IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Dynamic Energy Efficient Resource Allocation Strategy for Load Balancing in Fog Environment

Anees ur Rehman[1], Zulfiqar Ahmad[1], Ali Imran Jehangiri[1], Mohammed Alaa Ala'anzy[*,2], Mohamed Othman[*,2,3], Arif Iqbal Umar[1], Jamil Ahmad[1]

[1]Department of Information Technology, Hazara University, Mansehra, KPK, Pakistan
(aneesawan19@gmail.com, zulfiqarahmad@hu.edu.pk, ali_imran@hu.edu.pk, arifiqbalumar@yahoo.com, jamil@ieee.org)
[2]Department of Communication Technology and Networks, Universiti Putra Malaysia (UPM), Serdang 43400, Malaysia
[3]Laboratory of Computational Science and Mathematical Physics, Institute of Mathematical Research (INSPEM), Universiti Putra Malaysia (UPM), Serdang 43400, Malaysia

*Corresponding Authors: Mohammed Alaa Ala'anzy (m.alanzy.cs@gmail.com) and Mohamed Othman (mothman@upm.edu.my)

This work is supported by Universiti Putra Malaysia and the Ministry of Education Malaysia, (UPM journal publication fund 9001103).

ABSTRACT The Internet of Things is a flexible, emerging technology and an innovative development of the environmental trend. It is a large and complex network of devices in which fog computing plays a growing role in order to handle the information flow of such large and complex networks. Influence of their activities on carbon emissions and energy costs in unlimited results. Dynamic and efficient load balancing technology can be used to improve overall performance and reduce energy consumption. Load can be transferred or shared between computer nodes through load balancing technology. Therefore, the design of energy-efficient load balancing solutions for edge and fog environments has become the main focus. In this research work, we have proposed Dynamic Energy Efficient Resource Allocation (DEER) strategy for balancing the load in fog computing environments. In the presented strategy, initially the user submits tasks for execution to the Tasks Manager. Resource Information Provider registers resources from Cloud Data Centres. The information about the tasks and resources are then submitted to the Resource Scheduler. The resource scheduler arranges the available resources in descending order as per their utilization. The resource engine after receiving the information of tasks and resources from the resource scheduler assigns tasks to the resources as per ordered list. During execution of tasks, the information about the status of the resources is also sent to the Resource Load Manager and Resource Power Manager. The Resource Power Manager manages the power consumption through the resource On/Off mechanism. After successful execution of tasks, the resource engine returns the result to the user. Simulation results reveal that the presented strategy is an efficient resource allocation scheme for balancing load in fog environments to minimize the energy consumption and computation cost by 8.67 % and 16.77 % as compared with existing DRAM scheme.

INDEX TERMS: Internet of Things; load balancing; fog computing; energy efficiency; resource management.

I. INTRODUCTION

The "Internet of Things" (IoT) is a flexible, emerging technology and is one of the most advanced environmental trends in which over 50 billion things (e.g. sensors, mobile devices, and other computer nodes) are linked to the Internet by 2020 [3-5, 10]. Fog computing will play an increasing role in managing the information flow of such large and complex networks [2, 13]. The effect of their activities on carbon emissions and related costs of energy has unlimited consequences. Fog computing is a next level of cloud computing that provides applications and service to the network edge in a decentralized paradigm [1, 8, 17]. It is an IoT and edge computing platform [18-19]. Centralized and geo-distributed computing nodes allocate resources to IoT applications [38-40]. Choosing the appropriate computing node for each request is the responsibility of the resource scheduler and managers [47-50]. The computing nodes may be overloaded or under-loaded after assigning the tasks [7, 11]. Dynamic and efficient load balancing techniques can be used to increase overall performance and reduce energy consumption. Because "high-performance cloud infrastructure" consumes large amounts of energy, and eliminates carbon and heat emissions in the environment, green cloud

measures focus on efficient use of "cloud computing infrastructure" and minimizing power consumption [12, 42]. Consider the bulk of power consumption in a data center comes from computing processes, disk access, networking and cooling needs. Increase of power consumption in data centers contributes to higher operational costs as well as it has a severe impact on the environment [9, 25]. Therefore, energy saving techniques have become necessary because of their environmental and economic benefits [51]. In addition, it is easy to understand the relationship between workload style behaviour and random changes [37, 41]. The load can be transferred or shared between the computer nodes in load balancing techniques [20-24, 29]. The design of energy-efficient load balancing solutions for the edge-fog environment has therefore been focused.

In [6], a "Dynamic Resource Allocation Strategy" (DRAM) for balancing the load in fog computing is presented. The major objective of the paper is to obtain high load balancing in the fog and cloud platforms regarding all kinds of computing nodes. The DRAM procedure consists of four key steps, i.e. (a) partition fog operation, (b) spare node storage detection, (c) Dynamic source allocation for fog web subsets, and (d) globally allocation of resources based load-balancing. Although the DRAM strategy obtains great load balancing for all kinds of computer nodes in cloud and fog environments, however, the main issues of fog environment and cloud infrastructure were not considered by the researchers as energy usage and computing prices.

In our work we systematically inquired and solved the above mentioned challenges by presenting Dynamic Energy Efficient Resource Allocation (DEER) strategy for balancing the load in fog environment. In the presented strategy, initially the user submits tasks for execution to the Tasks Manager. Resource Information Provider register resources from Cloud Data Centers. The information about the tasks and resources are then submitted to the Resource Scheduler. The resource scheduler arranges the available resources in descending order as per their utilization. The resource engine after receiving the information of tasks and resources from the resource scheduler assigns tasks to the resources as per ordered list. During execution of tasks, the information about the status of the resources is also sent to the Resource Load Manager and Resource Power Manager. The Resource Power Manager manages the power consumption through the resource On/Off mechanism.

The main contributions of the proposed work are as follows:

- We have presented Dynamic Energy Efficient Resource Allocation (DEER) strategy for energy efficient resource scheduling and load balancing in cloud computing.
- In the proposed DEER strategy, the user will submit "n" numbers of tasks to Task Manager. It is assumed that for each task, the computational cost and energy consumption is predefined on the basis of the instructions that it contains.
- The Resource Information Provider (RIP) will register the "n" number of resources. It is also assumed that for each resource, the computational cost and energy consumption is predefined for each task on the basis of instructions contained in that task.
- The Resource Scheduler obtains information about the tasks from Tasks Manager. The tasks are sorted according to computational cost and energy consumption in ascending order S1<S2<S3... Sn.
- The Resource Scheduler obtains information about the resources from RIP. The available resources are sorted according to computational cost and energy consumption in descending order R1>R2>R3... Rn.
- The Resource Scheduler transmits the tasks and resource information to the Resource Engine.
- The Resource Engine assigns tasks to the resources as per sorted lists and starts execution and also shares the status of tasks and resources with Resource Load Manager.
- The Resource Load Manager examines the resource status during task execution, which transfers this status to the Resource Power Manager.
- The Resource Power Manager manages the resource on / off power status based on resource load status.
- The Resource Engine will compile the result and send the results to the user after successful execution of tasks.

