task producer devices, vehicular fog nodes, and the remote cloud as shown in Figure 1 [15].

In this paper, we propose an optimal energy-efficient algorithm for task offloading in a VFC system that minimizes the energy consumption of the system. We propose an energy-based reward function for the considered problem and utilize parallel offloading. Vehicles with the tasks divide it into several parts and offload it to the neighboring vehicles for processing. One part of the task is computed locally, and the rest of the parts are offloaded to the remaining other cooperative vehicles such that the energy efficiency of the whole system is maximized.

The main contribution of our work is summarized as follows:

(i) We propose a novel reward function that considers the energy of the system for parallel task offloading

(ii) We formulate the problem of task offloading as semi-Markov decision process (SMDP) that consider factors such as (1) arrival of task, (2) departure of task, (3) arrival of the vehicle, and (4) departure of the vehicle

(iii) The state space, actions, reward, and transition probabilities of the VFC system are analyzed and defined to obtain the optimal policy, which determines the best action for the specific state for task offloading

(iv) We used the iterative algorithm to solve the optimal task offloading problem and increased the long-term expected reward in the form of saved energy and time

(v) We compare the results of the proposed technique with a greedy algorithm and show significant performance gains

The rest of the paper is organized as follows. Section 2 provides the related literature review. In Section 3, we describe the system model of the VFC system. In Section 4, we formulate the problem as an SMDP optimization problem. The solution to the optimization problem is given in Section 5 as an iterative algorithm. Section 6 presents the numerical results and analysis of the performance. Finally, in Section 7, conclusion and future work are presented.

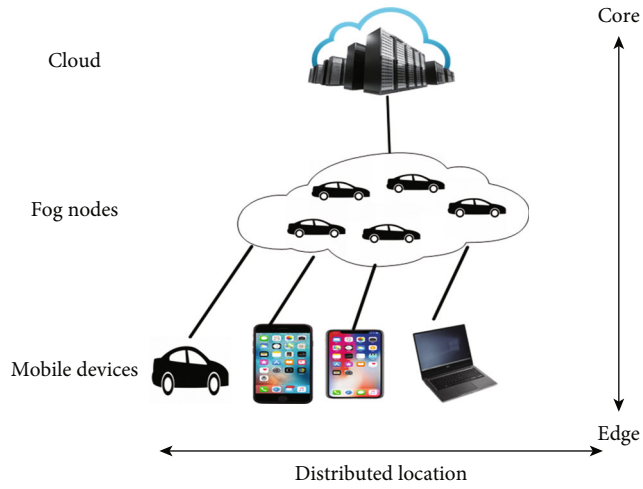

Figure 1: Basic three-tier architecture of VFC.

2. Related Work

In recent years, few works have been carried out to investigate the task offloading problem in vehicular fog computing. Wu et al. in [16] proposed a task offloading policy for the VFC and used IEEE 802.11p protocol for the transmission of tasks. Tasks are divided based on priorities according to the delay requirements. The problem is formulated as SMDP, and the task offloading scheme is presented to maximize the long-term reward in the form of reduction of processing time of a prioritized task. To solve this problem, iterative algorithm is used.

In [17], the authors improve the efficiency of application-aware offloading by proposing a VFC system in which public vehicles such as buses are being used as fog servers. A priority queuing system is applied to model the VFC for the application-aware delay requirements. The problem is formulated as SMDP. An application-aware task offloading policy is proposed to obtain the maximum long-term award of the VFC model.

In [18], a novel offloading scheme has been proposed to minimize the cost of energy consumption, failure in offloading, and service latency of the VFC network. At first, the overloaded cloudlet node has been determined, and then, an offloading policy has been introduced to determine which task will be offloaded and for the selection of vehicular node to place the offloaded task.

A game-theoretic approach can be used to overcome the demand-resource mismatch by minimizing the usage of resource energy and reducing the response time. The proposed resource allocation model has outperformed the state-of-the-art VFC models in terms of improved performance for the vehicles by keeping reducing the energy consumption [19].

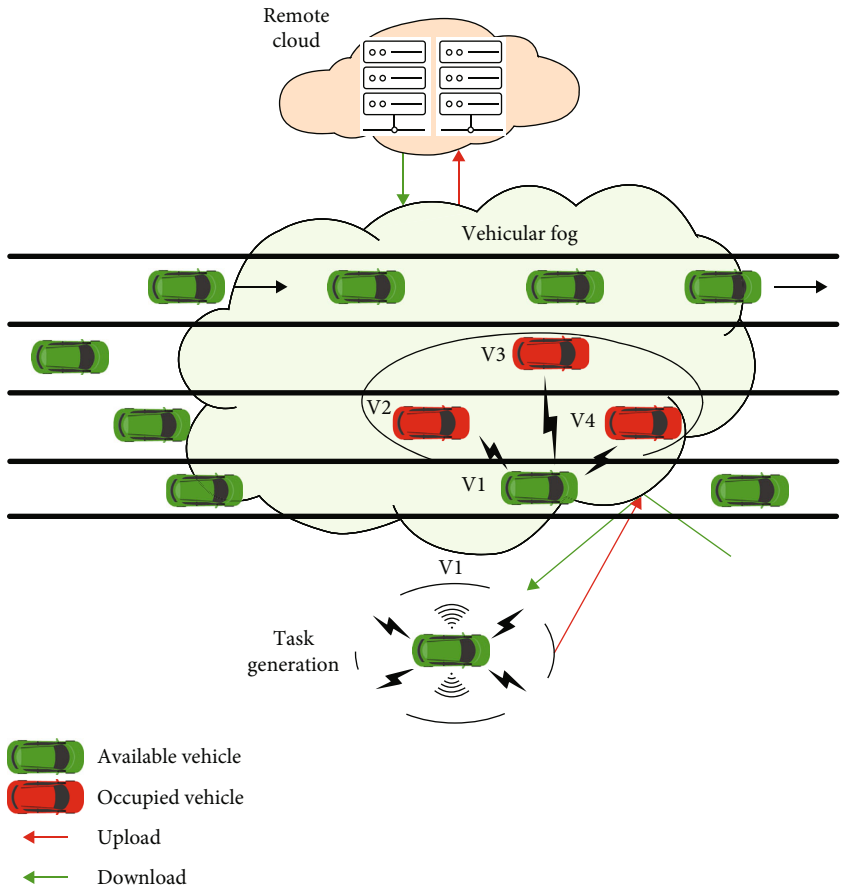

Figure 2: System model.

A distributed scheduler scheme for the energy consumption minimization for computing has been presented in [20]. The proposed model maximizes the efficiency of the system as well as maintains the quality of service. In [21], the authors show that the long-term reward of the VFC system can be increased by using an optimal task offloading strategy that uses the computation and transmission delays and available RUs. An iterative algorithm is used to solve the problem which is formulated as SMDP.

The work in [22] proposed the concept of VFC in which the electric vehicles are used as fog nodes and used to save energy for mobile devices. They used the Markov decision process to formulate the resource allocation problem and used dynamic programming to solve this problem. In [23], the authors proposed an efficient incentive mechanism based on the contrast matching approach for the problems of task offloading and computational resource allocation problems in the VFC system. By this approach, the base stations offload the task to nearby vehicles and minimize the task delays by using the underutilized resources of the vehicles.

