I J C T A, 9(40) 2016, pp. 237-245
© International Science Press

Optimizing Execution Delay in Mobile Cloud Computing

Sambit Kumar Mishra, Ranith Sardar, Bibhudatta Sahoo and Sanjay Kumar Jena

ABSTRACT

Getting quick response in mobile cloud computing is a newly developing area for research. Delay increases when the user requests (tasks) execute in the cloud environment instead of the mobile device, and it affects the quality of service also. So, cloudlet was introduced to execute the maximum number of tasks that provides the same infrastructure like cloud computing system. We have proposed an algorithm in this paper to optimize the execution delay of mobile applications. To get revolved from selecting same Virtual Machines from the data center, we use selection procedure as Best fit policy. In the proposed technique, if the specified data center (cloudlet) couldn't provide enough resources, then only the user request will be sent to the cloud system. As a result, it clearly showed in the simulation results that the execution time for each task had decreased after using cloudlet.

Keywords: Cloud Computing, Cloudlet, Latency, MCC, Task allocation, VM.

I. INTRODUCTION

In twenty century, the Mobile devices (smartphones, tablets) become an essential object of human life. Billions of global users are connected to the network and various information services through i-Phone, Android phones, Google Glasses, and other mobile devices. In the current report of Cisco Visual Networking Index in 2014, use of mobile data traffic has been increased by 69 percent, so it shows that mobile devices generate more traffic than other products [1]. Like users are connected with this world through Internet, social media, news, health-care, GPRS, and games. So, the quick revolution of mobile computing becomes an attacking field in the world of IT technology as well as trade and industry fields. However, the resources, battery life, computing capability, bandwidth, and storage size are still constant. Thus, a new gap is coming between the requirement of new generation mobile devices, and limited capacity of resources on mobile devices. To sort the problem of mobile computing, the mobile devices have to take the help of developed resources afforded by cloud computing [5, 8]. These cloud services are all about an offloading of a task, i.e., computationally expensive data from mobile devices to cloud system over wireless channels.

1.1. Mobile Cloud Computing

Simply, Mobile Cloud Computing (MCC), introduces an architecture where both data storage and data processing occur outside of the mobile device as shown in Figure 1. For the execution of cloud applications on Mobile cloud (MC), the computing power, and the data storage capability move away into the cloud services, bringing those applications and MC not to just smartphone users but a much extensive range of mobile subscribers.

According to the architecture of the MCC, the mobile devices are attached to the mobile networks through base stations (e.g., BTS (base transceiver station), satellite and access point) which make the connection between the network and customers. Mainly, mobile users send the request to the central processes

* Dept. of CSE, Nit Rourkela, India, *E-mails: skmishra.nirkl@gmail.com; ranithsardar.90@gmail.com; bibhudatta.sahoo@gmail.com; skjena@nitrkl.ac.in*

