A Volunteer Supported Fog Computing Environment for Delay-Sensitive IoT Applications

Babar Ali, Muhammad Adeel Pasha, Senior Member, IEEE, Saif ul Islam, Houbing Song, Senior Member, IEEE, and Rajkumar Buyya, Fellow, IEEE

Abstract—Fog Computing (FC) has emerged as a complementary solution to the centralized cloud infrastructure. An FC node is available in closer proximity to users and extends cloud services to the edge of the network in a highly distributed manner. However, with an increase in streaming and delay-sensitive Internet of Things (IoT) applications, FC also needs to address the issue of higher latency while forwarding compute-intensive jobs to remote cloud data centers. Hence, there is a need to investigate the use of computational resources at the edge of the network. Volunteer Computing (VC) offers a reduction in the cost of maintaining high performance computing by making use of user-owned underutilized or idle resources, e.g., laptops and desktop computers closer to fog devices. We propose Volunteer Supported Fog Computing (VSFC), as a computing paradigm, that explores the interplay of these two distributed computing domains to help minimize inherent communication delays of cloud computing, energy consumption and network usage. To this effect, we have extended iFogSim toolkit to support VSFC. Extensive simulations show that VSFC outperforms traditional FC-cloud computing by reducing delay by 47.5%, energy by 93% and network usage by 92% under normal to heavy load conditions.


I. INTRODUCTION

The plethora of smart devices has encouraged the industrial and research communities to envision the beauty of predicting and taking precautionary measures in advance to save plenty of resources. In this regard, Internet of Things (IoT) supports the phenomenon of connecting every object on the face of Earth irrespective of its platform, communication technology, etc. [1]. The IoT environment comprises of things that can be your wearables (smart watch, glasses, etc.), vehicles (autonomous cars, smart bikes, etc.), fun time gadgets (cell phones, tablets, gaming devices, smart cameras, etc.), and office or working place machines (laptops, computers, etc.) [2]. IoT laid the foundation of a new era of technology comprising of large pool of applications including smart parking, traffic control, smart cities, connected cars, smart grid and meters, and green houses, etc. [3], [4] as shown in Fig. 1. The surging of aforesaid and miniaturized ubiquitous devices led us to the cloud infrastructure due to unlimited resources and variety of services availability [5]. Cloud computing is provisioning resources by on-demand and self-service models including storage, applications, services, etc., to the users through Internet [6]. It provides outsourcing data facility to end users and releases them from maintaining their own infrastructures. Cloud is not only serving the humanity (e.g. Dell secure healthcare cloud) but also generating a huge revenue (e.g. social networking and Amazon).

However, the prodigal use of cloud resources, high bandwidth requirement, issue of mobility and federated infrastructure result in some drawbacks as well that include high latency, high congestion, huge idle energy consumption [7]. Cloud suffering from core network bandwidth limitation by forwarding deluge of data can be astutely foreseen in the IoT storm[8]. Similarly, though the IoT paradigm provides every time and anywhere connectivity of the objects [9], this connectivity is futile if we cannot use the sensed and gathered data in a timely manner. However, the deluge of data generated cannot be used by the devices on their own. Thus, to realize the future and benefits of IoT infrastructure, we need to have gadgets which can store, process and send back the insights to the devices to act upon the user input with a low latency. Cloud infrastructure is not able to meet the aforementioned demands of emerging compute-intensive applications which led the research community to coin a term of Fog Computing (FC).

FC is a hierarchical distributed infrastructure, introduced to provision cloud-based services in closer proximity to end users. Being located in the middle of cloud and IoT devices, fog supports mobility, distributed architecture, heterogeneity, interoperability, scalability, location awareness and Quality of Service (QoS) retaining low latency for time critical and delay-sensitive applications like tele-health, smart transportation, industrial control and online gaming [10], [11]. Fog nodes can be our routers or specifically designed devices as shown in Fig. 1. These are capable of storing, processing and communicating data with cloud, IoT and other fog nodes. However, these resources are available in limited capacity and on the arrival of proliferated time-sensitive requests, fog nodes have to divert...
Similarly, Volunteer Computing (VC) is another distributed computing model like fog that harvests the idle resources of inter-connected storage- and compute-excessive devices to execute compute-intensive projects [13]. People, who want to volunteer their resources, download the task file from the server and start executing on their respective devices, e.g., cell phones, laptops, desktop computers, etc. Recent studies show that the explosive growth of aforementioned devices can provide more resources than a centralized computing system [14]. In order to develop and surge the public interest towards resource sharing, Shahri et al. [15] proposed the idea of gamification while Beraldi et al. [16] proposed virtual coins based incentive mechanism. VC provides low-price, reliable and scalable platform in which a middleware device divides an extensive job into granular chunks for parallel execution. The utmost challenge faced by the VC is heterogeneity among available volunteers which requires efficient distribution of tasks among them considering their capabilities.