The work in [24] proposed a deep dynamic reinforcement learning algorithm by exploiting the Markov decision process. They used reinforcement learning to get an offloading decision that minimizes the cost of the VFC system consisting of consumption of energy and service delay. In [25], Liu et al. presented a three-layered architecture for service offloading in the VFC system consisting of vehicular fog, fog server, and central cloud. They formulated the probabilistic task offloading problem to minimize the energy consumption payment cost and the execution delay and solved it by the iterative coordination process.

In [26], a task offloading policy for the VFC system is proposed that considers the task priority, vehicle mobility, and the availability of service by the vehicles. The priority-based task offloading policy was formulated as MDP and solved by a soft actor-critic-based deep reinforcement learning algorithm to maximize the entropy of policy and the reward.

To achieve the benefits prevailed by vehicular fog computing (VFC), the authors in [27] have presented a three-layer VFC model to minimize the response time of the vehicles. The problem has been formulated as a real-time optimization problem for the effective management of decentralized traffic.

Different from the work in the literature that studied the task offloading scheme in VFC system, we develop an energy-efficient task offloading scheme in VFC system to maximize the long-term reward by using the parallel computing and computing one part of the task request locally and offloading the remaining task to the vehicular fog nodes.

We utilize vehicular fog computing model in [21] as basis of our work; however, there are two major differences from the previous work. The first difference is the consideration of local computing at the task-generating vehicle as well as remote cloud computing whereas [21] only considers resources from other cooperative vehicles. The second

Table 1: List of important notations used in the paper.

Notations	Description
K	Maximum vehicles in the VFC system
R	Available RUs
M_R	Maximum number of RUs that can be allotted to a task request
S	Maximum number states
λ_c	Vehicle arrival rate
μ_c	Vehicle departure rate
λ_h	Task request arrival rate
μ_h	Task request service rate
n_i	Total request tasks serviced by i RUs
T	Task request arrival
D_i	The departure of the task request serviced by i number of RUs
C_{+1}	Vehicle's arrival
C_{-1}	Departure of vehicle
ω_e	Income weightage of energy
ω_d	Income weightage of time
β_e	Price per energy saving
β_d	Price per delay saving
ζ	Cost for transmission of task request
d_c	Transmission time between VFC to RC
d_f	Transmission time between requested vehicles to VFC
I	Total income of system
P_c	Computation power of RUs
P_t	The transmission power of vehicles
η	Punishment to the VFC system due to vehicle departure
γ	Discount factor
α	The factor of continuous discount
ϵ	Threshold value
θ	Convergence rate

difference is that the focus of our work is on energy efficiency, and hence, we propose a new reward function that considers energy consumption of the vehicular computing node.

3. System Model

In the present section, we present the system model inspired by [21] as shown in Figure 2. For computing, VFC is a recent paradigm that has been furnished several applications that required high computation and time-critical applications by offering the computing resources for the processing. All the vehicles, in the VFC, have a processor with a RU and also a source of a task; i.e., the vehicles can offload their computation tasks between each other. Since the arrival and departure of the vehicles from the VFC are random, the resources of computation change randomly in the VFC system. We assumed that all vehicles have the same virtualized RUs, and also, they are conscious of the accessible RUs in the system through communication with other vehicles in real time. When a task request arrives at the system, it has to be decided whether to accept this request or transfer it to the remote cloud (RC) according to the availability of the resources. If the request is accepted by the VFC system, the system decides to allocate the number of RUs according to available RUs.

For the illustration, we consider an example present in Figure 2. Vehicle $V1$ generates a task and is accepted by the VFC system as there are sufficient available resources; the task is then divided into four equal subtasks; one part is computed by $V1$ itself and the remaining three parts are offloaded to the three RUs, i.e., $V2$, $V3$, and $V4$. If there are no available RUs, the task is transferred to the RC for the computation. After the computation, the result is feedback to the $V1$. The vehicles arrive and depart the system according to the Poisson process. The arrival rate of the vehicle is λc, and the departure rate of the vehicle is μc. The maximum number of vehicles that the

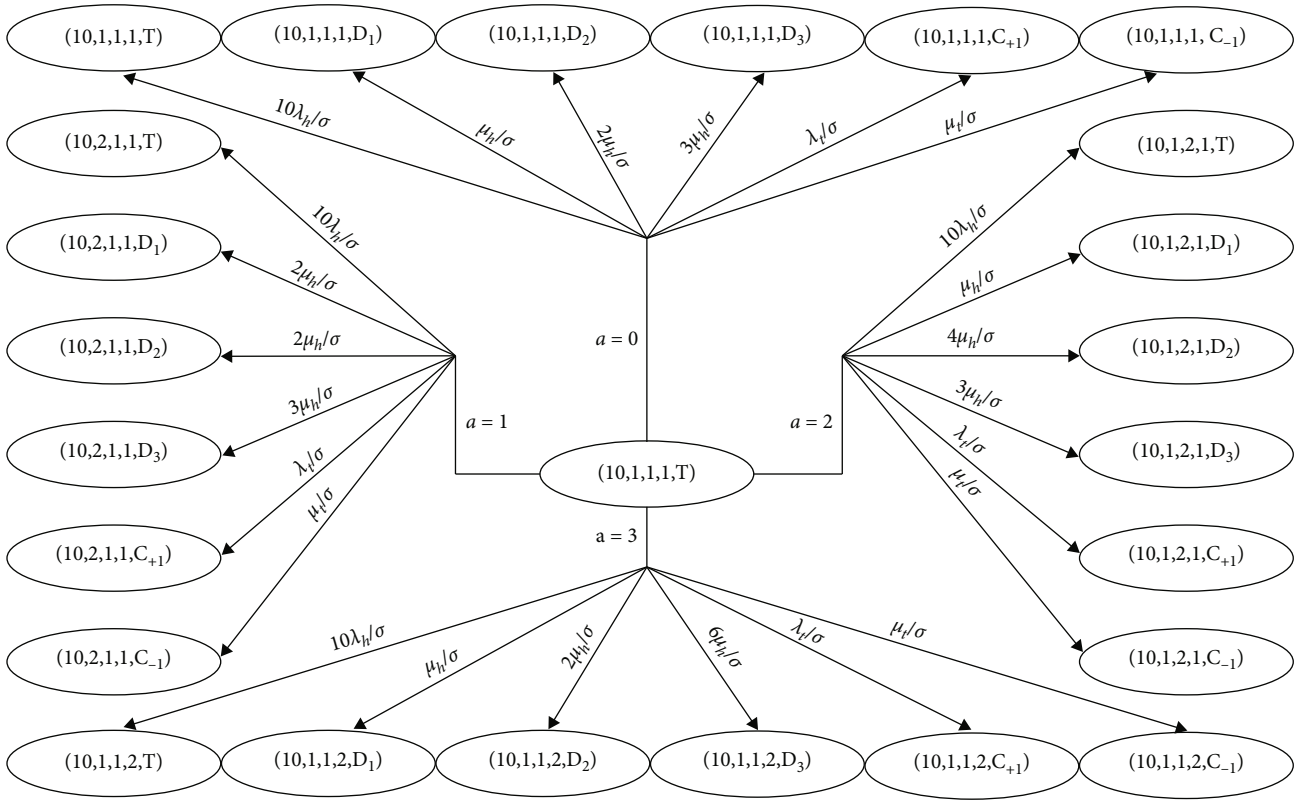

Figure 3: State transition diagram.

