Dissertation-II Report

Submitted in partial fulfilment of the requirements for the Degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

RUCHITA DUGGAL (11613905)

Under the guidance of

DR. Ramandeep Singh

Professor, L.P.U

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

December 2017

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering

Program : P172::M.Tech. (Computer Science and Engineering) [Full Time]

COURSE CODE : CSE548 REGULAR/BACKLOG : Regular GROUP NUMBER : CSERGD0352

Supervisor Name : Dr. Ramandeep Singh UID : 14105 Designation : Associate Professor

Qualification : ____________________ Research Experience : ____________________

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Ruchita Duggal	11613905	2016	K1637	7696563795

SPECIALIZATION AREA : Database Systems Supervisor Signature: ________________

PROPOSED TOPIC : Adaptive Technique for Making Smart Computational Offloading Decisions Using Network Traces

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	Parameter	Rating (out of 10)
1	Project Novelty: Potential of the project to create new knowledge	7.25
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	7.00
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.25
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.75
5	Social Applicability: Project work intends to solve a practical problem.	7.25
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.00

PAC Committee Members		
PAC Member 1 Name: Kewal Krishan	UID: 11179	Recommended (Y/N): Yes
PAC Member 2 Name: Raj Karan Singh	UID: 14307	Recommended (Y/N): NA
PAC Member 3 Name: Sawal Tandon	UID: 14770	Recommended (Y/N): NA
PAC Member 4 Name: Dr. Pooja Gupta	UID: 19580	Recommended (Y/N): Yes
PAC Member 5 Name: Kamlesh Lakhwani	UID: 20980	Recommended (Y/N): Yes
PAC Member 6 Name: Dr.Priyanka Chawla	UID: 22046	Recommended (Y/N): Yes
DAA Nominee Name: Kuldeep Kumar Kushwaha	UID: 17118	Recommended (Y/N): NA

Final Topic Approved by PAC: Adaptive Technique for Making Smart Computational Offloading Decisions Using Network Traces

Overall Remarks: Approved

PAC CHAIRPERSON Name: 11024::Amandeep Nagpal Approval Date: 04 Nov 2017

CERTIFICATE

This is to certify that Ruchita Duggal bearing Registration no. 11613905 have completed objective formulation/Base Paper implementation of the thesis titled, **"Adaptive Technique for Making Smart Computational Offloading Decisions Using Network Traces"** under my guidance and supervision. To the best of my knowledge, the present work is the result of her original investigation and study. No part of thesis has ever been submitted for any other degree at any university.

Dr. Ramandeep Singh
Professor
School of Computer Science Engineering
Lovely Professional University
Phagwara, Punjab

Date:

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. I would like to thank my senior supervisor Dr. Ramandeep Singh. His attitude towards research has greatly influenced me. His significant enlightenment, guidance and encouragement on my research are invaluable. His insights and helpful suggestions have made the road to initiate this thesis that helped me to improve the quality of this report. I would like to thank all the people for their help directly and indirectly to complete our Dissertation-II report.

Place: Lovely Professional University　　　　Ruchita Duggal

Date: 25-11-2017　　　　(11613905)

DECLARATION

I, Ruchita Duggal, student of M. Tech under Department of Computer Science and Engineering of Lovely Professional University, Punjab, hereby declare that all the information furnished in this Dissertation-II report is based on my own intensive research and is genuine.

This report does not, to the best of our knowledge, contain part of my work which has been submitted for the award of my degree either of this University or any other University without proper citation.

Ruchita Duggal
11613905

ABSTRACT

Computational offloading is the process of offloading heavy computations to the surrogate servers, which lies outside their computing environment. Offloading is done for a purpose of enhancing capability of mobile processor, increasing battery life and better performance of the user equipments. This is done by partition the application program into two parts one which resides at the client device and the other one is offloaded at the surrogate server for computation, whose only result will be sent back to the client after processing.

Our main aspect to carry out research is on generalizing an adaptive technique for smart decision making in computational offloading using network traces and combination of mobile cloud computing and mobile edge computing.

LIST OF ABBREVIATIONS

MUE- MOBILE END USERS

AP'S- ACCESS POINTS

MCC- MOBILE CLOUD COMPUTING

MEC- MOBILE EDGE COMPUTING

IAAS- INFRASTRUCTURE AS A SEVICE

PAAS- PLATFORM AS A SERVICE

SAAS- SOFTWARE AS A SERVICE

CPU- CENTRAL PROCESSING UNIT

F.M.M –FLEXIBLE MIXTURE MODE

POSNa-PERVASIVE ONLINE SOCIAL NETWORK

EECOM-ENERGY EFFICIENT COMPUTATIONAL OFFLOADING MODEL

SPaC- SEMI ONLINE PARTITIONING SYSTEM

BAAR-BINARY ACCELERATION AT RUNTIME

MPI-MESSAGE PASSING INTERFACE

MNO-MOBILE NETWORK OPERATOR

SIC-SUCCESIVE INTERFACE CANCELLATION

GPS-GEOGRAPHICAL POSITIONING SYSTEM

CO-GPS-CLOUD OFFLOADED GPS

VM-VIRTUAL MACHINE

ROVER-ATOOLKIT

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The spectacular evolvement in technology has made a road for the present computational and communication techniques. From abacus to mainframe computers and till modern Smartphone's we have came so far that it is unimaginable to think earlier that was it even possible to realize them into reality. Now we are in that era where the smart phones in our hands have drifted the Mobile end users (MUE's) from heavy super computers to the desktop computers and now to palmtops and wearable sensors embedded in the computing environments, the main reason of this gravity is the ease of access to such applications and their evolution as a portable supercomputers. The recent years have seen the growth of mobile subscriptions more than 20 per cent annually in the last 5 years and it is expected to reach up to 4.3 billion globally by the end of year 2017 [33]. The global mobile data traffic is also expected to increase by 7-fold between 2017-2021, which will surpass by 49 Exabyte per month by 2021 [34].On the other go with the unveiling unfold of high computational rigorous applications in mobile computing environment, such as speech recognizers, GPS navigation, wearable sensors (Sixth sense technology), autonomous robots and much more devices based on IOT have came into play, the list of benefits is endless and of vast possibilities which haven't been discovered yet. This has increased the expectations of MUE's and the demand for high computing power and smaller size mobile of devices. Many of these complex applications run on the resource constrained mobile devices which have low battery power, slow processors and limited storage abilities, lowered magnitude bandwidth the gap between the demands of these complex programs and the availability of limited resources is widening, resulting into slow functionality and lowered performances. In the past few decades as the only major source for the most of mobile terminal, battery has shown a relatively slow improvement of increasing by 5% annually as compared to the other software advancements in the because the software developers are more concerned about the physical development of advanced applications rather than the upgradation of hardware necessities. The need to make mobile devices smaller, lighter and have higher battery powers, either the energy capacity of battery needs to be increased or the computational capabilities needs to be compromised and to make these two contradictory conditions consistent to each other seems to be a difficult task. So in order to meet the ever growing demands of the users we need to find the better solutions for computing and may require data manipulation capabilities for lightweight smart devices with longer battery life.