The assumptions taken in the proposed work in respect of information about the tasks and resources are reasonable in reality. Because in cloud computing the resources in the form of virtual machines have predefined configurations for execution of tasks (since task is combination of multiple instructions) in terms of MIPS (Million Instructions Per Second), computational cost and energy consumption [26].

Rest of the paper is organized as follows: Section 2 presents the related work. Section 3 provides the system design and model. Section 4 presents evaluation methods including simulation tool, application modelling and performance evaluation parameters. Section 5 presents experimental setup, results and discussion. Finally, section 6 concludes the paper.

II. Related Work

The related work is reviewed in respect of cloud computing, fog computing, IoT applications, load balancing and simulation tools used for cloud and fog computing environments.

Cloud computing helps to control the next paradigm data centers and allows cloud deal earners to rent data center

resources based on client QoS (Quality of Service) criteria to deploy applications. Cloud applications have various requirements for composition, deployment and configuration. Measuring the performance of policies with respect to resource allocation and task scheduling schemes in more detail in cloud paradigm for various services models and applications under different loads, energy efficiency (heat dissipation, power consumption) and device size is a challenge to tackle [27]. Fog computing as a "distributed computing paradigm" extending to the corner of the web the facilities provided by the cloud. It allows the smooth use of cloud and corner incomes together with its own organization. It facilitates computer, networking and memory facilities administration and programming among data centers and end devices. Fog computing involves essentially parts of an application executing in the cloud and between last opinions. Sand the cloud devices, i.e. smart gateway sand routers [26]. Fog computing supports movement, diversity of asset and device, cloud interplay, and distributed information analytics to meet applications necessities that need low potential with large and dense geographic distribution. Fog computing benefits from both corner and cloud computing–while benefiting from the close proximity of edge devices to endpoints, it also leverages cloud resource scalability on demand [26].

In recent years, many smart devices and things, like wearable IoT devices, smartphone, manufacturing and office things, have been furnished with devices that can detect physical data in the environment in real time [30]. The implementation of the "Internet of Things" (IoT) is a concept according to which many smart instruments are linked via the Internet and supported by information analysis. Many IoT requests have been carefully read to advance everyday life, including smart conveyance, smart fitness, smart towns, and smart house [30]. Due to the large amount and speed of data flows produced by IoT devices, the cloud, which provides common and efficient computer resources, is an intelligent "head" for processing and storage of large data produced by spread IoT devices [31-32]. However, since the data stream produced by IoT devices is transmitted to a cloud data center via the Internet, the transmitted data can consume a large amount of bandwidth and power in the central network [33]. In contrast, remote clouds are often far from IoT nodes, so data flow delays may be too large, definitely for several responsive IoT applications [34]. So, we can reduce the load on traffic in the core network by using fog nodes that deliver computer devices to IoT devices and IoT manipulators [35-36]. In order to resolve the problem of balancing the load in fog environment [14], a dispersed IoT device association LoAd Balancing (LAB) strategy was developed that allocates IoT devices to the corresponding BS (Base Station) / Fog nodes to minimize the delay of all data streams. The BS constantly evaluates the traffic load and computational load in the circuit and sends the information. At the same time, for IoT devices, the corresponding BS can be selected at each repetition based on the assessed traffic load and the computational load of the BS / Fog node. In addition, it was shown that the proposed algorithm is convergent and efficient. In [6], a "Dynamic Resource Allocation Strategy" (DRAM) for balancing load in fog environments is presented. The major objective of the paper is to obtain maximum load balancing in the fog and cloud platforms for all kinds of computing nodes. The DRAM procedure consists of four key steps, i.e. (a) partition fog operation, (b) spare node storage detection, (c) Dynamic source allocation for fog web subsets, and (d) globally allocation of resources based load-balancing. Although the DRAM strategy obtains maximum load balancing for all kinds of computer nodes in cloud and fog platforms, however, the main issues of fog environment and cloud infrastructure were not considered by the researchers as energy usage and computing prices.

In [43], a GATS (Geography-Aware Task Scheduling) strategy was presented that schedules tasks geographically by considering spatial difference in green data centers. GATS adopts one of a queue model for analysis of green data centers. A random process was used with general process distribution for modeling of arriving process. The proposed strategy GATS collectively considers and utilizes spatial differences of several aspects such as the price of grid, active irradiation area of solar panels, solar radiation, on-site air density, maximum servers available in each green data center, wind speed, and rotor area of wind turbines. GATS was an optimal task scheduling strategy that solved the interior point method. GATS also optimally determines the allocation of tasks of all scheduled applications. Similarly, in [44], a dynamic mechanism was presented for allocation of heterogeneous resources on requests to various applications. The proposed mechanism was specifically designed for VCDC (virtualized cloud data center. The proposed approach utilizes as possible as minimum number of virtual machines to accommodate the current demand. The authors also tried to solve the problem of overhead between revenue and energy cost of VCDC.

In order to solve the issues of latency, deadline, availability of resources and bandwidth in fog environment, RTES (Real-Time Efficient Scheduling) strategy was presented in [45]. The major aim of the proposed RTES was to balance the load efficiently using the bandwidth available in fog environment. In [46], a simple Tabu search mechanism for load balancing optimally between the cloud and fog nodes was presented. The major aim of the proposed Tabu search mechanism is to compute and process the tasks received online.

In [15], a method of dynamic load balancing using a fog computer distributed system was proposed in which fog nodes can transfer computing tasks to neighboring nodes with available queue space based on the distribution of computing power and demand for tasks. Typically, load balancing procedures are based on an index or load index that produce an estimate of the node's workload relative to the global average. As such the said load catalog is used to identify load imbalances if the load catalog of one node is significantly upper or lesser than the load catalog of other

nodes. This shows that the CPU queue length is information about the remaining resource and can be used as the corresponding time-sharing workstation load index. In [35], a strategy was introduced for a multi-agent organization that implements the dynamic planning of smart devices. The Energy Internet model is based on the vertical setup of four main subsystems, namely, the architecture proposed in [36]: in which there are various levels i.e., perception level, network level, fog level and control level. The ultimate goal is to create a customizable power system that can perform intelligent energy planning in real time. The "CloudSim" [26] toolkit majorly supports modeling and building one or more practical technologies such as (VMs) over a "Data Center" simulated node, jobs, and then mapping them to appropriate VMs. Then it also permits several data center recreation to qualify a federation study and related VM migration policies for consistency and automatic application scaling [27]. New cloud claims, such as public web, business applications, game portals game portals scientific workflows, and content delivery, all work at maximum level of architecture. Real usage format of various real-world scenarios change over time and in most cases are unpredictable. These applications impose various requirements on the value of the package (QoS) depending on the user's time and interactive mode (online / offline).
In [27-28], basic CloudSim class abstraction enhanced and contributed to develop the iFogSim fog simulation environment. A simulation-based approach to testing the behavior of cloud computing systems and applications offers significant advantages, because cloud developers can fix quality bottlenecks before actually deploying the cloud for commercial use. Table 1 shows the brief summary of related work.