Input: state-space S, action space Λ, reward $R(s,a)$, transition probability $P(s' \mid s,a)$, rate of convergence ϵ
Output: the optimal policy π^*
Initialization: set value function $V(s) = 0$ for all $s \in S$ and $m = 0$
1. **For** every state $s \in S$
2. $\widehat{V}_{m+1}(s) = \max_{a \in A}[\widehat{R}(s,a) + \widehat{\gamma}\sum_{s' \in S}\widehat{P}(s' \mid s,a)\widehat{V}_m(s')]$
3. **If** $\| \widehat{V}_{m+1} - \widehat{V}_m \| < \epsilon$ **then**
4. **For** each system state $s \in S$ **do**
5. $'\pi^*(s)' = \arg \max_{a \in A}[\widehat{R}(s,a) + \widehat{\gamma}\sum_{s' \in S}\widehat{P}(s' \mid s,a)\widehat{V}_{m+1}(s')]$
6. **Else**
7. $m++8$ back to line 1
9. Return the optimal policy π^*

Algorithm 1: VIA.

VFC system can handle is denoted as K, and we assume that available RUs in the system are R which fluctuate according to the departure and arrival of the vehicle. The number of RUs R in the VF cannot surpass the maximum number of vehicles K, i.e., $R \leq K$. Task arrival rate and service rate also follow the poison distribution denoted as λ_h and μ_h, respectively. The computing service rate is μ_h when only one RU process the task for i RU service rate $i\mu_h$.

4. Problem Formulation

In this section, we formulate the problem of task offloading by using SMDP. The number of available RUs in the VFC system changes by the events of departure and arrival of tasks and vehicles. When a task request from the vehicle arrives at the system, the system allocates the different number of RUs or transfer it to the RC for processing. The system achieves a reward as a result of task offloading decision which depends on the energy saved and computation time for the processing of the task.

In this SMDP model, the state is a set consist number of available and occupied RUs under different events. An action indicates the choices for decisions for different states. The reward reflects the advantage of the system in terms of energy and time for different states and actions. The probabilities of transition from one state to another under different actions

are described by transition probabilities. The notations used in this section are summarized in Table 1.

4.1. States. The system state S indicates the available resources present in the VFC system in the form of RUs, the number of requests processed by various numbers of RUs, and the events of requests and vehicles [21], i.e.,

$$S = \left\{ s \mid s = \left(R, n_1, n_2, \cdots, n_{M_R}, e \right) \right\}, \tag{1}$$

where R is the available RUs in the system and n_i denotes the quantity of task request served by the i resource units $1 \le i \le M_R$, and e is a particular event that belongs to the E set

$$E = \left\{ T, D_1, D_2, \cdots, D_{M_R}, C_{+1}, C_{-1} \right\}. \tag{2}$$

Here, T denotes the task's arrival, M_R denotes the maximum RUs the system of VFC can assign to the task, D_i is the departure of the task that was serviced and accomplished by i RUs, C_{+1} denotes vehicle's arrival rate, and C_{-1} denotes the departure of the vehicle. The overall sum of vehicles K should be greater than the number of RUs in the system. The allocated RUs to the in the system can be calculated by $\sum_{i=1}^{M_R} i.n_i$ that should always be less than the available RUs in the system at any state, i.e., $\sum_{i=1}^{M_R} i.n_i \le R$. Moreover, the remaining number of available RUs is measured from $R - \sum_{i=1}^{M_R} i.n_i$.

4.2. Actions. Action in the VFC system indicates the different possibilities of decision that the system can be taken according to the specific event of the current state [21]. Action based on the state s belongs to the set Λ and is denoted by $a(s)$

$$\Lambda = \{-1, 0, 1, 2, \cdots, M_R\}, \tag{3}$$

where $a(s) = -1$ shows the case when the task is completed and depart the system; no action of allocation of RUs is taken similarly when a vehicle arrives or departs, only the VFC system is updating its information about the available RUs. When the task request arrived, there are two options; either the request is accepted or transfer it to the RC; $a(s) = 0$ indicates the action when the task has arrived and there are no available RUs in the system and transfer it to the RC; $a(s) = i$ means that the task arrives and i RUs allocated for the processing of the task. The relationship between action and events is shown in the equation below.

$$\Lambda = \begin{cases} \{-1\}, & e = \left\{ D_1, D_2, \cdots, D_{M_R}, C_{+1}, C_{-1} \right\}, \\ \{0, 1, 2, \cdots, M_R\}, & e = T. \end{cases} \tag{4}$$

4.3. Rewards. The reward reflects the advantage of the system of VFC after various actions undertaken for several states. As the main purpose of the system is to cut off the energy consumption and execution time of the tasks in the system by saving power and increasing the processing speed, the reward comprises both, the total income and cost of the system [21]. When an action is performed at a specific state s, the system earns an instant income $I(s, a)$.

The state s remains for a certain time till the next event is occurring, and state s is transitioned to the next state s'. This time is known as the cost $G(s, a)$ of the system. The difference of the income $I(s, a)$ and the cost $G(s, a)$ is known as reward $R(s, a)$.

$$R(s, a) = I(s, a) - G(s, a). \tag{5}$$

The income and cost of the system are derived below.

4.3.1. Income. The income of the system depends on different events and actions because the state is changed by the occurrence of the events. The income of the VFC system can be described as follows:

$$\begin{cases} \left[\omega_e \beta_e \left(P_c \left(\frac{1}{\mu_h} \right) - P_t d_f - P_c \left(\frac{1}{(i+1)\mu_h} \right) \right) + \omega_d \beta_d \left(\left(\frac{1}{\mu_h} \right) - d_f - \left(\frac{1}{(i+1)\mu_h} \right) \right) - \zeta d_f \right], & a = i, e = T, \\ \left[\omega_e \beta_e \left(P_c \left(\frac{1}{\mu_h} \right) - P_t d_f \right) + \omega_d \beta_d \left(\left(\frac{1}{\mu_h} \right) - d_f - d_c \right) - \zeta \left(d_{f+} d_c \right) \right], & a = 0, e = T, \\ 0, & a = i, e = \left\{ D_1, D_2, \cdots, D_{M_R}, C_{+1} \right\}, \\ 0, & a = i, e = \{C_{-1}\}, \sum_{i=1}^{M_R} i.n_i < R, \\ -\eta, & a = i, e = \{C_{-1}\}, \sum_{i=1}^{M_R} i.n_i = R. \end{cases} \tag{6}$$

Table 2: Values of parameters in the VFC system.

Parameter	Value	Parameter	Value
P_c	4.4 W	β_e	2
P_t	1.8 W	β_d	2
d_f	2 ms	α	0.1
d_c	4 ms	K	6-14
ω_e	0.7	M_R	3
ω_d	0.3	ζ	2
η	18	θ	10
λ_h	2-9	λ_c	9
μ_h	8, 16	μ_h	8

The income function is explained below.