This paper proposes a new computing paradigm called Volunteer Supported Fog Computing (VSFC) that is a hybrid of FC-Cloud and VC paradigms. The core concept of VSFC is to leverage the underutilized resources of end devices in the vicinity of a Fog node to address the issues of high latency, energy consumption and network usage faced by the cloud infrastructure. On the contrary to conventional FC-Cloud computing paradigm, once the Fog nodes run out of resources for delay-sensitive applications, the tasks are migrated to volunteer devices in the vicinity to alleviate the issue of high latency and conserve energy at system-level. The main contributions of our work are listed below:

- We propose VSFC, a new computing paradigm for IoT applications that combines FC and VC to enrich the computing capabilities at the edge of the network.
- We extend the well-known iFogSim toolkit [17] by incorporating a VC layer to enable VSFC. This layer provisions the volunteers to participate when the fog device is exhausted. Instead of sending delay-sensitive jobs to the federated cloud, fog device efficiently executes it over the nearby available volunteer devices.
- We perform extensive simulations to provide a comparative analysis of FC-Cloud and the VSFC paradigms. Moreover, we also investigate the tradeoffs of shifting delay-sensitive jobs from baseline FC-Cloud to VSFC.

The remainder of the article is structured as follows. In Section II, we present the literature review of FC and VC domains. In Section III, we detail our proposed idea of VSFC along with its application in a delay-sensitive application scenario. Section IV contains the simulation setup and details of considered performance evaluation metrics. Simulation results are discussed in Section V and finally the paper is concluded in Section VI with some future research directions.

II. RELATED WORK

In this Section, we cast some light on the research done in FC and VC paradigms. We start with the benefits and challenges of FC and then follow it up with the same in VC. We then conclude this section with a discussion to motivate the need of a hybrid approach that is being proposed in this article.

A. Fog Computing (FC)

A fog supported smart city architecture naming FOCAN is proposed in [18]. It is multi-tier energy, latency and communication efficient architecture in which applications and tasks are offloaded to distributed fog nodes. FOCAN was simulated over iFogSim with web traffic and I/O traces. The significant results opened gates for new dimensions for smart city projects.

Rafique et al. [19] proposed a hybrid optimization scheduling algorithm. It schedules the tasks to optimal devices and balances the load among fog devices. Simulation results show significant improvements in execution time, latency and energy consumption. For efficient module mapping in FC-Cloud architecture, Mahmud et al. proposed fuzzy logic models [20] while Taneja et al. leveraged a generic scheduling policy by considering RAM, CPU along with bandwidth [21]. Both improved Quality of Experience (QoE) and showed staggering decrease in latency, energy and network usage. However, proliferated requests degrade their respective performances due to execution at cloud.

Similarly, Toor et al. addressed energy consumption issue to ensure minimum QoS by varying CPU frequency for achieving energy efficiency [22]. The varying operating speed shows improvement in energy consumption and latency when compared with policy operating at constant higher CPU speed.

Mobile Edge Computing (MEC) is another promising technology paradigm like FC where mobile users offload task to an edge server or cellular base station for execution [23]. Authors in [24] proposed a mixed integer non-Linear programming model in MEC for optimal task offloading within statistical guaranteed QoS. However, our proposed work follows and extends the FC-Cloud paradigm and not only utilizes the resources of the central fog device but also explores the volunteer devices or the federated cloud to meet the QoS.

B. Volunteer Computing (VC)

In order to increase volunteer resource pool, Funai et al. [14] proposed an ad hoc networking based model. Any
device having internet access elects itself as a task distribution point and invites other devices to participate in computation through device-to-device communication using either WiFi Direct or Bluetooth. The simulations resulted in an increased number of tasks executed with the minimum possible energy consumption.