A solution to these problems is Offloading or Cyber foraging, a term coined by M.Satyanarayanan, which defines it as a paradigm of mobile cloud computing (MCC) distributed environment or a component of pervasive computing environment. Cyber foraging means “living off the land” an idea of augmenting the wireless mobile computing resources with the wired hardware infrastructure as it will utilize its idle resources. We can also call the wired resourceful infrastructure as a surrogate for its interim collaboration. Whenever a mobile computer enters a locale neighborhood, i.e to an offsite location outside its computing environment it looks up for a resourceful computer(surrogate), after the detection of the surrogates which do have potential in terms of available resources it tries to negotiate with that server for the usage of its resources. The communication between the mobile computer and its surrogate is through short ambience P2P technology where the surrogate server act as a gateway channels for the mobile computer network. Computational offloading, surrogate computing or cyber foraging are the terms interchangeably used for the offloading process. Computational offloading is different from the grid computing, as in grid computing migration of the process to another server is done for load balancing. It is also far different from tradition architecture of client-server, in which the migration of computation always takes place from the thin client side. The main difference between the offloading and all other methodologies are that it migrates its computational programs to more resourceful servers, outside its computing environment whereas in grid computing the migration is from one computer to another within the same environment of grid. Offloading is more similar to efforts like SETI@home in its principles, the main difference lies between the two is that the former is used to augment capabilities of a resource constrained device for a single user where as the latter is used for the commercial purpose, i.e large scale distribution of a of computational efforts where the users involved are thousands in count. When a comprehensive computations acquires a large amount of data to be accomplished, the mobile device shifts its computation to surrogate servers, which in turn cache data from the internet to perform computations locally on its disk. There is an alternate scenario in which the surrogate has arranged the data in advance through forecasting on the behalf of the mobile devices, even when the mobile computer is yet to arrive in the surrogate neighborhood, just to avoid the internet delays as the computations requested by the mobile devices can be delay sensitive. As soon as the mobile device escalates the neighborhood of surrogates its linkages with them also gets broken and the data cached on the servers for the particular mobile device is discarded on its behalf. In the years prior to 2000 offloading was just a mere concept which was researched upon to make it feasible into reality and what its importance is in the near future for mobile users was until the limitations for low bandwidth was not solved. After that research focused upon making offloading decisions and developing algorithms to perform offloading. The advancement in virtualization techniques and with the development of cloud computing environment, increase in network quality i.e. bandwidth they all took up the computational offloading to a new level of practicality. Now we are living into the times where offloading is just not a concept, a lot of research has been performed on its infrastructural development at different level of its crudeness and decision making while offloading and more to be done in the near future [1].

There is no doubt that offloading saves energy and improves the performance of mobile systems but this too depends on essential parameters such as network bandwidth and the amount of data exchanged over the networks between server and the mobile device, server speed, available memory, load on the server etc. although there are many algorithms proposed to make offload decisions either for performance enhancement or for the purpose of saving the energy, also there have been many proposed algorithms to perform proper partitioning of offloading data to be programmed, forecasting algorithms to predict the variations in application behavior and the executing environment outside the mobile systems.

Cyber foraging have opened up many research questions, for example:

- How can we discover the presence of a surrogate server in the environment?
- How can one establish trust level within surrogates, when it finds one?
- How does load balanced on surrogates?
- By how much earlier (in time) does a surrogate server needs to be notified for an effective staging that provides minimum execution delay?
- What are inferences for scalability? Any fixed infrastructure requirements for avoiding overloading during peak demand?

- Any particular support system needed to be provided by mobile user for surrogate servers for smooth working and less disturbance? Infrastructure requirements of that support system?

1.1.Classification of approaches for making offloading decisions:

These offloading approaches are based on various parameters and are classified as follows:

1. Why there is need to offload: do we want to save energy or we want to enhance the performance?
2. When to decide offloading: either during development of program or during its execution? (generally called as static and dynamic decision making)
3. Which kind of mobile systems will use this technology of offloading: laptops, Smartphone's, sensors etc?
4. What kind of applications require offloading: depending upon its computational capabilities (it includes which part of application needs to be offloaded and which will remain on users site) depending upon no. of users?
5. What kind of infrastructure do we require for offloading: Grid or cloud computing or both?

1.2.Background:

As there have been various technological advancements in the networking and communication field in the recent year that contributed to make offloading possible. This section will include the significant enabling technologies or mechanisms for offloading.

a. Wireless networks and mobile factors :

Until late 90's wireless connectivity was unstable and discontinuous, so was the low bandwidth as one of the main problem domain from the issues faced by the mobile systems. When the improvement in these problem domains occurred it urged for the development focus towards the mobile computing together with mobile agents.

Mobile agents are independent programs that can regulate their movement while moving from one machine to the other in the heterogeneous network. They are the one's responsible for introducing the concept of migrating computations from mobile devices. Mobile agents utilized the technologies like java, xml for designing its infrastructures as these technologies will enable it to have platform independence. Due to these features of mobile agents they are used as a major solution for accessing internet resources from portable machines and devices when there are poor network conditions, variating signal quality and different network addresses.

Different infrastructures, toolkit, agents based on the java and xml technologies are :

- TCL- a mobile agent used for creating an activity that is application independent, portable and with efficiency.
- CONCORDIA- it is a JAVA based infrastructure designed for mobile agents, with the goal of attaining flexible agent mobility, its transmission, security, persistency and ability to integrate.
- ROVER- a toolkit used for mobile data accessing- it consists of dynamic objects that are relocatable together with queued RPC's

b. Virtualization and Cloud Computing:

The concept of virtualization came into play in the year 1960 when IBM developed it to logically divide large mainframe computers into small but independent units. The concept was developed because the, mainframe computers were very large and high costs for deployment in large organization, after the development of virtualization mainframes were able to multitask and to run multiple applications at the same time. During 1990's virtualization lost its popularity due to the large scale development and cheaper cost of x86 Desktop computers, but during the last decade virtualization has emerged as a solution, as it made possible to run multiple operating systems and multiple application on the same computer, which increased the flexibility of the system. Virtualization can be attained by using virtual machines such as VMware. Virtualization provided separation and security to these virtual machines. These properties of the virtualization enabled cloud to provide computing resources as a service on lease to its cloud users based on their requirements. These services are named as

IAAS, PAAS and SAAS. Virtualization also enabled the offloading using cloud computing.

1.3 Models and Architecture framework for Offloading:

There are several model and frameworks defined till now by various researchers, out of those we are describing the two main modals used for offloading purposes:

- Mobile Cloud Computing: MCC is a diversified term as described by various researchers. According to Aepona mobile cloud computing is a distributed computing model designed for mobile applications where the storage is done on the centralized data-marts of clouds and the data processing is also migrated from smart internet devices (resource constrained) to the resourceful servers of the computational clouds. Satyanarayanan in his paper "Fundamental challenges in mobile computing", 1996 described that the pervasive nature of mobile devices have innate problems of finite battery life, restricted processing speed and slow connectivity and small memories. In order to overcome these inabilities cloud computing served as the leading model to provide an interminable lido of resources to effectively manage the resource dearth problems by providing remote computational and utility services. Google cloud, AWS etc is the main providers of these services. The MCC model consists of cloud computing and mobile computing linked together with the help of internet. The architecture of this model is as shown in the following figure.

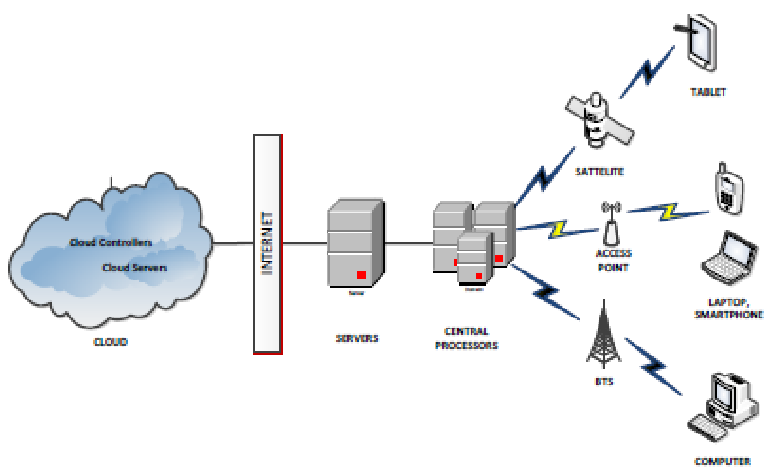

Figure 1 MCC Architecture

MCC is the expansion of mobile computing with the augmentation of cloud computing. On the extreme end of its architecture lies the mobile devices used to access the cloud services , these devices are suitable to use anywhere , as they don't require any physical link ages with the cloud or internet, they link through the Wi-Fi AP's or through base stations, which takes request from the user, pass it on to the central server, which is then exchanged with cloud servers or cloud controllers with the help of web, and processed on the cloud server site, only the result is sent in return to the mobile user instead of the whole process. The processing of heavy computational parts of the application program by shifting the load from the local resources to the cloud data centers is termed as computational offloading; this process not only reduces the load on the local resources but also reduces the battery consumption by computational intensive applications.