(1) $a = i, e = T$. When a task request arrives and is accepted by the system when resources are sufficient for the task, the system assigns i RUs to complete the request of the task; one part is computed locally by the requested vehicle so that no energy is wasted in idle. The instant income that can be earned by the VFC system is $[\omega_e\beta_e(P_c(1/\mu_h) - P_t d_f - P_c(1/((i+1)\mu_h))) + \omega_d\beta_d((1/\mu_h) - d_f - (1/((i+1)\mu_h))) - \zeta d_f]$ while $(P_c(1/\mu_h) - P_t d_f - P_c(1/((i+1)\mu_h)))$ is the energy saved and $((1/\mu_h) - d_f - (1/((i+1)\mu_h)))$ is the time saved during the processing of tasks in the VFC system. ω_d and ω_e are the weightage to the saved time and energy according to the various purposes; they can be predefined where $\omega_d + \omega_d = 1$. β_d and β_e are the saved price per unit time and energy to convert the energy and time into revenue. The cost of transferring the task to the VFC system and receiving the result from it is denoted as ζd_f and is known as the transfer expense. P_c and P_t are the computation and transmitting power, respectively. $P_c(1/((i+1)\mu_h))$ is the energy consumed when i RUs are assigned to the task request $i+1$ RUs process the task, as the requested vehicle also processes one part of the task. The total service time for the processing of the task is $(1/((i+1)\mu_h))$.

(2) $a = 0, e = T$. When a task request arrives and there are not sufficient resources in the VFC system, the request is not accepted by the system and transferred to the RC for processing. $[\omega_e\beta_e(P_c(1/\mu_h) - P_t d_f) + \omega_d\beta_d((1/\mu_h) - d_f - d_c) - \zeta(d_{f+}d_c)]$ is the immediate reward earned by the system when a task is processed by the remote cloud. Here, ζd_f and $d\zeta_c$ are the cost of transfer expense. ζd_c denotes the cost of transferring the task to the RC and receiving the result from the cloud. Remote cloud has a very large computation capability so the computation energy and time are not considered and have not affected the energy of the VFC system. But the delay is very large, so it is not a wise decision to transfer the task to the RC.

(3) $a = -1, e = \{D_1, D_2, \cdots, D_{M_R}, C_{+1}\}$. For the events of the arrival of the vehicle and the task's departure, there is no income as the system takes no action.

(4) $a = -1, e = \{C_{-1}\}, \sum_{i=1}^{M_R} i.n_i < R$. The system does not gain any reward when the vehicle departs the system and there are enough RUs to be allocated.

(5) $a = -1, e = \{C_{-1}\}, \sum_{i=1}^{M_R} i.n_i = R$. When the vehicle departs the VFC system and all the RUs already occupied and processing a task at this moment, the departure of the vehicle disturbs the processing, and the system has to pay a penalty of η.

4.3.2. Cost. To formulate the long-term cost, the discounted cost model is used from [28]. $G(s, a)$ is the expected discounted cost of the system during the duration when the state is transitioned from one state to another by taking an action and defined as

$$G(s, a) = k(s, a)\rho(s, a), \tag{7}$$

where $\rho(s, a)$ is the expected service time when system state s is changed to next state s' by taking an action a; this is assumed to be exponentially distributed according to [28]

$$G(s, a) = k(s, a)E_s^a\left\{\int_0^\tau e^{-\alpha t}dt\right\} = k(s, a)E_a^s\left\{\frac{1 - e^{-\alpha\tau}}{\alpha}\right\} = \frac{k(s, a)}{\alpha + \sigma(s, a)}, \tag{8}$$

where $k(s, a)$ is the expected service time's cost rate for the state s and action a that is characterized as a function of total occupied RUs, i.e.,

$$k(s, a) = \sum_{i=1}^{M_R} i \cdot n_i. \tag{9}$$

α is the factor of discount, and $\sigma(s, a)$ is the expected event rate of the system for the state s and action a that can be calculated by adding all the rates of arrival and departure of vehicles and task requests. The arrival and departure rate of the vehicle is λ_c and μ_c, respectively, while the arrival rate of the request of task and departure rate of the task request depends on the various events and actions of the system calculated as below.

(1) $a = i, e = T$. With the arrival of the task request in the VFC system, the system assigns i RUs for the task processing. $R\lambda_c$ is the arrival rate of the task, and the rate of departure for the task request is $(\sum_{j=1}^{M_R} jn_j + i)u_h$ because the allocated number of RUs is $(\sum_{j=1}^{M_R} jn_j + i)$, under action i.

(2) $a = -1, e = D_i$. The system takes no action when a task departs from the system which is allocated by i RUs. The task request arrival rate is $R\lambda_c$, while the departure rate of the task is $(\sum_{j=1}^{M_R} jn_j - i)u_h$.

(3) $a = -1, e = C_{-1}$. When the vehicle leaves the VFC system, no action is taken by the system but the available number of vehicles decreased by 1; hence, $(R-1)\lambda_c$ is the task request arrival rate, while the departure rate of tasks is $(\sum_{j=1}^{M_R} jn_j)u_h$.

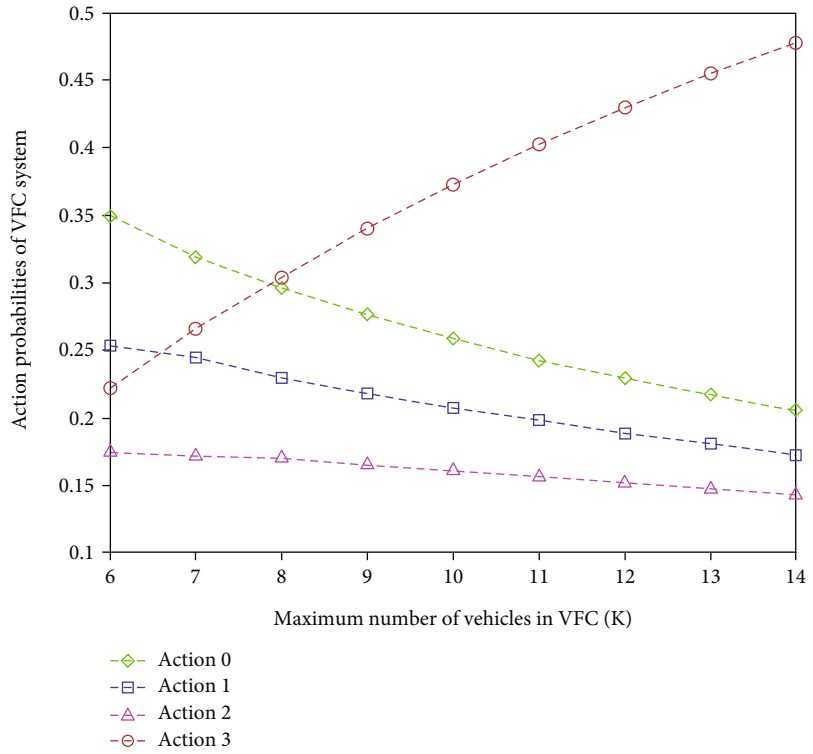

Figure 4: Action probabilities for the different numbers of maximum vehicles in the VFC system ($\lambda_h = 2$, $\mu_h = 8$).

