Resilient Framework for Distributed Computation Offloading: Overview, Challenges and Issues

Collinson Colin M. Agbesi[1]
Department of Computer Science
Koforidua Technical University
Koforidua, Eastern Region, Ghana

Dr. Katsriku Apietu Ferdinand[2]
Computer Science Department
University of Ghana, Legon
Greater Accra, Ghana

Dr. Jamal-Deen Abdulai[3]
Computer Science Department
University of Ghana, Legon
Greater Accra, Ghana

Kofi Adu-Manu Sarpong[4]
Computer Science Department
Valley View University, Oyibi
Greater Accra, Ghana

***Abstract*—Gradually, mobile and smart computing devices are becoming pervasive and prevalent in society and also increasingly being used to undertake the daily tasks and business activities of individuals and organizations worldwide as compared to their desktop counterparts. But these mobile and smart computing devices are resource constrained and sometimes lack the needed computational capacities; thus, memory, energy, storage, and processor to run the plethora of resource intensive applications available for mobile users. There is a lot of benefit of offloading resource demanding applications and intensive computations from mobile devices to other systems with higher resource capacities in the cloud. Mobile cloud computing is a form of cloud computing that seeks to enhance the capacity and capabilities of mobile and smart computing devices by enabling mobile devices to offload some computational tasks to the cloud for processing which otherwise would have been a challenge. The study setup an experiment to investigate computation offloading for mobile devices. The study also presented an energy model for computation offloading. It was observed during the experiment that by offloading intensive applications from mobile and smart computing devices to other systems with higher resource capacities, a great amount of resource efficiency is achieved.**

Keywords—Cloud computing; mobile cloud computing; computation offloading; distributed computation offloading

I. Introduction

There is an ever growing proliferation and widespread use of mobile and smart computing devices and applications worldwide to perform various kinds of work related activities. Devices such as smartphones, tablets, and e-book readers enable users to connect to the internet instantly, giving them access to a vast amount of information and enabling them to perform their daily activities and tasks using productivity applications on their mobile devices [15], [16], [23]. Despite the growing trends in mobile and smart computing technologies, these mobile devices still lack the needed computational resources to execute and run resource demanding applications and intensive computations [5], [24].

Mobile cloud computing is a variant of cloud computing which addresses the challenges faced by mobile and smart computing devices in running resource intensive applications [5], [23], [27]. Mobile cloud computing seeks to enhance the capacity and capabilities of mobile and smart computing devices in order to execute the many resource demanding applications available for mobile users [12], [20], [25]. Mobile Cloud Computing is an emerging type of cloud computing that provides resource efficiencies among mobile devices. Mobile cloud also enhances the capabilities of mobile devices to execute the plethora of resource demanding applications available for mobile users [3], [12]. Mobile cloud computing is a computing paradigm in which computationally intensive and resource demanding mobile applications such as 3D graphics, face recognition, voice recognition, games, videos, malware protection and data manipulation are transferred to other systems with higher resource capacities for efficient processing in the cloud [4], [6], [10]. The goal of mobile cloud computing is to enable mobile and smart computing devices to run and execute the plethora of resource hungry mobile applications which greatly consumes processing power, energy and battery life of mobile devices by offloading computations to surrogates in the cloud [19], [26]. Mobile cloud computing also boost the computational power of mobile and smart computing devices to enable them execute the plethora of resource hungry mobile applications which greatly consumes processing power, energy and battery life of mobile devices by offloading computations to surrogates in the cloud [11], [24], [25], [28]. From Fig. 1, mobile and smart computing devices can access computer applications, platforms and infrastructure via cloud systems.

A. Problem Statement

The computational power and capacity of mobile and smart computing devices have increased over the years. This has allowed mobile and smart computing devices to run resource demanding applications [1], [16], [27]. In spite of the increase in computational capacity such as processor, memory, storage, display and battery life; mobile and smart computing devices are still very much limited regarding the kind of applications that they can run and execute [9], [17], [18]. Also the open nature of mobile applications market has encouraged the development of applications by millions of enthusiasts over the world. Unfortunately many of these applications are resource intensive and put a lot of strain on the mobile device [12], [13], [16], [18]. In many instances the mobile and smart computing devices are unable to run these applications. This is making mobile and smart computing devices unattractive to users in terms of applications that these systems can run and execute [2], [15], [20]. The limitations in computational power, battery life, memory, storage capacities and other resources has led to a emergence of several techniques to enable mobile devices process and execute the many rich and resource demanding applications available for mobile and smart computing devices [9], [14], [19], [27].

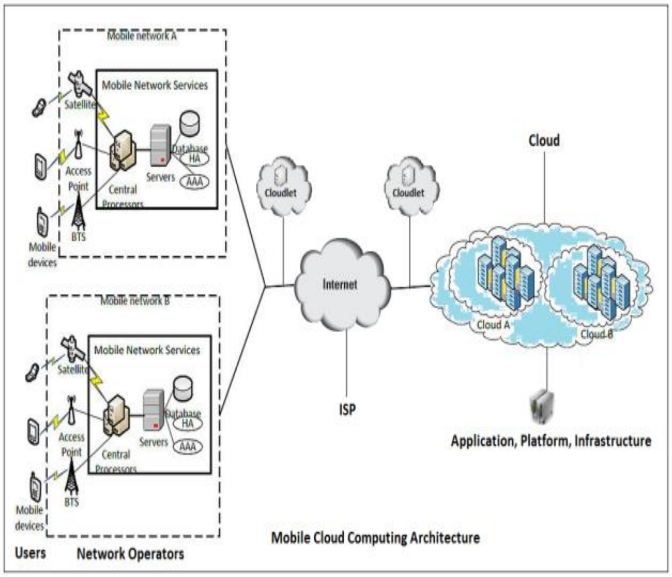


Fig. 1. Mobile Cloud Architecture [22].

II. Related Work

Mobile computational offloading is an aspect of mobile cloud computing that focuses on transferring computations from native applications running on resource limited mobile devices to other powerful systems such as cloud servers for processing and execution [5], [10], [13], [18], [26]. Increasingly, computation offloading is seen as a growing cloud based service for mobile and smart computing devices [9], [12], [20], [24]. Computation offloading is sending very intensive and resource demanding and complex computations to resourceful servers with higher computational capacities for efficient and effective execution and processing which would otherwise be impossible for mobile and smart computing devices to execute locally [16], [18], [27], [30]. Also computation offloading algorithms determine which tasks should be performed locally and which should be performed remotely in order to provide efficient use of resources thus saving battery life and processing power of mobile and smart computing devices [1], [17], [18].

