future internet

MDPI

Article

A Dynamic Application-Partitioning Algorithm with Improved Offloading Mechanism for Fog Cloud Networks

Adeel Abro [1], Zhongliang Deng [1], Kamran Ali Memon [1,*], Asif Ali Laghari [2], Khalid Hussain Mohammadani [1] and Noor ul Ain [3]

[1] School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China; adeelabro@bupt.edu.cn (A.A.), dengzhl@bupt.edu.cn (Z.D.), khalid.mohammadani@gmail.com (K.H.M.)

[2] Department of Computer Science, Sindh Madressatul Islam University, Karachi 74700, Pakistan, asifalilaghari@gmail.com

[3] School of Information & Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China; noorulain194014@bupt.edu.cn

* Correspondence: ali.kamran77@gmail.com

Received: 14 May 2019; Accepted: 21 June 2019; Published: 28 June 2019

Abstract: This paper aims to propose a new fog cloud architecture that performs a joint energy-efficient task assignment (JEETA). The proposed JEETA architecture utilizes the dynamic application-partitioning algorithm (DAPTS), a novel algorithm that efficiently decides and switches the task to be offloaded or not in heterogeneous environments with minimal energy consumption. The proposed scheme outperforms baseline approaches such as MAUI, Think Air and Clone Cloud in many performance aspects. Results show that for the execution of 1000 Tasks on fog, mobile offloaded nodes, JEETA consumes the leas, i.e., 23% of the total energy whereas other baseline approaches consume in between 50%–100% of the total energy. Results are validated via real test-bed experiments and trice are driven efficient simulations.

Keywords: MECA; offloading system; augmented reality; application partitioning; dynamic applications

1. Introduction

Applications of the Internet of things (IoT) such as e-healthcare, e-vehicle, e-commerce, etc. have been growing progressively since the last decade [1]. Fog computing is an extension of cloud computing. Cloud services are available on the pay per use, but the distance of end-user from cloud increase the network latency, which degrades the service perception and increase energy usage [2,3]. Fog computing (FC) is a promising paradigm that offers cloud services with lower end-to-end latency at the edge of the Internet [4,5] as illustrated in Figure 1. In the FC network, the fog nodes are servers, which are placed at the edge of the Internet [6,7]. In general, terms, fog nodes are the subset of public cloud, and fog nodes have limited capacity as compared to the public cloud [8,9]. As a reason, computation offloading is a method used to transfer resources intensive computation tasks of an application to the external server for processing [10,11]. Each IoT application is partitioned into local tasks and remote tasks so that the entire application executed efficiently [12]. Whereas, resource-constrained devices (i.e., storage, computation power, battery) only process lightweight tasks of an application, wherein, computation intensive tasks of an application are more likely to be offloaded to the FC for processing [13].

However, the applications above generate real time data; it is difficult to execute them on local devices with energy efficiency in FC [14]. Existing studies [15,16] proposed their strategies to save device energy. However, the energy consumption of FC resource due to the process of large tasks has been ignored for many years. This poses the following research challenges to be addressed when the workload of IoT application to efficient assignment on the fog cloud network is under consideration.

1. How to develop the data flow and data process in a fog cloud paradigm for the workflow IoT application? Due to the fact that it is different from traditional data flow and data process mechanism, which have been adopted by many cloud architectures to process the real time data [17].
2. How to make the assignment of local tasks on devices without degrading application performance and device energy must be under given threshold value?
3. How to make the assignment of remote tasks on the fog nodes efficiently with minimal consumption of the fog resources and without degrading the quality of experience (QoE) of the application?

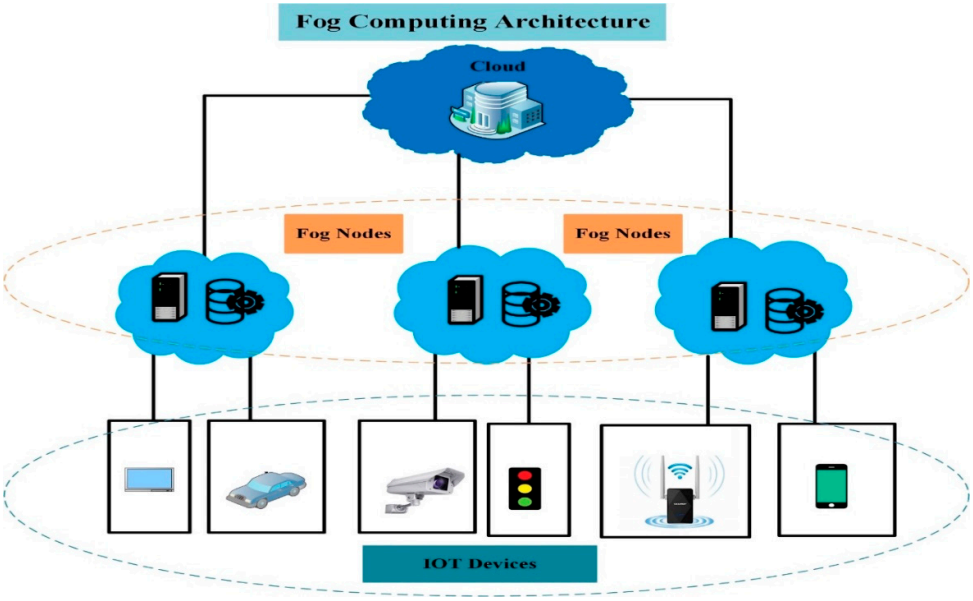

Figure 1. Fog computing: The general architecture [5].

This paper is based on the energy efficient task assignment in devices and FC networks. Every IoT application is represented as a workflow application. FC is comprised of heterogeneous virtual machines (VMs); each VM is different in the context of speed and storage from each other. The objective of the paper is to efficiently perform the assignment of local and remote tasks on devices and FC-VMs so that the total energy can be minimized. The total energy, in general, is the combination of local devices and FC resources, energy consumption, which incurs due to the execution of the local tasks and remote tasks, respectively. Apart from existing studies in terms of data flow and data and the process when assignments of tasks are mapped on the FC networks, this paper has the following contributions in response to challenges as mentioned earlier.

1. This paper proposes new fog cloud architecture (FCA), which efficiently manages the data flow and data processing mechanism for all IoT applications without losing any generosity of the system. FCA exploited many profiling technologies to regularly monitor the entire system performance in term of resource utilization and energy management. The propose FCA architecture will be explained in the proposed description section.
2. A joint energy efficient task assignment (JEETA) method is proposed to cope up with the problem in the fog cloud environment.

3. In order to minimize the total energy consumption of the entire system, a dynamic application partitioning task assignment algorithm (DAPTS) is presented. This DAPTS determines how to perform the assignment of local tasks on the devices and remote tasks on the fog nodes.
4. DAPTS exploits dynamic voltage frequency scaling (DVFS) method to manage and reduce the entire energy consumption of the FCA after task assignment phase without degrading any system performance.

