Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-020-01066-2

SDN-based offloading policy to reduce the delay in fog-vehicular networks

Alla Abbas Khadir[1] · Seyed Amin Hosseini Seno[1]

Received: 5 August 2020 / Accepted: 26 December 2020

Abstract
Integrating fog computing with vehicular networks led to the rapidly growing demands of vehicle applications regarding computation-intensive and low response time with meeting the request deadline. The limited resources of the fog node have made it unable to meet the demands of such applications. Offloading the requests to other Off-Load Destination (OLD) is a suitable solution for the fog node to deal with these demands. Nonetheless, this simultaneously faces two challenges. The first challenge is the offloading to a nearby fog node which stills not the fully efficient choice when this nearby fog node is busy. The second challenge is the selection decision of the optimal OLD where the fog node incurs additional burden through getting status information of all neighboring fog nodes, affecting the selection decision, which is why it may not fulfill the request deadline. To solve the first challenge, a new hybrid offloading architecture has been proposed, where the underutilized resources of Vehicular Fog Computing (VFC) are joined with the cloud to be an OLD, thus increase the processing chance of the offloaded requests. The second challenge has been solved by optimizing the selection decision of the fog node via taking the global network resources benefit of Software Defined Network (SDN) in the proposed offloading architecture to design an SDN-based offloading policy. The selection decision problem is formulated as a Binary-Linear Programming and solved by CPLEX software. The simulation results show that our proposed improves the performance of the fog node by providing less response time and significantly outperforming other offloading policies.

Keywords Vehicular fog computing · Infeasible request · Deadline · Offloading policy · OLD · SDN

1 Introduction

The Vehicular Ad-hoc Network (VANET) has been introduced as a special type of Mobile Ad hoc Network (MANET) in which the mobile nodes are vehicles [1]. The VANET's main contributions include improving traffic safety, providing infotainment, and commercial applications for the users, in addition to its being a crucial technology in the Intelligent Transport System [2]. Fast-growing vehicles demand computationally intensive applications (e.g., online gaming, video-audio streaming, and Augment Reality, etc.) have made cloud computing a convenient solution to meet such demands [3]; nonetheless, the centralized location of the cloud and the remote distance from the end device environments hold back the real-time services to the applications, causing a high delay sensitivity [4].To address these problems, the fog computing paradigm extends the cloud services (computation, storage, and application) to the network edge, and thereby remains in the proximity of the end devices (IoT, vehicles, and so on). According to this proximity, fog computing can provide a reduction in the latency and the bandwidth usage in the backbone network, and reducing the cost service with supporting the mobility [5]. Consequently, the end devices can offload their requests (or tasks) to the close fog node to obtain a fast and high service rate.

However, the computational and storage resources of fog nodes are limited and insufficient [6–9], thus it fails to meet the high computing demands of the vehicular applications [10], especially when the number of the requests grows and exceeds its capacity. Besides, different processing times of the application [11] cause a long waiting time (queuing delay), which can subsequently result in an upsurge in the average

✉ Seyed Amin Hosseini Seno
hosseini@um.ac.ir

Alla Abbas Khadir
alla.hunan@gmail.com

[1] Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

response time of the requests. Fog node to be able to address this problem, it simply offloads the requests to another Off-Load Destination (OLD) such as the nearby fog nodes or the cloud [7, 12], whereby the average response time of the requests would be reduced. This indeed helps face the run out of the fog resources, but the requests may be handled or not at that OLDs. This is the case, because the OLDs give a high priority for processing their requests, or they re-offload the offloading requests to others [13], which will increase the dropping probability of these requests. Furthermore, meeting the deadline constraint of the request [12, 14] should be considered in the problem formulation. Generally, in the VANET, any node has communication, computation, and storage resources can become a fog node [15].

Therefore, in this paper, Vehicular Fog Computing (VFC) [16] has been utilized to design a new hybrid offloading architecture and support high computation for vehicular applications such that the parked and moving vehicles would join their cost-based computation resources to be an OLD for the fog-vehicular network. In which the underutilized resources of these vehicles are enough to execute high computation requests while imposing fewer costs compared to the other OLD's expenses. The SDN has been designed to address the inflexibility in the legacy network via decoupling the forwarding plane from the control plane and providing programmability on the control plane. Also, the SDN controller is responsible for collecting all the information on the data plane periodically. This information such as computation capability, the communication delay, the average waiting time, and the unit cost to maintain a global view of the network [17]. Thus, it can manage, configure, and orchestrate all the resources in the network in an efficient and fast way [18, 19]. By bringing this feature to the proposed architecture, an SDN-based offloading policy is proposed to make a correct offloading decision on behalf of the fog node to select the optimal OLD. In other words, the offloading process will be centralized via the controller. The contributions of this paper are as follows:

- Designing a new hybrid offloading architecture by joining the VFC (parked and moving vehicles) and a cloud to provide more OLDs that enable the fog nodes to offload their requests.
- Formulating a mathematical model in the form of Binary-Linear Programming (BLP) to select the optimal OLD which has a minimum average response time and an acceptable average processing cost. This model can be applied to the central controller.
- Introducing an SDN-based offloading policy which is based on the global network view feature of the controller to solve the proposed mathematical model and optimize the offloading selection decision of the fog node.

The rest of the paper is organized as follows: In Section 2, the related work is discussed. In Section 3, the assumptions and limitations of the proposed network model are presented. In Section 4, the system model and problem formulation are illustrated. In Section 5, the proposed offloading policy is described. In Section 6, the simulation and performance evaluation of the proposed offloading policy are addressed. Section 7 culminates the paper by presenting the conclusions.

2 Related works

There are many works which have addressed fog offloading [7, 20] and the most recent investigations are reviewed in this section: Y. Xiao et al. [13] introduced an offload cooperation strategy to achieve the tradeoff between the Quality of Experience (QoE) of the request and power efficiency of the fog node. They addressed the request offloading problem, in which each fog node determines the optimal request fraction that would be either processed or offloaded to the neighboring fog nodes under the power efficiency constraint of them; as a result, the response time of the user will minimize. C. Zhu et al. [10] studied the feasibility and challenges for offloading crowdsourcing video to the VFC (e.g., buses and taxis) instead of forwarding to the cloud. These nodes have been utilized to gather and process the video from the vehicles within communication ranges. To enhance the processing ability of the cloud, H. Zhang et al. [21] proposed an offloading strategy to offload the mobile applications via vehicular cloud computing, in which an optimal moving vehicle with the longest connection time could be discovered by the fog and was then utilized. In [22], an offloading cooperative fog computing algorithm is proposed, where the author designed an intra-inter fog resource management architecture by considering the latency and efficiency network to deal with the big data of the IoV applications. As regards, the performance of the fog node was optimized by a hierarchical resource management model for the intra-fog (energy-aware) and inter-fog architectures (QoS-aware), respectively.

He. Xiuli et al. [23] presented a software-defined cloud-fog networking architecture to reduce the request processing time for delay-sensitive applications in the IoV, such that the SDN was integrated with the fog and the cloud. Moreover, they proposed an SDN-based modified constrained optimization particle swarm optimization centralized load balancing algorithm to balance the workload between the cloud and the fog devices. In this way, the request processing delay was effectively reduced. Furthermore, reducing the IoT service delay through the fog offloading has been studied in [24]. They introduced a general framework for the IoT-fog-cloud applications and proposed a delay-minimizing fog offloading policy in which the fog node could offload their requests either to other fog nodes or the cloud. Besides, they presented an

analytical model to evaluate their proposed policy and to show how this policy could help reduce the processing delay. X.Wang et al. [25] proposed a feasible solution that enables TMS for offloading the real-time traffic to the fog-based IoV, thus minimizing the average response time for the events reported by vehicles. Fog node collaboration for reducing service delay of the requests is considered in [24]. The proposed policy assumes that a fog node can accept and process the IoT requests based on its current load, otherwise, it would offload the requests to the best neighboring fog node or the cloud.