Authors in [25] leveraged the surging IoT compute-intensive devices and enlarged the resource pool by appending such volunteers in it with the goal of executing data near the user. This model uses volunteer cloud concept at the edge of network providing cloud-like mini data centres as processing units in the middle of the end user and traditional cloud.

Panadero et al. [26] proposed Multi Criteria Biased Randomized (MCBR) technique to solve the problem of unreliable participants in VC. The scheme proposed in MCBR ensures the selection of most suitable VN for computation offloading in fast and efficient manner by iterating the unreliable VNs.

C. Discussion

To alleviate the inherent unacceptable high delay of federated cloud infrastructure, FC has recently been explored as a complementary solution to enhance the performance of delay-sensitive tasks. Fog was guaranteeing the minimum possible latency to delay-sensitive applications being located in the closer proximity to the end users. In this essence, a Delay-Priority scheme was proposed in [12] that prioritized delay-sensitive over delay-tolerant applications at FC layer. However, in FC-Cloud paradigm, all the users are directed to cloud in the arrival of proliferated time-sensitive jobs due to limited computational capacity of fog devices as shown in Fig. 2. The higher delay of centralized cloud mainly comes from queuing, propagation and transmission. The queuing delay is directly proportional to the number of jobs, while the typical one-way propagation delay between fog and cloud is about 100ms [12] as shown in Fig. 2.

The aforementioned problems led us to revisit the computing model and to use the underutilized resources available at IoT layer in the vicinity of a fog node by leveraging 2ms [12] delay for task offloading to volunteer devices. This paved our path to propose a novel computing paradigm for delay-sensitive applications that is discussed in detail in the next section.

III. VOLUNTEER SUPPORTED FOG COMPUTING (VSFC)

In this section, we discuss system architecture and application scenario of our proposed VSFC paradigm.

A. System Architecture

The goal of our proposed approach is to meet the stringent-delay requirement by leveraging the underutilized resources of volunteer devices located in the closer proximity. This idea led us to revisit the traditional computing model of FC-Cloud due to lack of support for volunteers. Fog consists of switches, routers, embedded servers and a smart device responsible for decision making for incoming requests. For a device to be a fog, it needs to have characteristics of processing, storage and networking that are available in limited amount as shown in Fig. 2. Fog contains the following two types of modules, (i) the fog devices and (ii) the IoT devices in the vicinity. When a new IoT device connects to fog, the smart (fog) device asks for the permission of using the resources of IoT device in a voluntary role. The IoT device can decide to become a volunteer or not, and the fog device stores this information for future decision making. Both modules are updated when a new IoT device arrives or leaves the vicinity. The characteristics of the fog module are not updated frequently because of it being static in nature. On the contrary, the mobility of IoT devices results in frequent updates in IoT device module.

The IoT devices include laptops, computers, cameras, smart watches, etc. as well as small energy and power-constrained sensors and actuators. These devices generate the data that needs to be processed on delay-priority or delay-tolerant basis. According to the proposed approach, this data can be processed at a fog node, cloud or a VC device (depending on the job requirement and device computing power) as per decision of the subjected fog device which is also depicted in sequence of operations given in Fig. 2. When the execution is completed, the insights are sent to the fog which then forwards them to the source IoT device. In the proposed VSFC architecture, fog devices serve as an intermediate layer between the cloud and VC. Our goal is to efficiently utilize the resources of voluntary IoT devices having excessive underutilized processing power to achieve a resource-efficient computing paradigm at the edge of the network.

B. Delay-Sensitive Application Scenario

In order to illustrate the working of VSFC, we simulated Electroencephalography Tractor Beam Game (EEGTBG) – a delay-sensitive IoT application [17] – that is a multiplayer game helping the users to increase their level of concentration requiring stringent latency restriction. All the players can see the current status of game on their mobile screens. Every player has to concentrate at an item, initially, placed at the center of the screen. The item then starts to move towards the player with highest level of concentration sensed through the sensor placed over their heads. This complete application loop is delay-sensitive and needs to be executed with stringent time bounds to maintain the fairness among competing users. The game has five modules naming EEG sensor, client, display, concentration calculator and coordinator. The EEGTBG sensor is deployed over the head of the player connected to cell phone.
via Bluetooth. Client module receives raw sensed data from sensor and directs it towards the concentration calculator to measure the current concentration level of player. The insights are forwarded to client module to update the display of the game. While, the coordinator module is used on low level to update the status of all the players who might be present on distributed locations. Client and display modules are placed on the mobile device of individual player, while concentration calculator and coordinator modules can be placed on fog, volunteer device or cloud.