The rest of the paper is organized as follows. Section 2 elaborates related work and Section 3 explains the proposed model and formalizes the problem under study. A heuristic is proposed for the considered problem in Section 4, which describes the proposed algorithm and sequences. Section 5 evaluates the simulation part, and Section 6 presents the conclusive remarks.

2. Related Work

The energy efficient task assignment problem in the mobile fog-cloud computing has been investigated widely to achieve different goals. The author in [14] proposed a fog-cloud architecture to reduce the energy consumption of the resources. The goal was to process real time data generated by different sensors at the fog-nodes. These nodes are separated by long distances and need to transmit so much energy to the multiple hops away cloud computing. An energy-aware optimal task assignment for mobile heterogeneous embedded systems in cloud computing has proposed in [18]. This research work focused on the assignment of different tasks on heterogeneous resources of the public in order to execute them with a given deadline. The DVFS technique is exploited by prior works in order to manage energy consumption in cloud resources. K. Gai. et al. [18] in the extended version [19] proposed additional phases to assign the workload among different cloud servers in order to manage load and energy consumption for all offloaded tasks to the cloud system. Exploiting massive D2D collaboration for energy-efficient mobile edge computing algorithm is proposed in [20] where all application tasks are executed between devices in order to minimize devices energy utilization. Fog cloud and device computing environment and an energy-efficient with incentive-aware task offloading framework via network-assisted D2D has been investigated in [21], to observe the application performance and energy consumption tradeoffs. The present works [22,23] have proposed fog-cloud architecture and energy efficient task assignment and task scheduling methods emphasizing that all applications can be executed in a parallel manner in cloud-fog architecture. [24] stresses the same task assignment and scheduling approach via network-assisted D2D collaboration.

Several researchers have done work on energy efficiency in a cloud computing environment by considering different parameters, such as VM migration, device offloading and algorithms of allocation of VMs [25]. One such work provided by Verma et al. and proposed three heuristic models namely Maximum Fill (MF), Smallest Void Detection (SVD) and Median Migration Time (MeMT) for energy reduction with a slight change in SLA [26]. During the experiment, the energy problem of resource utilization was analyzed and cloud-computing model for energy consumption was developed. This work focused on the linear relationship between resource utilization and energy consumption. Focusing VM migration and resource allocation a technique was considering for synchronization oriented shorter stop-and-copy phase. The algorithms were developed for complete operation step for energy-aware heuristic models, including MeMT, SVD, and MF. The evaluation of data of VMs was employed in PlanetLab servers. The results show that the proposed approaches can reduce power consumption during host shutdown and VM migration while maintaining the high performance of the system.

The author proposed an adaptive heuristic based threshold algorithm for detecting the overloaded hosts in the cloud data center by gradient descent regression model using a statistical analysis of past CPU utilization data [27]. This algorithm sets a dynamic upper threshold for detecting overloaded host, thereby minimizing the performance degradation, which directly affects SLA. They also introduced the VM selection algorithm called bandwidth-aware selection policy. This algorithm selects a VM based on network traffic, which directly affects migration cost. Therefore, it significantly minimizes the VM migration cost from one host to another host. Proposed energy-aware methods for detecting overloaded host and VM selection from an overloaded host are minimizing energy

consumption and SLA violation of a cloud data center. After detecting underloaded hosts, all VM should be migrated from underloaded host to the most appropriate host and turn all idle host to energy saving mode. This would directly minimize the energy consumption of the cloud data center.

Rahul et al. proposed an adaptive energy-efficiency algorithm for overloaded host detection and VM selection from overloaded or underloaded hosts in the cloud data center [28]. They introduced the overloaded host's detection method based on a least median square regression model using a statistical analysis of past CPU utilization traces. The analysis of this real-time CPU utilization algorithm traces and sets an upper threshold for detecting overloaded hosts. This minimizes performances degradation of hosts in the cloud-computing environment. They used cloudsim simulator for implementing the proposed algorithm and used five performance efficiency matrices, namely, total energy consumption, percentage of SLA violation, number of host's shutdowns, and number of VM migration. They also introduced VM selection scheme based on the size of memory allocated to the particular VM. This scheme significantly minimizes the overall migration cost and SLA violation of the VM from overloaded hosts to appropriate hosts.

With the best knowledge, joint energy efficient task assignment problem for IoT workflow application has not been studied yet. Related literature reveals that IoT applications are divided into local and remote tasks but are limited to focus on one part, i.e., fog-cloud resource energy or device energy that does not meet the requirement of applications. On the other hand, IoT workflow applications data flow and data process model is quite different from existing workflow. Overall, existing studies have ignored this aspect for computation offloading and task assignment fog cloud architecture. The proposed FCA determines efficient data flow and data process mechanism in which workflow application local and remote tasks are effectively assigned to the local devices and fog nodes, which minimizes device and cloud resources energy simultaneously. The proposed JEETA has two phases, such as the assignment phase and energy minimization phase, respectively. The detailed analysis of the contributions is presented in the following sections.

3. Proposed Model

The data flow of the proposed architecture FCA is simple. The user submits the application to the system, based on different profiling technologies where tasks are sequenced according to estimation time in the master node. Figure 2 illustrates this. The master node in the proposed architecture is the FCA central controller, which is responsible for managing the priority of task sequencing and performance in the system. The data processing of tasks on devices and fog resources is scheduled by a scheduler. The scheduler maps their sources to the remote tasks on different heterogeneous VMs and local tasks are mapped on the devices. However, the data dependencies incur some extra communication cost due to the computation being held in different places. For instance, local tasks are computationally executed on devices and remote tasks are computationally executed on the fog VMs.

3.1. Offloading Cost

Workflow application tasks are closely linked to each other and follow by precedence constraints rules. According to the offloading system, rule partitions an application into two disjoint sets, for example, local disjoint set V_{loc} and remote disjoint set V_{cl}. In the work flow application, the offloaded disjoint set in the cloud via wireless network incurred with communication cost. The total cost C_{total} of an application partitioning, is formulated in the following equation.

$$C_{total} = \sum_{v \in V} Pr_v . w^{local}(v) + \sum_{v \in V} (1 - Pr_v) . w^{local}(v) + \sum_{e(v_i, v_{j)} \in E} Pr_e . w(e(v_i, v_j)) \quad (1)$$

However, partitioning P_r total cost is C_{total}, $\sum_{v \in V} Pr_v . w^{local}(v)$ denotes the local execution cost and $\sum_{v \in V} (1 - Pr_v) . w^{local}(v)$ denotes server execution cost. In the end $\sum_{e(v_i, v_{j)} \in E} Pr_e . w(e(v_i, v_j))$ is the communication cost between tasks v_i and v_j.

