An Approach for Offloading in Mobile Cloud Computing to Optimize Power Consumption and Processing Time

## Rakan Aldmour1, Sufian Yousef2, Thar Baker3, Elhadj Benkhelifa1

1*School of Computing and Digital Tech, Staffordshire University, UK*

[*Rakan.aldmour@staffs.ac.uk*](mailto:Rakan.aldmour@staffs.ac.uk), [*E.benkhelifa@staffs.ac.uk*](mailto:E.benkhelifa@staffs.ac.uk)

2*School of Engineering and Built Environment, Anglia Ruskin University, UK*

[*sufian.yousef@anglia.ac.uk*](mailto:sufian.yousef@anglia.ac.uk)

3*Department of Computer Science, University of Sharjah, UAE*

[*tshamsa@sharjah.ac.ae*](mailto:T.baker@ljmu.ac.uk)

Abstract

Smart phones are widely used but they still suffer from constrained resources in term of their processing capabilities, memory capacity and ~~more importantly~~ battery capacity. Mobile Cloud Computing (MCC) has recently emerged to overcome the shortcomings of the standalone smartphones, where Cloud Computing (CC) is leveraged for processing capabilities and memory capacity, while the smartphone would use minimal battery power. However, task offloading from smartphones to the cloud remains an active area of research, to achieve optimized performance and resource utilization ~~as well as~~ and enhance the overall Quality of Service (QoS). In this paper, we propose an approach where two servers are used alternatively, First Upload Round (FUR) offloading and Second Upload Round (SUR) offloading; both supported by a decision engine system. This approach shows better performance and less energy consumption over competing state-of-the-art approaches. The proposed approach shows reduction of the power consumption, 4G is the most improved, for example, for the file size 10 Mb the reduction of the power consumption was 93% on 4G, compared to Wi-Fi 85%.

*Keywords*:Cloudlet; Mobile Cloud Computing; Offloading Power Consumption; Processing Time.

1. Introduction

Smartphones are becoming powerful and easy to use. In fact, they are taking the place ~~instead~~ of normal desktop computers for both works use and personal use. Smartphone’s hardware technology has improved to ~~be able to~~ perform intensive computing tasks rather than basic computation such as image processing, face recognition, heavy calculation, and gaming. These smartphone applications are becoming energy hungry, which means they use more power consumption due to the continuous use as in [1] and [2].

Currently, the number of smart devices such as smartphones and tablets are growing noticeably compared to the computing desktop. Smartphones are small computing devices that have been equipped with features such as camera, user interface, sensors, and powerful processors. These features enabled many beneficial applications which were unexpected earlier. Additionally, it is needed to make them more attractive to customers. The Internet facilitates the connection with users in different ways, such as emails and social networking. This helps users to handle work and share their pictures and video between one another. Nowadays, it is obvious smartphones are playing a crucial role in social networking. As a result, these smartphones are needed to manage and perform different kinds of functionalities, and most of them are known as heavy computing tasks [3]. Notwithstanding, smartphones ~~Mobile devices~~ ~~have unique~~ suffer few limitations:

* Processing capabilities: some smartphones cannot ~~are unable to~~ handle heavy tasks such as face recognition and gaming.
* Limited battery energy: although there were improvements in the smartphone circuits, the battery is still one of the big issues.
* Memory capacity: rich media needs a massive storage and smartphones require big storage to cope with massive data.

The mobile battery is one of the biggest issues, which impacts the development of smartphones due to the following reasons:

* Smartphones ~~are consuming~~ require more energy ~~in excess of~~ than what the embedded batteries capacity ~~are able to~~ provide. This is due to processors becoming more powerful and faster, screens are getting multi-functions and more colorful, and devices are provided with more sensors and actuators. On the other hand, there are only slight improvements in the mobile battery functions; the current battery lifetime, and the future trends are not predictable [4].
* Mobiles were only used for basic computations, such as messages and voicemail. Yet, smartphones are used for intensive computations, videos, and social media. As a result, more power consumption has shortened the battery life. The mobile battery is consumed ~~quickly~~ because of the heavy computations and communications [5][6].

In the last seven years, these limitations have become prevalent among mobile clients, while the mobiles are becoming most popular because of their facilities and functionalities. Due to the powerful operating systems such as Apple iOS, Android, and Windows Mobile, smartphones are capable to perform advanced applications which are almost to the same standard as desktop computer applications. Every mobile device application executes some tasks, of which every task performs a particular computation on a given data [7]. Mobile marketplaces recently are overloaded by offering a variety of popular applications. These kinds of applications increase the amount of computation, communication, and memory resources of mobile devices. These include gaming applications, photo, video editing and mobile client applications because they require heavy processing resources [8]. Yet, mobile applications in these areas remain clearly inferior to desktops in terms of performance and functionality.

Cloud-assisted processing promises to overcome applications limitations which are running on constrained resources devices, such as smartphones by letting them offload processing to more powerful remote devices [9]. This idea of using the pool of powerful remote resources to extend smartphone devices capabilities was introduced in the early 2000s [10]. Moreover, different kinds of offloading systems have been introduced (e.g., MAUI by [11], ThinkAir [12], and CloneCloud is presented in [13]). MAUI work does not tackle the scaling of execution, work was completed on one server, and after processing file it did not keep a copy of the file in the cloud side to process it in future. Also, offloads file as portions could lead to increase the power consumption and processing time or lose the file parts. CloneCloud approach only considers conditions in the offline pre-processing. These tools offer one round of offload to cloud server, while the proposed solution benefits from first round followed by second round of offloading which proved more efficient as shown Section 5 in this paper.

~~Such systems reduce application processing time and power consumption on the mobile devices in different ways such as; partitioning applications among local and remote device, and by processing intensive tasks on the remote side [14].~~

The limitations of smartphone devices as mentioned in several studies ~~such as,~~ [14] [7] [1] [15][16] ~~[47],~~ which are met by Cloud Computing (CC). The rapid evolution of cloud computing and data centers offers new opportunities for covering these limitations. New applications and hardware resources are presented as services through the Internet by several suppliers [17]. The cost-effective and massive scalable resources are accessed on demand~~,~~ to achieve higher reliability, efficiency, and the most important issue is to use unlimited hardware resources. These resources help to remove infrastructure barriers of mobile applications.

The task offloading technique from mobile devices to the cloud server can enhance the computing tasks of mobile devices which, in turn, supports the extension of their battery life with minimum delay. Additionally, this ensures offloading is valuable. New functions in terms of reducing the mobile device battery usage and reduce the processing time will be developed in order to support offloading files from smartphones to the cloud server. The new processes consist of offloading tasks to another server if the first server is busy and save a copy of the file after processing on the server. It also examines the effect on the battery usage and processing time in terms of mobile devices.

The main contributions of the proposed approach are:

1. Offloading tasks to another server if the first server is busy and ~~Save~~ save a copy from the file after processing for the first time ~~to save~~ for saving battery power and reducing the processing time ~~less delay~~. Further details about offloading tasks are available in Section 4.2.
2. Offer a secondary server to serve the mobile users if they get acknowledgment from the main server that it is, as explained in Section 4.3.
3. Examine the server effect on the battery usage and processing time in terms of mobile devices, as in Section 5.
4. Develop a mathematical model to validate the experimental results in detail in Section 3.

The rest of the paper is structured as follows: Section 2 discusses Mobile Cloud Computing topologies. Section 3 Offloading techniques and frameworks, whilst Section 4 formulates a mathematical model for the proposed solution. Experiments set-up and mobile application architecture are discussed in Section 5. Section 6 highlights the benefits of the mobile application and power consumption, processing time scenarios and analyzing the results. Finally, the conclusions and future work are presented in Section 7.