1) User Arrival and Module Placement: In this section, we explain the process of new user arrival into the game. It is started with the assumption that the game is running at fog device. With the arrival of every new user, its concentration calculator and coordinator modules are placed at a suitable device. Under conventional FC-Cloud scenario, the existing fog architecture shifts all the modules of the users to the cloud when the cumulative computing requirement of the modules exceeds the fog node capacity, hence, resulting in high latency. The capacity of different computing modules, such as fog, cloud, volunteer IoT device, is measured in available and demanded Millions of Instructions Per Second (MIPS).

Similarly in VSFC, we attempt to accommodate arriving users on fog to make them play their game with minimum possible delay. However, when the fog device is fully utilized, VSFC looks for any available volunteer device in the vicinity. If there exists a volunteer device that can accommodate the arrived module, the latter is placed on this volunteer and its data is updated with the demanded processing power (in MIPS) of currently placed modules. Table I shows the peak MIPS requirements of each module used in EEGTBG game. If there is no volunteer device available or the available volunteer is saturated then the fog device directs these new modules towards the cloud. Algorithm 1 shows the overall working of our proposed VSFC scheme where, $D$, $M^f_i$, $M^f_{max}$, $\theta$, $M^c_i$, $M^c_{max}$ and $M^c_f$ represent arriving IoT device, old MIPS of fog, maximum MIPS limit of fog, MIPS required by the module being placed, old MIPS of VC device, maximum MIPS limit of VC and old MIPS of cloud, respectively.

2) Data Processing: After the successful deployment of application modules, the traffic needs to be diverted towards them. In this regard, on the arrival of every IoT job, the fog device checks its requirements. If it is time-critical and the desired module for execution exists at fog, it is prioritized and dealt at fog; otherwise the fog device explores connected VC devices for it. Similarly, if the processing module is found on a VC device, the task is directed toward it. The results are returned through fog device back to the source IoT device. In the worst case scenario, if there is no VC device available or all of them are fully occupied according to their capacities, then the tasks will be placed at the cloud.

To avoid degraded system performance, we decided to avoid using miniaturized and energy-constrained IoT devices as volunteers. Such devices offer very low processing power along with limited battery that results in an increase in job failures that, in turn, impacts the performance of delay-sensitive jobs. Hence, only plugged-in devices such as laptops, tablets and desktop computers are included in the VC layer of the proposed architecture.

IV. SIMULATION SETUP AND PERFORMANCE METRICS

In this section, we provide the detailed description of our simulation setup, followed by the metrics to compare the performance of VSFC against the conventional FC-Cloud computing paradigm.

A. Simulation Setup

We evaluated the proposed architecture using iFogSim [17] that is constructed over CloudSim simulator. iFogSim is widely used simulation environment to simulate edge computing, IoT, fog and cloud devices. We simulated delay-sensitive application, EEGTBG, with gradually increasing the number of users to compare the results of the proposed scheme against the Delay-Priority scheme [12]. Every fog has one dedicated volunteer device connected to it and initially, both Delay-Priority and VSFC schemes execute tasks on the fog. Subsequently, once the fog reaches its maximum limit, Delay-Priority shifts all the users to cloud while VSFC shifts them to the available volunteer device. We are assuming the VC device to be a laptop or desktop computer which is plugged-in and does not have energy constraint.