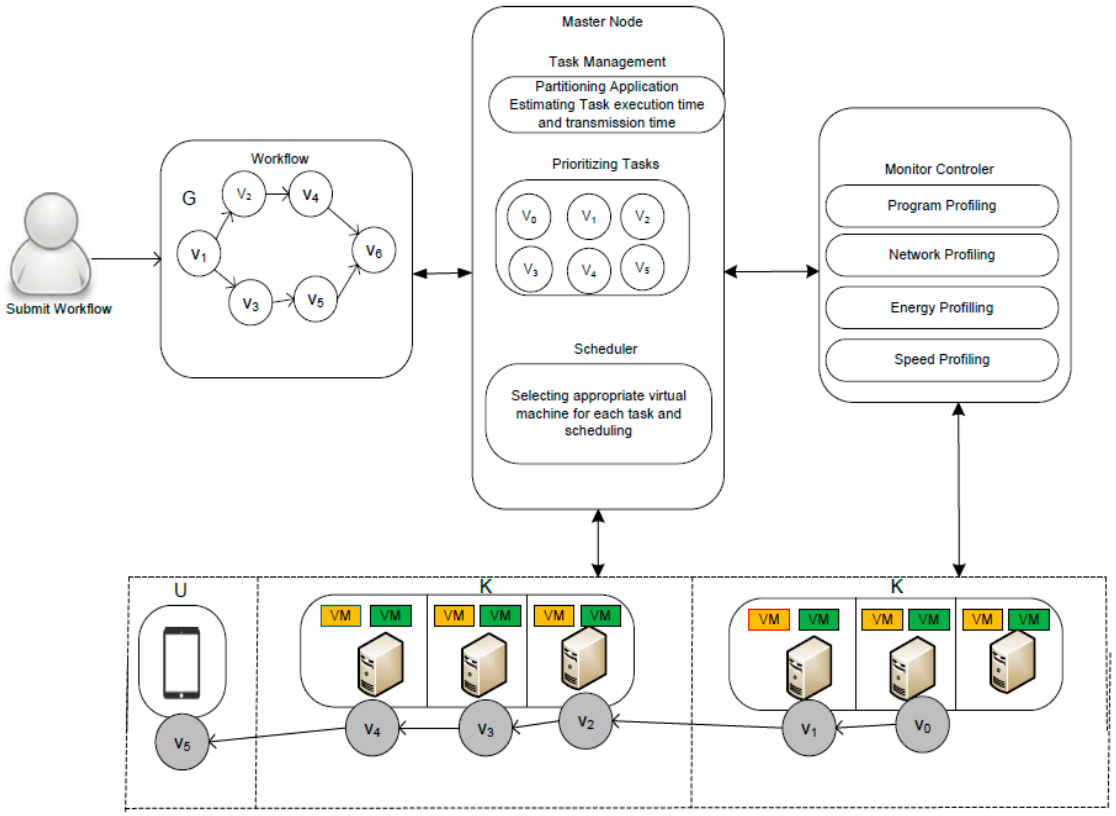

Figure 2. Mobile cloud architecture.

3.2. System Model and Problem Formulation

A mobile edge capture and analytics (MECA) is the combination of the wireless network, mobile edge server and cloud server [29]. The mathematical notations are marked in Table 1. In MCA a work flow, mobile health care application is modeled as consumption weighted directed acyclic graph (DAG), i.e., G (V, E). Whereas, each task v_i is represented by a node $v_i \in V$. An edge e(v_i, v_j) ∈ E represents communication between v_i to v_j. A task v_i could be started until all associated predecessors complete [30]. v_1 and v_n are two dummy tasks (i.e., entry task and exist task). A task could be started until associated predecessors complete. In simple words v_j cannot be started in anticipation of v_i to get the job done and $i < j$. A set of servers can be represented by $K = \{k_1, \ldots., k_n\}$. It is presupposed that each k server holds different *VM* type, that all *VM* instances are heterogeneous, every VM has dissimilar computation speed which are illustrated as $\zeta_j = (j = 1, \ldots., M)$. A set of VM instances can be shown by $V_K = (v_{k1}, \ldots., v_{kn})$, in which K_i^{vk} is the VM assignment for task v_i. Each workflow application task has workload $W_i = (i = 1, \ldots., N)$ with deadline $D_{ia\,a}$. To minimize the power consumption of the submitted workflow tasks, each application task is assigned to the lower speed VM while meeting the deadline D_i, because the lower speed VMs always leads to lower power consumption [31]. Since a task v_i can only be performed by one VM j, a decision variable $x_{ij} \in \{0,1\}$ is utilized, $x_{ij} = 1$ only if the task v_i is assigned to the VM v_j. The task v_i has two execution costs (i.e., local and cloud), one cloud execution time is determined by the speed ζ_j and power consumption P_w e.g., T_i^e, i.e., $C_i^e = \sum_{j=1}^{V} x_{ij} \times \frac{W_i}{\zeta_j} \times P_{cw}$ and task execution time on mobile $M_i^e = \sum_{j=1}^{\leftarrow} x_{ij} \times \frac{W_i}{\zeta_m} \times P_{mw}$.

Table 1. Mathematical notation.

Notation	Description
N	Number of tasks v

$\Omega.\lambda$	Application partitioning factor during offloading
D_i	Deadline of the application
ζu_i	Speed rate of jth virtual machine VM
ζu_m	Speed rate of mobile processor v_i
C_i^e	Execution cost of the task v_i on cloudlet k
M_i^e	Execution cost of the task v_i on mobile
P_{cw}	Power consumption rate at cloudlet virtual machine VM
P_{mw}	Power consumption rate at the mobile device
P_{ij}	Assignment of task v_i on virtual machine VM, j
B_i	Begin time of the task v_i
F_i	Finish time of the task v_i
G (V, E)	DAG call graph

3.3. Application Energy Consumption

The total power consumption of workflow application is a merger of computation time and communication time. Since computation cost could include location and remote execution after application partitioning. The communication cost is determined by the weight of data transport and available network bandwidth. The average power consumption of workflow application due to offloading is expressed as follows:

$$EG_{total} = \sum_{v \in V} Pr_v . EG_v^{loc} + \sum_{v \in V} (1 - Pr_v) . EG_v^{cl} + \sum_{e(v_i, v_j) \in E} Pr_e . T_e^{trans} \tag{2}$$

whereas Equation (2) describes the average consumption of the workflow application, which is the sum of local and remote computation cost and communication cost. In the same way, the vector $Y = \{y_{ij} : v \in V, j \in M\}$ with variable x_{ij} indicates either task offload or not, namely:

$$x_{ij} = \begin{cases} 1, \text{if } v_i \in v_{loc} \\ 0, \text{if } v_i \in v_{cl} \end{cases} \tag{3}$$