- **Mobile Edge Computing**: This architecture model is developed to subsist the drawback of delay as we offload the computations demanding high processing to the traditional centralized cloud via core network in order to save the battery life time and energy consumption. Execution delay consists of the time taken to offload the offloadable part of the application to the cloud in addition with the time taken by cloud to perform computation and back (i.e the delivery time in sending results) to the client device, which made the request for offloading. As these delays are problematic and inappropriate for real-time application, Mobile Edge Computing serves as a savior to resolve these delays. Although MCC have advantages like extending battery life, sophisticated application in mobile users reach, higher data storage capabilities for users but MCC have some disadvantages too, it exploits huge additional load on the radio resource and backhaul of the mobile networks which in turn instigates high round-trip delay time as data is directed to a far away servers from the mobile users in the network topology. To tackle with the problem of long latency MEC uses a concept of moving the cloud services in the vicinity of the mobile devices unlike the MCC which was a fully centralized approach accessing cloud services via internet. Due to near proximities of cloud services into the reach of mobile end user MEC provides a significantly lower latency and less fluctuation in comparison to MCC. Cloudlet was the foremost concept which brought up storage ad computations close to mobile user devices. Cloudlet uses the scenario similar to that of Wi-Fi hotspot, the key difference is that instead of using internet connectivity it brings cloud services to the mobile client.

Table 1:

Technical Aspects	MCC	Edge Computing
Deployment	Centralized	Distributed
Distance to User Equipment	High	Low
Latency	High	Low
Jitter	High	Low
Computational Power	Ample	Limited
Storage Capacity	Ample	Limited

Table 1 MCC Vs MEC

1.4 Types of offloading:

Offloading decisions based upon its types is categorized into mainly of two types:

- **Full offloading**: the main objective of performing full offloading decision is to minimize the execution delay and energy consumption at the mobile device level while satisfying a pre-defined delay constraint. It can also be used to determine the trade-off value between the two factors i.e Energy consumption and Execution delay.

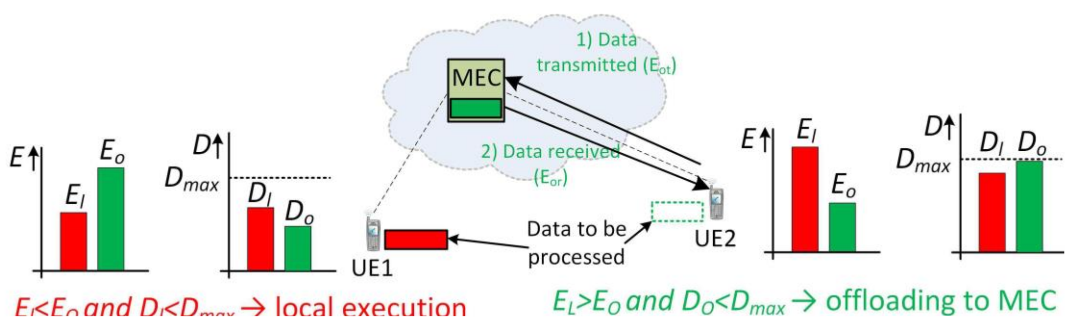

Figure 2 computational offloading in MEC on the basis of energy consumption while satisfying execution delay constraints

- **Partial offloading**: this type of offloading focuses upon the goal of minimizing the execution delay while satisfying a pre-defined execution delay constraint. And to find the tradeoff value between energy consumption and execution delay. An algorithm based on Lyapunov optimization is proposed to determine an optimized CPUs frequency for those mobile devices performing the local execution and to allocate transmission power and bandwidth to the user equipments, offloading the application to the MEC. The suggested algorithm is able to restrict the power consumption and the execution delay depending upon the chosen preferences. It also illustrates that the use of the Mobile Edge Computing for the computation offloading is able to reduce power consumption and the execution delay approximately by 90% and 98%. Respectively.

1.5 Offloading Decision making:

Offloading decision making is dependent the certain objectives such as 'whether' we need to offload computation if yes then 'what' part of computation needs to be migrated. Offloading decisions are also based upon purposes of improving performance or saving energy.

- **Improving performances**: performances can be improved if the execution time or response time is less. Another goal of improving performance is to achieve real-time constraints. This objective is required in those applications which are based upon context aware computing and real time sensing, as such application can not compromise with delay they are delay sensitive and may require real time information about user's context. Hence the mobile devices with limiting computing abilities, processing speeds can be enhanced by offloading.

  Let S_{Ue} be the speed of the mobile system

  ω= amount of computation for the 2^{nd} part

  Time to execute the second part is $\frac{\omega}{S_{Ue}}$ - (1)

  Input data = d_i

  Time taken in bandwidth B= $\frac{d_i}{B}$

  Server speed= S_s

Then, time taken to offload and execute the second part is = $\frac{d_i}{B} + \frac{\omega}{S_{Ue}}$ - (2)

Offloading will improve performance only when equation (1) will be greater than equation (2)

This implies, $\omega * (\frac{1}{S_S} - \frac{1}{S_{Ue}}) > \frac{d_i}{B}$ - (3)

The inequality equation 3 holds for :

Large ω : the program require heavy computation

Large S_s : the surrogate server is fast

Small d_i : a very small amount of data is exchanged.

Large B : the bandwidth of the network is high.

The equation 3 shows limited effects as if the server is infinitely fast i.e $S_s \rightarrow \infty$

Then , $\frac{\omega}{S_{Ue}} < \frac{d_i}{B}$, in this case offloading will not be able to improve performance.

- **Save energy**: Karthik Kumar [1] tried to answer the problem by analyzing the energy consumption during computational offloading. They proposed a formula (1) and the resulted energy may possibly be reserve during offloading process.

  PC x $\frac{C}{M}$- Pi x $\frac{C}{S}$ - Ptr x $\frac{D}{B}$ (1)

  Where, C - is the number of instruction to be offloaded.

  S & M - are the speeds instruction /second of Server and mobile device respectively.

  PC - mobile power consumption (watts),

  Pi -mobile idle power consumption (watts)

  Ptr -Mobile device's power consumption during transmission

  D -data in bytes to be exchanged.

  B -network bandwidth.

  If, the server speed considered F times faster than mobile speed then

  S = F x M (2)

  And by substituting eq. (2) in (1), the formula can be rewrite as,

  CM x (PC - $\frac{Pi}{F}$) – Ptr x $\frac{D}{B}$ (3)

In above equations Pi, Pc, Ptr, M are all constants, if the equation (3) give an positive output, then in the final outcome offloading will be able to reduce the power consumption of the mobile device. For the positive value $\frac{D}{B}$, must be very small, i.e the B needs to be very large also F needs to be very large. The quantities B,C , D are related to each other as follows:

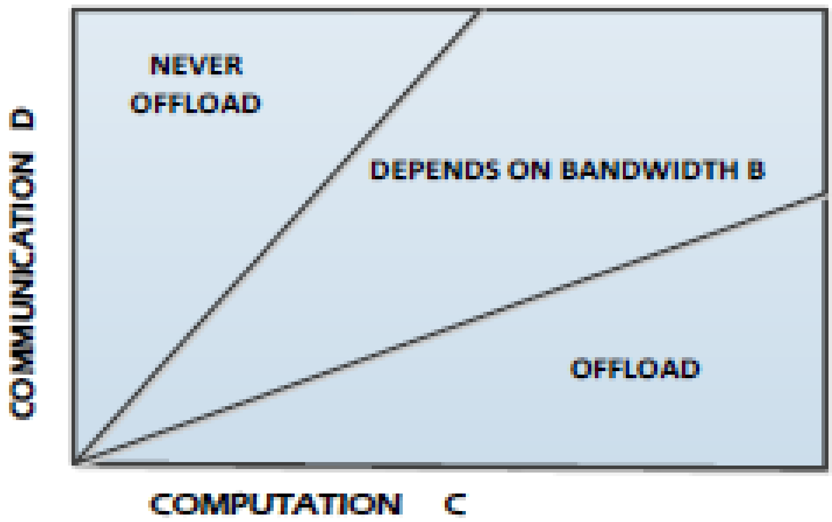

Figure 3 Offloading Decisions Depending upon B,D, C

1.6 Execution process of Offloading:

It comprises of the following steps as shown in the flow chart:

Step 1: To discover Surrogate Server – for starting computational offloading process, we need to identify the accessible surrogate servers which are static in nature.

Step 2: Context Gathering – sometimes offloading systems before offloading on a server requires the context information regarding those servers, that what kinds of resources do they have and in what amount. This information is necessary of some offloading applications for making decisions.

Step 3: Partitioning : The most crucial and important step is to partition application program or application task in order to determine that which part is transferrable i.e portable and which part is undividable, i.e un-transferrable in order to increase efficiency of offloading.

