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 FC (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 the 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.**

***Index Terms*—Cloud computing, distributed computing paradigms, fog computing (FC), Internet of Things (IoT), resource management, volunteer computing (VC).**

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, the 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 (smartwatch, 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 a large pool of applications, including smart parking, traffic control, smart cities, connected cars, smart grid and meters, greenhouses, 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 a 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 the Internet [6]. It provides outsourcing data facility to end users and releases them from maintaining their own infrastructures. Cloud is not only serving 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, and huge idle energy consumption [7]. Cloud is suffering from core network bandwidth limitation by forwarding the deluge of data that 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 the IoT infrastructure, we need to have devices that can store, process, and send back the insights to the devices to act upon the user input with 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 provide cloud-based services in closer proximity to end users. Being located in the middle of the 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, such as telehealth, 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

Manuscript received April 16, 2020; revised July 24, 2020; accepted September 10, 2020. Date of publication September 21, 2020; date of current version February 19, 2021. *(Corresponding authors: Muhammad Adeel Pasha; Saif ul Islam.)*

Babar Ali is with the Department of Computer Science, SBA School of Science and Engineering, Lahore University of Management Sciences, Lahore 54792, Pakistan (e-mail: 17030033@lums.edu.pk).

Muhammad Adeel Pasha is with the Department of Electrical Engineering, SBA School of Science and Engineering, Lahore University of Management Sciences, Lahore 54792, Pakistan (e-mail: adeel.pasha@lums.edu.pk).

Saif ul Islam is with the Department of Computer Science, Institute of Space Technology, Islamabad 44000, Pakistan (e-mail: saiflu2004@gmail.com).

Houbing Song is with the Department of Electrical Engineering and Computer Science, Embry-Riddle Aeronautical University, Daytona Beach, FL 32114 USA (e-mail: h.song@ieee.org).

Rajkumar Buyya is with the Cloud Computing and Distributed Systems Laboratory, School of Computing and Information System, University of Melbourne, Parkville, VIC 3010, Australia (e-mail: rbuyya@unimelb.edu.au).

Digital Object Identifier 10.1109/JIOT.2020.3024823

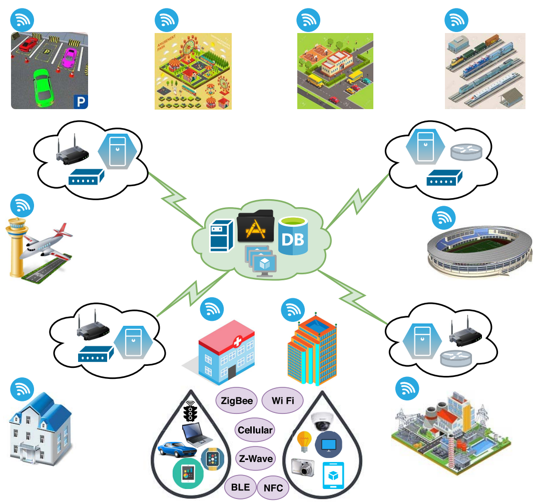


Fig. 1. Generic diagram of fog infrastructure and related applications.

communicating data with the cloud, IoT, and other fog nodes. However, these resources are available in a limited capacity, and on the arrival of proliferated time-sensitive requests, fog nodes have to divert them to the cloud for processing that results in an unacceptably high latency [12].

Similarly, volunteer computing (VC) is another distributed computing model like the fog that harvests the idle resources of interconnected storage-excessive 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 the aforementioned devices can provide more resources than a centralized computing system [14]. In order to develop and surge the public interest toward resource sharing, Shahri *et al.* [15] proposed the idea of gamification while Beraldi *et al.* [16] proposed a virtual coins-based incentive mechanism. VC provides a 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 article proposes a new computing paradigm called volunteer-supported FC (VSFC), which 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 the 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 the system level. The main contributions of our work are listed as follows.

1) 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.
2) 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, the fog device efficiently executes it over the nearby available volunteer devices.
3) 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 the 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 this article 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 for a hybrid approach that is being proposed in this article.