Previous works have identified different mechanisms to solve the seamless execution of offloaded computational tasks from resource limited mobile devices to other systems with higher computational power and resources in the cloud. Most of these research devised a variety of approaches to implement energy saving, time saving and both energy and time saving strategies in computation offloading for mobile cloud computing [28], [10], [22], [7]. Some of these researches have proposed sending computationally intensive tasks to nearby mobile devices with higher computational resources [28], [10].

Others have identified elastic computation offloading methods in which resource intensive applications are partitioned into user interaction activities and computational activities and some parts of the computational activities are sent to surrogates or remote servers for efficient and timely processing whiles the mobile devices process the other parts locally [7], [18], [24]. Some other works have also proposed the complete cloning, mirroring or imaging of a mobile device system together with its operating system and applications and sending the mirrored or cloned image to a clone cloud server located at a remote place for processing and the results sent back to the mobile devices [10]. Also some other works have suggested context-aware computation offloading schemes which enables mobile and smart computing devices to learn about their environment and state before making offloading decisions [22].

Furthermore, other research works [8] have studied a game-based approach to computation offloading in which mobile devices learn about the current state of surrogates before deciding to send computational tasks to these remote servers depending on their resource availability. Satyanarayanan et al. [27] and Chun & Maniatis [9] have proposed the encapsulation of a mobile device software stack including operating environment and applications into a virtual machine image and offloading it to a more powerful system in the cloud for execution. Also Giurgiu et al. [13] proposed consumption graphs to decide which part of the computation should run locally and which part should be executed remotely. Klovachev and Klamma [18] have also proposed a mobile augmented cloud services (MACS) middleware to enable the execution of elastic mobile applications in the cloud. Wolski et al. [29] presented a framework for computation offloading decisions in which schedulers determine when to move parts of a computation to more resourceful systems for execution in the cloud. Chen [7] have also suggested a decentralized computation offloading game in which resource constrained mobile devices wait and learn about the state of surrogates before offloading computational tasks depending on their resource availability. Also Orsini et al. [21] have proposed a context-aware mechanism for computation offloading in which mobile devices decide when and how to offload based on their state and resource capacities. Moreover, Cuervo et al. [6] presented a fine grained energy aware computation offloading mechanism dubbed MAUI. MAUI decides at runtime which part of computational tasks should be performed remotely to achieve energy savings based on the mobile devices connectivity constraints. Kemp et al. [17] also presented Cuckoo, a computation offloading framework that simplifies the development of mobile applications and provides a dynamic runtime system that determines which part of the computation should be offloaded to the cloud and which part should be locally executed to save battery life. Also Cong et al. [8] presented a computation offloading as a service (COSMOS) to provide leverage for all mobile device requests and reduce monetary cost to the cloud provider. Deng et al. [10] also proposed a computation offloading scheme for service workflow in which similar service components are manage during computation offloading. Some studies have provided alternative means of computation offloading but lack the fundamental models to evaluate energy consumed during the computational offloading process. Hence there is still much work to be done in order to find a good resource efficient computational offloading scheme for mobile and smart computing devices which have variable resource capacities and computational power [6], [11], [24], [30].

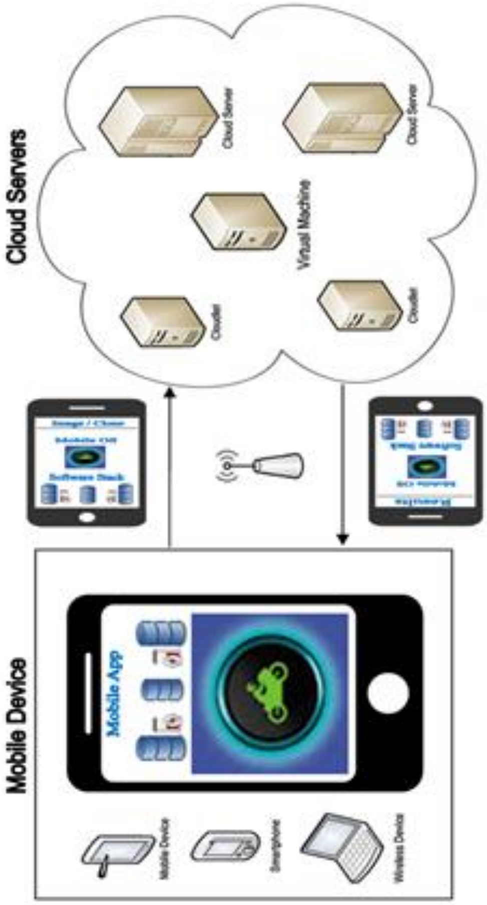


Fig. 2. Traditional Computation Offloading Scheme.

A. Traditional Computation Offloading

Fig. 2 shows the traditional computation offloading framework. The traditional computation offloading scheme allows mobile devices to examine the computational task to be executed to determine whether it can execute the task using its local resources or it cannot [5], [6], [11], [12], [15], [16], [18]. The resource manager determines the available resource capacity of the mobile device. Base on the available resource capacity of mobile devices, a decision is made whether or not to execute the computational task locally or to partition it and send the intensive part to a remote server in the cloud for processing. Then the partitioner divides the computational task or mobile application to be executed into user interaction and intensive tasks. The user interaction task is then assigned to the local execution manager to execute locally whiles the computationally intensive tasks are sent to the offload manager. The offload manager then transfers the intensive tasks to other powerful server systems with higher and scalable resource capacities in the cloud. The result of the computation in the cloud is sent back to the mobile device. The main setback of the traditional computation offloading approach is the limited available servers to offload computational tasks. The intensive part of the computation is offloaded to a server in the cloud for processing. This leads to an increase in the computational tasks load of the cloud server. Depending on the nature and volume of the resource intensive computation to be executed, this might bring additional burdens to the chosen server in the cloud.

III. Proposed Distributive Computation Offloading Framework

The distributive computation offloading framework in Fig. 3 seeks to overcome the challenges inherent in traditional computation offloading schemes for mobile cloud computing. The distributive offloading scheme seeks to provide a resource savings by enabling both resource poor mobile devices running intensive mobile applications to save a great amount of time and energy during computations of intensive mobile applications and also reducing the burden on servers in the cloud during the processing of computational tasks. This distributive approach to computation offloading helps to reduce the overwhelming burden of intensive, complex and huge computations on servers in the cloud by dividing and sharing tasks among cloud servers based on their resource capacities, availability and computational capabilities. The resource intensive and complex tasks is divided and shared among servers in the cloud with available computational resources to compute the tasks and the results returned back to the mobile devices instead of one server processing the entire computationally intensive and complex task as in the case of traditional computation offloading schemes. The distributive approach further reduces the workload and burden on resource poor mobile devices which have to process locally parts of the resource intensive mobile applications that is sent to the cloud for processing. As a result, the mobile device will have less workload and tasks to process than would have been in traditional computation offloading schemes.

