ISSN:0975-9646

Overview of Offloading in Smart Mobile Devices for Mobile Cloud Computing

Roopali, Rajkumari

Dep't of IT, UIET, PU
Chandigarh, India

***Abstract*- The recent advancement in cloud computing is leading to an excessive growth of mobile devices that can become powerful means for information access and mobile applications. Thus introducing a latent technology called Mobile Cloud Computing (MCC). Smartphone devices support wide range of mobile applications which require high computational power, memory, storage and energy but these resources are limited in number so act as constraints in smartphone devices. With the integration of cloud computing and mobile applications it is possible to overcome these constraints by offloading the complex modules on cloud. This review of literature focuses on challenges in offloading such as latency rate, network bandwidth and heterogeneity which mainly depends on factors like code to be offloaded, distance between smartphone device and cloud, wireless networks and complex computations. Final output is traced back to the client (smartphone device) and thus saves the resources of the smartphone device.**

***Keywords*- Heterogeneity, Latency, Mobile Cloud Computing, Offloading, Power consumption, and Smartphone devices.**

I. INTRODUCTION

Cloud Computing delivers infrastructure, platform, and software that are provided as services on the usage-based payment model to end users. These services are Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [1]. Mobile Cloud Computing at its simplest refers to an infrastructure where both the data storage and the data processing happen outside of the mobile device. Mobile Cloud applications move the computing power and data storage away from mobile phones and into the cloud, bringing applications and mobile computing to not just Smartphone users but a much broader range of mobile subscribers [2]. It is a productive business choice that transfers data from smartphone devices to powerful and centralized computing platforms located in the cloud. Thus reducing the development and running cost of mobile applications on smartphone devices. This technique consumes fewer resources of smartphone devices making them more efficient.

The architecture of MCC is such that the various mobile devices are connected to their respective mobile networks via base station (BTS, Access Points or Satellite). The requests made by users are transferred to servers providing mobile network services. Now the mobile network operator can provide services to mobile users as authentication, authorization and accounting based on data stored in database. After authorizing user the requests are transferred to cloud via internet. "Fig 1." explains the architecture of mobile cloud computing.

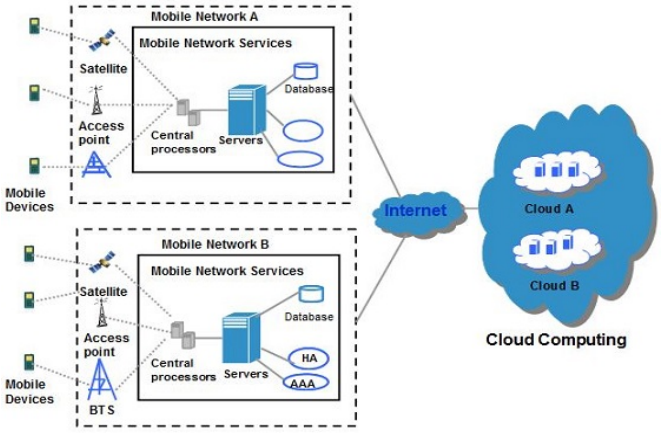

Fig1. Mobile Cloud Computing Architecture [1]

Increased number of smartphone users is due to the support smartphone devices have for diverse applications, to name few gaming, books & reference, entertainment, media & movies, social networking, e-commerce, video processing, health & fitness, photography, weather, travel, shopping, sports, news & magazine and like. Some of these applications require high processing power, large memory, battery life and high network bandwidth. For the availability of these resources to applications; smartphone devices require hardware and software changes. Hardware changes are constrained due to size and cost and may not be able to bring a change in the proper utilization of resources of the smartphone devices. As we define mobile cloud computing an integration of cloud computing technology with mobile devices resource-full in terms of computational power, memory, storage, energy, and context awareness [3]. Software changes are possible and are more effective. Mobile devices do not need to have powerful configurations because all the complex modules can be processed on the cloud.

Our vision is to augment the resources of smartphone devices by offloading. This paper is organized as follows.

Section II explains the concept of offloading. Section III shows the process of offloading with the help of a flow chart. Section IV discusses the issues in offloading. Section V proposes few solutions to overcome these issues. Section VI discusses the related work and towards the end Section VI concludes the paper.