A. Fog Computing

A fog-supported smart city architecture naming FOCAN is proposed in [18]. It is multitier 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 the FC-Cloud architecture, Mahmud *et al.* proposed fuzzy logic models [20] while Taneja and Davy [21] leveraged a generic scheduling policy by considering RAM, CPU along with bandwidth. Both improved Quality of Experience (QoE) and showed a staggering decrease in latency, energy, and network usage. However, proliferated requests degrade their respective performances due to execution at the cloud. Similarly, Toor *et al.* [22] addressed the energy consumption issues to ensure minimum QoS by varying CPU frequency for achieving energy efficiency. The varying operating speed shows the improvement in energy consumption and latency when compared with the policy operating at constant higher CPU speed.

Mobile edge computing (MEC) is another promising technology paradigm like FC where mobile users offload the task to an edge server or cellular base station for execution [23]. Li *et al.* [24] proposed a mixed-integer nonLinear programming model in MEC for optimal task offloading within statistical guaranteed QoS. However, our

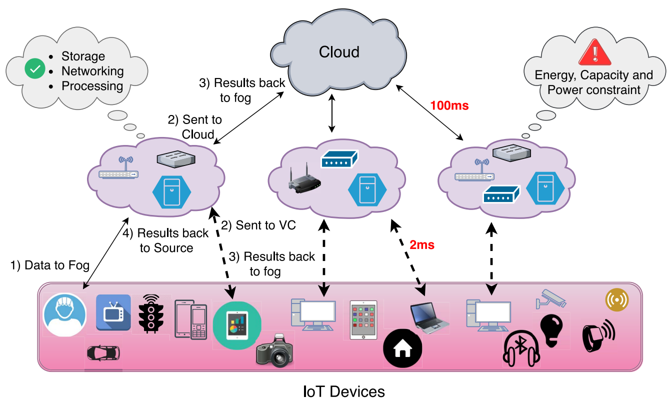


Fig. 2. Generic architecture of our proposed VSFC scheme.

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

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. Mengistu *et al.* [25] leveraged the surging IoT compute-excessive devices and enlarged the resource pool by appending such volunteers in it with the goal of executing data near the user. This model uses a volunteer cloud concept at the edge of the network providing cloud-like mini data centers as processing units in the middle of the end user and traditional cloud. Panadero *et al.* [26] proposed a multicriteria-biased randomized (MCBR) technique to solve the problem of unreliable participants in VC. The scheme proposed in MCBR ensures the selection of the most suitable volunteer node (VN) for computation offloading in a fast and efficient manner by iterating the unreliable VNs.

C. Discussion

To alleviate the inherent unacceptably high delay of federated cloud infrastructure, FC has recently been explored as a complementary solution to enhance the performance of delay-sensitive tasks. The 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 the FC layer. However, in the FC-Cloud paradigm, all the users are directed to the cloud in the arrival of proliferated time-sensitive jobs due to the limited computational capacity of fog devices as shown in Fig. 2. The higher delay of the 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 the fog and cloud is about 100 ms [12] as shown in Fig. 2.

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

III. Volunteer-Supported Fog Computing

In this section, we discuss the 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 closer proximity. This idea led us to revisit the traditional computing model of the FC-Cloud due to a 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 a limited amount as shown in Fig. 2. Fog contains the following two types of modules: 1) the fog devices and 2) the IoT devices in the vicinity. When a new IoT device connects to fog, the smart (fog) device asks for permission of using the resources of the IoT devices 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 the IoT device module.

The IoT devices include laptops, computers, cameras, smartwatches, etc., as well as small energy and power-constrained sensors and actuators. These devices generate data that need to be processed on a delay-priority or delay-tolerant basis. According to the proposed approach, these data can be processed at a fog node, cloud, or a VC device (depending on the job requirement and device computing power) as per the decision of the subjected fog device, which is also depicted in the 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 the 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 the game on their mobile screens. Every player has to concentrate on an item, initially, placed at the center of the screen. The item then starts to move toward the player with the 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 fairness among competing users. The game has five modules, namely, EEG sensor, client, display, concentration calculator, and coordinator. The EEGTBG sensor is deployed over the head of the player connected to the cell phone via Bluetooth. The client module receives raw sensed data from the sensor and directs it toward the concentration calculator to measure the current concentration level of the player. The insights are forwarded to the client module to update the display of the game. While the coordinator module is used on a global 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 an individual player, while the concentration calculator and coordinator modules can be placed on the 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 the fog device. With the arrival of every new user, its concentration calculator and coordinator modules are placed at a suitable device. Under the 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 in the fog to make them play their game with a 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 are updated with the demanded processing power (in MIPS) of currently placed modules. Table I shows the peak MIPS requirements of each module used in the EEGTBG game. If there is no volunteer device available or the available volunteer is saturated, then the fog device directs these new modules toward the cloud. Algorithm 1 shows the overall working of our proposed VSFC scheme where D, M_O^f, $M_{\max}^f$, θ, M_O^{vc}, $M_{\max}^{vc}$, and M_O^C represent the 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 toward 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 the fog, it is prioritized and dealt with 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 the 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 a 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.