The distributive framework enables resource savings for both mobile devices and cloud server systems in mobile cloud computing. It provides the smallest amount of time and energy needed to execute tasks per cloud server in mobile cloud computing. Therefore the distributive offloading scheme delivers the best resource savings in mobile cloud computing than traditional computation offloading schemes. In order to minimize the cost of computation in terms of energy and time and also to achieve optimum energy and time savings for both mobile devices and cloud servers, the distributive computation offloading framework divides the computationally intensive tasks among different servers in the cloud with available resources that can execute these tasks and reduce the total workload on the cloud servers thereby achieving greater energy and time savings. Since the computationally intensive task is divided among available servers with resources capable of computing the tasks, then each cloud server assigned a task can execute it within the shortest possible time and return the results back to the mobile device. Thus each cloud server assigned part of the computationally intensive task to execute then utilizes a small amount of energy compared to the amount of energy that would have been used if the entire computationally intensive task was executed on one single cloud server. The distributive computation offloading scheme enables computationally intensive tasks to be divided and shared among servers in the cloud with enough resource capacities to compute these tasks. This allows resource limited mobile devices to separate user interaction tasks from computationally intensive tasks that must be sent to the cloud for efficient and effective processing and execution.

The resource manager determines the resource capacity of the mobile device and which tasks it can process locally depending on its available resources thus; processor power, memory, storage and energy or battery capacity. The partitioner then divides the resource intensive mobile applications into less computation intensive tasks or user interaction activities and computationally intensive activities. The local execution manager is responsible for processing the less computation intensive or user interaction tasks on the mobile device locally. The offload manager sends the computationally intensive parts to the middleware software which acts as an interface between the mobile device and cloud servers.

The cloud resource manager determines the resource capacity and availability of surrogates and servers in the cloud and chooses M available cloud servers. Based on these resource availability decisions and computationally intensive task, a number of N surrogates and servers are selected in the cloud to execute the computationally intensive mobile applications. The computationally intensive task is divided into N parts by the partitioner depending on the number of cloud servers that have been selected to execute the tasks.

The cloud offload manager then transfers the allocated tasks to each server in the cloud up to the Nth server system depending on the number of resource available cloud server systems. The load balancer also ensures that tasks are assigned to different cloud servers based on their resource capacities and that no server in the cloud is over burdened with too much tasks to execute. It also helps to adjust the load capacity between various server systems in the cloud.

The network manager monitors and ensures a reliable connection between remote cloud servers and mobile devices. In case of any network connection failures, the network manager notifies the load balancer, resource manager and cloud offload manager to reassign the task to another server system in the cloud with available resources to execute the task.

The remote execution manager monitors and controls all tasks under processing in the cloud and helps to send results of the processed tasks back to the mobile device. The distributive computation offloading scheme thus offers an efficient and better approach to computation offloading that greatly saves energy and time for both mobile devices and remote servers in the cloud. It pursues an optimum approach to energy and time savings for both mobile devices and remote servers in the cloud.

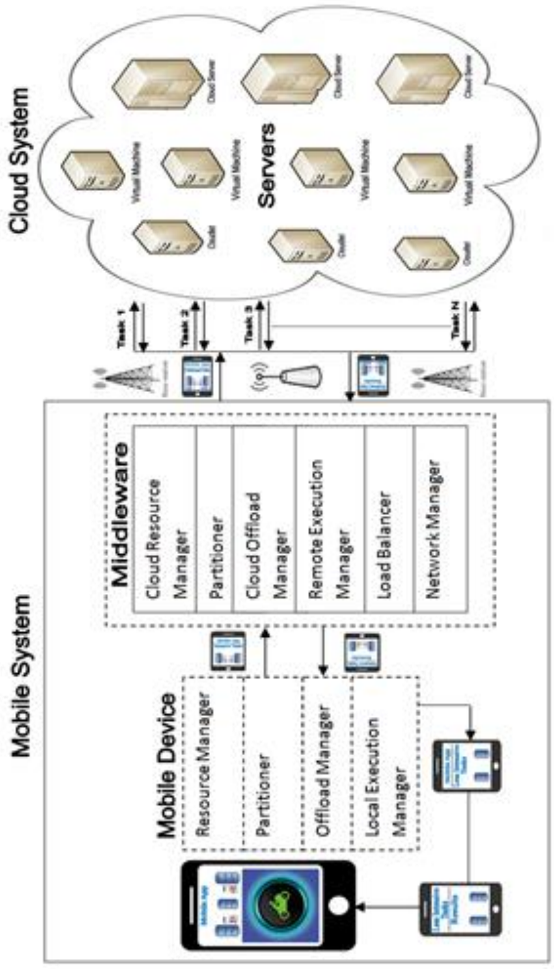


Fig. 3. Distributed computation offloading framework.

A. Algorithm for Distributive Computation Offloading

```
1. Start:
2. Initialize mobile device
3. Run mobile application
4. If mobile application is not intensive
5. Run application locally
6.  Else if mobile application is intensive
7. Determine resource capacity of mobile device
8. If mobile device is capable
9. Run application locally
10.  Else if mobile device is not capable
11. If wireless access is not available
12. Force local execution or end application
```

13. Else if wireless access is available
14. Partition application into user interaction and computationally intensive tasks
15. Send user interaction to mobile device for computation
16. Send computationally intensive tasks to middleware
17. Partition computationally intensive tasks into **n** selected cloud servers.
18. Send partitioned tasks to **n** selected cloud servers for computation
19. Return results to mobile device
20. End

As stated in the algorithm, without being switched on and without any computational activities, the computation offloading process cannot be followed by the mobile device. Therefore the mobile device has to be switched on and initialized. The initialization involves loading of the operative system and other essential component systems on the mobile device. After the initialization, the mobile application to be executed is run by the mobile device. If the mobile application is not intensive, then it is executed locally else if the mobile application is intensive, depending on the resource availability of the mobile device and also if wireless connectivity is available on the mobile device, the mobile application is divided into less intensive and intensive parts. Then the less intensive user interaction part is executed locally by the mobile device whiles the computationally intensive part is divided among available cloud servers and sent to them for remote execution and the results is sent back to the mobile device.

B. *Flowchart of Distributive Computation Offloading Framework*

The flowchart shows the rudimentary procedures of task execution in a distributive computation offloading framework. The distributive process is made possible due to the elastic nature of mobile applications which enables mobile applications to be partitioned into smaller chunks of executable tasks. Similarly intensive computations to be performed by mobile devices can be partitioned into smaller chunks of tasks. Mobile applications and computations can be partitioned during development (compile time) or execution (runtime).