~~Mobile Cloud Computing~~

~~Researchers have managed to merge cloud structure with smartphone devices, in order to control the limitations of smartphone resources to run suitable applications[2]. The new Mobile Cloud Computing (MCC) paradigm incorporates the technology of CC with the mobile environment and enters the IT world since 2009 [18]. The report from Innovative Business Leader Research (IBLR) about the amount of MCC clients, is predicted to increase from 42.8 million in 2008 to 998 million nearly (19% of all mobile clients) in 2014. This means the significant of CC for the users and how they depend on it in their daily life such as; E-mail, GPS, and Social media.~~

~~One of the reasons that MCC has become widely used is because of the complex architecture which aids the execution of processing to overcome the software and hardware mobile device limitations. So, the heavy tasks can be handled on the cloud server. Cloud process the files and sends the results to the smartphone devices[19].~~

~~Mobile Cloud Computing Structure and Components~~

~~MCC is a new technology that aims to provide an intelligent platform and is rich of applications which can be delivered between the user’s fingertips in an efficient way. The processing over the cloud without the intervention and use of the mobile device is called offloading and leads to a more efficient process. This is the key to integration between smartphones and cloud servers. Previous studies such as Mobile Assistance Using Infrastructure (MAUI) which is proposed by [11], and ThinkAir proposed by [12], and CloneCloud presented by [13], computation offloading is the main technology in MCC and it has grown to improve the end user’s experience of mobile applications. Figure 1 below shows the structure of mobile cloud computing:~~

**~~Figure 1 Mobile Cloud Computing Model~~** ~~[20]~~

1. State-of-the-Art of Offloading

In the early years of the 1970s, the offloading system was used at its early development stages. It was first used among servers to find an optimum load of balancing. Offloading can be described in many ways, for instance ~~[21]~~ has defined offloading as the method that can improve the proficiency, quality, or even the performance of some calculation tasks by assigning the servers with an enormous powerful computation to deal with heavyweight data rather than the local servers. Memory, energy consumption, processing, storage, as well as the processing times are examples of the computation factors, where more than one of these factors can be used at the same time ~~[21]~~[22][23].

The new CC resources support the limitations of the problems which faced mobile computing, meanwhile the cloud resources are scalable and are accessible anywhere and anytime in the world [24]. There is a massive number of cloud services such as Amazon, Apple iCloud, and Windows Azure. These cloud providers use their platforms to offer different kinds of services [25]~~[26]~~. To exploit these services, extensive research has been ~~studied~~ conducted on ~~cloud computing~~ CC and mobile networks especially as both are quite dynamic in terms of the infrastructure and services. To merge CC with mobile devices, cloud offloading is one of the distribution computing systems which supports leverage resources involving mobile devices.

*2.1 Offloading Techniques*

Offloading techniques, as described in the previous section, depends on the definition which has been agreed that it is the communication between local and remote machines that is the medium required in order for the offloading to be executed.

Hence, that is why the communication is an essential key for the offloading tasks. However, these communications require local processing, and this would result in too much energy consumption. Network interface, together with the processor, are the biggest pressure on power consumption therefore offloading needs to be aware of the total consumptions and communications cost. There are many benefits of offloading to enhance the performance and save energy [27]. To exploit these benefits, and make the offloading decision more beneficial, there are different factors which should be considered such as file size and network interface.

Offloading has many motivations and advantages that make it an essential to be used as [28], has proposed. To begin with, offloading minimizes the required time of processing and works towards raising the response for the processing. Offloading was initially used to balance the load between the used servers. Finally, offloading can be used as a technique to preserve the energy consumptions and minimize the latency which can be observed clearly from the smartphone’s devices [29][30][31].

The offloading mechanism is classified into three major styles based on the kind of remote contact; these are: ~~as shown below:~~

* The first ~~approach~~ style where they are offloaded to local, this means it could be on the same network or close to it, where the mobile existing~~[32]~~ [20]. The mobile sends the intensive computation to the cloud for execution of ~~the~~ tasks.
* The second ~~approach~~ style offloads to a Web proxy, while the proxy is working as an intermediate between mobile and Web server [33].
* The last ~~approach~~ style is to offload to the cloud where the cloud offers a pool of resources, such as storage and processing to a mobile device [28].

*2.2 Offloading Frameworks*

The offloading technique is one of the hottest and most attractive topics for many researchers. Cloud could be used for multiple, different types of tasks to help smartphones to save energy and time [34]. The offloading problem has been studied by researchers who proposed some of the solutions regarding offloading objectives, approaches, architectures and applications model. The following studies illustrate them:

CUCKOO is a module ~~which~~ used an Android application that has the capability of taking a decision whether to execute part of the task on the mobile or send it to the server. The CUCKOO module was developed by Kemp *et al.* [35], and was a perfect fit with an innovative tool to perform locally or remotely. Although our work follows a system centric approach, for the sake of completeness we discuss some recent state of the art research oriented towards nature inspired approaches to solve the task offloading problem are mentioned in the following. Kaur *et al.* [36]~~[42]~~ proposes a stacking model that works for human activities recognition from wearable and mobile devices. This has a better performance at the expense of more computation time, that is, due to the integration of cuckoo search metaheuristic algorithm. Kun Wang *et al.* [37]~~[43]~~ discusses a number of tasks offloading strategies and evaluates the rationality of task offloading to propose their deep reinforcement learning computing model at the edge in a scenario of Internet of Vehicles (IoV).

The offloading strategy is evaluated using double DQN algorithm which achieves faster convergence. Authors in [38]~~[44]~~ propose a system oriented towards transportation system that dynamically learns the workload allocation policies between the cloud and the fog servers and then converges on the optimal allocation that fulfils the energy and delay requirements of the applicative scenario. The model has shown significant improvement on the battery levels of fog nodes as well as the lowest delay compared to the state of art approaches. Abdel-Basset *et* al. [39]~~[45]~~ proposes three energy-aware models based on the Marine Predators Algorithm (MPA) for tackling the task scheduling in Fog computing to improve the QOSs required by users. The models along with some other meta-heuristic algorithms and genetic algorithms are evaluated based on various performance metrics such as energy consumption, make-span, cost, flow time, and carbon dioxide emission rate Wireless medium, memory, and device type are some examples of the local resources of the smartphones which have a huge impact on processing offloading decisions.

Kovachev and Klamma, [40] propose a model for offloading called Mobile Augmentation Cloud Services (MACS) which permits Android applications to perform over the cloud. The model is based on Android platform and study-based application allocation and decides which parts of the applications to perform locally and remotely over the cloud. The offloading decision is created as an enhancement problem regarding the cloud parameters and the device, such as the connection between the mobile and the cloud, and the remaining energy for the device. In Cuckoo, the framework allows only a static partitioning, whereas MACS permits a dynamic application splitting.

Flores and Srirama discuss in [27] a Mobile Cloud Middleware (MCM) model, which helps to delegate mobile tasks and dynamically assign to the cloud infrastructure. The testbed shows that mobile cloud middleware improves the effectiveness of the quality of service for smartphones and service them in real time responses in mobile cloud applications. MCM also enhances the orchestration and incorporation of mobile tasks allocated with a minimal data transfer.

*2.3 Critical Discussion*

MCC is a good choice to extend the abilities of resources for smartphones because the new computing model has fostered a lot of applications to perform over the cloud. Offloading tasks from smartphones to the cloud is one of the methods used to enhance the computing for mobile devices and to extend the battery life. To make the offloading technique valuable, it is vital to estimate power consumption and processing time whilst completing the communication activities. Therefore, if it is simply that the offloading task uses less power consumption and time than processing tasks on the mobile itself, then the task should offload to the cloud.

Overall, from previous studies such as [35], [41], [42], [43], ~~[44]~~, and[45] ~~[46]~~, it is noted that smartphones have limited resources, and this prevents them from providing an efficient processing system. In addition, the general consensus of the researchers is that battery power needs to be extended which means a reduction in power consumption is required. Different ways were proposed to reduce the power consumption and processing time, such as cloud computing, cloudlet, and virtual machine as this work has been completed by [46],~~[43]~~ and [47].

Jade is a new system developed by [43]. One of the major areas focused on in this research was the categorisation of data by the operating system such as Android and non-Android. This system works on improving the smartphone applications performance as well as reducing the energy consumption. They have been able to prove that their system was capable of decreasing the energy consumption to approximately 39%. The JADE system was designed to be able to handle the offloading tasks within the Android device themselves and also between the Android operating system and non-Android operating system. JADE is a multi-level offloading which means that JADE is capable of offering an optimal method to transmit the loads to the optimal server depends on the performance and the power consumption.

The author who proposed JADE system intended to extend his work by enhancing the model to include cloud platform. Also, he proposed the addition of a fast-wireless link to the experiments such as 4G, for computation offloading. Furthermore, he planned in the future to investigate how cloud could benefit the mobile application. Since it is not crucial to be concerned about the operating system of the phone there is no need to categorize data as Android and non-Android. Instead, the data can simply be classified as text (message and some stored documents) or multimedia (image or videos that are stored on phone or SD card). For example, if Mac or Windows phones are used then it is possible to use the same Gmail service platform easily through it. Also, if the mobile users save some documents then they can save it to Windows phone or Mac. There should be no concern raised about OS because the cloud already has facilities to keep the data compatible on VM or server OS. As a mobile user, there are no requirements to consider the OS compatibility as they are able to use the cloud services. In addition, in their study, the processing was completed by using a normal server to reduce the power consumption, however, tests such as these need to be completed by powerful and scalable servers such as a cloud server. A new system introduced by [43]which illustrated that MCC reduced power consumption significantly, although the author experimented on 3G and Wi-Fi, and concluded that there were limitations to the testing as the experiments were not run on newer protocols (4G). The implication of this is that there is insufficient knowledge of the impact of these factors when using a 4G network connection.