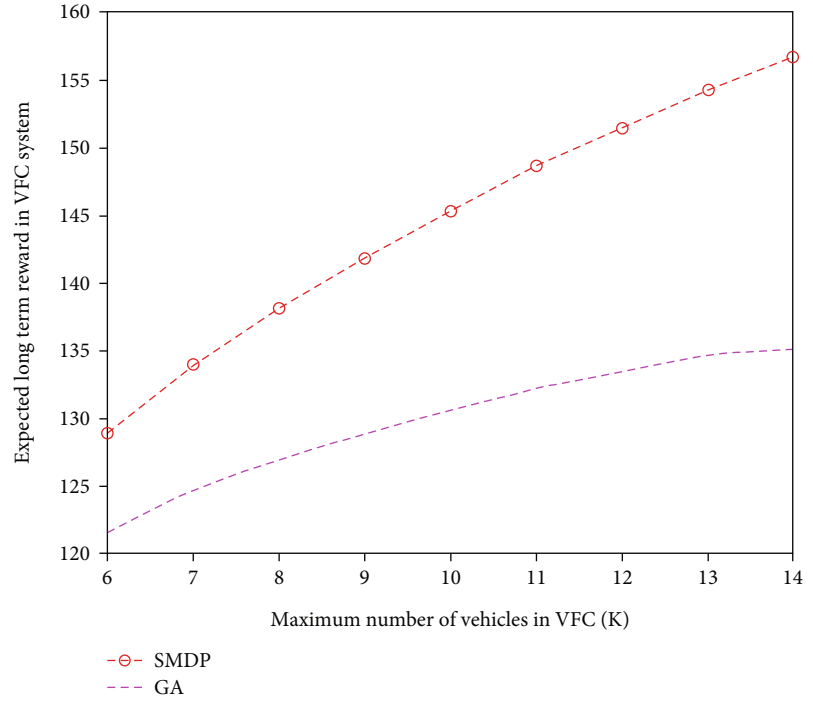

Figure 5: Reward for the VFC system for the different numbers of maximum vehicles ($\lambda_h = 2$ and service rate $\mu_h = 8$).

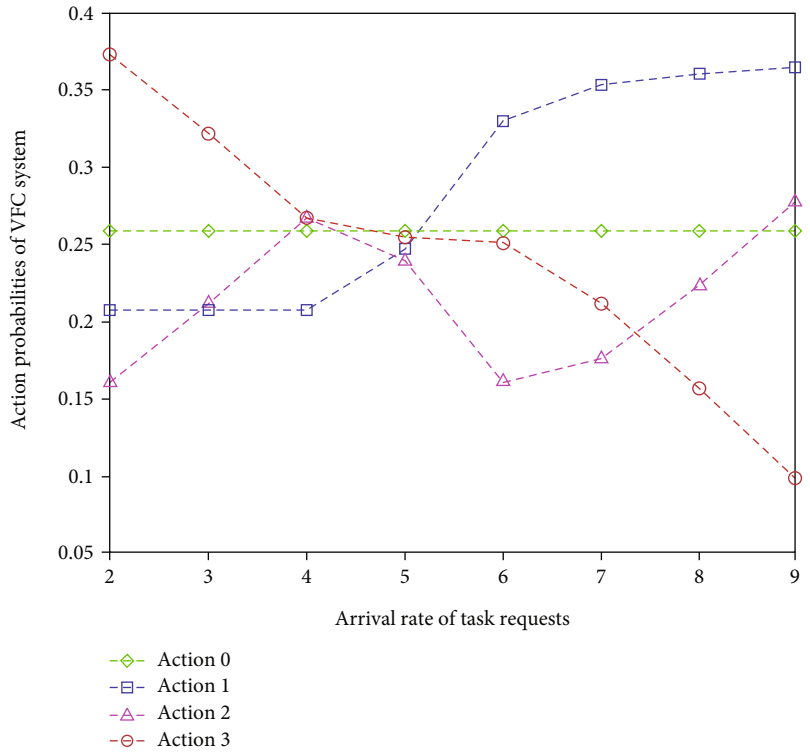

Figure 6: Action probabilities for different task request arrival rate λ_h ($K = 10$, $\mu_h = 8$).

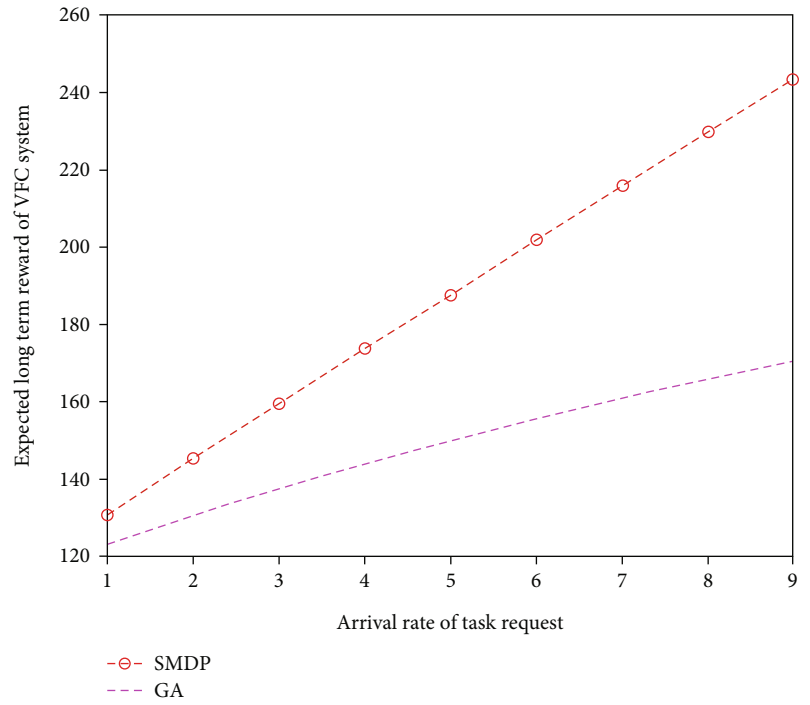

Figure 7: Reward for the VFC system for different numbers of different task request arrival rate λ_h ($K = 10$, $\mu_h = 8$).

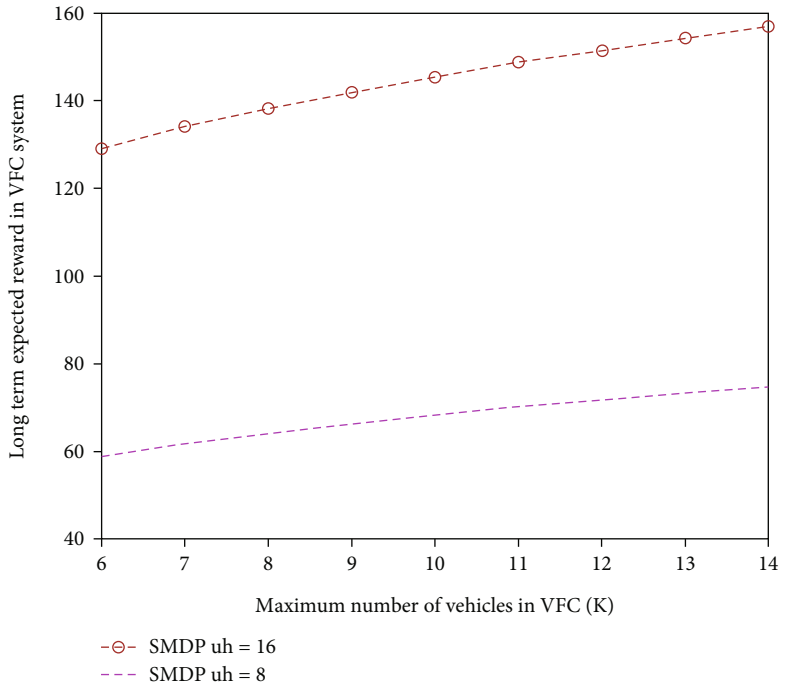

Figure 8: Reward of the VFC system for different service rates μ_h ($K = 6 - 14$, $\lambda_h = 2$).