According to Equation (4), the communication cost is determined by $E_{cut} = 1$, otherwise same location tasks have no communication cost.

$$P_{r_e} = \begin{cases} 1, \text{if } e \in E_{cut} \\ 0, \text{if } e \notin E_{cut} \end{cases} \tag{4}$$

The considered problem is mathematically modeled as below:

$$minZ = \sum_{i=1}^{N} \sum_{m=1}^{\psi} \sum_{j=1}^{M} x_{ij} \lambda_i \times \zeta_{\mu j} \times P_{cw} \times C_i^e + T_i^{trans} + x_{ij} \lambda_i \times \zeta_{\mu m} \times P_{cm} \times M_i^e \tag{5}$$

$$\sum_{v \in V} Pr_v . EG_v^{loc} = M_i^e = \sum_{i=1}^{N} x_{ij} \lambda_i \times \frac{W_i}{\zeta_{\mu m}} \tag{6}$$

$$\sum_{v \in V} (1 - Pr_v) . EG_v^{cl} = C_i^e = \sum_{i=1}^{N} x_{ij} \lambda_i \times \frac{W_i}{\zeta_{\mu j}} \tag{7}$$

$$\sum_{e(v_i, v_j) \in E} Pr_e . T_e^{trans} = w\left(e(v_i, v_j)\right) = \frac{in_{ij}}{B_U} + \frac{out_{ji}}{B_D} \tag{8}$$

$$T_{j,0} = 0,$$

$$T_{i,j} = T_{i-1,j} + \sum_{i=1}^{N} x_{ij} \lambda_i \, C_i^e, \tag{9}$$

$$C_i^e = \sum_{i=1}^{V_c} x_{ij}\lambda_i \times \frac{W_i}{\zeta_{\mu j}}, M_i^e = \sum_{i=1}^{V_{loc}} x_{ij}\lambda_i \times \frac{W_i}{\zeta_{\mu m}}, \quad (10)$$

$$F_i = \sum_{j=1}^{M} T_{i,j}\, x_{ij}, \quad (11)$$

$$\sum_{i=1}^{N} x_{ij} = 1, \sum_{j=1}^{M} x_{ij}\lambda = 1, \sum_{m=1}^{\psi} x_{ij}\lambda_0 = 1, \quad (12)$$

$$F_i \le D_{i,} x_{ij}\{0,1\}, \quad (13)$$

Equation (5) is proposed for the measurement of power consumption of all tasks from start to end on clouds and local nodes on execution. Equation (6) is given for power consumption of mobile devices that how much power consumed during the execution of a particular task. Energy consumptions at the cloud side can be calculated by using Equation (7) and the same way for the cost of communication in power consumption Equation (8) is given. According to Equation (9) initial time for any VM machine is assumed to be zero. The finish time of a task v_i on VM j is $T_{i,j}$ determined by the previous task $v_i{}^{-1}$ execution time that is $\sum_{v=1}^{N} x_{ij}\, C_i^e$ is shown in Equation (10). Equation (11) determines the execution cost of a task on the cloud and mobile device according to their speed and weight. The task finish time is determined by Equation (12). Equation (13) demonstrates that each task can be only assigned to one VM and each VM can be only assigned to one task. According to Equation (14) finish time of all tasks (V_{loc} and V_{cl}) should be less than a given deadline D_i.

4. Proposed Algorithm DAPTS

The proposed DAPTS algorithm will work on the transfer of tasks on local as well as on fog heterogeneous devices to optimize the overall energy consumption in fog/cloud environment. The proposed DAPTS algorithm is based on the task assignment and minimizes energy consumption. Initially, task assignment has two strategies, for instance, local task assignment and remote task assignment strategy. Both tasks determine the optimal assignment of tasks on their resource. After the assignment of all tasks, DAPTS exploits DVFS technique to reduce the energy consumption of the entire system in the FCA, which is given in Algorithm 1 JEETA framework.

Algorithm 1: JEETA Framework

Input : $v_i \in$ V ;
Output: min Z;
1 **begin**
2 Z ←0; 3 ;
3 **for each** $v_i \in$ V **do**
4 Call Local assignment of v_i ; //assignment of local task in JEETA
5 Call Remote assignment of v_i; // assignment of remote task in JEETA
6 Z ← Z_i–Z;
7 Call DVFS;
8 **return** Z;

4.1. Local Task Assignment

The purpose of the local task assignment algorithm is to transfer tasks they required low resources, which are available nearby location. This will support the fast execution of tasks as well as minimize energy consumption during the process. The local tasks assignment algorithm starts with input, i.e., v_i∈V_loc, put in Algorithm 2. These tasks are likely to map on the devices. These tasks are lightweight in size and require lower computation as compared to remote tasks.

Algorithm 2: Local Task Assignment

Input : $v_i \in V_{loc}$; //tasks to be mapped
begin
  $thm \leftarrow$ profile; //current energy level of the device
  $d_i \leftarrow D_i$; //system divides the deadline among all tasks
  $M_i^e \leftarrow 0$; //execution of each task on the device
  for each $v_i \in V_{loc}$ **do**
    if $M_i^e \leq d_i$ then //execution of local tasks under a given deadline
      $W_i \leftarrow v_i$;
      $M_i^e = \frac{W_i}{\zeta_{\mu m}}$; //time on a device resource
      $thm \leftarrow M_i^e . p_{mw}$;
    $thm \leftarrow$ initial energy after assignment;
  return *thm*;

4.2. Remote Task Assignment

The purpose of the remote task assignment algorithm is to transfer tasks they required more resources, which are available on other fog/cloud network location which far away from the client. This will support the more fast execution of tasks as compared to local task assignment but also required more middle network involved during the remote task assignment, which also adds cost and energy consumption as compared to local task assignment, but more efficient compared to previously proposed algorithms for task assignment and energy efficiency. The remote tasks assignment algorithm starts with input, i.e., $v_i \in V_{loc}$, put in Algorithm 3. These tasks are likely to be mapped on the fog nodes resources. These tasks are heavyweight in size and required higher computation as compared to local tasks.

Algorithm 3: Remote Task Assignment

Input : $v_i \in V_{loc}$; //tasks to be mapped
begin
  $thf \leftarrow$ profile; //current energy level of the device
  $d_i \leftarrow D_i$; // system divides the deadline among all tasks
  $C_i^e \leftarrow 0$; //execution of each task on the device
  for each $v_i \in V_{cl}$ **do**
    if $M_i^e \leq d_i$ then //execution of remote tasks under a given deadline
      $W_i \leftarrow v_i$;
      $C_i^e = \frac{W_i}{\zeta_{\mu j}}$;
      $thf \leftarrow C_i^e . p_{cw}$; // energy consumption of record task execution
    $thf \leftarrow$ initial energy after assignment;
  return *thf* ;

4.3. Energy Efficiency Phase

It is noteworthy that the presented approach uses energy minimization after initial task assignment, i.e., all tasks have already been done. Based on Algorithm 4, offloading engine module would try to re-shuffle tasks among different mobile cores without degrading application performance. Further, the DVFS technique is employed in the algorithm to create a balance between the execution frequency of a high efficiency with maximum frequency core. Thus, high potential cores have similar performance as a small core, the assigned kth core consumes more energy than the small performance core. So, DVFS Algorithm 4 is used after the task re-shuffling. However, the algorithm

determines each unit of execution for each task before applying the DVFS algorithm for task switching among different cores.