II. CONCEPT OF OFFLOADING

Offloading is defined as a procedure that migrate resource-intensive computations from a mobile device to the resource-rich cloud, or server (called nearby infrastructure). Cloud based computation offloading enhances the application's performance, reduces battery power consumption, and executes applications that are unable to execute due to insufficient smartphone resources [4]. It means that complicated parts of an application run on the remote servers and results are communicated back to the local smartphone device.
Three measures that affect the concept of offloading are as follows:

A. Threshold value

It is the minimum value which an application shall exceed to be offloaded. The threshold value can be measured in terms of processing time, energy consumption and memory usage. If a particular application, to be offloaded, takes enough time to compute on the smart phone device, consumes much energy and battery lifetime of the client (smart phone device), requires large amount of battery for storage and surpass the minimum threshold value; it is effective to offload the application onto the cloud.

B. Size of application

Offloading decision depends upon the size of the application to be offloaded. It saves energy for a code compilation, if size of code is large. However, if the code size to be computed is small then offloading consumes more energy and battery as compared to local processing. As an example, [5] shows the Gaussian application (to solve a system of linear algebraic equations) which offloads the entire matrix into remote server. In terms of energy efficiency, the cost of offloading is higher for small matrices (e.g., smaller than 500x500 in size) while the cost can be up to 45% for large matrices.

C. Critical Section

The part of the code which involves complex computations and requires more time to execute is referred to as critical section. The module of the application if falls under the critical section then it is offloaded. Complex processing of the application are offloaded to the cloud saving much amount of battery, memory and CPU usage of the smartphone device.

III. PROCESS OF OFFLOADING

Offloading the application is necessary as applications use resources of the smartphone. Resources of the smartphone devices are scarce in quantity, so offloading proves to be useful for such devices. Offloading is a process comprising of series of steps. Fig2 explains the procedure of computation offloading with the help of a flowchart.

Foremost step is to start the application. Three questions arises like to check the threshold value, size of the application and critical section; if the value is small then the application is executed normally that is without the cloud support or on the smartphone device itself. For larger values of threshold, code size and critical section; the application is offloaded with the help of one of the following techniques