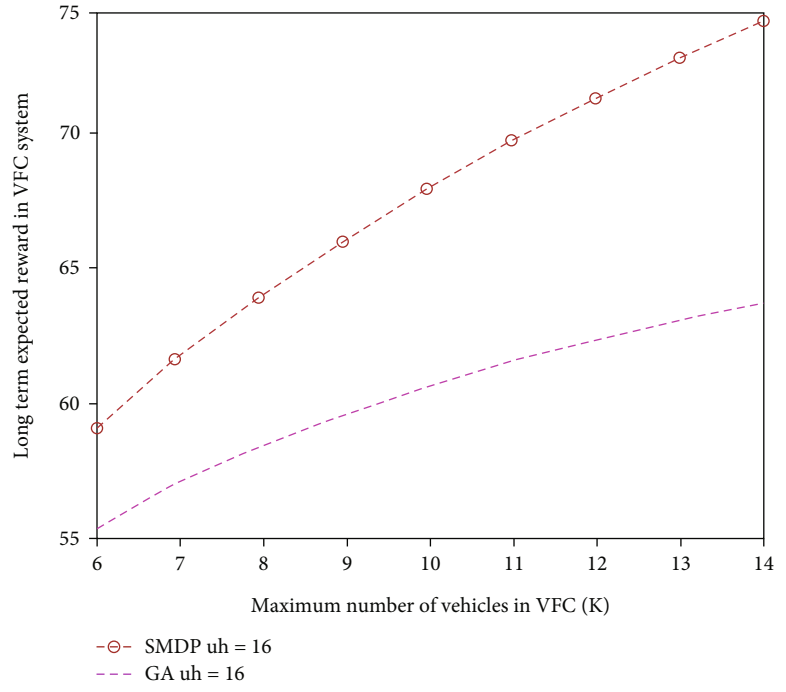

Figure 9: Reward for the VFC system for the different numbers of maximum vehicles K ($\lambda_h = 2$ and service rate $\mu_h = 16$).

(4) $a=-1, e=C_{+1}$. The system takes no action while the vehicle arrives, but there is an increment in the available number of vehicles by 1; hence, $(R+1)\lambda_c$ is the task request arrival rate, while the departure rate of tasks is $(\sum_{j=1}^{M_R} jn_j)u_h$.

The expected service rate $\sigma(s,a)$ for different events and action is calculated as

$$\sigma(s,a)=\begin{cases} R\lambda_c+\mu_c+\lambda_c+\left(\sum_{j=1}^{M_R} jn_j+i\right)u_h, e=T, a=i, \\ R\lambda_c+\mu_c+\lambda_c+\left(\sum_{j=1}^{M_R} jn_j-i\right)u_h, e=D_i, a=-1, \\ (R+1)\lambda_c+\mu_c+\lambda_c+\left(\sum_{j=1}^{M_R} jn_j\right)u_h, e=C_{+1}, a=-1, \\ (R-1)\lambda_c+\mu_c+\lambda_c+\left(\sum_{j=1}^{M_R} jn_j\right)u_h, e=C_{+1}, a=-1. \end{cases} \tag{10}$$

4.4. Transition Probability. The transition probability is the probability of going from the current state s to the future state s' after taking an action has been taken [21]. We have used the generalized SMDP transition probabilities [21]; however, the proposed work has an additional state for remote cloud computing.

The transition probability in the system of VFC can be explained by the ratio between the sum of all the events and the next event rate. The transition probability is denoted as $P(s'|s,a)$ and formulated as below:

(1) $s=(R,n_1,\cdots,n_N,T), a=i$

$$P\left(s'\mid s,a\right)=\begin{cases} \frac{R\lambda_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i+1,\cdots,n_{M_R},T\right), \\ \frac{(n_i+1)i\mu_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i+1,\cdots,n_{M_R},D_i\right), \\ \frac{n_j j\mu_h}{\sigma(s,a)}, i\neq j\, s'=\left(R,n_1,\cdots,n_i+1,\cdots,n_{M_R},D_j\right), \\ \frac{\lambda_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i+1,\cdots,n_{M_R},C_{+1}\right), \\ \frac{\mu_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i+1,\cdots,n_{M_R},C_{-1}\right). \end{cases} \tag{11}$$

(2) $s=(R,n_1,\cdots,n_N,T), a=0$

$$P\left(s'\mid s,a\right)=\begin{cases} \frac{R\lambda_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i,\cdots,n_{M_R},T\right), \\ \frac{n_i i\mu_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i,\cdots,n_{M_R},D_i\right), \\ \frac{\lambda_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i,\cdots,n_{M_R},C_{+1}\right), \\ \frac{\mu_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i,\cdots,n_{M_R},C_{-1}\right). \end{cases} \tag{12}$$

(3) $s=(R,n_1,\cdots,n_N,D_i), a=-1$

$$P\left(s'\mid s,a\right)=\begin{cases} \frac{R\lambda_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i-1,\cdots,n_{M_R},T\right), \\ \frac{(n_i-1)i\mu_h}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i-1,\cdots,n_{M_R},D_i\right), \\ \frac{n_j j\mu_h}{\sigma(s,a)}, i\neq j\, s'=\left(R,n_1,\cdots,n_i-1,\cdots,n_{M_R},D_j\right), \\ \frac{\lambda_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i-1,\cdots,n_{M_R},C_{+1}\right), \\ \frac{\mu_c}{\sigma(s,a)}, s'=\left(R,n_1,\cdots,n_i-1,\cdots,n_{M_R},C_{-1}\right). \end{cases} \tag{13}$$

(4) $s=(R,n_1,\cdots,n_N,C_{-1}), a=-1$

$$P\left(s'\mid s,a\right)=\begin{cases} \frac{(R-1)\lambda_h}{\sigma(s,a)}, s'=\left(R-1,n_1,\cdots,n_i,\cdots,n_{M_R},T\right), \\ \frac{n_i i\mu_h}{\sigma(s,a)}, s'=\left(R-1,n_1,\cdots,n_i,\cdots,n_{M_R},D_i\right), \\ \frac{\lambda_c}{\sigma(s,a)}, s'=\left(R-1,n_1,\cdots,n_i,\cdots,n_{M_R},C_{+1}\right), \\ \frac{\mu_c}{\sigma(s,a)}, s'=\left(R-1,n_1,\cdots,n_i,\cdots,n_{M_R},C_{-1}\right). \end{cases} \tag{14}$$

(5) $s=(R,n_1,\cdots,n_N,C_{+1}), a=-1$

$$P\left(s'\mid s,a\right)=\begin{cases} \frac{(R+1)\lambda_h}{\sigma(s,a)}, s'=\left(R+1,n_1,\cdots,n_i,\cdots,n_{M_R},T\right), \\ \frac{n_i i\mu_h}{\sigma(s,a)}, s'=\left(R+1,n_1,\cdots,n_i,\cdots,n_{M_R},D_i\right), \\ \frac{\lambda_c}{\sigma(s,a)}, s'=\left(R+1,n_1,\cdots,n_i,\cdots,n_{M_R},C_{+1}\right), \\ \frac{\mu_c}{\sigma(s,a)}, s'=\left(R+1,n_1,\cdots,n_i,\cdots,n_{M_R},C_{-1}\right). \end{cases} \tag{15}$$

The state transition diagram is shown in Figure 3 which shows the state transition process takes current state $s = (10, 1, 1, 1, T)$ and shows it transitioned to the next states for different actions and events with the transition probability [21].