Algorithm 4: DVFS Technique

Input : $v_i \in V_{loc}$;
Output: min E;
begin
  flag ← 0;
  flag ← 0;
  for each $v_i \in V_{loc}$ **do**
    while (flag = = 0 and m < M) **do**
      calculate new execution on k^{th}core T_i^{loc} if v_i task using k^{th} frequency;
      $E_{i,k}^{loc} = P_k . T_i^{loc}$;
      if $E_{i,k}^{loc} < K$ **then**
        $E_{i,k}^{loc} = P_k . T_i^{loc}$;
        flag = 1;
      if (flag = = 1) **then**
        update new execution for task v_i ;
        $E_{i,k}^{loc} = P_k . T_i^{loc}$ Apply DVFS technique for all task;
    update $\sum_i^{V_{loc}} E_{i,k}^{loc} = P_k . T_i^{loc}$
  return min E;

5. Performance Evaluation

To evaluate the performance of the proposed method JEETA, this paper presents an implementation of three baseline approaches in the mobile cloud computing environment. The baseline approaches are MAUI, Clone Cloud, and Think Air implemented within the simulation. The simulation parameters are used in the experiment system are illustrated in Table 2. Six benchmark applications are deployed in the mobile fog cloud environment, listed in Table 3 as workload analysis. The mobile fog cloud resource specification is illustrated in Table 4.

Table 2. Simulation parameters.

Experiment Parameters	Values
λ_i user arrival time	5 s
Languages	JAVA
Workload	A-G, Health-care
Experiment time	12 h
Replication Repetition	12
No. of Mobile devices	100 to 1000
Location user Mobility	Trajectory function
No. of VMs.	10–200

Table 3. Workload analysis.

Workload	W_i (byte)	T_i^k (MI)	N
Healthcare	825	5.8	700
Augmented Reality	525	4.8	800
Business	525	4.8	800
3D-Game	325	2.8	1000
Face-Recognition	725	7.8	700
Social Net	625	9.8	800

Table 4. VMs Specifications.

Fog Node VM	CORE	MIPS/CORE	Storage (GB)
VM1	1	200	200
VM2	1	400	400
VM3	1	600	600
VM4	1	800	800
VM5	1	1000	1000
Public Cloud VM	**CORE**	**MIPS/CORE**	**Storage (GB)**
VM1	1	500	500
VM2	1	1000	1000
VM3	1	1500	1500
VM4	1	2000	2000
VM5	1	3000	3000

5.1. Workflow Complex Application

The experiments are conducted on augmented reality application and complex health-care application to determine the effectiveness and efficiency of the proposed algorithm. The application source code is available at [32,33].

5.2. Profiling Technologies

A workflow application can be shown as a call graph in an open source connection scrutiny tool which shows that the energy profiling for upload and download of the data. This is also available at [34]; energy profiling one can use this tool mentioned in [35]. The health-care workflow application is partitioned into local and remote execution. Each task v_i is represented by a node, whereas, each node has exactly two costs (i.e., local execution and cloud execution cost). Further, the application is partitioned under F = 2 speedup factor and available bandwidth = 1 M/B respectively. Whereas, blue nodes are performed locally, and red nodes are offloaded to the cloud for performing. Among this, the proposed JEETA method will re-partition the application if the wireless bandwidth B or the speedup factor F diverges.

Figure 3 shows the individually mobile computing and the fog node energy consumption due to the execution of the benchmark application. JEETA manipulates all applications with minimum energy consumption as compared to baseline approaches because of JEETA partitions the application at run-time between device and fog in order to minimize energy consumption. Tasks divided into parts and performance of the proposed algorithm is tested.

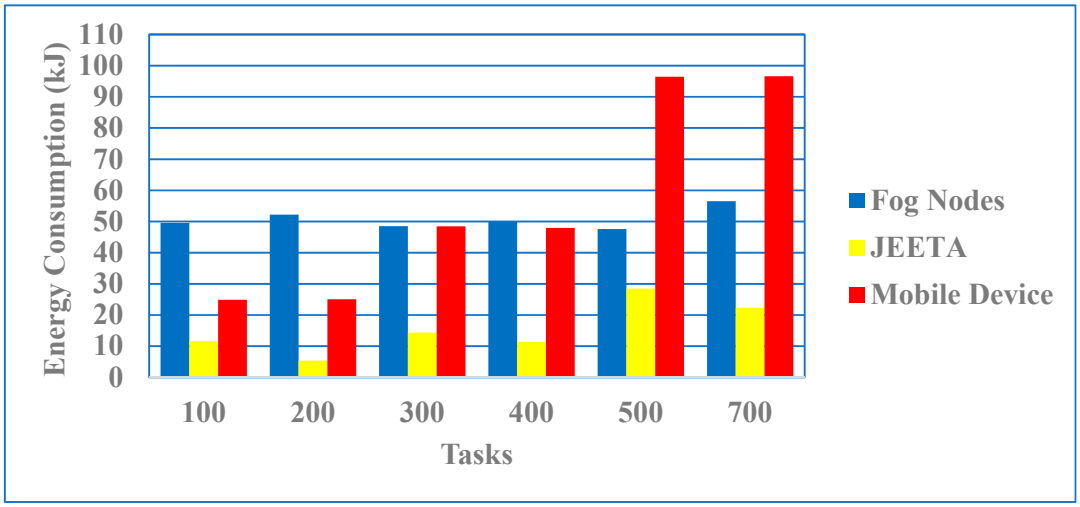

Figure 3. Energy consumption of entire mob-cloud.

Results show that in Figure 4 the proposed algorithm performs better on different speedup factor F and B bandwidth than existing heuristic techniques such as MAUI, ThinkAir, and Clone Cloud in application partitioning. It is also noted that on a higher number of applications, MAUI has nearly the same energy consumption as JEETA.