~~Another proposed model called CUCKOO which is created by [35] was used by researchers to check whether to complete processing locally or remotely; in this context they deployed a dummy remote implementation. This highlighted some issues with the cloud; however, it was not a realistic scenario. Therefore, it would be appropriate to suggest that future work should be based on real cloud examples, to provide more accurate and reliable outcomes in terms of delay and power consumption when transferring files.~~

Additionally, in this model, they attempted to save mobile battery power by offloading to a powerful server. Within the model there is a Quick Response (QR) code address for each server, which is provided to the mobile user to scan with their smartphone camera, thus enabling them to connect to the chosen server address. Although this is an innovative idea there are some barriers that the mobile user may experience. For instance, if the mobile user has some issues with their camera function or a low battery, they will be unable to use this function. In addition, this may also increase the power consumption used. It would be more advantageous if the mobile user is not involved in the connection between mobile and server. A preferred method would be for the mobile application to automatically choose an appropriate network interface.Some offloading frameworks such as Active Service Migration (ASM) which is proposed by Shiraz and Gani in [44] ~~[28],~~ complete the experiments using a virtual mobile device to delegate the computational task of the mobile application to offload it to the cloud server. This could cause additional pressure on the cloud server such as management and virtual device deployment as mentioned in Satyanarayanan *et al.* [48]~~[24]~~. It would be suitable if the mobile devices used are real and not virtual. Also, they should be connected to remote services in the cloud server which have powerful resources.

1. Formal Model

Developing a mathematical model for the energy consumption, which is consumed by network activities, is essential to make task offloading ~~useful~~ efficient. The offloading decision is based on the estimation of the power consumption cost, and the file size. This will be modelled mathematically for offloading the files from the mobile device to the cloud and process them.

Channel Quality Indicator (CQI) is vital in depicting the channel information to allocate Proper Modulation and Coding Scheme (MCS). On the other hand, obtaining CQI values to Transmission Time Interval (TTI) can increase the error rate. This means that providing a reliable and accurate CQI with minimum overhead is deemed a complex task as mentioned in [43]. CQI is the information for the user equipment to be sent to the network. It is an indicator which measures the communication quality either good or bad. ~~It suggests:~~

~~The current quality of the communication channel.~~

~~Data to the user equipment which can be obtained and transfer to throughput~~

If the channel quality is poor the user equipment reports that there is a high CQI, the network would send a large transport block size according to the value of the CQI. This occurs because there is a high probability that the user equipment failed to read files. Due to this error in the UE side, there is a negative acknowledgement which is sent to the network, meaning the network has to retransmit the files, which causes consumption to the resources.

The cellular networks such as 3G and 4G, have Radio Resource Controller (RRC) to manage the communication between the mobile user’s devices, for example the UE, and the network. The purpose of the RRC is to offer a high performance for the mobile connectivity by improving the network throughput, and decreasing the UE power consumption, and ~~signalling~~ signaling latency. So, it is noticeable that the RRC has a crucial impact on the power consumption for the smartphone devices. Two cases will affect the latency and the power consumption in different scenarios, one is connected when exchanging the data, which increase the network latency and the link setup ~~signalling~~ signaling. The other has allowed latency but affect the UE battery quickly which means more power consumption. To overcome these two cases, the RRC initiate a trade-off between them and chooses one of them, and this based on different conditions [7].

The structure and specifications for RRC are described and developed in the 3GPP standards. There are ~~two~~ three major states for RRC. ~~the~~ The first is RRC-Connected, the connection data is created where is the dedicated resources are assigned to the UE, in this state the power is high. When the User Equipment (UE) gets a notification from the network broadcast the transition to RRC-Connected has occurred and the data will be received. To create the connection, the UE sends a connection request to the network by the ~~signalling~~ signaling procedures. The second major state is RRC-Idle, where the radio resources are allocated to the UE, and the power state is low. In this state, the UE adjusts to the shared control channel. The third state is FACH, this state is created in particular for the applications which have low data throughput rate [49] [50]. Furthermore, the operation for the aforementioned of the RRC states, the entire energy was consumed whilst transferring the data, which consists of three parts; tail energy, data transfer, and promotion ~~signalling~~ signaling [51] [52].

Universal Mobile Telecommunications System (UMTS) Architecture is a third-generation mobile cellular system for the network, and it is based on GSM standard. In UMTS the key factor that affects the network energy efficiency and application performance is the RRC. This system consists of three components: one is Universal Terrestrial Radio Access Network (UTRAN), second, Core network (CN), and finally, the handset which is the mobile device. TRAN and CN consist of two components which are the node for the base station, and Radio Network Controller (RNC). The purpose of RNC is to govern the element in UTRAN and to control multiple nodes. The central CN is the backbone of the cellular network [50].

There are three states in RRC

* IDLE state (low power): this indicates to a handset when is turned on. ~~UE tunes to control channel counted (control traffic) Connected state: higher power radio+ radio resources tune to UE.~~
* Dedicated Chanel (DCH): the RRC connection is created and handset allocated dedicated transport channel for uplink and downlink.
* Forward Access Chanel (FACH): is an intermediate state between IDLE and DCH

~~The aforementioned states will be discussed in the following section:~~

The connection occurs when there is a chance to send while the buffer is low. Hence the request is sent for the connection through the ~~signalling~~ signaling procedures. When the mobile device is switched on, this means IDLE state by default. Until this stage, the mobile device does not establish an RRC connection with the RNC. Thus, no radio resource is assigned, and the mobile device could not send any user data. At this stage, the power consumption of the mobile radio interface is nearly zero at ADLE state. For the DCH state, the connection is established at the RRC state and the mobile device is generally allocated and dedicated. The radio power consumption for DCH is the highest which is (600 to 800 mW). The last state is FACH, and in this state the RRC link connection is established but there is a low speed for both uplink and downlink. This state is created for an application with low data throughput rate [50]. State Transitions: there are two kinds of RRC state transitions, which are discussed by [45] as follows:

* Promotion State: (IDLE→DCH, FACH→DCH, and IDLE→FACH).
* Demotion State: going in reverse directions.

Promotions (demotions) switch from a state with lower (higher) radio resource and radio power consumption to another state requiring more (less) radio resources and power. ~~Figure~~ Fig. 2 below shows the RRC state mechanism for Promotions and Demotions.

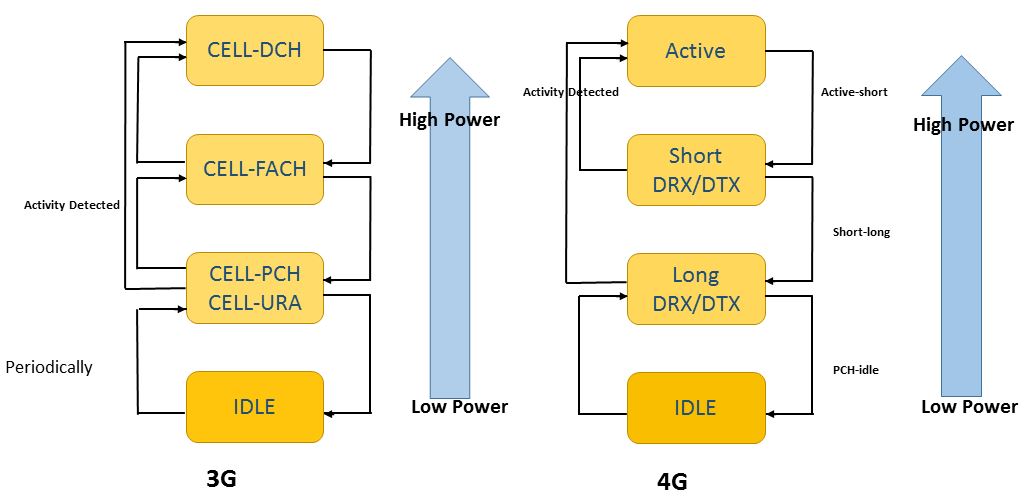


Figure 1: 3G and 4G RRC state mechanism

Form ~~Figure~~ Fig. 1 the total energy which is consumed in RRC state consists of:

Promotion ~~signalling~~ signaling energy

Data transfer energy

Tail energy

General energy is

[7]  *(1)*

*Where*

PS: ~~signalling~~ signaling Power

Etrx: Data Energy

Email: Tail time Energy

~~Etail: Tail Energy~~

*Energy= Power \** ~~x~~ *time*

Where *Power* is smartphone power in Watt and Time is smartphone time in second

(2)

~~Eps, Etail are constants for each smartphone; hence, Eps and Etail are computed separately.~~

For

*Ptrx* is constant as power, but *Ttrx* is variable depending on the file size ~~of data~~ (F) and the data rate as:

*= F (amount of data in bits) ÷ Rtrx* (3)

*Example: = (1000 bytes \** ~~x~~ *8) ÷ 2Mb*  (4)

In wireless networks, we have limited resources (low spectrum), high error rate and higher delay compared to wired, hence we work hard to increase spectrum and reduce delay, due to the BER limitation. New protocol ARQ and hybrid AQR are useful to protect the lower layers from received errors. Transfer Control Protocol (TCP) is an end-to-end protocol and hence is limited by the delay.