5. Solution

The solution of the above SMDP problem is presented in this section and find the solution to magnify the reward of the VFC system in a long term; by this we save energy and minimize the processing time. To solve the problem, value iteration algorithm is adopted. In the value iteration algorithm, Bellman optimal equation [21, 29] is used. In every iteration of all the states, maximum value function $V(s)$ is calculated for each action, i.e., $\Lambda \epsilon\{-1, 0, 1, 2, \cdots, M_R\}$. When $V(s)$ of every state is converging, the step is terminated. The Bellman optimal equation is shown below:

$$V_{M+1}(s) = \max_{a\in\Lambda}\left[R(s,a) + \gamma\sum_{s'\in S} P\left(s' \mid s, a\right) V_m\left(s'\right)\right]. \quad (16)$$

Here, $\gamma = \sigma(s,a)/(\alpha + \sigma(s,a))$ and known as discount function.

To transform the continuous-time semi-Markov decision process into a discrete-time process, a new parameter is defined $x = K\lambda_h + \lambda_c + \mu_c + K \cdot M_R \cdot \mu_h$ to normalize the reward, discount factor, and transition probabilities. The normalized equations are shown below.

$$\widehat{R}(s,a) = R(s,a)\frac{\alpha + \sigma(s,a)}{\alpha + x},$$

$$\widehat{P}\left(s' \mid s, a\right) = \begin{cases} 1 - \dfrac{[1 - P(s \mid s,a)]\sigma(s,a)}{x}, & s' = s, \\ \dfrac{P\left(s' \mid s, a\right)\sigma(s,a)}{x}, & s' \neq s, \end{cases} \quad (17)$$

$$\widehat{\gamma} = \frac{x}{x + \alpha}.$$

Thus, after normalization, the Bellman optimal equation becomes

$$\widehat{V}_{m+1}(s) = \max_{a\in\Lambda}\left[\widehat{R}(s,a) + \widehat{\gamma}\sum_{s'\in S}\widehat{P}\left(s'|s,a\right)\widehat{V}_m\left(s'\right)\right]. \quad (18)$$

Initially, the value function of all states was set to zero. Now, by Equation (18), the normalized value function of states is calculated by using values from the previous iteration. For example, the maximum value function $\widehat{V}_{m+1}(s)$ in $m+1$th iteration is calculated by using the mth iteration. For finding the optimal policy π^*, the absolute difference between two succeeding iterations is computed for each state, i.e., $\|\widehat{V}_{m+1} - \widehat{V}_m\|$. The algorithm is stopped when the maximal absolute value is lower by the threshold $\epsilon = \theta(1 - \widehat{\gamma})/2\widehat{\gamma}$ and optimal scheme $\pi^*$is obtained.

$$\pi^*(s) = \max_{a\in\Lambda}\left[\widehat{R}(s,a) + \widehat{\gamma}\sum_{s'\in S}\widehat{P}\left(s'|s,a\right)\widehat{V}_m\left(s'\right)\right]. \quad (19)$$

If the threshold is greater than the maximum absolute value, the algorithm goes into the next iteration and continues till the optimal scheme of offloading is obtained. The pseudocode of the VIA is presented in Algorithm 1.

6. Simulation Results

The performance of our proffered scheme of task offloading has been evaluated in this section by conducting trials and getting experimental results. For the comparison, we compare our proposed scheme with the greedy algorithm (GA), which constantly tries to assign the maximum number of RUs to offload the task for processing [28]. We used the MATLAB R2019b tool for the experiments.

The parameters that have been used in the evaluation are exhibited in Table 2. The maximum number of RUs that can be assigned to the task for processing is $M_R = 3$; i.e., 1, 2, or 3 RUs can be assigned to task requests according to the availability of resources in the VFC system.

Action 1, action 2, and action 3 denote the number of assigned RUs to the tasks, while action 0 is the special scenario of sending the service request to the RC. For evaluation, different parameters are adjusted, for example, the number of maximum available vehicle K in the system of VFC, the arrival rate of the task requests λ_h, and service rate μ_h.

Figure 4 illustrates the relationship among the maximum number of vehicles K supported by the VFC system and transition probabilities of the different actions taken by the system when the request arrival rate $\lambda_h = 2$ and service rate $\mu_h = 8$. It is depicted from the figure that when the number of vehicles in the VFC system is low the transition probability for action 2 and action 3 is lower than the transition probabilities of action 0 and action 1. This is because when the number of vehicles is less the available resources in the form of RUs are also less and the VFC system allocates a smaller number of RUs to the task request. With the increment of the number of vehicles in the system, a decreasing trend in the transition probability of actions 0, 1, and 2 and an increase in the trend for action 3 can be seen. The system mostly assigns three RUs to the arriving requests to enhance the expected reward in the long term.

Figure 5 shows the comparison of our proposed scheme of SMDP-based algorithm with the GA scheme when $\lambda_h = 2$ and service rate $\mu_h = 8$. It can be observed that with the increment in the number of vehicles in the VFC system the expected reward of the VFC system increases since with the increase in the maximum number of vehicles the number of the completed tasks increased, and as a result, reward increased. The SMDP-based scheme has 19% improved performance in terms of reward than the GA as shown from the figure. This is because the GA tries to allocate maximum RUs without taking into account the expected reward of the VFC system.

Figure 6 depicts the relationship among the requests of the task arrival rate of λ_h and the transition probabilities of the actions taken by the system, when $K = 10$ and service rate $\mu_h = 8$. It could be noticed from the figure that the transition probability of action 3 is higher than all other actions when the task arrival rate is low because there is abundant computation capability in VFC. The system tries to assign the utmost number of RUs to the requested task to increase the long-term expected award. With the increase in the rate of the task, arrival requests in the transition probabilities of action 1 start increasing while for action 3 there is a decreasing trend because the VFC system starts taking conservative decisions according to the available resources and to get the maximum reward taking action 1 or 2 by this complete more tasks without transferring the requests to the RC. The transition probability of action 2 shows random behavior; in the beginning, it starts increasing and after some time shows the decreasing trend and at the end start increasing again according to the optimal policy to improve the expected reward.