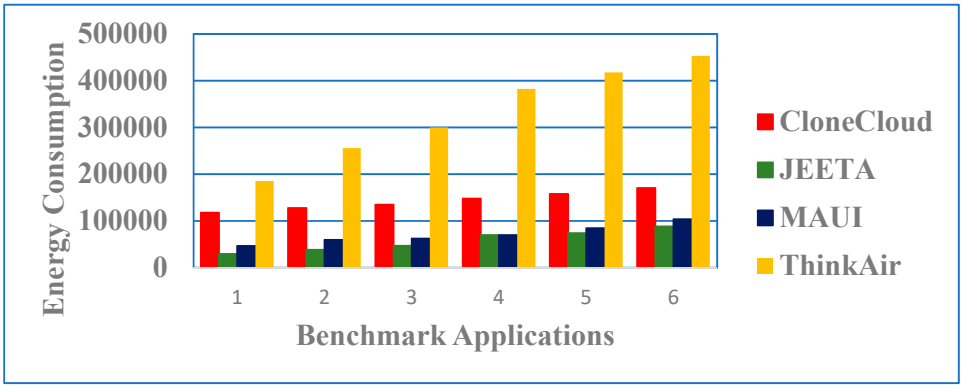

Figure 4. Application partitioning performance based on speed up factor F and bandwidth B.

Figure 5 illustrates the comparison of all three heuristic techniques against JEETA in terms of number of tasks completed within a given deadline. Results in Figure 6 show that JEETA consumes less energy as per the number of tasks are increased. MAUI consumes less energy on a low number of tasks, but when the number of the task increased then it consumes more energy. Clone Cloud consumes more energy as compared to all others in this figure. The proposed algorithm performs better on different speedup factor F and B bandwidth than existing heuristic techniques.

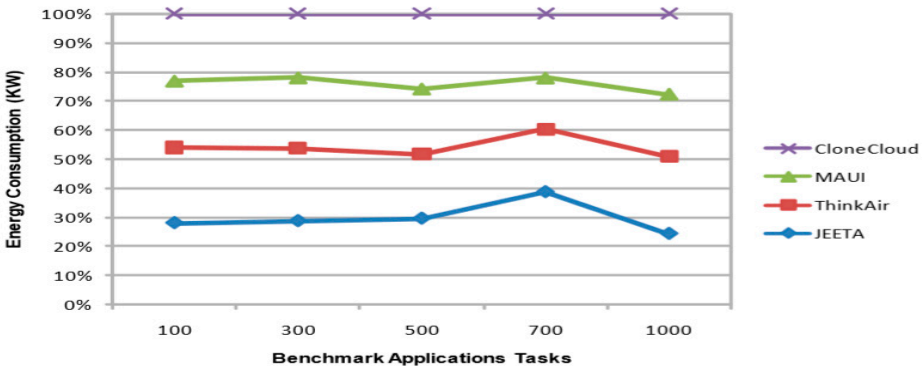

Figure 5. Workflow application: Number of tasks completed within a given deadline.

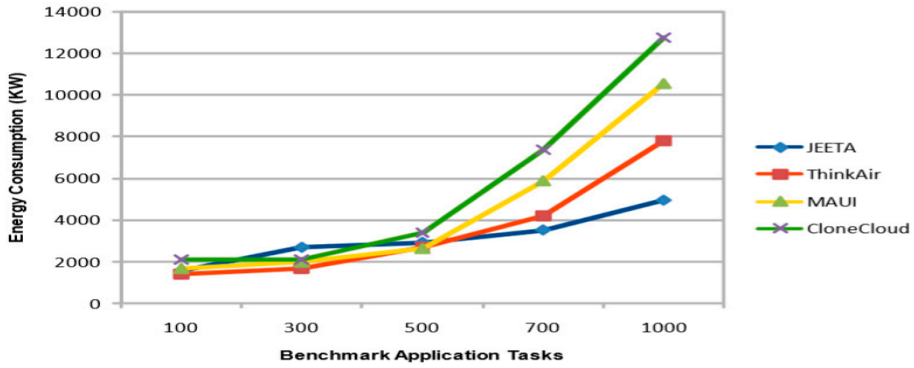

Figure 6. Workflow application number of tasks completed within a given deadline.

5.3. Total Energy Consumption in Scenario

Energy consumption of the mobile device and fog nodes is measured in terms of a kilowatt (kW) while executing the benchmark applications on the mobile cloud system; each mobile device has two interfaces such as local execution interface and network interface. Moreover, a previously mentioned interface directly affects the mobile device energy consumption during local execution and offloading application tasks. Thus, the tradeoff balance between local energy consumption and offloaded energy consumption in the proposed mobile fog environment is achieved. Whereas, the aim of a proposed joint energy efficient algorithm allows applications to execute application tasks among multiple resources such as local resource, network resource and fog resources in such way that total energy consumption is minimized.

Figure 7 shows that JEETA outperforms and minimizes the tradeoff energy consumption for all application tasks to the baseline approaches such as MAUI, CloneCloud, and Think Air. These approaches are just on minimizing the energy consumption of the mobile device by the offloading technique in the mobile cloud environment. They ignored the fog nodes or server energy consumption during execution. JEETA technique can optimize both user and provider end energy efficiency instead of partial optimization, which is done in existing studies.

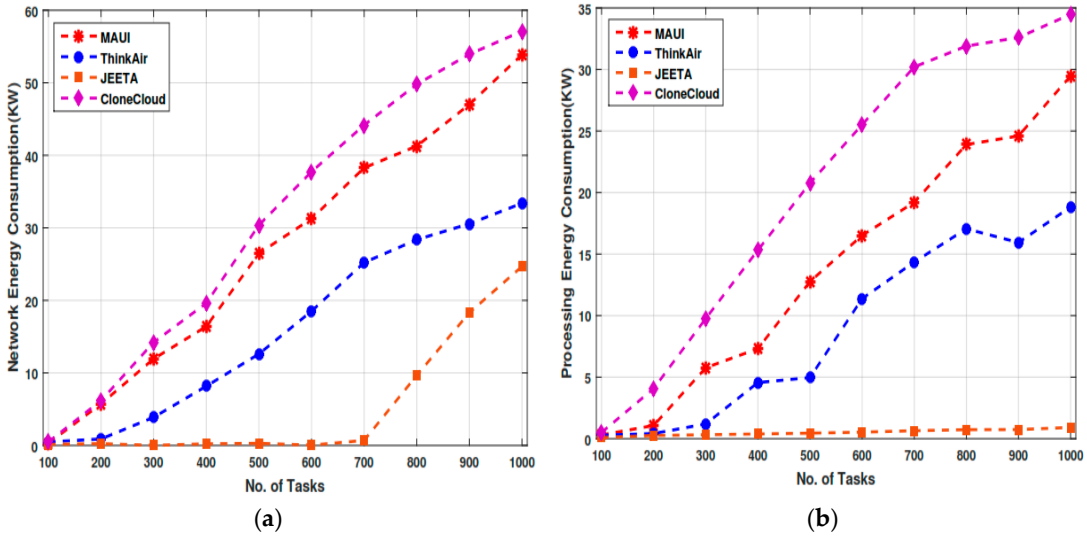

Figure 7. Energy consumptions. (**a**) Network ; (**b**) Processing.