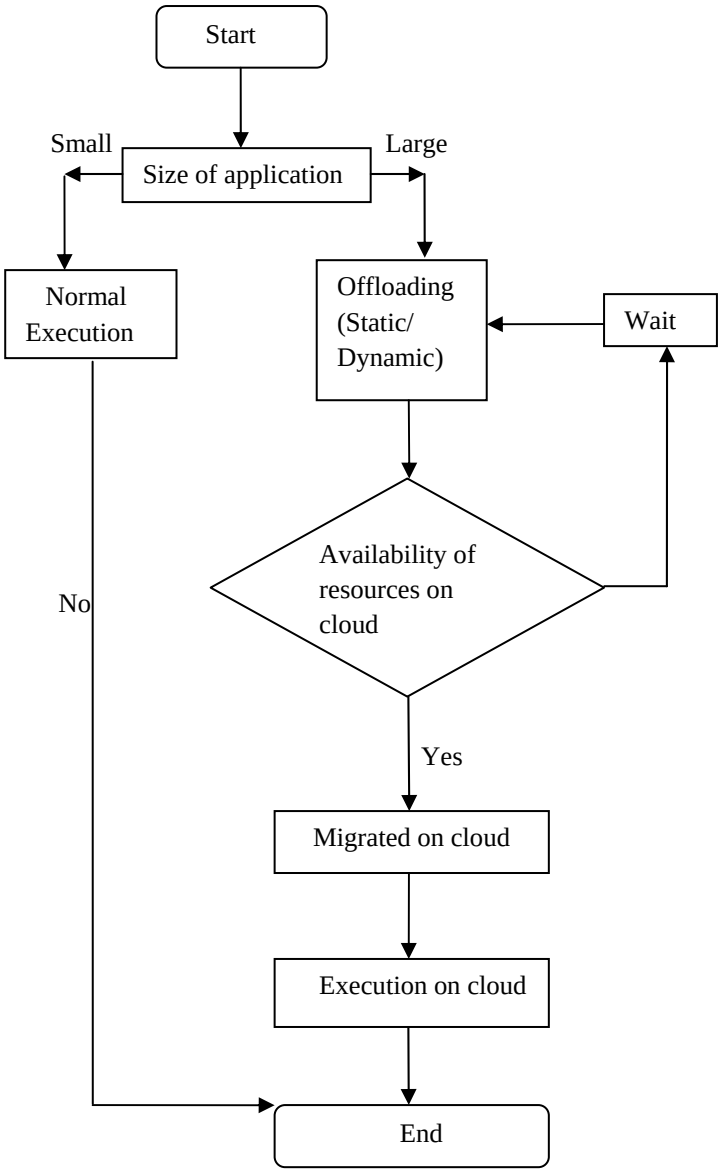

Fig2. Process of Offloading

A. Static Offloading

In static offloading application is partitioned during development. In static environment, parameters such as data size and execution time which acts as deciding factor for offloading are known beforehand. However, it is difficult to know the correct execution time before the actual execution takes place and the inaccurate data can result into inefficient offloading result.

B. Dynamic Offloading

Dynamic network environment means changing connection status and bandwidth that affect the process of

offloading. By the term dynamic offloading we mean that the modules may be transferred for execution onto cloud when the application is running.

Next step checks for the availability of resources on the cloud, if available then application is migrated. If the resources required for a particular application are consumed by some other application then the application will wait for random amount of time and will again check for the availability of resources.

After the resources are made available to the application then the application is migrated to the cloud. After the code migration, module is executed on the cloud. The application makes use of the resources of the cloud and the results are sent back to the smartphone device. This helps in reducing the battery consumption of the smartphone device.

Thus the various implications which can be drawn from the given figure are:

- There are issues of latency rate while offloading, checking the availability of resources and migrating to cloud.
- During the process of migration on cloud, code to be offloaded faces heterogeneity such as diverse mobile softwares, cloud architectures and wireless networks.
- Offloading saves battery consumption as the complex computations are performed on the cloud (server) and not on the smartphone device (client).

IV. ISSUES IN OFFLOADING

This section discusses issues in offloading. The various issues listed below affect the essential factors of smartphone device such as efficiency and energy consumption. The issues that affect the essential factors are as follows:

A. Latency rate

Latency rate is defined as the delay between the offloading of an application on the cloud and the final results to be retraced back to the smartphone device. The latency depends on number of factors, such as offloaded code size, data input size, location of required data, offloading scheme, network bandwidth, execution delay and the resultant data size [6].

Latency rate also depends upon the synchronization between the device and the cloud. If it is performed at run time then it may increase the latency rate. The devices using the cloud resources in the vicinity of the cloud are more efficient as compared to devices which are far from the cloud as they have low latency rate. The geographical distance between the mobile device user and the cloud service provider plays a vital role to measure the latency rate.

B. Network Bandwidth

Bandwidth utilization depends upon the amount of the code being offloaded. If large amount of data is to be offloaded to the cloud then it ultimately increases the latency rate which in turn affects the efficiency and energy consumption of the whole process. Wired bandwidth and wireless bandwidth are of equal concern because both are scarce with respect to number of users. There are number of solutions such as sharing of network bandwidth amongst the users of a particular geographical region which are sharing similar content. However, issues regarding distribution policy are still prevailing. 4G technology is supposed to pin down the bandwidth issues both in wired and wireless networks but network architecture is still an issue for the network bandwidth.

C. Heterogeneity

The English dictionary defines heterogeneity as non-uniform. MCC has diverse hardware, infrastructure, mobile device technologies, core cloud architectures and communicating networks thus increasing heterogeneity in MCC environment. Heterogeneity affects efficiency of the smartphone device. Various heterogeneous features that pose a challenge to MCC are discussed below:

1) *Heterogeneity in Mobile Devices:* Many existing smartphone devices are different in terms of hardware, OS, features and communication medium. Hardware heterogeneity includes different processors with different architectures (32-bit or 64-bit), different processor speeds and amount of cache memory. The software technologies used by various smartphone devices can be classified as follows:
   - Android is an open source operating system powered by Google, and its kernel is based on Linux. Android OS supports Java based applications [7].
   - iOS is a proprietary OS of Apple and is based on MAC OS X. iOS applications are mainly developed in objective C [8].
   - Symbian is an open source OS powered by Nokia, while its applications are developed in Java and C++ [9].
   - Mobile OS is a proprietary of Microsoft and support applications developed on .NET framework [10].
   - Blackberry OS is a proprietary of Research in Motion (RIM) and its applications are mainly developed in Java [11].