We setup two application scenarios: (i) in One-Fog scenario, there is only one fog and a connected VC device while (ii) in Two-Fog scenario, two fog nodes are connected to two VC devices resulting in traffic distribution between them under VSFC. Delay-Priority has to deal with the traffic coming from

---

**Algorithm 1 The working of VSFC**

1: procedure DEVICE_ARRIVAL
2: for each arriving device $D_i$ do
3: if $D_i$ agrees to be a volunteer then
4: $VC \leftarrow D_i$
5: end if
6: end for
7: end procedure
8: procedure MODULE_PLACEMENT
9: for $i = 1, i++, \text{while } i < N$ do
10: if $M^f_i + \theta \leq M^f_{max}$ then
11: Place on Fog
12: $M^f_i = M^f_i + \theta$
13: else if $M^c_i + \theta \leq M^c_{max}$ then
14: Place on VC
15: $M^c_i = M^c_i + \theta$
16: else
17: Place on cloud
18: $M^c_f = M^c_f + \theta$
19: end if
20: end for
21: end procedure
22: procedure TASK_ARRIVAL
23: for $t = 1, t++, \text{while } t < T$ do
24: if $m \in fog$ then
25: Execute on Fog
26: else if $m \in VC$ then
27: Execute on VC
28: else
29: Execute on Cloud
30: end if
31: end for
32: end procedure

**TABLE I**

<table>
<thead>
<tr>
<th>Module Name</th>
<th>CPU (MIPS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>400</td>
</tr>
<tr>
<td>Concentration Calculator</td>
<td>350</td>
</tr>
<tr>
<td>Coordinator</td>
<td>100</td>
</tr>
</tbody>
</table>

The capacity of different computing modules, such as fog, cloud, volunteer IoT device, is measured in available and demanded Millions of Instructions Per Second (MIPS).
both distributed fog nodes at the centralized cloud. We assume the number of simultaneous users to be same in both the fogs for the sake of simplicity but this does not limit our scheme to assume otherwise. Both scenarios are shown in Fig. 3.

Each fog has processing power of 4000 MIPS and for a fair comparison, we assumed the same for VC devices. Fog is connected to cloud with 10 Mbps bandwidth and posing a delay of 100 ms. While, the link between fog and IoT layer devices poses delay of 2 ms over the same 10 Mbps link. Table II shows the overall configurations of devices used in our simulations that are taken from the base paper on iFogSim [17]. B and I in the power column of Table II represent busy and idle status of the device, respectively.

B. Performance Metrics

We are comparing the performance of VSFC against the earlier proposed Delay-Priority scheme using three resources including overall application execution delay, energy consumption and network usage. In the following discussion, we provide the mathematical models that were used for these three performance metrics.

1) Delay: The delay of EEGTBG application comes from the processing of application loop among its modules and it is calculated when the loop completes. EEGTBG loop is explained in Section III-B. As explained in Section II-C, delay mainly comes from queuing, propagation and transmission, and it can be modeled as,

\[ D_{\text{total}} = D_q + D_p + D_t \]  

where \( D_{\text{total}} \), \( D_q \), \( D_p \), and \( D_t \) represent total, queuing, propagation and transmission delays, respectively. In EEGTBG game, the data is transferred among modules that are placed on different nodes, therefore, the EEGTBG processing loop faces all of the delays mentioned in Eq. (1) at every module of game.

2) Energy Calculation: We can calculate the total energy consumed by a computing device executing the application loop using,

\[ E_{\text{total}} = \sum_{i=1}^{n} E_{c_i} \]  

where \( E_{\text{total}} \) is the total energy which can be computed from the sum of \( E_{c_i} \) by executing \( n \) number of tasks. \( E_{c_i} \) shows the energy consumption on \( i^{th} \) task execution and can itself be calculated as,

\[ E_{c_i} = P(u_i) \times \Delta t_i \]

where \( P(u_i) \) is the power consumption depending on the utilization \( u_i \) of device for \( i^{th} \) task in the \( \Delta t \) time interval. The device utilization, \( u \), is the percentage of total compute resources a device is consuming while executing a certain task. Hence, its value is \( 0 < u < 1 \). This factor scales a device power consumption between busy (\( u = 1 \)) and idle (\( u = 0 \)) states given in Table II.

3) Network Usage: Data sharing among modules over the network results in network usage. When a module transmits data to another module, the link between them gets busy till the data successfully arrives at the destination module which can be modeled as,

\[ N_u = \sum_{i=1}^{n} L_i \times \Delta t'_i \]

where \( N_u \) is the total network usage, \( L_i \) is the data size and \( \Delta t'_i \) is the time for which the link was busy for the \( i^{th} \) task.