The achieved data rate Rtrx= min (Rtcp, R3G/4G), as Rtcp and (R3G / 4G) are limits of the rate due to TCP and wireless network. (TCP rate = CWD (Congestion window) / RTT), but (R3G / 4G) is the rate achieved at TCP layer which is limited by the rate of the lower layers (physical, MAC, PDCP), which is adaptive by ~~maximising~~ maximizing spectrum ~~utilisation~~ utilization. Adaptation is implemented for each ~~(TTI)~~ Transmission Time Interval (TTI). In adaptation, different modulation and coding systems are used using the (RSS) received signal strength and S/N. the receiver is a source of information by sending to TX, the current challenge condition using Channel Quality Index (CQI), which is calculated from RSS \* ~~x~~ S/N. Then, Tx selects the MCS according to the mapping from CQI and the user equipment (UE) energy to Modulation and Coding Scheme. At slow state there is high power consumption at Limited Window size (IWD) until it reaches CWD. The mobile uses too much energy at this stage of TCP slow-start hence,

*(5)*

FSS: data transferred during the slow start.

TSS is the time of slow start before it reaches CWD from (IWD) window size. Where

*TSS = 0.5 ms \** ~~x~~ *3.5 = 1.75 ms. (6)*

*TCP: data size = 1.5KB \** ~~x~~ *1000 \** ~~x~~ *8 = 12000 bits. (7)*

*Fss = Tcp size \** ~~x~~ *3.5 = 12000 \** ~~x~~ *3.5 = 42000 bits (8)*

*rx= 10 ÷ (. 5 \** ~~x~~ *10) = 20000 b/s (9)*

*(10)*

Where log2 (CWD/IWD) is the expo growth of the window size which is usually 2

For examples

(11)

(12)

*1.5 \** ~~x~~ *1000 \** ~~x~~ *8= 12000 bits*

*log2 (2/1) = 21*

*FSS=12000 \** ~~x~~ *1= 12000 bits*

(13)

(14)

Energy 3G, and 4G = Ptrx *\** ~~x~~ Time

Energy 3G, and 4G = 12 (watt) *\** ~~x~~ 5.94 = 71.28 Joules

1. Methodology

The purpose of this stage of experimentations is to address and enhance the QoS for the mobile users, such as mobile battery and reduce delay. To enable this to be investigated, the following objectives will be implemented to process the tests:

* + - 1. Offer a secondary server to serve the mobile users if they get acknowledgment from the main server that it is busy.
      2. Save a copy from the file after processing for the first time to save battery and for less delay.

*4.1 Experimental Setup Tools*

To conduct the experiments for processing files there are two perspectives, client and server. From the client’s viewpoint, the smartphone is required to install the deployed mobile application. Samsung Galaxy Tab 4 T335 will be used in each scenario, the specifications are: processor core Quad-core (4 Core), operating system Android 5.1.1, wireless LAN standard IEEE 802.11a, and flash memory capacity 16 GB. After the smartphone is ready for the use, files 1 Mb, 3 Mb, 5 Mb, 9 MB, 10 Mb, 15 Mb, 20 Mb, and 40 Mb, are required to be copied on the mobile SD card. Subscriber identification module (SIM) card is necessary to gain access to the Internet using cellular generation 4G which will be used by the mobile device. After these outlined elements are in situ, the mobile users are ready to commence the testing processes.

~~From the perspective of the server part~~ From the server side, there is a dedicated server from Google cloud with high specifications as follow; high memory machine type with 8 virtual CPUs and 52 GB of memory, 8 virtual CPU, 52GB memory, and 64TB HDD

In addition, several types of tools such as NetBeans, and Java are required to be installed to support the running of the program. After both parts are equipped, mobile users have two options to gain access to the Internet through either Wi-Fi or cellular data. In these experiments for Wi-Fi network, Anglia Ruskin University (ARU) wireless network will be used, which is called ‘*eduroam*’ network. The closest access point to the test base is nearly 4-5 meters.

~~Also 4G used in the experiments, the network that was chosen was Vodafone Company, because it uses two different frequencies as shown below:~~

* ~~The first is 2100 MHz and is on band one. The benefits from this band is that the 4G has Long Term Evolution (LTE) which means a high-speed network available.~~
* ~~The second is frequency is 800 MHz and it is on band 20. The value of this frequency is that it penetrates a good signal into buildings~~

~~The two frequencies combined to provide an overall better coverage specifically within buildings where the experiments scenarios were conducted. Compared to other companies, for example EE, they only use one frequency which 2800 MHz which is overall better speed of 4G, but they do not have the benefits of the second frequency which aids 4G signal within buildings.~~

~~After reviewing the available telecommunication companies, it was deemed that Vodafone has the best coverage among them on the ARU campus.~~

*4.2 Mobile Application Architecture*

The developed application~~, which is used in the previous chapters,~~ will be used in this ~~chapter~~ paper but ~~w[35]ith~~ the new functions to fulfil the aims. The new features could help in reducing the power consumption and with minimum delay. Due to the fact, there is a secondary server to support the processing whilst the primary server is occupied with another request from mobile users. In addition, there is a development of another service which duplicates the file on the server to prevent uploading the files for a second time, which is expected to reduce the delays in processing time and the limitations to the battery.

The mobile application design consists of two main parts as shown in Fig. 2:

Mobile Application: this application is installed on the smartphone. It consists of two parts in terms of the deploying (layout and class activities) which are as follows:

* Main Activity: contains a method called on-create to establish the activities
* Connectivity: the purpose of this is to check the connection type, either 4G or Wi-Fi.
* File array: this help helps to view the files as a list in the smartphone.
* File chooser: enables the mobile users to select the file for processing.
* Get file: take the selected file to commence processing either on the mobile or on the cloud.
* Receive: supports to demonstrate the results on the mobile screen.

Server Application: this is located on the cloud server. Different types of tools such as NetBeans and Java were installed on the server to support and facilitate the processing of files. After finishing the processing, the application server sends the results back to the mobile users.

Diagram, schematic

Description automatically generated

Figure 2: Various components of the proposed two server-based offloading

*4.3 Power Measurement Process*

Each smartphone has different power consumption properties, as well as requiring different kinds of power models to estimate the battery drainage. Smartphones do not contain these types of models to measure the accurate power consumption. A new power measurement process was developed from Android Debug Bridge (ADB). ADB is a command-line tool which allows users to communicate with a device either a connected Android device such as Samsung and, HTC or an emulator. The ADB command helps in a variety of device actions. In this ~~research~~ paper, it will be used to estimate the precise power consumption. ADB command works as a client-server program, this will be divided into three components:

* Client: ~~In~~ in this part, the commands will be sent. The client runs on the development machine. The client can send a request by issuing an ADB command, using Command Prompt Command CMD.
* A daemon (adbd): which allows the commands to run on a device. The daemon operates as a background process for each mobile device.
* Server: This part helps to manage the communication between the client and the daemon.

After performing the measurement processes on *cmd*, the text file is ready and shows ‘*BatteryRults.txt’*, it contains the estimated power separately for each application that the mobile users used at the same time, in detail. The most important factor in the text file in relation to this research is the battery measurement for the Mobile Energy Cloud Computing algorithm (MECCA) application.

*4.4 Experiments Scenarios*

During the experimental process, it is important to obtain the results many times. In this paper, around 140 experiments will be conducted at a random time. The variables that will be considered during the experiments will be the network interface, time of experiments, file size, and mobile battery usage. Each file will be tested 5 times using the Wi-Fi and 5 times using the 4G, during offloading files from the mobile device to the cloud. This means, that each file will be tested 20 times, 10 times for the First Uploading Round (FUR) and 10 times for the SUR. The reason for this is that it is important to provide an average of the results which will reflect and validate the outcomes of the proposed mechanism.