He et al. [26] utilized SDN for heterogeneous vehicular networks, where different network resources are properly scheduled to minimize communications costs. In [27], the authors proposed an architecture that integrated a satellite network with 5G cloud-enabled Internet of Vehicles to support seamless coverage and efficient resource management. They designed a joint optimization problem of computation offloading under delay and cost constraints that are based on an incentive mechanism. The simulated annealing based on the Markov Chain Monte Carlo is applied to solve the non-convex and NP-hard optimization problems. The authors in [28] presented a dual-side optimization offloading decision problem to minimize the cost of both vehicles and corresponding MEC servers simultaneously, where TV white space (TVWS) bands have been used to reinforce the bandwidth for computation offloading. Zhang et al. [29] proposed a cloud-based MEC transmission scheme to reduce the total cost of the offloading process under the delay tolerance of each computation task. They designed a task-file transmission strategy with a predictive V2V relay and proposed an optimal predictive combination-mode (V2I and V2V) offloading mechanism. The authors in [30] studied the energy consumption, execution delay, and payment cost of offloading processes in a mobile fog computing system. Based on the theoretical analysis, a multi-objective problem with various constraints is formulated and addressed by using an Interior Point Method (IPM)-based algorithm. However, these researches considered that the fog node can offload the request to neighboring fog nodes with neglecting the required deadline of the request. Also, they did not consider the processing cost of the request neither for the assistant parameters nor the constraints and how it can affect the response time. Qui et al. [31] investigated the resource limitation issue of the MEC server. They proposed a scalable vehicle-assisted MEC (SVMEC) paradigm to enhance the scalability of computing services through extending the resource of MEC. In the SVMEC paradigm, a MEC makes an offloading strategy to select proper computing nodes and amount of computing resources for the tasks. The problem of joint node selection and resource allocation has been formulated as a Mixed Integer Nonlinear Programming (MINLP) problem, where the objective was minimizing the total computation overhead, including task completion time and monetary cost for using computing resources. The problem is solved by decomposing the original problem into two sub-problem resource allocation problem with fixed task offloading decision and node selection problem with the optimal-value function before the resource allocation. Zhuang et al. [32] proposed a large number of SDN/NFV solutions in the IoV network. One of these solutions was a three-tier vehicular computing architecture to accommodate various computing offloading IoV services, where nearby vehicles provide opportunistic computing, edge nodes support fast computing with limited resources, and the cloud guarantees sufficient resources for sustainable computing. Specifically, the edge-cloud interplay problem can be formulated to optimize the tradeoff between energy efficiency and backhaul bandwidth consumption under the task execution latency, thus, the task offloading obtains the optimal decisions. Table 1 presents a comparison between our proposed and the reviewed research.

3 System model

3.1 Network model

The three-layer network architecture is presented in Fig.1.These layers are the fog nodes layer, intermediate layer, and the OLD layer. In the fog nodes layer, there are different types of fog nodes that support the OpenFlow protocol. These fog nodes have limited storage and processing capabilities that are provided to vehicles within the communication range. In the intermediate layer, there is an SDN controller with a network of programmable forwarding SDN switches. The SDN controller runs the OpenFlow protocol to communicate, configure, and manage these switches. The OLD layer includes computing resources of the VFC and cloud. As we observe, the intermediate layer connects all the elements in the network which establishes an efficient routing path between any fog nodes and the OLD; thus, it facilitates the offloading process between them through accelerating the transmitting process of the requests with minimum network delay. The SDN controller maintains a global view by collecting all the information on the network periodically. This information such as the available resources in the OLDs and their unit cost is essential to make an optimal offloading decision on behalf of the fog nodes. Here, the set of all cloud servers, parked vehicles, and mobile vehicles is defined as $\mathcal{N} = \{1, 2, \ldots, N\}$, $\mathcal{P} = \{1, 2, \ldots, P\}$, and $\mathcal{M} = \{1, 2, \ldots, M\}$, respectively. The set of fog nodes is denoted as $\mathcal{F} = \{1, 2, \ldots, F\}$.We modeled the proposed network as a weighted undirected graph $G = (V, E)$, where V and E represent the set of the nodes and edges, respectively. The node set represents the fog nodes, cloud servers, parked fog nodes, mobile fog nodes, and SDN switches, whereas the edge set represents the

Table 1 Comparison of the related works

Ref.	SDN	Parked fog	Mobile fog	Processing cost	Response time	Deadline
[13]	×	×	×	×	✓	×
[10]	×	×	✓	×	✓	×
[21]	×	×	✓	×	✓	×
[22]	×	×	×	×	✓	×
[23]	✓	×	×	×	✓	×
[25]	×	✓	✓	×	✓	×
[24]	×	×	×	×	✓	×
[26]	✓	×	×	×	×	×
[27]	✓	×	✓	✓	✓	✓
[28]	×	×	×	✓	✓	×
[29]	×	×	✓	✓	×	✓
[30]	×	×	×	✓	✓	×
[30]	×	×	✓	✓	✓	×
Proposed	✓	✓	✓	✓	✓	✓

communication links between them. The edge weight has been represented by the communication delay τ that contains the propagation and transmission delay. Table 2 presents the symbols used.

3.2 Average response time model

3.2.1 Fog node computing

In this paper, different edge network devices are considered as the fog nodes supplied with heterogeneous resources in which they provide different computing functions for the mobile vehicles within its transmission range. Each fog node communicates with all the OLDs through the SDN switches. Firstly, it is assumed that the fog node i received an offloaded request (e.g., face recognition) from a vehicle moving within its service area. This request has taken a service time, where some of the requests require intensive computation (processing) and have tight delay constraints, such as the application of image-aided navigation, natural language processing, and interactive gaming [29]. Therefore, we define $\mathcal{L}$ which is the required amount of processing for the request [11, 29], and this request is constrained by the deadline δ at the fog node . We assume that there are two control functions which have