V. EXPERIMENTAL RESULTS

To have better insights of VSFC and Delay-Priority schemes, we executed the following two different sets of simulations by changing the load over VC and cloud:

1) Fixed simulation time: in this scenario, we fixed the total time for which the EEGTBG game was run on our extended iFogSim simulator.

2) Fixed number of tasks completed: in this scenario, we varied total game time and fixed the total number of tasks executed over both VC and cloud.

A. Fixed Simulation Time

In this set of simulations, we fixed the total time of simulations i.e., 1.5 hours for both VSFC and Delay-Priority and compared their performances in terms of aforesaid metrics.

1) Delay: Fig. 4 (a) shows the EEGTBG loop delay for VSFC and Delay-Priority for One-Fog. Initially, both of them execute their data at fog resulting in a similar delay till the number of users reaches 11 where the fog device can no more accommodate the users. Subsequently, VSFC shifts the users to VC while Delay-Priority shifts them to cloud which results in a consistent increase in their respective delays. We can see the delay of VSFC is higher than that of Delay-Priority for each increasing user and Delay-Priority is outperforming our
proposed VSFC scheme. This behavior is quite justified since the cloud has a very high processing power than the VC device, therefore it executes the tasks much faster and reduces the overall loop delay. The delay of Delay-Priority scheme starts increasing from User-15 onward due to bandwidth limitation of the core network. As the number of users increase, the amount of data directed to the cloud also increases, ultimately, utilizing the maximum bandwidth available. Although cloud has excessive processing power, Delay-Priority suffers from transmission delay caused by bandwidth limitation.

Fig. 4 (b) shows the delay with two EEGTBG games running on two different fog devices. It can be noticed that the performance of VSFC is better than the Delay-Priority at each user. The main reason is the distributed work load over VC devices where the data generated from each fog is dealt at the respective VC device due to which VSFC is performing better than One-Fog scenario. The behavior of both VC devices in Fig. 4 (b) is similar to VC device in Fig. 4 (a) due to handling the same amount of workload. On the other hand, Delay-Priority suffers from heavy load because both the fogs are directing the data to centralized cloud. This results in increasing the overall loop delay due to queuing at cloud. In addition, the bandwidth limitation is also hurting the performance of Delay-Priority scheme. Hence, it can be concluded that our proposed VSFC scheme outperforms the traditional FC-Cloud scheme under normal to heavy load conditions that is quite expected with the deluge of data expected to be generated under IoT storm.

2) Energy Consumption: The energy consumption of VSFC and Delay-Priority schemes for One-Fog and Two-Fog is given in Fig. 5. It can be observed that from User-1 to User-11, the energy consumption is resulted from the fog devices in both VSFC and Delay-Priority; hence, it is identical (see the curves for VSFC-Fog-[*] and Delay-Priority-Fog-[*]). As the number of users increases, the utilization of fog device also increases resulting in the rise of energy curve. However, when the number of users reaches 11, the utilization of the fog devices in both VSFC and Delay-Priority reaches its peak and the processing is shifted from fog to cloud in Delay-Priority or VC in VSFC. At this point, fog energy of both schemes reduces to the static or idle energy. Consequently, the energy of VC devices rises and since the utilization, $u$, for VC devices is maximum, the energy consumption is capped throughout the simulation time.

Similarly, Fig. 6 (a) and (b) depict the energies of fog, VC and cloud for One-Fog and Two-Fog scenarios, respectively. It can be noted again that after User-11, all the users are shifted to either cloud (in Delay-Priority) or VC (in VSFC). Consequently, the fog energy reduces to its idle energy while cloud and VC’s energy consumption is affected. The cloud energy in VSFC remains constant at idle because cloud is inactive and VSFC leverages the available VC, while the cloud energy is linearly increased in Delay-Priority scheme due to placement of users over cloud. By comparing the scale of cumulative energy consumption in the graphs, we can conclude that the VSFC is outperforming Delay-Priority by a huge margin due to the inherent higher idle energy consumption of cloud while providing higher compute power.