Figure 8 illustrates that; total energy consumption and mobile energy consumption can be saved efficiently by proposed JEETA algorithm as compared to the prior studies. This idea is unique, to optimize the energy consumption from both user and providers for a similar application which will be executed partially among a mobile, a network, and a 231 cloud resource simultaneously. Previous research on offloading mechanisms is based on the code development, app performance increment, resource discovery and allocation, and Elastic execution between mobile devices and clouds [36]. While adapting the application partitioning, JEETA model is based on data flow and task assignment with less energy consumption on the same number of tasks as compared to fog nodes and offloaded mobile nodes.

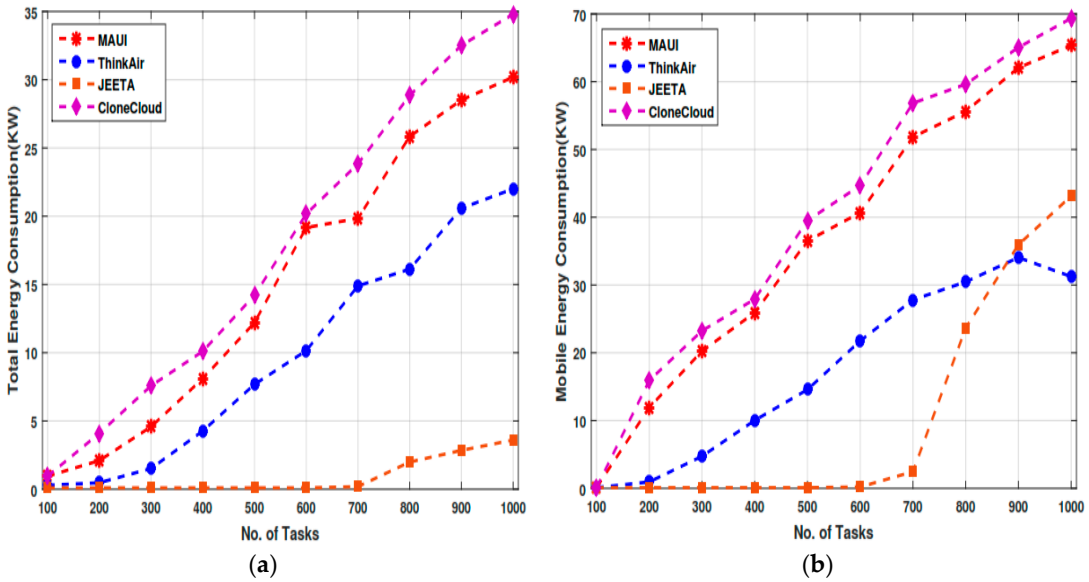

Figure 8. Augmented application performance. (**a**) Without mobility; (**b**) with mobility.

6. Conclusions

This paper presents a new fog cloud architecture that utilizes a joint energy-efficient task assignment (JEETA) method to cope up with the problem in the fog cloud environment. Additionally, a dynamic application-partitioning algorithm is proposed for this new architecture for real time healthcare application. The proposed algorithm evaluated in the healthcare workflow application and compared with benchmark heuristics such as MAUI, Clone Cloud and Think Air, and results show that JEETA finishes all workflow tasks within a given deadline. During the research, two goals were achieved, the first assignment of remote tasks to the fog nodes, which executed with minimal energy consumption of the fog resources. Second, it maintains the QoE of users for the application. In future work, response time and energy consumption minimization in the dynamic application partitioning and task-scheduling problem appear to be challenging and will be addressed in the future research.

Author Contributions: Conceptualization, methodology, original draft preparation and formal analysis was done by A.A. along with K.A.M.; Validation, supervision, and funding of the project by Z.D., A.A.L., K.H.M. and N.u.A. did the validation, original draft preparation and review of draft preparation.

Funding: This research work has been carried out in State Key Laboratory Intelligent Communication, Navigation and Micro-Nano System, School of Electronic Engineering, Beijing University of Posts and Telecommunications (BUPT), Beijing, China. The research work received financial supported by National High Technology 863 Program of China (No. 2015AA124103) and National Key R&D program no 2016YFB0502001. The authors are thankful for the financial support and acknowledge guidance and support provided by State Key Laboratory Intelligent Communication, Navigation and Micro-Nano System, BUPT.

Acknowledgments: The authors appreciate and acknowledge anonymous reviewers for their reviews and guidance.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jeon, K.E.; She, J.; Soonsawad, P.; Ng, P. BLE Beacons for Internet of Things Applications: Survey, Challenges and Opportunities. *IEEE Int. Things J.* **2018**, *5*, 17685540.