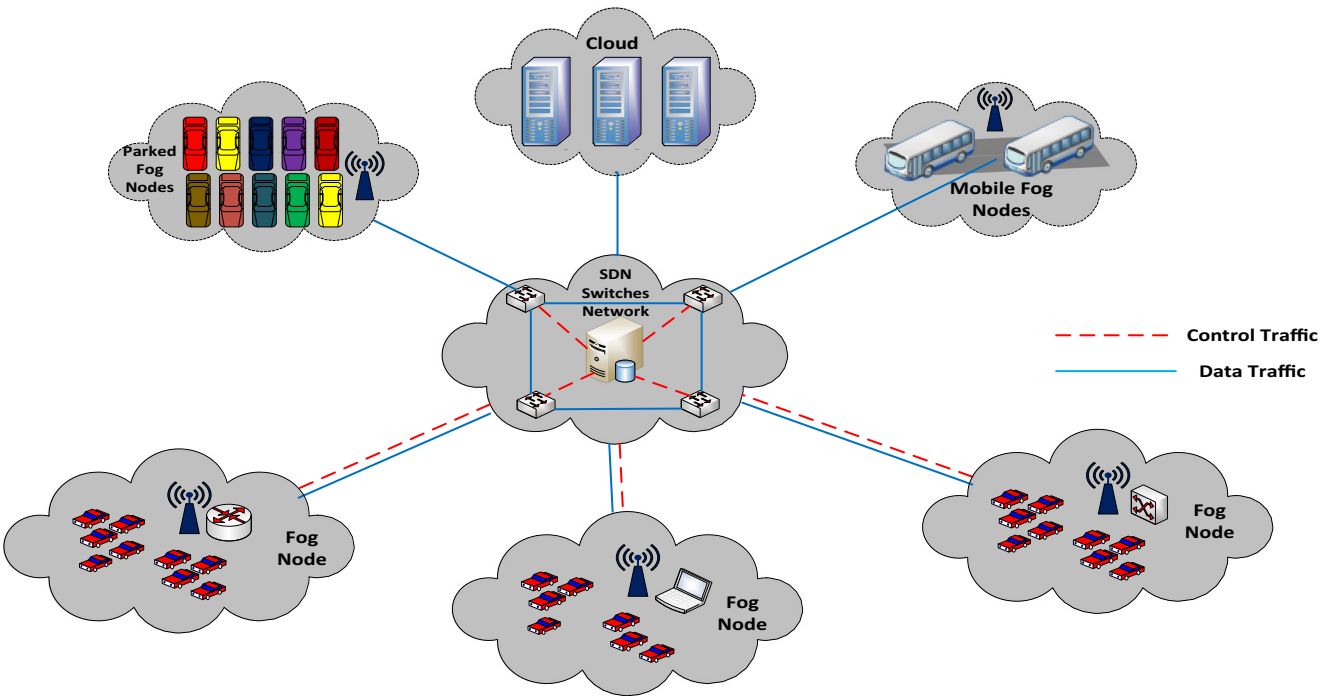

Fig. 1 The proposed offloading architecture model

Table 2 Symbol definition

Symbol	Definition
F	Number of fog nodes.
N	Number of cloud servers.
P	Number of parked fog nodes.
M	Number of mobile fog nodes.
R	Number of feasible requests.
$\overline{R}$	Number of infeasible requests.
λ_i	Arrival rate requests to fog node i.
λ_i^f	Arrival rate of feasible request to fog node i.
$\lambda_i^{\overline{f}}$	Arrival rate of infeasible request to fog node i.
μ_i	Service rate at fog node i.
$\mathcal{L}$	Required amount of processing for the request.
$\mathcal{L}_r\ \lambda_i^f$	Arrival rate of instructions of the feasible request to fog node i.
$\mathcal{L}_{\overline{r}}\ \lambda_i^{\overline{f}}$	Arrival rate of instructions of infeasible request to fog node i.
$\delta_{(i,\overline{r})}$	Deadline of infeasible request.
Δ_j^N	Arrival rate of instructions of infeasible request to cloud server j .
Δ_k^P	Arrival rate of instructions of infeasible request to parked fog node k.
τ	Communication delay from and to SDN switches network.
$C_{(i,\overline{r})}^F$	Average processing cost of infeasible request at fog node i.
S	Stretch time optimization parameter for infeasible request at OLD.

been delegated to the local SDN agent from the central controller and running on the fog node, where it could be run as an application on it [19, 33]. The first control function calculates the average completion time of the incoming request, while the second control function checks the feasibility condition for the incoming request at the fog node. Also, this local SDN agent can help the fog node to avoid the communication lost to the central controller or the single point of failure problem. In other words, the fog node can run a standard protocol to exchange the messages with the other nodes. Due to the heterogeneous nature of the fog nodes in terms of types, environment deployment, and providers [6, 34, 35], not any queuing model can be considered to characterize the fog node to obtain the required information. We consider the requests arrival rate at the fog node, i.e., λ_i, follows a Poisson distribution and the service rate μ_i is exponentially distributed [36]. Depending on the feasibility condition, the incoming requests to the fog node i can be classified into two types:

- *Feasible request type*

This type of request is processed locally at the fog node i when it meets the feasibility condition, which is,

$$\mathcal{L}\,\mu_i^{-1} + W_i < \delta \wedge (\mathcal{L}\,\lambda_i) < \quad \mu_i, \tag{1}$$

From (1), the feasibility condition consists of two sub-conditions. The first one is $\left[\mathcal{L}\,\mu_i^{-1} + W_i\right]$ that represents the average completion time of the incoming request. Here Where $\left[\mathcal{L}\,\mu_i^{-1}\right]$ is the average processing delay of the request, and W_i is the average waiting time at the fog node i which multiplies the average processing time of each request by the number of requests v_r of the same value in the queue, plus the average processing delay of the request under the service. The average waiting time for the fog node i can be obtained by,

$$W_i = \mu_i^{-1} \sum_{r \in R} \mathcal{L}_r v_r + \frac{1}{\mu_i} \quad , \forall i \in F. \tag{2}$$

Whereas, the second sub-condition $(\mathcal{L}\,\lambda_i) < \quad \mu_i$ explains the arrival rate of instructions associated with the incoming request that should be smaller than the processing capability of the fog node to ensure the stable state conditions of the fog system [37]. Hence, the arrival rate of the feasible requests λ_i^f that have been accepted at the fog node i with probability P_1 can be given as follows:

$$\lambda_i^f = \lambda_i\, P_1 \quad \forall i \in F. \tag{3}$$

- *Infeasible request type*

This type of requests will be forwarded to the SDN switches network which will then obtain the processing service from the other OLD because it violates the feasibility condition, in which:

$$\mathcal{L}\,\mu_i^{-1} + W_i > \delta \mathsf{v} (\mathcal{L}\,\lambda_i) > \mu_i, \tag{4}$$

According to the above condition, $\lambda_i^{\overline{f}}$ is defined as the arrival rate of the infeasible requests

$$\lambda_i^{\overline{f}} = \lambda_i P_2 \ , \forall i \in F. \tag{5}$$

where, P_2 is the probability of the offloading infeasible request. Therefore, the arrival rate of instructions of infeasible request can be defined:

$$\mathcal{L}_{\overline{r}} \lambda_i^{\overline{f}}, \forall i \in F, \forall \overline{r} \in \overline{R}. \tag{6}$$

Figure 2 illustrates the applicable processing procedure at the fog node i for the incoming request, where this procedure is based on satisfying the feasibility conditions.

3.2.2 Cloud server computing

The fog node can forward the infeasible request to the cloud. The cloud is considered a good option for the intensive computation request because the cloud servers provide a strong computation capability which leads to a low processing delay. However, This OLD is time-consuming because of the far distance between the fog node and the cloud as well as the bandwidth-constrained of the Internet; subsequently, this delay increases the response time of the offloaded request, particularly the delay-sensitive request. Therefore, to overcome the backbone delay, we propose that the SDN switch network can be connected with the cloud servers through a high-speed fiber link with a sufficiently large capacity [38, 39]. To calculate the average response time, the M/M/n queuing model is employed to characterize the cloud server, where n stands for the number of the virtual machines. For simplicity and without loss of generality, we assume that each of the virtual machines has the same service rate μ_j^N, thus the average delay of the infeasible request at the cloud server includes the queuing delay and the processing delay,

$$t_j^c = \frac{C(n,\rho)}{\left(n\mu_j^N - \Delta_j^N\right)} + \frac{1}{\mu_j^N}, \tag{7}$$

$$where\ \Delta_j^N = \sum_{i \in F} \sum_{\overline{r} \in \overline{R}} \mathcal{L}_{\overline{r}} \lambda_i^{\overline{f}} + MIPS_{others}^N$$

$$\text{and } C(n,\rho) = \frac{\left(\frac{(n\rho)^n}{n!}\right)\left(\frac{1}{1-\rho}\right)}{\sum_{k=0}^{n-1}\frac{(n\rho)^k}{k!} + \left(\frac{(n\rho)^n}{n!}\right)\left(\frac{1}{1-\rho}\right)}.$$

$C(n,\rho)$ denotes the Erlang's C formula [37] and $\rho = \Delta_j^N / \left(n\mu_j^N\right)$ is the occupation rate per virtual machine. Intuitively, the cloud server is faster than the fog node [24]; as a result, the processing delay at the cloud server will be less. To calculate the average network delay, namely, a delay from the SDN switches network to all the connected computing nodes directly (one hop) or indirectly across the multiple-hop, the SDN controller can use Dijkstra's algorithm. In this paper, the average network delay t includes the number of network hop and the communication delay τ (propagation and transmission delay). So, the average network delay between the fog node i and the cloud server j can be formulated as:

$$t_{i,j} = h_i \tau_{i,sw} + x_j \tau_{sw,j} \tag{8}$$

Where h_i and x_j stand for the hops count from the fog node i to the SDN switches network and from the SDN switches network to the cloud server j, respectively. Also $\tau_{i,\,sw}$ and $\tau_{sw,\,j}$ denote the communication delay from the fog node i to the SDN switches network, and from the SDN switches network to the cloud server j, respectively. Therefore, for simplicity but without loss of generality, we assume that the transmission link in terms of the distance segment and the bandwidth capacity between the computing resources is constant; thus, Eq. (8) will be updated as follows:

$$t_{i,j} = \left(h_i + x_j\right) \tau, \tag{9}$$

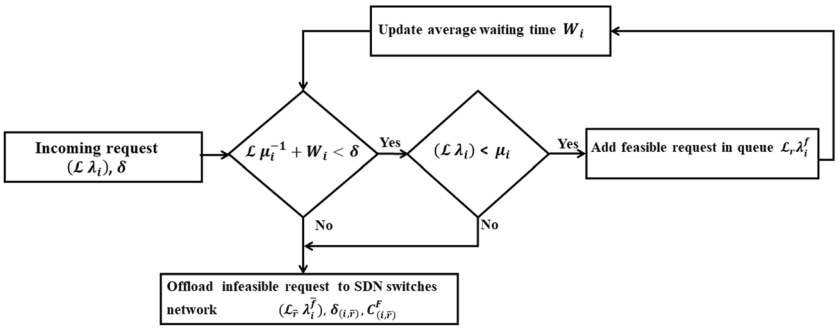

Fig. 2 The processing procedure of the fog node

Finally, the average response time for the infeasible request $\bar{r}$ from the fog node i at the cloud server j is given by:

$$T^N_{\left(i,\bar{r}\right),j} = \left(h_i + x_j\right)\tau + \frac{C(n,\rho)}{\left(n\mu_j^N - \Delta_j^N\right)} + \frac{1}{\mu_j^N} \quad . \tag{10}$$

3.2.3 Vehicular fog computing

Generally, in the VANETs, any node has communication, computation, and storage resources which can become a fog node [15]. Therefore, the VFC (parked and moving vehicles) architecture is designed [16] to support high computation and communication service for vehicular applications; As such, both parked and moving vehicles can be used as fog nodes. This computing resource model can be exploited as a cost-based OLD, where the underutilized or unused resources of these vehicles are abundant enough to execute big computation requests. In this paper, the two types of VFC have been utilized as a computing OLD.