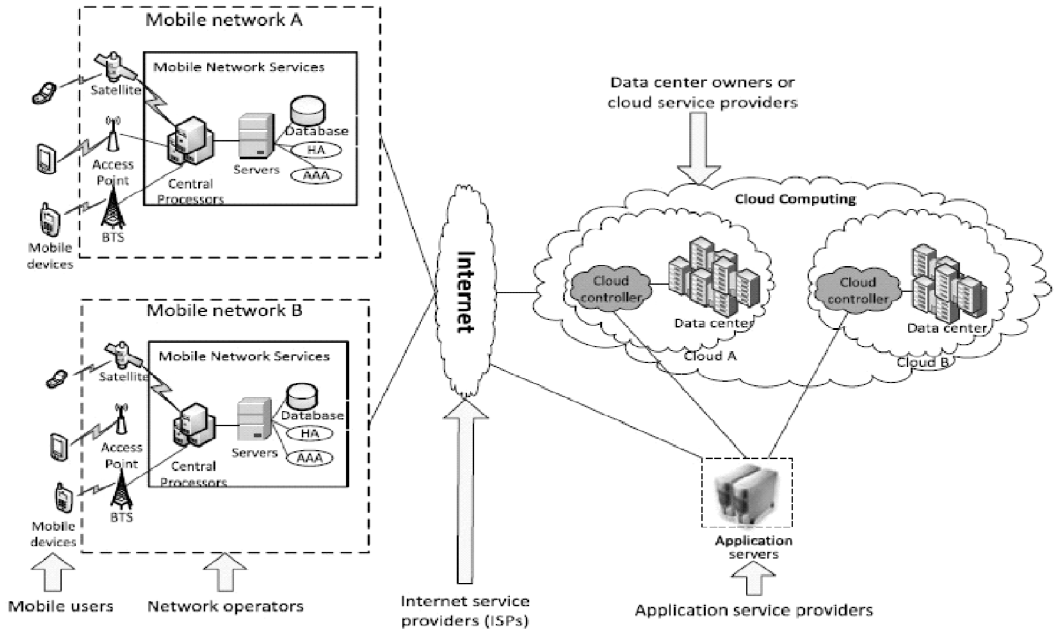

Figure 1: Frame work of Mobile Cloud Computing [1]

with ID and location, which connects the server by providing mobile network services. If users are already authenticated and authorized according to the home agent and subscriber's data stored in databases, then mobile network operator will deliver the service to the user. Then, the subscriber requests are transmitted to a cloud over the Internet. The cloud controllers process the user requests so that the mobile users can receive their corresponding services.

Cloud Computing (CC): CC is a large-scale distributed network system, consists of data centers, and each data center has a finite number of servers. Software-as-a-Service (SaaS), Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS) are part of CC [2].

Data center: In data center lab, all servers are connected with high connectivity to give the services, to process and distribute a large amount of data.

1.2. Benefits of MCC

CC is an ideal solution for mobile computing because of various reasons such as the system mobility, communication, and portability. In short, we will explain how cloud computing helps MC to overcome the obstacle.

Improve battery lifeline: In this generation, a battery is one of the prime concerns for mobile devices. So, computation offloading is proposed to transfer the computational task from mobile device to cloud. This helps mobile devices from consuming a large amount of power. If it happens on mobile devices, then it will take long execution time so that the battery consumption increases. Rudenko et al. [3] and Smailagic and Ettus [4] have proposed some efficient offloading techniques through different experiments.

Improving storage size and processing power: Storing capacity is also a significant barrier for mobile devices. MCC is developed to overcome this problem and store data in the cloud through the wireless network. One of the examples is the Amazon Simple Storage Service that provides file storage service. MCC also offers the user for more processors to solve it very fast even though mobile device's processing power is limited.

Improving reliability: An adequate way to enhance the reliability is shifting data or running application on a cloud as the data, and tasks are stored and processed by a group of computer systems. This process reduces the possibility of losing the data and application on the mobile devices. So, MCC is a most extensive data security model for both user and cloud service providers.

1.3. Limitation of MCC

The most useful solution of mobile resources is leverage CC. In MCC, all cell phones could execute an asset concentrated application on a far off elite process server or register bunch and bolster thin-customer client connections with the application over the Internet. Incidentally, high latencies (WAN) create huge problems [6]. So, total round trip time (RTT) is the biggest factor for getting the result. In a survey it has been proved that 50 millisecond (ms) difference in RTT can make a difference around 530 ms in downloading time.

1.4. Problem for Latency

Unfavorably, the present direction of Internet advancement makes it impossible that these crucial contemplations will change within a reasonable time-frame. To deal with those networks problems, we should take care of network security, the bandwidth of WAN, energy efficiency, and manageability. Firewalls and overlay systems, for instance, both accomplish their objectives by expanding the product way length that bundles must navigate. In wireless networks, the mobile devices transceiver turned on for a small period when it finds the possibility of receiving and acknowledging packets that have been buffered at a base station, through this technique energy can be saved. It increases average end to end packet latency performance. Though bandwidth will keep on improving over time, latency is a serious issue to improve the performance.

1.5. Motivation

The offloading of the task from mobile devices to a cloud system is not the best solution. The latency can take enough time in getting the service response. So, rather than depending on a far-off cloud, we might be able to address mobile devices resource poverty through a nearby powerful resource data-center (cloudlet). In this way, we could solve the problem of higher latency. The response will be faster if we use a local data-center, and also the propagation time will be lesser if the data center is closer to the mobile devices.

Although a cloudlet is a solution of low latency, it also has some fundamental drawbacks. As the cloudlet resources are not as rich as that is in an effective cloud, if all tasks queued up to a single virtual machine of a cloudlet, then the cloudlet may come up short on its assets rapidly. Especially, when many tasks are performed simultaneously, then the cloudlet becomes overloaded. This motivate us to find out a solution of overloading.