Table 1: An Overview of related work

Reference	Techniques	Evaluations Platform	Energy efficiency	Computational Cost	Features	Limitation
[6]	DRAM	CloudSim	×	✓	Dynamically balanced load	Not energy efficient
[14]	LAB	Mathematical Modeling	×	×	"Allocates IoT devices to the corresponding Fog nodes to minimize the delay"	Not energy efficient
[15]	Delay minimizing policy	Event Driven Simulation	×	×	"Fog-to-fog communication to reduce the service delay by sharing load"	The proposed work considers only latency parameter
[35]	Resource-efficient edge computing	Real world modeling	×	×	Managed resources for emerging intelligent IoT applications	Resource management only
[36]	"A multi-agent based flexible IoT edge computing architecture"	Real world modeling	×	×	Provide a flexible multi-agent edge computing architecture	Limited to the extent of architecture
[43]	GATS	Trae-driven simulation with realistic data	✓	✓	Schedule tasks geographically by considering spatial difference in green data centers	Task transmission time as overhead was not considered

III. System Design and Model

We have presented a Dynamic Energy Efficient Resource Allocation (DEER) Strategy for Load Balancing in Fog Environment. It is an effective load balancing scheme that allocates resources to the user on the bases of energy consumption and computational cost.
The proposed DEER strategy as shown in Figure 1 is work with following steps:

1) The user will submit "n" numbers of tasks to Task Manager. It is assumed that for each task, the computational cost and energy consumption is predefined on the basis of the instructions that it contains.
2) The Resource Information Provider (RIP) will register the "n" number of resources. It is also assumed that for each resource, the computational cost and energy consumption is predefined for each task on the basis of instructions contained in that task.
3) The Resource Scheduler obtains information about the tasks from Tasks Manager. The tasks are sorted according to computational cost and energy consumption in ascending order S1<S2<S3... Sn.
4) The Resource Scheduler obtains information about the resources from RIP. The available resources are sorted according to computational cost and energy consumption in descending order R1>R2>R3... Rn.
5) The Resource Scheduler transmits the tasks and resource information to the Resource Engine.
6) The Resource Engine assigns tasks to the resources as per sorted lists and starts execution and also shares the status of tasks and resources with Resource Load Manager.
7) The Resource Load Manager examines the resource status during task execution, which transfers this status to the Resource Power Manager.
8) Based on resource load status, the Resource Power Manager manages the resource on / off power status.
9) After successful execution of tasks, the Resource Engine will compile the result and send the results to the user.

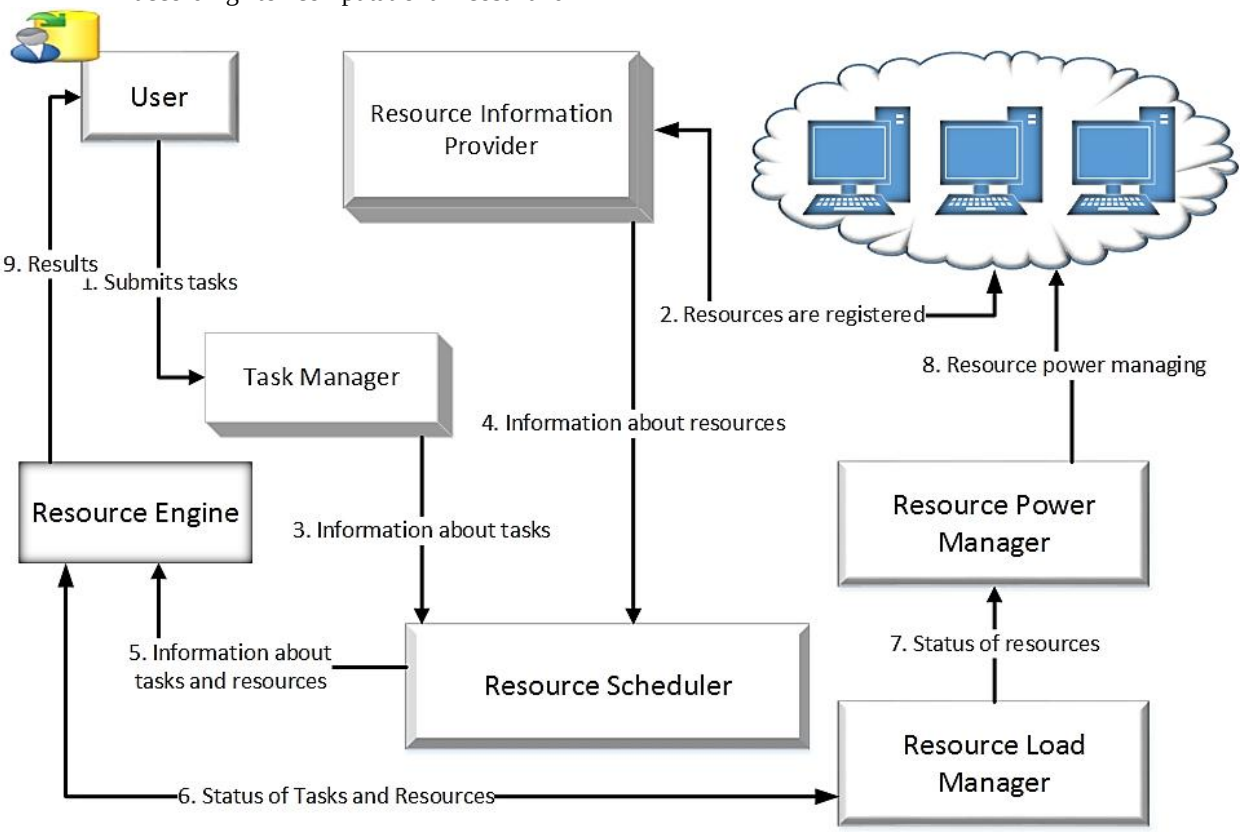


Figure 1: DEER Strategy

As reflected from Figure 1, in the proposed Dynamic Energy Efficient Resource Allocation (DEER) strategy, initially the user submits tasks for execution to the Tasks Manager. Resource Information Provider register resources from Cloud Data Centers. The information about the tasks and resources are then submitted to the Resource Scheduler. The resource scheduler arranges the available resources in descending order as per their utilization. The

resource engine after receiving the information of tasks and resources from the resource scheduler assigns tasks to the resources as per ordered list. During execution of tasks, the information about the status of the resources is also sent to the Resource Load Manager and Resource Power Manager. The Resource Power Manager manages the power consumption through the resource On/Off mechanism. After successful execution of tasks, the resource engine returns the result to the user.