Algorithm 1 Working of VSFC

procedure DEVICE –ARRIVAL
  for each arriving device D_i **do**
    if D_i agrees to be a volunteer **then**
      $VC \leftarrow D_i$
    end if
  end for
end procedure
procedure MODULE –PLACEMENT
  for $i = 1$, i++, while $i < N$ **do**
    if $M_O^f + \theta \leq M_{max}^f$ **then**
      Place on Fog
      $M_O^f = M_O^f + \theta$
    else if $M_O^{vc} + \theta \leq M_{max}^{vc}$ **then**
      Place on VC
      $M_O^{vc} = M_O^{vc} + \theta$
    else
      Place on cloud
      $M_O^C = M_O^C + \theta$
    end if
  end for
end procedure
procedure TASK –ARRIVAL
  for $t = 1$, t++, while $t < T$ **do**
    if $m \in fog$ **then**
      Execute on Fog
    else if $m \in VC$ **then**
      Execute on VC
    else
      Execute on Cloud
    end if
  end for
end procedure

TABLE I
CPU REQUIREMENTS OF EEGTBG APPLICATION MODULES

Module Name	CPU (MIPS)
Client	200
Concentration Calculator	350
Coordinator	100

IV. SIMULATION SETUP AND PERFORMANCE METRICS

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

A. Simulation Setup

We evaluated the proposed architecture using iFogSim [17] that is constructed over the ClouldSim simulator. iFogSim is a widely used simulation environment to simulate edge computing, IoT, fog, and cloud devices. We simulated a 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

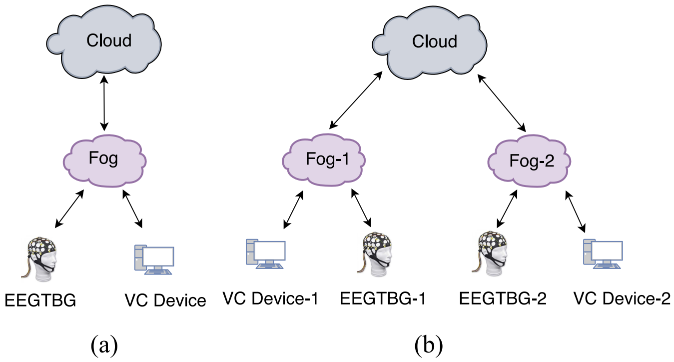


Fig. 3. Application scenarios used in the simulation setup. (a) One-Fog. (b) Two-Fog.

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 the 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 an energy constraint. We set up two application scenarios: 1) in the *One-Fog* scenario, there is only one fog and a connected VC device and while 2) in the *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 both distributed fog nodes at the centralized cloud. We assume the number of simultaneous users to be the same in both scenarios 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 a processing power of 4000 MIPS and for a fair comparison, we assume the same for VC devices. Fog is connected to the cloud with 10-Mb/s bandwidth and posing a delay of 100 ms. While the link between the fog and IoT-layer devices poses a delay of 2 ms over the same 10-Mb/s 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 the busy and idle status of a device, respectively.

B. Performance Metrics

We are comparing the performance of VSFC against the earlier proposed delay-priority scheme using three resources, including the 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 the EEGTBG application comes from the processing of the application loop among its modules and it is calculated when the loop is completed. 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

$$D_{\text{total}} = D_q + D_p + D_t \tag{1}$$

TABLE II
DEVICE CONFIGURATIONS

Device	CPU (GHz)	RAM (GB)	Processing (MIPS)	Power (W)
Cloud	3.0	40	44800	1648 (B), 1332 (I)
Fog	3.0	4	4000	107.339 (B), 83.433 (I)
VC Device	3.0	4	4000	107.339 (B), 83.433 (I)
Smartphone	1.6	1	1000	87.53 (B), 82.44 (I)

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

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

$$E_{\text{total}} = \sum_{i=1}^{n} E_{c_i} \tag{2}$$

where E_{total} is the total energy that can be computed from the sum of E_{c_i} by executing n number of tasks. E_{c_i} shows the energy consumption on the ith task execution and can itself be calculated as

$$E_{c_i} = P(u_i) \times \Delta t_i \tag{3}$$

where $P(u_i)$ is the power consumption depending on the utilization u_i of device for the ith task in the Δt 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 until the data successfully arrive at the destination module which can be modeled as

$$N_u = \sum_{i=1}^{n} L_i \times \Delta t_i' \tag{4}$$

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 ith task.