Step 4: Scheduling: according to many researchers this step is of very great importance. It collects the context information of all the surrogate servers, after gathering the data from all the available servers it determines that on which surrogate offloading our data will be more feasible in terms of location, resources available and cost of offloading.

Step 5: Remote Execution Control: it is the final step of offloading procedure. It helps in managing and establishing connection to the favorable remote server. It remains in charge, until we do not get the results for the offloaded task.

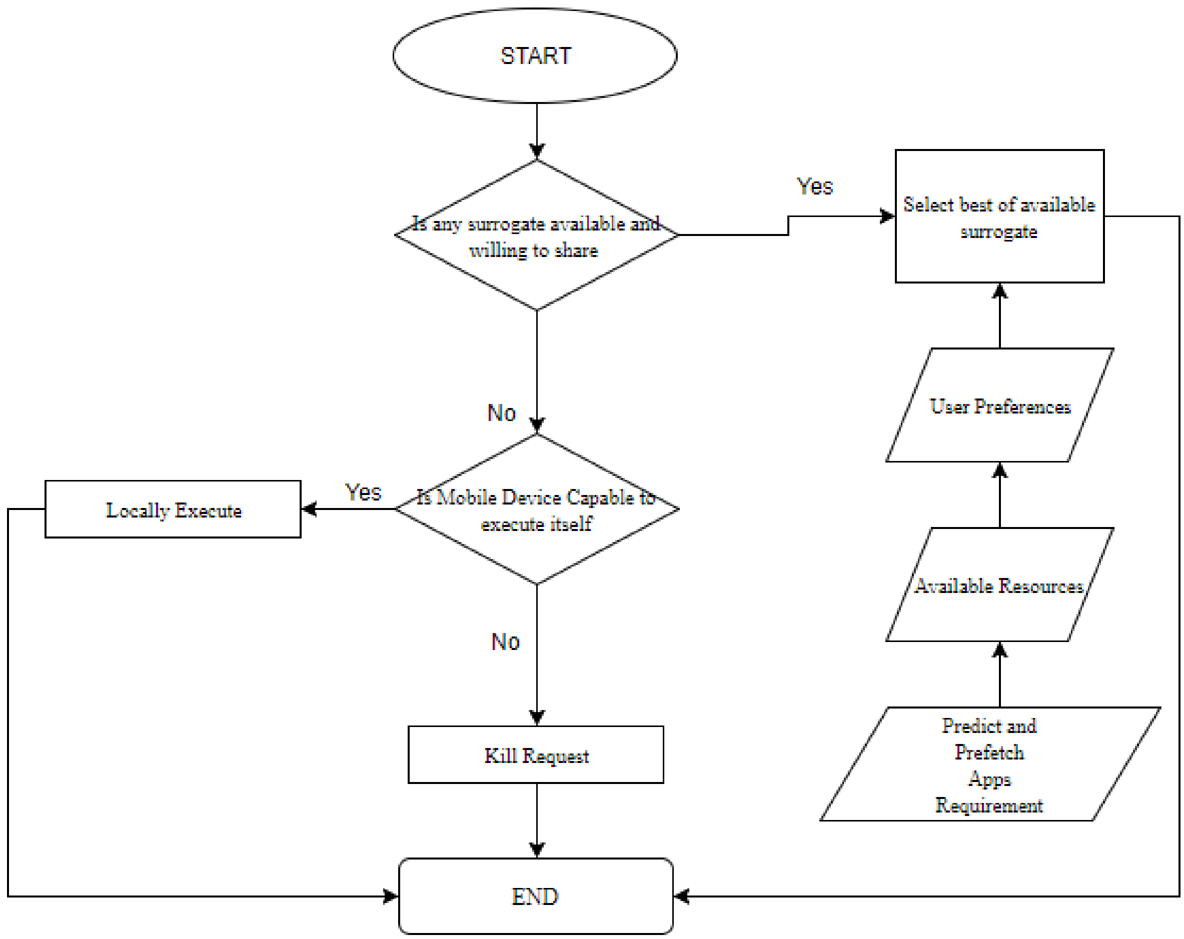

Fig 4 : Flow diagram for the offloading procedural steps.

1.7 Infrastructure Requirements for Offloading:

To make offloading practically possible we need to solve the issues addressed by various infrastructures. Some of those issues are described as follows:

- Interoperatibility : different types of devices consisting of various platforms can contact to each other via various networks. Sometime they could have one to one connection with servers or they may have many servers linked with them. So to make offloading decisions it is necessary that it could be possible between different systems operating on different networks with different computational capabilities, but it is necessary to keep this information transparent i.e hidden from the clients.
- Mobility and Fault Tolerance: we know that offloading is dependent on wireless networks and servers, so when a fault occurs like system failure, network congestion or failure. Mobility and fault tolerance will enable systems to continue executing application after all mishap.
- Privacy and Security: Privacy is a main concern as when data is offloaded to the servers it is no longer in control under users. Security is the main concern now days as third party may intrude the confidential data of the users. The solutions to these issues are encryption algorithms, such as homomorphism encryption, hardware based secured execution. Although it too have some disadvantages like encryption key may be too large which may hike the size of the data or How can we perform computations on the encrypted data
- Context Awareness: this feature allows the device to interact with user environment for gathering resource availability, state of user. This is of importance because now days the application being used in the Smartphone are requiring the geographical location or user contexts.

Paper	Decision	Contributions	Multisite-spot	Granularity	Privacy and security	Offloading strategies
[23]	Dynamic	Propose a function which embeds a disseminating framework and involves a lightweight mechanism for reducing the cost utilized energy and size of	Done	Component	No built-in Security	Application Partition

		bandwidth				
[24]	Dynamic	They provided with framework which utilized centralized monitoring technique to reduce the execution delay and to minimize the amount of energy consumption	Done	Method	No built-in Security	Entire Application Migration
[25]	Dynamic	They provided with a portable architecture consisting of a thread procedure for the implementation of runtime distributed surroundings and optimized the energy consumption and reduced the delay time	Done	Component	No built-in Security	Application Partition
[26]	Static	Designed an architecture for model generation which make automatic decisions and designed a plan for calculation of performance	MDC/Done	Task	Considering privacy Sensitivity	Entire Application Migration
[27]	Static	Proposed an algorithm for partitioning applications and also reducing the cost and using a negotiation agreement for max turn around time	Done	Component	No built-in Security	Application Partition
[28]	Static	Proposed a framework which provided individual control to the developer for assigning the sections to the servers which are offloadable in order to reduce delay in execution	Done	Task	No built-in Security	Entire Application Migration

		of code.				
[29]	Static	An architecture for automation of applications and reducing execution time and energy consumption	Done	Module	No built-in Security	Virtual Machine Migration
[30]	Dynamic	A middleware which is designed to enhance the authority to mobile client enter the cloud environment using android platform for offloading in order to save energy	Cloudlet/clone	Module	No built-in Security	Application Partition
[31]	Dynamic	An executable cloud architecture which provides isolation along with encryption facility in order to protect against intrusion from cloud administrators and for reducing the data size to be transmitted	clone	Method	Encryption and isolation	Virtual Machine Migration
[32]	Dynamic	Present an echo architecture for Smartphone applications to decide dynamically the location for offloading either locally or remotely	MDC/Cloudlet/Clone	Bundle	No built-in Security	Application Partition

Table 2: comparison of computational offloading framework

CHAPTER 2

REVIEW OF LITERATURE

Kumar K. et al. [1] Because of bounded battery life , narrow resources, network range, storage capacity and performance of processor, computational off-loading is becoming more successful and demanding these days. This paper presents the benefits of computational off-loading in modern days, one of those benefits proposed in this paper is: Use of resourceful servers for sending heavy computation data and receiving the results from intended severs. In former decades, lot many issues related to off-loading has been investigated. In presented paper of computational off-loading background, systems, techniques, working topologies and research areas has been overviewed. In conclusion survey presented in paper will allocate enormous body of research correlated with computational off-loading for mobile systems.