From the flowchart in Fig. 4, the mobile device is switched on and its operative system initialized together with other system components. If the mobile device is not powered on, then the computation offloading process is cannot start. If the mobile device is powered on, then the mobile application to be executed is run by the mobile device. If the mobile device detects that the mobile application is not intensive, the mobile device will then execute the mobile application locally using its own resources. But if it the mobile device detects that the mobile application is intensive or it cannot execute the computational task, it will determine its resource availability. If the mobile device can strain execute the computational task or mobile application, depending on the situation, it might attempt to execute it if it is still efficient in the worst case. Otherwise if there is a wireless connectivity capability on the mobile device, then the computational task or application is divided into intensive and less intensive tasks. The less intensive tasks is then sent to the mobile device for processing locally whiles the computationally intensive parts is then divided and sent to available servers in the cloud for remote execution and the results of the remote processing is sent back to the mobile device.

C. *Mathematical Model of Distributive Computation Offloading*

The study presented a mathematical model for distributive computation offloading. In addition, a model for calculating the energy and time cost of distributive computation offloading is presented.

1) Distributive Computation Offloading Decisions

Let MA denote mobile application

Let MD denote the mobile device

Let S denote cloud servers

Let m be the number of available cloud servers

Let n be the number of selected cloud servers

Let S_i denote a particular selected cloud server i

where i = 1 to n

Let the intensive part of the mobile application (MA) be partitioned into n parts and let MA_i denote intensive mobile application part i

where i = 1 to n

Also

Let MA_u denote the user interaction part of mobile application (MA)

Then we can assign the partitioned computation intensive tasks to available servers in the cloud for execution as follows;

MA_1 assign to S_1

MA_2 assign to S_2

MA_3 assign to S_3

MA_n assign to S_n

where i = 1 to n

Also we can assign the user interaction activities to mobile device for processing locally as follows;

MA_u assign to MD

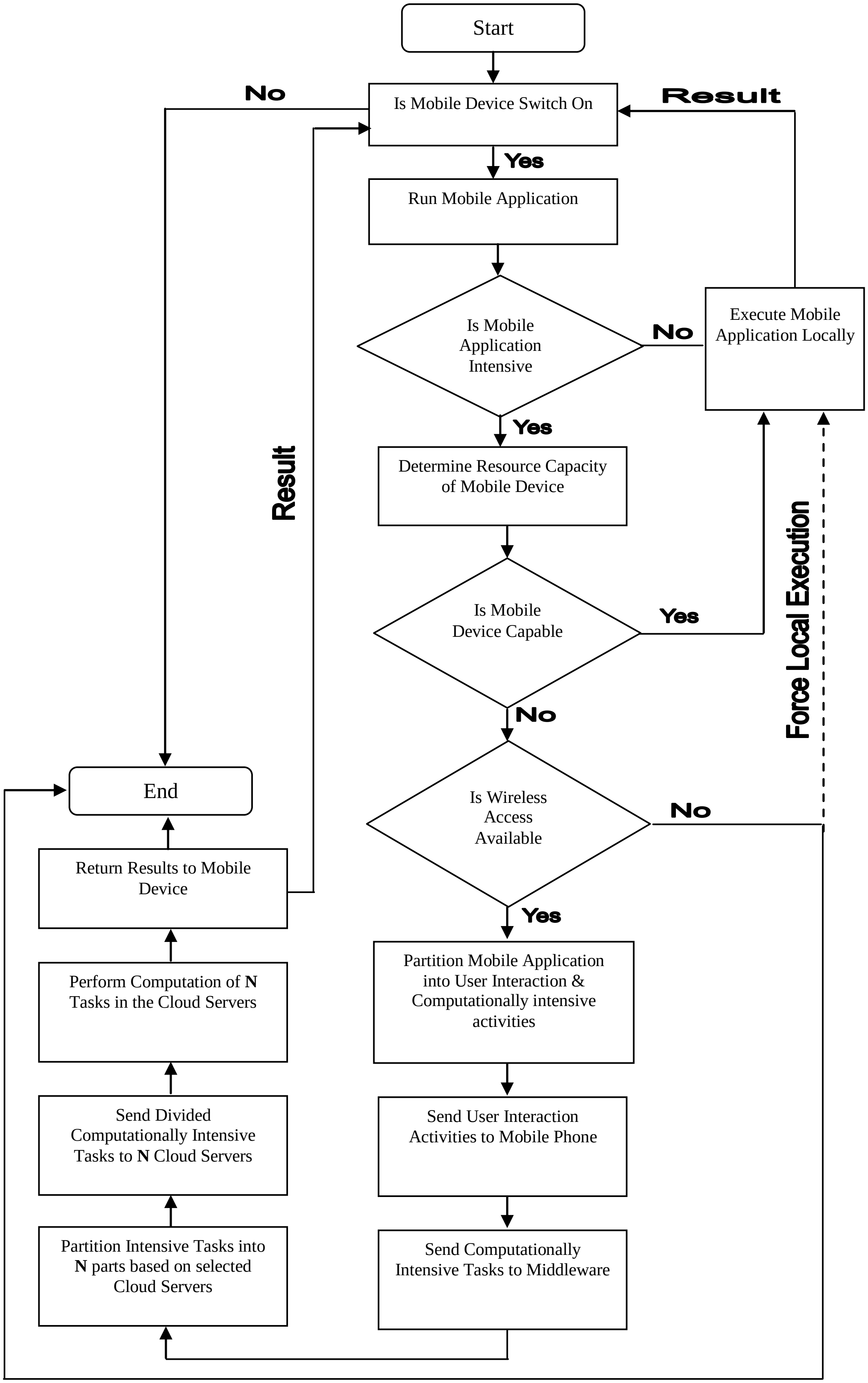

Start
Is Mobile Device Switch On
No
Result
Yes
Run Mobile Application
Is Mobile Application Intensive
No
Execute Mobile Application Locally
Yes
Determine Resource Capacity of Mobile Device
Result
Is Mobile Device Capable
Yes
Force Local Execution
No
End
Is Wireless Access Available
No
Return Results to Mobile Device
Yes
Partition Mobile Application into User Interaction & Computationally intensive activities
Perform Computation of N Tasks in the Cloud Servers
Send Divided Computationally Intensive Tasks to N Cloud Servers
Send User Interaction Activities to Mobile Phone
Partition Intensive Tasks into N parts based on selected Cloud Servers
Send Computationally Intensive Tasks to Middleware

Fig. 4. Flowchart of distributive computation offloading framework.