In the first time of offloading, ~~First Uploading Round (FUR)~~ FUR, the server is completely empty, this means there is no saved file on it. Once the mobile user sends the file, a new copy of it will be saved on the Google cloud and it is ready to process. Second Uploading Round (SUR) technique supports to save the offloaded files on the cloud server. This could effectively help to reduce the mobile battery usage with minimum delay, because the file is saved on the server during the FUR. After processing the file on the server, the battery usage and the time will be measured. In addition, the results of processing files will send back to the mobile user, and it will be shown on the mobile device screen. Table .1 below illustrates the scenarios for FUR and SUR for a variety of file sizes and network interfaces on the cloud.

Table 1: Scenarios for FUR and SUR

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| File Size (Mb) | Scenarios | Location | Network Interface | FUR | SUR |
| 1 | S1 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 3 | S2 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 5 | S3 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 10 | S4 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 15 | S5 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 20 | S6 | Cloud | Wi-Fi, 4G | 10 | 10 |
| 40 | S7 | Cloud | Wi-Fi, 4G | 10 | 10 |

The mobile user chooses one of the network interfaces either 4G or Wi-Fi. Once the connection is chosen, the mobile user browses the available files on SD card which is located on the mobile storage. After the file is chosen to be offloaded to the cloud, the processing commences. After a while, the results will be shown on the mobile screen with the measurement of the processing time.

The mobile battery usage measurement is essential, this process was discussed in detail in previous section. Once FUR is completed, the experiments will be repeated for the SUR. After SUR is completed, the used file needs to be removed from the server before the 4G experiments are conducted. This can be completed by copying the following link on the browser:

*http:// 104.154.110.104/ RAKAN\_WEB/ DELETE\_FILES? SURE=yes.*

This step needs to be repeated when changing the network interfaces from Wi-Fi to 4G or the opposite.

1. The Beneficial Aspects of MECCA Application

The goal of MECCA is to increase the benefits for the mobile users when the files are required to be offloaded from smartphone devices to Google cloud. In this part, Mecca components will be presented to demonstrate how they combine into one architecture structure for creating a mobile application.

The developed MECCA model support end users to decide where to offload the files for remote execution, which is a cloud server, additionally, selecting the suitable connection. MECCA uses its ~~optimisation~~ optimization processing to make the decision whether to offload or not.

In MECCA model, each function is responsible to estimate and reduce the cost of offloading, such as the file, which is required to be transferred, network interfaces, availability of the servers, and the existence of the file on the server. All these variables will be used to formulate the benefits of offloading or to execute locally on the smartphone.

As mentioned earlier, the MECCA model supports two options: 1) process the files locally on the smartphone which sometimes could incur more energy consumption and delay. 2) Send the files over the Internet to the cloud server which is already appointed for MECCA application.

Subsequently, when the MECCA is ready to process the files on both sides, end users should select the file, processing location, and network connection type. SUR technique is expected to support the reduction of the battery usage and the time delay. For example, when the client offloads the file to Google cloud to process it, the results of the processing will be sent back to the client over the Internet. If the mobile user needs to process the same file later, it will affect the mobile battery usage and increase the time delay. In this case, one of the proposed functions of the server part called ‘save files’ supports the model to save a copy of the file after processing. When there are additional uploads of the same files by the mobile users, the application will read the file and check with the server. Within the server part there is a process called ‘checking saved file’ to compare the selected file from the mobile with the saved files on the backup of server one and server two as illustrated in ~~Figure~~ Fig. 3. This will benefit the mobile users as the file will not be required to be uploaded again as its already exists on the server, and therefore if the file does not have to be re-uploaded it could save the mobile battery and the time.

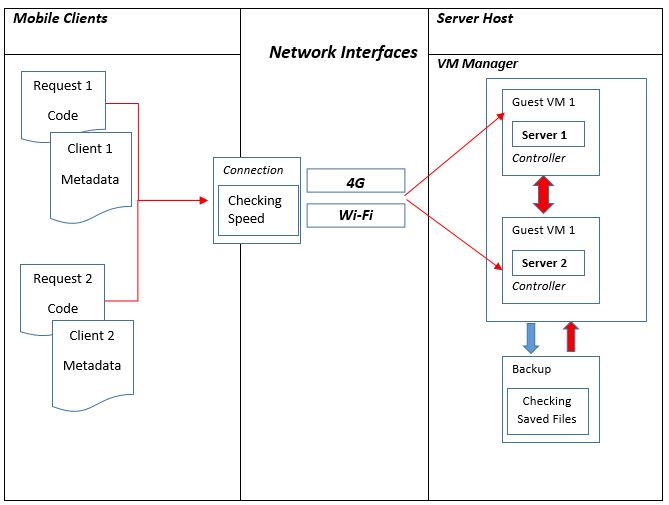


Figure 3: Mobile Application Architecture Components

According to the quality of service in terms of the delay, another server will be used for the experiments in this ~~chapter~~ paper. The benefit from this, is to serve a big number of incoming requests from the mobile users. When the request is sent to the server and if the server is equipped with another request, the new request automatically will be transferred to the second server. In this case, there is no delay for the mobile user to wait until the main server finishes processing the file. Additionally, it should also not miss the request from the clients, so the smartphone could save time and battery as it transfers the requests directly to the second server. The previous work is organized by the controller which is located in the server part as shown in Fig. 3.

*5.1 Decision Engine System*

The decision engine is a tool that requires input from users which is processed using a specific pre-established criteria or formula to direct the outcome to acquire the optimum solution. In MECCA model, there is a Decision Engine (DE) to find out whether to offload tasks remotely or to process it locally. Inside the DE there are conditions to take the suitable choice as shown below:

* File size: is one of the most important factors that could affect the smartphones resources whilst processing, offloading, and editing. So, it is valuable to take it into account. ~~Here in~~ In this ~~research~~ paper, it is paramount to consider file size in conjunction with the connection type.
* File name: it is one of the new features which will be added to MECCA, to compare the selected file with the stored files on the server. In order to make the decision of whether it needs to be uploaded or taken from the previously uploaded files.
* Last Modified: for each file, there are properties, one of these is the last modification, which shows date and time for the most recent updates on the file, for example, 05 ‎**June** ‎2017, ‏‎**15:48:49**.
* Connection type: as discussed in previous sections, for each network connection such as 3G, 4G, and Wi-Fi, there is an amount of power consumption and delay, which would have an impact because of the file size. So, it is significant to consider the connection type juxtaposed with the file size.

*~~Experiments Results~~*

~~In the following experiments scenarios, 120 tests were executed for offloading files to cloud server to check and reduce the power consumption for smartphone devices. The tables and figures below will illustrate and discuss the results in terms of mobile battery usage and processing time.~~

*5.2 Power Consumption Results*

In the following experiments scenarios, 120 tests were executed for offloading files to cloud server to check and reduce the power consumption for smartphone devices. The tables and figures below will illustrate and discuss the results in terms of mobile battery usage and processing time. The tables and figures below illustrate the results for the power consumption whilst sending and receiving files between the cloud and the mobile device using Wi-Fi and 4G connections in Milliampere hour (mAh). The results will be for FUR and the proposed mechanism SUR and to investigate the benefits of it. Tables and figures show the results for files sizes, 1 Mb, 3 Mb, 5 Mb, 10 Mb, 15 Mb, 20 Mb, and 40 Mb for both rounds.

Table 2 below illustrates the average of the usage of the mobile battery, while offloading files to Google cloud for different file sizes and for receiving the results of processing back. ~~These results can also be seen visually in figure 6.9.~~ The average for each file was obtained in relation to the offloading for both rounds, using 4G and Wi-Fi network interfaces. From the table below, there is a big difference in the average of the mobile battery usage between FUR and SUR. For example, the average of the mobile battery usage for file size 10 Mb in the FUR when using Wi-Fi was 0.3922 mAh, compared to the same file size and network connection but in the SUR the average was 0.05946 mAh. For 4G connection, for example file size 20 Mb, the average of the battery usage for FUR was 1.3408 mAh but in the SUR was 0.0659 mAh. The previous two examples illustrated the difference between FUR and SUR.