1.6. Contributions

We have proposed an efficient algorithm for online user requests into the MCC environments with an objective to improve the execution time of all requests as well as reduce the latency time, where each task can be taken as a demand resource vector, and each component of the vector is remarked as a demand by the user request. The main goal of this work is as follows: using scheduling policy in allocating the resources and setup an environment where we apply our logic. For the development of the algorithm to overcome the latency problem, we use the best-fit algorithm for choosing respective virtual machines (VMs) of cloudlet, and do first come first serve (FCFS) scheduling within the VMs. Through this procedure we will get total execution time for all task whether it is using cloud or using cloudlet [11]. If cloudlet is unable to give the response for the first time slot as required resources are not available, then the request will send to the cloud. The architecture of cloudlet is shown in Figure 2.

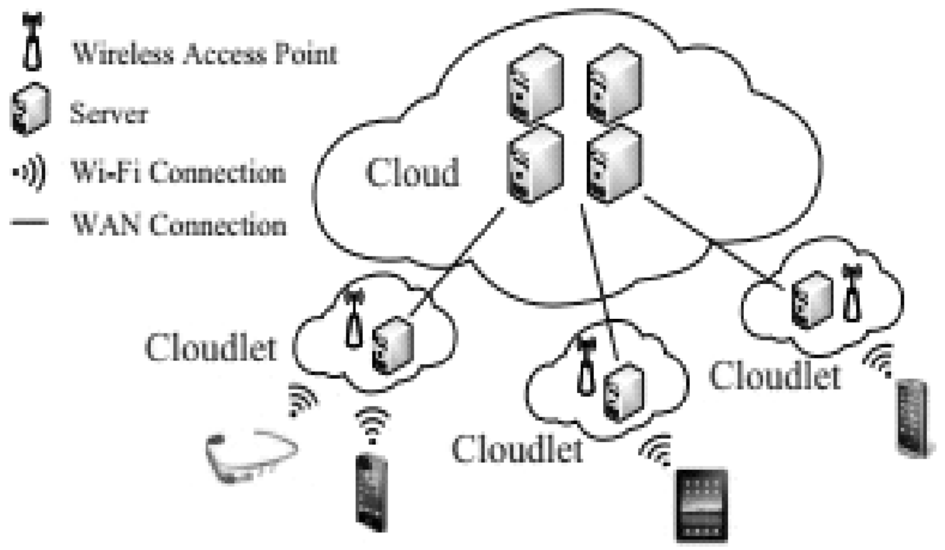

Figure 2: Architecture of cloudlet [13]

The remaining portion of this paper is presented as follows. Section 2 explains the related work. The next section gives the details about system model followed by the problem statement. In section 5, the algorithm is been discussed. The performance evaluation is given in section 6 and is concluded in section 7.

II. RELATED WORK

Cloud computing is gaining a lot of interest in diverse domains such as disaster management, healthcare, military [19, 20, 21]. Still, there is a scope to contribute security solutions to protect data in the cloud and intermediate edge data center [19]. In the current era, data turns to a new term as big data, which need unique solutions to support 4V's properties to process the big data in the cloud [20, 21]. In this generation, most of the people are using smartphones, laptops, tablets, i-pad, etc. rather than desktop systems. Everyone wants to have high-speed smartphones, but for the development, researchers are facing a lot of problems. Various network topologies have been proposed for its solutions. CC offers new service to provide pay per use offer to the users [14, 15]. How to allocate the resource and migrate, is explained in [16]. Using the logic of virtual memory, an energy saving scheduling is proposed in the cloud environment to allocate, relocate and cancel resources dynamically [17]. The allocation of the task using metaheuristic techniques is explained in [22], and the execution of the real-time task in the cloud environment is illustrated in [26]. Utilizing MCC in the application preparation and capacity left to the cloud rather than a cell phone.