2) Energy Cost of Computation

Let EC denote energy consumption

Let ECC denote energy cost of computation

Let MAC denote intensive parts of mobile application in the cloud

Let MAL denote intensive parts of mobile application locally executed on mobile device

Let EC_{MAC} denote energy consumption of intensive mobile applications in the cloud

Let EC_{MAL} denote energy consumption of less intensive mobile application locally executed

Let ETT_{MAi} be the energy used to transmit intensive mobile application i to the cloud

Let ETE_{MAi} be the energy used to execute the intensive mobile application i in the cloud

Let ETR_{MAi} be the energy spent to receive the results of executing mobile application i in the cloud

Then

$$EC_{MA1} = ETT_{MA1} + ETE_{MA1} + ETR_{MA1}$$

Likewise

$$EC_{MA2} = ETT_{MA2} + ETE_{MA2} + ETR_{MA2}$$

Thus

$$EC_{MAn} = ETT_{MAn} + ETE_{MAn} + ETR_{MAn}$$

$$EC_{MAC} = EC_{MA1} + EC_{MA2} + EC_{MA3} + \ldots + EC_{MAn} \tag{1}$$

where i = 1 to n

Now

$$EC_{MAL} = EC_{MAu}$$

Hence

$$EC_{MAL} = EC_{MAu} = ETE_{MAu} \tag{2}$$

Then the total energy cost of computation is given by Energy Cost of Computation (ECCMA) where

$$ECC_{MA} = EC_{MAC} + EC_{MAL} \tag{3}$$

$$ECC_{MA} = EC_{MA1} + EC_{MA2} + EC_{MA3} + \ldots\ldots\ldots + EC_{MAn} + EC_{MAu} \tag{4}$$

Therefore

$$ECC_{MA} = \sum_{i=1}^{n} EC_{MAi} + EC_{MAu} \tag{5}$$

3) Time Cost of Computation

Let TC denote time consumption

Let TCC denote time cost of computation

Let MAC denote intensive parts of mobile application in the cloud

Let MAL denote intensive parts of mobile application locally executed on mobile device

Let TC_{MAC} denote time consumption of intensive mobile applications in the cloud

Let TC_{MAL} denote time consumption of less intensive mobile application locally executed

Let TTT_{MAi} be the time to transmit intensive mobile application i to the cloud

Let TTE_{MAi} be the time to execute the intensive mobile application i in the cloud

Let TTR_{MAi} be the time spent to receive the results of computing mobile application i in the cloud

Then

$$TC_{MA1} = TTT_{MA1} + TTE_{MA1} + TTR_{MA1}$$

Likewise

$$TC_{MA2} = TTT_{MA2} + TTE_{MA2} + TTR_{MA2}$$

Thus

$$TC_{MAn} = TTT_{MAn} + TTE_{MAn} + TTR_{MAn}$$

$$TC_{MAC} = TC_{MA1} + TC_{MA2} + TC_{MA3} + \ldots + TC_{MAn} \tag{1}$$

where i = 1 to n

Now

$$TC_{MAL} = TC_{MAu}$$

Hence

$$TC_{MAL} = TC_{MAu} = TTE_{MAu} \tag{2}$$

Then the total time cost of computation is given by Time Cost of Computation (TCC_{MA})

$$TCC_{MA} = TC_{MAC} + TC_{MAL} \tag{3}$$

$$TCC_{MA} = TC_{MA1} + TC_{MA2} + TC_{MA3} + \ldots\ldots\ldots + TC_{MAn} + TC_{MAu} \tag{4}$$

Therefore

$$TCC_{MA} = \sum_{i=1}^{n} TC_{MAi} + TC_{MAu} \tag{5}$$

D. Computation Offloading Challenges

According to Suradkar & Bharati (2016), there are key challenges and technical issues affecting computation offloading have been identified. Some of these challenges include;

Latency: Mobile devices experience frequent intermittent delays whiles communicating with other devices on the network. This impedes the overall performance of the network thus affecting the throughput of mobile devices.

Bandwidth: Also lack of robust network technologies and inadequate access to networks with high speed, reliability and availability is a hindrance to computation offloading.

Mobility management: Mobile devices are nomadic in nature and this sometimes makes the management of these ubiquitous devices very difficult.

Context-processing: Mobile devices have limited context-aware capabilities this hinders them from processing location and context based computations.

Energy constraint: Mobile devices have limited battery life which hinders them from processing huge computational tasks that might require a lot of energy.

Security and privacy: Mobile devices lack the needed resource capacities to enable them execute some security and encryption algorithms thus exposing them to frequent attacks and hacks.

Operative Environment: Due to limitations in resources, mobile operative systems lacks the needed security and critical features that can properly safeguard it.

Heterogeneity: Diversity in access technologies such as GSM, WiMax, WiFi, WLAN, Bluetooth and their inherent challenges also hinder effective computation offloading.

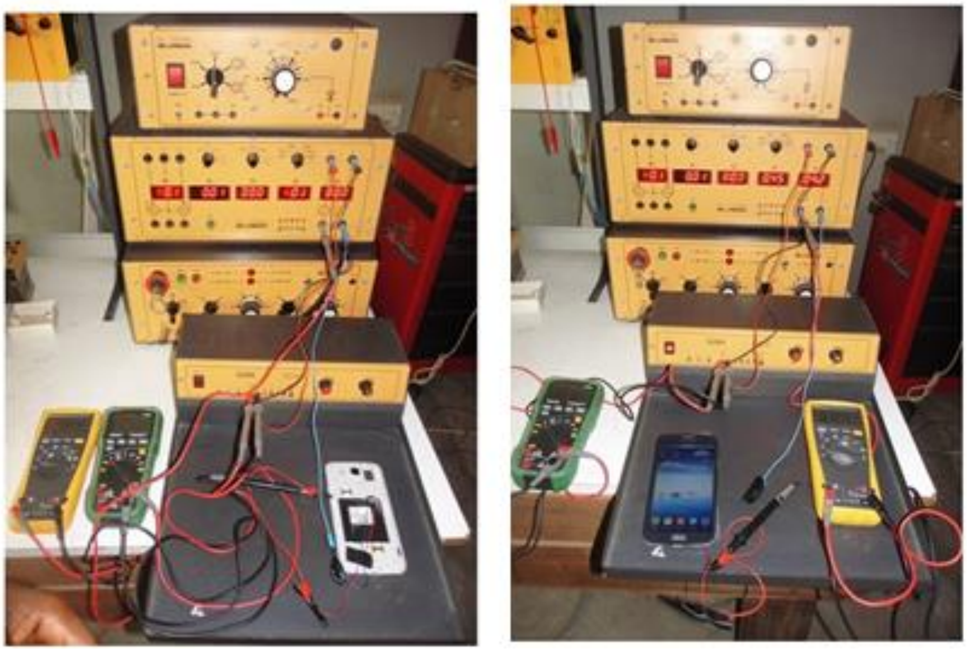
IV. Methodology

The study setup an energy test bed using a De Lorenzo DL3155AL power supply and De Lorenzo DL10060 power meter and two digital meters connected in parallel to the power supply and in series to the mobile device respectively. The test bed was used to measure the energy consumed during the processing of tasks by the mobile device client. Fig. 5 shows the experimental setup and energy test bed for the study.