   The following graph explains vertical heterogeneity by rise within varied versions of a single OS and horizontal heterogeneity by rise between different OSs.

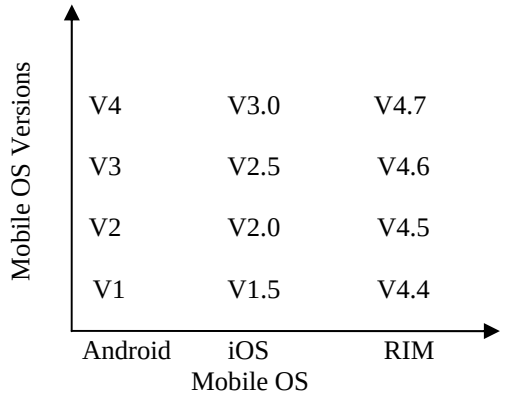

Fig3. Heterogeneity in Mobile OS showing different Mobile OS and their different versions.

2) Heterogeneity in Cloud Services: Many cloud service vendors provide end users with in-built policies, infrastructure, platform and software. Such amalgam of services poses challenges like portability and interoperability. Heterogeneity occurs among different cloud providers and also within a single cloud provider in terms of the services they provide.

The graph given below describes vertical heterogeneity by rise within varied cloud vendors and horizontal heterogeneity by rise between different services.

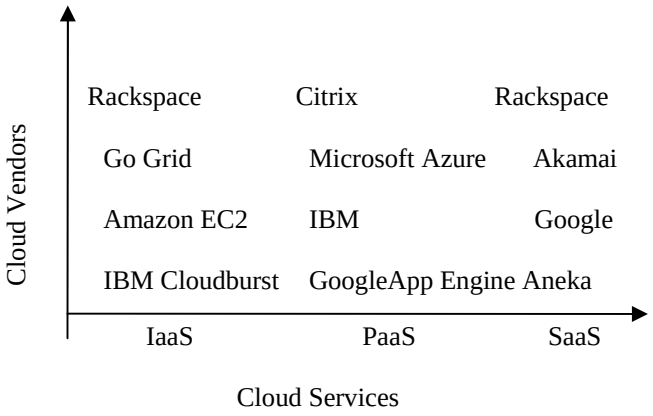

Fig4. Heterogeneity in Cloud Vendors and services provided by them.

3) Heterogeneity in Wireless Network: Most of the communication takes place with the help of wireless networks. The composition of various wireless technologies such as Wi-Fi, 3G, and WiMAX makes MCC more complicated compared to cloud computing [12]. Low bandwidth, high latency and jitter degrade the quality of communication between the mobile devices and cloud services. Handoff is a major issue when a mobile cloud user is moving between different wireless networks or within the versions of the same network. While offloading smartphone device must be compatible with wireless networks so that the code to be offloaded must reach the destination server.

The following graph depicts vertical heterogeneity by rise within a single type of wireless network and horizontal heterogeneity by rise between varied types of wireless networks.

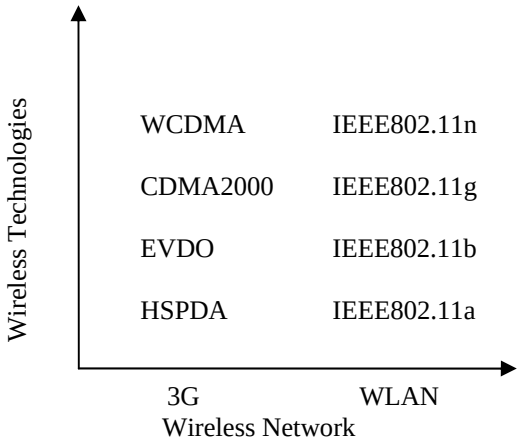

Fig4. Heterogeneity in Wireless Technologies

V. PROPOSED SOLUTION

In this section, we propose to figure out the issues with offloading. Managing the resources of the smart phone device and code of the application to be offloaded can be useful.