- *Parked fog node*

The parked vehicles spend a long time in the parking lot, whereas their processing resources are idle. Besides, they have a stable communication channel with the access point because they have a fixed position in the parking [40]. Therefore, we can use these resources with a rechargeable battery [16] to process and transfer high computation requests.

Here, we assume there is a parking lot $\mathcal{P}$ which can be seen as a cluster composed of a set of parked fog nodes under the responsible one access point. Therefore, it is proper to model each parked fog node k as M/M/1 queuing model. Due to diverse models and types of the parked vehicles, each parked fog node has a specified service rate μ_k^P.The access point transmits the status information of the cluster which includes the available resource, unit cost, and the communication delay to the SDN switches network. Based on this information, the central controller assigns each infeasible request to the suitable OLD. The selected parked fog node k should satisfy the minimum average response time in terms of less queuing delay, the average network delay (low hop count) with an acceptable processing cost. The average delay at the parked fog node k to complete the infeasible request is given by:

$$t_k^P = \frac{1}{\mu_k^P - \Delta_k^P} \quad , \tag{11}$$

where $\Delta_k^P = \sum_{i \in F} \sum_{\bar{r} \in \bar{R}} \mathcal{L}_{\bar{r}} \lambda_i^{\bar{f}} + MIPS_{others}^N$

Similarly, we can infer the average network delay $t_{i,k}$ from the fog node i to the parked fog node k, in which the hop count to the SDN switches network is described by y_k; therefore, the average network delay is given by:

$$t_{i,k} = \left(h_i + y_k\right)\tau \quad , \tag{12}$$

Thus, the average response time for the infeasible request $\bar{r}$ from the fog node i at the parked fog node k will be:

$$T^P_{\left(i,\bar{r}\right),k} = \left(h_i + y_k\right)\tau + \frac{1}{\left(\mu_k^P - \Delta_k^P\right)} \quad . \tag{13}$$

- *Mobile fog node*

The moving vehicles (fixed trajectory buses) are another computing OLD which can be utilized to process the infeasible request, where the mobile fog nodes (moving vehicles) provide their computing services on-demand with an efficient cost [41] via an access point for the surrounding vehicles or the fog nodes. Initially, we propose there is a set $\mathcal{M}$ of mobile fog nodes that have daily mobility on a fixed route with a constant speed [31] and there is an access point installed at the route to provide the wireless access service within a transmission range. The mobile fog nodes have registered their resources, attributes, and the unit cost per resource at the access point. All the information on the mobile fog nodes is updated periodically whenever they move close to the access point. Additionally, the access point caches such information in a vehicle table which contains the ID (vehicle number), the available resources (free servers), and the unit cost. When a mobile fog node l located at the position has a high coverage level, it transmits a beacon packet to the access point for activating this tuple of information. The access point delivers this information of the current mobile fog nodes periodically to the central controller through the SDN switches network, thus the controller obtains complete knowledge about all the available resources in the network and selects the optimal OLD to process the infeasible request. The access point can estimate the connection lifetime ϕ_l with the mobile fog node l [21], where the mobile fog node l should satisfy the processing requirement and has the connection lifetime ϕ_l greater than the deadline $\delta_{(i,\bar{r})}$ and the average response time $T^M_{(i,\bar{r}),l}$ of the infeasible request at the mobile fog node l . So, this constrain can

distinguish between the mobile and the parked fog nodes which are static. To calculate the average delay t_l^M of the infeasible request at the mobile fog node l, we have to formulate this computing resource according to M/M/m/m queue model (m servers, no waiting). This model is suitable for this computing resource because each moving vehicle has a variant connection lifetime with the access point, thereby, buffering (queuing) the request will be useless after moving far from the transmission range of the access point. In this model, the incoming request is not permitted to enter the system when all the servers are busy and blocked with the probability P_b . To avoid that, we assume the mobile fog nodes send their available resources (free servers) to the access point when they enter its transmission range, thus the blocking probability will be zero. Consequently, the average delay of the infeasible request at the mobile fog node l that has a service rate μ_l^M will be:

$$t_l^M = \frac{1}{\mu_l^M} \quad , \tag{14}$$

Also, the average network delay from the fog node i to the mobile fog node l will be:

$$t_{i,l} = (h_i + z_l)\,\tau \quad , \tag{15}$$

where, z_l denotes the hop count from the mobile fog node l to the SDN switches network. Finally, the average response time for the infeasible request $\overline{r}$ from the fog node i at the mobile fog node l could be formulated as below:

$$T^M_{\left(i,\overline{r}\right),l} = (h_i + z_l)\,\tau + \frac{1}{\mu_l^M} \quad . \tag{16}$$

3.3 Average processing cost model

The user has to pay for the fog node, the cloud servers [30], and the VFC [16] when he or she uses their resources. That means, whenever the fog node performs an infeasible request on another OLD, there are cost utilities associated with the offloaded requests. The average offloading cost includes the processing and communication cost; yet, in this paper, we considered only the cost associated with using the processing resource [11, 31] without the communication cost [30, 31]. Therefore, we will model the average processing cost for the infeasible request at the fog node as well as the OLDs; afterward, we compared them. We assume that the unit cost per MIPS (Millions of instructions per second) for the fog node i is C_i, thereby the average processing cost at the fog node is given by:

$$C^F_{\left(i,\overline{r}\right)} = C_i\,\mathcal{L}_{\overline{r}}\,\lambda_i^{\overline{f}}, \tag{17}$$

The VFC has a low cost in terms of infrastructure and application deployment [42]; therefore, intuitively the processing cost at the VFC will be relatively less compared with the cloud servers or other fog nodes. Generally, the processing cost at the park and the mobile fog nodes is different and is determined by the providers and the locations. To model the average processing cost of the infeasible request on each one of these OLDs, we assume that the unit cost per MIPS for the cloud server, the parked, and the mobile fog nodes are C_j, C_k and C_l, respectively. In the same manner, the average processing cost at each of the OLD is given by:

$$C^N_{\left(i,\overline{r}\right),j} = C_j\,\mathcal{L}_{\overline{r}}\,\lambda_i^{\overline{f}}, \tag{18}$$

$$C^P_{\left(i,\overline{r}\right),k} = C_k\,\mathcal{L}_{\overline{r}}\,\lambda_i^{\overline{f}} \quad , \tag{19}$$

$$C^M_{\left(i,\overline{r}\right),l} = C_l\,\mathcal{L}_{\overline{r}}\,\lambda_i^{\overline{f}} \quad . \tag{20}$$

3.4 Problem formulation

In this paper, the SDN controller on behalf of the fog node will select the optimal OLD by comparing the required deadline of the infeasible request with an average response time of each available OLD; to do so, we define a new optimization parameter named the stretch time S which is the available difference time between them, where the stretch time for each OLD is given by:

$$S^N_{\left(i,\overline{r}\right),j} = \delta_{\left(i,\overline{r}\right)} - T^N_{\left(i,\overline{r}\right),j} \quad , \tag{21}$$

$$S^P_{\left(i,\overline{r}\right),k} = \delta_{\left(i,\overline{r}\right)} - T^P_{\left(i,\overline{r}\right),k} \quad , \tag{22}$$

$$S^M_{\left(i,\overline{r}\right),l} = \delta_{\left(i,\overline{r}\right)} - T^M_{\left(i,\overline{r}\right),l} \quad , \tag{23}$$