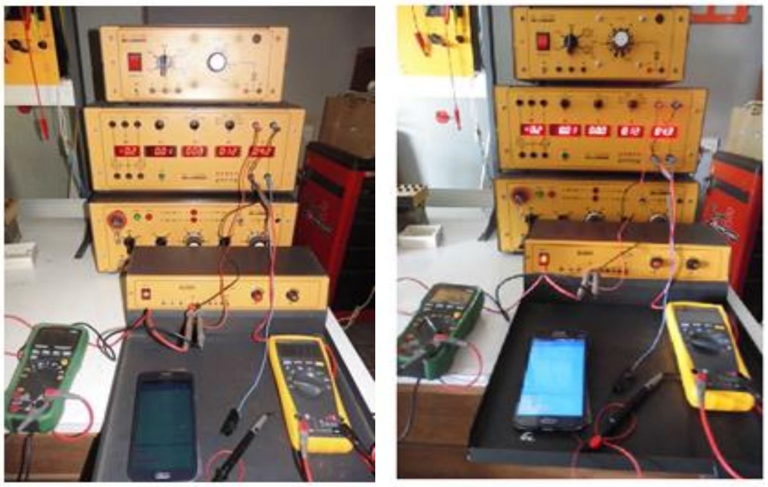
The study also developed a mobile application for performing intensive computations and processing of large volumes of data files in comma separated values (CSV) format. The study used a Samsung Galaxy Mega phone and a laptop which acts as a server to process and migrate very large volumes of data files in comma separated values (CSV) format into a database which was developed for the mobile phone and laptop systems to help collate, process and transfer huge volumes of data in a flat file format for further computations.

Different sizes of huge data files from thousands of records less than 1MB up to about ten million (10,000,000) records approximately 500MB in size were passed to the application to process and the results were logged for both mobile phone and laptop systems. The study experimented on five scenarios involving; very low computation, low computation, moderate computation, high computation and very high computation. The very low computation involved data of size between 1000 to 5000 records, the low computation involved data of size between 10000 to 50000 records, the moderate computation involved data of size between 100000 to 500000 records, the high computation involved data of size between 600000 to 1,000,000 records and the very high computation involved data of size between 2,000,000 to 10,000,000 records. The study also tried to determine the time taken to execute various computational tasks during the experiment.

The results of the demonstration were exported into Microsoft Excel for analysis and plotting some useful graphs to help the study discussions. It was observed during the experiment that by offloading intensive applications from mobile and smart computing devices to other systems with higher resource capacities, a great amount of resource efficiency is achieved.



(a)



(b)

Fig. 5. Energy emulation test bed.

V. RESULTS

A. Results Presentation

Below are the results of the study experiment:

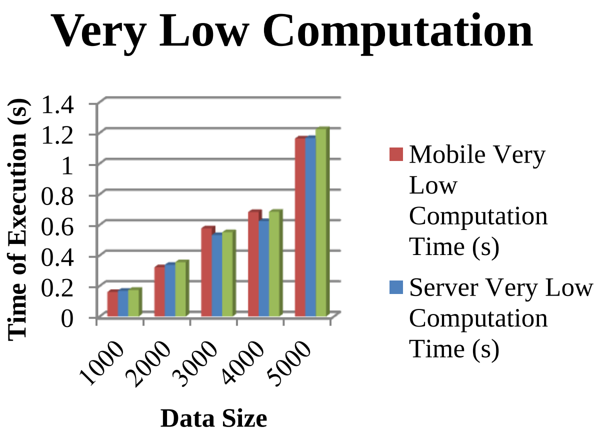


Fig. 6. Very low computation.

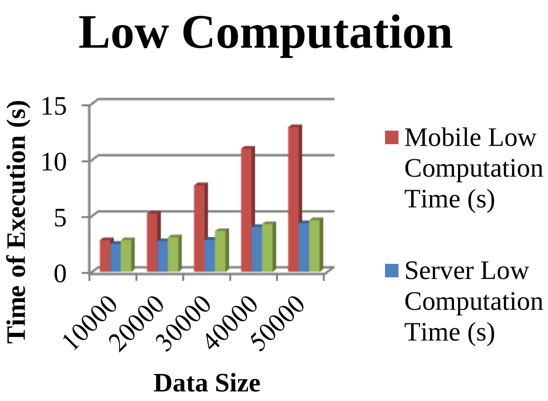


Fig. 7. Low computation.

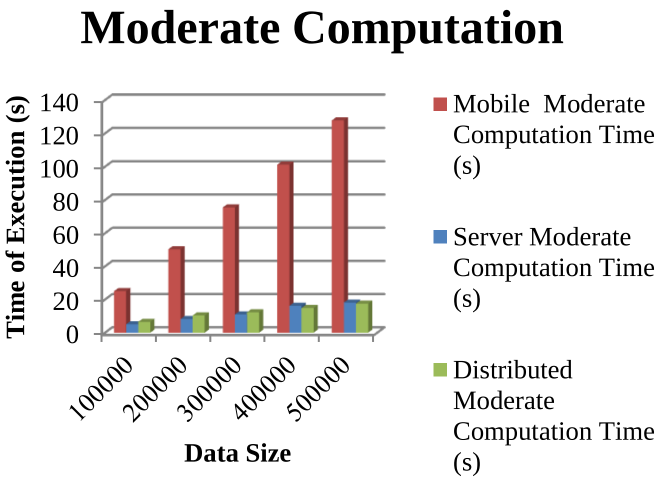


Fig. 8. Moderate computation.

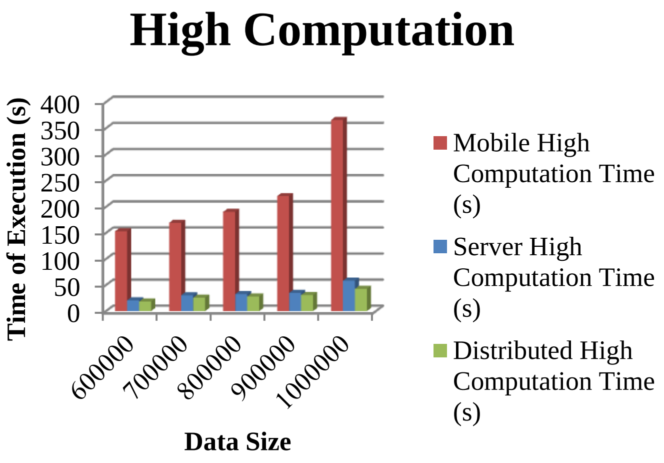


Fig. 9. High computation.