a. Components of DEER Strategy

The DEER strategy has following components:

i. User

User is an entity that will submit tasks for execution. One or more users can submit tasks at the same time.

ii. Task Manager

Tasks Manager collects the tasks from the user and then submits to Resource Scheduler for scheduling purpose. It is the responsibility of the Task Manager to check the validity of submitted tasks. Task Manager also maintains the information of tasks regarding computational cost and energy consumption.

iii. Resource Information Provider

Resource Information Provider (RIP) not only registers the resources but also provides the information about available resources. The information also contains the computational cost and power consumption of the available resources.

iv. Resource Scheduler

Resource Scheduler obtains information about the tasks from Task Manager and resource from Resource Information Provider. Then Resource Scheduler sorts the submitted tasks according to computational cost and energy consumption in ascending order S1<S2<S3... Sn. Similarly, the Resource Scheduler sorts the available resources according to computational cost and energy consumption in descending order R1>R2>R3... Rn. Then the Resource Scheduler transmits the tasks and resource information to the Resource Engine.

v. Resource Engine

Resource Engine takes information about the resources and tasks from Resource Scheduler. The Resource Engine then assigns tasks to the resources as per sorted lists and starts execution. Resource Engine also shares the status of tasks and resources with Resource Load Manager. After successful completion of tasks, the Resource Engine also returns the results to the user.

vi. Resource Load Manager

The responsibility of Resource Load Manager is to examine the resource status during task execution. After examining the status, it is transferred to the Resource Power Manager.

vii. Resource Power Manager

Resource Power Manager on reception of status regarding the resources, manages the power through resource on / off power status.

b. Pseudo Code for proposed DEER strategy

The pseudo code for proposed DEER strategy is given below.

Algorithm 1: DEER Strategy

Input: T_t (T_t = Total No. of tasks submitted)
Output: R_g (R_g = Results generated)

```
1:  procedure DEER( )
2:      T_t ← GetTasks() from Task Manager                ▷ T_t = Total No. of Tasks
3:      T_r ← GetResources() from Resource Information Provider  ▷ T_r = Total Resources
4:      for each resource (R_1 to R_n) do
5:          R_u ← GetUtilization()
6:          Sort R_u in descending order for R_1 to R_n
7:          R1≥R2≥R3...Rn
8:          Min_u ← Integer.max          ▷ Min_u = Resource with minimum utilization
9:          R_a ← null                                   ▷ R_a = Resource assigned
10:         if (R_u ≤ Min_u) then
11:             Resource = T_i
12:             R_a = Resource
13:             R_u = Min_u
14:         end if
15:     end for
16:     for (all available resources) do
17:         R_p ← Get Power()                            ▷ R_p = Resource Power
18:         if (R_p == null) then
19:             Resource.PowerOff()
20:         else
21:             if (Task == INQUEUE) then
22:                 Resource.PowerOn()
23:             else
24:                 Continue
25:             end if else
26:         end if else
27:     end for
28:     Return Result R_g
29: end procedure
```

Algorithm 1 shows the overall procedure of DEER Strategy. Initially the user submits tasks for execution to the Tasks Manager. Resource Information Provider register resources from Cloud Data Centers. The information about the tasks and resources are then submitted to the Resource Scheduler. The resource scheduler arranges the available resources in descending order as per their utilization. The resource engine after receiving the information of tasks and resources from the resource scheduler assigns tasks to the resources as per ordered list. During execution of tasks, the information about the status of the resources is also sent to the Resource Load Manager and Resource Power Manager. The Resource Power Manager manages the power consumption through the resource On/Off mechanism. After successful execution of tasks, the resource engine returns the result to the user.

The proposed DEER strategy is an efficient and novel scheduling and load balancing approach. As to the best of our knowledge and reflected from literature review, none of the existing work specially work done in [6], have considered the main problem of fog environment and cloud infrastructure as energy usage and computing price. Whereas, in our proposed work, it is systematically inquired and solved by presenting Dynamic Energy Efficient Resource Allocation (DEER) strategy for balancing the load in fog environment. Moreover, the

complexity of the proposed algorithm is linear as in the proposed algorithm two separate "for loops" are used. The first "for loop" is used for scheduling and load balancing purposes, while the second one is used for minimization of energy consumption. The runtime overhead of the proposed algorithm DEER is O(n) and the same is better than the existing DRAM algorithm which is O(n2). It is because of that in the proposed algorithm two separate linear "for loops" are used, while in the existing algorithm nested "for loop" is being used. Therefore, the proposed algorithm has linear time complexity, while that of the existing one has quadratic time complexity.

The overhead of the proposed algorithm is sorting of all the resources on the basis of their usage and it is because of efficient utilization of resources. The same overhead is negligible in respect of the proposed algorithm as the proposed algorithm initially sorts the available resources and then the algorithm works dynamically. Whenever a new resource comes to the sorted pool of resources it will be placed at its proper location dynamically.

IV. Evaluation Methods

This section provides comprehensive details on simulation toolkit and application modeling.

a. Simulation Tool

To enable modeling and establish fog and cloud computing scenarios, we have used CloudSim [26]. It quantifies the performance of resource management policies in cloud environments for various application and service models under different load, energy performance, and size of systems. It is also capable of creating multiple nodes/VMs and data centers. It provides "modeling and simulation of cloud computing infrastructure at large scale", including a single physical computing node based data center. It is a standalone platform for "modeling data centers, scheduling, service brokers, and allocations schemes" [26-27].