V. Sharma, et al [2] in this presented paper, for Pervasive Online Social Networks (P.O.S.Ns) pervasive trust management frame work is exhibited which is suited of generating high trust value between users with categorically lower cost of monitoring. To developed the system around six different properties use of Flexible Mixture Model (F.M.M) is exhibited in the given paper and then utilizes the concept of osmotic computing to perform computational off-loading which not only minimizes the number of computations but also the computational time. Intermediate state management procedure and lock door policy are used as a novel concept to allow trust visualization by providing efficient identification of trust-worthy and entrust-worthy users. The intended approach is capable of predicting user ratings efficiently with extremely low errors in the range of -+2%.

H. Gedawy, et al [3] in this paper author proposed 'Cumulus' as open source platform for edge cloud computational off-loading. Flexible and generic architecture of cumulus as well as test-bed prototype weighting a large region of heterogeneous devices, operating systems and communication methods. For allowing real-time monitoring of resources used by devices running on test-bed author uses visualization tool for enhancing cumulus. Evaluation of cumulus in action by demonstrating its performance under several offloading algorithms in different setting is intended by author. In conclusion author describes that cumulus, a computing test-bed uses hierarchical architecture allowing any worker devices to function either as part of IOT cloud, or as stand-alone device which further interact with cumulus system via a set of controller devices for edge computational off-loading.

C. Funai, et al [4] in this paper author presented and intended the performance of computational off-loading within a multi-hop cooperative network, for sharing the computational load with all other nodes in network through mobile to mobile net-working. In multi-hop cooperative network , additional iterative task assignment algorithm is presented by author that can optimize the assignment of computational task between devices, taking into account the communication over-head of multi-hop networks. Experimentally obtained results with implementation on android devices, are integrated with analytical model that can execute the evaluation of system performance under the variety of conditions. These results portraits the benefits of enabling computational off-loading in multi-hop cooperative network for all hardware and mobile devices. **Shiraz M, et al [5]** in this paper author proposed a novel method of Distributed Energy efficient Computational Off-loading Frame-work (EECOF) for the processing of intensive mobile application in MCC. The band-width focuses on Leveraging application processing services of cloud data-centers with less instances of computationally intensive component migration runtime. As a result cost of energy consumption and size of data transmission is shortened in computational off-loading for MCC. For benchmarking the prototype application in the real MCC environment author intended the proposed framework for computational off-loading. Author also described that size of data transmission over wireless network medium is reduced by 84% and 69.9% in the energy consumption cost after analysis of the results shown by employing EECOF methodology in off-loading different components.

J. P. Champati, et al [6] in this paper, author proposed solutions for problems faced in computational talk off-loading. They study problem under two scenarios, processing time on remote areas is larger than the corresponding communication time of the task to remote processor and vice-versa. Semi-online partitioning and communication (SPaC) algorithm for first scenario and SPaC- Restart algorithm for second problem. Author also intended the use of competitive ratios for both algorithms to enhance more general communication time. For analysis of results they use simulation for demonstration outperform online list scheduling of SPaC and SPaC-R.

A. Baba, et al [7] in this paper author analysis the impact of task off-loading on the efficiency of mobile face-reorganization system. Comparison of two different off-loading alternatives used in existing mobile face-reorganization system and there efficiencies in terms of energy consumption, processing time recognition accuracy had been compared in this paper. Only that method is outlined which leads to best energy performance trade-off. In this paper it is clearly mentions the overall process of face recognition from different methodology. Comparison of different face recognition methods had been carried out on Wi-Fi network based face recognition system which is experimented on Nexus 7 tablet.

A. Ferrari, et al [8] in this paper, solution is proposed to automatically extract the code needed in off-loading process from any android application. It also provides quick-fix for the extension of computational off-loading frame work, Any-run computing, to spread application code onto remote devices as needed. Author also evaluates the extra cost of off-loading process using three one of the well-known benchmarks: 1) Transparency, 2) Efficiency, and 3) Ease of utilization. In conclusion, approach to code mobility for on demand computational off-loading is presented. At the end it is described for code section to use cyclomatic complexity as a proxy for computational intensity.

M. Damschen, et al [9] in this paper author describes how binary applications can be transparently accelerated with novel heterogeneous computing resources, without requiring any manual porting or developer-provided hints. Proposed work is based on the function of Binary Acceleration At Runtime (BAAR) which is designed as client-server architecture. In this architecture client runs the desired program which allows program analysis and profiling and identifies and extracts suitable program parts to be offloaded. Author motivates the importance of a lightweight, high-performance communication between server and client and present a communication mechanism based on the Message Passing Interface (MPI). Conclusion describes that without any developer provided hints, a speedup of 3× is achievable when comparing the execution of the Jacobi 2D stencil from Poly-bench with BAAR to the execution of native code generated with the Intel Compilers with optimization level.

K. Kumar, et al [10] Author suggested that, cloud computing can potentially save energy for mobile users. However, not all applications are energy efficient when migrated to the cloud. Mobile cloud computing services would be significantly different from cloud services for desktops because they must offer energy savings. The services should consider the energy overhead for privacy, security, reliability, and data communication before offloading.

R. Wolski et al., [11], Author describes the use of computation offloading decisions in computational grid settings in which schedulers determine when to move parts of a computation to more capable resources to improve performance. Author works also portrait the unification of decision models by formulating the problem as a statistical decision problem that can either be treated "classically" or using a Bayesian approach. Using an implementation of this framework, Author simulates the efficacy of a number of different decision strategies. Outcomes indicate that a Bayesian approach employing the automatic change-point detection when estimating the *prior* distribution is the best-performing approach.

M. Barbera, et al. [12], Author evaluate and describes the feasibility of both mobile computation off-loading and mobile software/data backups in real-life scenarios. In paper there is a presentation of architecture where each real device is associated to a software clone on the cloud. There is considered two types of clones: The off-clone, whose purpose is to support computation offloading, and the back-clone, which comes to use when restore of user's data and apps is needed and intended the evaluation of the feasibility and costs of both off-clones and back-clones in terms of bandwidth and energy consumption on the real device. The smart-phones have been used as the primary mobile by the

Participants for the whole experiment duration.

Xin Kang et al. [13] in this paper, they have formulated the offloading problem with an MNO's point of view, who needs to maximize the utility if they are accessing wifi access points, on a usage-based cost estimation model. They have used SIC (successive interference cancellation) decoder for maximizing the utility of cellular service operator. The availability of the SIC is determined upon the three scenarios. 1) SIC is available at both the locations i.e on cellular base site and on third-party (Wi-Fi AP's). 2) SIC is neither available on cellular site nor on the third party location 3) SIC is available at the only cellular site.

They have obtained the close optimal value for maximizing the utility in each scenario separately, and by considering the no. of users to be large. After obtaining the values in the three scenarios, these optimal values are then compared to show that SIC decoder helps in maximizing the utility function. In order to reduce the load on the cellular network stations they have proposed a scheme in which data will be offloaded in a distributed manner and for this distributed offloading a threshold value is considered. This threshold value is chosen in such a way that it remains in accordance with the capacity of the cellular network station.

Insook Kim et al. [14] mobile data offloading is used to palliate congestion and to make better use of resources in the available network. When a cellular company (/MNO) wants to offload its congestion/traffic over the available resources such as WiFi access points(AP's) there occurs a problem that how much volume of data needs to be offloaded as per the resource AP's are available to us. In the given paper, MNO have their own utility function in the cob Douglas form i.e it will have two input functions for which one optimal output will be generated, so to maximize this utility .we use a convex optimization technique and KKT conditions which will give us the midpoint value for a certain range of given intervals, the value so obtained will be optimal value which tells us that how much of the volume of traffic can be offloaded onto AP's.
This value is in optimized form and is a convex function of the utility parameter of MNO's. This utility function will provide us with a solution that how we can place the AP's environment i.e where it will be operating based on its performance and other factors. This paper has helped us by proposing a theory that how we can maximize the value of utility parameter.

Liangfei Qiu et al. [15] the given paper has provided us with a smart way of choosing Wi-Fi hotspots to reduce its network congestion i.e to offload its traffic over these resources. As in today's world, we are dealing with unstructured data such as videos, audios over the social networking platforms. This has resulted in the explosion of traffic in mobile data, which in turn has given birth to a new problem called as network congestion. In order to reduce the costs of solving the problems(such as they have requested for a service but it isn't fulfilled) faced by the customers due to this congestion. we need to control this congestion as the demand for resources and services are high but the resources are either unavailable or underutilized. This paper has given a solution to the cellular service providers, who are willing to offload their traffic to wireless access providers, by paying them for their services in such a way that they can maintain their trust over its customers by serving their demands as well as reduce their congestion costs when their capacity is overloaded. In this paper a mechanism is used which will compute an optimal solution by densely merging the two i.e Economics and Computational methodology, which simulates an environment to determine which Wi-Fi AP will meet the requirements of the CSP in terms of cost, i.e for what amount of service what they are likely to pay.