A. Workload Management

The workload manager tracks workload requests, allocates workload tasks among service providers, and aggregates results [13]. It checks for the availability of cloud service providers for the requested duration and also determines the optimal distribution of tasks amongst the service providers. The requests made by the mobile application users are traced by the service provider and are sent to the cloud via workload manager. Cloud resources required for a particular application, sometimes are not available due to congestion. Workload manager grants the requests made by the client to the server (cloud) which is available. Thus increasing the throughput of the application being offloaded.

B. Migration Cost

Migration cost is the total amount consumed while migration of the code of an application onto cloud. It includes the communication cost which is determined by the device interaction. By device interaction, we mean the resources of the smart phone device which are required while execution of the application. The challenge is to minimize the migration cost while executing the application on the cloud. The level of migration granularity plays a huge role in minimizing the migration cost. Various levels such as *entire process, module level, class level, bundle level, method level, thread level and weblets* [14] account for different migration cost. This

cost sum up to latency rate. The distance between the client (smart phone device) on which an application is running and the remote server (cloud) also contribute to the latency rate. The near-by devices have low execution time as compared to farther ones.

C. Partitioning Algorithm

Before offloading, partitioning of the application is mandatory as single partition does not fit into the heterogeneous environment. Partitioning divides the application into modules. Modules can be classified as complex modules which have complex computations that shall be processed on the cloud and simple modules that can be computed on the smart phone device itself. Choice of partitioning algorithm depends upon the number of factors such as configuration of modules and optimization goals (execution time, battery power consumption). It can be Static or Dynamic. Static partitioning is when the application is divided into modules during development of the application and dynamic partitioning is when the application is divided into modules at run time. The processing time of the modules are known beforehand in static partitioning but the exact value is not known that makes the results ineffective. Dynamic partitioning is a good choice because it depends on the availability of the resources and the calculations are performed on the actual processing time which produces effective results.

D. Energy Saving

Energy is elementary constraint for smartphone devices. Offloading conserves energy and battery lifetime of the smartphone device by migrating energy-intensive computations to cloud. Offloading makes it possible for the applications to run on the remote server and the output is sent back to the local device. The complex computations when performed on the limited resources; consumes enough time and energy. For example, playing games on the smartphone device consumes enough battery of the device. When playing games with the cloud support; user just interacts with the GUI of the game and all the moves of the game are performed on the cloud. Many image sharing applications are designed that capture and share images on cloud such as Dropbox, Flickr, and Instagram. Social network applications like Facebook and Twitter also share images on cloud. By using these applications users can save energy and memory of their smart phone devices to large extent. Mobile applications do not face storage space as a constraint as their data is stored on the cloud.

VI. RELATED WORK

This paper is related to process of offloading and issues concerning it. We briefly summarize previous work on offloading an application to cloud. Previously many methods have been proposed that offload an application to cloud. An exhaustive approach suggests executing an application on a more powerful hardware than the original device.

Recently, [15] proposes that the offloading cost is the measure of memory cost of the mobile device, energy cost on the mobile device and minimum execution time for the remote execution of services.

Minimum memory usage is defined as the value less than the threshold value which is equal to the available memory on the smart phone device as an application cannot occupy the whole available memory on the device.

Minimum energy consumption is the value such that the local energy consumption (not offloading) should be greater than the remote energy consumption (offloading).

Minimum execution time is the sum of local execution time and the remote execution time. The value of local execution time should be greater than the remote execution time.

Local execution time is calculated as the ratio of local CPU instructions and local CPU frequency. As [16] proposes, remote execution time is total value of execution on cloud and the total round trip delay. Round trip delay is calculated as the time required to send the input to the cloud and to send the results back to the device which are produced on the cloud.