V. EXPERIMENTAL RESULTS

To have better insights into 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 the total game time and fixed the total number of tasks executed over both VC and cloud.

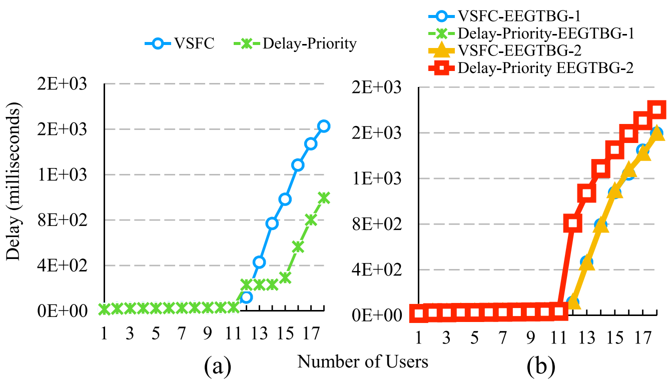


Fig. 4. VSFC and delay-priority delay comparison under fixed simulation time. (a) One-Fog. (b) Two-Fog.

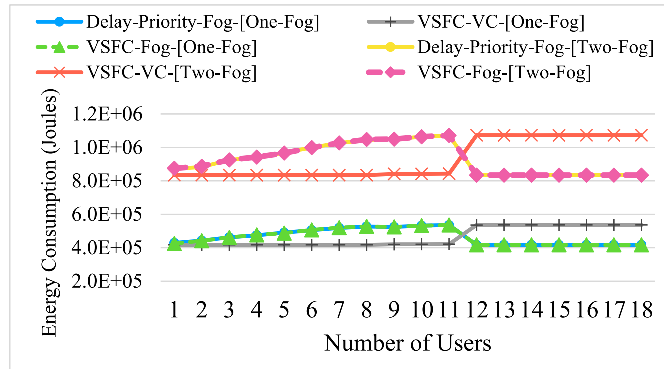


Fig. 5. Energy consumption of VC and fog devices under One-Fog and Two-Fog simulation scenarios.

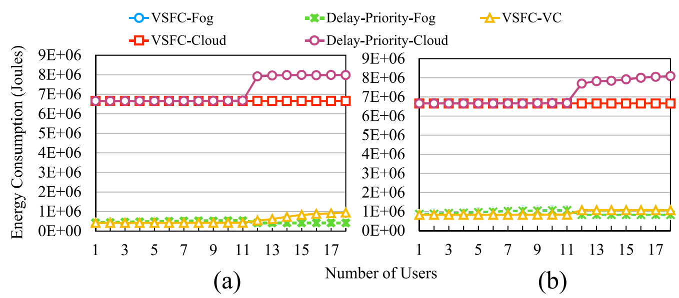


Fig. 6. Energy consumption comparison of VC and cloud under (a) One-Fog and (b) Two-Fog simulation scenarios.

A. Fixed Simulation Time

In this set of simulations, we fixed the total time of simulations, i.e., 1.5 h for both VSFC and delay-priority, and compared their performances in terms of the 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 the fog resulting in a similar delay until 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 the cloud, which results in a consistent increase in their respective delays. We can see the delay of VSFC is higher than that of the 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 the delay-priority scheme starts increasing from user-15 onward due to the bandwidth limitation of the core network. As the number of users increases, the amount of data directed to the cloud also increases, ultimately, utilizing the maximum bandwidth available. Although the 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 workload over VC devices where the data generated from each fog are dealt at the respective VC device due to which VSFC is performing better than the One-Fog scenario. The behavior of both VC devices in Fig. 4(b) is similar to the 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 a centralized cloud. This results in increasing the overall loop delay due to queuing at the cloud. In addition, the bandwidth limitation is also hurting the performance of the 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 are quite expected with the deluge of data expected to be generated under the 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 has 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 a fog device also increases resulting in the rise of the 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 the case of VSFC. At this point, the fog energy of both schemes reduces 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) depicts 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 the cloud is inactive and VSFC leverages the available VC, while the cloud energy is linearly increased in the delay-priority scheme due to the placement of users on the 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 the cloud while providing higher compute power.