Meng-Hsi chen et al [16] in this paper they have proposed an organised model with contemporary offloading computation by using a semi definite layout and a pioneering haphazard mapping strategies. The suggested model comprises of a portable computing circumstances in which there are many liberal plans/ schedules and one computing access point, alongside an offsite server. The given access point can either operate the accessed plans from the client or it can offload them to the cloud model. This model inflates the offloading choices of customers by restricting debased aggregation cost of dynamism, calculation and delay to ideal offloading of the schedules to the cloud by the customer. At that point a concern is brought up as an un-fashioned quadratically driven quadratic program which lies in NP-hard cliché in the long run. The issue is resolved with the help of planned agreement. The diversion consequences of the proposed classification signifies that it strengthen the implementation with just adequate number of randomization prominence and when Computing access point and a remote server is integrated together it comprises of the useful components in the traditional mobile computing environmental architecture and enhances computing implementation.

Feng Xia et al [17], In this paper they have framed a Phone2Cloud, which is a computational offloading based system which offloads computations of a programs regulating on Smartphone to the application cloud. The main objectives of this system are to augment the performance of the application program by reducing its execution time and to boost its energy efficiency. The system so developed has resulted into refining user’s experience. They have implemented the prototype framework of the application on android and Hadoop environment. They have regulated the experiments in two posit which consists of the application experiments and the scenario experiments for the system evaluation. The results so obtained during experiments shown that the outcome was upto the initial expectations it reduced the execution time and saved the energy for smartphone. And this also helped the user to make offloading decisions. For simplicity of prediction for execution time during simulation they have taken only CPU workload and input size into considerations.

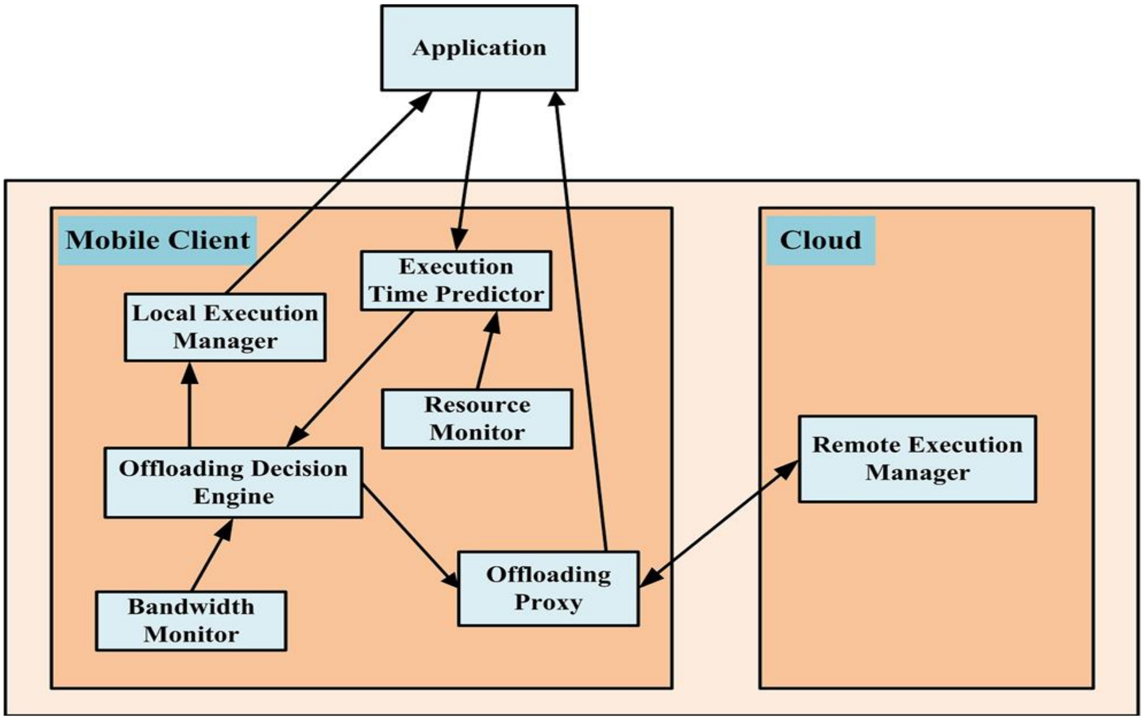

Fig 5: Phone2Cloud Architecture

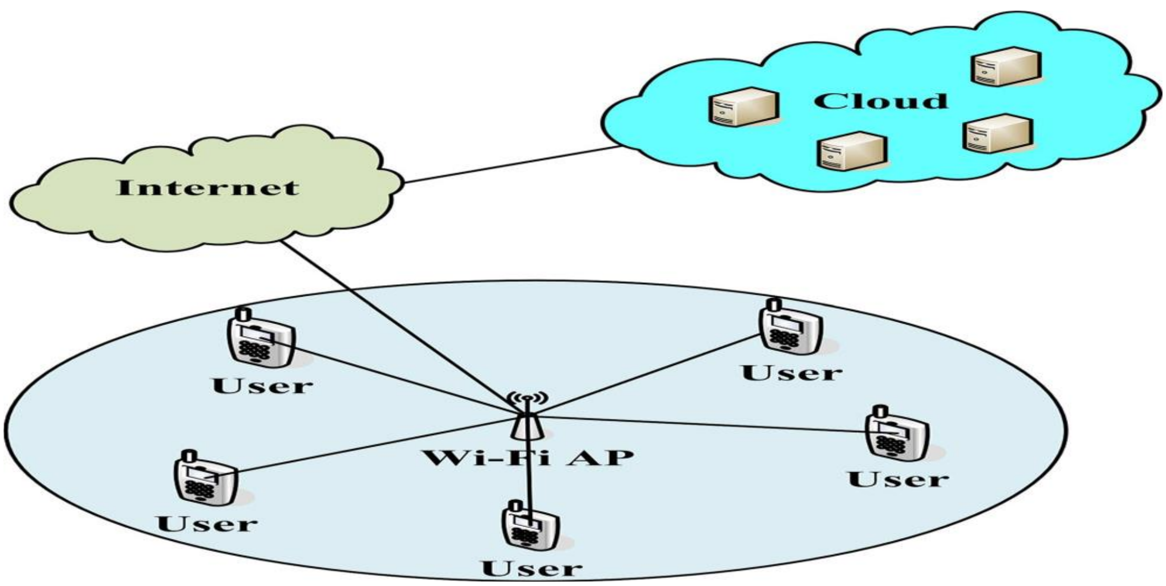

Fig 6: Environment used in application Experiment and Scenario Experiment

Feng Xia et al [17] Zhefeng Jiang In this paper cloud offloading is a promising methodology for the improvement for calculation and vitality protection. It proposes a Lyapunov improvement based plan for planning in cloud offloading , booking for cloud execution yield is downloaded for numerous applications running on a cell phone with a multi center CPU. Online calculation is inferred and execution limits are demonstrated to the ongoing usage for viable situations and it doesn't require stationary circulation of uses and any system conditions. trade-off between the normal power utilization and normal line length which showed the deferral is done. The follow driven reproduction consequences of proposed calculation approves its execution at long last.

CHAPTER 3

PROBLEM DEFINITION

In current scenario's MNO's are dependent on various issues which lead them to take decisions like when to offload, how to off-load (criteria) and is offloading feasible to them in terms of payoff?

To make these decisions they are dependent on Wi-Fi AP's or they have augmented with third party Wi-Fi AP's on contractual basis in terms of payoff as per the services used, or to determine the AP's as per their strength of resources i.e efficiency to offload congestion of traffic over the mobile networks over Wi-Fi bandwidth.

These solutions have been very useful in alleviating the problems like reducing traffic explosion, reducing costs of solving the problems of undeserved customers, maximizing the utility parameters for finding efficient AP's to offload, maximizing the usage of resources which are either less in number or are underutilized.