The proposed work is simulation based work, in which the cloud computing simulator i.e., CloudSim [26] is used to evaluate the presented strategy DEER. Since, the significance of proposed work is evaluated through simulation on the basis of existing and most relevant published work DRAM [6], in which the similar CloudSim configurations were used. However, in the future we will test the proposed model with a real testbed and real datasets, while currently we are testing our algorithm using the existing algorithm datasets to prove the superior within the same datasets.

b. Application Modelling

In our work, we conducted a simulation in cloud computing simulator i.e., CloudSim [26] to evaluate the proposed strategy DEER. The intermediate "computing nodes" and the edge "computing nodes" are simulated as two "computing data centers". The fog resources implemented three kinds of computing nodes, i.e., (a) "the edge computing node", (b) "the intermediate node", and (c) the Processing Machines (PMs). The number of resources for each type of computing node contained in the four various datasets comprising 500, 1000, 1500 and 2000 fog resources. For example, when the number of fog services is 1000, "there are 324 fog services that need edge computing nodes, 353 fog services that need intermediate computing nodes, and 323 fog services that need PMs in the remote cloud for resource response". The results of the proposed DEER strategy is then compared with the existing published work DRAM [6].

c. Performance Evaluation Parameters

We have used "energy consumption and computational cost as performance evaluation parameters". The detail description of each parameter is given below:

i. Energy Consumption

Energy consumption is the total power consumed by the resources on execution of submitted tasks [16]. It is calculated in joules. Energy consumption of presented and existing strategies has been calculated with the help of Eq. (4.1).

$$Energy_{consumed} = \sum_{i=1}^{n} E_{transmision(i)} + E_{execution(i)} + E_{sensing(i)} \qquad Eq.(1)$$

From equation 1, it reflects that the total energy/power consumption is the sum of energy consumption on transmission, execution and sensing the each individual task.

ii. Computational Cost

Computational cost is the total cost consumed by the resources on execution of submitted tasks [16]. It is calculated in dollars. Computational cost of proposed and existing strategies has been calculated with the help of Eq. (2).

$$Cost_{computation} = \sum_{i=1}^{n} MIPS_{HOST(i)} \times TimeFrame_{HOST(i)} \times Cost_{HOST(i)} \qquad Eq.(2)$$

From equation 2, it is clear that the cost of computation is the sum of cost of MIPS for each host, time frame occupied by each host and the computational cost of each individual host.

V. Experiment Setup

The cloud computing simulator i.e., CloudSim [26] is used to evaluate the presented strategy DEER. The algorithm parameters are computed/selected based on fog and cloud

computing environment. The intermediate "computing nodes" and the edge "computing nodes" are simulated as two "computing data centers". The fog resources implemented three kinds of computing nodes, i.e., the edge "computing node", the intermediate node, and the "Processing Machines (PMs)". The number of resources for each type of computing node contained in the four various datasets comprising 500, 1000, 1500 and 2000 fog resources. For example, "when the number of fog services is 1000, there are 324 fog services that need edge computing nodes, 353 fog services that need intermediate computing nodes, and 323 fog services that need PMs in the remote cloud for resource response". It is because of that the significance of proposed work is evaluated through simulation on the basis of existing and most relevant published work DRAM [6]. Similarly, we considered the evaluation parameters as energy consumption and computational cost because our main objective is to reduce energy consumption and computational cost. The setting of evaluation parameters are made based on cloud and fog computing environment [26] and the same were evaluated through equation 1 and 2 [16].

a. RESULTS AND DISCUSSION

The number of resources for each kind of computing node consists of 4 different datasets containing 500, 1000, 1500 and 2000 fog resources. Our objective is to evaluate the performance evaluation parameters in terms of nodes variation, while the number of tasks remains constant. We have considered simulation time as 24 hours and one user is simulated. We evaluate each performance evaluation parameter repeatedly and take the average values with graphical analysis.

i. ENERGY CONSUMPTION

The energy consumption of proposed strategy i.e. DEER is 269862.1 Joule for 500 nodes, 525555.89 Joule for 1000 nodes, 772671.25 Joule for 1500 nodes and 1002513.97 Joule for 2000 nodes. Whereas, energy consumption of existing strategy i.e. DRAM is 294000.17 Joule for 500 nodes, 568594.17 Joule for 1000 nodes, 852921.67 Joule for 1500 nodes and 1099242.23 Joule for 2000 nodes.

The results for proposed policy compared with existing policy in terms of energy consumption are plotted in Figure 2, which reflects that the proposed strategy i.e. DEER consumed 8.67% less energy as compared with the existing DRAM strategy.

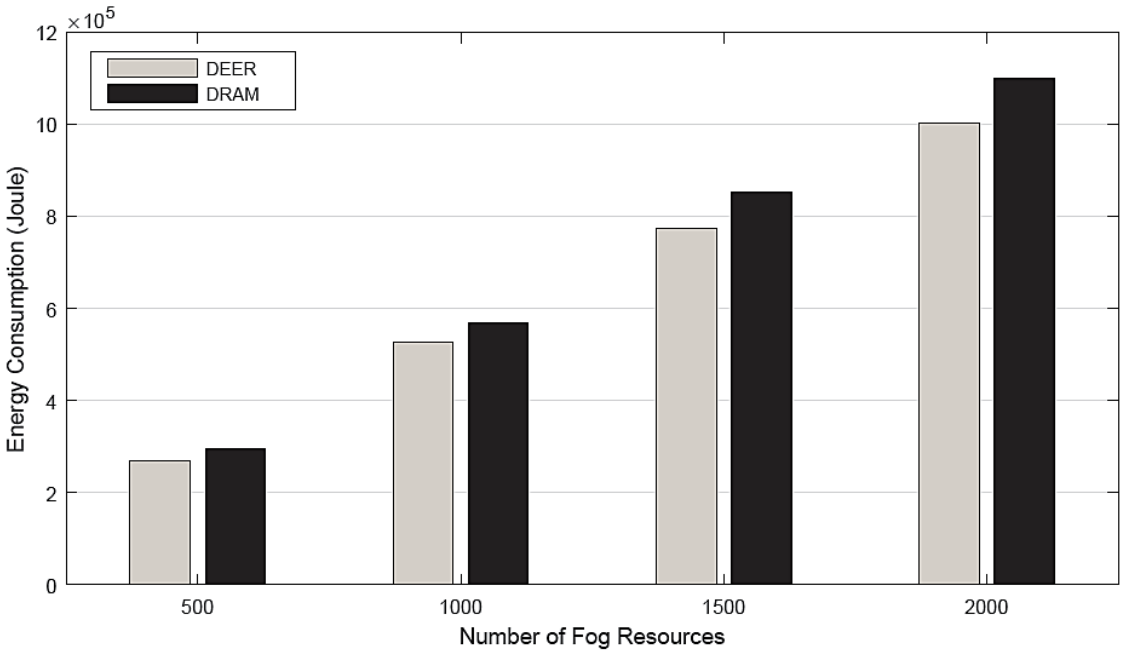


Figure 2: Energy Consumption of proposed DEER strategy compared with existing DRAM strategy

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3035181, IEEE Access

As reflected from Figure 2, the proposed DEER strategy consumed 8.67% less energy as compared with the existing DRAM strategy. It is because of that the proposed work not only balanced the load dynamically but also efficiently managed the resources through Power On/Off mechanism.

ii. Computational Cost

The computational cost of proposed strategy i.e. DEER is 19050 dollars for 500 nodes, 34155 dollars for 1000 nodes, 51195 dollars for 1500 nodes and 72810 dollars for 2000 nodes. Whereas, the computational cost of existing strategy i.e. DRAM is 24045 dollars for 500 nodes, 42435 dollars for 1000 nodes, 67740 dollars for 1500 nodes and 78705 dollars for 2000 nodes.