2. Laghari, A.A.; He, H.; Khan, A.; Kumar, N.; Kharel, R. Quality of Experience Framework for Cloud Computing (QoC). *IEEE Access* **2018**, *6*, 64876–64890.
3. Laghari, A.A.; He, H. Analysis of Quality of Experience Frameworks for Cloud Computing. Int. J. Comput. Sci. Netw. Secur. **2017**, *17*, 228.
4. Usman, M.J.; Ismail, A.S.; Abdul-Salaam, G.; Chizari, H.; Kaiwartya, O.; Gital, A.Y.; Abdullahi, M.; Aliyu, A.; Dishing, S.I. Energy-efficient Nature-Inspired techniques in Cloud computing datacenters. *Telecommun. Syst.* **2019**, *71*, 275–302.
5. Mahmud, R.; Kotagiri, R.; Buyya, R. Fog Computing: A Taxonomy, Survey and Future Directions BT—Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives; Di Martino, B., Li, K.-C., Yang, L.T., Esposito, A., Eds.; Springer: Singapore, 2018; pp. 103–130.
6. Kumar, V.; Laghari, A.A.; Karim, S.; Shakir, M.; Brohi, A.A. Comparison of Fog Computing & Cloud Computing. *Int. J. Math. Sci. Comput.* **2019**, *1*, 31–41 DOI: 10.5815/ijmsc.2019.01.03.
7. Asif-Ur-Rahman, M.; Afsana, F.; Mahmud, M.; Kaiser, M.S.; Ahmed, M.R.; Kaiwartya, O.; James-Taylor, A. Towards a Heterogeneous Mist, Fog, and Cloud based Framework for the Internet of Healthcare Things. *IEEE Internet Things J.* **2018**. doi:10.1109/jiot.2018.2876088.
8. Aliyu, A.; Tayyab, M.; Abdullah, A.H.; Joda, U.M.; Kaiwartya, O. Mobile Cloud Computing: Layered Architecture. In Proceedings of the 2018 Seventh ICT International Student Project Conference, Nakhon Pathom, Thailand, 11–13 July 2018.
9. Usman, M.J.; Ismail, A.S.; Chizari, H.; Abdul-Salaam, G.; Usman, A.M.; Gital, A.Y.; Kaiwartya, O.; Aliyu, A. Energy-efficient Virtual Machine Allocation Technique Using Flower Pollination Algorithm in Cloud Datacenter: A Panacea to Green Computing. *J. Bionic Eng.* **2019**, *16*, 354–366.
10. Kaiwartya, O.; Abdullah, A.H.; Cao, Y.; Altameem, A.; Prasad, M.; Lin, C.T.; Liu, X. Internet of Vehicles: Motivation, Layered Architecture Network Model Challenges and Future Aspects. *IEEE Access* **2016**, *4*, 5356–5373.
11. Zheng, J.; Cai, Y.; Wu, Y.; Shen, X. Dynamic Computation Offloading for Mobile Cloud Computing: A Stochastic Game-Theoretic Approach. *IEEE Transactions on Mobile Computing*, **2018**, *18*, 771–786.
12. Ravi, A.; Peddoju, S.K. Mobile Computation Bursting: An application partitioning and offloading decision engine. In Proceedings of the 19th International Conference on Distributed Computing and Networking, Varanasi, India, 4–7 January 2018.
13. Aliyu, A.; Abdullah, A.H.; Kaiwartya, O.; Cao, Y.; Usman, M.J.; Kumar, S.; Lobiyal, D.K.; Raw, R.S. Cloud Computing in VANETs: Architecture, Taxonomy, and Challenges. *IETE Techn. Rev.* **2018**, *35*, 523–547.
14. Cao, Y.; Song, H.; Kaiwartya, O.; Zhou, B.; Zhuang, Y.; Cao, Y.; Zhang, X. Mobile Edge Computing for Big Data-Enabled Electric Vehicle Charging. *IEEE Commun. Mag.* **2018**, *56*, 150–156.
15. Jalali, F.; Hinton, K.; Ayre, R.S.; Alpcan, T.; Tucker, R.S. Fog Computing May Help to Save Energy in Cloud Computing. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 1728–1739.
16. Baccarelli, E.; Naranjo, P.G.V.; Scarpiniti, M.; Shojafar, M.; Abawajy, J.H. Fog of Everything: Energy-Efficient Networked Computing Architectures, Research Challenges, and a Case Study. *IEEE Access* **2017**, *5*, 9882–9910.
17. Peng, M.; Yan, S.; Zhang, K.; Wang, C. Fog Computing based Radio Access Networks: Issues and Challenges. *arXiv* **2015** arXiv:1506.04233.
18. Yuan, D.; Yang, Y.; Liu, X.; Chen, J. A data placement strategy in scientific cloud workflows. *Future Gener. Comput. Syst.* **2010**, *26*, 1200–1214.
19. Gai, K.; Qiu, M.; Zhao, H. Energy-aware task assignment for mobile cyber-enabled applications in heterogeneous cloud computing. *J. Parallel Distrib. Comput.* **2018**, *111*, 126–135.
20. Gai, K.; Qiu, M.; Zhao, H.; Liu, M. Energy-Aware Optimal Task Assignment for Mobile Heterogeneous Embedded Systems in Cloud Computing. In Proceedings of the 2016 IEEE 3rd international conference on cyber security and cloud computing, Beijing, China, 25–27 June 2015.
21. Chen, X.; Pu, L.; Gao, L.; Wu, W.; Wu, D. Exploiting Massive D2D Collaboration for Energy-Efficient Mobile Edge Computing. *IEEE Wireless Communications*, **2017**, *24*, 64–71.
22. Souza, V.; Masip, X.; Marin-Tordera, E.; Ramírez, W.; Sanchez, S. Towards Distributed Service Allocation in Fog-to-Cloud (F2C) Scenarios. In Proceedings of the 2016 IEEE global communications conference (GLOBECOM), Washington, DC, USA, 4–8 December 2016.
23. Zhu, Q.; Si, B.; Yang, F.; Ma, Y. Task offloading decision in fog computing system. *China Commun.* **2017**, *14*, 59–68.

24. Pu, L.; Chen, X.; Xu, J.; Fu, X. D2D Fogging: An Energy-Efficient and Incentive-Aware Task Offloading Framework via Network-Assisted D2D Collaboration. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 3887–3901.
25. Aliyu, A.; Abdullah, H.; Kaiwartya, O.; Usman, M.; Abd Rahman, S.; Khatri, A. Mobile Cloud Computing Energy-aware Task Offloading (MCC: ETO); 2016; doi:10.1201/9781315364094-65.
26. Verma, J.K.; Kumar, S.; Kaiwartya, O.; Cao, Y.; Lloret, J.; Katti, C.P.; Kharel, R. Enabling Green Computing in Cloud Environments: Network Virtualization Approach Towards 5G Support. *Transactions on Emerging Telecommunications Technologies*, 2018, 29, e3434.
27. Yadav, R.; Zhang, W.; Kaiwartya, O.; Singh, P.; Elgendy, I.; Tian, Y.-C. Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing. *IEEE Access* **2018**, *6*, 55923–55936.
28. Yadav, R.; Zhang, W.; Li, K.; Liu, C.; Shafiq, M.; Karn, N. An adaptive heuristic for managing energy consumption and overloaded hosts in a cloud data center. *Wirel. Netw.* **2018**; doi: 10.1007/s11276-018-1874-1.
29. Vasile, M.-A.; Pop, F.; Tutueanu, R.-I.; Cristea, V.; Kołodziej, J. Resource-aware hybrid scheduling algorithm in heterogeneous distributed computing. *Future Gener. Comput. Syst.* **2015**, *51*, 61–71.
30. Satyanarayanan, M.; Bahl, P.; Caceres, R.; Davies, N. The case for VM-based cloudlets in mobile computing. *IEEE Pervasive Comput.* **2009**, *4*, 14–23.
31. Folkerts, E.; Alexandrov, A.; Sachs, K.; Iosup, A.; Markl, V.; Tosun, C. Benchmarking in the Cloud: What It Should, Can, and Cannot Be. In *Technology Conference on Performance Evaluation and Benchmarking*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 173–188.
32. Face-recognition. Available online: http://darnok.org/programming/face-recognition/ (accessed on 24 June 2019).
33. Health Application. https://github.com/mHealthTechnologies/mHealthDroid (accessed on 24 June 2019).
34. Application. Available online: https://powertutor.org/ (accessed on 24 *April* 2019).
35. Speed test. Available online: www.speedtest.net/ (accessed on 24 June 2019).
36. Khadija, A.; Gerndt, M.; Harroud, H. Mobile cloud computing for computation offloading: Issues and challenges. *Appl. Comput. Inf.* **2018**, *14*, 1–16.