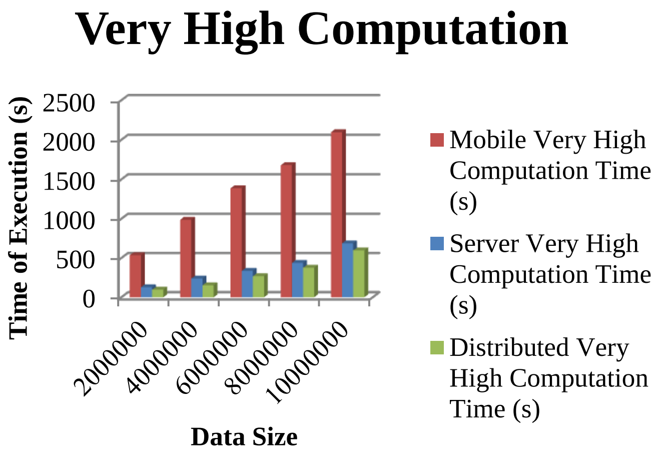


Fig. 10. Very high computation.

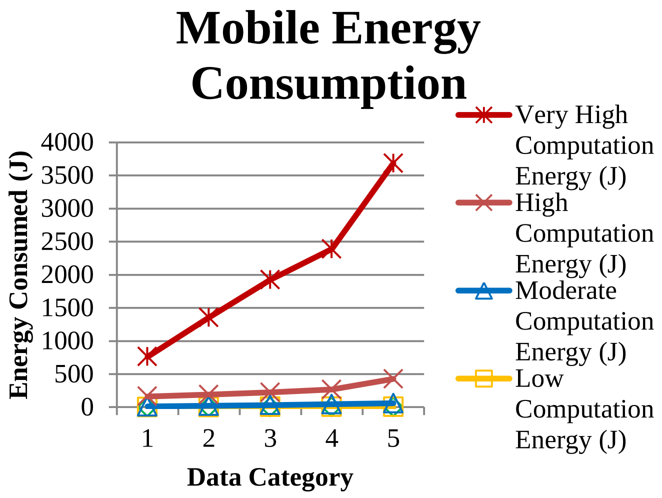


Fig. 11. Mobile energy consumed.

B. *Results Discussion*

The result from Fig. 6 depicts the case of very low computation. This shows that the time of execution for the mobile, server, distributed computations is almost indifferent. This is largely due to the fact that mobile devices are capable of executing some very low computations hence there is no need to offload such less intensive tasks to other systems with higher resource capacities.

Also the result from Fig. 7 shows the case of low computation. As the size of data increases, the time taken to execute the task by the mobile device also increases proportionally. Also the time taken to execute the task by the server and distributed also increases minimally. It is sometimes beneficial to run and execute low computation tasks on mobile devices than to offload to other systems with higher capacities. Some mobile devices are capable of running and executing low computational tasks faster and efficiently and hence there is no need to offload some low computational tasks.

Additionally, the result from Fig. 8 describes the case of moderate computation. As the size of data increases, the time taken to execute the task by the mobile device increases astronomically at a higher rate compared to the computational tasks hence the need to offload these tasks for efficient execution by other systems with higher resource capacity. As the task to be performed increases, the amount of time needed to perform the computation also increases with respect to the computational task. The server and distributed system has more computational resources thus memory, storage and processor power and are able to execute tasks faster and efficiently than the mobile device. It can be observed that with moderate computational tasks, the server and distributed systems performance is better than that of the mobile device. The time taken to execute tasks by the mobile device client is much higher compared to that of the server systems. In the case of moderately high computations, it is sometimes beneficial and efficient to offload some computational tasks to other systems with higher capacities than enable resource constrained mobile devices to execute the tasks.

Furthermore, the result from Fig. 9 illustrates the case of high computation. Again as the size of computational task increases, the time of execution by the mobile device also increases astronomically due to the limited capacity of mobile device hence the need to offload these computational tasks to other systems with higher resource capacities. Hence given any huge or intensive computational tasks, the server and distributed systems are able to execute computational tasks within the shortest possible time using a small amount of time compared to the mobile device. It can be observed that, for intensive computational tasks, it is more efficient to offload to other systems with higher resource capacities than to use resource poor mobile devices.

Lastly, the result from Fig. 10 illustrates the case of very high computation. As can be seen with very high computational tasks, mobile devices are very inefficient at processing these tasks thereby wasting a lot of time and energy. It was observed that, for very huge and intensive computations it is very inefficient to use resource poor mobile devices to execute these tasks. Therefore there is the need to offload very huge, intensive and complex computational tasks from mobile devices to other systems with higher resource capacities for efficient and effective processing.

In addition, Fig. 11 shows the energy consumption of the mobile device during the very low computation, low computation, moderate computation, high computation and very high computation experiments. It can be observed that, the amount of energy needed to execute a task increases with respect to the size of data and computational task to be executed. In the very low computation, low computation and moderate computation experiments; the amount of energy needed to execute a task increases proportionally with respect to the computational task to execute. Also in high computation and very high computation experiment, a large amount of energy is needed by the mobile device to execute intensive and resource demanding tasks. As the size of data and computation increases, a lot of energy is needed to execute these resource demanding and intensive tasks hence the need to offload intensive computations for efficient processing.

VI. Conclusion

Generally, mobile devices are resource limited and handicapped [13], [20], [23], [30]. During the study, it was observed from the experiments that, most mobile devices are unable to run and execute highly intensive computational tasks. Computation offloading is a key benefit of mobile cloud computing which focuses on transferring very intensive computations from resource limited mobile devices to other powerful systems such as cloud servers for efficient and timely processing and execution [8], [12], 24], [26].

It was discovered during the research experiment that, offloading or transferring intensive computations from mobile devices to other systems with higher resource capacities in the cloud is far more efficient than processing of these intensive applications locally on mobile and smart computing devices. A great amount of time and energy resource is saved when resource demanding applications are transferred to cloud servers for execution.

Also the mathematical model presented during the study provides a means of evaluating the time and energy consumed during computation offloading in mobile cloud computing. It provides a way of assessing how much time or energy is supposedly used during the computational offloading process by resource limited mobile devices.

Computation offloading provides a high degree of time and energy resource savings and optimizations than local processing of tasks on resource poor mobile devices [10], [11], [16], [17]. Moreover, it was observed that a great amount of resources such as time and energy is saved when intensive mobile applications and computations are transferred from mobile phones to other systems with higher resource capacities in the cloud [7], [9], [19], [28].

The study recommends a secured computation offloading scheme to safeguard computational tasks and computations being transferred and executed on third party systems in the cloud. Also it recommends a further research into network related issues to enable a smooth transfer and execution of computational tasks in the cloud.