The resource is a rule part that truly limits the class of uses that can be continue running on phones. In this paper, the vision of convenient figuring has been broken of this focal prerequisite. In this vision, adaptable customers faultlessly utilize near to PCs to get the advantages of cloud enlisting without incurring WAN deferrals. As opposed to depending on a distant cloud, a versatile client instantiates a cloudlet on close-by framework furthermore, utilizes it employing a remote LAN [5]. Li and Wang [7] explained the support of mobile applications with the help of mobile cloudlet. Especially, they investigate on the cloudlet capacity, cloudlet nodes lifetime, and resource reachable time.

In another method, cell phones send their demand to the closest cloudlet, if it can't give the reaction then the demand will return to the advanced mobile phones, least inactivity to execute the assignment among the storage room datacenters. Control utilization and defer issue can be lessened roughly by 29-32% than cloud-based offloading [10, 13].

III. SYSTEM MODEL

In the proposed system model, all cell phones are connected to the mobile networks via access point, then it directly sends the request to next platform (cloudlet). All cloudlets have some connection with several Cloud system as shown in Figure 3.

We have classified the system model into several parts and have given the model definition as follows.

3.1. Task Model

A mobile application (task) is represented by a sequence of tasks with linear topology. Here, we are offloading all the application from the mobile devices to the cloudlets. There are '**N**' tasks and each task has four tuples, i.e., $\theta_k = \{\mu_k; \alpha_k, \beta_k; ¥_k\}$, where $1 \le k \le N$, $\mu_k; \alpha_{k,} \beta_k; ¥_k$ are the parameter of each task. Especially

μ_k is the ram size needed to complete the task in terms of Mega Byte (MB).

θ_k is the number of instruction the CPU can execute at each second in terms of Million Instructions Per Second (MIPS),

β_k is the number of processing element need to execute the job,

$¥_k$ is the size of task in terms of Million Instructions (MI).

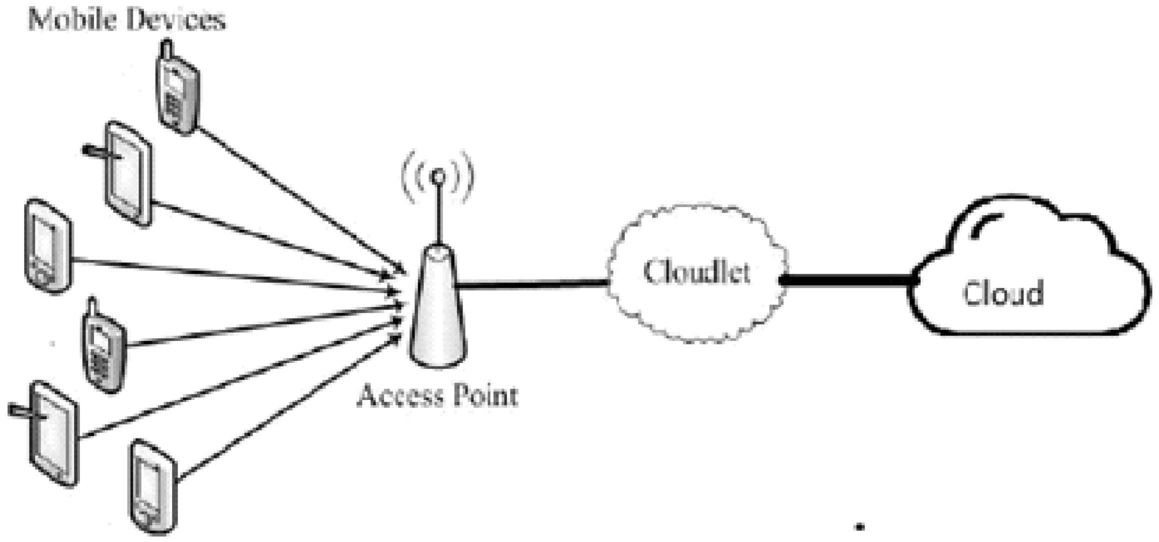

Figure 3: Connection protocol of System

3.2. Central Queue

As the cloud is always far from the mobile users, it is not the right choice to send all requests to the cloud. This distance increases the execution delay and to reduce the long delay, we are offloading tasks from the mobile devices to the cloudlet, and those are trusted, resource-rich servers (datacenters). The cloudlet then receives newly arriving tasks into the central queue. Each new task arriving at the cloudlet is inserted to the central queue.