3) Network Usage: Increasing the number of users also increases the network load and results in high network usage for Delay-Priority scheme. As the data is sent to the cloud, it makes the link busy for longer time period. Cloud being located at propagation delay of 100 ms increases network usage as given by the Eq. (4). The behavior of Delay-Priority is similar for both One-Fog and Two-Fog scenarios. However, if we look closely at Fig. 7, we can observe that the slope of network usage is steeper for Two-Fog curve than that of One-Fog due to higher load. This spike also depicts bandwidth limitation which causes the delay of Delay-Priority scheme in Two-Fog scenario to sharply increase as mentioned in the previous section. On the contrary, the network usage of VSFC is quite low as the delay of Fog-VC link is only around 2 ms. Therefore, the network gets busy for very little time when compared to FC-Cloud communication.
Fig. 7. Network Usage for VSFC and Delay-Priority Schemes under One-Fog and Two-Fog Scenarios.

Fig. 8. Energy Consumption of VC and Fog for One-Fog and Two-Fog Scenarios under Varied Time.

B. Fixed Number of Tasks Completed

In the previous set of simulations, we can observe that the energy consumption of VC device in VSFC scheme (VSFC-VC[*] in Fig. 5) was not changing even with the increase in the workload as the VC device utilization, \( u \), was already at 1. This maximum utilization puts the tasks in a queue when the fixed simulation time ends and results in a reduced number of completed jobs.

This inspired us to design a different set of simulations where the simulation time was varied to capture the impact of increase in VC energy under VSFC scheme. However, since VC and cloud have different processing power, we had to fix the total number of tasks completed, in order to have a fair comparison between VSFC and Delay-Priority schemes. For this, we first executed One-Fog scenario for 1.5 hours and recorded the number of tasks completed, in order to have a fair comparison between VSFC and Delay-Priority schemes. For this, we first executed One-Fog scenario for 1.5 hours and recorded the number of tasks completed under Delay-Priority scheme. We then varied the simulation time for VSFC scheme to achieve the same number of tasks completed and compared the energy results with Delay-Priority scheme for a fair comparison. The same was repeated for Two-Fog scenario and the results are shown in Fig. 8.

We can notice that VC energy is increasing with the increase in the number of users because it has to process a lot more data than the previous set of simulations. The VC device is still maximally utilized here, i.e., in Eq. (3), \( P(\mu) \) is at max but in the current set of simulations, \( \Delta t_i \) is increased that results in an increasing value for energy consumption. Fig. 9 (a) and (b) depict the energies of fog, VC and cloud for One-Fog and Two-Fog scenarios under fixed number of tasks completed. The results are even better than the previous set of simulations of fixed simulation time and the VSFC scheme is further outperforming the baseline Delay-Priority scheme in the overall system-level energy consumption under Two-Fog scenario.

Similarly, the delay and network usage results for fixed number of task completed scenario were consistent with the fixed simulation time scenario but those can not be included due to space limitation.

To summarize, under normal to heavy load conditions, VSFC provides a saving up to 47.5% and 85.1% under One-Fog and Two-Fog scenarios over baseline Delay-Priority scheme for communication delay. Likewise, VSFC provides 93% and 86% energy savings compared to traditional FC-Cloud architecture under One-Fog and Two-Fog scenarios, respectively. Finally, VSFC also outperforms the baseline scheme in the network usage by decreasing it by 92% under both One-Fog and Two-Fog schemes.

VI. CONCLUSIONS AND FUTURE WORK

With the emergence of delay-sensitive IoT applications, there is a dire need to deploy more computing power at the edge of the network. In this context, we have proposed a novel computing paradigm, VSFC, that integrates volunteer computing (VC) and fog computing (FC) for efficient utilization of the underutilized IoT resources available in the vicinity of fog devices. This results in energy savings by utilizing low-power VC devices instead of high-power cloud machines. It also enables bandwidth optimizations by avoiding the core network via handling majority of the traffic at the edge. Results show that the proposed VSFC scheme reduces the delay, energy consumption and network usage by 47.5%, 93%, and 92%, respectively when compared with traditional FC-Cloud architecture. In the future, we aim to propose and develop efficient scheduling policies for load-balancing among available resources at the edge of the network. Moreover, the adaptive policies can also be considered for optimizing the QoS. We also plan to extend the VSFC to include volunteer devices with more realistic characteristics, i.e., device heterogeneity, mobility, battery life, etc.

REFERENCES