VII. CONCLUSION AND FUTURE SCOPE

To sum up, Mobile Cloud Computing is the integration of cloud computing with smartphone devices. It provides rich communication between cloud-mobile users and cloud providers regardless of heterogeneous environments. Many applications can be accessed using MCC as smartphone devices do not support those few applications due to lack of enough hardware, software and battery lifetime. Offloading is the method to migrate the complex modules of the application to the cloud. All the complicated computations are performed on the cloud and the results are sent back to the smartphone device. This makes the smartphone device users to access all those applications which require high computational power, large memory, battery lifetime and large network bandwidth. This paper presented the challenges with offloading such as latency rate which mainly depends on factors like code to be offloaded, distance between smartphone device and the remote server, and results of computations. Network bandwidth is another issue as the wireless networks are unable to provide the required bandwidth to all the users. Heterogeneous environments have negative impact on the process of offloading but its effect can be reduced by synchronizing mobile device and the cloud. Mobile devices and server must be compatible to offload the data and perform complex computations on the cloud. These issues ultimately affect the efficiency of the process.

Further, this paper discusses proposed solutions for the issues such as workload management that assigns the tasks to available service providers so that the application is executed on time. Minimized migration cost is another solution to have a check on the latency rate. Partitioning algorithm helps in dividing the application for the complex calculations to be performed on cloud and limits energy consumption and battery usage of the smart phone device.

Finally, the future scope of the paper gives research directions pertaining to overcome the issues in offloading which are open and innovative.

REFERENCES

[1] R. N. Calheiros, R. Ranjan, A Beloglazv, C. A. F. De Rose and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", *in Software-Practice and Experience*, 2011, vol. 41, pp. 23-50.

[2] H. T. Dinh, C. Lee, D. Niyato and P. Wang, "A survey of mobile cloud computing: architecture, applications and approaches", *in Wireless Communications and Mobile Computing*, available at http://onlinelibrary.wiley.com.

[3] D. Huang, "Mobile cloud computing", *in IEEE COMSOC Multimedia Communications Technical Committee (MMTC) E-letter*, 2011, vol. 6, pp. 27-31.

[4] Amazon simple storage service. Available: http://aws.amazon.com/s3/

[5] A. Rudenko, P. Reiher, and G. H. Kuenning, "Saving portable computer battery power through remote process execution", *in Journal of ACM SIGMOBILE on Mobile Computing and Communications Review*, 1998, vol. 2,pp. 19-26.

[6] A. U. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A survey of mobile cloud computing application models", *in IEEE Communications Surveys & Tutorials*, in press, 2013.

[7] Android. Available at http://www.android.com, last accessed on 31 October, 2014

[8] Apple iOS5. Available at http://www.apple.com/iOS, last accessed on 25 October, 2014

[9] Nokia Symbian. Available at http://symbian.nokia.com, last accessed on 5 November, 2014

[10] Windows mobile. Available at http://windows.microsoft.com/en-in/windows/downloads, last accessed on 30 October, 2014

[11] Blackberry OS. Available at http://ca.blackberry.com/software/smartphones.html, last accessed on 27 October, 2014

[12] Z. Sanaei, S. Abolfazi, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: taxonomy and open challenges", *in IEEE communications Surveys & tutorials*, in press, 2013, vol. 16, pp. 369-392.

[13] H. Vishwanathan, E. K. Lee, I. Rodero, and D. Pompili, "Uncertainity-aware autonomic resource provisioning for mobile cloud computing", *in IEEE transactions on parallel and distributed systems*, 2013, vol. 99, pp. 1.

[14] 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", *in IEEE Communications, Surveys & Tutorials*, 2013, vol 15, pp. 1294-1313.

[15] D. Kovachev, T. Yu and R. Klamma, "Adaptive computation offloading from mobile devices into the cloud", *in IEEE International Symposium on Parallel and Distributed Processing with Applications*, 2012, pp. 784-791.

[16] S. Abolfazi, Z. Sanaei, M. Alizadeh, A. Gani and Feng Xia, "An experimental analysis on cloud-based mobile augmentation in mobile cloud computing", *in IEEE Transactions on Consumer Electronics*, 2014, vol. 60, pp. 146-154.