3.3. VM's of cloudlet

Every cloudlet consists of data centers and multiple virtual machines are the part of each data centers [8, 9]. Let, M be the total number of virtual machines, and $< C_p, I_p, PE_p >$ are the parameters of each virtual machines, where $1 \le p \le M$.

$C < t >=< C_1, C_2, C_{3,\ldots,} C_p, \ldots, C_M > C_p$ be the capacity of a resource for all p. Let, $H < t >=< H_1, H_2, H_3, \ldots, H_k, \ldots, H_N >$ be the amount of resources already admitted by user requests at the time slot t.

Let, $A < t >=< A_1, A_2, A_3, \ldots, A_p, \ldots, A_M >$ be the vector of available resources in the cloudlet at the time slot t. Then, $A_P <t >= C_P - H_P(t)$ for all p with $1 \leq p \leq M$. And as $PE < t >= < PE_1, PE_2, PE_3, \ldots, PE_P, \ldots, PE_M >$ for all k with $1 \leq p \leq M$ be the number of processing element available at the time slot. So, all task will be submitted to the vm0s based on Best fit algorithm when $PE_p \geq \alpha_k$ valid for all p with $1 \leq p \leq M$ and where $1 \leq k \leq N$.

3.4. Unsuccessful requests

At that time slot, if any task requests present in the queue are not answered, then the time stamp of this particular request will become 1, and they will be added to the queue again. If it happens again with this particular request, then the request will be sent to the cloud.

IV. PROBLEM STATEMENT

We have considered a single data center or a single cloudlet. Every mobile user submits their request, request k at a time slot t. Each request $r_i(t)$ consists of a set of amounts of resource needs for each task on the cloudlet and the time period i $\theta_{k,,i}(t) = \theta_{1,\, i}(t), \theta_{2,\, i}(t), \theta_{3,\, i}(t), \ldots \theta_{M,\, i}(t)$; θ_i, where $\theta_{M,\, i}(t)$ is the amount of resource k needed. Assume that each request arrives one-by-one and no information is there for the future requests. First, all request arrives at the cloudlet server, and then it will check whether the cloudlet has enough resources to execute the applications. If yes, then it will choose Best Fit solution to choose the virtual machine in Cloudlet. If no virtual machine in cloudlet has enough space, then it will send to the cloud.

V. PROPOSED ALGORITHM

Researchers have proposed various scheduling algorithms for the consolidation of user requests to cloud resources [15, 18]. In our proposed LBTAA (Latency Based Task Allocation Algorithm), initially all tasks will reach to cloudlet, then it starts searching VMs which can fulfill their respective resource requirements. On those selected VMs, the provider will find out the mapping between the task and VM, whose available RAM size is closer to the user request. In this way, all tasks will be allocated to all VMs. If it fails, then usually it goes to cloud and follow the same rule to get the cloud resources. Here, we have followed FCFS based scheduling to execute all tasks. Total execution time will depend on the propagation time and execution time in the specified VM. Propagation time is defined as the ratio of the distance of cloudlet or cloud from the user to the speed.

Algorithm 1: LBTAA

Input: $\theta_k = \{\theta_1, \theta_2, \theta_3, \ldots, \theta_k\}$, where $1 \leq k \leq N$ a set of requests from various mobile devices. We assume that getting RTT (Round Trip Time) we have to add propagation time and transmission time. To define propagation time we assume the distance of the cloudlet as well as cloud and the speed also.

Output: Execution time when using cloudlet and when not using cloudlet.

```
1:  for each $\theta_i$ where $1 \leq i \leq N$ do
2:          Add the value into central queue;
3:          for each cloudlet $VM_1$ to $VM_M$ do
4:              Check $\mu_k \leq C_k$ and $\beta_k \leq PE_k$;
5:              if Yes then
6:                  go to line 8;
7:              end if
```

```
8:              if No then
9:              go to line 6;
10:         end if
11: end for
12: for each cloudlet Vm1 to VMM do
13:         Search $C_k$ is closer to k;
14:         Submit the task to $VM_k$;
15:         $C_{k, M} = C_{k, M} - \mu_k$;
16: end for
17: for each cloud datacenter $VM_1$ to $VM_m$ do
18:         Search $D_k$ is closer to k;
19:         Submit the task to $D_k$
20:         $D_{k, M} = D_{k, M} - \mu_k$;
21: end for
22: for each cloudlet $C_1$ to $C_M$ do
23: for each task $\theta_1$ to $\theta_L$ do
24: $\theta_{K,TIME} \frac{\alpha k}{MIPS\_K}$; ;
25: $\theta_{K,TIME} = \frac{Distance of Cloudlet}{Speed}$ ;
26: end for
27: end for
28: for each cloud $C_1$ to $C_M$ do
29:         for each task $\theta_1$ to $\theta_K$ do
30: $\theta_{K,TIME} \frac{\alpha k}{MIPS\_M}$;
31: $\theta_{K,TIME} = \frac{Distance of Cloudl}{Speed}$; ;
32: end for33: end for
```