Acknowledgment

I want to express my profound gratitude and appreciation to Dr. Katsriku A. Ferdinand and Dr. Jamal-Deen Abdulai for their immense contributions during my final thesis work.

References

[1] Saeid Abolfazli, Zohreh Sanaei, Ejaz Ahmed, Abdullah Gani, Rajkumar Buyya, (2014), "Cloud-based Augmentation for Mobile Devices: Motivation, Taxonomies and Open Issues", IEEE Communications Survey, First Quarter, Vol. 16, No. 1

[2] Akinlolu Olumide Akande, Nozuko Aurelia April, Jean-Paul W Van Belle, (2013), "Management issues with Cloud Computing", Association of Computing Machinery, Published in ICCC 13 Proceedings of Second International Conference on Innovative Computing and Cloud Computing.

[3] Arockiam L., Monikandan S., Parthasarathy G., (2011) “Cloud Computing: A Survey” International Journal of Internet Computing (IJIC), ISSN No: 2231 – 6965, Vol1, Issue-2.

[4] Bento A. & Bento R., (2011) “Cloud Computing: A New Phase in Information Technology Management” Journal of Information Technology Management, ISSN #1042-1319.

[5] Michael Conley, Amin Vahdat, George Porter, (2015), "Achieving Cost-Efficient, Data-intensive Computing in the Cloud", Association of Computing Machinery (ACM), 2015.

[6] Eduardo Cuervo, Aruna Balasubramanian, Dae-ki Cho, Alec Wolman, Stefan Saroiu, Ranveer Chandra, Paramvir Bah, (2010), "MAUI: Making Smartphones last longer with code offload", Proceedings of 8th International Conference on MobiSystems, pp. 49-62

[7] Chen X., (2015), "Decentralised Computation Offloading Game for Mobile Cloud Computing", Proceedings of Parallel Distributed Systems, IEEE Transactions, Vol. 26, No. 4, pp. 974-983

[8] Cong Shi, Karim Habak, Pranesh Pandurangan, Mostafa Ammar, Mayur Naik, Ellen Zegura, (2014), "COSMOS: Computation Offloading as a Service for Mobile Devices", School of Computer Science, Georgia Institute of Tech., USA

[9] Chun B. & Maniatis P., (2009), "Augmented Smartphone Applications Through Clone Cloud Execution", Proceedings of the 12th Workshop on Hot Topics in Operating Systems, pp. 1-8

[10] Shuiguang Deng, Longtao Huang, Javid Taheri, Albert Y. Zomaya, (2015), "Computation offloading for service workflow in mobile cloud computing", IEEE Transactions Manuscript

[11] Dudeja M. & Soni K., (2014), "Offloading Schemes in Mobile Cloud", International Journal of Computer Applications, Vol 96, No. 8.

[12] Fangming Liu, Peng Shu, Hai Jin, Bo Li, (2013), "Gearing Resource-Poor Mobile Devices with Powerful Clouds: Architectures, Challenges and Applications", IEEE, Wireless Communications.

[13] Ioana Giurgiu, Oriana Riva, Dejan Juric, Ivan Krivulev, Gustavo Alonso, (2009), "Calling the Cloud: Enabling mobile phones as interfaces to cloud applications", Proceedings of 10th ACM/USENIX, International conference, pp. 1-20

[14] Hoang T. Dinh, Chonho Lee, Dusit Niyato, Ping Wang, (2013), "A survey of mobile cloud computing: Architectures, applications and approaches", Wireless Communications and Mobile Computing, Vol 13, Issue 18, pages 1587-1611

[15] Juntunen Antero, Kemppainen Matti, Luukkainen Sakari, (2012), "Mobile Computation Offloading - Factors Affecting Technology Evolution", Proceedings of 11th ICMB.

[16] Karthik Kumar, Jibang Liu, Yung-Hsiang Lu, Bharat Bhargava, (2012), "A survey of computation offloading for mobile systems", Mobile Network Application, DOI 10.1007/s11036-012-0368-0

[17] Roelof Kemp, Nicholas Palmer, Thilo Kielmann, Henri Bal, (2010), "Cuckoo: A computation offloading framework for smartphones" Proc. 2nd International Conference of Mobile Computing, Application, Services, CA, USA

[18] Kovachev D. & Klamma R. (2012), "Framework for Computation Offloading in Mobile Cloud Computing", Department of Information Systems and Databases, RWTH Aachen University.

[19] Atta ur Rehman Khan, Mazliza Othman, Sajjad Ahmad Madani, Samee Ullah Khan, (2014), "A survey of mobile cloud computing application models", IEEE Communications Survey.

[20] Niroshinie Fernando, Seng W. Loke, Wenny Rahayu, (2013), "Mobile Cloud Computing: A Survey", Journal of Future Generation Computer Systems

[21] Gabriel Orsini, Dirk Bade, Winfried Lamersdorf, (2015), "Context-Aware computation offloading for mobile cloud computing: Requirement Anaylsis, Survey and Design Guideline", 12th International Conference on Mobile Systems and Pervasive Computing, MobiSPC 2015.

[22] Prasad M.Rajendra, Jayadev Gyani, Murti P.R.K., (2012), "Mobile Cloud Computing: Implications and Challenges", Journal of Information Engineering and Applications, Vol. 2, No. 7

[23] Pragaladan R. & Leelavathi M., (2014), "A Study of Mobile Cloud Computing and Challenges", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 3, Is. 7

[24] Srinivasa R. V., Nageswara R. N. K., Ekusuma K. (2009) “Cloud Computing: An Overview” Journal of Theoretical and Applied Information Technology.

[25] Abid Shahzad & Mureed Hussain, (2013), "Security Issues and Challenges of Mobile Cloud Computing", International Journal of Grid and Distributed Computing, Vol 6, No 6, pp. 37-50

[26] Suradkar J. A. & Bharati R.D., (2016), "Computation offloading: Overview, Frameworks and Challenges", International Journal of ComputerApplications, Vol 134, No 6

[27] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Cáceres, Nigel Davies, (2009), "VM-based Cloudlets in mobile computing", IEEE Pervasive Computing Proceedings

[28] Shravanthi C. & Guruprasad H.S., (2014), "Mobile Cloud Computing as Future for Mobile Applications", International Journal of Research in Engineering and Technology.

[29] Rich Wolski, Selim Gurun, Chandra Krintz, Dan Nurmi (2008), "Using Bandwidth data to make computation offloading decisions", Proceedings of IEEE, IPDPS, pp. 1-8.

[30] Xiaopeng F. & Cao J. (2011), "A Survey of mobile cloud computing", ZTE Communications, Vol. 9, No. 1