Therefore, the selection decision problem is formulated as a Binary-Linear Programming and the objective function can be defined as follows:

$$\mathrm{O}\left(i,\overline{r}\right)=\underset{j,k,l}{\operatorname{argmax}}\left(\sum_{j\in N}\alpha_j S^N_{\left(i,\overline{r}\right),j}+\sum_{k\in P}\beta_k S^P_{\left(i,\overline{r}\right),k}+\sum_{l\in M}\gamma_l S^M_{\left(i,\overline{r}\right),l}\right) \quad (24)$$

$$\mathbf{s.t.}C1:\ \alpha_j,\beta_k,\gamma_l\in\{0,1\},\quad \forall j\in\mathcal{N},k\in\mathcal{P},l\in\mathcal{M},$$

$$C2:\ \sum_{j\in N}\alpha_j+\sum_{k\in P}\beta_k+\sum_{l\in M}\gamma_l=1,$$

$$\forall j\in\mathcal{N},k\in\mathcal{P},l\in\mathcal{M},$$

$$C3:\ \sum_{j\in N}\alpha_j T^N_{\left(i,\overline{r}\right),j}+\sum_{k\in P}\beta_k T^P_{\left(i,\overline{r}\right),k}+\sum_{l\in M}\gamma_l T^M_{\left(i,\overline{r}\right),l}\ \le\delta\left(i,\overline{r}\right),$$

$$\forall i\in F,\forall\overline{r}\in\overline{R},$$

$$C4:\ \sum_{j\in N}\alpha_j C^N_{\left(i,\overline{r}\right),j}+\sum_{k\in P}\beta_k C^P_{\left(i,\overline{r}\right),k}+\sum_{l\in M}\gamma_l C^M_{\left(i,\overline{r}\right),l}\le C^F_{\left(i,\overline{r}\right)},$$

$$\forall i\in F,\forall\overline{r}\in\overline{R},$$

$$C5:\ 0\le\Delta^N_j\le\mu^N_j,\qquad\forall j\in\mathcal{N},$$

$$C6:\ 0\le\Delta^P_k\le\ \mu^P_k,\qquad\forall\ k\in\mathcal{P},$$

$$C7:\ T^M_{\left(i,\overline{r}\right),l}\ \le\delta\left(i,\overline{r}\right)\le\phi_l,\quad\forall\ l\in\mathcal{M},$$

$$C8:S^N_{\left(i,\overline{r}\right),j},S^P_{\left(i,\overline{r}\right),k},S^M_{\left(i,\overline{r}\right),l}\ \ge0.$$

where the objective function $\mathrm{O}(i,\overline{r})$ can be translated as selecting the index (j,k,l) of the OLD that has the maximum stretch time S for processing the infeasible request $\overline{r}$ of the fog node i. C1 and C2 make sure that among α_j, β_k and γ_l, one and only one of them is one at any time; C3 is ensuring each infeasible request which should be performed before the deadline; C4 is to guarantee the average processing cost of the infeasible requests at each OLD as much as possible which should not exceed the average processing cost at the fog node; C5 and C6 are to ensure the required service rate for the arrival rate of the instructions for the infeasible requests should not exceed the service rate of the OLD; C7 ensures that the average response time and the connection duration of the mobile fog node should meet the needed deadline for the infeasible request; At last, C8 is the non-negative constraint on the stretch time parameter of each OLD. The mathematical model is solved using the CPLEX software and two examples are implemented to explain how this model works.

Example 1 This example examines the efficiency of this model in optimizing the selection decision, where the fog node was identified by the id number (ID) 55, service rate of 200 MIPS, and 0.2 s communication delay to the SDN switch network. The infeasible request has a deadline of 2.4 s, the arrival rate of instructions of 1500 MIPS, and the average processing cost of 2 unit costs. The data of the related OLDs is tabulated in Table 3. The execution time of the mathematical model at the CPLEX was 0.33 s and the objective function is optimized via selecting the optimal OLD for this fog node, which is a mobile fog node (701id) with an average response time equal to 0.3 microseconds and an average processing cost equal to 1.5 unit costs.

Example 2 In this example, some parameter values of the OLDs have been changed while being observed to know how they can affect the selection decision. The fog node was identified by the id number 55, a processing capability of 200MIPS, and 0.2 s communication delay to the SDN switch network. The fog node offloads the infeasible request that has 2.4 s deadline, 1000 MIPS an arrival rate of instructions, and 1.5 average processing costs. The data of the related OLDs is presented in Table 4. The execution time of the mathematical model at the CPLEX is 0.37 s. So, the optimal OLD is a parked fog node (650 id) which has the maximum stretch time (minimum average response time) equal to 1.1 microseconds and the average processing cost equal to 1 unit cost.

4 SDN-based offloading policy

This policy is proposed to improve the offloading selection decision process by selecting the optimal OLD and guarantee the required deadline of the infeasible request as well as the average processing cost for the fog node, where the fog node can benefit from the powerful centralized feature of the SDN controller that has the global conditions of all the computing resources. As illustrated in Fig. 3, the application plane of the SDN controller contains two modules, which are the OLD information collection module and the OLD selection decision module. The former is responsible for collecting the information about the OLDs which are updated periodically, where each one of them sends the information to the controller (OLD information collection module) about its status including the ID node, the average waiting time, the communication delay, the processing capability, and the unit cost through the SDN switches network. The OLDs have been stored in form of a

Table 3 The related data of Example 1

OLD	ID	Queuing delay	Comm. delay	Service rate	Cost
Cloud server	100	0.5	0.9	6000	0.3
Cloud server	200	0.6	0.5	7000	0.6
Cloud server	300	9.8	3.5	8000	0.8
Parked fog	450	0.2	0.3	4000	0.004
Parked fog	550	0.2	2.8	3000	0.03
Parked fog	650	0.1	0.4	2000	0.08
Mobile fog[1]	**701**	**x**	**0.1**	**2000**	**0.001**
Mobile fog	801	x	0.4	3000	0.002

[1] Optimal OLD of example 1

Table 4 The related data of Example 2

OLD	ID	Queuing delay	Comm. delay	Service rate	Cost
Cloud server	100	10.5	10.9	5000	0.1
Cloud server	200	0.5	1.5	7000	0.2
Cloud server	300	9.8	3.5	8000	0.5
Parked fog	450	0.2	0.3	2000	0.002
Parked fog	550	0.2	2.8	1000	0.009
Parked fog[1]	**650**	**0.1**	**0.9**	**2000**	**0.001**
Mobile fog	701	x	5.2	3000	0.001
Mobile fog	801	x	6.3	3500	0.007

[1] Optimal OLD of example 2

table and ordered according to the smallest average waiting time. The latter is responsible for processing the infeasible request that is received from the OLD information collection module which will be through selecting the optimal OLD from the stored table then returning the ID node to the corresponding SDN switch.

First of all, the fog node sends the infeasible request to the SDN switch that stores the requested data then sends an encapsulated message (reactive mode) to the controller (OLD information module). This message contains essential information about the infeasible request such as the deadline, the arrival rate instructions, and the average processing cost. The controller schedules the received messages of the SDN switches based on the minimum deadline. Following that, the controller calculates the average response time and the average processing cost for the infeasible request at each available OLD. Afterward, it compares them with the deadline via calculating the stretch time (**S**) only for the OLD that has the average response time less than the deadline of the infeasible request and the average processing cost less than the specified average processing cost of the fog node, whereas the others that have a big value will be excluded. After a while, the optimal OLD that satisfied the requirements of the fog node will be chosen. Subsequently, the ID of the selected OLD in the form entry flow table is directed to the corresponding SDN switch which will forward the data of the infeasible request to the selected OLD. Finally, the result of the infeasible request after processing will be sent back to the corresponding fog node via this SDN switch. Algorithm 1 is the pseudo-code of the proposed offloading policy.