Table 2: Average of the power consumption whilst using Wi-Fi and 4G connections

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Wi-Fi | | 4G | |
| File Size (Mb) | FUR (mAh) | SUR (mAh) | FUR (mAh) | SUR (mAh) |
| 1 | 0.18182 | 0.03604 | 0.0574 | 0.03396 |
| 3 | 0.2674 | 0.0462 | 0.21402 | 0.037176 |
| 5 | 0.3636 | 0.051 | 0.3076 | 0.03986 |
| 10 | 0.3922 | 0.05946 | 0.6202 | 0.0425 |
| 15 | 0.4668 | 0.06816 | 0.6558 | 0.05898 |
| 20 | 0.547 | 0.08176 | 1.3408 | 0.0659 |
| 40 | 0.6328 | 0.124 | 2.638 | 0.1468 |

Table 3 illustrates the percentage of the reduction of the power consumption whilst offloading files to the cloud server with the new mechanism SUR. There is a significance in the reduction for the power consumption in both connections Wi-Fi and 4G when using SUR mechanism. For example, when offloading the file size 5 Mb, the power consumption is reduced in both network connection Wi-Fi and 4G to 86% and 87% respectively.

Overall, from tables 2 and 3, the results illustrated that the new mechanism SUR supports the reduction in the power consumption by more than the triple in some file size. In addition, the average of the reduction of power for SUR in 4G connection is more efficient compared to SUR in Wi-Fi connection. This can be seen in files 3 Mb, 5 Mb, 10 Mb, 15 Mb, and 20 Mb. The outcome of these results reflect that it is crucial to use the 4G connection in the SUR, because it supports to save the mobile battery.

Table 3: Average of the reduction for the power consumption for FUR and SUR

|  |  |  |
| --- | --- | --- |
| **File Size (Mb)** | **FUR and SUR using Wi-Fi** | **FUR and SUR using 4G** |
| 1 | 80% | 41% |
| 3 | 83% | 83% |
| 5 | 86% | 87% |
| 10 | 85% | 93% |
| 15 | 85% | 91% |
| 20 | 85% | 95% |
| 40 | 53% | 95% |

~~Figure 4 below demonstrates the average for the power consumption in mAh whilst processing files from different sizes on the cloud. The network interfaces were Wi-Fi and 4G. For Wi-Fi connection, it is noticeable that the increase in the power consumption was gradual while increasing the file size. On the other hand, the increase in the power consumption for 4G connection was steady for files 1 Mb, 3 Mb, 5 Mb, and 10 Mb. At 15 Mb, there was a sharp increase for all files which is clearly illustrated in figure 8. For example, in file size 15 Mb the power consumption was .655 mAh compared to file size 40 Mb which was 2.4024 mAh, this means, there is an increase of more than triple. The benefit from the SUR can be seen in figure 8 for both connections, Wi-Fi and 4G. The reduction in the power consumption for the SUR was better in 4G connection than Wi-Fi. For example, in Wi-Fi for file size 5 Mb the power consumption average was 0.051 mAh compared to the same file size in 4G which was .03986. A further example, for file size 10Mb was .05946 mAh, .0425 mAh, respectively.~~

**~~Figure 4 Power consumption for processing files whilst using Wi-Fi, 4G~~**

~~Figure 5 below illustrates the average for the power consumption for FUR and SUR whilst offloading to cloud server using Wi-Fi connection. The results show there is a significant enhancement in the reduction of the power consumption. For example, the enhancement for file size 10 Mb was more than four times in the SUR. In FUR the average for power consumption was .3922 mAh but in the FUR was .05946 mAh. As discussed in figure 8, the average of the increase in the power consumption in the FUR was gradually increased with the file size, but in the SUR was slightly increased for the entire file sizes, except from the file size 40 Mb which was noticeably increased compared to file size 20 Mb.~~

~~Figure 5 Power Consumption when using Wi-Fi in FUR and SUR~~

~~Figure 6 illustrates the average of mobile battery consumption for FUR and SUR whilst the various sized files were sent to the Google cloud using the 4G connection. In FUR, the increase of the file size had a significant impact on the power consumption, but it was not stable. For the SUR, it is clear there is a substantial reduction in the power consumption, for example, file size 20 Mb in the FUR was 1.3408 mAh and for SUR was 0.0659 mAh. The increase in the power consumption was slightly apparent for all file sizes.~~

~~Figure 6 Power Consumption when using 4G in FUR and SUR~~

~~Figure~~ Fig. 4 shows the difference between Wi-Fi and 4G connection in terms of the reduction of the power consumption in SUR mechanism. For both connections, it is evident the increase in the power consumption was stable apart from file size 40 Mb. 4G connection is slightly lower than Wi-Fi, this means it has saved the mobile battery. It is noticeable there is a rapid increase in the consumption of the power between file size 20 Mb and 40 Mb in both connections.

Figure 4: Power Consumption when using SUR for 4G and Wi-Fi

~~From the previous figures 3,4,5, and 6~~ It is noticeable that there is a substantial saving in the power consumption whilst using the new technique SUR in both network interfaces, Wi-Fi and 4G technology. So, in this instance, the smartphones users will save more of the mobile battery while processing the files on the cloud using SUR. In addition, it is better to use the 4G connection in the SUR to save more battery as the results have shown.

*5.3 Processing Time Results*

Another vital parameter has been measured using the deployed mobile application. The parameter is processing time for executing several files on the cloud server. The tables and figures below present the results for sending the files and processing them, then receiving the results back from the cloud server. The results will be presented for both FUR and SUR in a millisecond for both network connections, Wi-Fi and 4G. Moreover, there will be a further comparative analysis between the Wi-Fi and 4G network interfaces.

Table 4 below shows the average processing time for offloading files over the cloud. The processing time included the time taken to, upload, process, and send the results back to the mobile device. As discussed previously, there are two rounds FUR and SUR, in table 4, the processing time will be provided for both to evaluate the significance of the proposed SUR. From the average, it is noticeable for all file sizes that there is a large significance in the reduction of the proccing time.

Table 4: Processing time for FUR and SUR using Wi-Fi

|  |  |  |
| --- | --- | --- |
| File Size (Mb) | FUR- 4G (Millisecond) | SUR- 4G (Millisecond) |
| 1 | 3745 | 640.4 |
| 3 | 6696.4 | 793.2 |
| 5 | 10013 | 875 |
| 10 | 20573.2 | 1335.4 |
| 15 | 19529.2 | 1698 |
| 20 | 34861 | 2315 |
| 40 | 74307.6 | 3398.2 |

Table 5 below presents the processing time for sending files and returning the results back to the mobile users in milliseconds. As discussed in table 4, there is a significant improvement in using SUR, this can be seen in the table below for all of the file size scenarios. For example, in the File size 5 Mb, the processing time in FUR was 10013 milliseconds compared to SUR was 875 milliseconds.

Table 5: Average of processing time for FUR and SUR using 4G

|  |  |  |
| --- | --- | --- |
| **File Size (Mb)** | **FUR- Wi-Fi (Millisecond)** | **SUR- Wi-Fi (Millisecond)** |
| 1 | 2674.4 | 345.8 |
| 3 | 4224.8 | 597.8 |
| 5 | 5601.4 | 702.8 |
| 10 | 6124.6 | 1302.8 |
| 15 | 9700.8 | 1818.2 |
| 20 | 10960.2 | 1979.8 |
| 40 | 12814.6 | 3407.6 |

~~Figure 8 illustrates the average of the processing time for various file sizes. There is an increase in the processing time in both rounds, but in SUR it is noticeable the increase is slight. From the figure below, there is a significant value in the reduction of the processing time in the SUR. For example, in file size 20 Mb, the FUR was 10960.2 Millisecond compared to SUR was 1979.8 Millisecond, this means the time is reduced by more than five times.~~

~~Figure 8 Processing time for FUR and SUR using Wi-Fi~~

~~Figure~~ Fig. 5 below shows the results of offloading files to the cloud server. The results illustrate the large difference between FUR and SUR, for example for file size 40 Mb the FUR was 74307.6 milliseconds and for the SUR was 3398.2 milliseconds. It is clear in the SUR that the processing time was less than the time in FUR. This proves the value of developing the new mechanism which is SUR.

Figure 5: Processing time for FUR and SUR using 4G

Table 6 below illustrates the average of the processing time for the various files’ sizes in milliseconds. The processing time is for two network interfaces 4G and Wi-Fi, this included sending, processing, receiving results from mobile to cloud for both FUR and SUR. Overall, there is a significant reduction in processing time whilst using SUR in offloading for 4G and Wi-Fi, but it is clear in Wi-Fi connection the processing time is less compared to 4G. For example, for file size 1 Mb, the processing time was 345.8 milliseconds in Wi-Fi, whilst in 4G was 640.4 milliseconds.