VI. PERFORMANCE EVALUATION

Here, we find out the result of the proposed algorithm and investigate the performance of the algorithm.

6.1. Simulation environment

The requirements of resources by each task are different. If a request needs memory k, then it first searches in all cloudlets, whether this amount is present or not in VM. The amount of resource k is required randomly within the certain range as shown in Table 1.

Table 1
Parameters of a user request

Type	*Size*
Total size of Instruction (MI)	10000
Memory (MB)	512
Storage (GB)	20
CPU (MIPS)	800
Bandwidth (MBPS)	75
No. of Processing Element	1

Table 2
Parameters for all VMs in the cloudlet

Type	*Size*
Memory (MB)	1024
Storage (GB)	20
CPU (MIPS)	1000
Bandwidth (mbps)	75
No. of Processing Element	2
Distance of cloudlet from mobile device (K.M)	5000
Propagation speed (Km/s)	10

We are assuming the resource sizes of cloudlets environment. There exit servers and a set of mobile devices, who are sending their task to the cloudlet for getting the resource. The cloudlet have four VMs, and attributes are depicted in Table 2 [12, 13].

6.2. Results

We have considered two heuristic techniques as our assessment benchmarks. The first one allows requests according to the FCFS strategy, and a user request will be granted to utilize the cloudlet resource on demands and it also follows the Best-fit algorithm. Otherwise, the user request will be denied immediately and go to the cloud. We have suggested this algorithm as Best-fit algorithm and FCFS-Batch scheduling algorithm. Firstly, we have considered that all requests are sent to the cloudlet if it fails to respond then the response will be received from cloud. From the graph shown in Figure 4, we can see the difference in the execution time required for the task while using cloud and cloudlet. For cloud, the distance between mobile devices to the cloud is 10000km [13]. Secondly, we will consider cloudlet as a middleware service. By default, we have taken memory size of the VMs as 1024 MB, and after applying Best-fit algorithm for all task in a time slot, we get to know about the unused memory in each VM, which is shown in Figure 5.

VII. CONCLUSION

We have studied various techniques which have been proposed in the literature for the allocation of mobile requests to the resources (VMs) of MCC. Various scheduling policy has been used there. Latency is one of the important factor for the cloud environment. To reduce this latency value several algorithm was proposed by the researchers. The proposed LBTAA algorithm is based on Best-Fit and FCFS algorithm to minimize the latency of the MCC system. This method can make the execution faster in the cloud environment. Simulation results are showing that proposed algorithm are promising. We are going to implement this model practically in the laboratory in the future.

REFERENCES

[1] Fernando, N., Loke, S. W., & Rahayu, W., 'Mobile cloud computing: A survey', *Future Generation Computer Systems*, *29*(1), pp. 84-106, 2013.

[2] D. Puthal, B. P. S. Sahoo, S. Mishra, S. Swain, "Cloud computing features, issues, and challenges: a big picture", *In IEEE International Conference on Computational Intelligence and Networks (CINE),* pp. 116-123, 2015.

[3] Zhang, W., Wen, Y., Guan, K., Kilper, D., Luo, H., & Wu, D. O., 'Energy-optimal mobile cloud computing under stochastic wireless Channel', *IEEE Transactions on Wireless Communications*, *12*(9), pp. 4569-4581, 2013.

[4] Gai, K., Qiu, M., Zhao, H., Tao, L., & Zong, Z., 'Dynamic energy-aware cloudlet-based mobile cloud computing model for green Computing', *Journal of Network and Computer Applications*, *59*, pp. 46-54, 2016.