Algorithm 1 the SDN-based offloading policy

Input: $\delta_{(i,\bar{F})}, \mathcal{L}_{\bar{F}}\ \lambda_i^{\bar{f}}, C_{(i,\bar{F})}^{F}, n$
Output: $\mathrm{OLD_{ID}}$
Initialize: $Array\ S \leftarrow [\ ]$
Procedure Finding $\mathrm{OLD_{ID}}$
for each *ID* in ***n*** tables **do**
  Calculate $\boldsymbol{T_{ID}}$ and $\boldsymbol{C_{ID}}$ of the infeasible request
  if $(T_{ID} \leq \delta_{(i,\bar{F})}$ and $C_{ID} \leq C_{(i,\bar{F})}^{F}$ **then**
    Calculate $s_{ID} \leftarrow \delta_{(i,\bar{F})} - T_{ID}$
    $S \leftarrow s_{ID}$
  else
    Continue;
  end if
end for
$\mathrm{OLD_{ID}} \leftarrow$ argmax S
return $\mathrm{OLD_{ID}}$
end procedure

5 Performance evaluation

Simulation results and performance evaluations are presented in this section. First, we describe the simulation environment setting and a greedy offloading policy. Then, we evaluate the performance of the proposed offloading policy against this greedy and the SVMEC offloading policy [31]. SVMEC was proposed to expand the capacity of the MEC by renting resources from a remote cloud and vehicular cloud, thus process its offloading tasks. After that, the evaluation results in terms of some metrics have been presented and discussed in different scenarios for these policies.

5.1 Simulation setup

In this paper, the simulation results were obtained by using the MATLAB environment [43], and the proposed architecture was utilized for these policies. The network consists of 5 hybrid SDN switches that can operate as the SDN switches with a controller for conducting the proposed offloading policy or traditional switches without a controller for conducting other policies. The parameters used in the simulation are given in Table 5.

5.2 Greedy offloading policy

A greedy algorithm is an algorithmic paradigm that follows the problem-solving approach of making a locally optimal choice at each stage with the hope of finding a global optimum. In other words, it is local optimization and does not provide a globally optimum solution. This approach does not intend to find the best solution, but it terminates in a reasonable number of steps; finding an optimal solution to such a complex problem typically requires many unreasonable steps [44–46]. Greedy algorithms mostly (but not always) fail to find the globally optimal solution because they usually do not operate exhaustively on all the data like dynamic programming. Nevertheless, they are

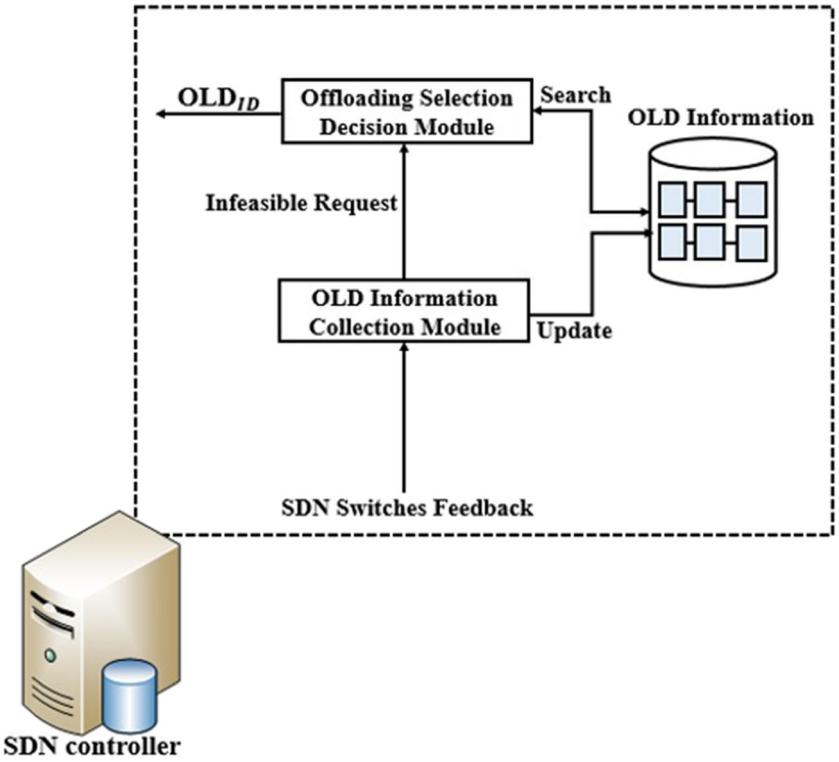

Fig. 3 The application plane of the SDN controller

simple, easy to implement, and run fast as well as giving a good approximation to the optimum.

A greedy offloading policy for our problem goes as follows: At each offloading stage, the fog node broadcasts a query message about the node ID, the network delay, the waiting time, and the unit cost to all the OLDs in the network. Then, it waits for a certain time to receive a response from the nearby OLDs. After receiving the above information, the fog node calculates the ratio of the average response time and the average processing cost for the infeasible request at each OLD. The average response time here includes the network delay and the waiting time without the processing delay. Then, the fog node arranges them in an ascending order based on a minimum ratio. Afterward, the fog node selects each one to compare it with the deadline and the average processing cost ratio, in which the numerator and denominator satisfy the comparison. Whenever there is more than one OLDs with the same ratio, the fog node gives a high priority for the lower response time with sacrificing the average processing cost. Finally, the data of the infeasible request will be offloaded by the fog node to the selected OLD to attain the result. Consequently, waiting until getting a response from the available OLDs as well as recalculating the required information to select the optimal OLD adds an overhead time to the fog node. Besides, increasing the number of OLDs extends the operating time complexity, thus degrading the performance of the offloading process in terms of the average response time.

5.3 Performance metrics

5.3.1 Average response time

Figure 4 shows the average response time of the infeasible request in three policies for different arrival rate instructions.

Table 5 Simulation parameters

Parameters	Values
Number of fog nodes	4
Number of cloud servers	5, 7, 10
Number of parked fog nodes	5, 7, 10
Number of mobile fog nodes	5, 7, 10
Number of OLDs	5 – 30
Arrival rate instructions $\left(\mathcal{L}_{\bar{r}}\ \lambda_i^{\bar{I}}\right)$	2000 12,000 MIPS
Deadline of infeasible request $\left(\delta_{(i,\bar{r})}\right)$	5 25 s
Average processing cost of fog node $(C^F_{(i,\bar{r})})$	5, 10 (unit cost)

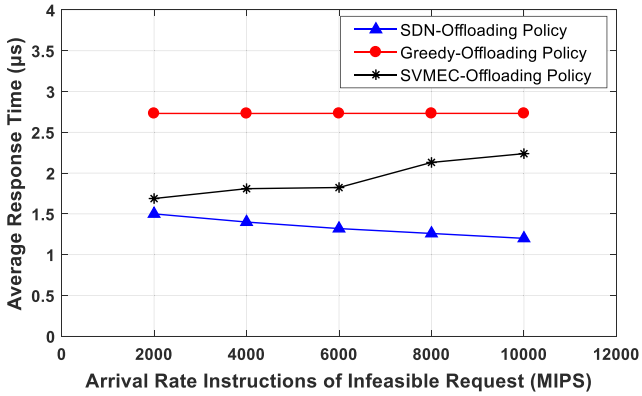

Fig. 4 The average response time

Here an infeasible request has been used with a deadline of 5 s and an average processing cost of 10 unit cost per MIPS at the fog node. As seen, the proposed policy has a lower average response time compared to other policies when the arrival rate instructions of infeasible requests are increased. The reason is that our proposed tends to get a maximum processing resource capability among all the OLDs with an acceptable average processing cost to lower the processing delay, thus reducing the average response time. The SVMEC policy has a higher average response time because it utilizes the sum of weighted parameters of the response time and resource cost for selecting the OLD, therefore SVMEC sometimes prefers the cost on the response time through these parameters. The greedy policy has the worst average response time. The reason is that it does not consider the capability processing of the selected OLDs that may have a value less than the required computing of the infeasible request. Thus, this can add a long processing delay.

5.3.2 Meeting the deadline

The meeting deadline in terms of the stretch time in both policies (proposed and greedy offloading policy) with different deadline values has been presented in Fig. 5. The arrival rate instructions of the infeasible request and the average processing cost of the fog node that used here are 5000 MIPS and 10 unit cost per MIPS, respectively. From Fig. 5, it is observed that the meeting deadline for both policies increase linearly and tend to meet the deadline even if the deadline is high. At the same time, it is observed that the meeting deadline in the proposed policy is higher than the greedy policy, which means the selected average response time for the latter is less than the average response time for the former. The reason is that the selection decision of the proposed policy considers the average response time as an objective and the average processing cost as a constraint. Whereas the selection decision of the greedy policy considers the result of the ratio (average response time and average processing cost) as an objective, thus they are affected by each other value.