Table 6: Average of processing files for FUR and SUR using Wi-Fi and 4G

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Wi-Fi | | 4G | |
| File Size (Mb) | FUR (Millisecond) | SUR (Millisecond) | FUR (Millisecond) | SUR (Millisecond) |
| 1 | 2674.4 | 345.8 | 3745 | 640.4 |
| 3 | 4224.8 | 597.8 | 6696.4 | 793.2 |
| 5 | 5601.4 | 702.8 | 10013 | 875 |
| 10 | 6124.6 | 1302.8 | 20573.2 | 1335.4 |
| 15 | 9700.8 | 1818.2 | 19529.2 | 1698 |
| 20 | 10960.2 | 1979.8 | 34861 | 2315 |
| 40 | 12814.6 | 3407.6 | 74307.6 | 3398.2 |

Table 7 shows the average of reduction of the processing time whilst offloading files to Google cloud. The average of the processing the files was for FUR and SUR when using Wi-Fi and 4G connections. From the table it is clear, that there is a large reduction in the processing time when using SUR. For example, the average of the saving time for the file size 10 Mb was 94% whilst using 4G connection compared to Wi-Fi which was 79%. Overall, this means using the 4G connection in the SUR is more efficient to save the time compared Wi-Fi.

Table 7: Average of reducing the processing time using Wi-Fi and 4G

|  |  |  |
| --- | --- | --- |
| File Size (Mb) | FUR and SUR using Wi-Fi | FUR and SUR using 4G |
| 1 | 87% | 83% |
| 3 | 86% | 88% |
| 5 | 87% | 91% |
| 10 | 79% | 94% |
| 15 | 81% | 92% |
| 20 | 82% | 93% |
| 40 | 74% | 95% |

1. Conclusion

There was a need to have a second server and to save files on the cloud. To link these features with network interfaces to investigate if one of them was more valuable while the file is already on the server and to process over the cloud, known as SUR. In addition, the fact that there is the availability of a second server means that the need to re-upload due to ‘server busy’ is eliminated and therefore automatically limits the negative impact on the battery and processing time.

The results were reviewed and analyzed and indicated that, through the implementation of the two new features, battery usage and processing time were improved whilst using both network interfaces Wi-Fi and 4G. This was recorded to be more than 80% for the SUR compared to FUR in most cases. ~~For example, when the file size was 3Mb, the reduced power consumption for the SUR is 83%. For processing time, there is a significant decreasing for example, in the file 10 Mb, the time is reduced to 94% when using the proposed technique.~~

The results illustrate that overall, during FUR, Wi-Fi was more efficient in reducing power consumption for big files apart from files 1 Mb, 3 Mb, and 5 Mb. In contrast, in relation to power consumption, 4G is the most improved whilst using the new feature, SUR. For example, for file size 10 Mb the reduction of the power consumption was 93% on 4G, compared to Wi-Fi which was 85%. However, the performance for Wi-Fi in FUR was significantly lower than 4G, which shows that although the improvement between two rounds was 93% for 4G, it still used more battery than Wi-Fi. In addition, this pattern of efficiency and improvement for the network interfaces are parallel for processing time.

References

[1] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, “Optimal joint scheduling and cloud offloading for mobile applications,” *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 301–313, Apr. 2019, doi: 10.1109/TCC.2016.2560808.

[2] H. Atre, K. Razdan, and R. K. Sagar, “A review of mobile cloud computing,” in *Proceedings of the 2016 6th International Conference - Cloud System and Big Data Engineering, Confluence 2016*, 2016, pp. 199–202, doi: 10.1109/CONFLUENCE.2016.7508113.

[3] M. Al-Khafajiy, T. Baker, A. Waraich, D. Al-Jumeily, and A. Hussain, “Iot-fog optimal workload via fog offloading,” in *Proceedings - 11th IEEE/ACM International Conference on Utility and Cloud Computing Companion, UCC Companion 2018*, 2019, pp. 349–352, doi: 10.1109/UCC-Companion.2018.00081.

[4] M. Lauridsen, P. Mogensen, and L. Noël, “Empirical LTE smartphone power model with DRX operation for system level simulations,” in *IEEE Vehicular Technology Conference*, 2013, doi: 10.1109/VTCFall.2013.6692179.

[5] N. Moghadasi, M. G. Arani, and M. Shamsi, “A Novel Approach for Reduce Energy Consumption in Mobile Cloud Computing,” *Int. J. Inf. Technol. Comput. Sci.*, vol. 7, no. 11, pp. 62–73, Oct. 2015, doi: 10.5815/ijitcs.2015.11.08.

[6] B. Khargharia *et al.*, “Autonomic power & performance management for large-scale data centers,” in *Proceedings - 21st International Parallel and Distributed Processing Symposium, IPDPS 2007; Abstracts and CD-ROM*, 2007, doi: 10.1109/IPDPS.2007.370510.

[7] M. Altamimi, A. Abdrabou, K. Naik, and A. Nayak, “Energy Cost Models of Smartphones for Task Offloading to the Cloud,” *IEEE Trans. Emerg. Top. Comput.*, vol. 3, no. 3, pp. 384–398, Sep. 2015, doi: 10.1109/TETC.2014.2387752.

[8] H. F. Sheikh, I. Ahmad, Z. Wang, and S. Ranka, “An overview and classification of thermal-aware scheduling techniques for multi-core processing systems,” *Sustainable Computing: Informatics and Systems*, vol. 2, no. 3. Elsevier Inc., pp. 151–169, 01-Sep-2012, doi: 10.1016/j.suscom.2011.06.005.

[9] P. Kendrick, T. Baker, Z. Maamar, A. Hussain, R. Buyya, and D. Al-Jumeily, “An Efficient Multi-Cloud Service Composition Using A Distributed Multiagent-based, Memory-driven Approach,” *IEEE Trans. Sustain. Comput.*, pp. 1–1, Nov. 2018, doi: 10.1109/tsusc.2018.2881416.

[10] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H. I. Yang, “The case for cyber foraging,” in *Proceedings of the 10th Workshop on ACM SIGOPS European Workshop, EW 10*, 2002, pp. 87–92, doi: 10.1145/1133373.1133390.

[11] E. Cuervoy *et al.*, “MAUI: Making smartphones last longer with code offload,” in *MobiSys’10 - Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, 2010, pp. 49–62, doi: 10.1145/1814433.1814441.

[12] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, “ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading,” in *Proceedings - IEEE INFOCOM*, 2012, pp. 945–953, doi: 10.1109/INFCOM.2012.6195845.

[13] B. G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, “CloneCloud: Elastic execution between mobile device and cloud,” in *EuroSys’11 - Proceedings of the EuroSys 2011 Conference*, 2011, pp. 301–314, doi: 10.1145/1966445.1966473.

[14] Q. Qi, J. Liao, J. Wang, Q. Li, and Y. Cao, “Dynamic resource orchestration for multi-task application in heterogeneous mobile cloud computing,” in *Proceedings - IEEE INFOCOM*, 2016, vol. 2016–September, pp. 221–226, doi: 10.1109/INFCOMW.2016.7562076.

[15] B. Gao, L. He, and S. Jarvis, “Offload Decision Models and the Price of Anarchy in Mobile Cloud Application Ecosystems,” *IEEE Access*, pp. 1–1, Jan. 2016, doi: 10.1109/access.2016.2518179.

[16] L. Tawalbeh, Y. Haddad, O. Khamis, E. Benkhelifa, Y. Jararweh, and F. AlDosari, “Efficient and secure software-defined mobile cloud computing infrastructure,” *Int. J. High Perform. Comput. Netw.*, vol. 9, no. 4, pp. 328–341, 2016, doi: 10.1504/IJHPCN.2016.077825.

[17] X. Xu, “From cloud computing to cloud manufacturing,” *Robot. Comput. Integr. Manuf.*, vol. 28, no. 1, pp. 75–86, Feb. 2012, doi: 10.1016/j.rcim.2011.07.002.

[18] D. Dev and K. L. Baishnab, “A review and research towards mobile cloud computing,” in *Proceedings - 2nd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, MobileCloud 2014*, 2014, pp. 252–256, doi: 10.1109/MobileCloud.2014.41.

[19] J. Wang, J. Peng, Y. Wei, D. Liu, and J. Fu, “Adaptive application offloading decision and transmission scheduling for mobile cloud computing,” in *2016 IEEE International Conference on Communications, ICC 2016*, 2016, doi: 10.1109/ICC.2016.7510721.

[20] M. Shiraz, A. Gani, R. H. Khokhar, and R. Buyya, “A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing,” *IEEE Commun. Surv. Tutorials*, vol. 15, no. 3, pp. 1294–1313, 2013, doi: 10.1109/SURV.2012.111412.00045.