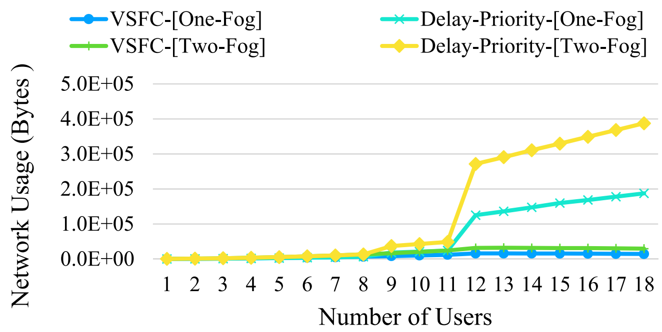


Fig. 7. Network usage for VSFC and delay-priority schemes under One-Fog and Two-Fog scenarios.

3) Network Usage: Increasing the number of users also increases the network load and results in high network usage for the delay-priority scheme. As the data are sent to the cloud, it makes the link busy for a longer time period. Cloud being located at a propagation delay of 100 ms increases network usage as given by the (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 the Two-Fog curve than that of One-Fog due to higher load. This spike also depicts bandwidth limitation, which causes the delay of the delay-priority scheme in the 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 the Fog-VC link is only around 2 ms. Therefore, the network gets busy for very little time when compared to FC-Cloud communication.

B. Fixed Number of Tasks Completed

In the previous set of simulations, we can observe that the energy consumption of the VC device in the 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 an increase in VC energy under the 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 the One-Fog scenario for 1.5 h and recorded the number of tasks completed under the delay-priority scheme. We then varied the simulation time for the VSFC scheme to achieve the same number of tasks completed and compared the energy results with the delay-priority scheme for a fair comparison. The same was repeated for the 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 (3), $P(u_i)$ is at max but in the current set of simulations, Δt_i is increased that results in an increased value for energy consumption.

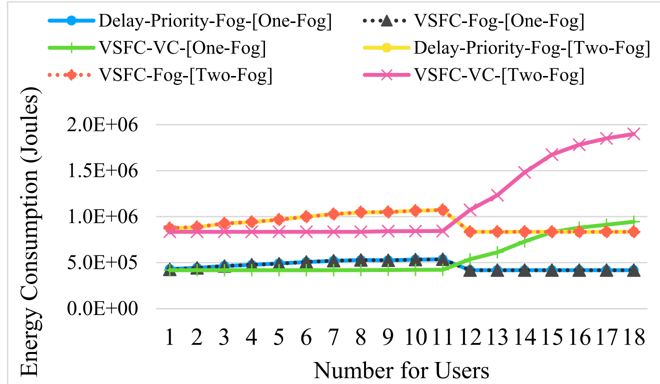


Fig. 8. Energy consumption of VC and fog for One-Fog and Two-Fog scenarios under varied time.

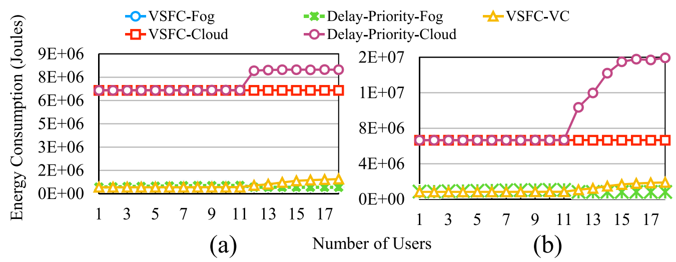


Fig. 9. Energy consumption comparison of VC and cloud under (a) One-Fog and (b) Two-Fog scenarios under varied time.

Fig. 9(a) and (b) depicts the energies of fog, VC, and cloud for One-Fog and Two-Fog scenarios under a 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 the Two-Fog scenario.