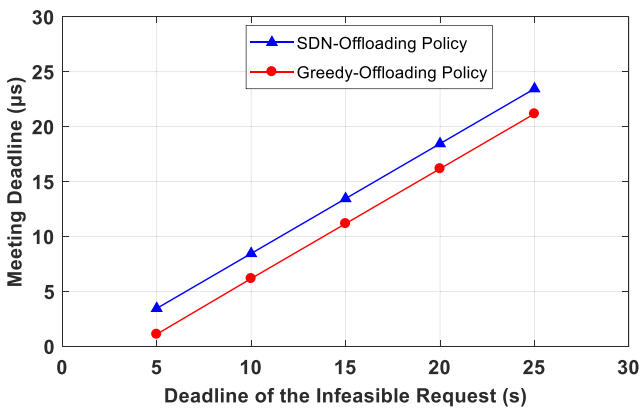

Fig. 5 The meeting deadline

5.3.3 Average processing cost

Figure 6 explains the relation between the deadline of the infeasible request and the acceptable average processing cost. The deadlines that are used here are 5, 10, 15, 20, and 25 s and the arrival rate instructions of the infeasible request are 5000 MIPS. From Fig. 6, it can be seen that the proposed policy tends to get a lower average processing cost among the OLDs when the deadline of the infeasible request is high. As a result, the proposed policy takes into account the compatibility between the required deadline of the infeasible request and the satisfactory average processing cost of the fog node. Whereas the greedy policy has no difference between the high or low deadline and it tends to be constant at a lower average processing cost value.

5.3.4 Reliability of response

Figure 7 illustrates the performance of the proposed policy through the reliability of the response of the OLDs. In other words, it has been investigated in the number of

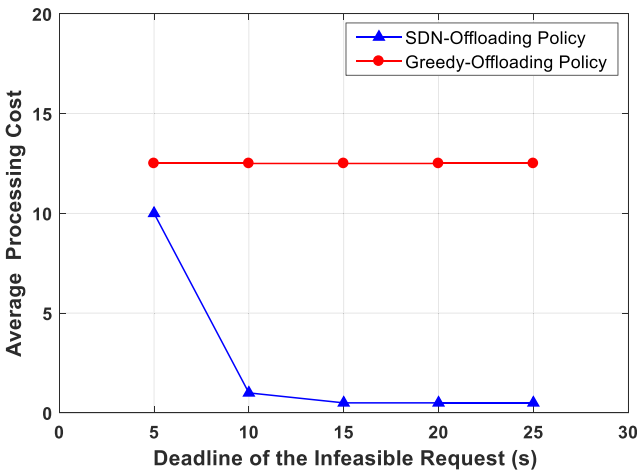

Fig. 6 The average processing cost

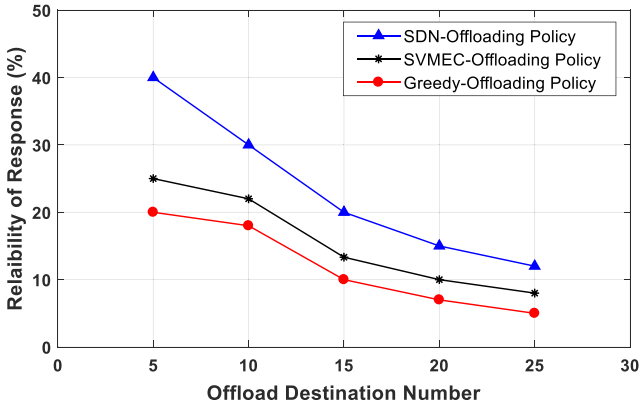

Fig. 7 The reliability of response

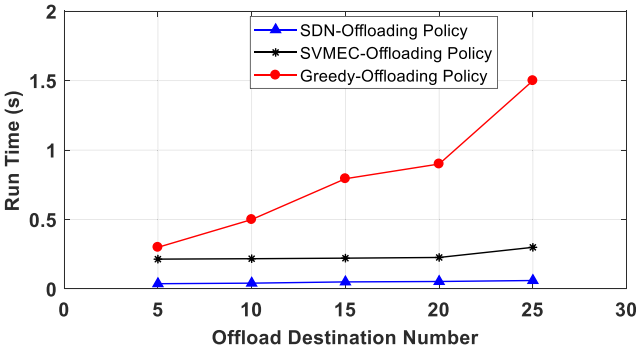

Fig. 8 Comparison of scalability

OLDs that satisfy the deadline and the average processing cost. The reliability of the response can be concluded by calculating the ratio of the number of the OLDs satisfying the requirement of the infeasible requests to the total number of the available OLDs in the network. In this scenario, the number of OLDs has ranged from 5 to 25 which are selected randomly in their numbers. Moreover, the deadline, average processing cost, and the arrival rate of instructions of infeasible request are 10 s, 10 unit cost per MIPS, and 5000 MIPS, respectively.

From Fig. 7, it can be seen that by increasing the number of the OLD, the reliability of the response for these policies declines, but at each point, the reliability of the response for the proposed policy is the best. That is, the proposed policy can increase the processing chance of the infeasible request by increasing the number of the selected OLDs. Conversely, the SVMEC policy considers only the cloud and a limited number of mobile fog nodes as an OLD, where some times many of the mobile fog nodes do not satisfy the SVMEC offloading condition which is connection duration with the fog node. As a result, the reliability of the response will be down. The same thing for the greedy policy which selects a few numbers of the OLDs based on its local information.

5.3.5 Scalability

To evaluate the scalability of the proposed policy, we compare its run time with the SVMEC and greedy offloading policy for different numbers of OLDs. All other parameters are initialized as the previous metric. As shown in Fig. 8 the run time of the proposed policy and SVMEC are unrelated to the number of the OLDs, whereas the greedy policy's run time is almost exponential to the increasing number of the OLDs. Also, we can see our proposed is the best for some reasons that are: in the SVMEC, the solution of the problem consists of two dependent sub-solutions which are resource allocation and OLD selection. The former sub-solution can take a long time with an increasing number of OLDs that make the run time is bigger than our proposed. In the greedy, the fog node takes a long time to select the suitable OLD. This time includes the waiting time to get a response of the OLDs and recalculating time for the required information, after that the optimal OLD will be selected. Conversely, our proposed utilizes the global network status information, which can reduce the time that is used to select a suitable OLD.

6 Conclusions

In this paper, a new hybrid offloading architecture was introduced for fog-vehicular networks to increase the processing chance of the offloaded requests. The VFCs (parked and moving vehicles) have been joined to be an OLD, in which the VFCs offer their underutilized resources on-demand with an effective cost. Moreover, a new offloading policy based on the SDN was proposed to make a correct offloading decision on behalf of the fog node to select the optimal OLD which meets the deadline of the offloaded request and the average processing cost of the fog node. Also, it helps to reduce the burden of the fog node regarding the aware topology and the status of the OLDs in the network. The simulation results proved that the proposed offloading policy exceeds the SVMEC and greedy offloading policy in many metrics and it is more suitable for a high computation request with a low deadline in the fog-vehicular network.