The results for proposed policy compared with existing policy in terms of computational cost are plotted in Figure 3, which shows that the proposed strategy i.e. DEER consumed 16.77% less computational cost as compared with the existing DRAM strategy

.

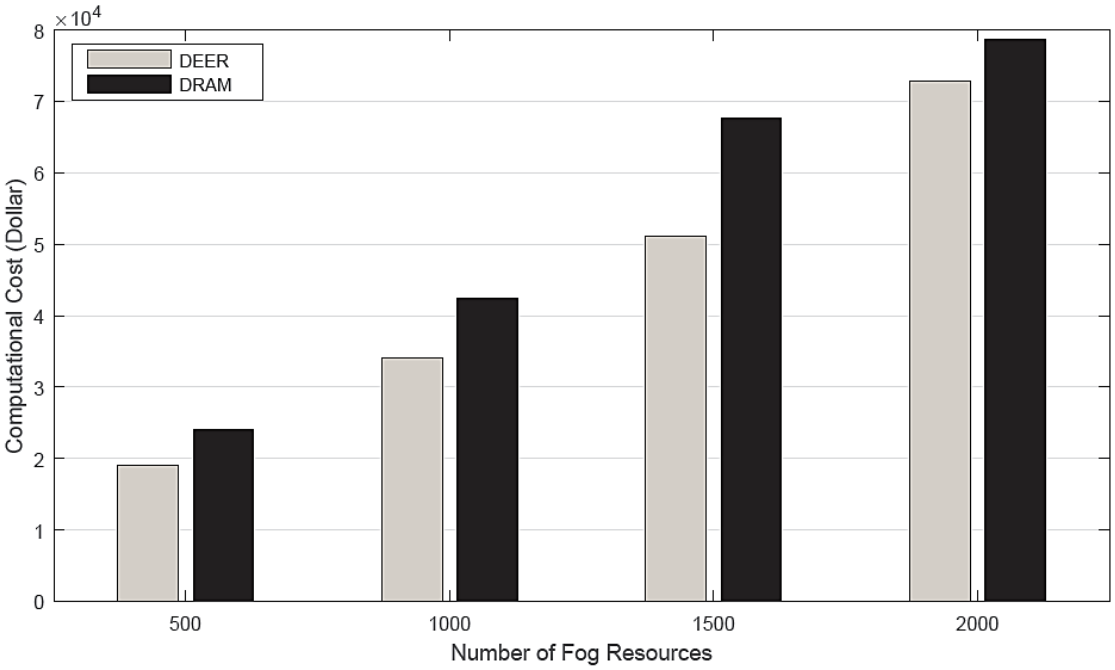


Figure 3: Computational Cost of proposed DEER strategy compared with existing DRAM strategy

As reflected from Figure 3, the proposed DEER strategy consumed 16.77% less computational cost as compared with the existing DRAM strategy. It is because of that the proposed work not only balanced the load dynamically but also efficiently managed the resources through Power On/Off mechanism.

VI. Conclusion

In this research work, we have proposed Dynamic Energy Efficient Resource Allocation (DEER) strategy for balancing load in fog environments. In the proposed strategy, initially the user submits tasks for execution to the Tasks Manager. Resource Information Provider register resources from Cloud Data Centers. The information about the tasks and resources are then submitted to the Resource Scheduler. The resource scheduler arranges the available resources in descending order as per their utilization. The resource engine after receiving the information of tasks and resources from the resource scheduler assigns tasks to the resources as per ordered list. During execution of tasks, the information about the status of the resources is also sent to the Resource Load Manager and Resource Power Manager. The Resource Power Manager manages the power consumption through the resource On/Off mechanism. After successful execution of tasks, the resource engine returns the result to the user. Simulation was performed in CloudSim simulation environment and the results of proposed DEER strategy is compared with existing DRAM strategy. Simulation results show that the proposed strategy is an efficient resource allocation strategy for load balancing in fog environments in order to reduce the energy consumption and computation cost by 8.67% and 16.77%.

In future work, we intend to propose fault-tolerant based dynamic energy-efficient resource allocation strategy in fog environments.

Acknowledgement: We would like to thank Universiti Putra Malaysia and the Ministry of Education Malaysia for supporting our work.

REFERENCES

[1] Velte, T., Velte, A., & Elsenpeter, R. (2009). *Cloud computing, a practical approach*. McGraw-Hill, Inc.

[2] Stojmenovic, I., & Wen, S. (2014, September). The fog computing paradigm: Scenarios and security issues. In *2014 federated conference on computer science and information systems* (pp. 1-8). IEEE.

[3] Khan, R., Khan, S. U., Zaheer, R., & Khan, S. (2012, December). Future internet: the internet of things architecture, possible applications and key challenges. In *2012 10th international conference on frontiers of information technology* (pp. 257-260). IEEE.

[4] Lee, I., & Lee, K. (2015). The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Business Horizons*, *58*(4), 431-440.

[5] Hendricks, Drew. "The Trouble with the Internet of Things." *London Data store. Greater London Authority. Retrieved* 10 (2015).

[6] Xu, Xiaolong, et al. "Dynamic resource allocation for load balancing in fog environment." *Wireless Communications and Mobile Computing* 2018 (2018).

[7] Kong, Yan, Minjie Zhang, and Dayong Ye. "A belief propagation-based method for task allocation in open and dynamic cloud environments." *Knowledge-Based Systems* 115 (2017): 123-132.

[8] Sarkar, Subhadeep, Subarna Chatterjee, and SudipMisra. "Assessment of the Suitability of Fog Computing in the Context of Internet of Things." *IEEE Transactions on Cloud Computing* 6.1 (2015): 46-59.

[9] Peng, Kai, et al. "Link importance evaluation of data center network based on maximum flow." *Journal of Internet Technology* 18.1 (2017): 23-31.

[10] Luo, Entao, et al. "Privacyprotector: Privacy-protected patient data collection in IoT-based healthcare systems." *IEEE Communications Magazine* 56.2 (2018): 163-168.

[11] Li, Peng, Shengli Zhao, and Runchu Zhang. "A cluster analysis selection strategy for supersaturated designs." *Computational Statistics & Data Analysis* 54.6 (2010): 1605-1612.

[12] Tian, Guo-Liang, Mingqiu Wang, and Lixin Song. "Variable selection in the high-dimensional continuous generalized linear model with current status data." *Journal of Applied Statistics* 41.3 (2014): 467-483.

[13] Baek, Jung-yeon, et al. "Managing Fog Networks using Reinforcement Learning Based Load Balancing Algorithm." *arXiv preprint arXiv:1901.10023* (2019).