Similarly, the delay and network usage results for a fixed number of task completed scenarios were consistent with the fixed simulation time scenario but those cannot 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 the baseline delay-priority scheme for communication delay. Likewise, VSFC provides 93% and 86% energy savings compared to the 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. Conclusion

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 proposed a novel computing paradigm, VSFC, that integrates VC and FC for efficient utilization of the underutilized computing 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 the majority of the traffic at the edge. Results showed that the proposed VSFC scheme reduces the delay, energy consumption, and network usage by 47.5%, 93%, and 92%, respectively, when compared with the 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, and battery life.

REFERENCES

[1] H. Wu *et al.*, "Dynamic edge access system in IoT environment," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2509–2520, Apr. 2020.

[2] S. Andreev, C. Dobre, and P. Misra, "Internet of Things and sensor networks," *IEEE Commun. Mag.*, vol. 58, no. 4, pp. 34–74, Apr. 2020.

[3] J. Granat, J. M. Batalla, C. X. Mavromoustakis, and G. Mastorakis, "Big data analytics for event detection in the IoT-multicriteria approach," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4418–4430, May 2020.

[4] M. Mukherjee, S. Kumar, M. Shojafar, Q. Zhang, and C. X. Mavromoustakis, "Joint task offloading and resource allocation for delay-sensitive fog networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–7.

[5] P. Cai, F. Yang, J. Wang, X. Wu, Y. Yang, and X. Luo, "Jote: Joint offloading of tasks and energy in fog-enabled IoT networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3067–3082, Apr. 2020.

[6] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2018.

[7] J. Ren, D. Zhang, S. He, Y. Zhang, and T. Li, "A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet," *ACM Comput. Surveys*, vol. 52, no. 6, pp. 1–36, Oct. 2019.

[8] Z. Liu, Y. Yang, K. Wang, Z. Shao, and J. Zhang, "Post: Parallel offloading of splittable tasks in heterogeneous fog networks," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3170–3183, Apr. 2020.

[9] X. Huang, Y. Cui, Q. Chen, and J. Zhang, "Joint task offloading and QoS-aware resource allocation in fog-enabled Internet-of-Things networks," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7194–7206, Aug. 2020.

[10] M. Mukherjee *et al.*, "Task data offloading and resource allocation in fog computing with multi-task delay guarantee," *IEEE Access*, vol. 7, pp. 152911–152918, 2019.

[11] M. Mukherjee *et al.*, "Computation offloading strategy in heterogeneous fog computing with energy and delay constraints," in *Proc. IEEE Int. Conf. Commun. (ICC) QoS, Rel. Model. Symp.*, 2020, pp. 1–5.

[12] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, "Mobility-aware application scheduling in fog computing," *IEEE Cloud Comput.*, vol. 4, no. 2, pp. 26–35, Mar./Apr. 2017.

[13] M. N. Durrani and J. A. Shamsi, "Volunteer computing: Requirements, challenges, and solutions," *J. Netw. Comput. Appl.*, vol. 39, pp. 369–380, Mar. 2014.

[14] C. Funai, C. Tapparello, H. Ba, B. Karaoglu, and W. Heinzelman, "Extending volunteer computing through mobile ad hoc networking," in *Proc. IEEE IEEE Global Commun. Conf. (GLOBECOM)*, 2014, pp. 32–38.

[15] A. Shahri, M. Hosseini, R. Ali, and F. Dalpiaz, "Gamification for volunteer cloud computing," in *Proc. 7th IEEE/ACM Int. Conf. Utility Cloud Comput.*, 2014, pp. 616–617.

[16] R. Beraldi, A. Mtibaa, and A. N. Mian, "CICO: A credit-based incentive mechanism for cooperative fog computing paradigms," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2018, pp. 1–7.

[17] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, edge and fog computing environments," *Softw. Pract. Exp.*, vol. 47, no. 9, pp. 1275–1296, 2017.

[18] P. G. V. Naranjo, Z. Pooranian, M. Shojafar, M. Conti, and R. Buyya, "Focan: A fog-supported smart city network architecture for management of applications in the Internet of everything environments," *J. Parallel Distrib. Comput.*, vol. 132, pp. 274–283, Oct. 2019.

[19] H. Rafique, M. A. Shah, S. U. Islam, T. Maqsood, S. Khan, and C. Maple, "A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing," *IEEE Access*, vol. 7, pp. 115760–115773, 2019.