Till now this has helped MNO's in taking offloading decisions (with the help of Wi-Fi -AP) for their benefits. As our devices are resource constrained and the short duration battery life is the biggest challenge to smart devices till now. Many algorithms have been proposed which tried to resolve this issue by offloading data to surrogates and maximizing the efficiency and enhancing the battery life. With the advancement in technology, applications with more complex computations are embedded which is still not in consistency with slow CPU processors of smart phones.

Now we will make use of the algorithms with which they took decisions for offloading their traffic to reduce backhaul of the networks. We will make use of GPS i.e location sensing technology, where our smart device in mobile environment will be augmented with an interface which will have context awareness of the resources available with mobile user as well as it will sense the surrogates in its routes during mobility and will have context awareness about their location, network range, security, load on the server as well as equipment of resources. We will build a smart system which will make use of Adaptive Technique for Making Smart Computational Offloading Decisions Using Network Traces.

CHAPTER 4

SCOPE OF THE STUDY

As we have already used techniques for computational offloading for enhancing battery life by reducing load over the battery, in which we are proposed with certain algorithms and mobile agents. The research we are going to peruse will not only take care of the limited resources but it will also make use of the Network traces, which we will collect using various tools and applications available in our smart devices for the same route and static location. The augmented interface with the smart device will be smart enough to make choice of offloading based on the following self estimation and will pre-stage the surrogates for advance computational processing using raw signals from GPS or location traces, stored in the database and this is the main feature our smart device will be augmented with because the GPS based application or GPS itself is the main factor in utilizing battery efficiency and reducing its battery life.

It will assess its own decision on offloading based on the following factors:

When offloading decision need to be made:

- How much battery life I am with at particular instant of time?
- Is there any free Wi-Fi available in my surrounding environment with favorable resources, as earlier our computations were offloaded by our network operators i.e Base station itself, now our smart system will evaluate that will third party Wi-Fi AP's will serve better or the base station will work in cost-effective manner. (very few earlier researches have been done to minimize cost in computational offloading)
- Am I available with much resource?

Now that we have decided to offload, then a problem arises:

- For how much long I will stay in the range of particular network?
- Is the mobile network providing me with great speed for offloading or Wi-Fi (i.e depending upon the strength of their respective signals)
- Is offloading to be done is for real-time application or it can bear delay ?
- Which offloading will suit best for me full or partial ?
- Which offloading type I will perform static or dynamic ?

- Is it feasible to resume offload depending upon my battery efficiency so as to locate a Wi-Fi hotspot.
- Will our smart system offload data to cloud or on local base stations in its surrounding surrogate environment?

After evaluating all the results it will reach up to one solution which will reduce energy utilization, maximize battery life and reduce network cost for offloading using a convex utility function through using KKT conditions as done in traffic offloading we will obtain a maximized utility function which will serve our purpose.

CHAPTER 5
OBJECTIVES OF THE STUDY

The main objectives of this research study are as follows:

- To detect the strength between Mobile Network Operators (MNOs) and WiFi Access Points (APs) for making the decision for automatic offloading.
- To reduce the number of WiFi APs from the large pool of accessible points using database prepared by MNOs using routing algorithm.
- To determine a maximum value of utility function generated by routing algorithm that will be used by augmented system to make smart offloading decision.

CHAPTER 6

PROPOSED RESEARCH METHODOLOGY

Localization being the fundamental service in mobility, helps us I tracking sensing through various smart application. It has became the most essential feature of smart application that they need geographic location of the users examples of such applications are camera, whats-app, facebook etc. Most or all applications depends upon GPS as a common location sensor, very few applications have their own location sensors. GPS being globally available and cheaper in cost has a disadvantage that this application is highly processing intensive and more energy consuming. Even we are not using this application on our Smartphone's still it is processing in the background consuming energy from battery. Researchers have found out that there are few main reasons behind its high energy consumption that is

1) Ephemeris- which gives information of time and satellite trajectory details. And the GPS needs to be on for at least 30 seconds to receive full data packets.
2) Amount of Signal processing required to acquire and track satellites is significant due to weak signal strength and unknown Doppler frequency shifts.
3) Post processing and least Square calculation requires resourceful and powerful CPU's.
4) Satellites moves at higher speed so if we switch off the GPS for few minutes then the previous codes and Doppler information becomes un-useful and GPS need more energy to reacquire the satellites.

In this research we will break GPS location sensing into two parts (off-loadable part) on cloud and (un–off-loadable) part on device. For sustaining this problem we can reference to sensor node known as CLEON in CO-GPS(cloud offloaded GPS) as described in [21].

Existing solutions for different type of offloading issues:

- Specialized languages
- Frameworks and middle-wares
- Distributed VM's
- Pervasive and Ubiquitous computing solutions
- Native MCC-solutions(non VM's based)
- Native MCC- solutions(VM based)

Design Decisions based upon architecture, partitioning and offloading itself:

- Centralized offloading
- Opportunistic offloading

- General Aspects
- Partitioning or pre-phase to offloading
- Intermittent connectivity and handling of a global state. [22]

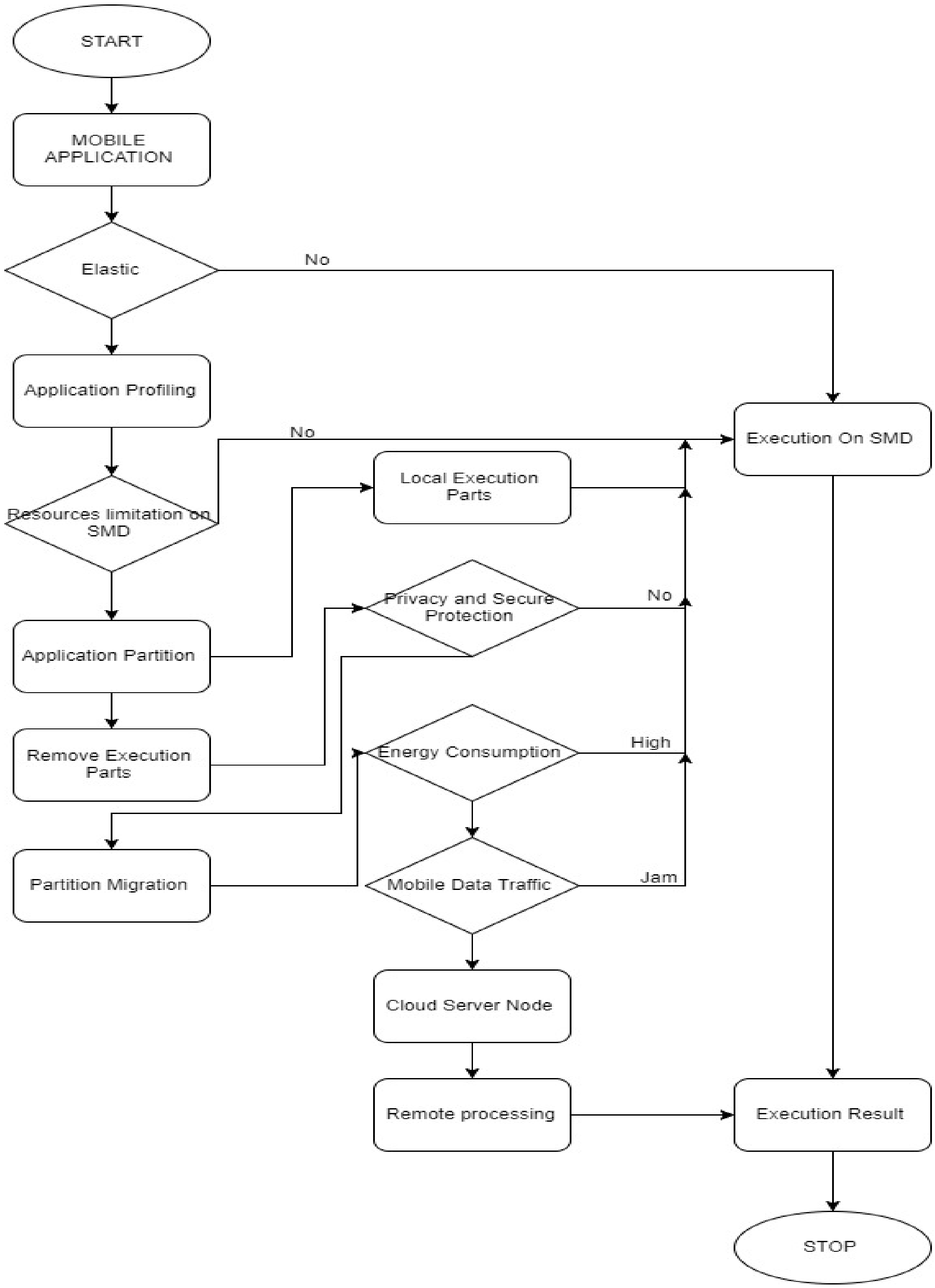

Fig 7: Process of computational offloading Decision in Mobile Cloud Computing

CHAPTER 7

SUMMARY AND CONCLUSION

In the given report we have summarized that what is computational offloading, how it is different from client server architecture and Grid Computing, then we summarized types of offloading, what are the factors that affect offloading decisions. What kind of architecture do we need as an enabling technology for offloading to be possible? We then compared different researches done in past decade having offloading as a keyword in their research in our literature review. After going through all the aspects of offloading, its advantages and disadvantages we have proposed a plan in which we will create a system architecture that will be smart enough to make offloading decision, in order to optimize the cost of constraint resources and minimize the battery consumption by augmenting it with an mobile edges and cloud environment.