Figure 7 demonstrates the relationship between λ_h and long-term reward of SMDP and compares it with the GA when $K = 10$ and service rate $\mu_h = 8$. The long-term expected reward shows an increasing trend with the increase in the rate of task arrival request because with the increment in the number of tasks request the completed task also increased, and as a result, long-term expected reward increases. Moreover, it can be conceivable from the figure that our proposed methods outperform the GA.

Figure 8 shows the comparison of the long-term reward of the system when μ_h changes from 16 to 8. It can be perceived from the figure that the expected reward for $\mu_h = 8$ is lower than that for $\mu_h = 16$ because the number of tasks computed is less when the rate of service is low. Moreover, when the service rate is high, processing of tasks by RUs is faster; as a result, the number of available RUs increases, and more tasks can be offloaded and gain more reward.

In Figure 9, there is a comparison of our proposed scheme of SMDP-based algorithm with the GA scheme when $\lambda_h = 2$ and service rate $\mu_h = 16$. It can be seen that our proposed scheme outperforms the GA and shows a similar trend as in Figures 5 and 7.

In this section, we present the performance of our work with the help of different experiments. We see that the proposed algorithm exhibits superior performance than the GA and gains more reward under different parameters; i.e., varying maximum numbers of vehicles in the VFC system, different service rates, and different rates of the task request, our scheme outperforms in all the cases.

7. Conclusion and Future Work

In this paper, we propose an optimal energy-aware task offloading technique for the Internet of Vehicles. When a vehicle with the task request arrives, the system decides for the allocation of computational resources, i.e., RUs, and divides the task according to the decision. We use parallel computing and save energy and minimize time delay for our VFC system and formulate the problem as an infinite horizon SMDP. The Bellman optimal equation is used in value iteration algorithm to get an optimal policy that amplifies the long-term expected reward that saved energy and time in this problem. The proposed scheme demonstrates the improved performance of the greedy algorithm as established by the substantial numerical results. In the future, we aim to consider mobility of the vehicles and dynamic wireless connectivity in task offloading.

Data Availability

Data is available from the corresponding author on request.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

The authors extend their appreciation to the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University for funding this work through Research Group no. RG-21-07-06.

References

[1] Q. Wu, S. Xia, P. Fan, Q. Fan, and Z. Li, "Velocity-adaptive V2I fair-access scheme based on IEEE 802.11 DCF for platooning vehicles," *Sensors*, vol. 18, no. 12, pp. 1–23, 2018.

[2] J. Han, A. Sciarretta, L. L. Ojeda, G. De Nunzio, and L. Thibault, "Safe-and eco-driving control for connected and automated electric vehicles using analytical state-constrained optimal solution," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 163–172, 2018.

[3] M. A. Javed, S. Zeadally, and Z. Hamid, "Trust-based security adaptation mechanism for vehicular sensor networks," *Computer Networks*, vol. 137, pp. 27–36, 2018.

[4] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, "Simultaneous localization and mapping: a survey of current trends in autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[5] Q. Wu, H. Ge, H. Liu, Q. Fan, Z. Li, and Z. Wang, "A task offloading scheme in vehicular fog and cloud computing system," *IEEE Access*, vol. 8, pp. 1173–1184, 2020.

[6] J. Zhao, Y. Liu, Y. Gong, C. Wang, and L. Fan, "A dual-link soft handover scheme for C/U plane split network in high-speed railway," *IEEE Access*, vol. 6, no. c, pp. 12473–12482, 2018.

[7] W. Xu, H. Zhou, N. Cheng et al., "Internet of vehicles in big data era," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 19–35, 2018.

[8] Q. Fan, N. Ansari, and X. Sun, "Energy driven avatar migration in green cloudlet networks," *IEEE Communications Letters*, vol. 21, no. 7, pp. 1601–1604, 2017.

[9] Y. Zhang, "Optimization strategy of mobile data transmission based on optimal crowd feedback," *EURASIP Journal on Embedded Systems*, vol. 2016, no. 1, Article ID 26, 2017.

[10] M. Báguena, C. T. Calafate, J. C. Cano, and P. Manzoni, "An adaptive anycasting solution for crowd sensing in vehicular environments," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7911–7919, 2015.

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

[12] Y. Bin Zikria, M. Khalil Afzal, and S. Won Kim, "Internet of multimedia things (Iomt): opportunities, challenges and solutions," *Sensors*, vol. 20, no. 8, 2020.

[13] M. Bagaa, A. Ksentini, T. Taleb, R. Jantti, A. Chelli, and I. Balasingham, "An efficient D2D-based strategies for machine type communications in 5G mobile systems," in *2016 IEEE Wireless Communications and Networking Conference*, vol. 2016, Doha, Qatar, 2016.

[14] J. Zhao, S. Ni, L. Yang, Z. Zhang, Y. Gong, and X. H. Yu, "Multiband cooperation for 5G HetNets: a promising network paradigm," *IEEE Vehicular Technology Magazine*, vol. 14, no. 4, pp. 85–93, 2019.

[15] R. K. Naha, S. Garg, D. Georgakopoulos et al., "Fog computing: survey of trends, architectures, requirements, and research directions," *IEEE Access*, vol. 6, pp. 47980–48009, 2018.

[16] Q. Wu, H. Ge, Q. Fan, W. Yin, B. Chang, and G. Wu, "Efficient task offloading for 802.11p-based cloud-aware mobile fog computing system in vehicular networks," *Wireless Communications and Mobile Computing*, vol. 2020, 12 pages, 2020.

[17] Z. Wang, Z. Zhong, and M. Ni, "Application-aware offloading policy using SMDP in vehicular fog computing systems," in *2018 IEEE international conference on communications workshops (ICC Workshops)*, pp. 1–6, Kansas City, MO, USA, 2018.

[18] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, and S. Yu, "Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14198–14211, 2020.

[19] J. Klaimi, S. M. Senouci, and M. A. Messous, "Theoretical game approach for mobile users resource management in a vehicular fog computing environment," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 452–457, Limassol, Cyprus, 2018.

[20] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud computing*, vol. 7, no. 1, pp. 196–209, 2019.

[21] Q. Wu, H. Liu, R. Wang, P. Fan, Q. Fan, and Z. Li, "Delay-sensitive task offloading in the 802.11p-based vehicular fog computing systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 773–785, 2020.

[22] H. M. Birhanie, M. A. Messous, S. M. Senouci, E. H. Aglzim, and A. M. Ahmed, "MDP-based resource allocation scheme towards a vehicular fog computing with energy constraints," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Abu Dhabi, United Arab Emirates, 2018.

[23] Z. Zhou, P. Liu, J. Feng, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Computation resource allocation and task assignment optimization in vehicular fog computing: a contract-matching approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3113–3125, 2019.

[24] Y. Wang, K. Wang, H. Huang, T. Miyazaki, and S. Guo, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, pp. 976–986, 2019.

[25] Z. Liu, P. Dai, H. Xing, Z. Yu, and W. Zhang, "A distributed algorithm for task offloading in vehicular networks with hybrid fog/cloud computing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–14, 2021.

[26] J. Shi, G. S. Member, J. Du, and J. Wang, "Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16067–16081, 2020.

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

[28] S. Radhakrishnan, D. Kolippakkam, and V. S. Mathura, *Introduction to Algorithms*, MIT press, 2007.

[29] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, 2008.