[14] Fan, Qiang, and Nirwan Ansari. "Towards workload balancing in fog computing empowered IoT." *IEEE Transactions on Network Science and Engineering* (2018).

[15] Yousefpour, A., Ishigaki, G., & Jue, J. P. (2017, June). Fog computing: Towards minimizing delay in the internet of things. In *2017 IEEE international conference on edge computing (EDGE)* (pp. 17-24). IEEE.

[16] Mustafa, S., Nazir, B., Hayat, A., & Madani, S. A. (2015). Resource management in cloud computing: Taxonomy, prospects, and challenges. *Computers & Electrical Engineering*, *47*, 186-203.

[17] Bonomi, Flavio, Rodolfo Milito, Preethi Natarajan, et al. *Fog Computing : A Platform for Internet of Things and Analytics*. pp. 169–86, doi:10.1007/978-3-319-05029-4.

[18] Bonomi, Flavio, Rodolfo Milito, Jiang Zhu, et al. *Fog Computing and Its Role in the Internet of Things*. no. March, 2014, pp. 2–5, doi:10.1145/2342509.2342513.

[19] Puthal, Deepak, et al. *Secure and Sustainable Load Balancing of Edge Datacenters in Fog Computing*. no. May, 2018, doi:10.1109/MCOM.2018.1700795.

[20] Qian, Chen. *SDLB : A Scalable and Dynamic Software Load Balancer for Fog and Mobile Edge Computing*. pp. 55–60.

[21] Vaquero, Luis M., et al. *Finding Your Way in the Fog : Towards a Comprehensive Definition of Fog Computing Abstract : Finding Your Way in the Fog : Towards a Comprehensive Definition of Fog Computing*. 2014.

[22] Sharma, S., & Saini, H. (2019). A novel four-tier architecture for delay aware scheduling and load balancing in fog environment. Sustainable Computing: Informatics and Systems, 24, 100355.

[23] Wan, Jiafu, et al. *Fog Computing for Energy-Aware Load Balancing and Scheduling in Smart Factory*. Vol. 14, no. 10, 2018, pp. 4548–56, doi:10.1109/TII.2018.2818932.

[24] Baccarelli, Enzo, et al. “Fog of Everything : Energy-Efficient Networked Computing Architectures , Research Challenges , and a Case Study.” *IEEE Access*, vol. 5, IEEE, 2017, pp. 9882–910, doi:10.1109/ACCESS.2017.2702013.

[25] Gia, Tuan Nguyen, et al. Fog Computing in Body Sensor Networks : An Energy Efficient Approach Fog Computing in Body Sensor Networks : An Energy Efficient Approach. no. January, 2015.

[26] Buyya, Rajkumar, et al. “Modeling and Simulation of Scalable Cloud Computing Environments and the Cloudsim Toolkit: Challenges and Opportunities.” *Proceedings of the 2009 International Conference on High Performance*

Computing and Simulation, HPCS 2009, 2009, pp. 1–11, doi:10.1109/HPCSIM.2009.5192685.

[27] Gupta, Harshit, et al. "IFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in the Internet of Things, Edge and Fog Computing Environments." *Software - Practice and Experience*, vol. 47, no. 9, 2017, pp. 1275–96, doi:10.1002/spe.2509.

[28] Mahmud, Redowan, and Rajkumar Buyya. "Modeling and Simulation of Fog and Edge Computing Environments Using IFogSim Toolkit." *Fog and Edge Computing*, 2019, pp. 433–65, doi:10.1002/9781119525080.ch17.

[29] Mostinckx, Stijn, et al. "Mirror-Based Reflection in Ambienttalk." *Software - Practice and Experience*, vol. 39, no. 7, 2009, pp. 661–9

[30] S. S. Roy, D. Puthal, S. Sharma, S. P. Mohanty, and A. Y. Zomaya, "Building a sustainable Internet of Things: Energy-efficient routing using low-power sensors will meet the need," IEEE Consumer Electronics Magazine, vol. 7, no. 2, pp. 42–49, March 2018.

[31] W. Bao, D. Yuan, Z. Yang, S. Wang, W. Li, B. B. Zhou, and A. Y. Zomaya, "Follow me fog: Toward seamless handover timing schemes in a fog computing environment," IEEE Communications Magazine, vol. 55, no. 11, pp. 72–78, Nov. 2017.

[32] H.-L. Truong and S. Dustdar, "Principles for engineering IoT cloud systems," IEEE Cloud Computing, vol. 2, no. 2, pp. 68–76, 2015.

[33] L. Wang and R. Ranjan, "Processing distributed internet of things data in clouds," IEEE Cloud Computing, vol. 2, no. 1, pp. 76–80, 2015.

[34] M. Ogura and V. M. Preciado, "Stability of spreading processes over time-varying large-scale networks," IEEE Transactions on Network Science and Engineering, vol. 3, no. 1, pp. 44–57, Jan 2016.

[35] X. Chen, Q. Shi, L. Yang, and J. Xu, "Thriftyedge: Resourceefficient edge computing for intelligent IoT applications," IEEE Network, vol. 32, no. 1, pp. 61–65, Jan 2018.

[36] T. Suganuma, T. Oide, S. Kitagami, K. Sugawara, and N. Shiratori, "Multiagent-based flexible edge computing architecture for IoT," IEEE Network, vol. 32, no. 1, pp. 16–23, Jan 2018.

[37] Thilagavathi, N., Dharani, D. D., Sasilekha, R., Suruliandi, V., & Uthariaraj, V. R. (2019). Energy Efficient Load Balancing in Cloud Data Center Using Clustering Technique. International Journal of Intelligent Information Technologies (IJIIT), 15(1), 84-100.

[38] Sookhak, M., Yu, F. R., Khan, M. K., Xiang, Y., & Buyya, R. (2017). Attribute-based data access control in mobile cloud computing: Taxonomy and open issues. Future Generation Computer Systems, 72, 273-287.

[39] Mahmud, R., Kotagiri, R., & Buyya, R. (2018). Fog computing: A taxonomy, survey and future directions. In Internet of everything (pp. 103-130). Springer, Singapore.

[40] Ray, P. P. (2018). A survey on Internet of Things architectures. Journal of King Saud University-Computer and Information Sciences, 30(3), 291-319.

[41] Mishra, S. K., Sahoo, B., & Parida, P. P. (2020). Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences, 32(2), 149-158.

[42] Botta, A., De Donato, W., Persico, V., & Pescapé, A. (2016). Integration of cloud computing and internet of things: a survey. Future generation computer systems, 56, 684-700.

[43] Yuan, H., Bi, J., & Zhou, M. (2020). Geography-Aware Task Scheduling for Profit Maximization in Distributed Green Data Centers. IEEE Transactions on Cloud Computing.