References

1. Biswas S, Tatchikou R, Dion F (2006) Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. IEEE Commun Mag 44(1):74–82

2. Saini M, Alelaiwi A, Saddik AE (2015) How close are we to realizing a pragmatic VANET solution? A meta-survey. ACM Computing Surveys (CSUR) 48(2):29
3. Kai K, Cong W, Tao L (2016) Fog computing for vehicular ad-hoc networks: paradigms, scenarios, and issues. The Journal of China Universities of Posts and Telecommunications 23(2):56–96
4. Stojmenovic I (2014) Fog computing: a cloud to the ground support for smart things and machine-to-machine networks. In: Telecommunication Networks and Applications Conference (ATNAC), 2014 Australasian, pp 117-122. IEEE
5. Stojmenovic I, Wen S (2014) The fog computing paradigm: scenarios and security issues. In: Computer Science and Information Systems (FedCSIS), Federated Conference on 2014, pp 1-8. IEEE
6. Hu P, Dhelim S, Ning H, Qiu T (2017) Survey on fog computing: architecture, key technologies, applications and open issues. J Netw Comput Appl 98:27–42
7. Aazam M, Zeadally S, Harras KA (2018) Offloading in fog computing for IoT: review, enabling technologies, and research opportunities. Futur Gener Comput Syst 87:278–289
8. Mukherjee M, Shu L, Wang D (2018) Survey of fog computing: fundamental, network applications, and research challenges. IEEE Commun Surv Tutor 20(3):1826–1857
9. Mukherjee M, Kumar S, Zhang Q, Matam R, Mavromoustakis CX, Lv Y, Mastorakis G (2019) Task data offloading and resource allocation in fog computing with multi-task delay guarantee. IEEE Access 7:152911–152918
10. Zhu C, Pastor G, Xiao Y, Ylajaaski A (2018) Vehicular fog computing for video crowdsourcing: applications, feasibility, and challenges. IEEE Commun Mag 56(10):58–63
11. Yousefpour A, Patil A, Ishigaki G, Kim I, Wang X, Cankaya HC, Zhang Q, Xie W, Jue JP (2018) QoS-aware dynamic fog service provisioning. arXiv preprint, arXiv:1802.00800. [Online]. Available: https://arxiv.org/abs/1802.00800
12. Mukherjee M, Kumar S, Mavromoustakis CX, Mastorakis G, Matam R, Kumar V, Zhang Q (2019) Latency-driven parallel task data offloading in fog computing networks for industrial applications. IEEE Transactions on Industrial Informatics
13. Xiao Y, Krunz M (2017) QoE and power efficiency tradeoff for fog computing networks with fog node cooperation. In: INFOCOM 2017-IEEE Conference on Computer Communications, IEEE 2017, pp 1-9. IEEE
14. Kadhim AJ, Seno SAH (2018) Maximizing the utilization of fog computing in internet of vehicle using SDN. IEEE Commun Lett 23(1):140–143
15. Grover J, Jain A, Singhal S, Yadav A (2018) Real-time VANET applications using fog computing. In: Proceedings of First International Conference on Smart System, Innovations and Computing 2018, pp 683-691. Springer
16. Hou X, Li Y, Chen M, Wu D, Jin D, Chen S (2016) Vehicular fog computing: a viewpoint of vehicles as the infrastructures. IEEE Trans Veh Technol 65(6):3860–3873
17. Li C, Qin Z, Novak E, Li Q (2017) Securing SDN infrastructure of IoT–fog networks from MitM attacks. IEEE Internet Things J 4(5):1156–1164
18. Wickboldt JA, De Jesus WP, Isolani PH, Both CB, Rochol J, Granville LZ (2015) Software-defined networking: management requirements and challenges. IEEE Commun Mag 53(1):278–285
19. Tomovic S, Yoshigoe K, Maljevic I, Radusinovic I (2017) Software-defined fog network architecture for iot. Wirel Pers Commun 92(1):181–196
20. Jiang C, Cheng X, Gao H, Zhou X, Wan J (2019) Toward computation offloading in edge computing: a survey. IEEE Access 7:131543–131558
21. Zhang H, Zhang Q, Du X (2015) Toward vehicle-assisted cloud computing for smartphones. IEEE Trans Veh Technol 64(12):5610–5618
22. Zhang W, Zhang Z, Chao H-C (2017) Cooperative fog computing for dealing with big data in the internet of vehicles: architecture and hierarchical resource management. IEEE Commun Mag 55(12):60–67
23. He X, Ren Z, Shi C, Fang J (2016) A novel load balancing strategy of software-defined cloud/fog networking in the internet of vehicles. China Commun 13(Supplement2):140–149
24. Yousefpour A, Ishigaki G, Gour R, Jue JP (2018) On reducing iot service delay via fog offloading. IEEE Internet Things J 5:998–1010
25. Wang X, Ning Z, Wang L (2018) Offloading in internet of vehicles: a fog-enabled real-time traffic management system. IEEE Trans Industr Inform 14(10):4568–4578
26. He Z, Zhang D, Liang J (2016) Cost-efficient sensory data transmission in heterogeneous software-defined vehicular networks. IEEE Sensors J 16(20):7342–7354
27. LiWang M, Dai S, Gao Z, Du X, Guizani M, Dai H (2019) A computation offloading incentive mechanism with delay and cost constraints under 5G satellite-ground IoV architecture. IEEE Wirel Commun 26:124–132
28. Du J, Yu FR, Chu X, Feng J, Lu G (2018) Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization. IEEE Trans Veh Technol 68(2):1079–1092
29. Zhang K, Mao Y, Leng S, He Y, Zhang Y (2017) Mobile-edge computing for vehicular networks: a promising network paradigm with predictive off-loading. IEEE Veh Technol Mag 12(2):36–44
30. Liu L, Chang Z, Guo X, Mao S, Ristaniemi T (2018) Multiobjective optimization for computation offloading in fog computing. IEEE Internet Things J 5(1):283–294
31. Pham X-Q, Nguyen T-D, Nguyen V, Huh E-N (2019) Joint node selection and resource allocation for task offloading in scalable vehicle-assisted multi-access edge computing. Symmetry 11(1):58
32. Zhuang W, Ye Q, Lyu F, Cheng N, Ren J (2019) SDN/NFV-empowered future IoV with enhanced communication, computing, and caching. Proc IEEE 108(2):274–291
33. Truong NB, Lee GM, Ghamri-Doudane Y (2015) Software defined networking-based vehicular adhoc network with fog computing. In: Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on 2015, pp 1202-1207. IEEE
34. Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. In: Proceedings of the first edition of the MCC workshop on Mobile cloud computing 2012, pp 13-16. ACM
35. Mahmud R, Kotagiri R, Buyya R (2018) Fog computing: A taxonomy, survey and future directions. In: Internet of everything, pp 103–130. Springer
36. Wu Y, Wu J, Chen L, Yan J, Luo Y (2020) Efficient task scheduling for servers with dynamic states in vehicular edge computing. Comput Commun 150:245–253
37. Kleinrock L (1975) Queueing systems, vol 1. Wiley, New York
38. Du J, Zhao L, Feng J, Chu X (2018) Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee. IEEE Trans Commun 66(4):1594–1608

39. Guo H, Liu J (2018) Collaborative computation offloading for multiaccess edge computing over fiber–wireless networks. IEEE Trans Veh Technol 67(5):4514–4526
40. Reis AB, Sargento S, Tonguz OK (2017) Parked cars are excellent roadside units. IEEE Trans Intell Transp Syst 18(9):2490–2502
41. Xiao Y, Zhu C (2017) Vehicular fog computing: vision and challenges. In: Pervasive Computing and Communications Workshops (PerCom workshops), 2017 IEEE International Conference on 2017, pp 6-9. IEEE
42. Menon VG, Joe Prathap P (2017) Moving from vehicular cloud computing to vehicular fog computing: issues and challenges. Int J Comput Sci Eng 9(2)
43. Kim OTT, Nguyen V, Hong CS (2014) Which network simulation tool is better for simulating vehicular ad-hoc network? In: Proceedings of the Korean Information Science Society 2014, pp 930-932
44. Hazewinkel M (2001) Greedy algorithm. Encyclopedia of Mathematics
45. Gutin G, Yeo A, Zverovich A (2002) Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP. Discret Appl Math 117(1–3):81–86
46. Albu-Salih AT, Seno SAH (2018) Energy-efficient data gathering framework-based clustering via multiple UAVs in deadline-based WSN applications. IEEE Access 6:72275–72286

Alla Abbas Khadir received the B.S. in Electrical Engineering from University of Technology, Iraq, in 2001, M.E. in Computer Engineering from Hunan University, Changsha, China, in 2011, and is currently pursuing a full-time Ph.D. degree in computer Engineering with Ferdowsi University of Mashhad (FUM), Mashhad, Iran. His major research interests include vehicular ad hoc networks, software-defined networks, cloud, and fog computing.

Seyed Amin Hosseini Senoo received B.E. and M.E. degree in Computer Engineering from Ferdowsi University of Mashhad, Iran in 1990 and 1998, respectively, and a Ph.D. degree in Computer Networks from University Sains Malaysia (USM) in 2009. Currently, he is Head of Information & Communication Center at the Ferdowsi University of Mashhad. His research interests include Wireless and Sensor Networks, Network protocols, QoS and Network Security.