[5] Abolfazli, S., Sanaei, Z., Gani, A., Xia, F., & Lin, W. M., 'RMCC: Restful Mobile Cloud Computing Framework for Exploiting Adjacent Service-Based Mobile Cloudlets'. In *2014 IEEE 6th International Conference on Cloud Computing Technology and Science (CloudCom),* pp. 793-798, December 2014.

[6] Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N., ‘The case for vm-based cloudlets in mobile computing’, *IEEE pervasive Computing*, *8*(4), pp. 14-23, 2009.

[7] Li, Y., & Wang, W., ‘Can mobile cloudlets support mobile applications’, In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications,* pp. 1060-1068, April 2014.

[8] Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N., ‘The case for vm-based cloudlets in mobile computing’, *IEEE pervasive Computing*, *8*(4), pp. 14-23, 2009.

[9] Hoang, Dinh Thai, Dusit Niyato, and Ping Wang. ‘Optimal admission control policy for mobile cloud computing hotspot with cloudlet’, *2012 IEEE Wireless Communications and Networking Conference (WCNC),* 2012.

[10] You, Changsheng, et al., ‘Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading’, *arXiv preprint rXiv:1605.08518* 2016.

[11] Chen, M., Hao, Y., Li, Y., Lai, C. F., & Wu, D., ‘On the computation offloading at ad hoc cloudlet: architecture and service modes, *IEEE Communications Magazine*, *53*(6), p. 18-24, 2015.

[12] Xia, Qiufen, Weifa Liang, and Wenzheng Xu., ‘Throughput maximization for online request admissions in mobile cloudlets’, *2013 IEEE 38th Conference on Local Computer Networks (LCN)*, 2013.

[13] Brooks, D. M., Bose, P., Schuster, S. E., Jacobson, H., Kudva, P. N., Buyuktosunoglu, A., & Cook, P. W., ‘Power-aware microarchitecture : Design and modeling challenges for next-generation microprocessors’, *IEEE Micro*, *20*(6), pp. 26-44, 2000.

[14] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., & Zaharia, M., ‘A view of cloud computing’, *Communications of the ACM*, *53*(4), pp. 50-58, 2010.

[15] S. K. Mishra, R. Deswal, S. Sahoo, B. Sahoo, “Improving energy consumption in cloud”, *In 2015 Annual IEEE India Conference (INDICON)*, pp. 1-6, December, 2015.

[16] J. Pooyan, A. Aakash, and P. Claus, ‘Cloud Migration Research: A Systematic Review’, *IEEE Trans. Cloud Computing*, vol.1, no. 2, pp. 142-157, 2013.

[17] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, ‘Real-Time Tasks Oriented Energy-Aware Scheduling in Virtualized Clouds’, *IEEE Trans. Cloud Computing*, vol. 2, no. 2, pp. 168-180, 2014.

[18] S. Sahoo, S. Nawaz, S. K. Mishra, B. Sahoo, “Execution of real time task on cloud environment”, *In 2015 Annual IEEE India Conference (INDICON)*, pp. 1-6, December, 2015.

[19] D. Puthal, S. Nepal, R. Ranjan, and J. Chen. “Threats to Networking Cloud and Edge Datacenters in the Internet of Things”, *IEEE Cloud Computing*, Vol. 3(3), pp. 64-71, 2016.

[20] D. Puthal, S. Nepal, R. Ranjan, and J. Chen. “A Secure Big Data Stream Analytics Framework for Disaster Management on the Cloud”, In *18th IEEE International Conferences on High Performance Computing and Communications,* pp. 1218-1225, 2016.

[21] D. Puthal, S. Nepal, R. Ranjan, and J. Chen. “DLSeF: A Dynamic Key Length based Efficient Real-Time Security Verification Model for Big Data Stream”, *ACM Transactions on Embedded Computing Systems*, Vol, 16(2), pp.51, 2016.

[22] S. K. Mishra, K. S. Sahoo, B. Sahoo, S. K. Jena, “Metaheuristic Approaches to Task Consolidation Problem in the Cloud”, *Resource Management and Efficiency in Cloud Computing Environments*, pp. 168, 2016.

[23] S. Sahoo, B. Sahoo, A. K. Turuk, S. K. Mishra, “Real Time Task Execution in Cloud Using MapReduce Framework”, *Resource Management and Efficiency in Cloud Computing Environments*, pp. 190, 2016.