[44] Bi, J., Yuan, H., Tan, W., Zhou, M., Fan, Y., Zhang, J., & Li, J. (2015). Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center. IEEE Transactions on Automation Science and Engineering, 14(2), 1172-1184.

[45] Verma, M., Bhardwaj, N., & Yadav, A. K. (2016). Real time efficient scheduling algorithm for load balancing in fog computing environment. Int. J. Inf. Technol. Comput. Sci, 8(4), 1-10.

[46] Téllez, N., Jimeno, M., Salazar, A., & Nino-Ruiz, E. (2018). A tabu search method for load balancing in fog computing. Int. J. Artif. Intell, 16(2).

[47] Yuan, H., & Zhou, M. (2020). Profit-Maximized Collaborative Computation Offloading and Resource Allocation in Distributed Cloud and Edge Computing Systems. *IEEE Transactions on Automation Science and Engineering*.

[48] Yuan, H., Zhou, M., Liu, Q., & Abusorrah, A. (2020). Fine-grained resource provisioning and task scheduling for heterogeneous applications in distributed green clouds. *IEEE/CAA Journal of Automatica Sinica*, *7*(5), 1380-1393.

[49] Anglano, C., Canonico, M., & Guazzone, M. (2018, June). Profit-aware resource management for edge computing systems. In *Proceedings of the 1st International Workshop on Edge Systems, Analytics and Networking* (pp. 25-30).

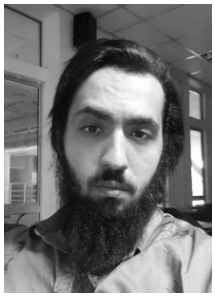
[50] Bitam, S., Zeadally, S., & Mellouk, A. (2018). Fog computing job scheduling optimization based on bees swarm. *Enterprise Information Systems*, *12*(4), 373-397.

[51] Yuan, H., Liu, H., Bi, J., & Zhou, M. (2020). Revenue and Energy Cost-Optimized Biobjective Task Scheduling for Green Cloud Data Centers. *IEEE Transactions on Automation Science and Engineering*.

Anees ur Rehman is MS Scholar at Department of Information Technology Hazara University Mansehra, KPK, Pakistan. His research interest is Fog computing, Cloud computing, high Performance Computing, and Internet of Things.

Zulfiqar Ahmad is currently pursuing a PhD in Computer Science at Department of Information Technology, Hazara University, Mansehra, KPK, Pakistan. He has received his MSc (Computer Science) degree with distinction from Hazara University Mansehra in 2012 and MS (CS) degree from COMSATS University, Abbottabad, KPK, Pakistan in 2016. His research interest is Fog computing, Cloud computing, high performance computing, and Scientific Workflows execution and management.

Ali Imran Jehangiri is a lecturer in the Department of Information Technology, Hazara Universtiy, Mansehra, Pakistan. He graduated from Bergische University Wuppertal Germany in 2010. He received Ph.D. degree in Computer Science from the Georg-August-Univesity Goettingen, Germany in 2015. He gained industrial experience with Service Computing working as research assistant at GWDG. He is involved in research activities dealing with parallel, Grid computing, Cloud computing and Big data. He is author of several publications in international journals, and conferences.

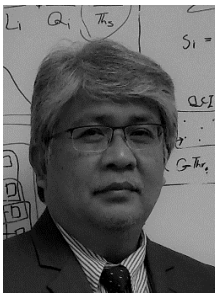


Mohammed Alaa Ala'anzy

Received his Master's degree in computer science from University Putra Malaysia, in 2017.

He is currently pursuing a Ph.D. with the Faculty of Computer Science and Information Technology, University Putra Malaysia. His current research interests include cloud computing, green computing, load balancing, task scheduling, and fog computing. He has authored several high reputed journal/conference papers and a reviewer in Scientific.Net journal.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3035181, IEEE Access



Mohamed Othman

Received his Ph.D. (Hons.) from the National University of Malaysia. He is currently a Professor in computer science with the Department of Communication Technology and Networks, Universiti Putra Malaysia (UPM). Prior to that he was Deputy Director of the Information Development and Communication Centre, where he was in charge of the UMPNet network campus, uSport Wireless Communication Project, and the UPM Data Center. He is also an associate researcher and a Coordinator of high-speed machines with the Laboratory of Computational Science and Informatics, Institute of Mathematical Science, UPM. In 2017, he received an Honorary Professorship from SILKWAY International University (formerly known as South Kazakhstan Pedagogical University), Shymkent, Kazakhstan, and was also a Visiting Professor with South Kazakhstan State University, Shymkent, and the L. N. Gumilyov Eurasian National University, Astana, Kazakhstan. He has published more than 300 International journals and 330 proceeding papers. His main research interests are computer networks, parallel and distributed computing, high speed interconnection networks, network design and management (network security, wireless and traffic monitoring), consensus in IoT, and mathematical models in scientific computing. He is a senior member of IEEE, and a Life Member of the Malaysian National Computer Confederation, and the Malaysian Mathematical Society. He was awarded the Best PhD thesis in 2000 by Sime Darby Malaysia and the Malaysian Mathematical Science Society. He has also filed six Malaysian, one Japanese, one South Korean, and three U.S. patents.

Arif Iqbal Umar earned his MSc (Computer Science) degree from University of Peshawar Pakistan and PhD (Computer Science) degree from BeiHang University (BUAA) Beijing P.R. China. His research interests include Data Mining, Machine Learning, Information Retrieval, Digital Image Processing, Computer Networks Security and Sensor Networks. He has supervised 07 PhD candidates and 34 MS candidates. He is author of more than 70 research publications in the leading research journals and conferences. He has at his credit 27 years' experience of teaching, research, planning and academic management. He is working as Associate Professor of Computer Science in the department of Information Technology Hazara University Mansehra. He is leading the Department as Head. arifiqbalumar@Yahoo.com & drarif@hu.edu.pk

Jamil Ahmad (Senior Member, IEEE) received the M.Sc. degree in information technology from the University of Warwick, U.K., the M.Sc. degree in computer science from the University of Peshawar, and the Ph.D. degree in artificial neural network from the Department of Electrical and Electronics Engineering, King's College London, U.K. He is currently holding the position of the Vice Chancellor at the Hazara University Khyber Pakhtunkhwa, Pakistan. He is a Fellow of the British Computer Society, a Chartered Engineer (CEng), and a member of IET and ACM. He received grants for his research and academic projects from various organizations, including the National Information and Communication Technology (ICT) Research and Development Fund, Pakistan, the HEC-British Council Linkage Program, and the U.K. Government under PMI 2 Program. He was a recipient of the Charles Wallace Pakistan Trust Visiting Fellowship, U.K., from 2012 to 2013, and the International Visiting Leadership Program (IVLP) Fellowship, from August 2007 to September 2007, USA.