[21] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, “A Context Sensitive Offloading Scheme for Mobile Cloud Computing Service,” in *Proceedings - 2015 IEEE 8th International Conference on Cloud Computing, CLOUD 2015*, 2015, pp. 869–876, doi: 10.1109/CLOUD.2015.119.

[22] R. Aldmour, S. Yousef, M. Yaghi, S. Tapaswi, K. K. Pattanaik, and M. Cole, “New cloud offloading algorithm for better energy consumption and process time,” *Int. J. Syst. Assur. Eng. Manag.*, vol. 8, no. 2, pp. 730–733, Nov. 2017, doi: 10.1007/s13198-016-0515-2.

[23] T. Baker *et al.*, “GreeAODV: An Energy Efficient Routing Protocol for Vehicular Ad Hoc Networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 10956 LNAI, pp. 670–681, doi: 10.1007/978-3-319-95957-3\_69.

[24] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing,” *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016, doi: 10.1109/TNET.2015.2487344.

[25] M. Zhang *et al.*, “An Infrastructure Service Recommendation System for Cloud Applications with Real-time QoS Requirement Constraints,” *IEEE Syst. J.*, vol. 11, no. 4, pp. 2960–2970, Jun. 2015, doi: 10.1109/jsyst.2015.2427338.

[26] H. A. Hassan, S. A. Mohamed, and W. M. Sheta, “Scalability and communication performance of HPC on Azure Cloud,” *Egypt. Informatics J.*, vol. 17, no. 2, pp. 175–182, Jul. 2016, doi: 10.1016/j.eij.2015.11.001.

[27] H. Flores and S. N. Srirama, “Mobile cloud middleware,” *J. Syst. Softw.*, vol. 92, no. 1, pp. 82–94, 2014, doi: 10.1016/j.jss.2013.09.012.

[28] K. Kumar, J. Liu, Y. H. Lu, and B. Bhargava, “A survey of computation offloading for mobile systems,” *Mob. Networks Appl.*, vol. 18, no. 1, pp. 129–140, Apr. 2013, doi: 10.1007/s11036-012-0368-0.

[29] C. You, K. Huang, H. Chae, and B. H. Kim, “Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading,” in *IEEE Transactions on Wireless Communications*, 2017, vol. 16, no. 3, pp. 1397–1411, doi: 10.1109/TWC.2016.2633522.

[30] H. F. Sheikh, H. Tan, I. Ahmad, S. Ranka, and P. Bv, “Energy- and performance-aware scheduling of tasks on parallel and distributed systems,” *ACM J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 4, pp. 1–37, Oct. 2012, doi: 10.1145/2367736.2367743.

[31] H. F. Sheikh, I. Ahmad, and D. Fan, “An Evolutionary Technique for Performance-Energy-Temperature Optimized Scheduling of Parallel Tasks on Multi-Core Processors,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 3, pp. 668–681, Mar. 2016, doi: 10.1109/TPDS.2015.2421352.

[32] R. Wolski, S. Gurun, C. Krintz, and D. Nurmi, “Using bandwidth data to make computation offloading decisions,” in *IPDPS Miami 2008 - Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium, Program and CD-ROM*, 2008, doi: 10.1109/IPDPS.2008.4536215.

[33] I. Kelényi and J. K. Nurminen, “CloudTorrent - Energy-efficient BitTorrent content sharing for mobile devices via cloud services,” in *2010 7th IEEE Consumer Communications and Networking Conference, CCNC 2010*, 2010, doi: 10.1109/CCNC.2010.5421712.

[34] R. S. Ahmad Ishfaq, “Handbook of Energy-Aware and Green Computing - Two Volume Set - 1st Ed,” 2012. [Online]. Available: https://www.routledge.com/Handbook-of-Energy-Aware-and-Green-Computing---Two-Volume-Set/Ahmad-Ranka/p/book/9781138198715. [Accessed: 13-Feb-2021].

[35] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, “Cuckoo: A computation offloading framework for smartphones,” in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 2012, vol. 76 LNICST, pp. 59–79, doi: 10.1007/978-3-642-29336-8\_4.

[36] M. Kaur, G. Kaur, P. K. Sharma, A. Jolfaei, and D. Singh, “Binary cuckoo search metaheuristic-based supercomputing framework for human behavior analysis in smart home,” *J. Supercomput.*, vol. 76, no. 4, pp. 2479–2502, Apr. 2020, doi: 10.1007/s11227-019-02998-0.

[37] K. Wang, X. Wang, X. Liu, and A. Jolfaei, “Task Offloading Strategy Based on Reinforcement Learning Computing in Edge Computing Architecture of Internet of Vehicles,” *IEEE Access*, vol. 8, pp. 173779–173789, Sep. 2020, doi: 10.1109/access.2020.3023939.

[38] M. Abbasi, M. Yaghoobikia, M. Rafiee, A. Jolfaei, and M. R. Khosravi, “Energy-efficient workload allocation in fog-cloud based services of intelligent transportation systems using a learning classifier system,” *IET Intell. Transp. Syst.*, vol. 14, no. 11, pp. 1484–1490, Nov. 2020, doi: 10.1049/iet-its.2019.0783.

[39] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, “Energy-Aware Marine Predators Algorithm for Task Scheduling in IoT-based Fog Computing Applications,” *IEEE Trans. Ind. Informatics*, pp. 1–1, Jun. 2020, doi: 10.1109/tii.2020.3001067.

[40] D. Kovachev and R. Klamma, “Framework for Computation Offloading in Mobile Cloud Computing,” *Int. J. Interact. Multimed. Artif. Intell.*, vol. 1, no. 7, p. 6, 2012, doi: 10.9781/ijimai.2012.171.

[41] M. Altamimi, R. Palit, K. Naik, and A. Nayak, “Energy-as-a-Service (EaaS): On the efficacy of multimedia cloud computing to save smartphone energy,” in *Proceedings - 2012 IEEE 5th International Conference on Cloud Computing, CLOUD 2012*, 2012, pp. 764–771, doi: 10.1109/CLOUD.2012.72.

[42] T. Justino and R. Buyya, “Outsourcing resource-intensive tasks from mobile apps to clouds: Android and aneka integration,” in *2014 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2014*, 2015, doi: 10.1109/CCEM.2014.7015494.

[43] H. Qian and D. Andresen, “Reducing mobile device energy consumption with computation offloading,” in *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2015 - Proceedings*, 2015, doi: 10.1109/SNPD.2015.7176219.

[44] M. Shiraz and A. Gani, “A lightweight active service migration framework for computational offloading in mobile cloud computing,” *J. Supercomput.*, vol. 68, no. 2, pp. 978–995, Jan. 2014, doi: 10.1007/s11227-013-1076-7.

[45] F. Qian, A. Gerber, Z. M. Mao, S. Sen, O. Spatscheck, and W. Willinger, “TCP revisited: A fresh look at TCP in the wild,” in *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, 2009, pp. 76–89, doi: 10.1145/1644893.1644903.

[46] P. Shu *et al.*, “ETime: Energy-efficient transmission between cloud and mobile devices,” in *Proceedings - IEEE INFOCOM*, 2013, pp. 195–199, doi: 10.1109/INFCOM.2013.6566762.

[47] D. Fesehaye, Y. Gao, K. Nahrstedt, and G. Wang, “Impact of cloudlets on interactive mobile cloud applications,” in *Proceedings of the 2012 IEEE 16th International Enterprise Distributed Object Computing Conference, EDOC 2012*, 2012, pp. 123–132, doi: 10.1109/EDOC.2012.23.

[48] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, “The case for VM-based cloudlets in mobile computing,” *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct. 2009, doi: 10.1109/MPRV.2009.82.

[49] “High Performance Browser Networking [Book].” [Online]. Available: https://www.oreilly.com/library/view/high-performance-browser/9781449344757/. [Accessed: 18-Aug-2020].

[50] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, “Profiling resource usage for mobile applications: A cross-layer approach,” in *MobiSys’11 - Compilation Proceedings of the 9th International Conference on Mobile Systems, Applications and Services and Co-located Workshops*, 2011, pp. 321–334, doi: 10.1145/1999995.2000026.

[51] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, “Profiling resource usage for mobile applications: A cross-layer approach,” in *MobiSys’11 - Compilation Proceedings of the 9th International Conference on Mobile Systems, Applications and Services and Co-located Workshops*, 2011, pp. 321–334, doi: 10.1145/1999995.2000026.

[52] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, “A close examination of performance and power characteristics of 4G LTE networks,” in *MobiSys’12 - Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, 2012, pp. 225–238, doi: 10.1145/2307636.2307658.