REFERENCES

[1] K. Kumar; J. Liu; Y. Lu; B. Bharghava " A Survey of Computational Off-loading for Mobile Systems" in Journal Mobile Network and Applications archive Volume 18 Issue 1, February 2013 Pages 129-140.

[2] **V. Sharma; I. You; R. Kumar; P. Kim, "Computational offloading for efficient trust management in pervasive online social networks using osmotic computing," in *IEEEAccess* ,vol.PP,no.99,pp.1-1doi: 10.1109/ACCESS.2017.2683159.**

[3] **H. Gedawy, S. Tariq, A. Mtibaa and K. Harras, "Cumulus: A distributed and flexible computing testbed for edge cloud computational offloading," *2016 Cloudification of the Internet of Things (CIoT)*, Paris, 2016, pp. 1-6.**

[4] **C. Funai, C. Tapparello and W. Heinzelman, "Mobile to Mobile Computational Offloading in Multi-Hop Cooperative Networks," *2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, 2016, pp. 1-7.**

[5] **Shiraz M, Gani A, Shamim A, Khan S, Ahmad RW (2015) Energy Efficient Computational Offloading Framework for Mobile Cloud Computing. Journal of Grid Computing 13 (1):1-18.**

[6] **J. P. Champati and B. Liang, "Semi-Online Algorithms for Computational Task Offloading with Communication Delay," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1189-1201, April 1 2017.**

[7] **A. Baba, N. Sumi and V. G. Moshnyaga, "Impact of computation offloading on efficiency of wireless face recognition," *2014 24th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, Palma de Mallorca, 2014, pp. 1-7.**

[8] **A. Ferrari, D. Puccinelli and S. Giordano, "Code mobility for on-demand computational offloading," *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks*, Coimbra, 2016, pp. 1-6.**

[9] **M. Damschen, H. Riebler, G. Vaz and C. Plessl, "Transparent offloading of computational hotspots from binary code to Xeon Phi," *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, 2015, pp. 1078-1083.**

[10] **K. Kumar Y.-H. Lu "Cloud computing for mobile users: Can offloading computation save energy?" Computer vol. 43 no. 4 pp. 51-56 2010.**

[11] **R. Wolski et al., "Using Bandwidth Data to Make Computation Offloading Decisions," *Proc. IEEE Int'l Symp. Parallel and Distributed Processing* (IPDPS 08), 2008, pp. 1-8.**

[12] **M. Barbera S. Kosta A. Mei J. Stefa "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing" Proc. IEEE Intl. Conf. Comput. Commun. pp. 1285-1293 Apr 2013.**

[13] X. Kang, Y. K. Chia, S. Sun and H. F. Chong, "Mobile Data Offloading Through A Third-Party WiFi Access Point: An Operator's Perspective," in *IEEE Transactions on Wireless Communications*, vol. 13, no. 10, pp. 5340-5351, Oct. 2014.

[14] Insook Kim and Dongwoo Kim, "Utility-maximizing WiFi offloading from a perspective of mobile network operators," *2015 Seventh International Conference on Ubiquitous and Future Networks*, Sapporo, 2015, pp. 746-748.

[15] L. Qiu, H. Rui and A. Whinston, "When Cellular Capacity Meets WiFi Hotspots: A Smart Auction System for Mobile Data Offloading," *2015 48th Hawaii International Conference on System Sciences*, Kauai, HI, 2015, pp. 4898-4907.

[16] M. Chen, B. Liang, and M. Dong, "A Semidefinite Relaxation Approach to Mobile Cloud Offloading with Computing Access Point," pp. 1–5

[17] F. Xia, F. Ding, J. Li, X. Kong, L. T. Yang, and J. Ma, “Phone2Cloud : Exploiting computation offloading for energy saving on smartphones in mobile cloud computing,” pp. 95–111, 2014.
[18] Z. Jiang, S. Member, S. Mao, and S. Member, “Energy Delay Tradeoff in Cloud Offloading for Multi-Core Mobile Devices,” vol. 3, 2015.
[19] H. Wu, Q. Wang, and K. Wolter, “Mobile Healthcare Systems with Multi-cloud Offloading,” 2013.
[20] B. Zhou, A. V. Dastjerdi, R. N. Calheiros, S. N. Srirama, and R. Buyya, “A Context Sensitive Offloading Scheme for Mobile Cloud Computing Service,” 2015.
[21] J. Liu *et al*., "CO-GPS: Energy Efficient GPS Sensing with Cloud Offloading," in *IEEE Transactions on Mobile Computing*, vol. 15, no. 6, pp. 1348-1361, June 1 2016. doi: 10.1109/TMC.2015.2446461
[22] Orsini, G., Bade, D., Lamersdorf, W.: Context-aware computation offloading for mobile cloud computing: requirements analysis, survey and design guideline. Procedia Comput. Sci. 56**, 10–17 (2015)**
[23] Shiraz M, Gani A, Shamim A, et al. Energy efficient computational offloading framework for mobile cloud computing[J]. Journal of Grid Computing, 2015, 13(1): 1- 18.
[24] Shiraz M, Gani A, Ahmad R W, et al. A lightweight distributed framework for computational offioading in mobile cloud computing[J]. PloS one, 2014, 9(8): eI02270.
[25] Shiraz M, Gani A. A lightweight active service migration framework for computational offloading in mobile cloud computing[J]. The Journal of Supercomputing, 2014, 68(2): 978-995.
[26] Folino G, Pisani F S. Automatic offloading of mobile applications into the cloud by means of genetic programming[J]. Applied Soft Computing, 2014, 25: 253-265.
[27] Kaushik N, Kumar J, DCRUST M. A computation offloading framework to optimize energy utilization in mobile cloud computing environment[J]. Int. Journal of Computer Applications and Information Technology, 2014, 5(2): 6 1-69.
[28] Wang M, Novel Mobile Computation Offloading Framework for Android Devices, Engineering and Applied Science Theses & Dissertations, 2014, 10(2): 16-17.
[29] Arora M, Kalra M, Singh S. ACOF: Autonomous computation offloading framework for Android using cloud[Clllnformation Management in the Knowledge Economy (IMKE), 2013 2nd International Conference on. IEEE, 2013: 143-149.
[30] Kovachev D, Yu T, Klamma R. Adaptive computation offloading from mobile devices into the c1oud[ClI120 12 IEEE 10th International 2743 Symposium on Parallel and Distributed Processing with Applications. IEEE, 20 12: 784-79 1.
[3 1] Hung S H, Shih C S, Shieh J P, et al. Executing mobile applications in the cloud: Framework and issues[J]. Computers & Mathematics with Applications, 2012, 63(2): 573-587.
[32] Kemp R, Palmer N, Kielmann T, et al. Cuckoo: a computation offloading framework for smartphones[Clllnternational Conference on Mobile Computing, Applications, and Services. Springer Berlin Heidelberg, 20 10: 59-79.
[33] ICT facts and figures. http://www.itu.int/en/ITU-D/Statistics/Pages/facts/default.aspx. Available: 2017
[34] Cisco Visual Networking Index. Global mobile data traffic forecast update, 2017-2021. Cisco white paper, 2017