[20] R. Mahmud, S. N. Srirama, K. Ramamohanarao, and R. Buyya, "Quality of experience (QoE)-aware placement of applications in fog computing environments," *J. Parallel Distrib. Comput.*, vol. 132, pp. 190–203, Oct. 2019.

[21] M. Taneja and A. Davy, "Resource aware placement of IoT application modules in fog-cloud computing paradigm," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manag.*, 2017, pp. 1222–1228.

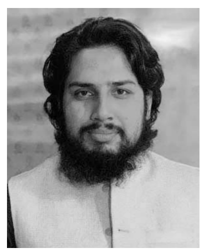
[22] A. Toor *et al.*, "Energy and performance aware fog computing: A case of dvfs and green renewable energy," *Future Gener. Comput. Syst.*, vol. 101, pp. 1112–1121, Dec. 2019.

[23] C. X. Mavromoustakis, G. Mastorakis, and J. Mongay Batalla, "A mobile edge computing model enabling efficient computation offload-aware energy conservation," *IEEE Access*, vol. 7, pp. 102295–102303, 2019.

[24] Q. Li, S. Wang, A. Zhou, X. Ma, F. Yang, and A. X. Liu, "QoS driven task offloading with statistical guarantee in mobile edge computing," *IEEE Trans. Mobile Comput.*, early access, Jun. 23, 2020, doi: 10.1109/TMC.2020.3004225.

[25] T. M. Mengistu, A. Albuali, A. Alahmadi, and D. Che, "Volunteer cloud as an edge computing enabler," in *International Conference on Edge Computing*. Cham, Switzerland: Springer, 2019, pp. 76–84.

[26] J. Panadero, J. de Armas, X. Serra, and J. M. Marquès, "Multi criteria biased randomized method for resource allocation in distributed systems: Application in a volunteer computing system," *Future Gener. Comput. Syst.*, vol. 82, pp. 29–40, May 2018.



Babar Ali received the B.S. degree in computer science from COMSATS University, Islamabad, Pakistan, in 2017, and the master's degree in computer science from the Lahore University of Management Sciences, Lahore, Pakistan, in 2019.

His research interests include resource management in IoT, fog computing, and wireless sensor networks.

Muhammad Adeel Pasha (Senior Member, IEEE) received the B.S.-E.E. degree in electrical and computer engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2004, the M.S. degree in electrical and computer engineering from the University of Nice Sophia-Antipolis, Nice, France, in 2007, and the Ph.D. degree in electrical and computer engineering from the University of Rennes, Rennes, France, in 2010.

He is an Assistant Professor with the Electrical Engineering Department, Lahore University of Management Sciences, Lahore. His research interests include low-power microarchitecture, energy-efficient hardware design, and futuristic computing platforms for green computing and communications.

Saif ul Islam received the Ph.D. degree in computer science from the University Toulouse III Paul Sabatier, Toulouse, France, in 2015.

He is an Assistant Professor with the Department of Computer Science, Institute of Space Technology, Islamabad, Pakistan, where he is currently leading the Research and Development Cell. He has published in various high-impact factor journals. His research interests include resource and energy management in large-scale distributed systems and the Internet of Things.

Houbing Song (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the University of Virginia, Charlottesville, VA, USA, in 2012.

Since August 2017, he has been working as an Assistant Professor with the Department of Electrical Engineering and Computer Science, Embry-Riddle Aeronautical University, Daytona Beach, FL, USA. His research interests include cyber–physical systems, cyber security and privacy, Internet of Things, edge computing, AI/machine learning, big data analytics, and networking.

Dr. Song has served as an Associate Technical Editor for the *IEEE Communications Magazine* from 2017 to 2020, and an Associate Editor for IEEE INTERNET OF THINGS JOURNAL in 2020.

Rajkumar Buyya (Fellow, IEEE) received the Ph.D. degree in computer science and software engineering from Monash University, Melbourne, VIC, Australia, in 2002.

He is a Professor and Future Fellow of the Australian Research Council, and the Director of the Cloud Computing and Distributed Systems Laboratory, University of Melbourne, Parkville, VIC, Australia. He has authored more than 425 publications and four text books including *Mastering Cloud Computing* (McGraw Hill and Elsevier/Morgan Kaufmann, 2013). Microsoft Academic Search Index ranked him as the world's top author in distributed and parallel computing from 2007 to 2012.